Machine Intelligence and Pattern Recognition 1

PROGRESS IN PATTERN RECOGNITION 2

Edited by
L.N. Kanal and A. Rosenfeld

North-Holland

PROGRESS IN PATTERN RECOGNITION 2

Machine Intelligence and Pattern Recognition

Volume 1

Series Editors

L. N. KANAL

and

A. ROSENFELD

University of Maryland
College Park
Maryland
U.S.A.

NORTH-HOLLAND
AMSTERDAM · NEW YORK · OXFORD

Progress in Pattern Recognition 2

Edited by

Laveen N. KANAL

and

Azriel ROSENFELD

University of Maryland
College Park
Maryland
U.S.A.

1985

NORTH-HOLLAND
AMSTERDAM · NEW YORK · OXFORD

ISBN: 0 444 87723 1

Publishers:
ELSEVIER SCIENCE PUBLISHERS B.V.
P.O. Box 1991
1000 BZ Amsterdam
The Netherlands

Sole distributors for the U.S.A. and Canada:
ELSEVIER SCIENCE PUBLISHING COMPANY, INC.
52 Vanderbilt Avenue
New York, N.Y. 10017
U.S.A.

PRINTED IN THE NETHERLANDS

PREFACE

We are pleased to present the second volume of our series Progress in Machine Intelligence and Pattern Recognition (formerly entitled "Progress in Pattern Recognition"). The present volume contains nine papers, one of them nearly of monograph length, covering a broad range of topics in pattern recognition and computer vision.

The first three papers deal with the architectural and computational aspects of image processing and computer vision. Yalamanchili et al. present a hierarchical taxonomy of image processing architectures based on the nature of the processing and communication strategies employed. At the first level of their hierarchy, image processing systems are divided into functionally dedicated architectures and programmable general purpose architectures. Each of the first level categories are then further divided into three groups: those with fixed interconnection structures, those with bus oriented structures, and those having reconfigurable interconnection structures. In the context of this taxonomy, the authors classify and analyze a wide range of image processing systems appearing in the literature. Guerra and Levialdi present a different way of looking at many of the same image processing systems discussed in the first paper. Their paper first reviews classical models of computation and then considers how image processing architectures, especially parallel architectures, can be derived from such models, and how architectures can be matched to problems. Voss et al. survey a number of interactive software systems for computer vision. Noteworthy is their mention of several European contributions which have often been neglected in past surveys. The proliferation of architectural developments and software systems for computer processing of images make these surveys and analyses especially useful.

The second trio of papers deals with three fundamental areas of vision: texture, shape, and motion. Chellappa presents a systematic and detailed

exposition of two-dimensional discrete Gaussian Markov random field models and their applications in the synthesis, compression, classification, and restoration of images. Shapiro discusses selected examples of recent results on shape decomposition, description, and matching. Jain surveys the rapidly expanding field of time-varying imagery analysis, covering a variety of key areas including such topics as computing optical flow and recovering information on the three-dimensional motion of objects. These reviews should be of considerable help in assessing the state of the art in three areas in which the literature is voluminous and scattered among numerous publications.

The final set of papers deal with pattern recognition theory. Nearest neighbor rules and decision trees are two topics which are important for applications but on which many theoretical and design questions remain. Fukunaga, who has been a prolific contributor to error estimation, density estimation, and classification using K-nearest neighbor rules, presents an overview of his work and discusses the nonparametric analysis of data structure. In the second paper, Dattatreya and Kanal discuss the major methodologies for decision tree design, bring out their commonalities, point out the reasons why decision tree design and use are not simple, provide an insight into the mechanisms of multistage hierarchical classification, discuss how a decision tree designed by any method can be adaptively tuned, and mention areas of application.

In the final paper, Goldfarb presents a mathematical treatment of pattern recognition based on representing patterns in pseudoeuclidean vector spaces, i.e., vector spaces in which the scalar products are given by symmetric bilinear forms which are not necessarily positive definite; the Minkowski space of special relativity is an example of such a space. Goldfarb is interested in exploratory techniques for doing data analysis and pattern recognition with dissimilarity matrices by embedding data points in such vector spaces and considering discriminant functions and clustering in such a space in which some properties of a distance function still hold but in which the triangle inequality need not hold. Although there are many open questions, both theoretical and applied, the approach is of interest because all types of data, statistical, structural, and mixed, could be embedded in some pseudoeuclidean vector space and the type of analytical machinery used in the classical case of statistical data could thus be applied to all such data.

With the broadening of the scope of this series to include machine intelligence, we expect the frequency of these volumes to increase. Several

volumes dealing with specific topics are also in preparation. We hope that readers will continue to find them useful.

Laveen N. Kanal
Azriel Rosenfeld

College Park, MD
October 1984

CONTENTS

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

Image Processing Architectures: A Taxonomy and Survey

S. Yalamanchili,[1,3] K. V. Palem,[4] L. S. Davis,[2,5] A. J. Welch,[4] and J. K. Aggarwal[1,3]

[3]Laboratory for Image and Signal Analysis
[4]Department of Electrical Engineering
The University of Texas at Austin
Austin, Texas, 78712.

[5]Computer Vision Laboratory
Department of Computer Sciences
The University of Maryland
College Park, Maryland, 20742

Abstract

Conventional general purpose computers are unable to meet the computational needs of digital image processing. This inadequacy has led to the conception and subsequent development of a number of special purpose computer architectures for the analysis of static and time varying imagery. Clearly, the evolution of architectures spurred by technological development is progressing rapidly along several dimensions spanning the control, communication and computation domains. It is also becoming increasingly obvious that various classes of image processing problems require widely differing architectural solutions especially when optimality in a cost-performance sense is a major issue. A hierarchical taxonomy is proposed in this paper with the goal of identifying distinct classes of image processing architectures which share similar performance and reliability characteristics. A number of representative systems belonging to each class of the taxonomy are described. Also, performance and reliability issues are discussed within the context of the proposed classification scheme.

Index Terms: *Bus oriented structures, communication elements, functionally dedicated modules, image processing architectures, interconnection structures, performance, processing elements, reconfigurable interconnection structures, reliability and VLSI.*

1 Introduction

Image processing algorithms are characterized by large volumes of data and higher computational requirements than conventional sequential processors can offer [5]. However, images represent a special class of data which allows a significant amount of concurrent processing - a large number of image processing tasks require repetitive logical and arithmetic operations over individual pixels and neighborhoods. These requirements and features resulted in a number of specialized approaches towards developing computer systems for image processing applications.

Advances in technology, particularly the evolution of discrete logic into Very Large Scale Integrated circuit (VLSI) and Very High Speed Integrated circuit (VHSIC) technologies, have allowed the cost effective implementation of several architectures which were previously considered impractical due the attendent cost [6-7]. These advances also

[1]This research was supported in part by a grant from the Air Force Office of Scientific Research under grant AFOSR 82-0064 and in part by a grant from IBM Corporation.

[2] This research was supported in part by a grant from the National Science Foundation under grant EN6-7904037

contributed to the broadening of the range of computational models which can be efficiently mapped onto silicon. In addition, related concepts in parallel and distributed processing have significantly influenced system design philosophies.

An increase in the computational throughput and concomitant complexity of image processing systems has transformed the factors influencing system design. Though maximizing processing speed has remained a constant underlying theme, other considerations have started assuming relative significance. As the number of processors and memory units included in a particular system increased, inter-processor and processor-memory communications have started playing an important role in system design and performance. Awareness of this issue has resulted in extensive research efforts directed towards analyzing the role and impact of interconnection networks and related cost versus complexity tradeoffs in a multiprocessor environment. In addition, considerations related to system reliability and fault tolerance have been recognized to be significant, especially in the context of image processing.

Fu has surveyed special purpose architectures for image processing [8]. These architectures were broadly classified based on bit plane or distributed processing orientations. Davis [9] has proposed a taxonomy which utilizes increasing flexibility with decreasing processing speed as a basis. Danielson and Levialdi [10] have also proposed a classification scheme based on distinct classes of parallelism: operation parallelism, image parallelism, neighborhood parallelism and pixel bit parallelism.

A wide range of image processing architectures are surveyed in this paper. The taxonomy which forms the basis for subsequent discussion is defined initially. The salient features of representative machines belonging to the classes of this taxonomy are highlighted next. Generic examples are chosen from each class and reviewed in the light of relevant issues including system performance and reliability. Finally, observations are made concerning interrelationships between image processing operations ranging from low level to high level tasks and the distinct architectural classes delineated through the taxonomy. These are important in the progressive evolution of architectures towards providing optimal solutions for image processing problems.

2 The Hierarchical Taxonomy

In choosing an appropriate taxonomy for a diverse class of architectures, it is important to identify and emphasize the aspects which reflect the overall design philosophies. Performance and reliability are identified as appropriate choices for achieving the above goal. In addition, a generalized architectural model of an image processing system is developed and serves as a basis for deriving the taxonomy. This model (figure 1) includes a function M(a) which maps an algorithm, a, onto three distinct and possibly distributed architectural entities; control, communication and processing. The taxonomy to be proposed addresses issues related to communication and computation, and results in the partitioning of image processing systems into classes possessing similar performance and reliability properties.

In general, the processing and communication aspects of a multicomputer system can be represented by a pair of *fundamental elements*,

1. Processing elements (PE modules) and

2. Communication elements (CE modules)

respectively. Each processing element could be provided with local memory, share global memory with other PE modules or both (figure 2(a)-(c)). Similarly, each communication element (CE) in turn consists of link elements and a switching element as constituents. A given switching element can have multiple link elements connected to it (figure 3).

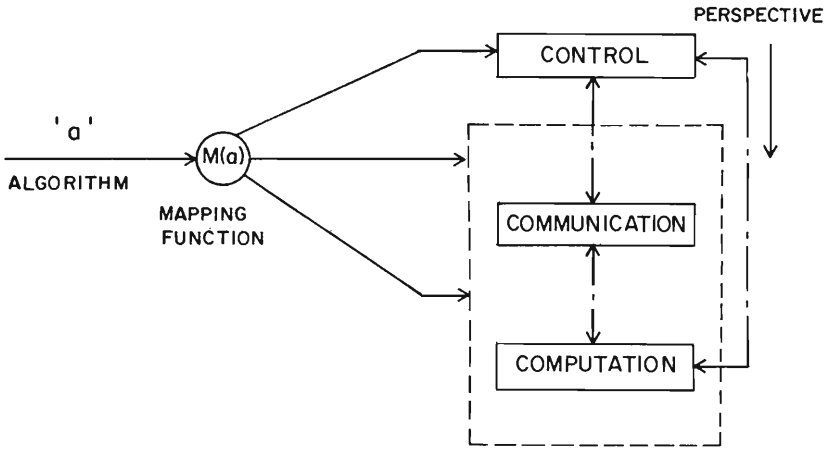


Figure 1: Generalized model of an image processing system. The *perspective* identifies the viewpoint adopted in defining the taxonomy

Technological advances, especially in the LSI and VLSI areas, have influenced image processing architectural philosophies yielding two distinct paths. On one hand, frequently used image processing functions are being implemented as dedicated PE modules to maximize throughput. One such proposal, due to Nudd [11], involves realizing a general class of low level operations as LSI primitives. A distinction between low level and high level processing was made based on the throughput and accuracy required to sustain real time processing rates (figure 4). Alternate proposals [12] recommend systems which strive to match a wide variety of possible processor structures to a broad spectrum of different algorithms. These differing architectural philosophies justify the classification of image processing systems based on the PE module structure into:

1. Homogenous or general purpose programmable modules and
2. Heterogenous or functionally dedicated modules

Parallel processing of images involves mapping the set of tasks into several temporally concurrent sub-tasks generally requiring inter-task communication. This communication is significantly influenced by the interconnection structure of the system [13-15]. In general several logical processing structures, e.g., trees, pipelines, etc., may be realized with a single physical interconnection structure of PE modules through appropriate control and mapping strategies. Three major classes of physical communication structures can be distinguished :

1. *Fixed Interconnection Structures*: Each PE module is connected to n ($1 \leq n \leq N$)

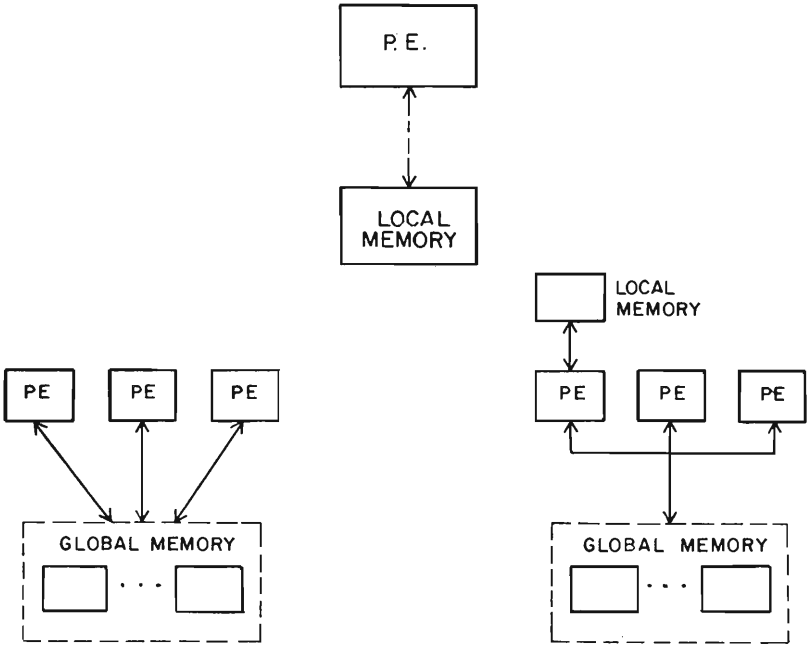


Figure 2: Three cases of processing elements (PEs) (a) Local memory, (b) Global memory and (c) Shared and Global memory

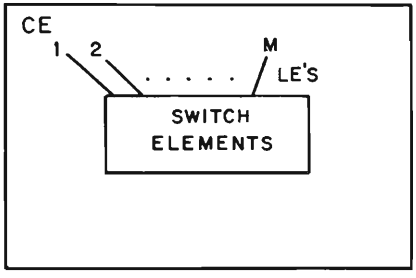


Figure 3: Communication element (CE) with constituent switch element and link elements

neighboring PE modules through a set of CE modules. Two examples of such structures are the near neighbor mesh and ring structures.

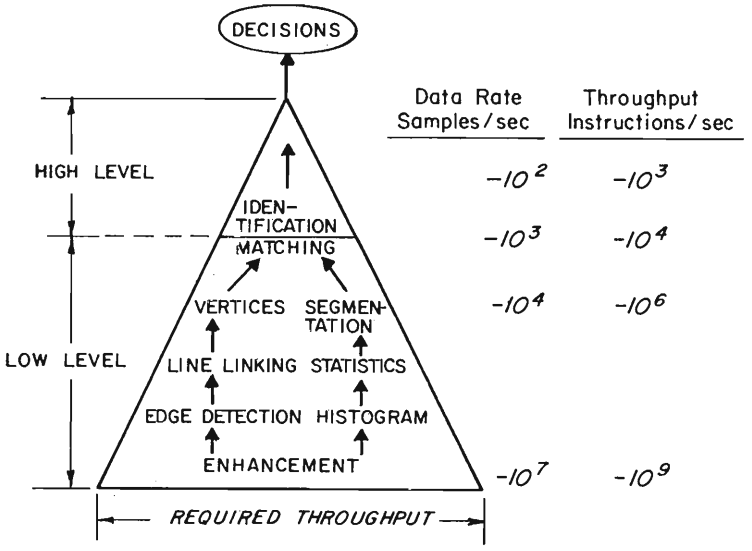


Figure 4: Distinction between low level and high level processing of images

2. *Bus Interconnection Structures*: All PE modules are interconnected through a single CE module.

3. *Reconfigurable Interconnection Structures*: PE modules can be configured to realize any one of several fixed interconnection structures.

The global taxonomy for image processing systems is the result of combining the two classification criteria presented above and is shown in figure 5. Image processing systems are classified into general purpose and functionally dedicated architectures at the first level. Each of these classes is further classified based on the interconnection strategy adopted, into those employing fixed interconnection structures, bus interconnection structures and reconfigurable interconnection structures.

3 Image Processing Architectures

Architectures belonging to the various classes of the taxonomy are described in this section. In describing individual systems, characteristic features of the systems belonging to each class are emphasized.

3.1 General Purpose Architectures

The programmable nature of the PE modules that these architectures are comprised of, provides the flexibility for performing a wide range of operations. The desired throughput rates are achieved through the concurrent operation of a large number of such modules.

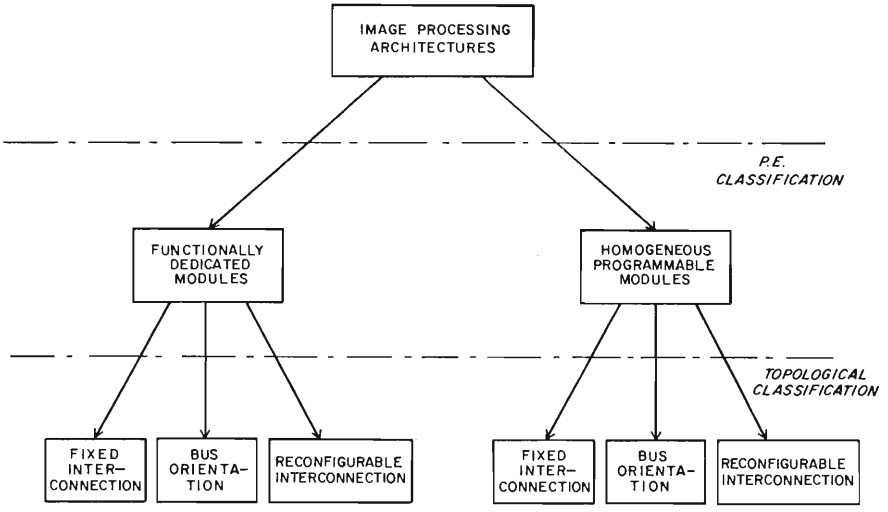


Figure 5: Global taxonomy for Image Processing Architectures

However the use of numerous PE modules to cooperatively perform an image processing task presents complex control problems. A certain amount of overhead is experienced in synchronizing the operations of the individual modules. In addition the communication subsystem introduces delays in the exchange of data and control information. The choice of the control strategies and communication mechanisms employed produces systems of varying performance and reliability.

3.1.1 Fixed Interconnection Structures

Though many of these structures may have existed in theory for quite some time [5], cost effective implementation is only now becoming feasible. An example is a proposal by Unger [5] who suggested a two dimensional array of identical processors, each connected to its four neighbors. A central controller broadcasts an instruction to all the array elements which execute the same instruction on data in its local memory and/or data from its neighbors. Such a system represents a class of fixed interconnection, single instruction stream, multiple data stream (SIMD) machines and are commonly referred to as *cellular logic arrays*. At the very extreme, these arrays may provide a one processor/pixel processing capability. Due to size and cost constraints however, the processor complexity must be kept low, and at the same time consistent with a minimum acceptable level of performance. Similar considerations are applicable to the interconnection network. This has typically resulted in processing elements which operate on pairs of single bits, have limited local memory and single bit wide interconnections to four or eight neighbors.

A number of research efforts have centered around the development of such arrays. The CLIP [16] (*Cellular Logic Image Processor*) program of work being pursued at University College, London, has conducted a significant amount of research on the theoretical and

practical applications of cellular logic to image processing [17]. A series of arrays have been proposed and constructed. These range from earlier special purpose resistor-transistor arrays through the switch and gate array, to the present CLIP series. The latest in this series is the CLIP4 array.

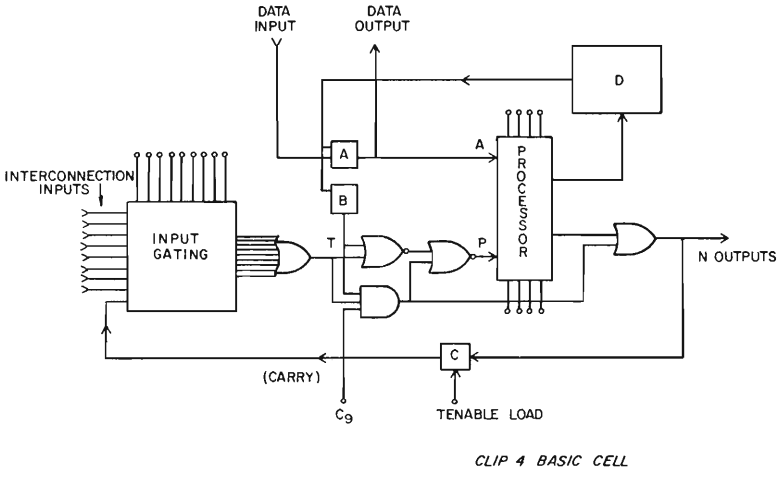


Figure 6: Single cell of the CLIP4 array processor

Figure 6 shows the structure of a single cell of the CLIP4 processor built using LSI technology. This is a 96X96 array fabricated with eight processors per LSI chip. It is capable of performing boolean functions on pairs of single bits. Each cell has 32 bits of memory. Bit serial arithmetic is performed under software control enabling operations on grey level pictures. Each cell is connected to its eight neighbors. From figure 6, it can be seen that the communication elements are integrated into the processing elements. The array is operated by an external controller which decodes program instructions stored in controller memory and drives the array and it's peripherals. CLIP4 operates with a four phase 400ns clock. Complete array operations take under just 9 microsec with 1.2 microsec/cell for propagation beyond the immediate neighborhood. A detailed study of the computational cost of performing some typical image processing functions using CLIP4 as compared to a conventional minicomputer is presented in [18].

Experience with CLIP4 identified areas where further improvements might be made. This along with the need for a processor more closely suited for the processing of grey level images led to the design of CLIP6 [19]. In a CLIP6 processor, all data paths and functional blocks are parallel (6 bits) and double precision where necessary (data paths to the ALU). A condition code register provides some degree of autonomy in reacting to control conditions. Local memory is removed from the processor to allow increased storage

per processing element. The system implementation is currently being investigated and its projected performance is being compared with the performance of CLIP4.

MPP (Massively Parallel Processor [20]) is a cellular logic array built by Goodyear Aerospace Corporation. It consists of a 128X128 array of processing elements built using VLSI technology with eight PE's per VLSI chip. Each processing element possesses a full adder, a variable length shift register and 1K bits of local memory. It can process variable length operands in bit serial fashion. An interesting feature of this array is that it incorporates fault tolerance through redundancy. When a processing element is found to be faulty, the column in which it is contained is bypassed and an extra column of processing elements may be included.

Other cellular arrays which are similar in overall structure, consist of a two dimensional array of bit serial processing elements with a limited amount of local memory associated with each processing element [21-24]. However, they do exhibit considerable differences in processor complexity, implementation, amount of memory associated with each processor, and a few other features.

Cellular arrays, due to their regular structure and synchronous operation are a natural choice for VLSI implementation. Single chip cellular logic arrays of reasonable size appear to be feasible in the near future. To this end, an NMOS VLSI chip containing a cellular logic array [25] to provide a one processor/pixel processing capability is being developed at Stanford University. The goal is to construct a 512X512 array on a single chip. Each processing element consists of a full adder and has the capability for bit serial arithmetic. Local memory consists of 32 bits as two 8 bit registers and one 16 bit register.

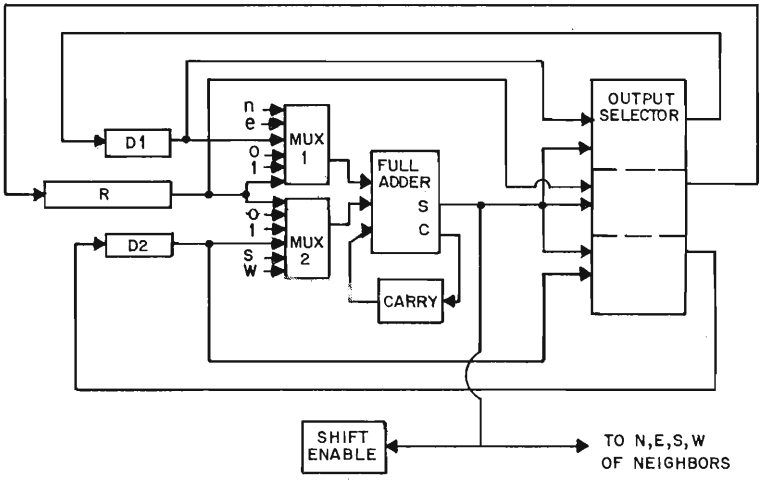


Figure 7: Basic cell of the single chip VLSI array processor

Figure 7 shows a block diagram of the processor. Due to the high density requirements however, a number of problems are expected using currently available fabrication techniques. These include reliablity and extremely low yield. More precise yield models

are being developed so that the fault distribution can be estimated apriori. With a modular design and further advances in fabrication technology, it is expected that a processor array of 128X128 cells on a single chip can be fabricated, which would be capable of 2500 times the operating speed of a DEC KL-10 performing comparable operations.

These arrays derive their high speed from the simultaneous operation of all the processing elements. They are well suited for the computation of independent local operations. However, when communication between processors does not occur in a tightly orchestrated way, efficient algorithm design becomes difficult [26]. In fact, control and communication issues may begin to dominate with a consequent degradation in system performance. As a result, increased interest is being focussed on methods for efficient propagation of results across cellular array structures [27]. Cellular arrays are considered to hold promise for high speed computing in the future [28].

Two dimensional array processing structures not based on cellular logic have also been proposed. One such structure [29] utilizes a state space model to formulate image processing tasks that are linear and space invariant. An implementation using a two dimensional array of microprocessors is proposed. Communication paths between the processors are completely defined apriori. The array operates synchronously and each processor executes a sequence of operations determined by the problem formulation. Separate memories are available to each processor for storing different data types. This increases the number of simultaneous accesses to memory and helps maintain synchronous operation of the array. However, a given problem has to be formulated according to the state space model before this system can be used.

The structure and operation of two dimensional arrays of PE modules are closely matched to the structure of the image data and the spatial dependencies between the pixel elements. A number of alternate data structures have also been developed for the processing of image data, and fixed interconnection SIMD machines which match these data structures have followed. Two examples of such data structures are quad trees and pyramids. A number of algorithms have also been devised which exploit these data structures. Efficient implementations of these structures are being explored. One investigation has resulted in a proposal for a tree structured parallel architecture termed a *pyramid machine* [30]. A pyramid machine is realized by stacking successively smaller two dimensional arrays of processing elements. For example the base of the pyramid may consist of a 64X64 array of processing elements, the next higher level a 32X32 array, the next level a 16X16 array and so on. At any level, except at the boundaries of the pyramid, each processor is connected to a single processor at the next higher level, four processors at the next lower level and to its four or eight neighbors on the same level. The image data is loaded into the base of the array and data and/or results propagate up through the pyramid to the top. Each level operates in SIMD mode in a manner similar to cellular logic arrays. All the levels may be in operation simultaneously and each level can be performing a different operation. In this manner, pipelining across the array may be achieved. This structure is very efficient for executing many of the pyramid and quad tree algorithms and hence represents a multiprocessor architecture for solving a class of problems. Variations of this type of architecture include arrays of simple bit-serial processors operating in SIMD mode at the lower levels with higher levels consisting of networks of smaller numbers of more powerful processors [31].

ZMOB [32] is a multimicroprocessor system consisting of 256 Z80A microprocessors that communicate over a high speed shift register ring called a *conveyor belt*. Each processing element consists of an 8 bit Z80A microprocessor and possesses 63K of RAM. It has a 8X8 hardware multiply unit and a 32 bit floating point arithmetic processor unit. Each PE is interfaced to the conveyor belt through its *mail stop* which is a set of 16 registers that appear as memory locations in its address space. By reading and writing into these locations the processor controls all transmission and reception of data. The buffers in the mail stop allow simultaneous transmission and reception of two bytes of data. Each PE

can also communicate to the external world through a synchronous/asynchronous serial interface and a high speed 24 bit parallel interface. The conveyor belt is conceptualized as 257 *bins*. Each PE can send messages through its own bin to any of the 256 remaining mail stops or receive messages from any of the other mail stops. The 257th mail stop is the unibus interface to the host VAX. A complete revolution around the belt takes 25.7 microseconds (10Mhz clock), and this represents the longest transmission time from one PE to another. This provides each PE with a data transmission rate of 80,000 bytes/sec. This is almost the maximum rate at which a PE can move data into or out of its local memory. Thus a PE never waits for communication. It has access to its bin as fast as it is required, providing for these PEs the effective performance of a crossbar switch. A distributed operating system is being developed for ZMOB [33]. Research is also continuing to determine efficient implementations of image processing algorithms on ZMOB [34].

The fixed interconnection architectures described in this section consist of large numbers PE modules operating in parallel to efficiently perform an image processing task. The CE modules in these structures possess a comparitively simple structure and in many cases are integrated with the PE modules. Any data routing in these systems requires decisions to be made by the PE modules with a consequent increase in the time spent on control and communication. These structures are most efficient, in terms of processing speed, for solving problems that map onto them in a way that minimizes the fraction of time spent by the processing elements on control and communication.

3.1.2 Bus Oriented Structures

In contrast to the fixed interconnection structures these architectures provide a common medium of communication between a set of processing elements. A bus (CE module) represents a shared resource for which *contentions* may arise. In such systems a great deal of effort is spent on devising schemes for efficient allocation of this resource so that contention is minimized.

A block diagram of the *Flexible Image Processor* (FLIP) [35] system is shown in figure 8. FLIP acts as a peripheral device to a host computer. It consists of 16 processors. All the processors are interconnected by individual buses. The Peripheral Exchange Processor (PEP) manages the flow of data into and out of this set of processors. A separate dedicated bus connects each processor to the PEP providing a maximum data rate of 45 Mbytes/sec. Each processor has two input ports and one output port enabling both functions to proceed simultaneously. The flow of data on the buses synchronizes the execution of the processors. Instruction execution involves reading data on the input ports, performing the required computation and placing the result on the output port. Any processor can communicate with any other processor or the PEP in an asynchronous mode of operation. Hence data may flow through any group of processors realizing any logical processing structure. The instruction set of a single processor includes eight bit multiplication, shift instructions and the capability for multiple precision arithmetic. Data and instructions are separated internally both in storage and in the signal paths.

A vision system being developed by the Robot research group at the University of Rhode Island [36] is shown in figure 9. This system is centered around a bus which can be split into a number of independently operating segments. This allows the simultaneous operation of several processors which may access memory or other peripherals attached to the same segment of the bus. This configuration significantly reduces overall bus contention. The bus splitters and processors are supervised by a central controller. Two consecutive segments of the bus may be pipelined by this controller. Thus, dynamic resetting of the bus splitters can provide some degree of reconfigurability. This architecture is currently being tested using two DEC LSI-11 processors, memory modules and a bus splitter.

The requirements of *real time processing* impact system design in various ways, more so

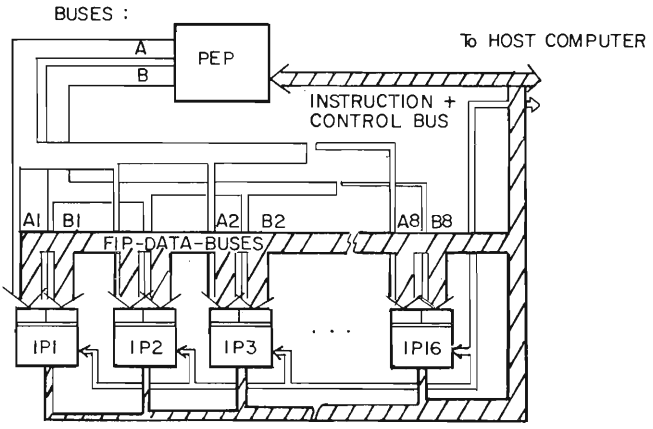


Figure 8: Bus structure of the FLIP system

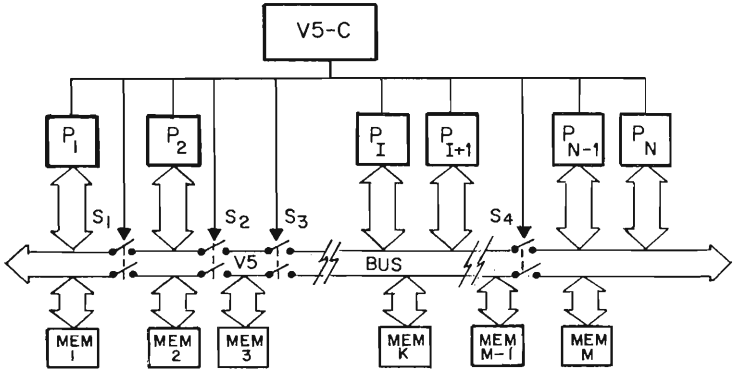


Figure 9: Image processing system based on the splitting bus

in the case of *time varying imagery* which involves the analysis of sequences of images. Figure 10 shows the architecture of a multiprocessor system designed for the real time analysis of time varying imagery [37]. In the development of this system the requirements that it should satisfy were determined to be efficient I/O, sufficient memory for storage of image sequences, capability for parallel access of data and a high degree of physical parallelism. This resulted in the incorporation of dedicated I/O controllers, interleaved

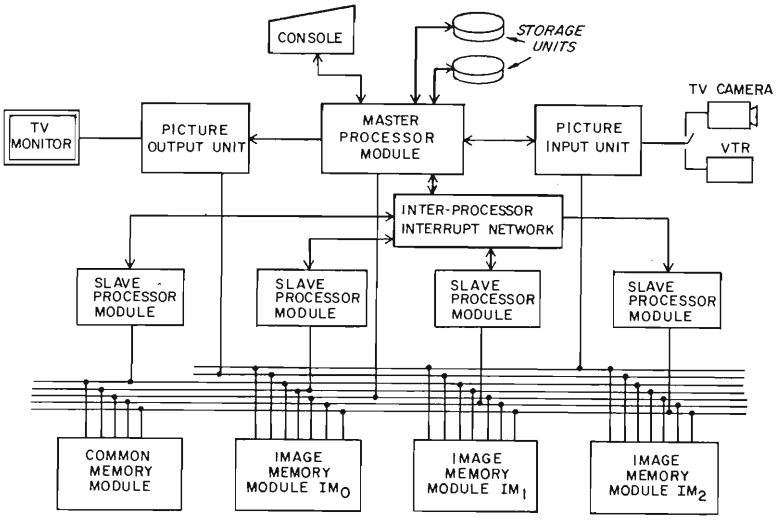


Figure 10: Multiprocessor architecture for analyzing image sequences

memory and a two level processor hierarchy operating in a master-slave mode; a single master processor controlling a set of slave processors. A separate bus for data, address and control connects each processor to the memory modules. Memory conflicts are resolved by an arbitrater associated with each memory bank. A common memory module is used for communication between the slave processors and between the master and the slave processors. Communication between processors is initiated by an interrupt mechanism. On receiving an interrupt the interrupted processor checks the area in the common memory for a message while the interrupting processor may resume processing without having to wait for an acknowledgement. The master processor communicates in this manner with the slave processors to distribute commands, files, programs, etc. It manages the I/O controller, the console and provides an interface to the users. More generally, it fulfills resource allocation and management functions. The master can direct interaction between slave processors to efficiently accomplish any image processing task. In this manner, SIMD mode of operation is possible without the instruction broadcasting or data routing which is characteristic of a typical SIMD environment. This architecture is aimed towards maximizing the simultaneous operation of I/O, control, and processing to achieve real time throughput. The system can be extended to incorporate more than four processors if necessary.

Buses represent shared CE modules. Communication between PE modules pairs using shared buses proceeds one at a time, and as a result, the degree of concurrency in communication that is possible in such systems is reduced in comparison to that in

systems possessing fixed interconnection structures. However, the number of buses typically employed in a system is small compared to the number of dedicated links employed in a fixed interconnection structure. Therefore it is possible to provide high speed buses with larger bandwidths than provided by dedicated links. The overall delay introduced due to communication may be reduced in some instances to a level comparable to that of a fixed interconnection structure executing similar operations.

3.1.3 Reconfigurable Interconnection Structures

Complete interconnection (every PE module connected to every other PE module through dedicated links) is provided by the *crossbar interconnection network.* However its cost grows as the square of the number of PE modules and at present, rapidly becomes infeasible (in terms of cost) for large numbers of PE modules. Reconfigurable interconnection structures have a cost that grows as $O(N\log_t N)$, where N is the number of PE modules, and the networks are built with txt crossbar switches (t is typically equal to two). However, these networks do not provide all possible interconnections between processors. The ability to reconfigure the system to provide the most efficient hardware organization for a given task enables these structures to adapt to a wide range of processing needs possessing diverse communication requirements. Additional flexibility is achieved by being able to partition the interconnection structure into groups of independently operating PE modules.

The block diagram of the PASM [38] (Partitionable SIMD/MIMD) computer system, which is being developed at Purdue University, is shown in figure 11.

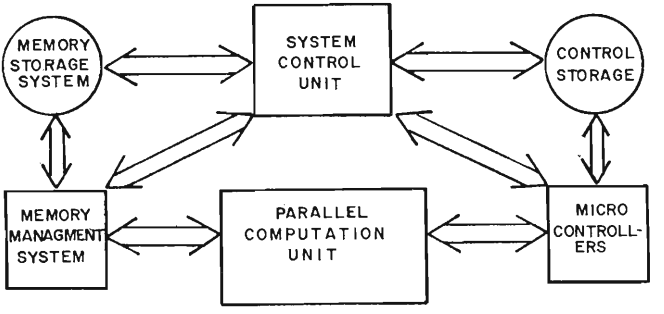


Figure 11: Block diagram of the PASM system

The parallel computation unit contains N processors connected to N memory modules through an interconnection network. N is typically a power of two. A PE module here refers to a *processor-memory module pair.* The micro controllers consist of microprocessors, each of which controls a set of PE modules. Therefore each microcontroller can control an independent SIMD process being executed by the processors under its control. If the microcontrollers are provided with differing instruction streams, MIMD mode of operation results. The independent operation of the controllers allows

multiple SIMD processes to take place. The interconnection network is a multistage switching network with distributed control. This network can be partitioned into independent subnets of PEs facilitating the concurrent operation of independent MIMD and SIMD processes. This network also supports automatic rerouting around busy or faulty network switches and fault tolerance through redundant paths for frequently used permutations. The memory management system employs a double buffering technique to minimize delays in transfers to and from secondary storage. The system control unit is responsible for the overall coordination of the activities of all the components.

A second architecture being developed at Purdue University is the PUMPS architecture for pattern analysis and image data base management [39]. This architecture has been designed to meet the needs of image processing as well as the management of large image data bases. A block diagram of the system is shown in figure 12.

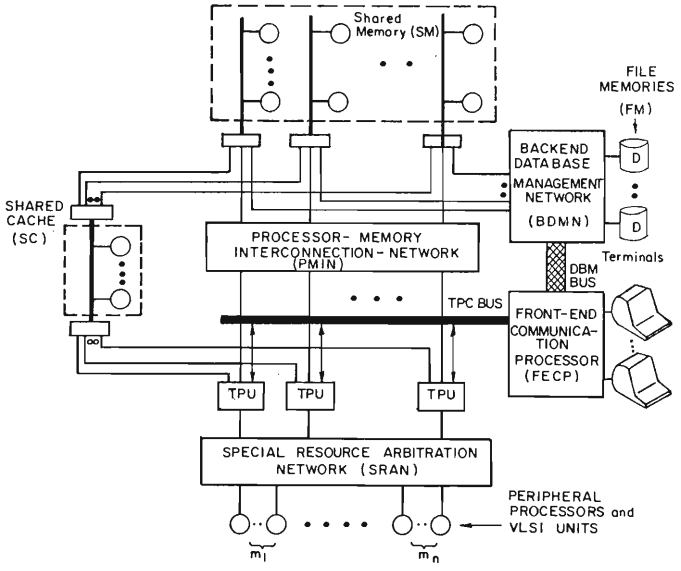


Figure 12: Block diagram of the PUMPS system

It consists of a set of task processing units (TPUs), connected to a shared memory (SM) via a block oriented interconnection network (PMIN). The TPUs communicate over a bus and share a common cache memory. The TPUs are connected through a special resource arbitration network to a shared pool of peripheral processors and VLSI units (PPVUs). A pipeline of PPVUs can be configured under the control of a TPU to perform a specific image processing task. This organization allows convenient addition of special purpose hardware. In addition, the TPUs can operate independently in a MIMD mode. This system has a combination of fast but dedicated low level special purpose architectures in conjunction with more powerful and flexible but slower general purpose processors. The

memory organization consists of a local cache memory in each TPU along with some additional slower local memory. There is also a large cache memory which is shared by all the TPUs. The next level of storage is a shared primary memory and finally a file system on which the database is stored. The file system is connected to the shared memory through the backend database management network. Ideally, multilevel memory hierarchies yield performance of the fastest level at a system cost proportional to the lowest level. The memory management function is distributed over the entire system in order to let the TPUs spend more time on useful computation. Interactive file editing and manipulation capabilities are provided by the frontend communication processor.

A block diagram of the PM^4 system [40] which is also being developed at Purdue University is shown in figure 13.

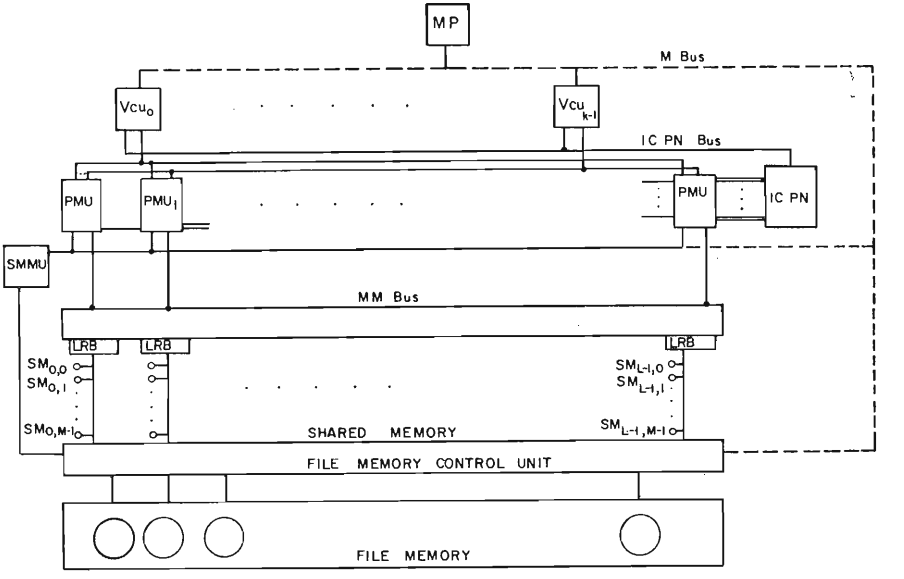


Figure 13: Organization of the PM^4 system

The system contains two types of PE modules - Vector Control Units (VCU) and Processor Memory Units (PMU). The former operate as controllers of an SIMD process executed by a set of PMUs. Each VCU controls a set of PMUs. Therefore, each VCU, in a manner similar to the microcontrollers in PASM, can control an independent SIMD process. The VCUs can execute different instruction streams to operate in MIMD mode. Combinations are possible resulting in distributed mixed modes. The memory has a hierarchical organization consisting of three levels. The highest level is the local cache

memory in each PMU and VCU. The next level consists of the shared memory modules followed by the file storage system at the lowest level. A distributed multiprogramming operating system is being designed for this system. Also an interconnection network for the PMU and shared memory modules is currently being investigated [40].

An architecture for the efficient implementation of algorithms requiring real and complex arithmetic, frequency domain processing of images for example, has been proposed in [41]. A block diagram of this system is shown in figure 14.

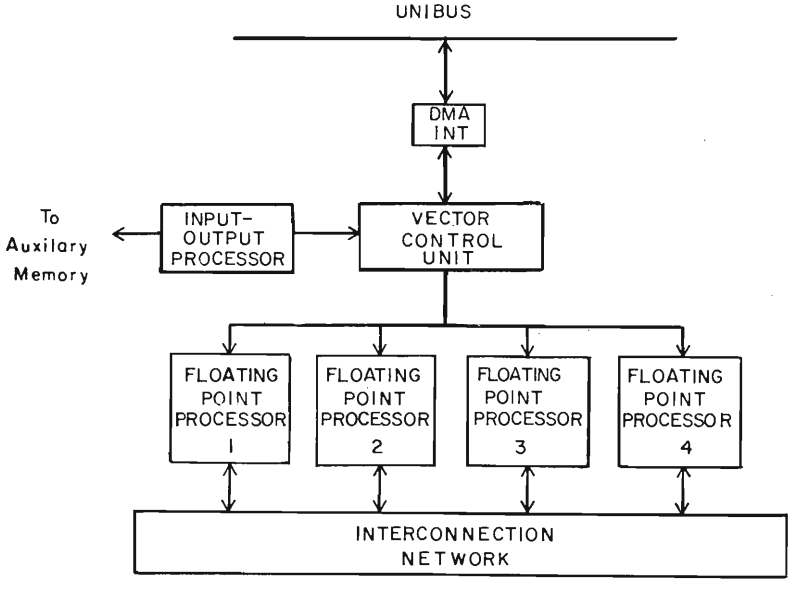


Figure 14: Block diagram of a microprogrammable vector processor

It consists of four floating point processors controlled by a vector control unit. Each processor consists of a floating point adder, a floating point multiplier and local memory interconnected by a flexible bus system allowing concurrent operation of the adder and multiplier. A crossbar network provides the interconnection between the four processors. This vector processor is provided with auxillary memory with which it communicates through an I/O processor. The system operates in a SIMD mode. This architecture is efficient for the processing of large arrays of data. Examples have been presented [41] for performing the FFT and FIR filtering. The crossbar network is used for the exchange of intermediate data. This allows some pipelining of sequences of vector operations. The whole unit is attatched as a peripheral to a host PDP 11/45.

The systems in this section employ interconnection networks consisting of a number of

CE modules. Through various control mechanisms, these networks can be partitioned into independent subnets allowing groups of PE modules to function in MSIMD and MIMD mode of operation. In addition, the feature of reconfigurability enables adaption to various computing structures and data flow requirements. In this manner these systems appear to offer enormous potential for satisfying a wide range of processing and communication requirements.

3.2 Functionally Dedicated Architectures

The extremely high chip densities afforded by LSI and VLSI technology and the associated drop in hardware costs are two significant factors that have contributed to the development of functionally dedicated architectures. In addition, the smaller physical sizes and reduced problems in power and heat dissipation [42], have made them attractive for many applications; real time graphics simulation for example. For a specific problem, it is possible to obtain the required amount of processing logic at a lower cost and packaged in smaller physical volumes.

From an overall system viewpoint, the control and communication requirements of an image processing system based on dedicated functional modules are different from those of general purpose systems. In general, the majority of the communication is for the transfer of data to and from functional units. Cooperation between processors for performing a given task is significantly reduced. Synchronization between processors is mainly for the purpose of initiating and terminating functions, and coordinating data transfers. This simplifies the design of the interface between the various functional units. In such an environment little is to be gained by dynamic hardware reconfiguration capability; hence the lack of such systems. These architectures focus on optimizing performance at the functional module level. This may be at the level of individual image operations, e.g., convolution, or groups of operations constituting an image processing algorithm, such as edge detection, or a set of algorithms corresponding to a specific processing environment, as in real time object tracking. In such systems, *units that perform computations within a PE module will be referred to as PE cells.*

In exploiting VLSI technology it should be noted that the metrics for implementation in VLSI are different from those employed in conventional systems. Excessive communication, long communication paths, or irregular communications degrade performance [42]. As a result, two approaches have evolved in exploiting VLSI technology. The first approach is to *map algorithms directly onto silicon* . The flow of data through the system is completely defined apriori and the PE cells themselves are optimized for high performance. The second approach involves *utilizing a few different types of cells interconnected in a regular pattern.* Typically several data streams flow synchronously through this network, interacting at the cells where they meet. Thus extensive use is made of pipelining and parallel processing. Exploitation of such a structure necessitates careful design of algorithms to produce regular communication patterns and require only a few different types of operations. Examples of architectures with optimization of performance at all the three levels identified above, are described in the remainder of this section.

3.2.1 Fixed Interconnection Structures

A systematic study of applying LSI and VLSI technologies to image processing has been conducted by Nudd [11]. The primary design goal was to provide hardware for performing specific image processing functions at video rates. Towards this end, the first approach, i.e., mapping algorithms onto silicon, was adopted. Certain commonly used low level image processing functions were selected as LSI primitives and LSI circuits were designed to perform each of these functions. A functionally distributed architecture was proposed using a single LSI module for each functional primitive and is illustrated in figure 15. Complex functions are realized as combinations of these primitives. Decision

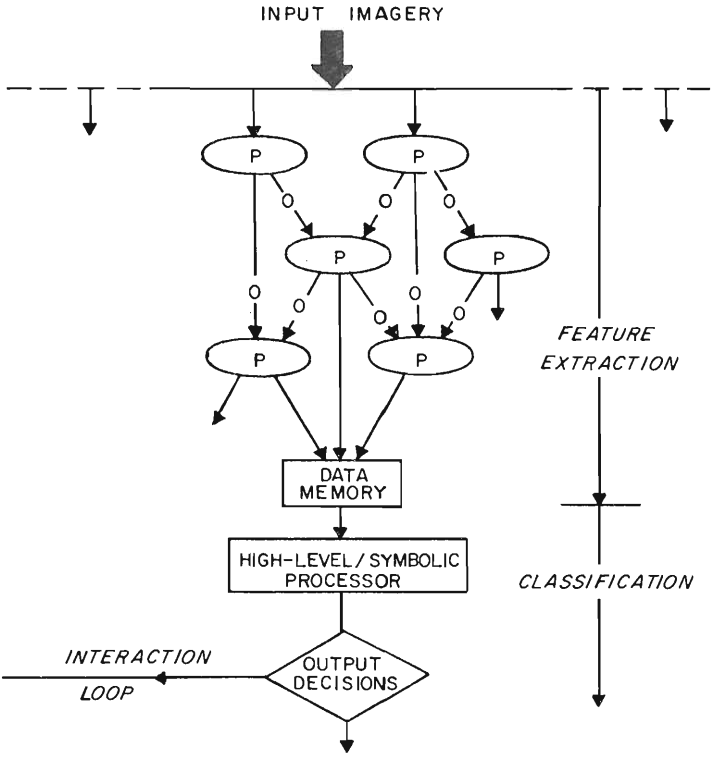


Figure 15: Functionally distributed architecture based on LSI primitives

making and semantic interpretation is associated with a higher level process characterized by low throughput requirements and decreased data rates. In exploiting VLSI technology for the same purpose, Nudd has analyzed some representative systems such as a line finder and a texture analyzer to determine commonality of function. A 5X5 convolution was determined to be a widely employed function and hence was selected for implementation. A design for this chip based on residue arithmetic has been formulated. This approach presents a very modular and regular design, particularly important for VLSI implementation. This chip [43] is now being fabricated and tested. The next stage will involve the development of a local area logic processor to complete a VLSI image processing system structured around 4 chips.

The second approach of employing regular arrays of simple cells is typified by the *systolic processing* approach pioneered by H. T. Kung and his associates at Carnegie Mellon University [44-46]. The basic idea of this approach is to match the computation rate to the I/O bandwidth of the system. Little benefit is derived from possessing processing power in excess of the maximum rate at which data may be acquired.

Therefore in scenarios where data accesses are costly, it is necessary to maximize the use of data fetched from memory or acquired from any other source, such as a sensor. The systolic approach makes use of extensive pipelining and parallel processing in a framework that satisfies the constraints of VLSI implementation. The basic principle is

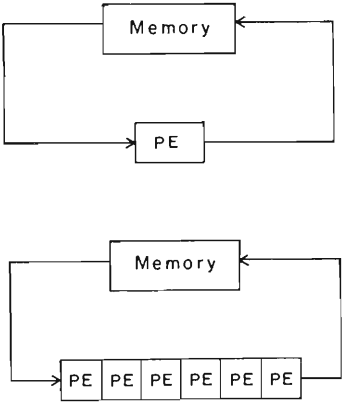


Figure 16: Basic principle of the systolic approach to computation

illustrated in figure 16. The computation throughput is increased six times while the memory bandwidth is constant. Alternatively the memory bandwidth requirements can be reduced for a given computation rate.

As an illustrative example consider one dimensional convolution. A systolic cell and the computation it performs on every clock cycle is shown in figure 17. Now consider a set of numbers $\{x_i\}$, i=1,..,n, to be convolved with a set of weights, $\{w_j\}$, j=1,...,3, to produce an output sequence $\{y_k\}$, k=1,...,n. This may be accomplished by using a linear array of three cells as shown in figure 18 with the weights preloaded as shown. On each successive clock cycle the x_i are input at the left and flow from left to right in figure 18, while each cell performs its computation. The output values are delayed by three clock cycles and also flow from left to right but at twice the speed. After a fixed delay the output values are available at the output of the last cell every successive clock cycle. *Once an element is fetched from memory, all the partial products that may be produced with this element in the course of a computation are computed.* These are then accumulated in different ways to produce the corresponding output values.

Different implementations are possible by employing more complex cells and varying patterns of data movement through linear and higher dimensional arrays. Systolic designs for two dimensional convolution [47-48] and template matching [49] have been developed. These typically employ either a linear array or a two dimensional mesh connected array of cells. It should be pointed out that these arrays possess specific advantages for image processing [50]. Arrays can be tailored to match the data acquisition rates from the sensor for real time operations. In addition many low level processing functions are typically repetitive and the operations are defined independent of the data, i.e, there are no data dependent branches in the algorithm. The small number of different types of operations and the regular computations makes them suitable for VLSI implementation. A 20 cell

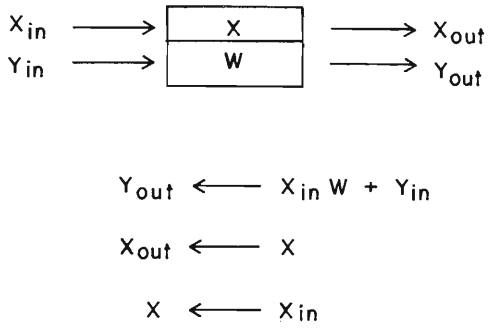


Figure 17: A single systolic cell

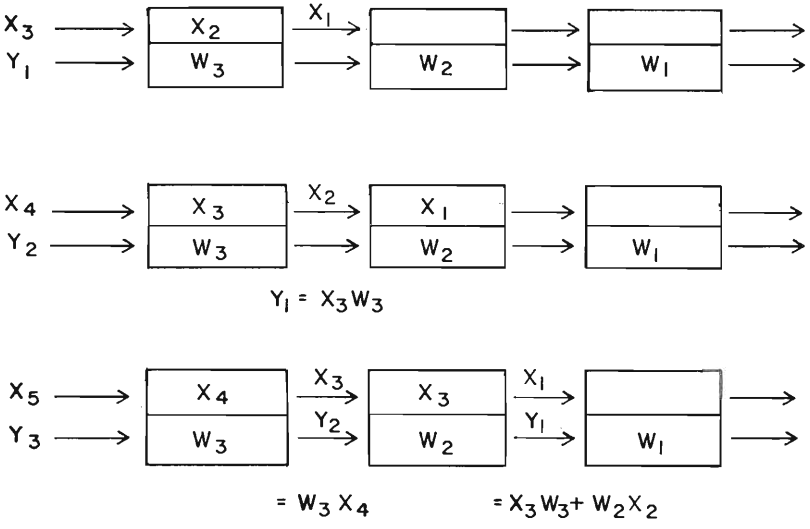


Figure 18: Example of one dimensional systolic convolution

systolic array is currently being developed by ESL Inc., San Jose, CA. This array can be attached to a host VAX 11/780 and has a peak projected computation rate of 200 million operations per second.

Another field that has benefited from the VLSI implementation of image processing functions is high speed graphics. Applications such as real time graphics for flight simulators require high computational throughput. Generation of real time graphics typically involves the following functions: transformation of a scene description in terms of an object centered description to a viewer centered description, clipping into the field of view, perspective scaling, lighting calculation, visible surface calculations and pixel coloring. With the exception of the first operation, all the others are performed at each pixel and the high resolution required in the generation of real world scenes poses a formidable computational burden. Fuchs et. al. [51] have designed a *pixel plane architecture* to satisfy the computational requirements of real time graphics. The basic idea involves enhancing the architecture of a standard display memory which would be normally used for storing an image, by incorporating some additional logic into each pixel storage location. Calculations at each pixel may now be performed in parallel. These pixel planes perform almost all of the image related processing but the total silicon area required is only approximately twice that of the bitmapped display memories implemented using RAM chips [51]. Again, the regularity and use of a single basic cell type makes them amenable to implementation in VLSI. An alternate approach due to Fussell and Rathi is described in [52].

Sternberg [53] has proposed a serial array processor called the *cytocomputer*, based on neighborhood transformations. The heart of this structure is a neighborhood processing unit (PE module) that performs a fixed function; computing the output pixel value as a function of the values of a fixed neighborhood around and including the input pixel. For an image that is supplied in a raster scan manner, a linear shift register is used to perform a raster to window conversion. This presents a neighborhood of a pixel to the processing unit.

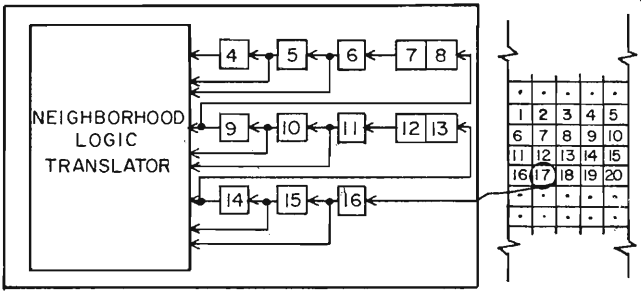


Figure 19: Single stage of a cytocomputer

The organization of a single unit is shown in figure 19. This has been optimized for real time image processing, and has been developed and is operational at the Environmental Research Institute of Michigan. A second generation cytocomputer employing custom LSI technology is currently being developed [54].

Special purpose hardware optimized to perform fixed functions at high speeds is one approach towards achieving the real time capability important for the tracking of moving objects. The development of one such real time automatic tracking system is described in [55]. The system was designed based on the premises that fine tuning of parameters would not be possible in real time and excessive management of data would require too

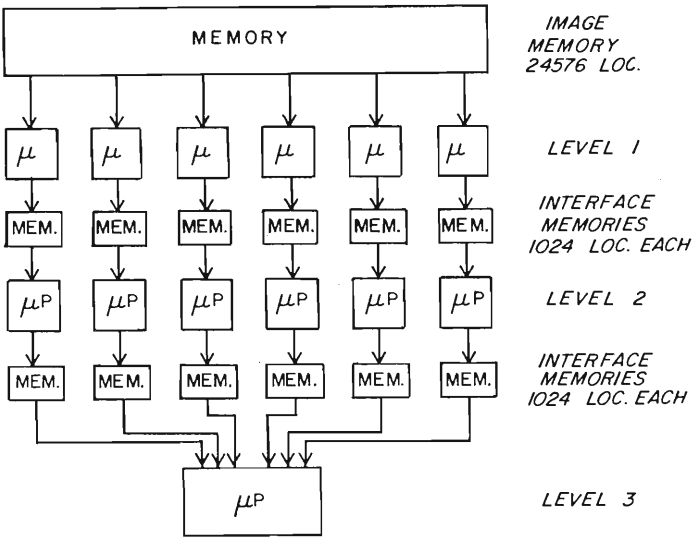


Figure 21: A distributed pipeline architecture for real time image analysis

optimization at the level of an image processing algorithm is described in [57]. Here an algorithm developed to examine a sequence of images has been analyzed to determine the requirements of a multiprocessor implementation that provides real time performance. Based on the results, a dedicated multiprocessor organization was designed.

The advent of *charge transfer technology sensor arrays* stimulated a great deal of interest in *focal plane architectures*. These are architectures integrated into the sensor at the focal plane. Charge transfer technology holds promises for image processing for a variety of reasons [58]. Low power requirements (two orders of magnitude lower than conventional bipolar devices) and smaller device geometries make highly parallel approaches feasible. Performing processing directly in the charge domain, results in greater accuracy, linearity and larger dynamic range. All these factors combine to provide increased sensitivity. These and other potential advantages are spurring the development of focal plane architectures. Systems using such arrays and fast bit slice microprocessors are being developed to implement simple tracking algorithms. A typical aproach is to window the target, threshold the image and process the resulting binary image. Tests on systems using a 100X100 charge injection device array developed by General Electric and a 16 bit microprocessor lend credibility to this approach [59].

An automatic target cueing system integrated into the focal plane is being developed by Westinghouse Corp. [60]. A *preferred* set of algorithms have been evolved which are now being investigated for charge coupled device implementation. This would result in a special purpose but extremely fast architecture for target tracking. The low cost implementation makes special purpose focal plane architectures attractive in spite of their relative inflexibility. The CE modules in these systems are dedicated links between

modules and are therefore in some sense, degenerate. Control of such systems is conceptually simple, being mainly concerned with the coordination of the flow of data to and from the individual modules.

3.2.2 Bus Oriented Structures

Functional modules can be specialized to perform complete image processing tasks. In this case, it would be desirable to share them by providing direct access to any of these modules from a host computer. Several bus oriented systems provide this capability.

A system for the real time acquisition and analysis of sequences of X-ray images is described in [61]. A block diagram of this system is shown in figure 22.

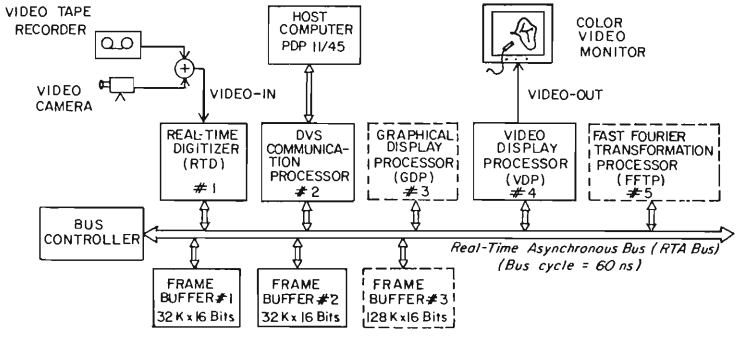


Figure 22: An image processing system for the analysis of image sequences

This system is meant to be an interactive user guided system for clinical applications. In this context, real time implies that the system is as fast as required for convenient interactive analysis. The actual processing of images themselves is limited for the most part to simple unary transformations since the emphasis is on convenient display of images for easier visual interpretation. Most of the effort was concentrated on the problems of real time acquisition, digitization and storage of data (on the RTD and RTA bus in figure 22). An important requirement of this application is the ability to add new special purpose processors in a simple fashion. This, along with an interactive capability, has prompted the choice of a bus oriented system. Similar considerations have led to the adoption of a bus oriented architecture in an environment where it is necessary to handle large amounts of high resolution images for diagnostic medicine [62]. The majority of the design effort is focussed on efficient manipulation of images since the environment is primarily interactive and in comparison, small amounts of processing is required.

A time shared bus forms the basis of the PICAP II image processing system being developed at Linkoeping University, Sweden (figure 23) [63-64]. In designing this system some of the goals included a modular open ended architecture, high speed, and an interactive multi-user environment. Each of the processors is specialized to perform a fixed function: linear and non-linear filtering and segmentation, being typical examples [65].

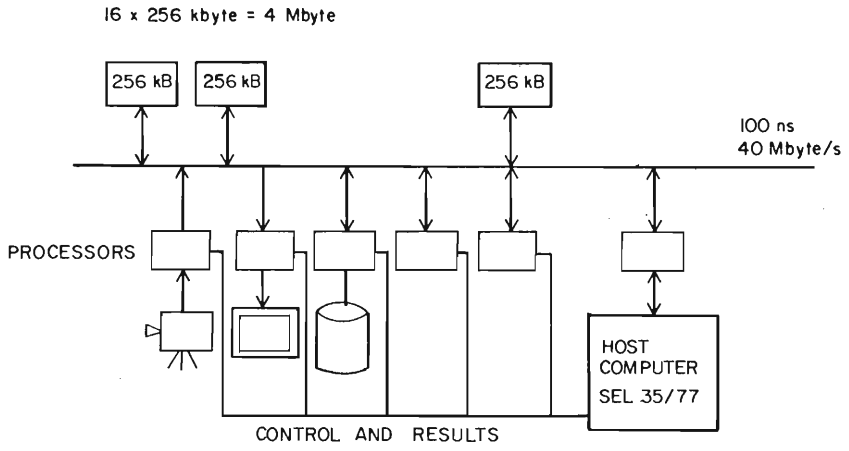


Figure 23: Organization of the PICAP II system

Functional units can be added on as desired upto a total of 16 modules. A picture data base is provided for long term storage. A picture memory is provided for storing the intermediate results during the processing stage [66]. The complete system provides a wide range of facilities for image processing on an interactive basis. Each user has a virtual machine available, as shown in figure 24. The user may operate on any of the images stored in one of the picture registers using the special purpose processors and can then store the results in the data base or have them displayed. Software developed for PICAP II includes PPL, a language developed to reflect the structure of PICAP II.

Another system developed along similar lines is described in [67]. Unlike PICAP II it is not a complete stand alone system. This architecture is proposed to be aimed at the lowest level in a multilevel hierarchy. The organization is shown in figure 25. Here functional modules are tailored to perform a set of low level image processing functions. For example, the edge detection unit is a realization of a 3X3 Sobel operator. The edge detector is implemented as a four level tree and can compute the Sobel gradient magnitude at a point in 240 nanoseconds. The modules operate on an image stored in a common memory. This image can be accessed in fixed size blocks by any of the modules over a 64 bit bus. The output of these processors would proceed to the memories of the next level of processors. Considerable effort was spent on the design of the shared memory so that arbitrarily placed square blocks of fixed size (side of the square being a power of two) could be accessed in parallel.

Considerations that arose in the context of general purpose bus oriented systems are applicable to the systems described in this section. In addition, it has to be noted that inter PE module communication is not as important, since it is not necessary for PE modules to cooperate in performing an image processing task. Communication mainly consists of the transfer of large blocks of data. Hence high speed bus structures may be particularly advantageous, in terms of cost, feasibility, and performance.

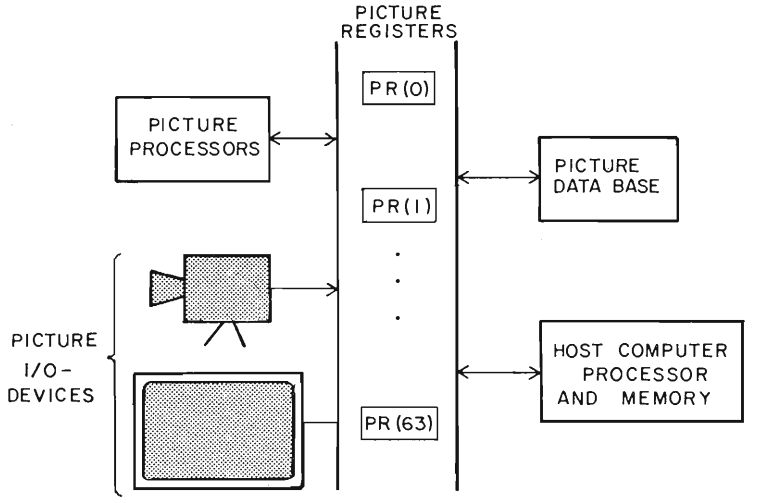


Figure 24: Virtual machine presented to the programmer on the PICAP II system

4 Performance and Reliability Issues

In general, the impact of the organization of PE and CE modules on system performance and reliability is a complex and diverse issue. In addition to addressing several issues of interest in an image processing context, more general issues are also presented wherever deemed appropriate.

4.1 PE Module Considerations

Performance issues related to the PE modules assume special significance when dealing with functionally dedicated modules. Kung [68] has evaluated the implications of implementing algorithms in VLSI. In addition, Nudd [11] has reported performance figures ranging from 80 K Operations/Second for the edge detection operation, to upto 10 Million Operations/Second for convolution using CCD/MOS circuits. Similarly, considerations related to fault tolerance are important in the functionally dedicated PE module case. Since the architecture is implemented on a single chip, the question of replacing faulty elements does not arise. As a consequence, the fault tolerance issues tend to be different when dealing with single chip architectures. Maximizing yield or minimizing faulty elements is a dominant consideration since currently, the size of a chip is limited by the probability of a fatal production flaw occuring within it. This probability factor grows exponentially with the area of the layout [69]. Several groups have addressed this problem in general [70-71] and in image processing domains in particular [25]. Typical solutions strive to provide means for dynamic rerouting of data inside the module to avoid pockets

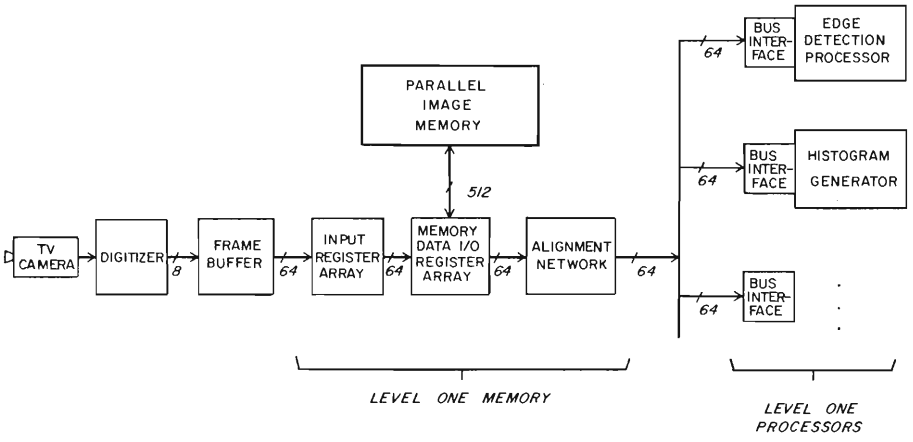


Figure 25: A functionally distributed low level architecture

of faulty locations. It has been observed [70] that to this end, a two dimensional near neighbor mesh is a particularly convenient layout structure. Recent proposals aimed at embedding interconnection structures in a single chip [72] are also bound by similar reliability constraints.

4.2 CE Module Considerations

A wide variety of parameters have been proposed to characterize and classify various interconnection schemes [13-14,73-77]. One such proposal recommends *modularity, connection flexibility, cost of fault tolerance, bottlenecks in communication, and logical complexity* as possible measures for comparative evaluation [13]. In contrast, Siegel et.al [14] have proposed a class of measures which are relevant when reconfigurable multiprocessor systems are considered. The parameters considered by them include *partitionability, homogenity, LSI compatability* and related measures. Lipovski et.al [74] have proposed a graph theoretic basis for modeling and comparing multicomputer interconnection networks. A *rent factor* has been introduced for network assessment which includes the time during which shared reusable resources are unavailable to contending processes. The more traditional complexity and delay factors have also been included. Franklin [72] has observed that while bandwidth and switch complexity [75] are valid and important issues in an environment where switch element costs override link element costs, these assumptions are not consistent when entire networks are implemented on a single chip. The *area occupied by the link elements* assumes significance in the latter case when the link and switch elements share a common wafer surface. Wittie [77] has proposed *average message delay, message densities in a link of the network, worst loss in the event of a single point failure* and related factors for characterizing and evaluating a variety of interconnection strategies involving large numbers of *standard computer modules*.

4.2.1 Fixed Interconnection Structures

In the class of fixed interconnection structures the following three categories are considered: near neighbor meshes, rings and pipelined structures. These structures account for the majority of the interconnection schemes employed in the architectures discussed in the previous section.

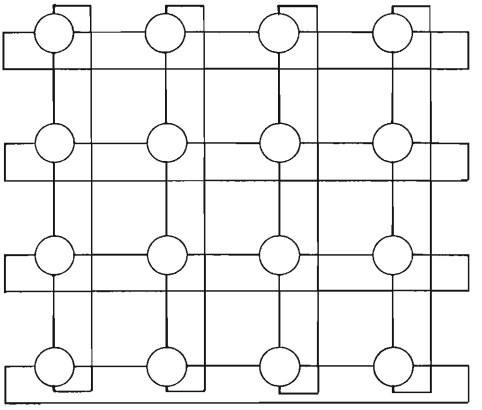


Figure 26: Near neighbor mesh with end around connections

The regular structure of a near neighbor mesh (figure 26) supports simple message routing techniques. It can be shown [77] that the average message traffic density per link in a D dimensional mesh with W elements per dimension ($N=W$ nodes in a cellular array since $D=2$) equals $W/4$. Also, a two dimensional mesh with N nodes and rectangular tesselations requires $2N$ link elements assuming end to end connections. In addition, the regularity of the mesh yields a certain degree of fault tolerance since a single fault does not cause a total loss of communication. However, severe irregularities could result in the routing algorithms as a consequence of local fault pockets. For efficient operation, the majority of the communication should be confined to local areas of the mesh.

Ring structures are represented by a closed loop where any node i in the loop is connected to and hence communicates with it's neighboring nodes $(i+1) mod\ N$ and $(i-1) mod\ N$. The routing algorithms are very simple in this case since a given node of the ring can communicate with either neighbor in a regular fashion (compare this with figure 26 for a near neighbor rectangular mesh in two dimensions). The average message traffic densities per link during a time unit approaches $N/2$, and the average message delays grow linearly with N, the number of nodes in the ring which is configured as a unidirectional loop [77]. In general, a fault at any point results in disconnecting the ring structure and therefore tends to be catastrophic. Redundant paths have to be established if a degree of fault tolerance is to be expected of these structures. Zafiropulo [78] has evaluated alternate techniques for improving reliability in a ring configured as a loop. Finally, the link complexity of a ring grows linearly with N the number of nodes in this loop. The Gatlinburg ring structure, a specalized hierarchy of multiple rings [79] is a typical example of a complex interprocessor communication scheme with rings as fundamental elements.

A typical N stage pipelined system consists of the output of level i serving the input of level $i+1$. Pipelining is an extremely efficient approach both from the performance and resource utilization viewpoints. However, it's implementation grows in complexity except in situations where the precedence graph representation of the particular operation is fully determined apriori. Since the flow of information is regular and controlled in a pipelined system, routing procedures are implicit and simple. However, varying routing patterns requiring variable length or reconfigurable pipelines pose severe problems especially when the various stages of the pipeline function in an asynchronous fashion. While conventional message density and link counts are not meaningful in the context of static pipelines, reliability remains a critical issue. A faulty link at any point along the length of the pipeline results in fragmentation. Redundant links can solve this problem. Ramamoorthy et.al. have surveyed the advantages and needs of pipelined units in considerable detail [80]. The issues related to dynamically reconfigurable pipelines have been addressed by Vick et.al. [81].

4.2.2 Bus Interconnection Structures

Both single global bus and multiple bus structures have been used for connecting ensembles of processors and memory units (PE's). Here, the memory units in addition to serving as storage media, also function as common locations for information exchange. Information or message routing is very simple and is commonly accomplished through a memory mapped scheme. Sytem expansion is accomplished through the addition of PEs to the bus at the cost of degraded bandwidth. Contention for the shared bus becomes an important problem and several solutions strive to resolve this contention; for example the Fast Time Shared Bus proposal for the PICAP II system [63-64]. Thus, expanding the system bandwidth affects bus design considerations considerably. In general, a single fault is sufficient to cause bus fragmentation. However, the cost of the interconnection structure remains a constant for a given bus.

4.2.3 Reconfigurable Interconnection Structures

The crossbar switch is perhaps the earliest interconnection scheme used for inter PE module communication in restructurable and partitionable multicomputer systems. In order to connect N input nodes to an equal number of output nodes, a crossbar network requires N^2 switching elements with a delay equal to one level of switching. Later proposals, while reducing the network complexity, introduced additional delay into the communication [14]. These proposals became increasingly optimum in a cost sense as N, the number of input and output nodes, grew large. In general, these interconnection schemes require multiple levels of switching between the input and the output node sets. This fact emphasizes the need for efficient routing techniques in this case for good performance.

Chow et.al [82] have reported the development of optimum routing methods for some reconfigurable interconnection networks. They have observed that while a variety of routing algorithms have been proposed in the past [83-86], these techniques either require extremely large memory or large amounts of processing time during backtracking. It is claimed that their technique requires less memory and supports synchronous and asynchronous routing. Finally the presence of multiple communication paths at a given switch in the network requires that the routing scheme be capable of handling congestion problems dynamically. Wu et.al [87] have addressed these issues in the context of the hypercube interconnection scheme.

An important characteristic of this class of interconnection networks is the ability to partition the resource set into discrete subsets functioning independently in MSIMD, MIMD and SIMD modes of operation. Thus, operational states involving parallel operations on multiple images become feasible in an image processing context. Fault tolerance and related issues of the Adaptive Data Manipulator network have been

reported by Siegel et.al [88]. Typically, delays for these multistage networks for end to end communication, grow in proportion to the depth as $O(log_2 N)$ while the cost in terms of the switch and link complexity is $O(N log_2 N)$.

4.2.4 Comparative Evaluation

Kushner et.al [89,32] have compared the performance of the MPP (near neighbor mesh) and ZMOB (a ring interconnection system) for a class of frequently used image processing functions including: (1) iterative local operations, (2) histogramming, (3) Co-occurrence matrices and (4) two dimensional transforms. Also Kushner [32,89-90] has compared the performance of each of the above configurations to that of a system using switching networks for inter PE communication. For purposes of uniformity, it was assumed that the switching network case utilizes PE modules driven by a host computer through a unibus. Typical expressions representing the computational delay for an iterative local operation are indicated in Table. 1 for each of these configurations.

Table 1

System	*Time for the iterative local operation*
ZMOB	$2rN^2 + (4n+4)nsP + n^2mC$
MPP	$2rN^2 + (4n+4)mlp + n^2mC$
Switching Networks	$2rN^2 + (4n+4)ms2log_2P + n^2mC$

N = Number of pixels in a column of a square image
P = Number of processors
C = Time to compute one local operation
n = Number of pixels in one column of the square image partition assigned to one processor
m = Number of iterations of the local operation
r = Time to pass one pixel over the unibus
p = Time to pass one bit between adjacent processors (MPP)
l = Number of bits/pixel
s = Single stage pixel propagation delay

5 Concluding Remarks

A wide range of image processing architectures have been surveyed in this paper. A taxonomy which focuses on the communication and computational aspects of a generalized model of an image processing system, has been proposed. This approach leads to a hierarchical classification of image processing architectures based on the nature of processing elements (PEs) and interconnection networks comprised of communication elements (CEs) for inter PE communication. Several representative architectures belonging to each class were subsequently described. Typical characteristics of each architecture were emphasized with the intent of highlighting the distinction between

classes. Finally, the performance and reliability characteristics of each class of the taxonomy were discussed.

It is possible to establish certain relationships between low level and high level image processing tasks and the architectural classes described earlier. A majority of low level processing algorithms involve independent point and local operations over an image and are thus amenable to highly parallel implementations. The diversity in the type of computations, and in the communication requirements of these operations is, in general, small. In addition, the communication requirements of algorithms at this level are fairly well understood and are generally characterized by *regular* patterns. Thus, architectures employing functionally dedicated modules and fixed interconnection networks are attractive choices in this context. Architectures developed for low level processing reflect these observations [11,42,43, 45-55,57-60,63-67].

When image processing tasks are implemented in a dedicated fashion as in the implementation of edge detection modules [67], communication with the environment is limited to regular block data transfers. Thus, the use of functionally dedicated PE modules restrict the variance in communication requirements from the point of view of a host. Therefore in such environments, a bus interconnection structure offers a cost effective solution. These observations suggest the evolution of systems employing several dedicated implementations of low level operations centered around a global bus.

The feasibility of developing highly parallel systems of functionally dedicated PE modules for low level image processing has been enhanced with the advent of VLSI systems. While decreasing device sizes have resulted in improved speed and cost functions, several new problems had to be contended with. Firstly, this evolution of technology led to significantly altered optimization criteria which in turn have affected the design process. In exploiting VLSI technology, a large number of PE cells could be encapsulated in a module to provide the high throughput rates required of low level processing. However, the accompanying I/O problem due to pin-out constraints imposes restrictions on the corresponding range of problems which can utilize such solutions. Also, the impact of the layout geometry on performance increases with decreasing device sizes and thus, plays a critical role during the chip design process. Therefore, systematic system analysis, synthesis and verification procedures become intractable soon. Increasing system design and verification costs are a direct consequence of these factors which tend to offset the advantages gained through reduced device geometries.

Perusal of the issues summarized above suggests the need to automate the various stages of the design and verification processes if lower system costs are desired. This need has been recognized, and as a result several independent research efforts have produced various automatic placement routing and partitioning techniques for VLSI chip layout and design. It seems likely that further research will be directed towards developing compilation techniques which would yield silicon embeddings of algorithms based on procedural descriptions of the function.

High level operations possess contrasting features. Currently, algorithms are developed based on relatively complex models of scene analysis and human vision. The diversity in the types of computations performed at this level is large. In addition, the process of detecting and exploiting parallelism at this level is not well characterized and is therefore an area of current research. In general, the variability in the communication requirements of parallel high level algorithms is large. This large variability in computation and communication requirements suggests that flexibility is a necessity, implying the need for programmable PE modules and reconfigurable interconnection structures. These observations are reflected in the architectures developed primarily for high level image processing[12,16,20-22,36,38-41]. Typically these systems possess a large degree of flexibility both from a computation and communication point of view. However, with such diverse processing requirements, ***mismatch*** between the processing requirements of the algorithm and the solution offered by the host architecture may be relatively high.

These facts point to the need for determining strategies for effectively matching problem classes with reconfigurable systems. Therefore, it is important to identify classes of high level algorithms with similar computation and communication requirements. This understanding will further the development of dynamic run time systems for optimally mapping these problem classes onto the underlying architectures.

6 References

1. *Proceedings of the 5th International Conference on Pattern Recognition*, Miami Beach, Fla., Dec. 1-4, 1980.
2. *Proceedings of the Pattern Recognition and Image Processing Conference*, Dallas, Tx, Aug. 3-5, 1981.
3. *Proceedings of the Workshop on Picture Data Description and Management*, Asilomar, CA., Aug. 27-28, 1980.
4. *Proceedings of the Workshop on Computer Architecture for Pattern Analysis and Image Database Management*, Hot Springs, Va., Nov. 11-13, 1981.
5. S. H. Unger, A computer oriented towards spatial problems, *Proc. IRE.*, pp. 1744, October 1958.
6. E. Bloch and D. J. Galage, Component progress: Its effect on high speed computer architecture and machine organization, in *High Speed Computer and Algorithm Organization*, D. J. Kuck, D. H. Lawrie and A. H. Sameh (eds.), Academic Press, pp. 13-39, 1977.
7. D. A. Peterson and C. H. Sequin, Design considerations for single chip computers of the future, *IEEE Trans. on Computers*, C-29, No. 2, Feb. 1980.
8. K. S. Fu, Special purpose architectures for image processing, *Proc. of the National Computer Conference*, pp. 1003-1013, 1978.
9. L. S. Davis, Computer architectures for image processing, pp. 249-254, in [3].
10. P. E. Danielsson and S. Levialdi, Computer architectures for pictorial information systems, *Computer*, pp. 53-67, Nov. 1981.
11. G. R. Nudd, Image understanding architectures, *Proc. of the National Computer Conference*, pp. 377-390, 1980.
12. P. H. Swain, H. J. Siegal and J. El-Achkar, Multiprocessor implementation of image pattern recognition: A general approach, pp. 309-317, in [1].
13. G. A. Anderson and E. D. Jensen, Computer interconnection structures: Taxonomy, characteristics and examples, *ACM Comput. Surveys*, Vol. 7, pp. 197-213, Dec. 1975.
14. H. J. Siegel, R. J. McMillen and P. T. Mueller Jr., A survey of interconnection methods for reconfigurable parallel processing systems, *Proc. of the National Computer Conference*, Vol. 48, pp. 387-400, 1979.
15. T. Feng, A survey of interconnection networks, *Computer*, Vol. 14, pp. 12-27, Dec. 1981.

16. M. J. B. Duff, Review of the CLIP image processing system, *Proc. of the National Computer Conference*, pp. 1055-1060, 1978.

17. K. Preston Jr., M. J. B. Duff, S. Levialdi, P. E. Norgen and J. Toriwaki, Basics of cellular logic with some applications to medical image processing, *Proc. of the IEEE*, Vol. 67, No. 5, pp. 826-856, May 1979.

18. L. P. Cordella, M. J. B. Duff and S. Levialdi, An analysis of computational cost in image processing: A case study, *IEEE Trans. on Computers*, Vol. C-27, No. 10, pp. 904-910, Oct.1978.

19. T. J. Fountain, Towards CLIP6 - An extra dimension, pp. 25-30, in [4].

20. K. E. Batcher, Design of a Massively Parallel Processor, *IEEE Trans. on Computers*, Vol. C-29, pp. 836-840, 1980.

21. A. P. Reeves, A systematically designed Binary Array Processor, *IEEE Trans. on Computers*, Vol. C-29, pp. 278, 1980.

22. P. Marks, Low level vision using an array processor, *Comp. Graphics and Image Proc.*, Vol. 14, pp. 281-292, 1980.

23. P. M. Flanders, D. J. Hunt, S. F. Reddaway and D. Parkinson, Efficient high speed computing with the Distributed Array Processor, in *High Speed Computer and Algorithm Organization*, D. J. Kuck, D. H. Lawrie and A. H. Sameh (eds.), Academic Press, New York, pp. 113-127, 1977.

24. S. F. Reddaway, The DAP approach, in *Infotech State of the Art Report on Supercomputers*, Vol. 2, pp. 309-329, 1979.

25. M. R. Lowry and A. Miller, A general purpose VLSI chip for computer vision with fault-tolerant hardware, *Proc. of the DARPA Image Understanding Workshop*, pp. 184, April 1981.

26. K. Preston Jr., Application of parallel processors, pp. 263-265, in [3].

27. M. J. B. Duff, Propagation in cellular logic arrays, pp. 259-261, in [3].

28. M. J. B. Duff, Future trends in cellular logic image processing, pp. 294-297, in [3].

29. R. P. Roesser, Two dimensional microprocessor pipelines for image processing, *IEEE Trans. on Computers*, Vol. C-27, No. 2, pp. 144-156, Feb. 1978.

30. C. R. Dyer, A VLSI pyramid machine for hierarchical parallel image processing, pp. 381-386, in [2].

31. L. Uhr, Converging pyramids of arrays, pp. 31-34 in [4].

32. T. Kushner, A. Wu and A. Rosenfeld, Image processing on ZMOB, *IEEE Trans. on Computers*, Vol. C-31, No. 10, pp. 943-951, Oct. 1982.

33. R. Bane, C. Stanfill and M. Weiser, Operating system strategy on ZMOB, pp. 125-132, in [4].

34. C. Rieger, ZMOB: Doing it in parallel, pp. 119-124, in [4].

35. K. Luetjen, P. Gummar and H. Ischen, FLIP : A flexible multiprocessor system for image processing, pp. 326-328, in [1].

36. J. D. Dessinoz, J. Birk, R. Kelly and J. Hall, A vision system with a splitting bus, pp. 62-66, in [4].

37. W. H. Tsai, C. C. Chou, P. C. Cheng and Z. Chen, Architecture of a multiprocessor system for parallel processing of image sequences, pp. 104-111, in [4].

38. H. J. Siegel, L. J. Siegel, F. Kemmerer, P. T. Mueller Jr., H. E. Smalley Jr., and S. D. Smith, PASM: A Partitionable Multimicrocomputer SIMD/MIMD system for image processing and pattern recognition, *IEEE Trans. on Computers*, Vol. C-30, No. 12, pp. 934-946, Dec. 1981.

39. F. A. Briggs, K. Hwang, K. S. Fu and M. Dubois, PUMPS architecture for pattern analysis and image data base management, *IEEE Trans. on Computers*, Vol. C-31, No. 10, pp. 968-982, Oct. 1982.

40. F. A. Briggs, K. S. Fu, K. Hwang and J. M. Patel, PM^4--- A reconfigurable multiprocessor system for pattern recognition and image processing, *Proc. of the National Computer Conference*, pp. 255-265, 1979.

41. B. Pavin and K. S. Fu, A microprogrammable vector processor for image processing applications, pp. 287-292, in [3].

42. M. J. Foster and H. T. Kung, The design of special-purpose VLSI chips, *Computer*, pp. 26-40, Jan. 1980.

43. S. D. Fouse, G. R. Nudd and G. M. Thore-Booth, A residue-based image processor for VLSI implementation, *Proc. of the DARPA Image Understanding Workshop*, pp. 188-198, April 1981.

44. H. T. Kung, Lets design algorithms for VLSI systems, *Proc. Caltech Conf. in VLSI*, Cal. Inst. of Tech., Pasadena, Calif., pp. 65-90, 1979.

45. H. T. Kung and P. L. Picard, Hardware pipelines for multidimensional convolution and resampling, pp. 273-277, in [4].

46. S. L. Tanimoto, Systolic cellular logic: Inexpensive parallel image processors, *TR. 80-11-02*, Department of Computer Sciences, University of Washington, Seattle, Washington, 98175, Nov. 1980.

47. D. W. L. Yen and A. Kulkarni, The ESL systolic processor for signal and image processing, pp. 265-271, in [4].

48. H. T. Kung and S. W. Song, A systolic 2-D convolution chip, pp. 159-160, in [4].

49. T. Y. Young and P. S. Liu, VLSI arrays and control structures for image processing, pp. 257-264, in [4].

50. P. Narendra, VLSI architectures for real time image processing, pp. 303-306, *Compcon*, Spring 1981.

51. H. Fuchs, J. Poulton, A. Paeth and A. Bell, Developing pixel planes, a smart

memory-based raster graphics system, *Proc. of the Conf. on Advanced Research in VLSI*, M. I. T., Cambridge, Mass., pp.137-146, Jan. 1982.

52. D. Fussell and B. D. Rathi, VLSI oriented architectures for real time raster display of shaded polygons, *Proc. of the Graphics Interface 82 Symposium*, pp. 373-380, May 1982.

53. S. R. Sternberg, Architecture for neighborhood processing, pp. 374-380, in [2].

54. R. M. Loughheed, D. L. McCubbery, and S. R. Sternberg, Cytocomputer: Architecture for parallel image processing, pp. 281-286 in [3].

55. A. L. Gilbert, M. K. Giles, G. M. Flachs, R. B. Rogers and Y. H. U, A real time video tracking system, *IEEE Trans. on Patt. Anal. and Mach. Intell.*, Vol. PAMI-2, No. 1, pp. 47-56, Jan. 1980.

56. T. R. Husson and A. M. Abdulla, Real time infrared image processing, pp. 478-480, in [2].

57. D. P. Agrawal and R. Jain, A pipeleined pseudoparallel system architecture for dynamic scene analysis, *IEEE Trans. on Computers*, Vol. C-31, No. 10, pp. 952-962, Oct. 1982.

58. G. R. Nudd and P. A. Nygaard, Charge transfer technology for image processing, *Proc. of the 7th Annual Automatic Imagery Pattern Recognition Symposium*, pp. 4-7, 1977.

59. J. L. Reidl, CCD sensor array and microprocessor applications to military missile tracking, *Proc. of the SPIE*, Vol. 95, pp. 148-154, 1976.

60. T. J. Wille and N. Bluzer, Automatic target cueing system on the focal plane, *Proc. of the 7th Annual Automatic Imagery Pattern Recognition Symposium*, pp. 8-9, 1977.

61. von G. C. Nicolac, K. H. Hotine, Multiprocessor system for the real time digital processing of video image series, *Elektronische Rechenanlagen*, No. 21, pp. 171-183, 1979.

62. D. Meyer-Ebrecht, The mangement and processing of medical pictures: An architecture for systems and processing devices, pp. 202-206, in [3].

63. P. Danielson, The time-shared bus-A key to efficient image processing, pp. 296-299, in [1].

64. D. Antonsson et.al., PICAP - A system approach to image processing, pp. 35-42, in [4].

65. B. Kruse, B. Gudmundsson, D. Antonsson, FIP- The PICAP II filter processor, pp. 484-488, in [1].

66. P. Danielson, B. Kruse, B. Gudmundsson, Memory hierarchies in PICAP II, pp. 275-280, in [3].

67. D. Sherdell, A low level architecture for a real time computer vision system, pp. 290-295, in [1].

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

COMPUTATIONAL MODELS FOR IMAGE UNDERSTANDING

C.Guerra*, S.Levialdi**

* Dipartimento di Informatica e Sistemistica, University of Roma, Roma, Italy
** Istituto di Scienze dell'Informazione, University of Bari, Bari, Italy

INTRODUCTION

The motivation for improving computational efficiency is the need to process an ever growing amount of information. This is particularly true in fields like image processing and pattern recognition where a large number of sensors are continually gathering information from the earth and the skies (remote sensing), biomedical sources (computer tomography), industrial environment (robot vision), image sequence analysis (at TV rate), etc.

For this reason the interest in performing very fast computations on large sets of structured data (such as digital images) has increased in recent years. The problem of augmenting throughput must be solved by designing new computer architectures that exploit parallelism of some kind instead of simply improving the efficiency of present sequential machines.

These new architectures give rise to new problems in the areas of interprocessor communication , memory contention, synchronization and overall reliability. On all these issues current research is active, generating a large number of papers and international meetings [34] [35], both in the field of scientific computation and in image processing.

On a more abstract level, computational models provide a general description and interpretation of information flow and control that takes place in a given class of machines.

It seems appropriate to explain both terms (computation and model) since they involve a large number of concepts, which are in most instances difficult to formalize.

The word "model" generally reminds us of an analogy between a real situation and an "imaginary

machine" which depicts such a situation [11]. The analogy may be of phisycal nature (an orange to model the sun) or it may be more abstract, namely, of a relational nature. In this last case the relationship among the variables in the situation is preserved in the proposed model. Frequently such models are a crude approximation to reality and therefore the non-significant details are ignored in order to highlight the important features. This is done for two reasons: a) to deeepen the understanding of the situation one is trying to model and b) to build a tool that may also have a predictive potential for different instances of the same situation.

Strange as it may seem, both scientific and magic attitudes bear a close resemblance from the point of view of the model builder. They are both embodiments of knowledge models in much the same way as a politician believes and operates in an ideological model in which he "lives". It may often happen that reality is completely substituted for the model which amounts to living in the model instead of being an observer of the model.

Understanding is an interactive process between the observer and the reality; the model is an intermediate reality, one that may be more easily grasped. If a first model is refined, the closeness to the real situation will be reinforced but its complexity will also be increased; this suggests the search for a tradeoff between accurateness and sophistication of the model.

As for the second term, computation, we denote by it a process by means of which a mapping is performed between the elements of two structured sets (domains), one corresponding to the input and the other to the output. In the case of image processing both the input and the output will be sets of structured data of equal size made of elements called pixels; in the case of image analysis and pattern recognition the output will be a description which may be either a list of primitives or a set of names assigned to the objects present in the image. The computation is achieved by a system which has resources of different kinds: processors, memory and time.

The availability of many processing elements and their local memories, made possible by the development of VLSI technology, suggests a wide number of configurations, both static and dynamic, and this has resulted in the implementation of a number of different computational systems. The features of such systems and their classification acccording to the way they match the data structure (image) or the computation to be performed (algorithm) have been recently discussed in [7]. In order to start from the begining we will now proceed to briefly review some classical models for computation which will shed a light on the major aspects of parallel processing in pictorial data systems.

MODELS

Basic definitions

A formal definition of a parallel computational model may be given in terms of two crucial

components, the control structure and the interpretative capacity, which act in a manner similar, but not equivalent, to the syntactic and semantic description of programming languages.

The control structure may be defined as

$C = (Q,\delta,\Sigma,S,F)$

where

Q is the state space (not necessarily finite) containing states q

$\delta: Q \times \Sigma \rightarrow Q$ is the transition function

Σ is the alphabet of operation symbols

$S \in Q$ is the start state

$F \subset Q$ is the set of final states.

The transition function is a partial function, meaning that only some transitions are allowed to occur since not all states can start a transition.

When an operation is executed, $\delta(q,\sigma)$ determines the new state; furthermore, an initial state S exists, and the computation ends whenever a final state belonging to F is reached. The computation can be seen as a sequence of operations $\sigma_{i1},\ldots,\sigma_{in}$ and a sequence of states $q_{j0},\ldots,q_{jn}$ such that

i) $q_{j0} = 0$

ii) $\delta(q_{jk-1},\sigma_{ik}) = q_{jk} \qquad 1<k<n$

iii) $q_{jn} \in F$

The set of all the computation sequences for a control C is called its computation sequence set L(C).

CELLULAR AUTOMATA

Two-dimensional networks of finite state automata are known as cellular automata and were introduced as a model for self reproduction. In the pattern recognition field, such automata are employed to detect specific configurations. This process is accomplished by using the information local to each automaton (performing in a way similar to a myopic eye) and by a set of transition rules that allow state transformations of the automata.

A two dimensional cellular automaton (Fig. 1) is a rectangular array of cells, each one having four neighbors (upper, lower, left and right) except at the edges of the array, in which case either three or two neighbors exist according to the position of the cell (corner or side, respectively). Each cell of the automaton changes state according to a set of transition rules which consider the state of the cell and those of its neighbors. More formally, for the set of states Q the corresponding δ will map a state quintuple into one of a set of possible states, i.e.

$\delta : QxQxQxQxQ \text{ ---- } 2^Q$

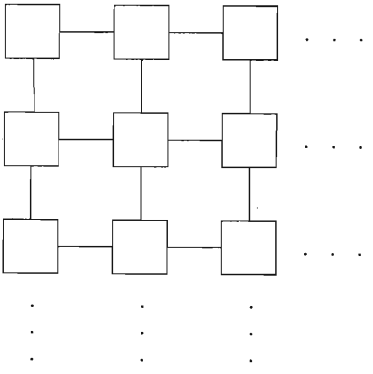


FIG. 1

Whenever the input state and the transition rules are known the final state is unique and cellular automaton is deterministic. Non-deterministic cellular automata may also be considered in which the transition function defines for each cell not a unique new state but a set of new states. Non-deterministic cellular automata are useful for the analysis of the computational complexity of algorithms in NP-completeness theory.

Each cell of the automaton performs a transition according to a rule which is common to all cells; such a state transition takes place simultaneously in all cells so that the operation mode is synchronous. The transition rule is equivalent to a single instruction executed on a iterative array.

A powerful model, which consists of many cellular arrays stacked to form a pyramid like structure, has been introduced in order to reduce the complexity of many algorithms. Each cell in a layer of the pyramid can communicate not only with its neighbors on the same layer but also with its parents on the next layer towards the apex of the pyramid and with its children on the next layer towards the base of the pyramid. In Tanimoto's pyramid, for instance, each cell is connected to four sons and one father so that in total it has thirteen connections involving elements of three successive layers.

If n is the base array diameter the height of the pyramid is O(log n); this ensures that many operations, such as counting etc., require a time O(log n), a good advantage over the O(n) time required by a simple

cellular array.

In cellular automata the interconnection scheme may also be of an irregular nature, allowing any cell to have a variable (but bounded) number of neighbors; such an automaton, which reflects a graph structure with the nodes corresponding to the cells and the arcs corresponding to the interconnections between cells, are called graph cellular automata. The compactness of a graph structure representing a set of intercommunicating cells may be measured by the diameter and the degree, which should be small and large respectively. The minimum number of nodes which must be traversed to reach a node from any given node is called the diameter of the graph while the maximun number of nodes adjacent to a given node is called the graph degree. For a detailed discussion of different possible groupings of processors both on the plane and in space see [32]. Many important graph properties can be evaluated by graph cellular automata in time proportional to the graph diameter. Many of the algorithms require visiting all the nodes of the graph; for example, in order to count the nodes themselves or to detect nodes which have a given label. This visit can be accomplished by fixing a node C as the starting one and by traversing in some order (depth or breadth) the other nodes, which are all at most O(diameter) away.

An interesting graph cellular model is one in which the configuration is not fixed but may vary according to the computational requirements. This is particular useful in image processing where the structure of the image expressed in terms of regions and relations between regions varies from image to image and also during computation.

COMPUTATION GRAPHS (DATA FLOW MODELS)

Let us now briefly introduce a model which represents a data driven computation. This model [13] considers a labeled directed graph G = (V,E) whose nodes v represent computation steps and whose edges e = (u,v) are the pathways for data and control.

More specifically, each arc e = (u,v) of the graph G has a quadruple (A,U,W,T) associated with it: A stands for the initial number of tokens (where a token is the data that enables computation) present on the arc; U and V represent, respectively, the number of tokens added to (removed from) the arc when the node v (u) executes its operation; finally, T is the number of tokens required to initiate the action of v.

The sequence of data produced by any one node proceeds until it is totally consumed by another node. A node is activated whenever all the necessary data have reached it. The flow of data drives the computation in FIFO style. This model is deterministic in nature; that is the results of the computation do not depend on the sequence of the computation steps among the enabled nodes. The weakness of the model is that it does not allow conditional branching. This model, although particularly suitable for recursive procedures,is rather cumbersome when handling input/output (and intermediate output) in

conventional programs.

As an example of a data flow graph modeling a computation, we may consider the case shown in Fig. 2 in which an algebraic expression is first "parallelized" and then executed on the nodes. Note that the same information (data) may be necessary at different nodes, thus requiring 'copying nodes" (denoted by "C" in Fig. 2). This example is originally from Gurd and Watson (1980) and may be also described by a quadruple with $A_{ij} = 0/1$, $U_{ij} = 1$,$W_{ij} = 1$ and $T_{ii} = 1$.

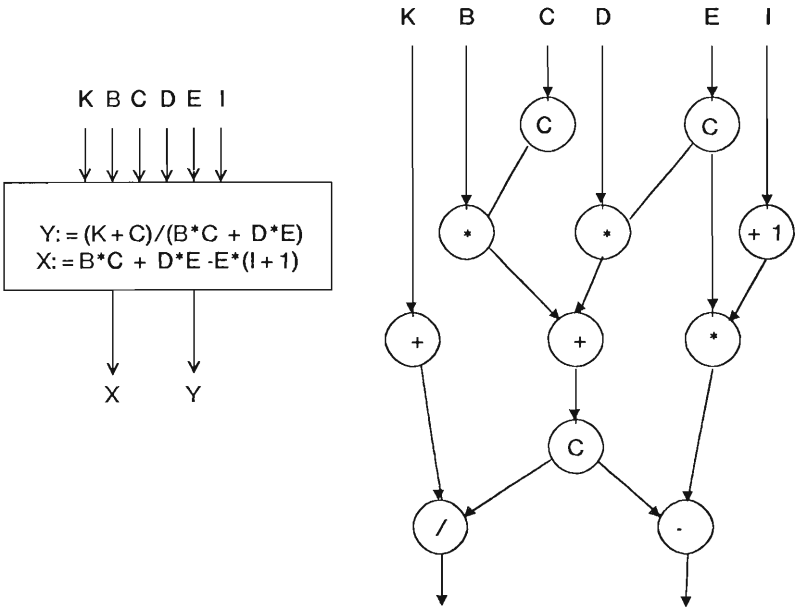


Fig. 2

As another example, consider the Fourier transform, a fundamental operation for image enhancement and restoration. The Fourier transform requires butterfly operations including sums and products of complex numbers as in the following expression:

$$X_R = A_R + B_R * W_R - B_I * W_I$$

$$X_I = A_I + B_R * W_I - B_I * W_R$$

$$Z_R = A_R - B_R * W_R + B_I * W_I$$

$$Z_I = A_I - B_R * W_I - B_I * W_R$$

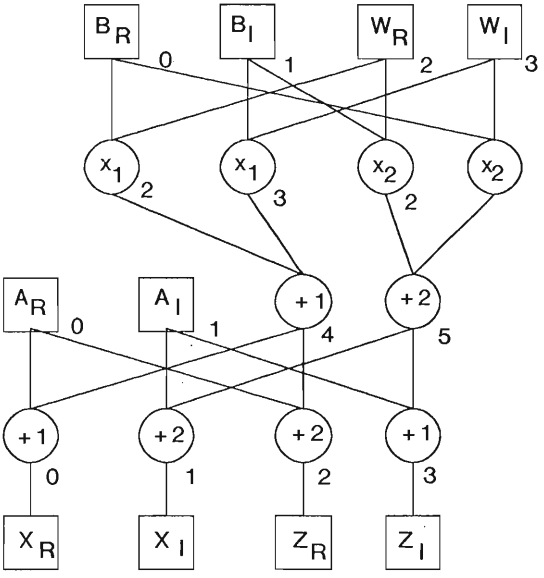


FIG. 3

The corresponding data flow graph is illustrated in Fig.3.

ARCHITECTURES FROM MODELS

We have reviewed two abstract models which aim at describing and clarifying the nature of the computation, as seen from the point of view of actions and control flow between actors. Such a model is generally represented by a graph-like structure which interconnects the different actors and shows the required inputs as well as the possible outputs from each actor.

Computational models have been shown to give some insight in the way in which information is processed within a system which has a number of processors acting simultaneously and where the configuration of the interconnections between the processors may change during the execution of the task.

Within the area of pattern recognition and image processing a number of machines have been developed which reflect the different computational approaches modelled in the previous section.

We will briefly describe representative machines for each one of the two classes so as to provide some insight into the architectural structure of such machines. Finally some typical algorithms that are well matched to these architectures will be discussed.

As we shall see, the comparison of these machines in terms of efficiency is extremely difficult since, in most cases, the computational efficiency is dependent on the nature of the algorithm. A recent approach to analyzing the time dependency of a class of algorithms on a given machine is included in [21]. The starting point is an algorithm grouping into "paradigms ". As an example, a paradigm considers $n = 2^r$ data items and a sequence of $r = \log n$ steps, each executable in parallel, with the property that at each step each operand interacts with an operand adjacent to it on a specified r-cube dimension.

Although some benchmarks may be defined in order to characterize the properties of the algorithms, no absolute performance evaluation may be obtained, due to the relevance of such issues as: data structures, input/output, local and regional operations, grey level manipulation, statistical operations, etc.

SIMD ARCHITECTURES

The class of so-called SIMD (Single Instruction Multiple Data) machines [9] can be modelled by the above mentioned cellular automata. In a SIMD machine elementary processors are all executing the same instruction on a set of independent data which is available to them. One processor, generally called host computer, acts as a controller of the whole system; it decodes the instructions and drives the elements accordingly. Although many years ago a class of spatial computers had already been suggested [33], it is only recently that the available technology made possible the construction of machines such as CLIP IV and DAP and ensured the feasibility of projects such as MPP, etc.

CLIP IV

The CLIP (Cellular Logic Image Processor) [6] family of parallel computers , built at University College London, represents the embodiment of a cellular automaton. CLIP IV, the latest of CLIP series, consists of an array of 96x96 processing elements, each of which is connected to eight (or, as an option, six) neighbors and has a memory capacity of 32 bits. An image from the video system is first converted and coded as a 96x96 image and then each pixel is stored in a processing element as a 6-bit binary word. Each processing element receives two independent inputs. The first input is the local value of the image element and the second is derived from the output of the neighboring cells. Two outputs are produced; one is sent to the neighboring cells and the other becomes the new image element value. A clock cycle of 2 sec is typical of the MOS technology used in the CLIP IV, but functions like propagation of a signal from a cell to the neighbors take much less. Due to the limitation in memory capacity and to a slow clock cycle only elementary binary operations are possible on CLIP IV. Moreover, this machine is suitable for problems where repeated elementary local operations are to be performed.

DAP

The Distributed Array Processor (DAP) [23] was not specifically designed for image processing applications, but is capable of handling a variety of computing applications with high performance. It consists of a two-dimensional array of 32x32 processors, each of which has a memory of 4k bit associated with it. It is in many aspects similar to CLIP IV except that only 4-connectivity between processors is allowed; it has a 5MHz instruction cycle and lacks facilities for feature extraction. Any image size or neighborhood size or number of bits per pixel may be selected due to the high memory capacity and to the general purpose nature of the 32x32 processors.

RECONFIGURABLE ARRAYS

The fixed array size and the fixed nearest neighbor connections can severely limit the range of processes that can be handled efficiently by cellular arrays. For some problems it is necessary for two arbitrary processors to exchange data; although this can be achieved by passing messages through a sequence of intermediate processors, this can slow down the process enormously. A reconfigurable array can improve the situation drastically but at a relatively high price.

Considering the array of processors as a graph in which the nodes correspond to the processors and the arcs to the interconnections between processors, a reconfigurable array may be represented as a graph whose structure dynamically changes during computation.

In a reconfigurable array, processors are not hardwired together into a fixed graph structure but each processor has attached a list of processors with which it can communicate directly [26].

Even if no existing machines implement a reconfigurable array, such an array may be simulated on an existing multiprocessor architecture such as ZMOB [24].

PIPELINED ARCHITECTURES

In pipelined systems processing elements are arranged along a pipe, with a data stream traversing each of the processing stages at fixed time intervals. Each processor is designed to perform a specific function. After an initial delay due to the propagation of data through the functional units, called set-up time, data are output at the same rate they are entered. In a broad sense, a pipelined machine can be considered to operate as in data flow model, but instead of the data driving the process here an external clock drives the computation; the clock is long enough to ensure proper propagation of data.

Pipelined machines are particularly appropriate when a computational process can be decomposed into several subprocesses [5] each of which can be executed efficiently on specially tailored stages which operate simultaneously with others on the same data stream, with the data arriving to each processor with a constant delay. For a review of pipeline systems see [22].

Pipelined architectures represent one of the best solutions to image processing timing problems. The

main advantage of the pipeline over the array architecture is in the low number of processors necessary to perform a given task; in an array as many processors as there are pixels in the image are required, which is often prohibitive in practice. In pipeline processors the input pixels are arranged in raster scan format; in one cycle a pixel is input into a processor while another pixel is output by the same processor to the next one in the pipe.

CYTOCOMPUTER

The Cytocomputer [16], built at the Environmental Research Institute of Michigan, strongly employs the concept of pipelining. It consists of a pipeline of programmable processors, called stages, specially designed for real-time image processing. In this machine provision is made for the implementation of a 3x3 neighborhoods by means of suitable delays, and each processing task is carried out on the pixels flowing from the image. One of the most important features of the Cytocomputer architecture is that the input/output of images is sequential and this is well adapted to input/output devices for image data.

PYRAMIDS (CONES)

New ideas on the structure of parallel computing systems for image processing have led to the design of machines which combine both the concepts of SIMD and pipelining. These machines are hierarchical systems which consist of layers of arrays which are stacked and where the computing cells have both lateral and vertical connections. The layers, generally, are tapered starting with a large base and ending with a vertex (apex), thus resembling a pyramid (cone) structure. (Recent contributions in this area may be found in [17, 7, 3].

An important approach to fast computation on images [31] was called the recognition cone. Basically, it is a layered system of elementary processors (cells) each layer being structured as an array. At the bottom layer, the base array inputs information from the outside. After a number of layers, at each of which the number of cells is reduced by a constant factor, the apex of the cone is reached, where the result of the computation is output. In each layer a transformation can be accomplished in parallel: all processors execute the same instruction in SIMD mode. Different layers can execute different transformations. The control structure is such that no specific layer is privileged during processing. This multilayered system can be considered as pipeline of processors. Data flows through the layers; at each layer an operation on the data is performed and the output of this operation becomes the input for the next layer.

In a broad sense, the high image definition in the base array corresponds to foveal vision, while decisions for pattern recognition are made at the apex which simulates cortical processing of visual information. Much of the work is based on the fact that not all information present in the input image is significant for solving a specific task, as shown in Kelly's paper on "planned" search for edge detection [14].

This general model has also been considered by Levine [15], Riseman [18] and Tanimoto [28]. Information can take place downwards as well as upwards and, in some cases, also laterally. A processor should be able to access information from its own memory (small at the base and larger memories towards the apex) as well as from its neighbors on three successive layers. Such a structure has a number of advantages: the arrays process a large amount of data simultaneously; the pipeline further speeds up the processing; input and output can be done very efficiently. This architecture may seem expensive considering the increase in the number of processors when compared to a single array: however, this is not so, since for a pyramid in which each layer has 1/4 as many processors as the preceding layer, only one third more processors is required. Pyramids have been simulated on a sequential computer [29]. The fast development of VLSI technology plus the decrease in memory cost will soon enable the practical construction of machines of the cone-pyramid type.

DATA FLOW ARCHITECTURES

Data flow architectures are the natural implementation of the computation graph model. These machines allow each instruction of a program to be executed as soon as all its operands are available to it. There no notion of a single controller. Thus a highly concurrent computation can be carried out by these machines by assigning all the ready instructions to different processing units. This is a very important feature since it allows one to compute complex algorithms in reasonable time.

The concept of data-driven computation is old, but only in recent years architectural schemes employing it have been developed and processors with data driven execution have been built. At MIT a static data flow architecture is under development and its potential performance for large scientific applications is being investigated [4]. Also at MIT another data flow machine which allows dynamic procedures is being built. Most of data flow projects use the mechanism shown in (Fig. 4).

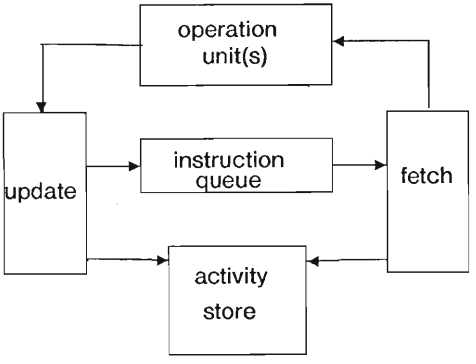


FIG. 4

The program, in the form of a data flow graph, is stored in the activity store. As soon as an instruction is ready for execution, its address is entered in a queue unit (a FIFO buffer store). The first instruction on the queue is read by a fetch unit and sent to an operation unit, where the action specified by the instruction is performed. The output produced by the operation unit is then sent to an update unit which enters the results into the operation fields of some specified instructions. The update unit also tests whether the instructions which recieved the data are now ready for execution and, if so, enters their addresses in the instruction queue. The degree of concurrency present in a program can be measured by the number of entries present in the instruction queue. If a sufficiently large number of operation units are available, the parallelism can be fully exploited.

A project of the Nippon Electric Co., called the Template-Controller Image Processor (TIP) [12], combines the data driven computation concept with the pipeline technique. In this way TIP overcomes the disadvantages of the pipeline processor, that is (a) the lack of flexibility to adapt to different processing requirements and (b) the need for all processing stages to be synchronized by the same clock to avoid improper propagation through the pipe.

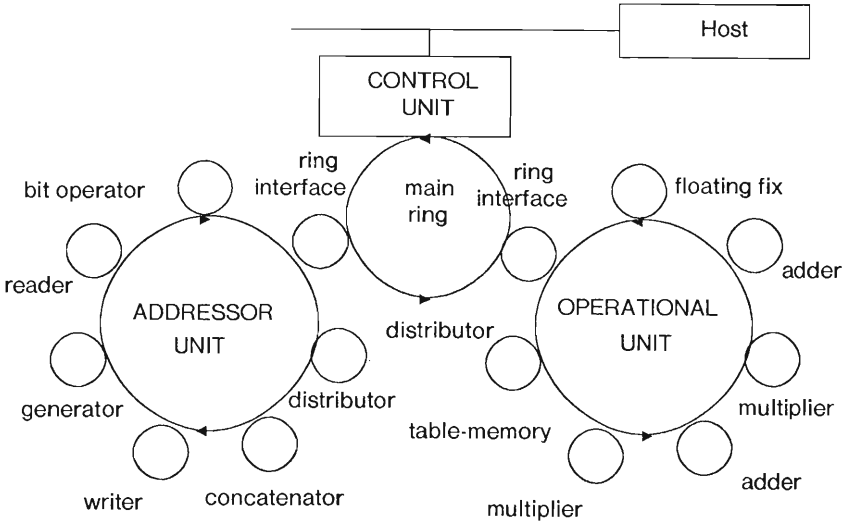


FIG. 5

TIP is essentially composed of three rings (Fig. 5): 1) a ring of processing units, each implementing a specific function; 2) a ring called the addressor unit, containing generator, reader, writer, bit-operator, and distributor; and 3) a main ring which interconnects the other two. The addressor unit reads data

from the memory and provides them, via the main ring, to the processing units; it also writes back into the memory the output data from the processing units. During the initialization phase, the addressor unit receives from the host computer a set of templates which are defined externally. After the interpretation, these templates are sent to the processing units to control data flow and image definition.

Data flows through the rings,each data item bearing an identifier and one or more destination flags. When the data reaches the right processor, i.e. the processor indicated by its destination flag, an operation is performed according to the template the data identifier matches; the original data disappears unless it contains other destination flags.

The performance of the TIP system on a typical image processing task, the Fourier transform, is evaluated in terms of the number of data transfers through the ring bus between processing modules.

A data flow machine is effectively programmable and its base language is a data flow graph.

MIMD ARCHITECTURES

The class of MIMD (Multiple Instruction Multiple Data Stream) machines, which are more versatile and therefore have higher inner complexity, cannot be modelled by the above mentioned models. The asynchronous actions of the set of the processors and the control of the system, which is defined by a set of precedence rules, mutual exclusion, and other requirements, are not considered in these models. Some aspects of these machines have a counterpart in the Petri net model [19], but MIMD cannot be considered to directly descend from Petri nets. A MIMD system may run asynchronously with each processor executing a different and not necessarily equally long sequence of instructions; a host machine will load all programs and arbitrate the data flow between the processors, common memory, and individual memories. The asynchronous mode stems from the different execution times of the processors thus allowing greater operational flexibility. The communication of partial results between processors is established according to precedence rules in order to avoid indeterminacy.

ZMOB

This multiprocessor machine uses a bus, called the conveyor belt, which allows a fast and flexible communication between processors in a byte parallel mode. Built at the University of Maryland, it consists of 256 Z80A microprocessors, each of which has a local memory of 65K bytes and a clock cycle of 2.5 microseconds. Thus the high number of 100 MOPS on 8-bit data is possible on ZMOB.

The processors may execute arbitrarily complex programs which may be either identical or different, due to their high memory capacity. When performing operations on images, each processor may work on a different subimage; this generates the problem of choosing a partition of the image in such a way that both a limited number of interactions between processors occurs and the input of data is not

slowed down by requiring a special format.

PYRAMIDAL ARRAY NETWORK

The recent pyramidal array network proposed by Uhr et al. [30] is a combination of SIMD and MIMD machines, and is particularly attractive for many applications such as image processing, pattern recognition, and data base implementation. Furthermore, this structure appears useful for a variety of problems which can be solved by a divide and conquer technique and for situations which require one to examine several alternatives, etc.

The pyramidal array network consists of large arrays of very simple processors which operate in SIMD mode and smaller networks of more powerful processors which operate in MIMD mode. These arrays and networks are arranged to form a pyramid-like system, which, starting from a base, a large array of 1-bit processors, converges to an apex. At each layer of the pyramid the number of processors decreases while their power and flexibility increases. At the higher levels the processors have their own controllers, thus allowing asynchronous operation.

Since the network operates in MIMD mode it might be useful for a processor to read from distant memory locations which are normally not addressable. A partition of this MIMD system is therefore needed to solve both memory access and contention.

MATCHING TASKS AND ARCHITECTURES

For a good match between the computation to be achieved and the system achieving it, not only must there be a logical correspondence between the control points on the model and those on the machine, but also the information should reach the processors in the shortest time, so as to use all the active processors most of the time.

In the case of image processing systems where the amount of data is remarkably high (consider, for instance, a Landsat image taken at six different wavelengths, with an 8-bit grey scale on a roughly 2000 by 3000 pixel image) a high throughput rate is mandatory. Data from the pixels should be available to the processors at the clock rate of the machine, thus avoiding time delays due to the interprocessing communication and memory contention.

The choice of a best architecture for a class of problems should take into account both the required computation and the exploitation of the system resources.

In this last section we will review some classes of algorithms for image processing and their well matchedness to the architectures discussed above.

SIMD machines are well suited for local processing of images in parallel. By a local operation is

meant an operation which computes the value of a pixel in an image in terms of its own value and the values of its neighboring pixels. Most of the preprocessing tasks ,such as smoothing, thinning, etc. involve local operations and can be efficiently performed on a SIMD machine where each processor is acting on a single pixel and there is a fixed interconnection scheme between processors. Also statistical pattern recognition on images, which uses matrix multiplication, well matches this class of machines. In the paper by Dyer and Rosenfeld [8] the cellular automaton model is used for the analysis of the computational complexity of a number of image processing and recognition tasks. For a memory capacity of each cell proportional to the log of the number of pixels in the image, typical results show that the execution times are of the order of the diameter of the array (one task, for instance, was area computation of a connected region). Closer to the machine level is the analysis in [2] of the computational cost of some typical image processing algorithms when performed on a SIMD machine (CLIP IV). In this paper the performance of the parallel versus sequential implementation of the same tasks is evaluated in terms of the ratio of the number of clock cycles of the machine, showing that there is still a gain in using CLIP IV even for problems which strongly use matrix arithmetic.

Region level operations [26] are involved in many algorithms that perform higher level tasks on images. These operations differ from the local ones in an essential way: pixels remain fixed during the computation, but regions, resulting from the segmentation of an image, may vary in size and number both during computation and from image to image. Thus fixed interconnection schemes for processing elements do not match this class of algorithms; a reconfigurable array could be the optimal solution. Processing at the region level may involve merging or splitting of regions or matching configurations of images between two regions or with an image model.

An increasingly important area in image processing is the analysis of time-varying images; applications in this field include meteorology, biomedicine, and industrial automation. One of the most fundamental problems in dynamic image analysis is the tracking of objects from image to image in order to evaluate the motion of the objects, if rigid, or to describe how the shape of the object has changed [1]. The amount of data here is dramatically high and a general-purpose computer usually is inadequate for real-time processing. With a multilayered pyramid considerable speed-up can be obtained; images, input at the bottom layer of the pyramid, flow through the layers upwards, and while a layer is operating on an image the preceding layer is operating on the next image.

While SIMD machines are suitable for image processing applications, where repetitive operations are to be performed, MIMD machines appear more appropriate for pattern recognition and image understanding. The goal of pattern recognition is to describe an image in terms of the objects present in it, their features and their interrelationships, and this involves more complex high level operations. A typical example is given by a matching problem where an instance of an object or, more generally, a fixed configuration of objects is to be detected in an image. This problem arises in many applications involving satellite images, biomedical images, etc. Several approaches have been proposed for solving such problems; some of them are based on relaxation techniques [25], and others on structural

techniques [10]. When using a MIMD machine for this task, different processors may be assigned to different objects and then the search for a match occurs independently on each processor; alternatively, more than one processor may be assigned to an object with each processor implementing a different part of the search.

The problem of recognizing an object in a picture can also be efficiently solved on a pyramidal array network, which can be considered to model both retinal and cortical information processing. In this case the base of the pyramid is used to input data from the external environment, as previously mentioned; a few successive layers,operating in SIMD mode, perform simple preprocessing operations such as the extraction of elementary features, contours, etc; then the higher layers perform clustering and classification on objects in the picture using different groups of processors, each operating in an asynchronous way.

Since a number of image processing machines are currently available, it is important and useful to be able to introduce some objective standard parameters in order to evaluate them. In sequential computers processing images the instruction time of the machine may be considered as a starting point or, even better, the number of elementary operations performed on a single picture element per unit of time (pixop for short). For a discussion of the ingredients to be used in assessing the merits of parallel image processing machines see [20]. Furthermore, the programming effort required to run a correct sequence of instructions on a machine also has to be considered in machine evaluation. Finally, typical tasks should be discussed, so that, when different machines perform them, a relative time ordering can be established. Typically, image input/output, local operations on pixels, histogramming and thresholding are the most frequently used components of a general image processing package, and so may be useful in defining benchmarks. It is important to note that pixops have increased by seven orders of magnitude in the last twenty years and that, for a project at Goodyear Aerospace called MPP [27], the expected pixop rate is of the order of 10^9. Although a benchmark has been suggested in [20] (a program for detecting, sizing and counting tissue cells) it is also pointed out in the same paper that some tasks can be better performed on certain machines than on others, quite independently on their pixop rate.

Conclusion

Models have an impact both on the understanding of the computational process and on the design of future computer architectures. Such new arhitectures will be better matched to the problems they want to solve and therefore less time will be spent in data search, housekeeping, instruction loading, etc and more time will be used for real processing of data. At the same time the implementation of algorithms will be easier since the resources will be particularly tailored both to handle large data streams and for parallel computations on subsets of the data. Conversely, these new machines will require complex operating systems, the design of new high level languages in order to ignore the hardware details of the machine (as seen from the programmer's point of view) and guarantee transportability. As in many

other situations, technological improvements will increase computer efficiency; they will also generate new problems of a higher level of complexity but, is this not the only way to proceed?

ACKNOWLEDGMENTS

We want to thank A. Rosenfeld for useful suggestions and his critical reading of this paper

References

1. J.K. Aggarwal, L.S. Davis, W.N. Martin . "Correspondence Process In Dynamic Scene Analysis ." *Proc. IEEE 69* (1981), 562-572.

2. L.P. Cordella, M.J.B. Duff, S. Levialdi. "An analysis of the Computational Cost in Image Processing." *IEEE Trans. on Comp. 27* (1978), 904-910.

3. P.E. Danielsson, S. Levialdi. "Architectures for Pattern Recognition and Image Processing." *Computer* (1981).

4. B. Dennis. "Data Flow Supercomputers." *Computer 1* (1980), 48-56.

5. R.J. Douglass. A Pipeline Architecture for Image Segmentation. Personal Communication

6. M.J. Duff. Review of the CLIP Image Processing System. AFIPS Conf., 1978, pp. 1055-1060.

7. M.J.B. Duff, S. Levialdi. *Languages and Architectures for Image Processing.* Academic Press, 1981.

8. C.Dyer, A. Rosenfeld. "Parallel Image Processing by Memory Augmented Cellular Automata." *IEEE Trans. on PAMI 3* (1981), 29-41.

9. M.J.Flynn. "Some Computer Organization and their Effectiveness." *IEEE Trans. Comp. 21* (1972), 948-960.

10. K.S. Fu. *Syntactic Methods in Pattern Recognition.* Academic Press, 1974.

11. G. Geymonat, G. Gioiello. Modello. Enciclopedia Einaudi, 1980, pp. 383-422.

12. S. Hanaki, T. Temma. Template-Controlled Image Processor (TIP) Project. Multicomputers and Image Processing, 1982, pp. 343-352.

13. R.M. Karp, R.E. Miller. "Properties of a Model for Parallel Computations: Determinacy, Termination Queueing." *SIAM Journal of Applied Mathematics 14* (1966), 1390-1411.

14. M.D. Kelly. Edge Detection of Pictures by Computer Using Planning. Machine Intelligence, 71, pp. 397-409.

15. M. D. Levine. A Knowledge based Computer Vision System. Computer Vision Systems, 78, pp. 335-352.

16. R. M. Lougheed, D.L. McCubbey, S.R. Sternberg. Cytocomputers: Architectures for Parallel Image Processing. Workshop on Picture Data Description and Management, 80, pp. 281-286.

17. M. Onoe, K. Preston, A. Rosenfeld. *Real Time Parallel Computing.* Plenum Press, 81.

18. C.C. Parma, A. R. Hanson, E.M. Riseman. Experiments in Schema-Driven Interpretation of a Natural Scene. NATO ASI on DIgital Image Processing, 1981, pp. 449-510.

19. J.L. Peterson. "Petri Nets." *Computing Surveys 9* (1977), 223-252.

20. K. Preston. Comparison of Parallel Processing Machines: a Proposal. Languages and Architectures for Image Processing, 81.

21. F.P. Preparata, J. Vuillemin. Area-Time Optimal VLSI Networks Based on Cube-Connected Cycles. Tech. Rept. 875, University of Illinois at Urbana, 80.

22. C.V. Ramamoorthy, H.L. Li. "Pipeline Architectures." *Computing Surveys 9* (1977), 61-103.

23. S. F. Reddaway. The DAP Approach. Infotech state of the Art Report on Supercomputers, 79, pp. 309-329.

24. C. Rieger, J. Bane, R. Trigg. ZMOB: a Highly Parallel Multiprocessor . Workshop on Picture Data Description and Management, 81, pp. 298-304.

25. A. Rosenfeld,R. Hummel, S.W.Zucker. "Scene Labelling by Relaxation Operations." *IEEE Trans. Systems, Man and Cybernetics 6* (1976), 420-433.

26. A. Rosenfeld, A. W. Wu. "Parallel Computers for Region-Level Image Processing." *Pattern Recognition 15* (1982), 41-51.

27. D.H. Schaeffer. Massively Parallel Information Processing for Space Application. AIAA Second Conf. on Computers in Aerospace, 1979, pp. 284-286.

28. S. Tanimoto. Programming Techniques for Hierarchical Parallel Image Processors. Multicomputers and Image Processing, 81, pp. 421-429.

29. S. Tanimoto. Towards a Hierarchical Cellular Logic: Design Considerations for Pyramid Machines. Tech. Rept. 81-02-01, University of Washington, Seattle, 1981.

30. L. Uhr, L. Schmitt,T. Hanrahan. Cone-Pyramid perception Programs for Arrays and Networks. Multicomputers and Image Processing, 81, pp. 179-191.

31. L. Uhr. "Layered Recognition Cone Networks that Preprocess, Classify and Describe." *IEEE Trans. Comp. 21* (1972), 758-768.

32. L. Uhr. *Algorithm-Structured Computer Arrays and Networks.* Academic press, 1983.

33. S.H. Unger. Pattern Recognition and Detection. IRE, 1959, pp. 1737-1752.

34. *Workshop on 'Elaboratori Paralleli e Calcolo Scientifico'.* IBM - Italia, 1982.

35. . Workshop on Non-Conventinal Computers for Image Processing. Multicomputers and Image Processing. K. Preston, L. Uhr (edits) - Academic Press 1982.

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

INTERACTIVE SOFTWARE SYSTEMS FOR COMPUTER VISION

Klaus Voss and Peter Hufnagl

Laboratory for Automated Microscope
Image Analysis
Institute of Pathology
Humboldt University Berlin, DDR-1040 Berlin
German Democratic Republic

Reinhard Klette

Image Processing Laboratory
Friedrich Schiller University
DDR-6900 Jena
German Democratic Republic

This paper gives an overview of the main directions of program system development in computer vision, where special emphasis is paid to interactivity. A systematic representation of interactive computer vision systems is given. Brief information about 39 different software systems is listed in an Appendix.

INTRODUCTION

Among the different areas of information processing, computer vision is characterized by some typical features developed during its progress over the last 25 years. Neglecting special hardware requirements such as on-line scanning systems, array processors, or display systems, these typical features of computer vision may be described as follows:

- Images have to be processed, i.e., large amounts of two-dimensionally structured data, e.g., 10^3 to 10^7 elements (image points), are transformed by certain operations defined on images.
- Images have to be analyzed, i.e., the primary information of 10^3 to 10^9 bits has to be reduced, even to a one bit message (yes-no) in the extreme case.

In general, in both types of tasks no well-suited theories are available to allow straightforward problem solutions. In fact, the recent practice in computer vision (image processing and image analysis) is dominated by heuristic methods used in the design of practical, relevant algorithms. These heuristic methods include the experiences of experts which introduce goal oriented components into the otherwise unrestricted trial-and-error procedures.

Therefore, in designing computer vision algorithms the following, periodically repeated sequence of steps is typical:

(i) new design or modification of an algorithm, or of some operations within an algorithm,

(ii) transmission of these conceptual ideas to the computer, and performance of the related instructions by the computer,

(iii) evaluation of the results which may be available in the form of an image, as a graphic, or as a data file.

The interaction between human and computer has to be realized at step (ii) of this sequence. To ensure high efficiency in the process of designing computer vision algorithms, the communication between human and computer should be as direct as possible, i.e., should not suffer from unnecessary repetition, or from other unimportant conversation with the system, and should be supported by high computation speed. Taking into account the currently available computer vision hardware (up to now, final solutions or concepts for specialized hardware structures for distributed computer vision do not exist), these requirements have to be satisfied primarily by software tools.

This paper is organized as follows: In Section 2 general components or features of software systems for computer vision are listed allowing a qualitative judgment of these systems. In Section 3 a general orientation to high-level languages in computer vision is sketched. In Section 4, a systematic representation of interactive computer vision systems is given. Finally, an Appendix contains brief information on 39 different software systems. For information and references to software systems in computer vision, see Rosenfeld (1983).

COMPARISONS OF SOFTWARE SYSTEMS FOR COMPUTER VISION

Any software system contains problem-oriented and system-oriented components. The problem-oriented features are related to the application area. Obviously, for analyzing printed patterns different algorithms or methodological steps are required than for processing of multi-spectral pictures, for example. On the other hand, the system oriented components are essentially determined by the typical features of computer vision such as those mentioned above, and these components may be found in all application areas to a

certain extent.

To compare different software systems, Preston (1979) has considered the following grouping of problem oriented instructions.

- Utilities: identifiers, executives, formatters, I/O commands, test pattern generators, help files.
- Image Display: CRT, hard copy, interactive graphics.
- Arithmetic Operators: point, vector, matrix, complex number, Boolean variables.
- Geometric Manipulation: scaling and rotation, rectification, mosaicking and registration, map projection, gridding, and masking.
- Image Enhancement: noise removal, Fourier analysis and other spectral transforms, power spectrum, filtering, cellular logic.
- Image Analysis: histogramming, statistical principal components.
- Decision Theoretic: feature selection (training), classifying (unsupervised), classifying (supervised), evaluation of results.

This listing of problem oriented system components may be considered to be the fundamental instruction set of computer vision, cp. /42/. On the other hand, in Voss et al. (1983) system oriented components were studied, and the following list was used.

- Modularity: For a given library of subroutines, these subroutines may be called individually, or may be connected to main programs.
- Interactivity: A working method is guaranteed allowing the direct input of commands (dialogue system), or supporting program development without explicitly reverting to the computer's user system.
- Simplicity: The command language (dialogue system) or the programming language is easy to learn and contains mnemonic terms.
- Efficiency: Fast reactions of the system are guaranteed. Organization tasks (file processing, editing, etc.) as well as computer vision operations run with high speed.
- Reliability: The system works reliably. Besides fault-freeness of the subroutines, the fault-free work of the user is assured. The destruction of the system is generally prevented, and a detailed analysis of user inputs is guaranteed.

- Flexibility: The modular system allows the design or arbitrary programs; access to a single image point, bit, or character is possible.
- Variability: In the modular system, simple possibilities for altering or extension exist, i.e., extensive sequences of operations may be treated as a new system unit, and may be integrated into the total system.
- Portability: The system may be transferred to, or implemented on, another hardware system in a relatively simple way.

The fulfillment of these partially contradictory demands "as optimally as possible" represents a very complicated task. An optimal system should take away all unnecessary organization tasks from the user, and should allow the user to concentrate on the intrinsic tasks of computer vision. The only demand all systems have to satisfy is modularity. This is enforced by the great complexity of operations in computer vision, e.g., extraction of shape features, generation of texture features, Fourier-filter operations, or image model comparisons.

Besides the modularity requirement, the other listed requirements are satisfied by existing computer vision software systems to different extents. In the present paper, these software systems will be discussed, and special attention will be paid to the demand of interactivity. A "genetic" organization scheme was conceived relating to the historic progress, more or less. This systematization attempt is based on the study of more than 50 software systems.

HIGH LEVEL LANGUAGES IN COMPUTER VISION

In principle any programming language may be used for the needs of computer vision. The problem consists in the linkage of simple modules (subroutines, algorithms, operations, etc.) into a program. For this, three different possibilities exist.

- Standard procedures: These procedures are integrated parts of the system or the language. The user may apply them without any programming or loading. The disadvantage of these procedures consists in the fixed status which hinders optimal, problem specific usage.
- Internal procedures: Within a given main program the possibility is provided to define program parts which will be used as

subroutines or functions within the main part of the program. Internal procedures will be formulated within the context of the main part of the program, and will be translated together with the main part of the program. The disadvantage of this approach is the possibility that short main programs of 100 to 1,000 instructions may be inflated by complex procedures to 10,000 to 100,000 instructions.

- External procedures: These procedures may be designed and translated independently of the main program, and the translated versions may be stored in external memories in case of lack of central memory. In general, before starting a main program a loading and/or linking process is required. The advantage of external procedures is the possibility of short main programs, even by writing external procedures in assembly language.

Any high level language includes standard procedures, and nearly all of them include internal procedures, too. Modern versions of high level languages permit calling of external procedures, but this is not the case for the original versions of these languages such as ALGOL 60, BASIC, PASCAL, APL, or PPL. However, in FORTRAN the use of external procedures (assembler routines) was considered at the very beginning. This is one of the reasons that FORTRAN found relatively widespread use in computer vision.

Commercial computer vision systems make use of high level languages and subroutine libraries. This approach is obvious and leads straightforwardly to results. Within the last 5-10 years, disadvantages of this approach have been recognized connected with the difficulty of designing new procedures and programs. Any algorithmic change has to be realized by calling the editor of the user system. Then, the new procedure or program has to be compiled, and all required program parts have to be loaded and linked. Finally, the program testing may take place. The use of editing, compiling, loading, and linking leads to high expenses in time and organization. (By our own experience, for a given FORTRAN program of 15 lines the simple operation of replacing one addition by one subtraction requires 90 seconds at least if a disk memory is used.) The second disadvantage of high level languages is given by the fact that in nearly all cases no direct communication between human and computer is possible because procedures cannot be used as commands.

Only by installation of a procedure within a program frame is its direct execution possible by using a start command, see Kupka (1980).

The way out as chosen by many computer vision groups consists in turning to interactive working methods which frequently involve abandoning the available high-level language, or even designing one's own more or less comfortable interactive user system. Existing interactive programming languages such as BASIC or LISP do not offer flexibility and variability as needed .

INTERACTIVE SOFTWARE SYSTEMS

In the following a systematic overview of interactive software systems in computer vision is given. The subsequent seven stages of development are bottom-up; any state contains the possibilities of the preceding stages, in general. This systematic approach reflects the historic development of interactive software systems in computer vision; see Figure 1.

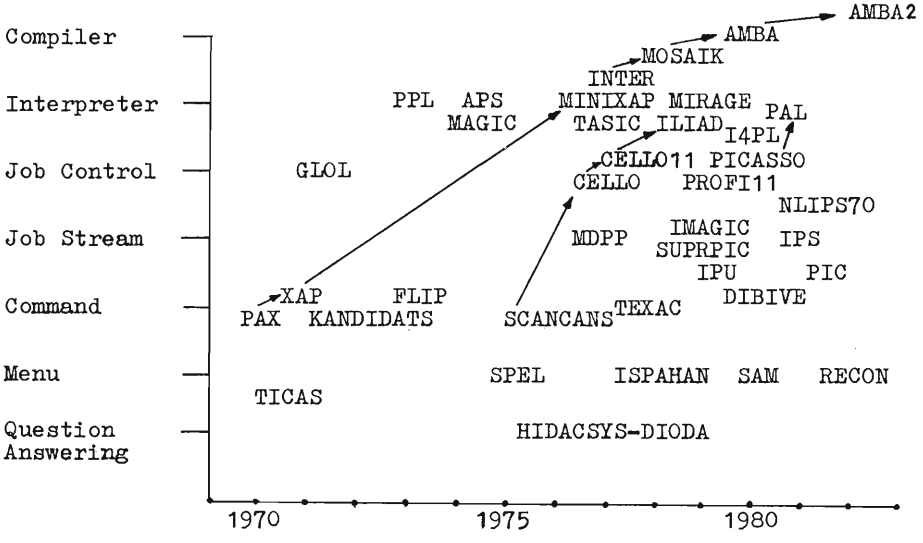


Figure 1
Historical development of interactive software systems in computer vision

execution. The requirements for text processing are more complex than in the case of command systems, and a supervising partial system is required containing instructions, such as "formulation of a job stream," "storing of a job stream under a certain name," and "call and start of job stream."

A job stream system may work in execute mode (immediate realization of input commands) or in program mode (execution of job streams). Examples of job stream systems are MDPP (1978), SUPRPIC (1980), IPU (1980), IMAGIC (1981), IPS (1981), and NLIPS-70 (1981). Though they have many advantages, job stream systems possess the disadvantage of realizing linear sequences of commands only. No branching, looping, or internal procedures are possible.

JOB CONTROL SYSTEMS

A job stream system may be extended to a job control system if control commands are possible in job streams too, allowing non-linear processing by the use of labels, jumps, loops, or internal procedures. For job control systems, languages with well-developed syntax are required so that we may already speak of programming languages in this case. The (in general non-linear) command processing is realized by interpretation. All commands possess a procedure-like structure, i.e., they are formed by names of procedures and lists of parameters. This is true for arithmetic operations too; see CELLO, GLOL, or PICASSO. Because extensive programs may be written within the job control system language, the use of comments is normally allowed in these systems.

INTERPRETER SYSTEMS

The syntax of system languages is made very rich by the possibility of formulating arithmetic operations and comparisons through independent instructions in common mathematical notation. Thus, the system language has taken the final step to being an autonomous high level programming language. The realization of single commands in execute mode, or of programs in program mode, is done by interpretation. For that reason, these systems are called interpreter systems. By using an interpreter, the execution of a program written in the source language takes place command by command, without first compiling or assembling the program as a whole; see Gould (1977).

For example, BASIC and LISP are interpreter languages. In the sense of negation of negation, we have reached our starting-point again, which was characterized by high-level languages and their use in modular systems. But interactivity is the new feature, which seems to point out that interpreter systems are advantageous for computer vision. (Indeed, the efficiency of program execution is low.) The question now is, why the computer vision community does not use interpreter languages such as BASIC, APL, or LISP without alterations?

The main reason is given by the restricted possibility of using external procedures within these languages. The use of internal procedures is forbidden because of the frequently large number and size of the necessary algorithms. Furthermore, the standard procedures of these languages are not written for the needs of computer vision. But an extension of the set of admissible standard functions is not possible without alteration of the language definition, and of the compiler for the given high level language.

The only way out is the creation of a new, autonomous language on the basis of existing interpreter languages, by extension of the set of admissible standard functions reflecting the special interests of the user. In this sense, some of the most recent computer vision software systems may be considered to be extensions of standard high-level programming languages; see Table 1.

System	Basic Language	Literature
ILIAD	PASCAL	Eriksson (1982)
MINI-XAP	LISP	Luczak (1977)
APS	APL	Reeves (1979)
MIRAGE	BASIC	Wade (1980)
PPL	PL/I	Kruse (1980)
TASIC	BASIC	Nawrath (1979)
MAGIC	BASIC	Taylor (1978)
I4PL	LISP	Granlund (1982)

Table 1

List of interpreter systems

COMPILER SYSTEMS

In general, interpreter systems satisfy the software system demands of users for the needs of computer vision. The only disadvantage is given by the low operation efficiency caused by the interpretations. While this disadvantage, in general, is not observable in execute mode, it will be realized to be troublesome in the case where a program recognized to be well-suited, and having many loop structures, has to run very often in daily practice. A way out may be to compile the program as a whole before running. In this case, between loading and starting of the program, a supervisory command "compile" is required, making necessary the control of a compiler system by its own user system. In these compiler systems no external procedures are included; thus loading and linking is not required. For computer vision essential procedures may be included as standard procedures in the compiler system language. Exchange or extension of the set of standard procedures can be done without altering the compiler, i.e., without altering the user system.

In the literature only the compiler systems INTER (1979), MOSAIK (1979), AMBA (1981), and AMBA2 (1983) were accessible to us. But, as illustrated in Figure 1, in the near future, an increasing number of compiler systems for computer vision can be expected.

APPENDIX: Software Systems

AMBA = Automatisierte Mikroskop-Bild-Analyse

Compiler system, Humboldt University Berlin /39,49/, developed since 1979 for biomedical applications (histology), Computer EPR 1100 (48k), own language LAMBA, system and 100 procedures in assembler. The complete system, including editor, compiler, procedures, parameter lists, as well as sections for source programs and translated programs, requires 20k memory. The language allows comments, labels, jumps, internal procedures, and arithmetic for fixed point variables and indexed variables.

AMBA2 = AMBA system 2

Compiler system, Humboldt University Berlin /50/, developed since 1982 for histology and other biomedical applications, Compter EPR 1100 (48k), improved language LAMBA, system and 110 procedures

in assembler require 24k. Additional arithmetic for floating point numbers.

APS = Array Processing System

Interpreter system, developed since 1975 at McGill University in Montreal for PDP-15, and at Purdue University in West Lafayette for PDP-11/45 /36/. System in FORTRAN, 54 procedures in FORTRAN or assembler. The system language is APL allowing interactive work in execute mode as well as interpreter programs.

CELLO

Job control system, developed since 1975 at the University of Uppsala for cytology /19/, Computer PDP-8/e (32k). System in FORTRAN, 200 procedures in FORTRAN and assembler. The command language allows comments, labels, jumps, and internal procedures. No arithmetic expressions. Execute mode and interpretative work both possible in program mode.

CELLO11

Improvement of CELLO, since 1977, for PDP-11/55 (128k). System in PASCAL, 150 procedures in FORTRAN and assembler, indexed variables /4/. New procedures may be included into the external library.

DIBIVE = Digitale Bild-Verarbeitung

Command system, Institute for Radiation Protection, Neuherberg, Munich /37/, developed since 1979 for cytology, Computer SIEMENS p330. The System consists of an interpreter and about 300 procedures written in FORTRAN and assembler. Single commands may be input interactively and will be interpreted.

DIODA

Question-answering system, developed since 1978 at the University of Leiden for cytophotometry /51/, Computer PDP-11/60. DIODA is a dialogue system completly written in FORTRAN and requires about 6k. It allows the interactive call of six problem specific subprograms.

FLIP = Free-form Language for Image Processing

Command system, developed since 1973 at Los Alamos Scientific Laboratory for materials research (radiography) /20/, Computer CDC 6600 and CDC 7600, procedures in FORTRAN and assembler, interactively callable commands with parameters, or empty parameter list.

GLOL = Golay Logic Processor Operating Language

Job stream system, developed since 1971 at the University of Pittsburgh on computer VARIAN-620i /31,32/ with application to white blood cell analysis, about 40 procedures. The interpreter language is similar to BASIC and allows interactive calling of single commands. In this language loops and branches are possible (DO V(N1), V(N2), V(N3)...STOP, or IF...ELSE...STOP).

HIDACSYS

Question-answering system, developed since 1976 at the University of Leiden for cytophotometry /45/, PDP-11/10 (24k), simple dialogue system for calling subprograms, these call for required parameters.

ILIAD = Interactive Language for Image Analysis and Display

Interpreter system, developed since 1978 at the University of Uppsala for cytology /8,9/, PDP-11/55 and VAX-11/780. The system consists of editor, translator, interpreter, and procedures. The language is PASCAL-like. The translator generates intermediate code which is processed by the interpreter. The system is written in PASCAL, 155 procedures in PASCAL and assembler. Resident system part 108k, altogether 600k. Possibilities of extension are given.

IMAGIC = Image Analysis in the Computer

Job stream system, developed since 1978 at the University of Groningen on computer NORD-10 (48k) /46/, for application to electron microscopy. The system is written in FORTRAN, 85 procedures are programmed in FORTRAN or assembler. The language IMAGIC is a command language without jumps, and without conditional operations. Job sequences of commands are possible for subsequent applications. In this way, small dialogue programs may be formed.

KANDIDATS

Command system, developed since 1971 at the University of Kansas /16/. The system allows the input of parameter commands. Procedures are written in FORTRAN; applications in remote sensing.

MAGIC = Image Basic

Interpreter system, developed since 1975 for the commercially available image processing system MAGISCAN /43/. MAGIC is implemented on computer NOVA-3/12 (32k) and is an extension of BASIC.

MDPP = Micrograph Data Processing System

Job stream system, developed since 1976 at University of Basel on IBM-370/155 (120k) /40/, application to electron microscopy. 100 procedures written in FORTRAN; all commands consist of two characters only; required parameters are called by the system in dialogue. In the case of command sequences parameters have to be input by data cards.

MINIXAP

Interpreter system, developed since 1977 at the University of Maryland on PDP-11/45 /27/. The system language is LISP; procedures may be called interactively.

MIRAGE

Interpreter system, developed since 1978 on computer MULTI-20 (32k) /52/. Application in electron microscopy. The system is a BASIC extension with special image procedures which are written in assembler language.

MOSAIK = Modulares System zur Analyse isolierter Komponenten

Compiler system, developed since 1978 for the commercially available image processing system MORPHOQUANT from VEB Carl Zeiss Jena /14,48/. Extension and improvement of system INTER. The system and 60 procedures are written in assembler. The computer KSR 4100 (16k) requires overlay processes.

NLIPS70 = NEC Laboratory Image Processing System

Job control system, developed since 1979 for the computer NEAC-3200/70 at Research Laboratories, Nippon Electric Company, Kawasaki /41/, application to remote sensing. About 100 procedures, written in FORTRAN. Commands allow missing parameters, command sequences are interpretatively processed, restricted possibilities of loops, branching, and arithmetic operations.

PAL = Picture Analyzing Language

Interpreter system, developed since 1980 at Institute for Biocybernetics in Warsaw on computer K-202 /23/, application to biomedical problems. The language is similar to ALGOL68 and allows interactive work because necessary declarations may be placed at arbitrary positions within a program.

PAX

Command system, developed since 1971 at the University of Maryland /21/. 145 procedures are written in FORTRAN; the system is written in assembler. In execute mode, a given parameter command is analyzed and realized.

PIC

Command system, developed since 1979 at Cancer Research Center, Heidelberg on computer IBM-3032 /28/, biomedical applications. 80 procedures are written in FORTRAN; the command interpreter is programmed in assembler.

PICASSO = Picture Assembly-Programmed Operations

Job control system, developed since 1976 at Institute for Biocybernetics, Warsaw on computer K-202 /23,24/, for biomedical applications. 170 procedures written in assembler may be connected to programs with jumps and loops, but without arithmetic operations. The number of variables is restricted (V1...V9).

PPL = Picture Processing Language

Interpreter system, developed since 1974 at the University of Linköping on computer D5/30 (64k) /15,22/. Procedures in FORTRAN,

language interpreter in assembler. The language is PL/I-like but structured (no jumps). Both execute mode and program mode are possible.

PROFI11 = Processing and Retrieval of Functional Images

Job control system, developed since 1977 at the Institute for Data Processing in Medicine, Hamburg, on computers PDP-11/45 and VAX-11/780 /5/, application to radiology. The commands may be put together in structured sequences; these sequences may be stored in this form. The system is written in an assembler-like language SIMPL-11. New procedures may be inserted very easily; arithmetic operations are represented as procedures. Programs will be transformed into an intermediate code used by the interpreter for efficiency.

RECON = Reconstruction Program

Menu system, developed since 1981 at the National Institute for Medical Research, London on computer PDP-11/34 /30/. Application to representation of three-dimensional pictures. RECON consists of five different dialogue programs. In these programs, via menu, 27 operations may be selected (DI=display, RZ 30=rotate about Z axis to 30°, etc.).

SAM = Sensor System for Automation and Measurement

Menu system, developed since 1980 at Fraunhofer Institute for Information and Data Processing, Karlsruhe /10/, application to robot control. Software in PLZ, a PASCAL-like language. A menu is shown on the display; the user selects functions by the input of single characters.

SCANCANS = Scanning Cell Analysis System

Command system, developed since 1975 at the University of Uppsala on computer PDP-8/e /3/, biomedical applications. Input of 40 different commands with 0 to 5 positive integer parameters each.

SPEL = Simple Picture Evaluation Language

Menu system, developed since 1975 for the commercial image processing system MAGISCAN /43/. Via light pen, on a display about

30 commands may be called. For these commands, parameters and decisions are specified in dialogue form.

SUPRPIC

Job stream system, developed since 1978 at the University of Pittsburgh on computer Perkin-Elmer 7/32 /34,35/, biomedical applications. About 20 procedures written in assembler which may be used for a wide variety of image operations, in dependence of the specified parameters. System is written in FORTRAN. Commands are two-digit numbers. In the following example, the first image (01) from a specified memory unit (09) will be copied (04) to the second image (02): 0409010902.

TASIC

Interpreter system, developed since 1977 for the commercial image processing system TAS /29/. The computer LSI-11 (32k) is controlled by the BASIC-like language TASIC. In execute mode it is also possible to input menu commands via light pen. The commands are written as standard procedures in TASIC.

TEXAC = Texture Analysis Computer

Command system, developed since 1977 at Georgetown University on PDP-11/34 /26/, for biomedical applications. The command language allows more than 20 procedures written in FORTRAN. Any command consists of a three-letter mnemonic code followed by 0 to 5 parameters. Examples of image commands are MOV A,B (store image A in image B) or ADD A,B,C (pointwise addition of images A and B, result in image C).

TICAS = Taxonomic Intra-Cellular Analytic System

Menu system, developed since 1972 on computers CDC 6400, PDP-10, and PDP-12 /1,2/. Application in cytology. Interactivity is ensured by a dialogue program, but menu selection by integer input is possible, too. Procedures are written in FORTRAN and assembler; the system is written in FORTRAN.

XAP

Command system, developed in 1972 at the University of Maryland on UNIVAC-1108 /17,18/. System in FORTRAN, procedures in assembler. The syntax of parameter commands is similar to FORTRAN.

REFERENCES

/1/ Bartels, P.H., G.F. Bahr, M. Bibbo, G.L. Wied (1972): Objective cell image analysis. J. Histochem. Cytochem. 20, 239-254.

/2/ Bartels, P.H., G.L. Wied (1976): High resolution prescreening systems for cervical cancer. Proc. 1st Conf. Automation of Uterine Cancer Cytology, Chicago 1976, 144-184.

/3/ Bengtsson, E., J. Holmquist, B. Olsen, B. Stenkvist (1976): SCANCANS - An interactive scanning cell analysis system. Comp. Progr. Biomed. 6, 39-49.

/4/ Bengtsson, E., O. Eriksson, T. Jarkrans, B. Nordinh, B. Stenkvist (1981): CELLO - An interactive system for image analysis. In: Bolc (1981), 21-45.

/5/ Böhm, M., G.C. Nicolae, K.H. Höhne (1982): PROFI11 - A simple dialog language for the processing of image sequences. Proc. ISMIII '82, Berlin-West, 386-391.

/6/ Bolc, L., Z. Kulpa (1981): Digital Image Processing Systems. Lecture Notes in Computer Science, Vol. 109, Springer Verlag, Berlin/Heidelberg/New York.

/7/ Bolc, L. (1982): Natural Language Question and Answering Systems for Image Analysis. Carl Hanser Verlag, München, & Macmillan, London.

/8/ Eriksson, O., B. Nordin, E. Bengtsson, T. Jarkrans, B. Stenkvist (1980): ILIAD - An interactive language for image analysis. Techn. Rep. Uppsala Univ. No. 78-4169/79-6187

/9/ Eriksson, O., E. Bengtsson, T. Jarkrans, B. Nordin, B. Stenkvist (1982): ILIAD - A high level dialogue system for picture analysis. In: Bolc (1982).

/10/ Foith, J.P., C. Eisenbarth, E. Enderle, H. Geiselmann, H. Ringshauser, G. Zimmermann (1981): Real-time processing of binary images for industrial applications. In: Bolc (1981), 61-168.

/11/ Gelsema, E.S., G. Eden (1980): Mapping algorithms in ISPAHAN. Pattern Recognition 12, 127-136.

/12/ Gould, I.H. (1977): IFIP-Sachwärterbuch der Datenverarbeitung. Schriftenreihe Informationsverarbeitung, Teubner-Verlags-Gesellschaft, Leipzig.

/13/ Granlund, G.H., J. Arvidsson, H. Knutsson (1982): GOP - A paradigm in hierarchical image processing. Proc. ISMIII '82, Berlin-West, 392-397.

/14/ Gretscher, P. (1982): Der Mikroskopbildanalysator MORPHOQUANT des VEB Carl Zeiss Jena. Acta Histochem. Suppl. 26, 141-144.

/15/ Gudmundsson, B. (1982): An interactive high-level language system for picture processing. CGIP 18, 392-403.

/16/ Haralick, R.M., G. Minden (1978): KANDIDATS - An interactive image processing system. CGIP 8, 1-15.

/17/ Hayes, K.C. (1972): XAP - An 1108 file-oriented picture management system. Techn. Rep. TR-213, Comp. Sc. Center, University of Maryland.

/18/ Hayes, K.C. (1975): XAP user's manual. Techn. Rep. TR-348, Comp. Sc. Center, University of Maryland.

/19/ Holmquist, J., E. Bengtsson, O. Eriksson, B. Stenkvist (1977): A program system for interactive measurements on digitized cell images. J. Histochem. Cytochem. 25, 641-654.

/20/ Hunt, B.R., D.H. Janney (1974): Digital image processing at Los Alamos Scientific Laboratory. Computer 7, 57-61.

/21/ Johnston, E.G. (1972): The PAX user's manual. Computer Note CN-7, Comp. Sc. Center, University of Maryland.

/22/ Kruse, B., P.E. Danielsson, B. Gudmundsson (1980): From PICAP I to PICAP II. In: K. S. Fu, T. Ichikawa (eds.): Special Computer Architecture for Pattern Processing, CRC Press, Boca Raton, Florida.

/23/ Kulpa, Z. (1981a): PICASSO, PICASSO-SHOW and PAL - A development of a high-level system for image processing. In: M. J. B. Duff, S. Levialdi (eds.): Languages and Architectures for Image Processing. Academic Press, New York, 13-24.

/24/ Kulpa, Z., J. Dernalowicz, H.T. Nowicki, A. Bielik (1981b): CPO-2/K-202 - A universal digital image analysis system. In Bolc (1981), 169-200.

/25/ Kupka, I., N. Wilsing (1980): Conversational Languages. J. Wiley & Sons, New York.

/26/ Ledley, R.S., Y.G. Kulkarni, C.M. Park, M.R. Shiu, L.S. Rotolo (1978): TEXAP - A powerful new picture pattern recognition computer. Proc. Pattern Recognition and Image Processing Conf., Chicago, 396-401.

/27/ Luczak, E.C., K.C. Hayes (1977): The MINIXAP image processing system. Techn. Rep. TR-601, Comp. Sc. Center, University of Maryland.

/28/ Meinzer, H.P., U. Engelmann (1982): A language for the interactive manipulation of digitized images. Proc. 6th ICPR, München, Oct. 1982, 68-70.

/29/ Nawrath, R., J. Serra (1979): Quantitative image analysis - theory and instrumentation. Microsc. Acta 82, 101-111.

/30/ Perkins, W.J., R.J. Green (1982): Three-dimensional reconstruction of biological sections. J. Biomed. Enging. 4, 37-43.

/31/ Preston, K. (1971): Feature extraction by Golay hexagonal pattern transforms. IEEE Trans. C-20, 1007-1014.

/32/ Preston, K., P.E. Norgren (1972): Interactive image processor speeds pattern recognition by computer. Electronics No. 10.

/33/ Preston, K. (1979): Languages for biomedical image processing. COMSAC Conference Proc., 1-6.

/34/ Preston, K. (1980): Interactive system for medical image processing. In: M. Onoe, K. Preston, A. Rosenfeld (eds.): Real-Time Medical Image Processing. Plenum Press, New York, 193-206.

/35/ Preston, K. (1981): Tissue section analysis - feature selection and image processing. Pattern Recognition 13, 17-36.

/36/ Reeves, A.P. (1979): An array processing system with a FORTRAN-based realization. CGIP 9, 267-281.

/37/ Rodenacker, K., P. Gais, W. Abmayr (1980): Analysis of textures with DIBIVE. A system for digital image processing. Proc. 5th Int. Congr. Stereology, Salzburg, Sept. 1979, Mikroskopie 37, Suppl., 421-424.

/38/ Rosenfeld, A. (1983): Picture Processing: 1982. TR-1239, Comp. Sc. Center, University of Maryland.

/39/ Simon, H., K. Voss, K. Wenzelides (1983): Automated Microscopic Image Analysis. Suppl. Exp. Pathol., in press.

/40/ Smith, P.R. (1978): An integrated set of computer programs for processing electron micrographs of biological structures. Ultramicrosc. 3, 153-160.

/41/ Tajima, J., T. Temma, K. Naito, T. Arakawa, S. Hanaki (1981): An interactive image processing system (NLIPS-70). NEC Research & Development No. 63, 10-20.

/42/ Tamura, H., F. Tomita, S. Sakane, N. Yokoya, K. Sakaue, M. Kaneko (1982): A Transportable Image Processing Software Package: SPIDER. Proc. 6th ICPR, München, Oct. 1982, 75-78.

/43/ Taylor, C.J., J.N. Brunt, R.N. Dixon, P.J. Gregory (1978): The MAGISCAN - A new generation, software based, automatic image analyser. Spec. Issues of Practical Metallography (ed. J.C. Chermant), Dr. Riedener-Verlag, Stuttgart, 433-442.

/44/ Troxel, D.E. (1981): An interactive image processing system. IEEE Trans. PAMI-3, 95-101.

/45/ Van der Ploeg, M., K. Van der Broek, A.W.M. Smuelders, A. M Vossepoel, P. Van Duijn (1977): HIDACSYS - Computer programs for interactive scanning cytophotometry. Histochem. 54, 273-288.

/46/ Van Heel, M., W. Keegstra (1981): IMAGIC - A fast, flexible and friendly image analysis software system. Ultramicrosc. 7, 113-130

/47/ Voss, K. (1979a): Konzeption eines Morphometriesystems für die Mikroskopbildanalyse. Bild und Ton 32, 369-371

/48/ Voss, K., E. Neumann, W. Witsack (1979b): Universelles Programmsystem für den automatischen Mikroskopbildanalysator MORPHOQUANT. Jenaer Rundschau 24, 167-169

/49/ Voss, K., H. Simon (1981): AMBA - Ein Softwaresystem für die automatisierte Mikroskopbildanalyse in Medizin und Biologie. Wiss. Zeitschr. HU Berlin, Math.-Nat. 30, 341-346

/50/ Voss, K., P. Hufnagl (1983): AMBA/2 - Ein Softwaresystem für die automatische Bildverarbeitung. MKR Weiterbildungszentrum, TU Dresden, Heft 60/82, 88-104

/51/ Vossepoel, A. M., A. W. M. Smeulders, K. Van den Broek (1979): DIODA-Delineation and feature extraction of microscopical objects. Comp. Progr. Biomed. 10, 231-244

/52/ Wade, R. H., A. Brisson, L. Tranqui (1980): The application of image treatment to structural analysis in biology. J. Microsc. Spectrosc. Electron. 5, 695-712

/53/ Wilhelmi, W. (1980): A programming tool for image processing, Proc. 5th ICPR, Miami Beach, Dec. 1980, 742-744

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

TWO-DIMENSIONAL DISCRETE GAUSSIAN MARKOV RANDOM FIELD MODELS FOR IMAGE PROCESSING

R. CHELLAPPA
Department of EE-Systems and
Image Processing Institute
University of Southern California
Los Angeles, California

Abstract

This paper is concerned with a systematic exposition of the usefulness of two-dimensional (2-D) discrete Gaussian Markov random field (GMRF) models for image processing applications. Specifically, we discuss the following topics: Notion of Markovianity on a plane, statistical inference in GMRF models, and their applications in several image related problems such as image synthesis and compression, image classification and image restoration.

I. INTRODUCTION

In the analysis and processing of 2-D images one encounters a large amount of data. In a typical image restoration or compression problem, it is not uncommon to work with images defined on grids of dimensions 256 x 256 or 512 x 512. To enable efficient processing of this data, it would be preferable to have an underlying model that explains the dominant statistical characteristics of the given data. Subsequent processing of the images can be efficiently done using the models fitted to the images. Although it is very difficult to identify the underlying physical mechanisms that could have possibly generated the observed data, any analytical expression that explains the nature and extent of dependency of a pixel intensity on intensities of its neighbors can be said to be a model.

A typical image is represented by the gray level variations defined over a rectangular or square lattice. One of the important characteristics of image data is the special nature of the statistical dependence of the gray level at a lattice point on those of its neighbors. The different classes of image models suggested in the literature attempt to characterize this dependence among the neighboring pixels. One way of characterizing the statistical dependence among the neighboring pixels is to represent y(s) as a linear weighted combination of $\{y(s+r), r \varepsilon N\}$ and additive noise where N is known as the neighbor set and does not contain (0,0). Specific restrictions on the members of the neighbor set N yield representations familiar in image processing literature For example, the popular "causal models" are obtained when N is defined as a subset of the set $\{(i,j): i \leq 0, j \leq 0, (i,j) \neq (0,0)\}$. One of the features of using the causal model is that the resulting image processing algorithms are recursive. One can generalize the causal models to obtain the class of "unilateral" models by including more neighbors, but still preserving the recursive structure of the image processing algorithms. The unilateral models result when N is a subset of the non symmetric half plane S^+ defined recursively as in [Goodman & Ekstrom, 1980]. Unlike in 1-D discrete stochastic process, where the existence of a preferred direction is inherently assumed, no such preferred ordering is appropriate for a 2-D discrete lattice. Thus, it is possible that an observation y(s) is dependent on neighboring observations in all directions leading to noncausal or bilateral representation. The simplest non-causal model is obtained when y(s) is dependent on its east, west, north and south neighbors. One can consider, more general representation by including the diagonal neighbors and so on.

In this paper we are concerned with a particular class of 2-D noncausal models known as the Gaussian Markov random field (GMRF) models. Let $\{y(s),\ s\in\Omega\}$, $\Omega=\{s=(i,j);\ 0\leq i,\ j\leq M-1\}$ be the observations from an image. It is postulated that this data is generated by an appropriate 2-D GMRF model. The 2-D GMRF models characterize the statistical dependency among pixels by requiring that

$$p(y(s)|\ \text{all } y(r),\ r\neq s) = p(y(s)|\ \text{all } y(s+r),\ r\in N)$$

where N is the appropriate symmetric neighbor set. For instance $N=\{(0,1),(0,-1),(-1,0),(1,0)\}$ corresponds to the simplest GMRF model and by including more neighbors we can construct higher order GMRF models. Since GMRF models are defined only for symmetric neighbor sets, often N is equivalently characterized using an asymmetrical neighbor set N_s; i.e., if $r\in N_s$ then $-r\notin N_s$ and $N=(r:r\in N_s)\cup(-r:r\in N_s)$. Since the introduction of GMRF models in the literature [Rosanov, 1967; Woods, 1972] there has been considerable interest in using GMRF models for image restoration, image classification, synthesis and coding.

Prior to any practical application of GMRF models two major problems have to be tackled, viz. the estimation of parameters in GMRF models and the choice of appropriate N for the given image. We discuss several estimation methods for GMRF models. Of particular interest are the statistical properties of these estimates like asymptotic consistency and efficiency. We also discuss decision rules for choosing the appropriate N for the given image. To illustrate the usefulness of GMRF models for image processing applications, algorithms for texture synthesis and coding, texture classification and image restoration are given alongwith pictorial examples.

The organization of the paper is as follows: the representation of GMRF models in 1-D and 2-D is discussed in Section 2. Procedures for the synthesis of an image pattern obeying a known GMRF model are given in Section 3. Methods for estimation of parameters in GMRF models are discussed in Section 4 along with the statistical properties of the estimates. Decision rules for choosing the appropriate structure of the model are given in Section 5. The usefulness of the GMRF models for texture synthesis, texture classification and image restoration is illustrated in Sections 6,7, and 8.

2. MODEL REPRESENTATION

We first give a brief review of the relevant theory of 1-D models and subsequently discuss the representation of 2-D models.

2.1 One-Dimensional GMRF Models

Definition 2.1: Suppose we have a set of discrete Gaussian observations $\{r(t),\ t=1,2,\ldots,N\}$. Then, $\{r(.)\}$ is said to be a unilateral m^{th} order Markov process in a strict sense [Doob, 1953] if,

$$p(r(t)|\ \text{all } r(j),\ j<t) = p(r(t)\ |\text{all } r(t-j),\ 1\leq j\leq m), \tag{2.1}$$

The sequence $\{r(.)\}$ obeying (2.1) has the representation

$$r(t) = \sum_{j=1}^{m} \theta_j\, r(t-j) + \sqrt{\beta}\ \omega(t),\ t=1,2,\ldots,N, \tag{2.2}$$

with associated initial conditions $\{r(0),\ r(-1),\ldots,r(1-m)\}$. In (2.2), $\{\omega(.)\}$ is an independent and identically distributed (IID) Gaussian noise sequence with mean zero and variance unity.

Definition 2.2. The set $\{r(.)\}$ is said to be a unilateral m^{th} order Markov process in a wide sense [Doob, 1953] if,

$$E(r(t) \mid \text{all } r(j),\ j < t) = E(r(t) \mid \text{all } r(t-j),\ 1 \leq j \leq m)$$

and

$$\text{Var}(r(t) \mid \text{all } r(j),\ j < t) = \text{Var}(r(t) \mid \text{all } r(t-j),\ 1 \leq j \leq m), \tag{2.3}$$

For any arbitrary continuous distribution of $\{\omega(.)\}$, the sequence $\{r(.)\}$ in (2.2) is a wide sense unilateral m^{th} order Markov process. Whether Gaussian or not, strict sense Markovianity implies Markovianity in wide sense, but the converse is true only for Gaussian variables. The notion of unilateral Markov process can be extended to include dependence on neighbors on either side leading to bilateral Markov process.

Definition 2.3: The sequence $\{r(.)\}$ is said to obey a m^{th} order bilateral Markov process in a strict sense if

$$p(r(t) \mid \text{all } r(i),\ i \neq t) = p(r(t) \mid r(t-1),\ldots,\ r(t-m),\ r(t+1),\ldots,r(t+m)) \tag{2.4}$$

The sequence $\{r(.)\}$ obeying (2.4) has the representation [Woods, 1972; Bartlett, 1975]

$$r(t) = \sum_{i=-m}^{m} \theta_i\, r(t-i) + e(t),$$

where

$$\theta_k = \theta_{-k} \text{ and } \{e(\bullet)\}$$

is a correlated Gaussian noise sequence with the conditional structure

$$p(e(s) \mid \text{all } r(t),\ t \neq s) = p(e(s) \mid \text{all } r(s+t),\ -m \leq t \leq m,\ t \neq 0), \tag{2.5}$$

Equation (2.5) implies a weaker condition involving second order properties, as in

$$E(e(s) \mid \text{all } r(t),\ t \neq s) = 0,$$

and

$$\text{Var}(e(s) \mid \text{all } r(t),\ t \neq s) = \nu, \tag{2.6}$$

which in turn implies that the correlated noise sequence $\{e(.)\}$ has the following correlation structure,

$$\begin{aligned} E(e(t)\ e(s)) &= \nu,\ t=s \\ &= -\theta_{t-s}\nu,\ |t-s| \leq m \\ &= 0, \text{ otherwise} \end{aligned} \tag{2.7}$$

Equation (2.7) can be derived as a special case of the derivation for 2-D GMRF models given later. The bilateral Markov Model is driven by a correlated noise sequence. A consequence of (2.5) is

$$E(r(s) \mid \text{all } r(t),\ t \neq s) = \sum_{i=-m}^{m} \theta_i\, r(s-i)$$

A wide sense bilateral Markov model can be defined similar to a wide sense unilateral process. As in the case of unilateral models whether Gaussian or not, strict sense Markovianity implies wide sense Markovianity, but the converse is true only for Gaussian variables.

If $\{r(.)\}$ obeys a unilateral strict sense model of order m then it also obeys a bilateral strict sense model of order m [Woods, 1972]. Further, if 1-D spectral factorization theorem applies, a 1-D bilateral strict sense Markov process (2.5) of order m always possesses an equivalent unilateral wide sense Markov Process (2.3) of order m with identical spectral density function. Thus, under Gaussian assumption, when spectral factorization applies

$$(2.1) \leftrightarrow (2.4) \tag{2.8}$$

For non-Gaussian variables, (2.8) is valid in the wide sense.

Example 2.1 [Bartlett, 1975]: Consider the first-order unilateral Markov model $r(i) = \theta r(i-1) + \sqrt{\beta}\,\omega(i)$ where $\{\omega(.)\}$ is an IID Gaussian sequence. The spectral density function of $\{r(.)\}$ is

$$S_r(\lambda) = \frac{\beta}{(1+\theta^2 - 2\theta\cos\lambda)}$$

The equivalent first-order bilateral model has the representation

$$r(i) = \frac{\theta}{1+\theta^2}\,(r(i-1) + r(i+1)) + e(i)$$

where $\{e(i)\}$ is a correlated noise sequence with the correlation structure

$$E(e)i)\ e(j)) = \frac{\beta}{1+\theta^2},\ i=j$$

$$= \frac{-\beta\theta}{(1+\theta^2)^2},\ |i-j| = 1$$

$$= 0,\ \text{Otherwise}$$

2.2 Unilateral 2-D GMRF Models

Assume that the observations $\{y(s), s\varepsilon\Omega\}$, obey the difference equation;

$$y(s) = \sum_{r\varepsilon N} \theta_r\, y(s+r) + \sqrt{\beta}\,\omega(s),\ \forall\ s, \tag{2.9}$$

The set N associated with (2.9) is a subset of the set $\{(i,j)\colon i\leq 0,\ j\leq 0,\ (i,j)\neq(0,0)\}$ and the noise sequence $\{\omega(s)\}$ is IID. Typically, (2.9) has a set of initial conditions associated with it. For $N=\{(-1,0),\ (0,-1),\ (-1,-1)\}$, (2.9) is characterized by a set of initial conditions made of the observations along the first row and first column. The familiar "causal" models are obtained by restricting N as a subset of quarter plane. The causal models can be generalized to include as many previously scanned points as possible by using the notion of non-symmetric half plane S^+ [Goodman & Ekstrom, 1980] recursively defined as

1. $s\varepsilon S^+,\ r\varepsilon S^+ \rightarrow r+s\varepsilon S^+$

2. $s\varepsilon S^+ \rightarrow -s \notin S^+$

3. $(0,0) \notin S^+$

The definition of S^+ is not unique. Without loss of generality, one can consider those half planes generated by vectors collinear with cartesian axes. With N being a subset of S^+, the corresponding $\{y(s)\}$ is said to be strict sense unilateral Markov with respect to a unilateral neighbor set N if

$$p(y(s) \mid \text{all } y(r),\ r \in \Omega_{s,N}) = p(y(s) \mid \text{all } y(s+r),\ r \in N) \tag{2.10}$$

where $\Omega_{s,N}$ is defined in terms of s and N as follows:

1. $s \notin \Omega_{s,N}$
2. $s+r \in \Omega_{s,N}$ for all $r \in N$
3. $r \in \Omega_{s,N} \rightarrow (r+t) \in \Omega_{s,N}$ for all $t \in N$ provided $r+t \neq s$

Some examples of possible structures of $\Omega_{s,N}$ are shown in Fig. 2.1 [Kashyap, 1981b].

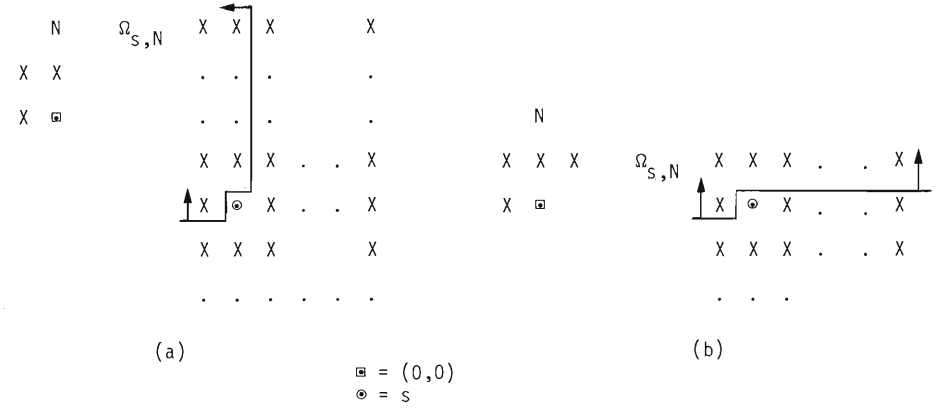


Fig. 2.1 Examples of N and $\Omega_{s,N}$ for unilateral GMRF Models [Kashyap, 1981b].

2.3 Non-Causal or Bilateral 2-D GMRF Models

Unlike 1-D discrete time series, where the existence of a preferred direction is inherently assumed, no such preferred ordering of the discrete lattice is appropriate. In other words, the notion of "past" and "future" as understood in unilateral 1-D Markov processes is restrictive in 2-D as it implies a particular ordering of the lattice. It is quite possible that an observation at s, may be dependent on neighboring observations in all directions. The simplest non causal model is obtained when the dependence is on the nearest north, south, east and west neighbors. One can think of more general dependence on nearest diagonal neighbors and neighbors farther. Such a noncausal representation in 2-D is of interest not only due to its generality, but is significant due to lack of spectral factorization in two-dimensions. Thus, if the given observation set $\{y(s)\}$ indeed obeys a non causal model whose spectrum does not factorize, any finite order unilateral representation for this data will only be approximate. In our subsequent discussion, the noncausal or bilateral GMRF models will be simply refered to as GMRF models.

Definition 2.4: The observation set $\{y(s)\}$ is said to be strictly Markov with respect to a symmetric neighbors set N if,

$$p(y(s) \mid \text{all } y(r),\ r \neq s) = p(y(s) \mid \text{all } y(s+r),\ r \in N), \tag{2.11}$$

A hierarchy of GMRF models can be defined as follows: when $N_S = \{(0,1), (1,0)\}$, we obtain a first-order GMRF model and with $N_S = \{(0,1),(1,0),(-1,1),(1,1)\}$, a second-order model and so on. Figure 2.2 illustrates this hierarchy up to seventh-order model. A wide sense GMRF model can be defined as follows:

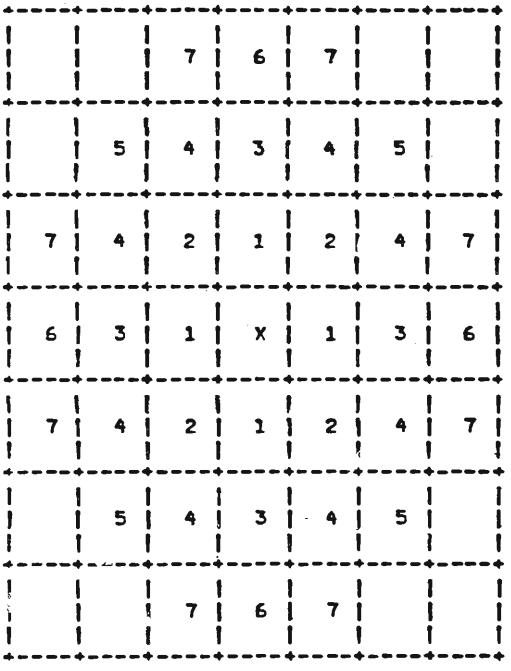


Fig. 2.2 Hierarchy of GMRF models. The numbers indicate the order of the model relative to x. [G.R. Cross, Ph.D., Thesis, Michigan State University, 1980].

Definition 2.5: The observation set $\{y(s)\}$ is said to be wide sense Markov with respect to a symmetric neighbor set N if,

$$E(y(s) \mid \text{all } y(r),\ r \neq s) = E(y(s) \mid \text{all } y(s+r),\ r \in N) \tag{2.12}$$

and

$$\text{Var}(y(s) \mid \text{all } y(r),\ r \neq s) = \text{Var}(y(s) \mid \text{all } y(s+r),\ r \in N) > 0 \tag{2.13}$$

In our discussion, the second of the two conditions is restricted to be

$$\text{Var}(y(s) \mid \text{all } y(r),\ r \neq s) = \nu > 0. \tag{2.14}$$

The definition (2.4) of strict Markovianity is often referred to as local Markov property compared to the global Markovian definition used in [Rosanov, 1967; Woods, 1972]. The global definition of Markovianity is as follows [Woods, 1972]. Let a band of minimum width P (a band of minimum width P is a set of contiguous lattice points with a well-defined inside and outside Ω^+ and Ω^-, such that all points in Ω^+ are at least distance P from every point in Ω^-) denoted as $\partial\Omega$ (the "present") separate the discrete lattice Ω into two regions Ω^+ and Ω^- (the "future" and "past"). The set $y(s)$, $s \in \Omega^+$ is said to be Markov of order P if,

$$p(y(s),\ s \in \Omega^+ \mid y(s),\ s \in \Omega^-,\ s \in \partial\Omega) = p(y(s),\ s \in \Omega^+ \mid y(s),\ s \in \partial\Omega)$$

This global definition includes the local Markov definition 2.4. A proof is given in [Rosanov, 1967] to show that for GMRF models local Markovianity implies global Markovianity.

Given a finite image we can analyze the image as a finite slice of an underlying infinite lattice image. This approach leads to the class of GMRF models known as infinite lattice GMRF models. In general the infinite lattice models, do not give computationally attractive algorithms for image processing. Another class of models, known as the finite lattice models, obtained by assuming special boundary conditions, yields computationally efficient processing algorithms using fast transforms like discrete Fourier, discrete sine and cosine. We first give the representations of infinite lattice GMRF models, discuss some computational problems associated with this representation and proceed to the representation of finite lattice GMRF models.

2.3.1 Infinite Lattice GMRF Models [Rosanov, 1967; Woods, 1972]

Assume that the observations $\{y(s), s\varepsilon\Omega\}$ have zero mean, Gaussian distribution and obey the following difference equation

$$y(s) = \sum_{r\varepsilon N_s} \theta_r(y(s+r) + y(s-r)) + e(s) \tag{2.15}$$

where the stationary Gaussian noise sequence e(s) has the following properties.

$$\begin{aligned} E(e(s)\, e(r)) &= -\theta_{s-r}\nu,\ (s-r)\varepsilon N \\ &= \nu,\ s=r \\ &= 0,\ \text{otherwise.} \end{aligned} \tag{2.16}$$

The set N_s is the asymmetric neighbor set mentioned in Section 1. Using (2.15) and (2.16), one can prove that

$$\begin{aligned} E(e(s)\, y(r)) &= 0,\ r \neq s \\ &= \nu,\ r = s \end{aligned} \tag{2.17}$$

In view of the Gaussian assumption (2.17) implies

$$E(e(s)\ |\text{all } y(r),\ r \neq s) = 0$$

which in turn implies

$$p(y(s)\ |\text{all } y(r),\ r \neq s) = p(y(s)\ |\text{all } y(s+r),\ r\varepsilon\ N) \tag{2.18}$$

where N is related to N_s as mentioned in Section 1.

Another equivalent way of deriving the representation is to begin with the definition of Markovianity as in (2.11) and show that $\{y(s)\}$ and $\{e(s)\}$, satisfy (2.15) and (2.16) respectively. This is done as follows [Rosanov, 1967; Woods, 1972]: since (2.11) is true,

$$E(y(s)\ |\text{all } y(r),\ r \neq s) = \sum_{r\varepsilon N} \theta_r\, y(s+r) \tag{2.19}$$

Equation (2.19) means that the difference

$$e(s) = y(s) - \sum_{r\varepsilon N} \theta_r\, y(s+r)$$

is orthogonal to all the variables $y(r)$, $r \neq s$. Thus,

$$E(e(r)\, y(s)) = 0, \quad r \neq s$$
$$= \nu \text{ (for some } \nu), \quad r=s \tag{2.20}$$

Since $y(s)$ is Gaussian and zero mean, (2.20) implies

$$E(e(r) \,|y(s),\ r \neq s) = 0 \tag{2.21}$$

Equations (2.19) – (2.21) imply that $y(s)$ should satisfy (2.15). Equation (2.15), together with (2.21), helps to establish (2.16).

Since it is required that

$$E(e(s)e(t)) = -\theta_r \nu, \quad r = s-t$$
$$= E(e(t)e(s)) = \theta_{-r} \nu$$

it follows that GMRF model representation is defined only for symmetric neighbor sets.

A fundamental difference between 1-D and 2-D Markov models is that in 2-D case, there is no simple connection between the noncausal and unilateral GMRF models. This is mainly due to lack of spectral factorization in two-dimensions. Consequently, given that a 2-D sequence $\{y(s)\}$ is strict or wide sense Markov with respect to a neighbor set N, there may not exist any neighbor set N', so that $\{y(s)\}$ is unilateral strict or wide sense Markov with respect to N'. However, any 2-D sequence $\{y(.)\}$ which is unilateral strict Markov with respect to a neighbor set N is also bilateral strict Markov with respect to a symmetric neighbor set N". An example of N and N" is given below and more examples are in [Kashyap, 1981b; Abend, et al., 1965].

Example 2.2 [Jain, 1976]: Consider the 2-D causal Gaussian white noise driven model,

$$y(s) = p\, y(s+(0,-1)) + p\, y(s+(-1,0))$$
$$- p^2 y(s+(-1,-1)) + \omega(s)$$
$$\text{with } E(\omega(s)) = 0 \text{ and } E(\omega^2(s)) = (1-p^2)$$

The equivalent noncausal GMRF model is given by

$$y(s) = \alpha \sum_{r \in N_1} y(s+r) + \alpha^2 \sum_{r \in N_2} y(s+r) + e(s)$$

where

$$N_1 = \{(-1,0),(1,0),(0,-1),(0,1)\},$$
$$N_2 = \{(-1,1),(1,-1),(-1,-1),(1,1)\}$$

with the correlation structure of the Gaussian noise sequence $\{e(s)\}$ being specified as

$$E(e(s)\, e(r)) = \beta^2, \quad s = r$$
$$= -\alpha\beta^2, \quad r-s \in N_1$$
$$= \alpha^2\beta^2, \quad r-s \in N_2$$
$$= 0, \text{ otherwise}$$

where

$$\alpha = \frac{p}{1+p^2} \text{ and } \beta = \frac{1-p^2}{1+p^2}$$

2.3.2. Finite Lattice GMRF Models

Effectively, finite lattice models are obtained by making specific assumptions regarding the neighbors of the boundary pixels. The infinite lattice GMRF models considered in the previous section, although devoid of these assumptions, are difficult to analyze. For example, computation of even some of the basic quantities like covariance matrices is very expensive. The synthesis of images obeying infinite lattice GMRF models can be done only by iterative methods [Cross & Jain, 1983] which are time consuming. On the other hand, the finite lattice GMRF models are easy to analyze and lead to computationally elegant solutions in Wiener filtering, synthesis and compression of images. We discuss a specific finite lattice GMRF model representation that assumes doubly periodic boundary conditions. This model is represented as,

$$y(s) = \sum_{r \in N} \theta_r \, r(s \oplus r) + e(s) \tag{2.22}$$

where $\oplus$ indicates sum modulo M along both the co-ordinate axes and $\{e(s)\}$ is a stationary noise sequence with the correlation structure in (2.16). By defining

$$\underline{y} = \text{Col}\cdot [y(0,0),y(0,1),y(0,M-1),y(1,0)\ldots$$

$$y(1,M-1),\ldots,y(1,M-1),\ldots,y(M-1,M-1)],$$

an M^2-vector,

and

$$\underline{e} = \text{Col.}[e(0,0),\ldots,e(0,M-1),\ldots,e(M-1,0),\ldots,e(M-1,M-1)], \text{ an } M^2\text{-vector}$$

Equation (2.22) can now be equivalently written as,

$$B(\underline{\theta})\, \underline{y} = \underline{e}, \tag{2.23}$$

where $B(\underline{\theta})$ is a block-circulant symmetric matrix

$$B(\underline{\theta}) = \begin{bmatrix} B_{0,0} & B_{0,1} \cdots & B_{0,M-1} \\ B_{0,M-1} & B_{0,0} & \ldots B_{0,M-2} \\ B_{0,1} & \cdots & \ldots B_{0,0} \end{bmatrix} \tag{2.24}$$

where each $B_{0,i}$ is M x M matrix and

$$B_{0,i} = B_{0,M-i}$$

A necesssary and sufficient condition to ensure bounded input bounded output stability is

$$\mu_s \triangleq (1 - 2\underline{\theta}^T \underline{\phi}_s) > 0, \; \forall \, s \in \Omega, \tag{2.25}$$

where

$$\underline{\phi}_s = \text{Col.}[\cos \frac{2\pi}{M} s^t r, \; r \in N_s], \tag{2.26}$$

The specific representation given in (2.22) can be arrived at by assuming that the given image is represented on a toroidal lattice. Restrictive as it may appear, the toroidal structure has been widely used in several investigations. It was shown in a pioneering work [Onsager, 1944] that in 2-D, one can solve problems relating to finite lattices analytically using periodic boundary conditions and then obtain the results corresponding to infinite lattices by limiting arguments. In [Moran, 1973; 1973], it was shown how an infinite GMRF model can be constructed from a toroidal lattice GMRF model by letting the lattice size tend to infinity. Toroidal lattice assumptions have been made in [Moran & Besag, 1975] to obtain Gaussian maximum likelihood estimates of parameters of GMRF model. One of the important conclusions of these investigations is that, the final results obtained with and without periodic boundary conditions are not significantly different from each other. Simulation results supporting the above may be found in [Chellappa, 1981].

Several other boundary conditions have been considered in the literature [Kashyap, 1981a; Jain, 1976; 1981] yielding different finite lattice GMRF models. For instance, a useful finite lattice GMRF model is obtained when $\{y(s)\}$ is reflected along the boundaries and N is $\{(-1,0), (1,0),(0,-1),(0,1)\}$. When toroidal representation is used the efficient computational algorithms are obtained for any arbitrary symmetric N; however, the computational advantages are obtained using the boundary conditions mentioned above only for the particular set $N=\{(0,1),(0,-1),(-1,0),(1,0)\}$. Hence, the toroidal representation is more useful.

At this point we would like to point out the differences between the noncausal GMRF models discussed in this section and the class of noncausal models known as spatial autoregressive (SAR) models introduced in a pioneering paper [Whittle, 1954]. The representation of $\{y(s)\}$ obeying a SAR model is

$$y(s) = \sum_{r \in N} \theta_r \, y(s+r) + \sqrt{\beta} \, \omega(s), \tag{2.27}$$

In (2.27), $(\theta_r, r \in N)$ and β are unknown parameters, and $\omega(.)$ is an IID noise sequence with zero mean and unit variance. The neighbor set N does not include (0,0) and need not be symmetric. However, if N has symmetric neighbors the corresponding coefficients should be equal; i.e., if $r \in N$ and $-r \in N$, $\theta_r = \theta_{-r}$, otherwise the parameters are not identifiable [Besag, 1974]. If N is symmetric, a necessary and sufficient condition for $\{y(.)\}$ to be stationary is

$$\{1 - \sum_{(i,j) \in N} \theta_{i,j} \, z_1^{\,i} \, z_2^{\,j}\} \neq 0,$$

for all z_1 and z_2 such that $|z_1| = |z_2| = 1$. Stability conditions for other cases may be found in [Bose, 1982; Jain, 1981].

The main difference between the SAR model and the Gaussian GMRF model is that the $\{y(s)\}$ obeying (2.27) is not Markov with respect to the noncausal neighbor set N, i.e.,

$$p(y(s) \mid \text{all } y(r),\ s \neq r) \neq p(y(s) \mid \text{all } y(s+r),\ r \in N)$$

However, when $\{y(s)\}$ is Gaussian, it is possible to construct a set N_1 which is a superset of N such that $y(.)$ obeying a Gaussian SAR model (2.27) is non causal Markov with respect to N_1. For instance, if $N = \{(0,1),(0,-1),(-1,0), (1,0)\}$ then $N_1 = [s: s \in N_s \text{ or } -s \in N_s]$ where $N_s = [(1,0), (0,1),(1,1),(-1,1),(0,2),(2,0)]$. The converse is not always true. Given a GMRF model with neighbor set N, there may not exist a finite parameter SAR model with the same second-order properties. A simple example is the GMRF model with $N=\{(0,1),(0,-1),(-1,0),(1,0)\}$. The SAR models are useful for image restoration [Chellappa & Kashyap, 1982a], texture synthesis [Chellappa & Kashyap, 1982b] texture classification [Kashyap et al., 1982] and 2-D spectral estimation [Chellappa & Sharma, 1983]. Discussion of estimation methods and the decision rules for choice of appropriate N may be found in [Kashyap & Chellappa, 1983].

3. SYNTHESIS OF IMAGES USING GMRF MODELS

Prior to application of GMRF models for image synthesis and coding, it is useful to see the kind of image patterns that are generated by GMRF models. In this experiment we assume certain model structures and parameters characterizing the models, use a pseudo random number generator to construct {e(s)} and generate {y(s)} obeying (2.15). It should be pointed that since the models are noncausal, in general recursive generation schemes will not be useful; for the case, when the spectral density function of the GMRF model factorizes, recursive generation schemes can be used.

The synthesis procedure begins with the representation

$$B(\underline{\theta})\ \underline{y} = \underline{e} \tag{3.1}$$

where $B(\underline{\theta})$ is a symmetric $M^2 \times M^2$ matrix. Assuming that $\underline{\theta}$ takes values so that $B^{-1}(\underline{\theta})$ exists, the image vector $\underline{y}$ can be written as,

$$\underline{y} = B^{-1}(\underline{\theta})\ \underline{e}, \tag{3.2}$$

Direct implementation of (3.2) is not very attractive due to the computational burden of inverting an $M^2 \times M^2$ symmetric matrix where M could be 32, 64 or 128. However, if we assume that y(s) is represented on a toroidal lattice as in (2.22), then $B^{-1}(\underline{\theta})$ is a block-circulant matrix with eigenvalues $1/\mu_s$ and eigenvectors

$$\underline{f}_s = \text{Col.}[1, \lambda_i, \lambda_i^2\underline{t}_j, \ldots, \lambda_i^{M-1}\underline{t}_j],\ s = (i,j),\ \text{an } M^2\text{-vector}$$

$$\underline{t}_j = \text{Col.}[1, \lambda_j, \lambda_j^2, \ldots, \lambda_j^{M-1}],\ \text{an M-vector} \quad \text{and} \quad \lambda_i = \exp(\sqrt{-1}\ \frac{2\pi i}{M})$$

Thus, $\underline{y}$ can be computed as [Kashyap, 1981a; 1981b]

$$\underline{y} = \frac{1}{M^2} \sum_{s\in\Omega} \frac{\underline{f}_s\, x_s}{\mu_s}$$

where

$$x_s = \underline{f}_s^{*T}\ \underline{e}, \tag{3.3}$$

Since $\underline{e}$ is Gaussian with correlation matrix $B(\underline{\theta})$, one can generate $\underline{e}$ as follows: let $\{\omega(s), s\in\Omega\}$ be an array of zero mean unit variance white Gaussian array. Generate an array of random variables, $\{\varepsilon(s), s\in\Omega\}$ as,

$$\varepsilon(s) = \omega(s)(\mu_s)^{1/2}$$

and let

$$\xi(s) = \frac{1}{M} \sum_{r\in\Omega} \varepsilon(r) \exp[-\sqrt{-1}\ \frac{2\pi}{M}\ (s^T r)]$$

It can be shown [Woods, 1972] that the correlation matrix of $\xi\{(s)\}$ is identical to that of $\{e(s)\}$. For the case of toroidal lattice representation one can save some computations by using the fact that the square root of $B^{-1}(\underline{\theta})$ is also block-circulant with eigenvalues $1/\sqrt{\mu_s}$. Consider the representation [Kashyap, 1981a]

$$\sqrt{B}(\underline{\theta})\ \underline{y} = \sqrt{\nu}\ \underline{n} \tag{3.4}$$

where $\underline{n}$ is M^2 vector of zero mean and unit variance Gaussian random numbers. It can be easily checked that (3.4) is an equivalent representation of (2.23). The synthesis of $\underline{y}$ obeying (3.4) can be done as

$$\underline{y} = \frac{1}{M^2} \sum_{s \in \Omega} \frac{\underline{f}_s x_s}{\sqrt{\mu_s}} \tag{3.5}$$

where

$$x_s = \underline{f}_s^{*T} \underline{n} \tag{3.6}$$

We have given in Fig. 3.1, sixteen 64 x 64 image patterns generated using (3.5). The details of the models are in Table 3.1. To make the display of synthetic textures pleasing to the eyes, a constant α = 30.0 (corresponding to the mean of the image) has been added. The gray scale values of the images lie in the range 0-63. It can be seen from an inspection of Fig. 3.1, that the generated patterns are quite varied and some of them look similar to natural textures. Diagonal neighbor sets seem to induce diagonal patterns, as in the windows in positions (1,2),(1,3),(2,2), and (2,3). The role played by adding nearest neighbors can be illustrated using patterns (4,2),(4,3), and (2,1). The window (4,2) corresponds to N_s = {(1,0),(0,1),(2,0),(0,2)} and produces horizontally oriented, macro structured strip patterns. By adding the symmetric neighbor (1,1) a diffused version in window (4,3) is produced. When an extra symmetric neighbor (1,-1) is added we obtain a more micro-structured pattern resembling water. By adding similar neighbors to the model in (4,1), vertically oriented patterns can be produced. To illustrate the role played by the coefficients in generating the patterns, twelve, 64 x 64 patterns corresponding to the GMRF model N_s = {(-1,1),(1,1)}, α=30.0, ν =1.1111, $\theta_{-1,1}$ = .28 and $\theta_{1,1}$ = -.14 were generated. It was found [Chellappa, 1981] that the basic pattern is still retained in all images but the "busyness" of the pattern varied. All the patterns considered, thus far, were generated using the same pseudo random number generator. As illustrated in [Chellappa, 1981], different pseudo random sequences produce very small perturbations in the patterns.

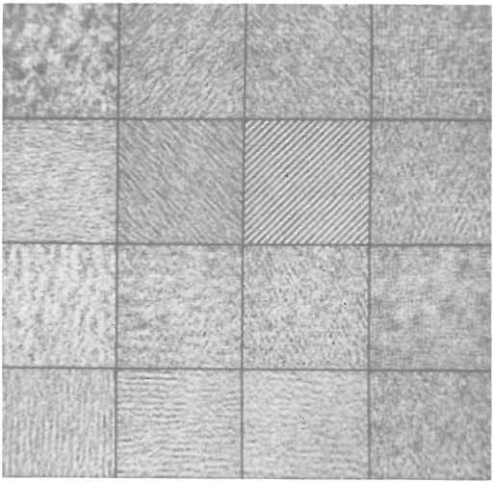


Fig. 3.1 Synthetic patterns generated by GMRF models in Table 3.1.

4. ESTIMATION IN GMRF MODELS

4.1 Introduction

In any practical application of GMRF models, two problems have to be tackled, namely, given the structure of the model, methods for estimating the parameters of the model and

TABLE 3.1 DETAILS OF GMRF MODELS CORRESPONDING TO SYNTHETIC IMAGES IN FIG. 3.1. FOR ALL OF THEM $\alpha = 30.0$, $\nu = 1.1111$

Model Number (row, column)	Neighbor Set N_S and Coefficients
(1,1)	(1,0), (0,1) .2794 .1825
(1,2)	(−1,1), (1,1) .28 −.14
(1,3)	(−1,1), (1,1) −.14 .28
(1,4)	(1,0), (1,−1), (0,1), (1,1) .3357 −.25 .3246 −.2126
(2,1)	(1,0), (1,−1), (0,1), (1,1), (2,0), (0,2) −.2061 .0536 −.2061 .0536 −.0123 −.0580
(2,2)	(1,−1), (1,1), (2,0), (0,2), (2,−2), (2,2) −.2341 .4682 .0655 .0655 −.0655 −.0655
(2,3)	(−1,1), (1,1) .28 −.22
(2,4)	(1,0), (0,1), (2,0), (0,2), (3,0),(0,3),(4,0),(0,4) .12 −.10 .08 −.09 −.11 .11 −07 .09
(3,1)	(1,0), (0,1), (3,0), (0,3) .16 .10 .12 −.14
(3,2)	(1,0), (0,1), (3,0), (0,3) .10 .16 −.14 .12
(3,3)	(1,0), (0,1), (1,1), (−1,1), (3,0), (0,3) .12 −.10 .08 −.09 −.11 .11
(3,4)	(0,2), (2,0) .2794 .1825
(4,1)	(1,0), (0,1), (2,0), (0,2) .12 −.24 .16 −.18
(4,2)	(1,0), (0,1), (2,0), (0,2) −.24 .12 −.18 .16
(4,3)	(1,0), (0,1), (1,1), (2,0), (0,2) −.24 .12 .11 −.18 .16
(4,4)	(0,1), (1,0) −.12 .18

secondly, estimating the structure of the appropriate model itself. By first assuming that the neighbor set N of the GMRF model is known, we discuss several estimation schemes for GMRF models. Particularly, we stress the asymptotic statistical properties like consistency and efficiency of these estimates. The problem of choosing appropriate GMRF model for the given data is considered in Section 5.

Since in noncausal GMRF models, the expectation of $y(s)$ conditioned on $y(r)$, $r \neq s$ is a function of $y(s+r)$, $r \in N$,

$$p(y(s),\ s \in \Omega) \neq \prod_{s \in \Omega} p(y(s) |\ \text{all } y(s+r),\ r \in N),$$

Hence, it is extremely difficult to write the likelihood function $p(y(s), s \in \Omega)$ except in Gaussian situations. In Gaussian case, one can easily write an expression for log $p(y(s),\ s \in \Omega\ |\underline{\theta}, \nu))$ by evaluating the mean and covariances of $\{y(s)\}$. In general, due to the fact that the Jacobian of the transformation matrix $B(\underline{\theta})$ in (2.24) is not unity, but a complicated function of $\underline{\theta}$ for

GMRF models, log $p(y(s), s\varepsilon\Omega|\underline{\theta},\nu)$ is computationally involved for infinite lattice GMRF models. However, for the special case of finite lattice GMRF models, discussed in Section 2.3.2, one can evaluate easily the Jacobian of the transformation matrix and hence the likelihood function. The resulting likelihood function is a non-quadratic function of the parameters necessitating the use of numerical optimization algorithms.

To avoid the difficulty associated with obtaining the likelihood function in a general case and the associated computations, an ingenious estimation scheme known as the coding method was developed in [Besag, 1972; 1974]. This method is useful for both Gaussian and non-Gaussian models. However, the coding estimate is not as efficient as the ML estimate as it only uses part of the given data. An estimate whose efficiency lies between that of the coding and ML estimates for Gaussian GMRF models was analyzed in [Kashyap & Chellappa, 1983]; this estimate was recommended in an earlier paper [Woods, 1972].

In this section, we discuss several estimation schemes mentioned above and give the results of the LS method applied to synthetic data generated by known GMRF model. A comparison of efficiencies of the coding, LS and ML estimates is given for a simple isotropic GMRF model with $N=\{(0,-1), (0,1), (-1,0), (1,0)\}$.

4.2 Coding Method

Assume that the observations $\{y(s)\}$ obey the GMRF model

$$y(s) = \sum_{r\varepsilon N_s} \theta_r[y(s+r) + y(s-r)] + e(s), \forall s,$$

where $\{e(s)\}$ is a zero mean correlated noise sequence with variance ν and correlation structure given in (2.16). Consider the case of GMRF model with $N_s = \{(0,1), (1,0)\}$, characterized by $\underline{\theta} = \text{Col.}\ [\theta_r, r\varepsilon N_s]$. The conditional distribution $p(y(s)|y(s+r'),$ all $r'\varepsilon N)$ is of the form,

$$p(y(s)\ |y(s+r),\ r\varepsilon N) = \frac{1}{(2\pi\nu)^{1/2}} \exp[-\frac{1}{2\nu}(y(s)- \sum_{r\varepsilon N} \theta_r\ y(s+r)^2]$$

and the coding estimate $\underline{\theta}'_c$ is given by

$$\underline{\theta}'_c = [\sum_{\Omega_0} \underline{q}(s)\underline{q}^T(s)]^{-1}\ (\sum_{\Omega_0} \underline{q}(s)y(s))$$

where

$$\underline{q}(s) = \text{Col.}[y(s+r') + y(s-r'),\ r'\varepsilon N_s]$$

and Ω_0 is a subset of Ω (consisting of sites marked X in Fig. 4.1). The coding scheme yields another estimate similar to $\underline{\theta}'_c$, say $\underline{\theta}''_c$, with the sum being evaluated over $\Omega - \Omega_0$. One of the main disadvantages of this method is that the estimates thus obtained are not efficient [Moran & Besag, 1975] due to the partial utilization (50%) of the data. Also, for a particular GMRF model, more than one coding scheme can be realized, yielding several estimates likely to be highly dependent. For instance, in the Mercer-Hall wheat data and a GMRF model with $N_s = \{(0,1), (1,0), (1,1), (-1,1)\}$ the estimates of $\theta_{0,1}$ obtained by four possible coding schemes are •043, •085, •243, •236 [Besag, 1974] and a simple averaging of these highly dependent estimates is not satisfactory.

4.3 A Consistent LS Estimation Scheme

Consider the estimates

$$\underline{\theta}^* = [\sum_{\Omega_1} \underline{q}(s)\ \underline{q}^T(s)]^{-1}\ (\sum_{\Omega_1} \underline{q}(s)y(s)) \tag{4.1}$$

$$\nu^* = \frac{1}{M^2} \sum_{\Omega_I} (y(s) - \underline{\theta}^{*T} \underline{q}(s))^2, \tag{4.2}$$

where Ω_I is the region of Ω containing the interior pixels. The estimate $\underline{\theta}^*$ obtained is an improvement over $\underline{\theta}'_c$ or θ''_c. The statistical properties of this estimate derived in [Kashyap & Chellappa, 1983] are summarized below in a theorem.

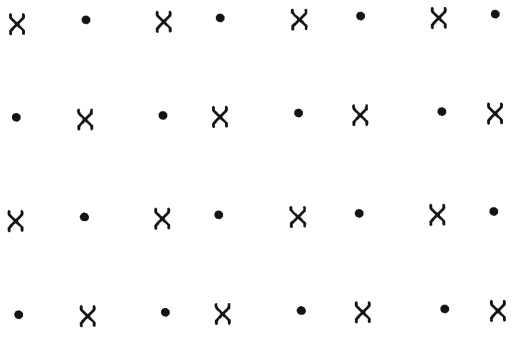


Fig. 4.1 Coding pattern for a first-order scheme [Besag, 1974].

Theorem 4.1: Let $y(s)$, $s \in \Omega$ be the set of observations obeying the GMRF model (2.15). Then

1. The estimate $\underline{\theta}^*$ is asymptotically consistent.

2. The asymptotic covariance matrix of $\underline{\theta}^*$, is

$$E(\underline{\theta}-\underline{\theta}^*)(\underline{\theta}-\underline{\theta}^*)^T = \frac{1}{M^2}\left[\nu\, \underline{Q}^{-1} + 2\nu^2(\underline{Q}^T\underline{Q})^{-1} - \frac{\nu}{M^2}\underline{Q}^{-1}\sum_s \sum_{\substack{r \\ (s-r)\in N}} \theta_{(s-r)} \underline{W}_{r,s}(\underline{Q}^T)^{-1}\right]$$

where

$$\underline{Q} = E[\underline{q}(s)\underline{q}^T(s)]$$

and

$$\underline{W}_{r,s} = E[\underline{q}(r)\underline{q}^T(s)]$$

For the isotropic conditional model with $N_s = \{(0,1), (1,0)\}$, the asymptotic expected mean square error is

$$E(\theta-\theta^*)^2 = \frac{2\theta^2(1-4\theta\rho_{1,0})^2}{4M^2\rho^2_{1,0}} \quad \text{where} \quad \rho_{1,0} = \frac{\text{cov}(y(s),y(s+(1,0)))}{\text{cov}(y^2(s))} \tag{4.3}$$

The elements of matrices $\underline{Q}$ and $\underline{W}_{r,s}$ are functions of normalized autocorrelation coefficients $\rho_{k,1}$.

We give the results of applying the LS estimation scheme to some synthetic data generated from a known GMRF model. The synthetic generation scheme in Section 3 was used. The results of the estimation scheme are given below.

Experiment 4.1: (Gaussian Data and a GMRF model with $N_s = \{(-1,1), (1,1)\}$.

The true model is defined as follows: $\nu = 1.1111$, $\theta_{-1,1} = -.14$, and $\theta_{1,1} = .28$. Using this model, synthetic data was generated. For estimation of parameters, consistent estimation scheme in (4.1), (4.2) was used. The actual values of the estimates of the parameters are $\theta^*_{-1,1} = -.1410$, $\theta^*_{1,1} = 0.27875$ and $\nu^* = 1.1033$ which are numerically close to the true parameters. As a consequence, the pattern generated by the GMRF model with estimates of parameters is close to the true pattern.

4.4 ML Estimation of Parameters

The ML estimates are obtained by minimizing the function, [Kunsch, 1981]

$$(-2/M^2)\log p(\underline{y}|\underline{\theta},\nu) = \frac{1}{(2\pi)^2}\int[\log S(\lambda,\underline{\theta})]\, d\lambda + (1/\nu)[C_0 - 2\sum_{r\in N_s}\theta_r C_r] + \log 2\pi \tag{4.4}$$

where

$$S(\lambda,\underline{\theta}) = \frac{\nu}{(1-2\sum_{r\in N_s}\theta_r \cos \lambda^T \cdot r)}, \tag{4.5}$$

and C_r are the sample correlations

$$C_r = \frac{1}{M^2}\sum_{s\in\Omega} y(s)y(s+r)$$

The estimates obtained by minimizing (4.4) should satisfy

$$(1-2\sum_{r\in N_s}\theta_r \cos \lambda^T \cdot r) > 0$$

for the model to be stable. The ML estimate $\hat{\underline{\theta}}$ thus obtained also satisfy

$$\frac{1}{4\pi^2}\int \cos(\lambda^T \cdot r) S(\lambda,\hat{\underline{\theta}})\, d\lambda = C_r, \tag{4.6}$$

The ML estimate are computationally involved due to the evaluation of the integral. Preliminary results indicate [Sharma & Chellappa, 1983] that convergence can be obtained for first- and second-order MRF models in about 30-40 iterations. The IMSL routine ZXMIN was used in this study for parameter estimation.

An usefulness of (4.6) is in developing a GMRF model based approach for 2-D maximum entropy power spectrum (MEPS) estimation. Using the fact that the 2-D MEPS has a structure similar to that of the spectrum (4.5) of GMRF models and (4.6), a model based approach has been developed [Sharma & Chellappa, 1983] recently for 2-D MEPS estimation. The ML estimation scheme may not be practical for image and signal processing applications involving large values of M. By using special boundary conditions, computational load can be reduced as discussed in the next section.

4.5 Estimation of Parameters in Finite Lattice Models

The coding method and LS estimation scheme in Section 4.3 can be directly extended to finite toroidal lattices by summing over Ω. Numerical simulations indicate that the differences in the values of $\underline{\theta}^*$ for infinite and finite toroidal lattice representations are

negligible. The ML estimate for general GMRF models can be obtained by assuming the toroidal lattice representation for {y(s)} and Gaussian structure for {e(s)} and writing down the log-likelihood function. For the toroidal representation corresponding to (2.22) the log-likelihood function log $p(\underline{y}|\underline{\theta},\nu)$ can be written as [Moran & Besag, 1975],

$$\log p(\underline{y}|\underline{\theta},\nu) = \sum_{s\in\Omega} \log (1-2\underline{\theta}^T\underline{\Phi}_s) - (M^2/2) \log 2\pi\nu - \frac{1}{2\nu}\underline{y}^T B(\underline{\theta})\underline{y} \tag{4.7}$$

Note that the contribution of the exponent term of the probability density function is linear in $\underline{\theta}$ unlike the case of SAR models, where this term is quadratic [Whittle, 1954; Kashyap & Chellappa, 1983]. Numerical optimization procedures like Newton-Raphson can be used to obtain the ML estimates. Since the summation term can be computed using 2-D FFT, (4.7) is easier to evaluate compared to (4.5). An asymptotic technique for 2-D MEPS estimation using toroidal representation is in [Chellappa, Hu & Kung, 1983].

Similarly likelihood functions can be written down for GMRF models represented on other types of finite lattices. For instance, [Kashyap, 1981a] suppose we divide the finite lattice Ω into two mutually exclusive and totally inclusive subsets Ω_I, the interior set and Ω_B, the boundary set.

$\Omega_B = \{s = (i,j);\ s\in\Omega$ and $(s+r) \notin \Omega$ for any member $r\in N\}$

$\Omega_I = \Omega - \Omega_B$

Let the GMRF model be characterized by a neighbor set $N = \{s_1=(0,1), \bar{s}_1=(0,-1), s_2=(1,0), \bar{s}_2=(-1,0)\}$. Consider

$$y(s) = \sum_{i=1}^{2} \theta_i(y(s+s_i) + y(s+\bar{s}_i)) + e(s),\ s\in\Omega_I$$

and

$$y(s) = \sum_{i=1}^{2} \theta_i(y_1(s+s_i) + y_1(s+\bar{s}_i)) + e(s),\ s\in\Omega_B$$

where

$$y_1(s+\bar{s}_i) + y_1(s+s_i) = 2\ y(s+s_i) \text{ if } (s+s_i)\in\Omega,\ (s+\bar{s}_i)\notin\Omega$$

$$= 2\ y(s+\bar{s}_i) \text{ if } (s+s_i)\notin\Omega,\ (s+\bar{s}_i)\in\Omega$$

$$= y(s+s_i) + y(s+\bar{s}_i), \text{ otherwise}$$

The above equations can be written in the form of

$$B(\underline{\theta})\ \underline{y} = \underline{e}$$

where $\underline{y}$ and $\underline{e}$ are $M^2 \times 1$ vectors of arrays {y(s)} and {e(s)}. One can show [Kashyap, 1981a] that the eigenvalues $\mu_{i,j}$, $(i,j)\in\Omega$ of $B(\underline{\theta})$ are

$$\mu_{ij} = 1 + 2\theta_1\phi_1 + 2\theta_2\phi_j$$

where

$$\phi_j = \cos(j\pi/M-1)$$

with the corresponding eigenvector

$$\underline{n}_{ij} = \text{Col.}\ [\eta_0(\phi_j)\ \eta^i,\ \eta_1(\phi_j)\eta^i,\ldots,\eta_{M-1}(\phi_j)\eta^i],\ 0 \le i, j \le M-1$$

$$\eta^j = \text{Col.}\ [\eta_0(\phi_j),\ \eta_1(\phi_j),\ldots,\eta_{M-1}(\phi_j)],\ \text{an M-vector and}$$

$$\eta_i(\phi_j) = \cos(j\ i\ \pi/M-1),\ i,j = 0,1,\ldots,M-1.$$

Hence, the exact likelihood function for this finite model is

$$\log p(\underline{y}|\underline{\theta},\nu) = \frac{1}{2}\sum_{(i,j)\in\Omega}\log \mu_{ij} - (M^2/2)\log 2\pi\nu \quad -\frac{1}{2\nu}\sum_{(i,j)\in\Omega}\|z(\lambda_{ij})\|^2\mu_{ij}, \tag{4.8}$$

where $z(\lambda_{i,j})$ are the 2-D discrete cosine transform of $\{y(s)\}$, defined by the eigenvectors of $B(\underline{\theta})$ given in (2.24). However, it may be noted that the likelihood function in (4.8) is exact only for the particular neighbor set $N = \{(0,1), (0,-1), (-1,0), (1,0)\}$. Similar likelihood function using discrete sine transforms may be obtained for the finite lattice model

$$y(s) = \sum_{i=1}^{2}\theta_i(y(s+s_i) + y(s+\bar{s}_i)) + e(s),\ s\in\Omega_I$$

and

$$y(s) = \sum_{i=1}^{2}\theta_i(y_1(s+s_i) + y_1(s+\bar{s}_i)) + e(s),\ s\in\Omega_B$$

where

$$y_1(s+s_i) + y_1(s+\bar{s}_i) = y(s+s_i) \text{ if } (s+\bar{s}_i) \notin \Omega$$

$$= y(s+\bar{s}_i) \text{ if } (s+s_i) \notin \Omega$$

The families of finite lattice GMRF models having sine or cosine eigen-vectors is not as rich as that of finite lattice GMRF models having discrete Fourier eigenvectors, in that the latter representation is valid for arbitrary symmetric N while the former two are valid only for symmetric N involving nearest neighbors. Since in any given application, models with neighbor sets more general than the nearest neighbors may be required, one may obtain ML estimates for these models using toroidal assumption.

4.6. Comparison of Estimates

We compare the asymptotic variance of the estimate (4.1), with the asymptotic variances of the coding estimate and ML estimate for the isotropic conditional model with $N = \{(0,1), (1,0)\}$. From [Moran & Besag, 1975], the asymptotic variance of the coding estimate is

$$M^2 \text{ Var } (\theta'_c) = \frac{(1-4\rho_{1,0})}{2\rho_{1,0}},$$

Also from [Moran & Besag, 1975], the variance of the ML estimate, $\hat{\theta}_{ML}$ is

$$\text{Var } (\hat{\theta}_{ML}) = \frac{0.5}{M^2(I(\theta)-4V^2_{10}(\theta))},$$

where

$$I(\theta) = \frac{1}{4\pi^2}\int_0^{2\pi}\int \frac{(\cos x+\cos y)^2\, dxdy}{(1-2\theta(\cos x+\cos y))^2}, \tag{4.9}$$

and

$$V_{st}(\theta) = \frac{1}{4\pi^2}\int_0^{2\pi}\int \frac{\cos(sx+ty)\, dxdy}{(1-2\theta(\cos x+\cos y))}, \tag{4.10}$$

Tabulated values of $V_{10}(\theta)$, $\alpha_{1,0}$ and $I(\theta)$ are available in [Moran & Besag, 1975] for different values of θ. Using these values, and (4.3), (4.9), and (4.10), the columns 2-4 of Table 4.1 are

TABLE 4.1 COMPUTATION OF ASYMPTOTIC VARIANCES AND EFFICIENCIES OF DIFFERENT ESTIMATES IN ISOTROPIC CONDITIONAL MODEL WITH $N_S = \{(0,1),(1,0)\}$

4θ	$M^2 var(\hat{\theta}_{ML})$	$M^2 var(\theta_C)$	$M^2 var(\theta^*)$	$eff(\theta_C)$	$eff(\theta^*)$
.1	.4928	.497	.494	.991	.9975
.2	.472	.489	.478	.965	.987
.3	.437	.474	.450	.921	.971
.4	.390	.454	.412	.859	.946
.5	.333	.427	.365	.779	.912
.6	.267	.393	.309	.681	.864
.7	.197	.349	.244	.564	.807
.8	.1243	.296	.1753	.419	.709
.9	.0556	.224	.1004	.248	.553

*Columns 2, 3, and 5 are from [Moran & Besag, 1975].

computed. The asymptotic efficiencies, in columns 5 and 6, are defined as below:

$$eff(\theta'_c) = Var(\hat{\theta}_{ML})/Var(\theta'_c)$$

and

$$eff\ (\theta^*) = Var\ (\hat{\theta}_{ML})/Var(\theta^*)$$

It is evident that the estimate θ^* computed using (4.1) is more efficient than the coding estimate but is not as good as ML estimate.

5. DECISION RULES FOR THE CHOICE OF APPROPRIATE GMRF MODEL

5.1 Motivation and Possible Approaches

One of the problems to be tackled in fitting a model to the given image data is the selection of the order of the model to be used. Since the structure of the GMRF model is defined by the underlying neighbor set N of the model, the problem is to decide the appropriate N to be used.

From 1-D time series analysis [Box & Jenkins, 1976], it is known that a model of appropriate order should be fitted to obtain good results in applications like forecasting and control. A similar situation is true in the case of 2-D GMRF models. In fact, the problem becomes more difficult due to the rich variety of model structures. For instance, within the class of GMRF models, different neighbor sets account for different image patterns as shown in Section 3 and the quality of the reconstructed image varies considerably depending on how similar the underlying model is to the true model and hence the use of appropriate neighbor set is important.

The possible approaches are using pairwise hypothesis testing [Anderson, 1962], Akaike's information criterion (AIC), [Akaike, 1974], Bayes approach [Kashyap, 1977; Schwarz, 1978], and Hannan's procedure [Hannan & Quinn, 1979]. The pairwise hypothesis testing method has been used for GMRF models in the modeling of field data [Besag, 1974] and textures [Cross & Jain, 1983]. The main criticisms of this approach are that the resulting decision rules are not always transitive, i.e., if a model C_1 is preferred to C_2 and C_2 is preferred to C_3 then it does not follow always that C_1 is preferred to C_3 [Kashyap, 1977]. Further, the decision rules are not consistent, i.e., the probability of choosing an incorrect model does not go to zero even as the number of observations goes to infinity.

The model selection problem comes under the category of multiple decision problem. A method that is well suited for this problem is to compute a test statistic for different models and choose the one corresponding to the minimum. The AIC criterion and the Bayes method are two such procedures. The AIC test statistics can be computed for GMRF models from the expression of the log-likelihood function given (4.5) or (4.7); the best model is the one which minimizes the AIC statistic. The AIC method, in general gives transitive decision rules but is not consistent even for one-dimensional autoregressive models [Kashyap, 1980].

In the Bayes approach of fitting models to the image, various possible models are postulated as mutually exclusive hypotheses C_i, $1 \leq i \leq k$. The hypothesis that maximizes the posterior probability density $P(C_i|y(s),s\epsilon\Omega)$ is chosen as the appropriate model with minimum probability of error. This approach involves obtaining an expression for the likelihood of the observations and integrating it over the parameters using an appropriate prior probability density function. One can also develop a decision rule for assigning $\{y(s)\}$ to one of the hypotheses C_i, i=1,...,k so as to minimize a suitable criterion function and also determine the probability of error associated with the decision. The loss function can be chosen to reflect the particular needs of the problem, such as forecasting or estimation of spectral density [Kashyap, 1977].

In this section, we give a decision rule obtained using Bayesian method for finding an appropriate GMRF model. We assume that the given observation set could have possibly been generated by one of k mutually exclusive models or hypotheses denoted by C_i, $1 \leq i \leq k$. These hypotheses are GMRF models of different neighbor sets. Under this assumption one can write the expression for joint density of observations, as described in Section 4 and integrate this probability density function over the parameter space using a regular prior probability density function and asymptotic integration [Lindley, 1961]. Using this expression and the prior probabilities $P(C_i)$, i=1,...,k, of the hypotheses, a decision rule for choosing a model with minimum probability of error can be designed. If the prior probability densities are given, one can explicitly compute the posterior densities and take appropriate decisions. For situations where prior densities are not known explicitly, we suggest using that portion of the posterior density function, which does not involve the prior densities. Though this rule does not have the minimum error rate property, it is asymptotically weakly consistent.

5.2 Decision Rules

Currently, known methods for choosing an appropriate GMRF model from a class of such models use pairwise hypothesis testing procedures employing coding estimates. In addition to being non-transitive and inconsistent, this method has several drawbacks due to the usage of coding estimates. Since more than one coding estimate results for a GMRF model, it is quite possible that a GMRF model gets accepted or rejected when different coding estimates are used. Further, when we are choosing between a first-order GMRF model and a second-order GMRF model, to ensure that the likelihoods are comparable, the coding schemes corresponding to the higher order model should be used leading to the use of highly inefficient estimates of the lower order model. Decisions rules that are transitive, asymptotically consistent and which do not possess the above mentioned drawbacks can be derived by using the Bayes procedure. Instead of getting into details, we simply state the problem, give the decision rule and simulation results.

Suppose we have k sets $N_{s1},\ldots N_{sk}$ of neighbors containing $m_1, m_2,\ldots,m_k$ members, respectively. Corresponding to each N_{si}, we write the GMRF model as

$$y(s) = \sum_{r\epsilon N_{si}} \theta_{ir} \, (y(s+r) \oplus y(s-r)) + \sqrt{\nu_i}\, e(s), \; s\epsilon\Omega$$

$\oplus$ denotes modulo M along both the co-ordinate axes, $\theta_{ir} \neq 0$, $r\epsilon N_{si}$, $\nu_i > 0$, i=1,...,k and e(s) is Gaussian. Then, the decision rule for the choice of appropriate neighbors is: [Kashyap & Chellappa, 1983] choose the neighbor set N_{si}^* if,

$$i^* = \text{Argument} \min_n \{g_n\}$$

where

$$g_n = -2 \sum_{s \in \Omega} \log(1 - \underline{\theta}_n^{*T} \underline{\phi}_{ns}) + M^2 \log \nu_n^* + m_n \log (M^2), \tag{5.1}$$

and

$$\underline{\theta}_n^{*T} = \text{Col.}\,[\theta_{rn}^*,\ r \in N_{sn}]$$

$$\underline{\phi}_s = \text{Col.}\,[\cos \frac{2\pi}{M} (s^T r),\ r \in N_{sn}]$$

The model selection procedure consists of computing g_n for different models and choosing the one corresponding to the lowest g_n. One can write similar test statistics for general GMRF models using (4.4).

To illustrate the usefulness of the decision rule in (5.1), we consider the synthetic data generated by the GMRF model with $N_s = \{(-1,1), (1,1)\}$ considered in Section 4.3. The test statistics g_n in (5.1) were computed for each of the fitted models in Table 5.1. The decision rule correctly picks up the true model. To illustrate the usefulness of the decision rule for choosing GMRF models in image modeling, synthetic patterns corresponding to the models in Table 5.1 were generated in [Chellappa, 1981]. It was found that the patterns corresponding to the inappropriate models 1,2, and 4 are not similar to the original pattern. The patterns

TABLE 5.1 DETAILS OF GMRF MODELS FITTED TO THE GAUSSIAN DATA GENERATED BY $N_S = \{(-1,1),(1,1)\}$, $\nu = 1.1111$, $\theta_{-1,1} = -.14$, $\theta_{1,1} = .28$

Number	Neighbor Set N_S	$\hat{\nu}$	Estimate of Coefficients	Test Statistics g_n
1	(1,1)	1.1638	$\theta_{1,1} = .3116$	1575.3
2	(−1,1)	1.3520	$\theta_{-1,1} = .2093$	1628.9
3 (True Model)	(−1,1), (1,1)	1.1033	$\theta_{-1,1} = -.1410$, $\theta_{1,1} = .27875$	1464.30
4	(0,1), (1,0)	1.4934	$\theta_{0,1} = -.0052$, $\theta_{1,0} = -.0020$	1659.7
5	(−1,1), (1,1), (0,1)	1.1033	$\theta_{-1,1} = -.14101$, $\theta_{1,1} = .27877$, $\theta_{1,0} = -.0051$	1472.7
6	(−1,1), (1,1), (0,1)	1.1033	$\theta_{-1,1} = -.1410$, $\theta_{1,1} = .27873$, $\theta_{0,1} = -.001717$	1472.2
7	(−1,1), (1,1) (0,1), (1,0)	1.1033	$\theta_{-1,1} = -.14101$, $\theta_{1,1} = .27876$, $\theta_{1,0} = .0049$, $\theta_{0,1} = -.0009$	1481.0

corresponding to models 5,6, and 7 are similar to the original pattern. In general, the patterns corresponding to a specific model A and another model, which includes all the neighbors in model A and some extra neighbors appear very similar, making visusal judgement subjective. However, the quantiative decision rule correctly rejects these overparametrized models.

Another test statistic for the choice of GMRF models can be given by extending Hannan's results [Hannan & Quinn, 1979] for GMRF models. The decision rule is to choose the neighbor set N_{si}^* if,

$$i^* = \text{Argument min } \{g_n\}$$

where

$$g_n = -2 \sum_{s \in \Omega} \log (1 - \underline{\theta}_n^{*T} \underline{\phi}_{n,s}) + M^2 \log \nu_n^* + 2m_n \, c \log \log M^2, \; c > 1 \tag{5.2}$$

Since log log M^2 increases with M^2slower than log M, the probability of choosing a higher order model is less when (5.2) is used.

6. TEXTURE SYNTHESIS AND CODING

In Section 3, it was shown that the class of GMRF models is capable of generating a wide variety of image patterns which exhibit local replication attribute, an essential ingredient of texture. In this section we give experimental results of synthesis of some 128 x 128 textures from Brodatz album. The synthetic textures generated using 12 parameter GMRF models retain most of the characteristics of the original textures. Before giving the experimental results, we justify the use of GMRF models for texture synthesis by showing that the theoretical variograms of many GMRF model possess an oscillatory behavior, a characteristic of variograms of many naturally occurring textures.

6.1 Theoretical Variograms of GMRF Models

The second-order properties of y(.) can be conveniently described in terms of the variogram V(r) at displacement r defined as

$$V(r) = E[y(s) - y(s+r)]^2$$

For stationary case,

$$V(r) = 2 R_0 (1 - \rho_r) \tag{6.1}$$

where ρ_r is the normalized autocorrelation function and

$$R_0 = E(y^2(s))$$

Since for the infinite lattice GMRF model in (2.15)

$$\rho_r = \frac{\int_{-0.5}^{0.5} \int \exp(\sqrt{-1}\,(2\pi/M)\,\lambda^T r) S(\lambda) d\lambda}{\int_{-0.5}^{0.5} \int S(\lambda) d\lambda}, \tag{6.2}$$

where

$$S(\lambda) = \frac{\nu}{(1 - 2\underline{\theta}^T \underline{\phi}_\lambda)}$$

and

$$\underline{\phi}_\lambda = \text{Col. } [\cos (2\pi/M)\,\lambda^T r, \; r \in N_s]$$

the variogram at an arbitrary displacement may be obtained using (6.2). Simple expressions (not involving numerical integration) for the variograms may be obtained for GMRF model in (2.22); specifically, the normalized autocorrelation function ρ_r at lag r can be written as

$$\rho_r = \frac{\sum_{s \in \Omega} \exp(\sqrt{-1}\ (2\pi/M)\ s^T r)/\mu_s}{\sum_{s \in \Omega} 1/\mu_s}$$

where μ_s is in (2.24). The variogram V(r) related to ρ_r as in (6.1) are plotted in Fig. 6.1 for several GMRF models. The details of the models are given [Chellappa, 1981]. For each model we have plotted variograms along the four directions r = (o,i), (i,o), (i,i) and (i,-i) in a discrete lattice. The structure of the possible variograms of 2-D GMRF models is quite varied and many of them possess an oscillatory behavior, a characteristic of the variograms of natural textures [Schacter, et al., 1978].

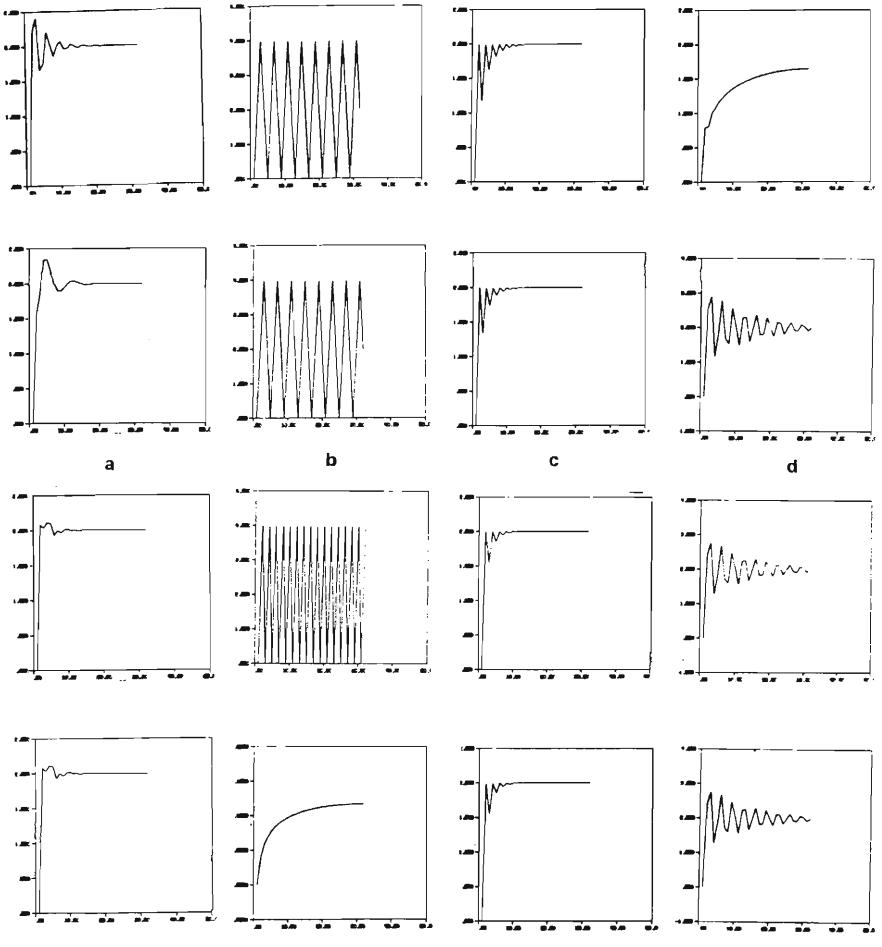


Fig. 6.1 The theoretical variograms of some GMRF models. V(o,k) versus k is plotted in the first row, V(k,o) versus k in the second row, V(k,k) versus k in the third row, and V(k,-k) versus k in the fourth row.

6.2 Real Texture Synthesis and Coding

Our discussions in Sections 3 and 6.1 have been concerned with synthetic image patterns and variograms of simulated GMRF models. The next step is to investigate the appropriateness of MRF models for four 128 x 128 natural textures wood, grass, tree bark, and plastic bubble from Brodatz album. The gray levels of the texture are in the range 0–255. We give the results of fitting models for textures as a sequence of experiments. In the first experiment a fourth-order GMRF model with N_s = {(0,1), (1,0), (−1,1), (1,1), (2,0), (0,2), (−2,1), (1,2), (1,−2), (2,1)} was fitted to wood, grass, tree bark, and an eighth-order model was fitted to plastic bubble. To synthesize natural textures pseudo random Gaussian numbers were used instead of $\underline{n}$ in (3.4). The original and synthesized textures are given in the top rows of Figs. 6.2–6.5. Note that we have to store only 12 (10 dimensional vector θ^*, α^* and v^* where α^* is the sample mean) parameters for the fourth-order and 24 parameters for the eighth-order model. The generated wood texture retains the dominant vertical streakline patterns. The synthesized grass and plastic buble are not very close to the original texture.

When a GMRF model is fitted to the given texture, the information contained in the orginal texture is split between the estimates of parameters of the model and the residuals defined as

$$\hat{\underline{n}} = \sqrt{H(\underline{\theta}^*)}\underline{y},$$

In the previous experiment only the parameters of the fitted GMRF model were used to generate the texture. To the extent that the GMRF model or toroidal lattice is appropriate, the components of $\underline{n}$ are uncorrelated and hence can be represented by much fewer number of bits compared to the original texture. In Figs. 6.2–6.5 (bottom left), we have given the synthetic textures generated using the fourth- and eighth-order models and residuals quantized to one bit. For quantization, the Max quantizer [Max, 1960] for Gaussian residuals was used. Since there are 128 x 128 residuals each represented by 1 bit and 12 parameters of the fourth-order model each represented by 32 bits, the compression factor is around 7.816 for wood, grass, and tree bark; for plastic bubble the comparison factor is 7.641 as an eight-order model is required. However, by only retaining half of the total number of quantized residuals (only every other quantized residual is used) and using a pseudo random number to generate the other of the residuals one can almost double the compressions factors. The synthetic textures generated using this procedure are given in the bottom right windows of Figs. 6.2–6.5.

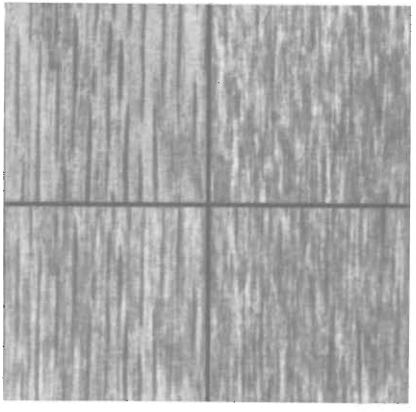


Fig. 6.2 Synthesis of wood texture using a fourth-order GMRF model (12 parameters). Top left: original, top right: synthesized using the model parameters and a pseudo random number array. Bottom left: synthesized using the model parameters and residuals quantized to 1 bit each. Bottom right: only half the residuals are used.

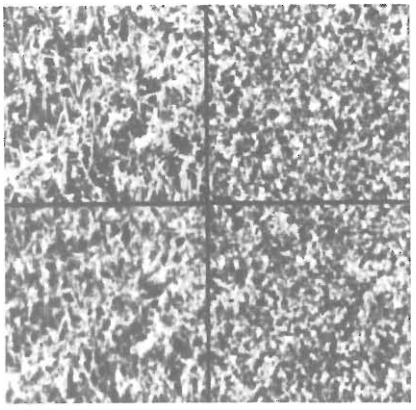


Fig. 6.3 Same as Fig. 6.2 for the grass texture

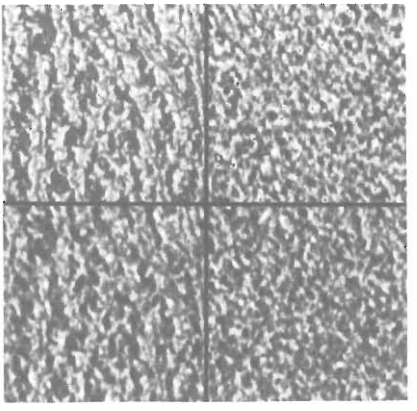


Fig. 6.4 Same as in Fig. 6.2 for the tree bark texture.

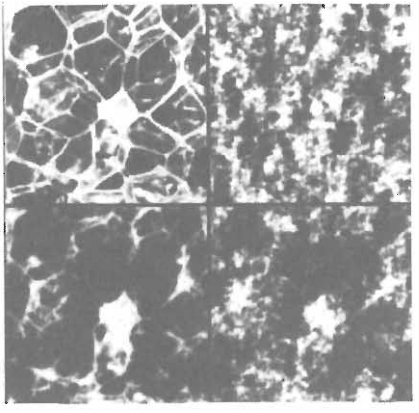


Fig. 6.5 Same as in Fig. 6.2 for the plastic bubble texture with an eighth-order model.

7. CLASSIFICATION OF TEXTURES

7.1 Preliminaries

Textural and contextual features are two important pattern elements in human interpretation of data. There are many applications like identification of large scale geological information, land use patterns and interpretation of aerial images where texture classification is an important element. A good review of literature on this subject can be found in [Haralick, 1979]. The texture classification problem can be stated as follows: there are a finite number of classes C_i, i= 1,2,...,r. A number of training textures belonging to each class are available. Based on the information extracted from these sets a rule is designed which classifies the given test image of unknown class to one of r classes. The key step in a classification problem is the choice of features which reduces the dimension of data to a computationally reasonable amount while preserving much of the information present in the actual data. Currently used methods utilize features from among the categories given below.

1. Features derived form second-order gray level statistics including the gray level co-occurrence matrices [Haralick, et al., 1973]

2. Fourier Power Spectrum [Weszka, et al., 1976]

3. Gray level difference statistics [Weszka, et al., 1976]

4. Gray level run length statistics [Weszka, et al., 1976]

5. Decorrelation methods [Faugeras, et al., 1980]

There are several drawbacks associated with the feature sets mentioned above. The justification of the use of any feature set is its ability to preserve all the relevant information contained in the M^2 pixels so that an image close to the original can be generated from the feature set. Such features are called as information preserving features. Since it is impossible to generate an image even close to the original image using any of the above mentioned features, these features are not information preserving. The success of information preserving features has been amply demonstrated in speech recognition using linear predictive coefficients and one can hope similar approaches might be successful in 2-D texture classification.

The decorrelation method feature extraction [Faugeras, et al., 1980] using Laplacian window has the underlying modeling interpretation: suppose the texture is represented by a 2-D separable, causal, Markov model

$$y(s) = \theta_1 y(s+(-1,0)) + \theta_2 y(s+(0,-1)) - \theta_1\theta_2 y(s+(-1,-1)) + \sqrt{\beta}\, w(s)$$

Then as θ_1, $\theta_2 \to 1$ decorrelated residuals generated by the Laplacian operator are similar to those obtained by

$$\hat{\omega}(s) = y(s) - \theta_1^* y(s+(-1,0)) - \theta_2^* y(s+(0,-1)) + \theta_1^*\theta_2^* y(s+(-1,-1))$$

where θ_1^*, θ_2^* and β^* are LS estimates. Thus, from the point of view of image modeling, in the limit, the decorrelation features are extracted from this moments of the residuals. The loss of information during the process of dimensionality reduction may be significant since residuals themselves contain only partial information in the original data. One should also add θ^*_1, θ^*_2 as feature components. In the following section we show how GMRF models can be used for feature selection and classification of textures.

7.2.1: Fourier Features:

The Fourier features can be extracted by fitting an GMRF model of appropriate neighbor set to the given texture. The power spectrum given by

$$S(\lambda = (\frac{2\pi i}{M}, \frac{2\pi j}{M})) = \frac{\nu^*}{(1-2\underline{\theta}^{T^*}\underline{\Phi}_{ij}} \qquad (7.1)$$

can be used to extract ring and wedge features as in [Weszka, et al., 1976]. In (7.11), ν^* and $\underline{\theta}^*$ are the LS estimates (4.1) and (4.2) and $\underline{\Phi}_{ij}$ is

$$\underline{\Phi}_{ij} = \text{Col.}\,[\cos 2\pi/M\,(ik + j1),\,(k,1)\,\varepsilon N_s]$$

Well documented results in 1-D spectral estimation [Childers, 1975] show that the spectral estimates obtained by fitting a model possess better statistical properties compared to the

periodogram estimates. Thus, it may be expected that textural features extracted from S(.) in (7.1) perform better than the periodogram features.

7.2.2: Feature Extraction Using Model Parameters

To extract decorrelation features we simply fit an GMRF model (2.22) to the given texture and obtain the residuals

$$\hat{\underline{n}} = \sqrt{H(\underline{\theta}^*)}\,\underline{y}, \tag{7.2}$$

where $\underline{y}$ denotes the M^2 vector constructed from the texture. The decorrelation features can be extracted from the moments of $\hat{\underline{\nu}}$. Since the underlying modeling assumption is more general than the one corresponding to the Laplacian operator, the features derived from $\hat{\underline{n}}$ may perform better. One can also consider an expanded feature set using the estimates $\underline{\theta}^*$ as feature components. Since we can construct textures close to the original using $\underline{\theta}^*$ and ν^* alone (as discussed in Section 6.2), the set $(\underline{\theta}^*, \nu^*)$ is information preserving and hence should prove to be a good feature vector. A recent work using Bhattacharya distance between two textures obeying Gaussian GMRF models for texture classification is in [Kaneko & Yodogawa, 1982].

The transformation from the given texture to the feature vector is many-to-one and usually involves some loss of information. One can however derive a lossless feature vector for texture classification using GMRF models by using the notion of sufficient statistics [Kashyap, 1978]. Suppose that the vector $\underline{y}$ from the texture is represented by a Gaussian GMRF model (2.22) characterized by $\underline{\theta}$ and ν. Then the probability density of $\underline{y}$ has the form

$$p(\underline{y}|\underline{\theta},\nu) = \frac{(\det\,(B(\underline{\theta})))^{1/2}}{(2\pi\nu)^{M^2/2}} \exp\,\{-(1/2\nu)\,\underline{y}^T B(\underline{\theta})\,\underline{y}\}, \tag{7.3}$$

where $B(\underline{\theta})$ is specified in (2.24). Now define a sample correlation function

$$C_d(r) = M^{-2} \sum_{s\in\Omega} y(s)\,y(s+r).$$

The quadric form $\underline{y}^T B(\underline{\theta})\,\underline{y}$ in (7.3) can be simplified as

$$\underline{y}^T B(\underline{\theta})\,\underline{y} = M^2[C_d(0) - \sum_{r\in N_s} \theta_r C_d(r)] = M^2(C_d(0) - \underline{\theta}^T \underline{C}_d)$$

where

$$\underline{C}_d = \text{Col.}[C_d(r),\ r\in N_s]$$

Thus, (7.3) can be written as

$$p(\underline{y}|\underline{\theta},\ \nu) = \frac{(\det\,(B(\underline{\theta})))^{1/2}}{(2\pi\nu)^{M^2/2}} \exp\,\{\frac{M^2}{2\nu}(C_d(0) - \underline{\theta}^T \underline{C}_d)\} \tag{7.4}$$

Using the factorization theorem [Duda & Hart, 1973],

$$\underline{\xi} = (C_d(0),\ \underline{C}_d)$$

is a sufficient statistic for $(\underline{\theta},\ \nu)$ and hence is a lossless feature vector. One can now design Bayes rule or other standard classification rules like nearest neighbor and so on [Duda & Hart, 1973] using $\underline{\xi}$ as the feature vector. Note that the dimensionality of the feature vector is (m+1) where m is the number of independent θ_r's. Experiments are currently being done to determine the effectiveness of $\underline{\xi}$ as the feature vector.

8. IMAGE RESTORATION

The restoration of degraded images has many fields of application, including space and biomedical imagery. The literature on image restoration is too enormous to be listed in detail and there are many different methodologies like minimum mean-square error (MMSE) restoration, maximum a posteriori probability restoration, and maximum entropy restoration. In this paper, we are concerned with MMSE methods of restoration.

Suppose $\underline{y}$ and $\underline{x}$ represent lexicographic ordered arrays of the original and degraded image, related as in (8.1).

$$\underline{x} = \underline{H}\,\underline{y} + \underline{\eta} \tag{8.1}$$

In (8.1), $\underline{H}$ is the block-circulant matrix corresponding to the blur caused by a nonseparable, space-invariant point-spread function (PSF) and $\underline{\eta}$ is a signal independent additive white noise of variance γ. Then, it is well known [Andrews & Hunt, 1977] that $\hat{\underline{y}}$, the MMSE estimate of $\underline{y}$, can be written as

$$\hat{\underline{y}} = \underline{Q}_y\,\underline{H}^T(\underline{H}\underline{Q}_y\,\underline{H}^T + \gamma\underline{I})^{-1}\,\underline{x} \tag{8.2}$$

where $\underline{Q}_y$ is the covariance matrix of $\underline{y}$. There are two problems to be considered in the evaluation of $\hat{\underline{y}}$, namely, the determination of the covariance matrix $\underline{Q}_y$ and the inversion of the matrix in (8.1). For an image of size M x M, $\underline{Q}_y$ is of dimension $M^2 \times M^2$, and it is not uncommon to have M = 128 in typical restoration applications. Thus, some assumptions have to be made to reduce the computational load. In [Andrews & Hunt, 1977, ch. 7], the block-Toeplitz covariance matrix $\underline{Q}_y$ is approximated by a block-circulant matrix. Since block-circulant matrices possess an eigenfunction expansion in terms of Fourier vectors, fast Fourier transform (FFT) computations are used for the implementation of (8.2). In the stochastic representation of images by finite difference approximations of partial differential equations [Jain & Jain, 1978], appropriate assumptions are made regarding the model representation for $\underline{y}$ so that $\underline{Q}_y$ has a symmetric, tridiagonal Toeplitz form. Since symmetric tridiagonal matrices are diagonalized by sine transforms, fast algorithms for the implementation of (8.2) have been developed. Another approach [Woods & Radewan, 1977; Woods, 1978; Murphy & Silverman, 1978] is to assume an underlying causal model for the image $\underline{y}$ displayed as a state space model. Then $\hat{\underline{y}}$ could be computed by Kalman filter recursive algorithms.

The other problem in evaluating (8.2) is the determination of $\underline{Q}_y$. Most of the MMSE algorithms assume the availability of a prototype of the original image. Then the covariance matrix $\underline{Q}_y$ is evaluated from the prototype by making appropriate assumptions about the correlation structure of the prototype image. For instance, considerable attention has been paid to the exponential separable autocorrelation function, whose parameters are estimated from the prototype. In [Andrews & Hunt, 1977], $\underline{y}$ of (8.2) is written in terms of spectral density function (SDF) of the prototype image, which is often estimated from the periodogram of the prototype. In [Pratt & Davarian, 1977], the correlation function is assumed to be separable, corresponding to an underlying causal separable model. The method of estimating $\underline{Q}_y$ or equivalently, the parameters of the corresponding models, is arbitrary. We are interested in the use of GMRF models in developing nonrecursive restoration schemes. We represent the given degraded image, $\underline{x}$ in (8.1), by appropriate GMRF models and formulate the MMSE restoration problem. The representation on a toroidal lattice leads to covariance matrices having a block-circulant structure leading to fast implementation of filters using FFT algorithms.

8.2. Restoration Using GMRF Model for $\underline{x}$

Assume that the degradation is due to noise only, i.e.,

$$\underline{x} = \underline{y} + \underline{n} \tag{8.3}$$

Assume that $\underline{x}$ obeys GMRF model in

$$x(s) = \sum_{r \in N} \theta'_r \, x(s \oplus r) + e(s) \tag{8.4}$$

where $\{e(s)\}$ is the correlated noise sequence with correlation structure specified in (2.16). The covariance matrix $\underline{Q}_x$ of $\{x(.)\}$ is

$$\underline{Q}_x = \underline{Q}_y + \gamma \underline{T} \tag{8.5}$$

Substitution of $\underline{Q}_y$ form (8.5) into (8.2) with $\underline{H} = \underline{I}$ yields $\underline{y}$, an estimate of the original image

$$\hat{\underline{y}} = (\underline{Q}_y - \gamma \underline{I})^{-1} \, \underline{Q}_x^{-1} \underline{x} \tag{8.6}$$

$\underline{y}$ in (8.6) can be computed using Fourier computations as,

$$\hat{\underline{y}} = \sum_{s \in \Omega} \underline{f}_s \frac{(\nu' - \gamma \mu'_s)}{\nu'} \underline{f}_s^{*T} \underline{x}, \tag{8.7}$$

where

$$\mu'_s = (1 - 2\underline{\theta}'^T \underline{\Phi}_s)$$

are defined in (2.25) and (2.26). The parameters $\underline{\theta}'$, and ν' can be estimated from the noisy image. The estimates $\underline{\theta}'^*$ and ν'^* are obtained as,

$$\underline{\theta}'^* = [\sum_{s \in \Omega} \underline{q}(s)\underline{q}^T(s)]^{-1} \, (\sum_{s \in \Omega} \underline{q}(s)x(s))$$

and

$$\nu'^* = (1/M^2) \sum_{s \in \Omega} (x(s) - \underline{\theta}'^{*T} q(s))^2$$

where

$$\underline{q}(s) = \text{Col.}[x(s+r) + x(s-r), \; r \in N_s]$$

One can use the steady-state component of the spectral density of the degraded image as an estimate of γ [Andrews & Hunt, 1977]. The experimental results using this scheme are given below.

The original girl's image in Fig. 8.1 is corrupted by additive white noise of SNR = 7dB. GMRF models of different neighbor sets N_s were fitted to the noisy image and (8.8) was used. The noisy image together with the filtered images are given in Fig. 8.2 for SNR = 7 dB. The filtered images of the 7 dB noisy image corresponding to neighbor sets N_{s3} and N_{s4} are good. Table (8.1) gives the details of the GMRF models fitted to the noisy image. Using the model with the neighbor set N_{s4}, the best estimate of γ for 7 dB noisy image was obtained as 378.74 (true value being 393.01). The MSE corresponding to the best filter in Fig. 8.2 was 91.08 using the neighbor set N_{s4} with the error between the original and noisy being 378.53, roughly a reduction in the MSE by a factor of 4. The numerical values of the MSE between $\underline{y}$ and $\hat{\underline{y}}$ for the models in Table 8.1 as well as the restoration results for 0dB additive noise case may be found in [Chellappa & Kashyap, 1982a].

Fig. 8.1 Original uncorrupted image.

TABLE 8.1 DETAILS OF GMRF MODELS CORRESPONDING TO THE RESTORED IMAGES IN FIG. 8.2

Number	Neighbor set N_S
N_{S1}	{(0,1)(1,0)}
N_{S2}	{(0,1),(1,0),(1,1)}
N_{S3}	{(1,0),(1,−1),(0,1),(1,1)}
N_{S4}	{(1,0),(0,1),(1,1),(1,−1),(0,2),(2,0)}

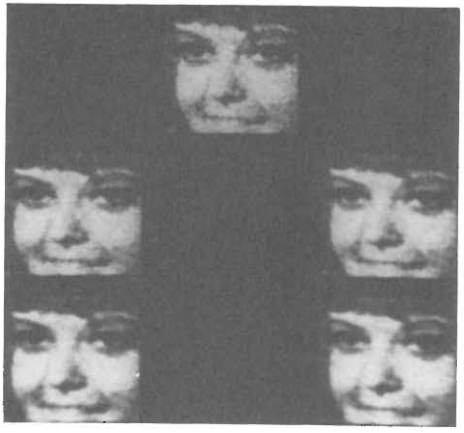


Fig. 8.2 Filtering of noisy images using GMRF models when the prototype of the original is not available. Top: noisy image of SNR = 5. Second row: filtered images using models N_{S1} and N_{S2}. Third row: filtered images using models N_{S3} and N_{S4}.

8.3 Errors in Variable Formulation of the Image Restoration Problem

Consider the image restoration problem

$$\underline{x} = \underline{y} + \underline{\eta}$$

where $\underline{x}$ and $\underline{y}$ are $M^2 \times 1$ vectors corresponding to the noisy and original noise-free images $\underline{\eta}$ is the zero mean additive signal independent white noise vector. Suppose that $\underline{y}$ is of zero mean and obeys an GMRF model of known structure and unknown parameters $\underline{\theta}$ and ν and that $\underline{\eta}$ is Gaussian of unknown variance γ. Since the MMSE filter involves, $\underline{\theta}$, η and γ and in practical applications one does not usually have $\underline{y}$ or a prototype but only $\underline{x}$, it is desirable to develop estimates of $\underline{\theta}$, ν, and γ from $\underline{x}$. For the sake of simplicity, let $\gamma = k\nu$, $k>0$. Then $\underline{x}$ has mean $\underline{\nu}$ and covariance matrix $\nu\underline{Q}^{-1}$ where

$$\underline{Q}^{-1} = K\underline{T} + B^{-1}(\underline{\theta})$$

where $B(\underline{\theta})$ is the transformation matrix from $\underline{e}$ to $\underline{y}$ in GMRF model. The log-likelihood function corresponding to a realization $\underline{x}$ is given by [Besag, 1977]

$$\log p(\underline{x}|\underline{Q},\nu) = -(1/2)\ M^2\log(2\pi\nu) + (1/2)\ \log\det\underline{Q} - \frac{1}{2\nu}(\underline{x}^T\underline{Q}\underline{x})$$

The estimate ν is

$$\hat{\nu} = \frac{1}{M^2}\ \underline{x}^T\underline{Q}\underline{x}$$

The determination of estimates for the remaining parameters consequently reduces to the problem of minimizing

$$-\frac{1}{M^2}\log\det\underline{Q} + \log\{\frac{1}{M^2}\ \underline{x}^T\underline{Q}\underline{x}\}$$

Suppose $\underline{x}$ is represented on a toroidal lattice then

$$\log\det\underline{Q} = -\sum_{s\in\Omega}\log h_s$$

where

$$h_s = k + (1-2\underline{\theta}^T\underline{\phi}_s)^{-1}$$

Also, using the block circulant property of $\underline{Q}$ it can be shown that

$$\underline{x}^T\underline{Q}\underline{x} = \sum_{s\in\Omega}|z_s|^2/h_s,\ \{z_s\} = FFT\{x(s)\}$$

which can be computed once for all. Once $\underline{\theta}$, ν, and γ are determined as discussed above, the MMSE filter can be implemented.

9. CONCLUSIONS

A systematic exposition of the usefulness of a class of 2-D noncausal models known as GMRF models for developing algorithms for image synthessis, clasification, and restoration has been given. Not covered in our paper are the representation of non Gaussian Markov random field (NGMRF) models, estimation of parameters and usefulness of NGMRF models in image processing. A scholarly paper on the representation and estimation by NGMRF models is that of Besag [Besag, 1974]. The NGMRF models have also been used for texture synthesis [Cross & Jain, 1983; Hassner & Sklansky, 1980].

10. ACKNOWLEDGEMENTS

The partial support of National Science Foundation under the grant ECS-82-04181 and USC Faculty Research Innovation Fund is gratefully acknowledged. The author is thankful to Professor R.L. Kashyap for introducing him to the area of stochastic modeling of images. The author is also thankful to Mr. Govind Sharma, Mr. Shankar Chatterjee, Mrs. Jerene Carey and Ms. Hilda Marti for their assistance in preparing this paper.

REFERENCES

[1] Abend, K., et al., Classification of Binary Random Patterns, IEEE Trans. on Inform. Theory, Vol. IT-11 (October 1965) pp. 538-544.

[2] Akaike, H., A New Look at the Statistical Model Identification, IEEE Trans. on Automat. Contr., Vol. AC-19 (December 1974) pp. 716-722.

[3] Anderson, T.W., Determination of the Order of Dependence in Normally Distributed Time Series, in Proc. of the Symposium on Time Series Analysis, M. Rosenblatt, (ed.) (Wiley, New York, 1962).

[4] Andrews, H.C. and Hunt, B.R., Digital Image Restoration. (Prentice-Hall, New Jersey, 1977).

[5] Bartlett, M.S., The Statistical Analysis of Spatial Pattern, (Chapman and Hall, London, 1975).

[6] Besag, J.E., Nearest Neighbor Systems and the Auto-Logistic Model for Binary Data, J. Royal Stat. Soc. Ser. B, Vol. 13-34 (1972) pp. 75-83.

[7] Besag, J.E., Spatial Interaction and Statistical Analysis of Lattice Systems, J. Royal Stat. Soc. Ser. B, Vol. B-35 (1974) pp. 192-236.

[8] Besag, J.E., Errors-in-Variable Estimation for Gaussian Lattice Scheme, J. Royal. Stat. Soc., Ser.B (1977) pp. 73-78.

[9] Bose, N.K., Applied Multidimensional Systems Theory, (Van Nostrand Reinhold, New York, 1982).

[10] Box, G.E.P. and Jenkins, G.M., Time Series Analysis-Forecasting and Control, (Holden-Day, San Francisco, California, 1976).

[11] Chellappa, R., Stochastic Models for Image Analysis and Processing, Ph.D Thesis, Purdue Unviersity, W. Lafayette, Indiana, (August 1981).

[12] Chellappa, R. and Kashyap, R.L., Digital Image Restoration Using Spatial Interaction Models, IEEE Trans. on Acoustics, Speech and Signal Processing, Vol.ASSP-30 (June 1982a) pp. 461-472.

[13] Chellappa, R. and Kashyap, R.L., Texture Synthesis Using Spatial Interaction Models, Proc. of IEEE Computer Society Conf. on Pattern Recognition and Image Processing, Las Vegas, (June 1982b) pp. 226-230.

[14] Chellappa, R. and Sharma, G., Two-Dimensional Spectral Estimation Using Spatial Autoregressive Models, Proc. Intl. Conf. on Acoustics, Speech and Signal Processing, Boston, (April 1983).

[15] Chellappa, R., Hu, Y.H., and Kung. S.Y., On Two Dimensional Markov Spectrum Estimation, IEEE Trans. on Acoust., Speech and Signal Proc., Vol. ASSP-31 (August 1983).

[16] Childers, D.G., Modern Spectral Estimation, (IEEE Press, New York, 1975).

[17] Cross, G.R. and Jain, A.K., Markov Random Field Texture Models, IEEE Trans. on Patt. Anal. and Mach. Intel., Vol. PAMI-5 (January 1983) pp. 25-40.

[18] Doob, J.L., Stochastic Process, (Wiley, New York, 1953).

[19] Duda, R.H. and Hart, P.E., Pattern Classification and Scene Analysis, (John Wiley, New York, 1973).

[20] Faugeras, O.D., et. al., Decorrelation Methods of Texture Feature Extraction, IEEE Trans. on Patt. Anal. and Mach Intel., Vol. PAMI-2 (July 1980) pp. 323-332.

[21] Goodman, D.M. and Ekstrom, M.P., Multidimensional Spectral Factorization and Univariate AR Models, IEEE Trans. on Automat. Control, Vol. AC-25 (April 1980) pp. 258-262.

[22] Hannan, E.J. and Quinn, B.G., The Determination of the Order of an Autoregression, J. of Royal Statistical Society, Ser. B, Vol. B-41 (1979) pp. 190-195.

[23] Haralick, R.M., Statistical and Structural Approaches to Texture, Proc. IEEE, Vol. 67 (May 1979) pp. 786-804.

[24] Haralick, R.M., Shanmugam, K. and Dinstein, I., Textural Features for Classification, IEEE Trans. on Syst., Man and Cybern., Vol. SMC-3 (November 1973) pp. 610-621.

[25] Hassner, M. and Sklansky, J., Markov Random Fields as Models of Digitized Image Texture, Computer Graph. and Image Proc., Vol. CG1P-12 (April 1980) pp. 376-406.

[26] Jain, A.K., A Fast Karhunen-Loeve Transform for a Class of Random Processes, IEEE Trans. on Commun., Com-24 (September 1976) pp. 1023-1029.

[27] Jain, A.K., A Fast Karhunen-Loeve Transform for Digital Restoration of Images Degraded by White and Colored Noise, IEEE Trans. on Computers, Vol. C-26 (June 1977) pp. 560-571.

[28] Jain, A.K. and Jain, J.R., Partial Difference Equations and Finite Differences in Image Processing, Part II: Image Restoration, IEEE Trans. on Automat. Control, Vol. AC-23 (October 1978) pp. 817-833.

[29] Jain, A.K., Advances in Mathematical Models for Image Processing, Proc. of IEEE, Vol. 69 (May, 1981) pp. 512-528.

[30] Kashyap, R.L., A Bayesian Comparison of Different Classes of Dynamic Models Using Empirical Data, IEEE Trans. on Automat. Contr., Vol. AC-22 (October 1977) pp. 715-727.

[31] Kashyap, R.L., Optimal Feature Selection and Decision Rules in Classification Problems with Time Series, IEEE Trans. on Inform. Theory, Vol. 17 (May 1978) pp. 281-288.

[32] Kashyap, R.L., Inconsistency of the AIC Rule for Estimating the Order of Autoregressive Models, IEEE Trans. on Automat. Contr., Vol. AC-25 (October 1980) pp. 996-998.

[33] Kashyap, R.L., Random Field Models on Finite Lattices for Finite Images, Proc. of the Conf. on Information Sciences and Systems, Johns Hopkins University, (March 1981a) pp. 215-220.

[34] Kashyap, R.L., Analysis and Synthesis of Image Patterns by Spatial Interaction Models, in Progress in Pattern Recognition, Vol. 1, L.N. Kanal and A. Rosenfeld, (eds.), (North-Holland Publishing Company, 1981b).

[35] Kashyap, R.L., Chellappa, R. and Khotanzad, A., Texture Classification, Using Features Derived from Random Field Models, Patt. Recn. Letters, Vol. 1 (October 1982) pp. 43-50.

[36] Kashyap, R.L. and Chellappa, R., Estimation and Choice of Neighbors in Spatial Interaction Models of Images, IEEE Trans. on Inform. Theory, Vol. IT-29 (January 1983) pp. 60-72.

[37] Kunsch, H., Thermodynamics and Statistical Analysis of Gaussian Random Fields, Z.W. Ver Gebiete (ed.), Vol. 58 (November 1981) pp 407-421.

[38] Lindley, D.V., The Use of Prior Probability Distributions in Statistical Inference and Decisions, in Proc. 4th Berkeley Symposium on Math. Statistics and Probability, Vol. 1 (1961) pp. 453-468.

[39] Max, J., Quantizing for Minimum Distortion, IEEE Trans. on Information Theory, Vol. IT-6 (March 1960) pp. 7-12.

[40] Moran, P.A.P., A Gaussian Markovian Process on a Square Lattice, J. Appl. Prob., Vol. 10 (1973) pp. 54-62.

[41] Moran, P.A.P., Necessary Conditions for Markovian Processes on a Lattice, J. Appl. Prob., Vol. 10 (1973) pp. 605-612.

[42] Moran, P.A.P. and Besag, J.E., On the Estimation and Testing of Spatial Interaction in Gaussian Lattice, Biometrika, Vol. 62 (1975) pp. 555-562.

[43] Murphy, M.S. and Silverman, L.M., Image Model Representation and Line by Line Recursive Restoration, IEEE Trans. on Automat. Control, Vol. AC-23 (1978) pp. 808-816.

[44] Onsager, L., Crystal Statistics, I: A Two-Dimensional Model with an Order-Disorder Transition, Phys. Ref., Vol. 65 (1944) pp. 117-149.

[45] Pratt. W.K. and Davarian, F., Fast Computational Techniques for Pseudo Inverse and Wiener Image Restoration, IEEE Trans. on Computers, Vol. C-26 (June 1977) pp. 571-580.

[46] Rosanov, Y.A., On Gaussian Fields with Given Conditional Distributions, Theory of Prob. and its Applications, Vol. II (1967) pp. 381-391.

[47] Schacter, B., Rosenfeld, A. and Davis, L.S., Random Mosaic Models for Textures, IEEE Trans. on Syst., Man and Cybern., Vol. SMC-8 (September 1978) pp. 694-702.

[48] Schwarz, G., Estimating the Dimension of a Model, Ann. Statist., Vol. 6 (March 1978) pp. 461-464.

[49] Sharma, G. and Chellappa, R., A Model Based Approach for Two-Dimensional Maximum Entropy Power Spectrum Estimation, Submitted for publication.

[50] Weszka, J., Dyer, C.R. and Rosenfeld, A., A Comparative Study of Texture Measures for Terrain Classification, IEEE Trans. on Syst., Man., and Cybern., Vol. SMC-6 (April 1976) pp. 269-285.

[51] Whittle, R., On Stationary Processes in the Plane, Biometrika, Vol. 41 (1954) pp. 434-449.

[52] Woods, J.W., Two-Dimensional Discrete Markovian Random Fields, IEEE Trans. On Information Theory, Vol. 18 (March 1972) pp. 232-240.

[53] Woods, J.W., Markov Image Modeling, IEEE Trans. on Automat. Contr. Vol. AC-23 (October 1978) pp. 846-850.

[54] Woods, J.W. and Radewan, C.H., Kalman Filtering in Two-Dimensions, IEEE Trans. on Information Theory, Vol. IT-23 (July 1977) pp. 473-482.

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

RECENT PROGRESS IN SHAPE DECOMPOSITION AND ANALYSIS

Linda G. Shapiro
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia 24061
U. S. A.

Two-dimensional shape analysis is important in the analysis of both two-dimensional and three-dimensional scenes. Shape is a critical factor in biomedical analysis and optical character recognition. It can provide important recognition cues when used as part of the knowledge in a system for understanding images of three-dimensional scenes. A number of techniques such as Fourier descriptors, moments, syntactic analysis, thinning, projection, and transformation to a graph structure have been proposed in the past. In this paper the recent progress that has been made in this area is surveyed.

INTRODUCTION

Two-dimensional shape analysis was one of the major topics of discussion at conferences and workshops in the 1970's. A number of papers appeared on the topic in the late seventies; these and previous work were surveyed by Pavlidis (1980). They included such techniques as Fourier descriptors, moments, syntactic analysis, thinning, decomposition, projection, transformation to a graph structure, and more.

Rather than reviewing any of these techniques, we will investigate what has gone on since. In particular, we will examine shape analysis research results that have been published since January 1980. We will attempt to present the flavor of such research by discussing a number of examples in some detail. We will not attempt to mention all the shape papers that have been printed; we leave this task to Azriel Rosenfeld's annual picture processing survey (Rosenfeld, 1980).

The recent papers on two-dimensional shape fall into two major categories: 1) shape decomposition and 2) shape description and matching. We divide the remainder of this paper similarly.

SHAPE DECOMPOSITION

Decomposition algorithms partition a shape into simpler parts. The need for such algorithms exists because global features are not suitable for describing complex shapes. The simple parts resulting from a decomposition algorithm are often used in structural matching algorithms. Two different classes of decompositions are still popular: area decompositions and boundary decompositions.

This research was supported by the National Science Foundation under grant MCS-8102874.

Area Decompositions

Several papers from the seventies and earlier appear to have influenced the area decomposition work reported in the eighties. Pavlidis (1968) started the ball rolling by decomposing shapes into convex pieces. Feng and Pavlidis (1975) extended this to convex pieces, spirals, and T-shaped pieces. Shapiro and Haralick (1979), citing the problem of noisy boundaries, invented a graph-theoretic algorithm to decompose a shape into near-convex pieces.

Bjorklund and Pavlidis (1981 employed a graph theoretic approach that seems to follow from both Feng and Pavlidis (1975) and Shapiro and Haralick (1979). The input to this algorithm is a straight-line polygonal approximation to the shape. The line segments of the polygonal approximation are considered to be atoms. A graph whose nodes are the atoms and whose edges correspond to various relationships between pairs of atoms is constructed. Only the K best relationships in which a vertex participates are stored, where K is a parameter of the graph creation process.

The relationships represented by the graph are divided into two general classes: global links, which are independent of geometrical structure, and local links, which are based on geometric proximity of the atoms. The links defined were

1) C link: a local link between consecutive line segments forming a convex corner,
2) N link: a local link between consecutive line segments forming a concave corner,
3) E link: an extension link between the current line segment and the next one in the same direction,
4) D link: a global closeness link between two sets of line segments which form concave vertices, and
5) S link: a global link between two approximately parallel segments of opposite directions at a specified distance or less apart.

Once the graph has been formed, it can be used to decompose the shape in several ways. For example, CV cycles are defined to be the cycles of the graph consisting of C, E, or D links. CV cycles identify convex subshapes. Furthermore, one can transform the graph by removing CV-cycles containing D links, one at a time, and then removing the remaining CV-cycles one at a time. After these transformations, the addition of S links may result in CES-cycles, which identify the strokes of the shape. Various other combinations of links could be used to extract other information about the shape. The idea may lead to further interesting results.

A second recent area decomposition algorithm was presented by Avis and Toussaint (1981). This is a fast ($\emptyset$(n log n) for n vertices) algorithm for decomposition of a polygon into disjoint star-shaped subsets (polygons entirely visible from at least one fixed point) as suggested by Maruyama (1972). The algorithm can be described by the following three steps.

1) Obtain a triangulation of the polygon. (A triangulation is a planar graph formed by adding as many non-intersecting interior edges between pairs of vertices of the polygon as possible. Triangulations are not unique for a given polygon.)
2) Color the vertices of the triangulation with three colors such that no two vertices with the same color are adjacent in the triangulation.
3) Select one of the colors and output each vertex of that color along with the list of all its adjacent vertices. Each such set of vertices will form a star-shaped polygon.

The resulting decompositions will be different for each of the three colors and none are guaranteed to be minimal in number of star-shaped polygons.

Another rather elegant approach is the work of Fairfield (1983). Shapes must be represented by a dot pattern, which can be the set of vertices of a defining polygon. The algorithm segments the shape into pieces described in (Fairfield, 1983) as "intuitively pleasing" as follows.

1) The Voronoi diagram of the dot pattern is generated. This can be done by an order n log n procedure for n dots.
2) At any point c on a directed path of the Voronoi diagram, the 'spread angle' of the path at c is the angle xcy where x is the dot closest to c on the left and y is the dot closest to c on the right. The concavity of a path can then be defined as the spread angle of the last point of the path minus the spread angle of the first point. A t-increasing path is a connected sequence of segments of the Voronoi diagram that contains a subpath of concavity less than or equal to - t for parameter t. The second step of the algorithm is to find all maximal t-increasing paths of the Voronoi diagram.
3) The last step is to cut each maximal t-increasing path into pieces of concavity less than t, yielding a shape decomposition.

Another area algorithm with pleasing results is that of Nevins (1982). This algorithm, which is related to that of Feng and Pavlidis (1975), inputs a polygon and outputs convex, spiral and biconcave regions. It uses knowledge of smoothness and symmetry criteria to reevaluate the existance and placement of slices after they are made. The technique can be described as follows.

Stage 1) The shape is segmented into convex regions, and the result represented by a tree having regions for nodes. An arc connects two nodes if the regions share a common boundary. A centrally located region is selected as the root of the tree. The segments introduced in this stage are called 'virtual segments'.

Stage 2) (and Stage 3) Beginning with the top node of the tree, each arc is examined and an attempt to either rotate it (replace it by another segment with common end point) or delete it is made. The tree is dynamically modified.

The criteria for rotation are complex and require some definitions.
An arc XY is 'stable' if

1) there exist arcs WX and YZ such that WX, XY, and YZ all emanate from the same node P of the tree,
2) WX and YZ each make an angle of greater than 65 degrees with XY when measured with respect to the interior of P, and
3) there are no arcs XV or YV emanating from a daughter node of P.

Stable segments are never dissolved and can only be rotated in stage 3).

An arc XY is 'semistable' if 1') and 2') or 3') hold.

1') If XY and YW emanate from the same node P and YZ emanates from a daughter of P, then either angle XYW (with respect to the interior of P) or angle XYZ (with respect to the interior of the daughter) exceeds 65 degrees.
2') If XY enamates from P, there exists a segment YZ which emanates from a daughter of P.
3') Dissolving XY would create an angle at X greater than 187 degrees and either the angle at Y greater than 187 degrees or an arc YZ emanating from the resulting node.

A semistable segment may be rotated, but not dissolved in stage 2 and rotated or

dissolved in stage 3.

The decision to rotate is based on trying to have the virtual segment under consideration mark a smooth transition along the boundary of one of the regions it separates. The other region should at least emanate from the virtual segment in a symmetrical manner.

A final area decomposition algorithm for digital shapes due to Arcelli and Sanniti di Baja (1982) deserves a brief mention. The algorithm determines the labeled medial line (ML) of the shape. The ML is a connected subset of pixels of unit width, similar to the medial axis concept. Each pixel of the ML is labeled with the radius of the digital circle of which it is the center. The algorithm actually produces a parametrized ML, excluding any pixel belonging to a protrusion of the shape having specified elongation. Thus a sequence of thresholds yields a sequence of different medial lines. Applying a reconstruction algorithm to each ML yields a sequence of shapes, each differing from the previous in the absence of some protrusions. Assigning labels to regions by which pairs of reconstructions differ creates a segmentation of the original shape.

Boundary Decomposition Algorithms

Boundary decomposition algorithms attempt to segment the boundary of a shape into meaningful segments to be used in further analysis. A paper by Guerra and Pieroni (1982) extends some of the ideas used in Shapiro and Haralick (1979) for area decomposition to a procedure for segmenting the boundary. As in Shapiro and Haralick (1979), the input is a sequence of boundary points representing a polygonal shape and the first step is to construct the L_I relation defined by $L_I = \{(p_1,p_2) \mid$ the line segment joining p_1 to p_2 lies interior to the shape$\}$. In Shapiro and Haralick (1979) the clusters of L_I resulting from application of a graph-theoretic clustering algorithm gave the area segmentation of the shape. In the Guerra and Pieroni paper, the boundary segmentation is achieved as follows.

1) Represent the L_I relation by the lower triangular portion of an adjacency matrix A. Extract from A, in a left top to right bottom scan, the triangular submatrices having main diagonal lying on the main diagonal of A and containing only ones. The points of each submatrix belong to a clique of the L_I relation. Call these submatrices the C_S cliques. Each C_S clique will consist of an ascending sequence of point indexes.
2) Combine C_S cliques as follows. Let $S_K = (P_h,\ldots,P_n)$ and $S_{K+1} = (P_n,P_{n+1},\ldots,P_m)$ be adjacent C_S cliques. When there is at least one interior line segment joining a point of S_K to a point of S_{K+1}, the two subsets are called correlated. Let T_K and T_{K+1} be the submatrices associated with S_K and S_{K+1} respectively. Define $T_K \; 0 \; T_{K+1}$ by

$$T_K \; 0 \; T_{K+1} = \{A(i,j) \mid i=n+1,\ldots,m; j=h,\ldots,n-1\},$$

and let $N(T_K,T_{K+1})$ be the number of ones in $T_K \; 0 \; T_{K+1}$. The 'correlation factor' $f_{T_K,T_{K+1}}$ between the two sequences of points is defined as

$$f_{T_K,T_{K+1}} = \frac{N(T_K,T_{K+1})}{(n-h)(m-n)} .$$

The closer f is to 1, the more acceptable it is to merge the two sets. The merging procedure is iterative, merging those pairs of adjacent subsequences of points satisfying a high threshold for f on the first iteration and gradually lowering the threshold.

Rutkowsky, Peleg, and Rosenfeld (1981) also decompose the boundary of a shape, but they use models and a relaxation procedure to aid in the segmentation. Their

objective is to segment a shape representing an object into nonoverlapping segments that can be assigned meaningful labels. Their procedure can be summarized as follows.

1) Form segments using segmentation points where the curvature is below a preset negative threshold.
2) Construct a digraph in which the nodes are segments and the arcs indicate the "follows" relation for adjacent segments. When there are several contours instead of one single contour, gaps are bridged by adding 'imaginary' segments and arcs to the graph. Parameters to the procedure control which non-touching segments are linked in this manner.
3) Each node in the graph is assigned a probability vector (N,R,T,L). The components N, R, T, and L are the probabilities that the segment represented by that node is a nose, right wing, tail, or left wing, respectively. Probabilities are assigned on the basis of a symmetry measure.
4) A second-level graph is obtained from the first by searching for paths that could possibly be one of the allowable objects.
5) A probabilistic relaxation procedure operates on the second-level graph to produce a final segmentation and labeling.

Comments on Decomposition Algorithms

Since the purpose of decomposition algorithms is to segment a shape into pieces that can be used for analysis or recognition, decomposition algorithms should yield a unique solution that is size invariant, translation invariant, and rotation invariant. The solution should, furthermore, be useful in some type of further analysis or matching algorithm. Of the area decomposition algorithms presented here, only the Bjorklund and Pavlidis algorithm seems to satisfy the uniqueness and usefulness criteria. It can identify convex subshapes and strokes, both of which can be useful in matching. The decomposition into star-shaped subsets by Avis and Toussaint is not unique; different colorings give different solutions. This would make it useless in matching. The last three area decomposition algorithms yield decompositions that do not satisfy any stated set of properties at all. It is not clear whether or not any of them could be useful.

Of the two boundary decomposition algorithms, the Guerra and Pieroni algorithm is completely data driven (bottom up), while the Rutkowsky, Peleg, and Rosenfeld algorithm is partly model driven (top down). The former has not been used for or suggested for any particular matching scheme. The latter is specifically used for airplane matching, but is not clearly easily extendable to other shapes.

SHAPE DESCRIPTION AND MATCHING

A sequence of boundary points is not a very useful representation for matching. Instead, the boundary points can be used to construct a description of the shape that is more suitable for matching. The decompositions of the previous section can be used as part of this description along with extracted features and relationships. In this section we discuss the construction of shape descriptions and the use of shape descriptions and shape models in matching.

Tanimoto (1981) suggests a descriptive mechanism called the polygon structure graph (PSG). A PSG is constructed from a polygon encoded as a string of tokens $z_1,\ldots,z_{2n}$ where the odd-numbered tokens represent side lengths and the even-numbered tokens represent vertex angles. Identical lengths (or identical angles) are represented by the same token. The PSG has the following properties.

1) There is a designated root node.
2) Each arc is labeled with a token.
3) Each node is labeled with a subset of the positions 1, 2, . . ., 2n.

4) If a path from the root to a node has the arc label sequence $a_1,\ldots,a_k$, then the edge-angle sequence corresponding to $a_1,\ldots,a_k$ occurs on the boundary of the shape with a_k occurring at each position in the subset of positions represented by that node.
5) Nodes with singleton labels have no arcs out; others have one or more arcs out.

The PSG can be considered to be an intermediate structure that allows quick access to all places in the polygon with a given substructure. Tanimoto gives an algorithm for computation of the PSG and suggests how to use it for determination of rotational symmetry and in matching.

Henderson and Davis (1981) use hierarchic syntactic descriptions to represent classes of shapes. Their formalism, called a stratified context free grammar, is a context free grammar with the addition of a level number for each vocabulary symbol and special syntactic/semantic production rules. The level numbers are set up so that the level number of each terminal symbol is 0 and if v := a is a production rule and level number(v) = k, then level number(s) = k-1 for each symbol s of the right-hand side a. Each vocabulary symbol v has the structure v = <name part>{<attachment part>}[<semantic part>] where <name part> is the name of the symbol, <attachment part> gives the attachment points for that symbol, and <semantic part> consists of predicates that must be satisfied concerning the attachment of that symbol.

Production rules are of the form (v:= a, A, C, G_a, G_s) where $a = v_1,v_2,\ldots,v_K$ is a string of vocabulary symbol, A is a set of applicability conditions on the syntax of the v_k's, C is a set of predicates on the semantic consistency of the v_k's, G_a is a set of attachment rules, and G_s is a set of semantic generating rules. The rules are rather complex and the construction of an entire grammar would be a lot of work for a human. It is not clear whether or not a computer could automatically generate such rules.

The Henderson/Davis system starts with an unknown shape and parses it using a stratified context free grammar and a hierarchical constraint reduction procedure. The constraints used in the relaxation are automatically derived from the grammar. Syntactic constraints allow for deletion of a hypothesis for a vocabulary symbol when, at some attachment point of the symbol, no hypothesis for any of the possible neighboring symbols (at any level of the hierarchy) exists. Semantic constraints allow for deletion of a hypothesis when predicates in the grammar rules are not satisfied.

The overall procedure can be summarized as follows.

1) Starting with a piecewise linear approximation to the shape, generate primitive segments. Use several different segmentations in order to increase the chance of finding all primitives.
2) Compute the relationships between segments.
3) Hypothesize labels (terminals) for each primitive, using features to reduce the number of labels per primitive. Construct the bottom level of a network with these terminal symbols.
4) Iteratively apply the hierarchical constraint procedure while building each level of the network from the next lower level.

The result is a parse tree representing a syntactically and semantically consistent parse of the shape.

Bhanu (1981) also matches input shapes to models, but he allows parts of the input to be occluded. His models are actual objects, represented by polygons. A model X_m is represented by a sequence of segments $X_m = (T_1,\ldots,T_N)$. The input object or objects have segments $O_1,\ldots,O_{L-1}$, each segment considered a class. An extra segment O_L represents the nil class. For a given model X_m, $P_{im}(k)$ is the probability that segment T_i of model X_m belongs to class O_k. The vector $P_{im} = [P_{im}(1),\ldots,P_{im}(L)]^T$ gives the probability that segment T_i of model X_m

belongs to the classes $O_1,\ldots,O_L$. The set of all such vectors is a stochastic labeling of the model segments. Once this set of vectors has been set up, a set of criteria for consistency and ambiguity are used to define a nonlinear optimization problem whose solution is an assignment of each object segment to a model segment. A gradient projection method with a penalty function is used to obtain the solution.

Smith and Jain (1981) propose a shape matching scheme that uses chord distribution to represent the shapes. Their procedures consists of the following steps.

1) Input two shapes to be compared.
2) Scale the shapes so that both perimeters have unit length.
3) Form the empirical distributions of the chord lengths of the shapes.
4) Perform a two-sample Kolmogrov-Smirnov test to determine if the distributions are the same. If so, the shapes match.

The method has been tested on circles, noisy circles, equilateral triangles, octagons, and country outlines obtained from maps.

Kashyap and Oommen (1982) perform shape matching using an area-based measure. The two input polygons U and V are first normalized to have either the same area or the same perimeter. Then one of several geometrical dissimilarity measures can be applied. We summarize these measures below.

1) Edge-based Measure Superimpose U on V with the i'th edge of U on the j'th edge of V, with midpoints coinciding. The dissimilarity of this configuration is defined to be the sum of the areas of U and V minus twice the area they have in common. The dissimilarity measure E(U,V) is defined to be the minimum dissimilarity over all possible values of i and j.

2) Vertex-based Measure This is the same as the edge-based measure except that the polygons are initially superimposed so that the i'th vertex of U coincides with the j'th vertex of V.

3) Integral Error Dissimilarity Measure Again U is superimposed on V so that the i'th edge of U lies on the j'th edge of V, with midpoints coinciding. Consider this midpoint to be the origin of a rectangular coordinate system with abscissa along the common edge. In a clockwise traversal of each polygon from this origin, let $H_i(t)$ be the unique point reached on U after traversing length t and let $G_j(t)$ be the unique point reached on V, $\emptyset \leq t \leq 1$. Then $||H_i(t) - G_j(T)||^2$ can be used as an error criterion. The dissimilarity $D_{ij}(U,V)$ is defined by

$$D_{ij}(U,V) = \int_{\emptyset}^{1} ||H_i(t) - G_j(t)||^2 dt.$$

Then the dissimilarity measure D(U,V) is the minimum $D_{ij}(U,V)$ over all possible values of i and j. A similar vertex-based measure can be defined.

Computationally, $\int_{\emptyset}^{1} ||H_i(t) - G_j(t)||^2 dt$ is approximately by $\sum_{k=\emptyset}^{K-1} ||H_i(t) - G_j(t)||^2 d$ where K is determined by the step size d of t, and only a subset of the M*N possible values of i and j are used to estimate the minimum. A 97% classification accuracy was obtained.

Burr (1981) performs an iterative elastic matching of line drawings. Each line drawing is represented as a vector of triples of the form $\{(x_i,y_i,s_i)|i=1,\ldots,N\}$ where $s_i = 1$ if point i connects to point i+1 and $\emptyset$ otherwise. Given two such line drawings L1(i) and L2(j), the idea is to overlay one on the other and, intuitively, stretch one to exactly fit the other. The stretching required to

force one shape to correspond to another can be used as a measure of how well the two shapes match.

The final paper on description and matching is our own work on a structural model of shape (Shapiro, 1980). In this work, we define formal relational models for shapes and show how to extract a relational description from the input shape and match it to the relational models. We briefly summarize the idea here.

Let X be a complex shape. X can be decomposed into a set of simple parts which are the near convex pieces of the shape (Shapiro and Haralick, 1979). The decomposition is the result of applying a graph-theoretic clustering algorithm to the interior line segment relation L_I defined by $(p_1,p_2) \in L_I$ if p_1 and p_2 are boundary points of the shape and the line segment joining them lies totally interior to the boundary of the shape. The same graph-theoretic clustering algorithm can be applied to a second relation, the exterior line segment relation L_E defined by $(p_1,p_2) \in L_E$ if p_1 and p_2 are boundary points of the shape and the line segment joining them lies totally exterior to the boundary of the shape. The results of clustering L_E are the intrusions into the boundary of the shape. We call the simple parts and intrusions shape primitives. Both the simple parts and intrusions are area-based primitives.

Our structural model of a shape is a description of its simple parts, its intrusions, and their interrelationships. Let $S = \{s_1,\ldots,s_n\}$ be the simple parts of a shape, and let $I = \{i_1,\ldots,i_m\}$ be the set of intrusions. We treat each simple part and each intrusion as a two-dimensional region having area and a closed boundary. Let $P = S \cup I$ be the set of shape primitives. Let $d: P \times P \rightarrow [0,\infty]$ be a distance function which gives a relative measure of the distance between two primitives. Let D be a non-negative real number. Define the Boolean function $p:P^3 \rightarrow \{\text{true,false}\}$ by $p(p_1,p_2,p_3)$ = true if $d(p_1,p_2) \leq D$, $d(p_2,p_3) \leq D$, and it is possible to draw a straight line from a boundary point of p_1 through p_2 to a boundary point of p_3 in such a way that there is significant area on either side of the line and false otherwise. Thus $p(p_1,p_2,p_3)$ is true if both p_1 and p_3 touch or almost touch p_2, and a part of p_2 lies between p_1 and p_3.

A shape description D is a triple $D = (A, R_p, R_i)$ where A is an attribute-value table or property list containing global properties of the shape, $R_p = \{(i_1,s,i_2) \in I \times S \times I \mid p(i_1,s,i_2) = \text{true}\}$, and $R_i = \{(s_1,i,s_2) \in S \times I \times S \mid d(s_1,s_2) \leq D$ and $p(s_1,i,s_2) = \text{true}$ and $d(s_1,s_2) \leq D\}$. Thus R_p consists of triples of the form (intrusion 1, simple part, intrusion 2) where the simple part protrudes between the two intrusions, and R_i consists of triples of the form (simple part 1, intrusion, simple part 2) where the two simple parts touch or nearly touch and form part of the boundary of the intrusion.

We say that two shapes match if there is a function that maps the primitives of the first shape to the primitives of the second in such a way that primitives map to like primitives and the R_i and R_p relations are preserved. Such a structure preserving function is called a relational homomorphism.

We have developed a procedure for determining if there is a homomorphism from a model or prototype shape to a candidate or unidentified shape. There were several important considerations in the design of the shape matching procedure.

1) Some of the relationships in the prototype shape may be more important than others. Thus it is desirable to assign weights to triples in the description of the prototype.
2) It is possible for two similar shapes to decompose into different numbers of primitives. Thus, the mapping should be able to assign the same primitive in the candidate to several primitives in the prototype if they are physically close enough.
3) A homomorphism corresponds to a match, but does not necessarily indicate a good match. A measure for the goodness of a match must be developed.

These considerations and others were embodied in a SNOBOL4 program that performs a treesearch (with relaxation for complex shapes) to determine if two shapes match. Experiments with character data showed this to be a viable approach to characterizing and matching shapes.

DISCUSSION

Shapes and/or shape models have been described by graphs, hierarchic grammars with syntactic and semantic components, chord distributions, and ternary relations. Matching has been achieved by a constraint reduction procedure (hierarchic relaxation), the solution of an optimization problem, statistical tests, area comparisons, measurement of the stretching required to force one shape to correspond to another, and determination of relational homomorphisms. It is impossible to say, at this time, which is the best method. The Henderson-Davis method seems very natural and very robust, but the stratified shape grammars are very difficult to construct and it is not clear that classes of shapes other than airplanes are easily describable. The Shapiro method (1980) is also rather natural and the shape and model descriptions can be computed automatically. Both the Henderson-Davis algorithm and the Shapiro algorithm are time consuming. The other algorithms are more restrictive in what pairs of shape will match; they are also more feature-oriented than structure-oriented. These may be faster and can be useful in controlled situations such as part inspection.

Only time will indicate which algorithms are most useful since these will show up in the production systems of the near future.

References

Arcelli, C. and G. Sanniti di Baja, "Shape Splitting Using Maximal Neighborhoods", Proceedings of the 6th International Conference on Pattern Recognition, 1982, pp. 1106-1108.

Avis, D. and G. T. Toussaint, "An Efficient Algorithm for Decomposing a Polygon into Star-Shaped Polygons", Pattern Recognition, Volume 13, Number 6, 1981, pp. 395-398.

Bhanu, Bir, "Shape Matching of Two-Dimensional Occluded Objects", Proceedings of the 6th International Conference on Pattern Recognition, 1982, pp. 742-744.

Bjorklund, C. and T. Pavlidis, "Global Shape Analysis by k-Syntactic Similarity", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-3, Number 2, March 1981, pp. 144-154.

Burr, D.J., "Elastic Matching of Line Drawings", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-3, Number 6, November 1981, pp. 708-713.

Fairfield, J., "Segmenting Dot Patterns by Voronoi Diagram Concavity", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-5, Number 1, January 1983, pp. 104-110.

Feng, H.F. and T. Pavlidis, "Decomposition of Polygons into Simpler Components: Feature Generation for Syntactic Pattern Recognition", IEEE Transactions on Computers, Volume C-24, June 1975, pp. 636-650.

Guerra, C. and G.G. Pieroni, "A Graph-Theoretic Method for Decomposing Two-Dimensional Polygonal Shapes into Meaningful Parts", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-4, Number 4, July 1982, pp. 405-407.

Henderson, T. C. and L. S. Davis, "Hierarchical Models and Analysis of Shape", Pattern Recognition, Volume 14, Numbers 1-6, 1981, pp. 197-204.

Kashyap, R.L. and B.J. Commen, "A Geometrical Approach to Polygonal Dissimilarity and Shape Matching", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-4, Number 6, November 1982, pp. 649-654.

Maruyama, K., A Study of Visual Shape Perception, Department of Computer Science, University of Illinois, Urbana, October 1972.

Nevins, A.J., "Region Extraction From Complex Shapes", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-4, Number 5, September 1982, pp. 500-510.

Pavlidis, T. "Algorithms for Shape Analysis of Contours and Waveforms", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-2, Number 4, July 1980, pp. 301-312.

Pavlidis, T., "Analysis of Set Patterns", Pattern Recognition, Volume 1, 1968, pp. 165-178.

Rosenfeld, A., "Picture Processing: 1981", Computer Graphics and Image Processing, Volume 19, Number 1, May 1982, pp. 35-75.

Rutkowski, W., S. Peleg, and A. Rosenfeld, "Shape Segmentation Using Relaxation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-3, Number 4, July 1981, pp. 368-375.

Shapiro, L.G., "A Structural Model of Shape", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-2, Number 2, March 1980, pp. 111-126.

Shapiro, L.G. and R.M. Haralick, "Decomposition of Two-Dimensional Shapes by Graph Theoretic-Clustering", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume PAMI-1, Number 1, January 1979, pp. 10-19.

Smith, S. P. and A. K. Jain, "Chord Distributions for Shape Matching", Proceedings of the IEEE Conference on Pattern Recognition and Image Processing, August 1981, pp. 168-170.

Tanimoto, S. L., "A Method for Detecting Structure in Polygons", Pattern Recognition, Volume 13, Number 6, 1981, pp. 389-394.

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

DYNAMIC SCENE ANALYSIS

Ramesh Jain
Department of Electrical and Computer Engineering
The University of Michigan
Ann Arbor MI 48109

Abstract

This paper presents an overview of dynamic scene analysis. It discusses techniques for change detection, segmentation, computing optical flow and extracting information therefrom, recovering 3-Dimensional information about the objects and the motion, representation of motion in terms of low level concepts and the verbalization of motion. The emphasis is on an overview of the state of the art, rather than an exhaustive survey.

1. Introduction

The World is dynamic. Most biological vision systems have evolved to cope with the changing world. The past decade has seen the emergence of computer vision systems [BaB82]. For a computer vision system engaged in the performance of non-trivial real world operations and tasks, the ability to cope with moving and changing objects and viewpoints is vital. Though early computer vision systems were concerned with static scenes, the last few years have seen an ever-increasing interest in computer vision systems capable of analyzing dynamic scenes.

The input to a dynamic scene analysis system is a sequence of frames of a changing world. The camera may also be moving. Each frame represents image of the scene at a particular time instant. The changes in a scene may be due to the motion of the camera or the motion of the objects, illumination changes, or the changes in the structure, size, or shape of an object. Usually it is assumed that the changes in a scene are due to motion of camera and/or objects and that the objects are either rigid or quasi-rigid; other changes are not allowed. The task of the system is to detect changes, to find motion characteristics of the observer and the objects, to recover the structure of the objects, to characterize the motion using high level abstraction, and to recognize moving objects. It is also possible that future systems will be required to observe a scene, then describe the events taking place in a language understandable by a possibly naive user of the system, who may not know anything about computers.

A scene usually contains several objects. An image of the scene at a given time represents a projection of a part of the scene, which depends on the position of the camera. It is necessary to consider separately the four possible relations between camera and scene:

1. Stationary Camera, Stationary Objects (SCSO).
2. Stationary Camera, Moving Objects (SCMO),
3. Moving Camera, Stationary Objects (MCSO), and
4. Moving Camera, Moving Objects (MCMO).

This work was supported by the National Science Foundation, under the Grant No. NSF-G-MCS-8219739.

The SCSO, static scene analysis, is of only peripheral interest in this paper, though it is clear that many techniques developed for static scene analysis will play an important role in the dynamic scene analysis. In many applications it may be necessary to process a single image to obtain required information and it may be possible to extract such information from the image. It appears, however, that there are many applications which require the information extracted from a dynamic environment. Some applications require understanding of a dynamic process, such as cell motion, using vision techniques. Clearly, availability of a frame sequence offers more information for the understanding of a scene, but at the cost of a significant increase in the amount of the data to be processed by the system. Applying static scene analysis techniques to each frame of the sequence for the analysis of dynamic events requires an enormous amount of computation and suffers from all the problems of the static scene analysis. Fortunately, research in the field of dynamic scene analysis has shown that the extraction of information in dynamic scenes may be easier than in static scenes. In most cases the increase in the amount of the data is not a problem. On the contrary, the total computational effort may be significantly less and the performance better for some tasks (eg., the segmentation of a scene).

In dynamic scene analysis research [MaA78, Nag78b, Nag82c], SCMO scenes have received maximum attention. In such scenes, usually it is desired to detect the motion, extract masks of the moving objects with the aim to recognize them, and compute their motion characteristics. The MCSO scenes received attention recently, particularly because of optical flow research, but the case of moving objects and moving camera, the MCMO, has received very little attention [MaA78]. Clearly, the MCMO is the most general, and possibly, the most difficult situation in dynamic scenes. SCMO and MCSO find application in many situations and have been studied by researchers in different contexts under different assumptions. Many techniques developed for stationary camera are not applicable if the camera is allowed to move. Similarly techniques developed for moving camera have assumed stationary scenes and are not applicable if the objects were allowed to move. Only recently [Jai83b, Jai83c, Jai83d] are techniques being developed that will be applicable to all dynamic scenes.

Some researchers [MaA78,Jai81a] view the process of analyzing a dynamic scene in three phases: ***peripheral***, ***attentive***, ***and cognitive***. The peripheral phase is concerned with extraction of information that is imprecise but is very helpful for later phases of analysis. The information extracted in the peripheral phase gives an indication of the activities in the scene and helps in deciding which parts of the scene need careful analysis. The attentive phase concentrates its analysis in the active part of the scene and extracts information that may be used for the recognition of objects, analysis of motion of objects, preparing a history of events taking place in the scene and other related phenomena. The cognitive phase applies the knowledge about the objects, motion verbs, and other application dependent concepts to analyze the scene in terms of objects and events taking place in a scene. A conceptual system based on these three phases is described by Jain and Haynes in [JaH82]; Jerian and Jain discuss some cognitive level issues related to the recovery of 3-Dimensional information in the design of such a system in [JeJ83b].

In this paper we present an overview of the research in dynamic scene analysis. Our aim is to discuss various aspects of the analysis of dynamic scenes, with an emphasis on the analysis of motion, and to outline briefly some approaches proposed by researchers. We will not try to give an exhaustive survey of the field. The field of dynamic scene analysis is very active; many different approaches for various aspects of the problem are being presented. For a good idea of the changes in the field see [MaA78, Nag78b, Nag82c]. The organization of this paper is influenced by the currently active areas of research, which are by no means disjoint. Many related areas, such as tracking [Sch79, Sch82, ScM81, LeY82], motion compensated coding [NeR79,NeR80, SNR80, NeS79], architecture for the dynamic scene analysis [AgJ82, GGF80], and applications are not discussed in this paper.

The section 2 discusses methods for change detection. Methods for the segmentation of dynamic scenes to extract images of moving objects are discussed in section 3. The segmentation of a dynamic scene may be only on the basis of motion to extract images of moving objects or may involve segmentation of the static components of the scene also. Optical flow has attracted researchers in psychology since Gibson [Gib79, Lee80] suggested that pilots and birds use it. The presence of information in the optical flow, methods for determining optical flow, and techniques to extract information from optical flow are the subject of section 4. Motion has been considered a potent clue for determining depth and structure of objects. Various methods for the extraction of structure of objects and for the determination of motion parameters are discussed in section 5. If the aim is to describe the events in a scene then the representation of complex motions in simpler forms and the abstraction of the temporal motion characteristics using motion verbs will be required. The high level processes used for such a description are the subject of section 6. Finally, we review the relationship between various aspects and methods inorder to integrate the outputs of different techniques and to suggest future directions for research in section 7.

1.1. Terminology

The input frame sequence is represented as $F(x,y,t)$ where x and y are the spatial specification of an element in a frame representing the scene at time t. The value of the function represents the intensity of the pixel and is generally quantized into 256 levels. It is assumed that the image is obtained using a camera located at the origin of the 3-dimensional coordinate system. This means that the coordinate system is observer centered. The projection used may be perspective or orthogonal. Orthogonal projection approximates the camera when the objects are located far away from the camera as compared to the focal length of the camera. The 3-D coordinate of a point will be denoted by (X,Y,Z) and the projection of the point will be denoted by (x_p,y_p). The line of sight, or the optical axis of the camera, is assumed to be along the Z axis. An example of an imaging geometry is shown in Figure 1.

Figure 1 The imaging geometry employs the left hand coordinate system and planar perspective projections.

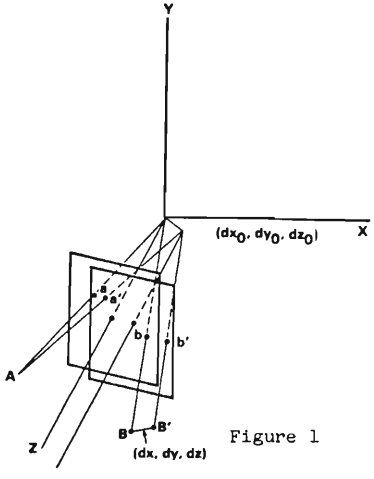


Figure 1

Since the frames are usually taken at regular interval, we will assume that t represents the t^{th} frame of the sequence, rather than the frame taken at absolute time t.

2. Change Detection

The result of a perceptible motion is some change in the frames of the sequence. By detecting such changes one may analyze motion characteristics. If the motion is restricted to a plane that is parallel to the image plane, then a good quantitative estimates about the motion components may be obtained; in case of 3-D motion only qualitative estimates are possible. Historically, early methods for dynamic scene analysis were based on change detection in a frame sequence. By analyzing frame-to-frame changes the global analysis of the sequence was performed. The changes were detected at different levels: pixel, edge, and region. The changes detected at pixel level can be aggregated to obtain useful information for constraining the computational requirements of the later phases [Jai81a]. In this section we discuss various methods developed for change detection and their application in dynamic scene analysis.

2.1. Pixel Level

The most obvious method of change detection in two frames is to compare the corresponding pixels of the frames to determine whether they are same or different. The subtractive TV used in [OH073] is one of the application using this method. In the simplest form a binary difference picture $DP_{jk}(x,y)$ for two frames $F(x,y,j)$ and $F(x,y,k)$ is defined as:

$$DP_{jk}(x,y)=1 \quad \text{if} \quad \left|F(x,y,j)-F(x,y,k)\right| > \tau$$

$$=0 \qquad otherwise$$

In the difference picture all pixels with value 1 are considered the result of the motion of objects. Clearly, this method assumes that the frames are properly registered and the illumination in the image remains constant. In real scenes such a simple test for change detection, usually, results in unsatisfactory results due to noise. A simple size filter may be used [JMN77] for ignoring noisy pixels which do not form a connected cluster by using a simple criterion that only those difference picture that belong to a 4-connected (or 8-connected) component of size above a threshold size will be attributed to motion. The motivation behind this filter is the fact that usually noise entries are isolated and the changes due to motion of surfaces form connected clusters in a difference picture. The filter is very effective in reducing noise but, unfortunately, it also filters some desirable signals, eg. those from slow or small moving objects.

Nagel [Nag78a] modified Yakimovsky's method of region growing to compare corresponding areas of two frames. By considering corresponding areas of two frames using the likelihood ratio

$$\lambda=\frac{\left[\frac{\sigma_1+\sigma_2}{2}+\left(\frac{\mu_1-\mu_2}{2}\right)^2\right]^2}{\sigma_1 * \sigma_2}$$

(where μ and σ denote the mean grayvalue and the variance for the sample areas from the frames.)

one may obtain the areas where changes are taking place, again by using a threshold. A minor problem in the application of the likelihood ratio is that it can be applied to the areas, not to single pixels. This problem may be solved by considering the corresponding areas of frames. It was suggested that the corresponding areas of the frames may be the

super-pixels formed by combining nonoverlapping rectangular areas of the frames comprising m rows and n columns. The values of m and n are selected to compensate for the aspect ratio of the camera. The likelihood ratio test combined with the size filter discussed above works quite well [JMN77, JaN79] for removal of noise. The problem of small and slow moving objects is exacerbated, however, since super-pixels effectively raise the threshold for the detection of the motion of such objects. Jain, Militzer, and Nagel [JMN77] introduced the concept of accumulative difference picture for removing this limitation. In place of comparing two frames and forming a difference picture, they compared every frame of the sequence to a reference frame and increased the entry in the accumulative difference picture by 1 whenever the likelihood ratio for the area exceeded the threshold. Thus

$$ADP_n(x,y) = ADP_{n-1}(x,y) + 1 \quad \text{if } \lambda > \tau$$

$$= ADP_{n-1}(x,y) \quad otherwise.$$

where τ is a preset threshold.

The first frame of the sequence was usually considered the reference frame and the accumulative difference picture ADP_0 was initialized to 0. An ADP allows for the detection of small and slow moving objects.

The likelihood test discussed above was based on the assumption of the uniform 2^{nd} order statistics of a region. Nagel has recently suggested the use of likelihood tests based on the approximation of intensity values of pixels belonging to the super-pixels using surfaces such as facets and quadratic surfaces [HNR82, Nag82a, NaR82]. These higher order approximation allow for the better characterization of intensity values and result in more robust change detection. He showed that performance of these tests is far superior to the test based on the linear intensity approximation of intensities.

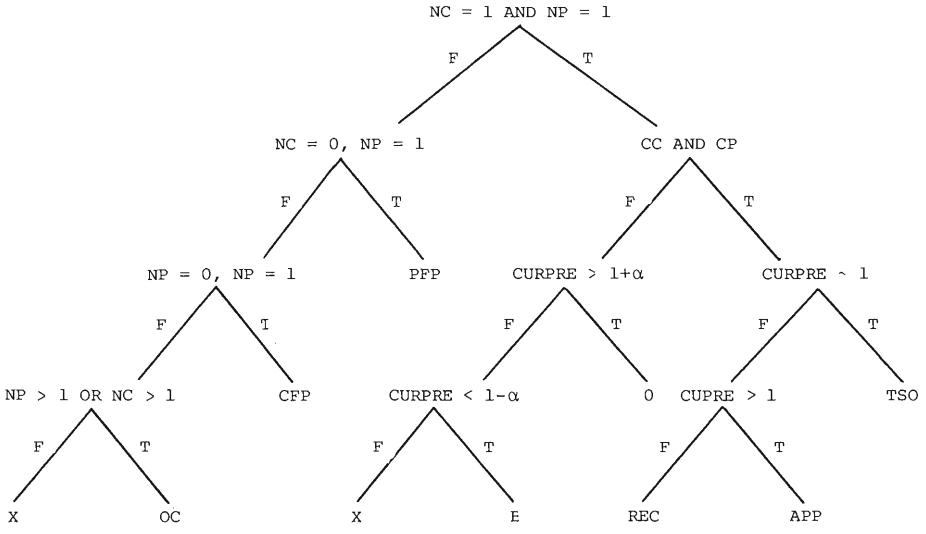
Note that the tests based on likelihood ratio result in the dissimilarity detection at super-pixel level. Moreover, since the tests are based on likelihood ratio, we can only decide whether the areas under consideration have similar greylevel characteristics or not; information about the relative intensities is not retained. Yachida et al [YAT78] propose the use of positive and negative difference pictures by using the sign of the difference. Jain [Jai83b, Jai83d] has shown that the use of the positive, negative, and absolute difference and accumulative difference pictures allows for a simple segmentation and motion characteristics extraction. These will be discussed in the section on the segmentation. It has also been proposed that more information about the objects and their motion can be obtained by combining consecutive difference pictures [Jai83d, LeG82].

Many systems have used the concept of difference picture in practical applications [LeG82, ReJ83]. Clearly, the most attractive aspect of difference picture is their simplicity. In the simplest form the use of the difference picture appears to be noise-prone in the analysis. The changes in the illumination and registration, in addition the electronic noise of the camera, may result in many false alarms in non-trivial real world scenes. By using a likelihood ratio and the size-filter most of the camera noise may be eliminated. The changes in illumination will create problems for any intensity based approaches and can only be handled at a symbolic level. The misregistration of the frames results in assigning false motion components. If misregistration is not severe, accumulative difference pictures can eliminate it.

It should be emphasized here that by using any dissimilarity measure at pixel level, we are detecting only intensity changes. In a dynamic scene analysis this is the lowest level of analysis. After such changes have been detected, some other processes are required to interpret these changes. Experience has shown that the most efficient use of difference picture is in peripheral processes for directing the attention of interpretation processes to the areas of the scene where some *activity* is taking place. Jain [Jai81a] developed a decision tree, shown in Figure 2, using simple features of difference pictures that enables extraction of approximate information about activities taking place in a dynamic scene. Several systems have been developed that start with the difference pictures in motion understanding[JaN79, JaJ83, TSR82].

Figure 2

A decision tree to extract motion information in peripheral phase from difference pictures. Using some features of the difference pictures, approximate information about events in the scene may be extracted.



Recently, it has been demonstrated that difference and accumulative difference pictures may be used for moving observer also [Jai83b]. Changes detected in frames at the pixel level may be used to extract useful information about the motion of objects and for the segmentation, as discussed in the following section, of a dynamic scene where camera and object motions are not restricted.

2.2. Static Segmentation and then Matching

Segmentation is the task of identifying semantically meaningful components of an image and grouping the pixels belonging to such a component. Segmentation of an image has attracted many researchers [BaB81]. It is not necessary that segmentation be performed in terms of objects; some predicate based on intensity characteristics may also be used. Usually the predicates based on intensity characteristics are called features. If an object or a feature appears in two or more images, then it may be required to segment the images to identify it in images. The process of identifying an object, or a feature, in two or more frames is called the correspondence process. In the following discussion, segmentation refers to the identification of objects and partial segmentation to feature extraction.

It is possible to segment, or at least partially segment, each frame of a sequence using static scene analysis techniques and then use matching to solve correspondence and detect changes in the location of the corresponding segments for the detection of motion. Crosscorrelation and features in the Fourier domain are used for the detection of cloud motion in [ALR75]. Aggarwal and Duda [AgD75] detect edges in images and then used the notion of false and real vertices for matching in simulated cloud images. Several systems have been developed that consider each frame of the sequence and segment, at least partially, each frame to find regions, corners, edges or some other features in each frame [BaT79, ChJ75, ChW77, WaC80, Law81, Law82, MaA79, Pot74, Pra79, RoA79, Wil80]. The next step is to try to match these features in consecutive frames to detect whether these features have been displaced. Some economy in matching can be achieved by using a prediction based on the displacements in the previous frames.

Price and Reddy [PrR77] show that using region features, such as the size, location, elongatedness, color, area of the bounding rectangle, orientation, images of a city scene taken from two different viewpoints can be matched succesfully. Roach and Aggarwal [RoA79] use multilevel matching for tracking objects in a sequence. Their objects are polyhedral and can move in 3-D space resulting in occlusion. They allow multiple interpretations and carry uncertainty until it can be resolved. The features used in matching vary from velocity values to local pictorial features. Initially, two frames of the sequence are completely segmented and only pictorial features are used for matching. In subsequent frames the results of matching from earlier frames can be utilized. Knowledge about the block-worlds is extensively used by the system. In [AYT81] the matching of vertices for the analysis of the motion is performed using the junction properties of the block-world. The extension of such a matching approach to real world scenes is a non-trivial task.

Barnard and Thompson [BaT79] propose a method for computation of disparities in images. Their approach is based on the theory that the discretness, similarity, and consistency properties can interact to enhance the performance of a matching process. They first detect discrete points in images by applying an interest operator to each image. An interest operator locates points in an image that are significantly different from the points in its neighborhood. Most interest operators exploit the variance in the intensity at the point. After finding interesting points in both images a set of possible matches for each point of image 1 is created by considering that a point may match any point from the other image within a window of limited size. A relaxation approach is used to refine the set of matches for each point. It is shown that using this approach good results can be obtained for stereo images or for two images of a sequence.

Grosky and Jain [GrJ83] approximate a region corresponding to an object by an elliptical paraboloid using least squares fit to the intensity values. The parameters of the parabolaoid are used first to establish correspondence and then to obtain the translation and the rotational component of the surface. It is shown that the parameters of the paraboloid may be computed recursively in a pyramidal data structure while finding connected components using linking in pyramids. This fact may allow implementation of this matching on special architecture.

The major problem in these approaches is the segmentation step. If one statically segments each frame completely, the computational cost becomes prohibitive. Moreover, segmentation is a complex operation. If features are used then the problem is what features should be used. Simple methods, such as Moravec's interest operator [Mor81], usually give too many features, making matching very expensive and unreliable; complex features, such as corners are computationally expensive and their precise location is difficult to obtain.

Ullman [Ull79] argues that human visual system performs a matching at intermediate level, such as features. He posits that intensity based methods may be used only by the peripheral processes.

The segmentation of a static scene from its image has been a difficult problem. Most efforts to segment a real world complex scenes indicate that even after Herculean efforts such system can give *good* segmentation only in Utopian world. It is widely believed that motion can be used for segmentation, in contrast to the previously described methods which segment each frame and then use matching for motion detection. It is interesting to note that most approaches that suggest that each frame be segmented individually and then correspondence be established, usually demonstrate their approach considering synthetic scenes or a very few frames of a sequence.

2.3. Time Varying Edge Detection

Considering the importance of edge detection in static scenes, it is reasonable to expect that techniques for time-varying edge detection may play a very important role in dynamic scenes. In segment-and-match approaches efforts are wasted by applying detection and matching operations to features many of which may be static and hence obstacles for the extraction of motion information. By detecting only moving features one may enhance matching substantially. In fact Dreschler and Nagel[DrN81] first extract moving objects and then detect feature points for matching in their approach for the recovery of the 3-D structure. If in each frame of a sequence only those edges that are moving could be detected, the subsequent processes may use only these edges.

It appears obvious to apply 3-D edge detectors [ZuH81] to dynamic scenes by considering that the frames are stacked. Haynes and Jain[Hay82,HaJ83] showed that 3-D edge detectors do not give good results when applied to dynamic scenes. The major problem with the application of the 3-D edge detectors in a frame sequence is that the motion of the objects may result in frame-to-frame displacement of the edges that is inappropriate for the resolution of the 3-D detector. When this happens, the 3-D edge detectors can not be applied because the surface coherence implicitly required by these detectors is not satisfied.

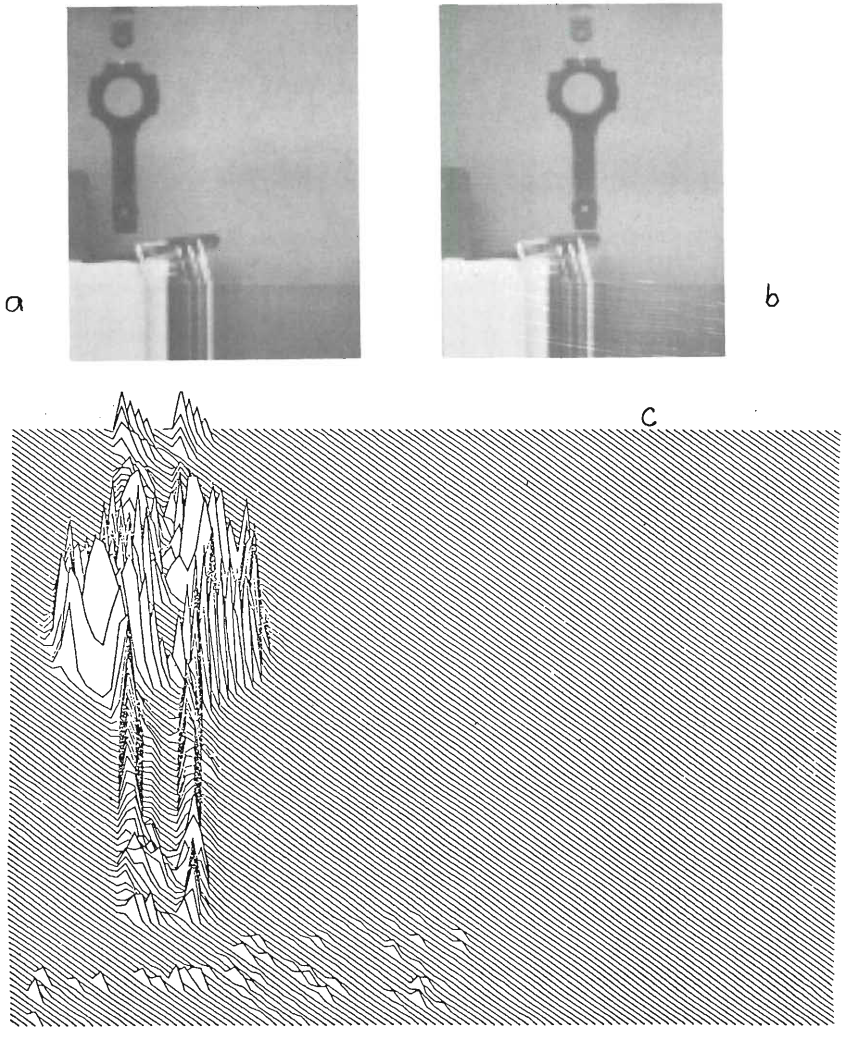
A moving edge is an edge in a frame *and* it moves. Haynes and Jain[HaJ83] argue that moving edges can be detected by combining temporal and spatial gradient using an *and* operator. For *and* they suggest the use of the product. Thus the time varying edginess of a point in a frame, $E(x,y,t)$ is given by

$$E(x,y,t) = \frac{dF(x,y,t)}{dS} * \frac{dF(x,y,t)}{dt}$$

where $\frac{dF(x,y,t)}{dS}$ and $\frac{dF(x,y,t)}{dt}$ are the magnitudes of the spatial and temporal gradients of the intensity of the point (x,y,t). In their experiments they use different conventional edge detectors to compute the spatial gradient and simple difference is used as temporal gradient. These simple operators were applied to many complex scenes, both the SCMO and the MCSO type. It was observed that this edge detector works effectively in most cases. By applying the threshold to the product rather than first differencing and then applying edge detector, as suggested by Jain [Jai81a], or by detecting edges and then computing their temporal gradient, as in [MaU79], this method overcomes the problem of slow moving and weak edges, see Figure 3 and 4. As shown in

Figure 4, this edge detector will respond to slow moving edges that have good edginess and to poor edges that are moving with appreciable speed. Another important fact about this detector is that there is no assumption of small displacement. The performance of the detector is satisfactory even when the motion is very large.

Figure 3 In 3a and 3b we show 2 frames from a sequence and in 3c the edges detected using the edge detector as in [HaJ83]. Compare the edges shown in 3c with those obtained using a 3-D detector, shown in 3d, and using a Sobel edge detector in the frame of 3a, shown in 3e.



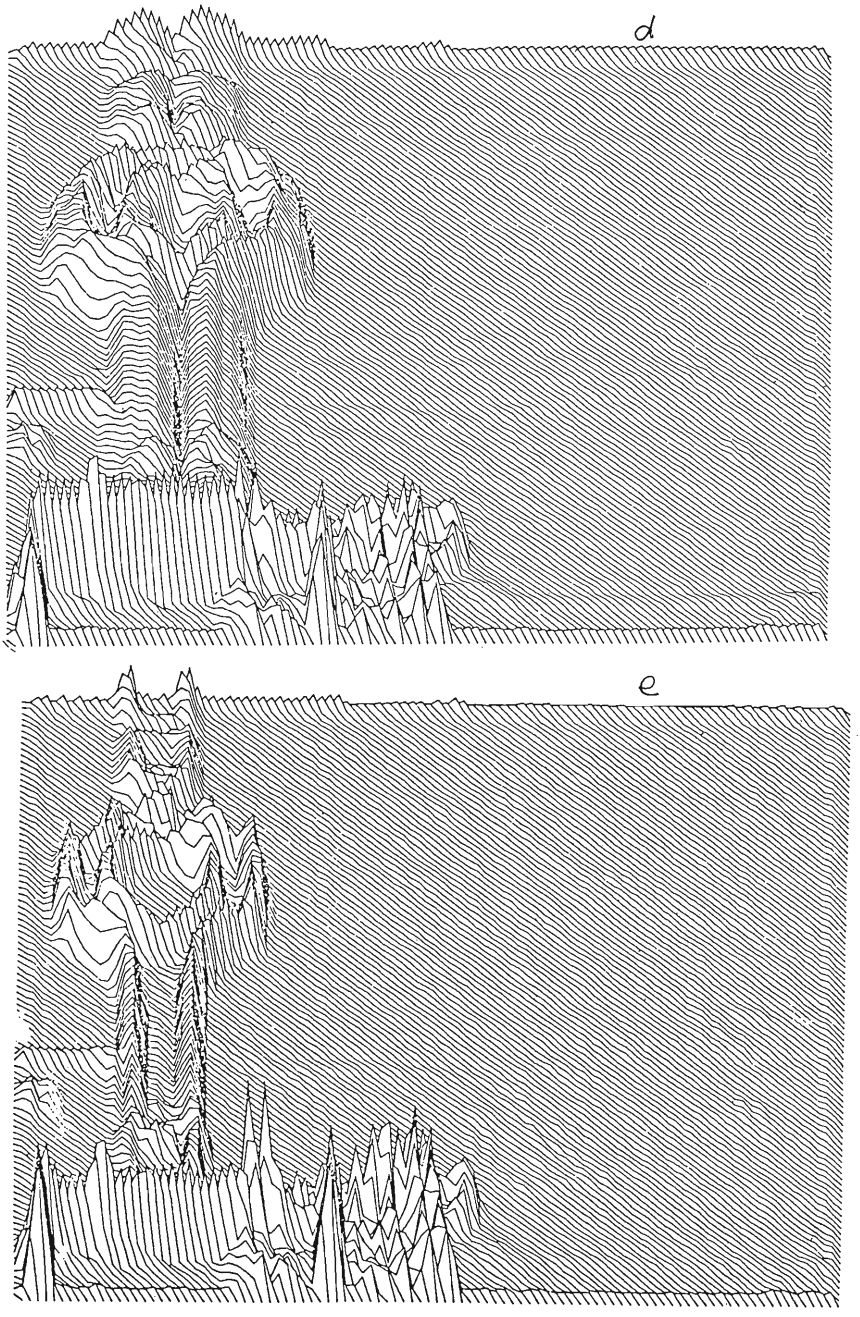
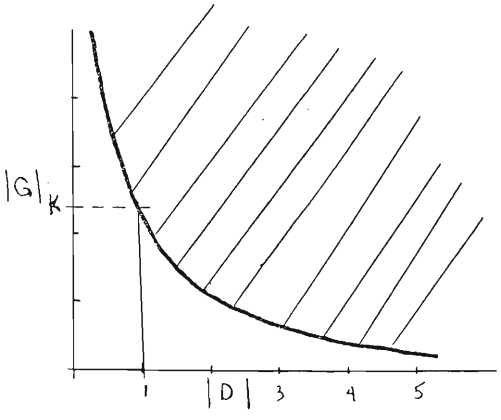

d
e

Figure 4

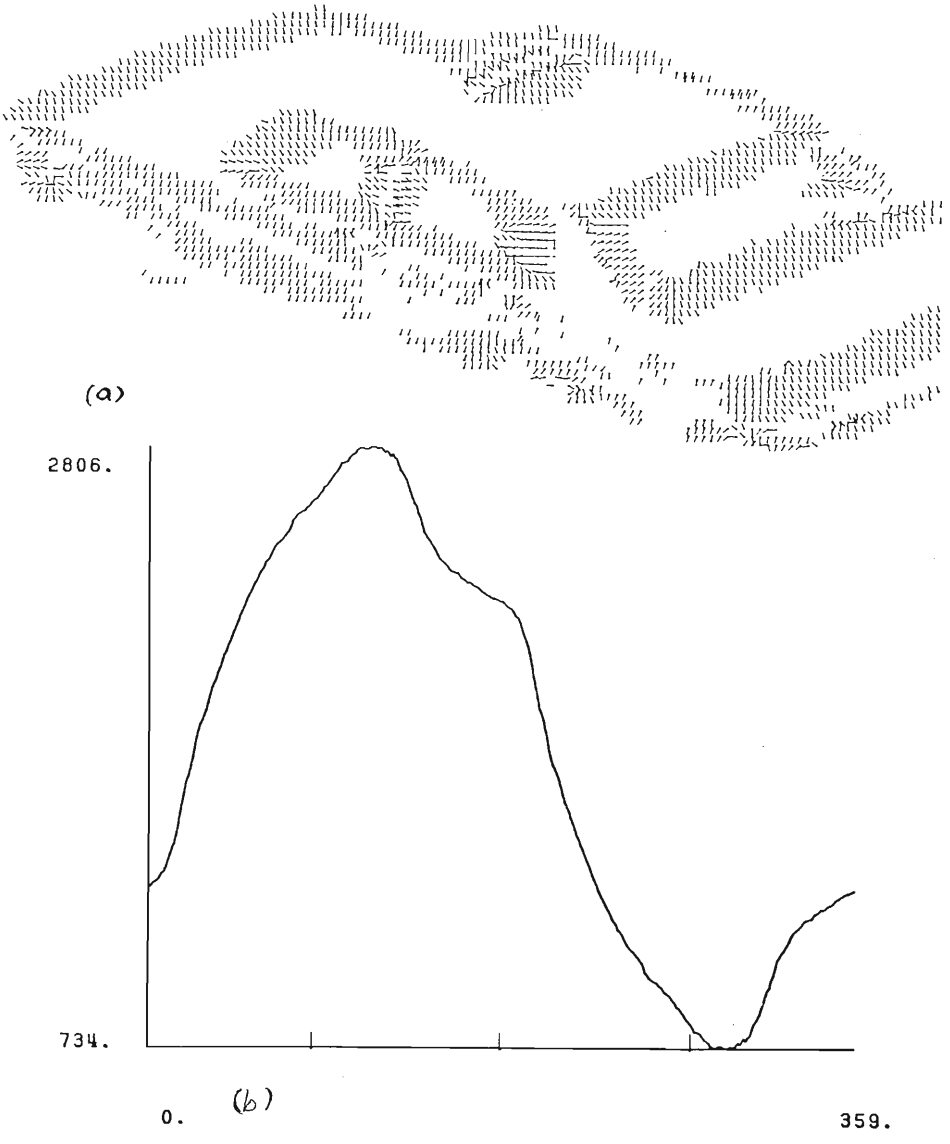
The curve showing the performance of the edge detector. Note that the slow moving edges will be detected if they have good contrast and poor contrast edges will be detected if they move well.



Though the direction of motion of an object can not be determined from the computations at a point, the direction can be determined by integrating the direction information along the edges for an object. The direction of motion computed at a point is within 180° of the direction of the spatial gradient at the point. If we use an accumulator array for an object and for each moving edge point of the object increase the count for all possible directions, then the peaks in the array will give the direction of motion of the object. In Figure 5 we show the direction of motion of an object as determined by this approach.

Figure 5

The direction of motion of every point on the image of a moving object are shown in the Figure 5a. To obtain the direction of motion of the object we used an accumulator array. The peak in the accumulator array indicates the direction of the motion of the object, as shown in the Figure 5b.



Marr and Ullman [MaU79] first find the zero crossings of $\nabla^2 G(x,y) * F(x,y,t)$ to detect edges in each frame of the sequence and then measure the derivative $\frac{\partial [\nabla^2 G(x,y) * F(x,y,t)]}{\partial t}$ at the zero crossings. They show that these measurements constrain the the local direction of motion to within 180°. They argue, as is done by Haynes and Jain [Hay82, HaJ83], that the direction of motion of an object can be determined in the second stage by combining the local evidence at every point of the object. Hildreth [Hil82] has proposed an optimization approach for computing the direction of motion of an object by using components at the boundaries.

3. Using Motion for the Segmentation

In many dynamic scene analysis systems the goal is to recognize moving objects and to find their motion characteristics. If the scene is acquired using a stationary camera, then segmentation generally refers to the separation of moving from stationary components of the scene and identification of individual moving objects based on velocity or some other characteristics. For MCSO and MCMO segmentation task may be same as above or may also include further segmentation of the stationary components of the scene by exploiting the motion of the camera. Most research efforts for the segmentation of dynamic scenes have been concerned with the extraction of the images of the moving objects observed by a stationary camera. It has long been argued by researchers in perception [Gib79, Ull79, Lee80] that motion cues help segmentation. The computer vision techniques for segmenting the SCMO dynamic scenes perform well in comparison to those for the segmentation of stationary scenes. The segmentation of moving camera scenes, into their stationary and nonstationary components, has received attention only recently [Jai82, Jai83b, KoK80, Law81, Law82, Wil80]. The major problem in the segmentation of moving observer scenes is that every surface in the scene shows motion. For separation of moving object images, the motion component assigned to various stationary surfaces in the images due to the motion of the camera, should be removed. The fact that the image motion of a surface depends on its distance from the camera and the surface structure, complicates the situation.

The segmentation may be performed either using region based approaches or edge based approaches. In this section we discuss some approaches for segmentation of dynamic scenes.

3.1. Stationary Camera

3.1.1. Using Difference Pictures

Jain, Militzer, and Nagel [JMN77] proposed an approach based on accumulative difference pictures for the separation of nonstationary component of a frame sequence. Jain and Nagel [JaN79] combine the properties of the ADPs and the DPs (they call them the first order difference picture and the second order difference picture, respectively) for the extraction of images of moving objects. In their approach several properties of a region of an ADP are computed to estimate the direction of motion, and the time for the object image to be displaced completely from its projection in the reference frame. They compute some properties of difference pictures obtained by comparing the reference frame with the current frame for the classification of a region into one of the following four classes:

1. static - where the object is in the reference frame,

2. mobile - where it is in the current frame,

3. o_grow - the part of the projection in the current frame, and

4. b_grow - the part of the projection in the reference frame.

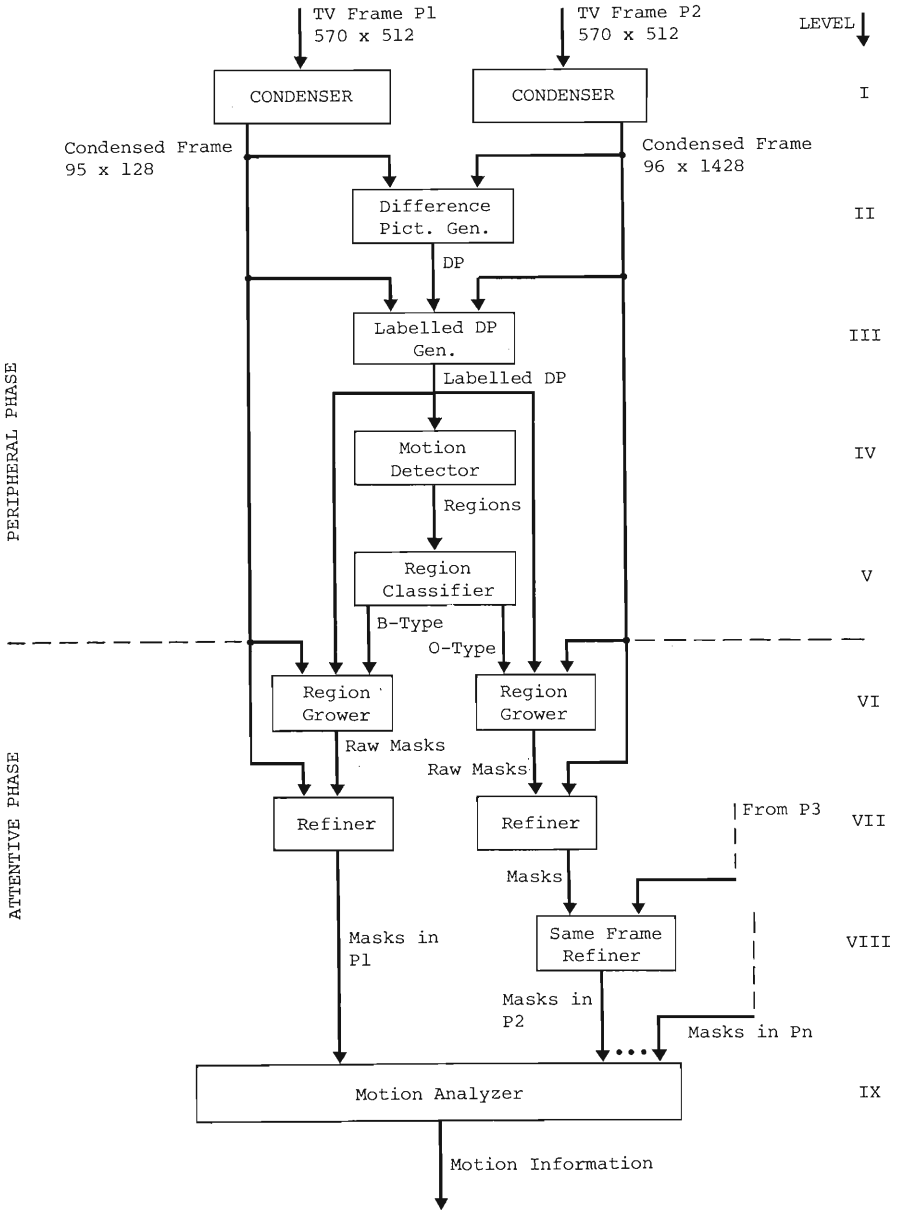
An algorithm was developed to combine the properties of regions in the ADP and the DP for the extraction of the images of the moving objects in the reference and the current frames after the objects are displaced from their projections in the reference frame. As discussed in [JaN79, Jai81b] this approach may be used to construct the reference frame containing only stationary components of the scene by replacing the part of the reference frame corresponding to the moving object by the background after the object images have been recovered. Since such a reference frame gives the images of moving objects in the difference picture, this may play an important role in the subsequent analysis, particularly in object recognition and motion understanding.

This algorithm has been used in many different real world TV frame sequences and with generally good results [Jai81a, DrN81]. A limitation of this algorithm, however, is that the object masks can not be recovered before the object is completely displaced from its projection in the reference frame. The limitation results in the failure of this algorithm in presence of the running occlusion. In many applications, even without running occlusion, it may be required to extract images of moving objects earlier.

Jain, Martin, and Aggarwal [JMA79] exploit the properties of a difference picture for the extraction of the images of moving objects. By comparing successive frames of the sequence, difference pictures are obtained and classified into o_grow and b_grow regions. Starting with the regions in the difference pictures many domain-independent properties of motion are employed for growing a region. The region growing approach depends on the type of region also. The masks thus obtained are further refined by comparing masks of the object in a frame obtained from two different pair of frames. The outline of the approach is shown in Figure 6. It was shown that this region growing approach results in good masks for the moving objects from two or three frames. A possible hardware implementaion of this approach for real time segmentation is discussed in [AgJ82]. A similar approach for segmentation is proposed in [YMA82] using some connectivity properties of difference pictures. Tang et al [TSR82] propose that the segmentation may be performed by finding difference region and then studying their frame-to-frame changes.

Figure 6

This figure gives the information flow in the algorithm for segmentation by Jain, Martin, and Aggarwal. Starting with the difference pictures, several refinements are applied to the regions to obtain the masks for the moving objects.


TV Frame P1
570 x 512
TV Frame P2
570 x 512
LEVEL
CONDENSER
CONDENSER
I
Condensed Frame
95 x 128
Condensed Frame
96 x 1428
Difference
Pict. Gen.
II
DP
Labelled DP
Gen.
III
Labelled DP
Motion
Detector
IV
Regions
Region
Classifier
V
B-Type
O-Type
PERIPHERAL PHASE
Region
Grower
Region
Grower
VI
Raw Masks
Raw Masks
ATTENTIVE PHASE
Refiner
Refiner
From P3
VII
Masks
Masks in
P1
Same Frame
Refiner
VIII
Masks in
P2
Masks in Pn
Motion Analyzer
IX
Motion Information

Recently, Jain [Jai83b, Jai83d] showed that using ADPs it is possible to segment a scene with very little computation. He defined absolute, positive and negative difference and accumulative difference pictures as following:

$$DP_{12}(x,y)=1 \text{ if } |F(x,y,1)-F(x,y,2)| > T$$

$$=0 \quad otherwise$$

$$PDP_{12}(x,y)=1 \text{ if } F(x,y,1)-F(x,y,2)>T$$

$$=0 \quad otherwise$$

$$NDP_{12}(x,y)=1 \text{ if } F(x,y,1)-F(x,y,2)<-T$$

$$=0 \quad otherwise$$

and

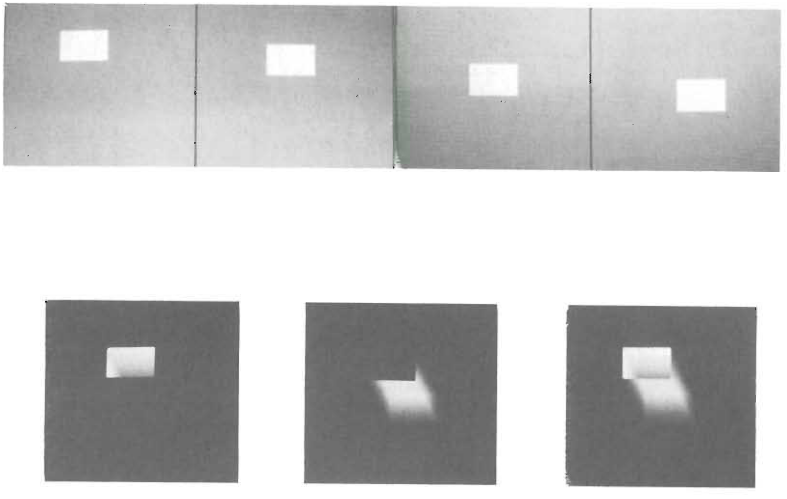
$$AADP_n(x,y)=AADP_{n-1}(x,y)+DP_{1n}(x,y)$$

$$PADP_n(x,y)=PADP_{n-1}(x,y)+PDP_{1n}(x,y)$$

$$NADP_n(x,y)=NADP_{n-1}(x,y)+NDP_{1n}(x,y)$$

Figure 7

Figure 7a shows frame 1, 5, 10, and 15 of a scene containing a moving object. The intensity coded positive, negative, and absolute ADPs are shown in figures 7b, c, and d, respectively.

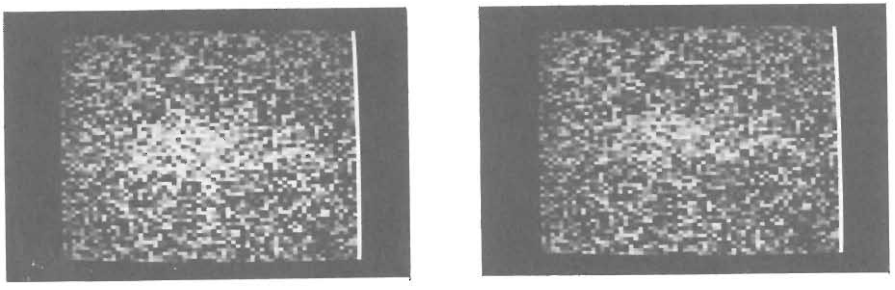


It is shown that in one of the ADPs (either PADP or NADP) the region due to the motion of the object continues growing even after the object has been completely displaced from its projection in the reference frame while in the other ADP it stops growing. The entries continue increasing in value in the area covered by the projection in the first frame. The accumulative difference pictures for a synthetic scene are shown in Figure 7. To obtain the mask of an object in the reference frame a test to check whether a region is still growing is required. The mask in the current frame may be obtained by considering the accumulative difference picture of the other kind. This method was applied in [Jai83b]. This approach, however, in its simplest form has the same limitation as the original approach based on the accumulative difference pictures proposed in [JaN79], namely, it can extract images of moving objects only after the object has completely displaced from its projection in the reference frame. It appears, however, that by computing properties in difference and accumulative difference picture, it is possible to segment images in complex situations, such as in running occlusion, also [Jai83d]. To avoid running occlusions from disruption of the segmentation process, the segmentation process should not wait until the object image is completely displaced from its position of the reference frame. In the system reported in [Jai83d], the region in the accumulative difference pictures are monitored to find the regions in place of the reference frame position of the object. It is shown that a simple test on the rate of increase of regions and on the presence of the *stale* entries allows the determination of the regions that are finally going to mature and result in the mask of the objects in the reference frames. The early determination of the reference frame position of the objects and hence extraction of masks for the objects allows necessary action to prevent the running occlusion.

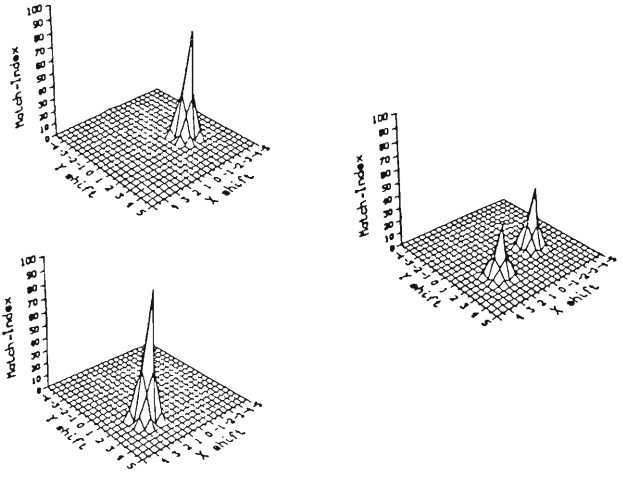
An approach for the segmentation of scenes containing textured objects and textured backgroud was proposed by Jayaramamurthy and Jain [JaJ82, JaJ83]. This approach also starts with the difference regions, then uses a shift-match and Hough transform to detect moving objects and their displacement component. The basic idea in this approach is to assume some displacement component for the objects and to verify the hypothesis using the structure of the intensities in the neighborhood of the points. A very attractive feature of the approach is the possibility of recovering objects images and the motion characteristics in presence of occlusion in textured scenes. In Figure 8 we show a textured sequence in which it is difficult to find what is happening. The algorithm successfully extracts the masks of the moving images and determines their motion characteristics.

Figure 8

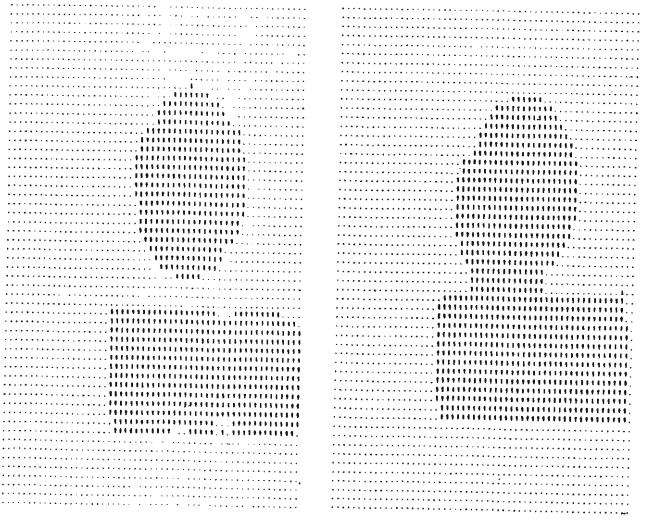
In Figure 8a and 8b two frames of a scene containing textured objects against textured backgrounds are shown. It is difficult to detect the objects and their motion. The approach presented in [JaJ82, JaJ83] gives the direction of motion and the images of the moving objects. The direction of motion is indicated by the accumulator array for the displacement component, shown in the Figure 8c, and the image of the moving objects are shown in the Figure 8d.



(a)


Match-Index
Y shift
X shift
Match-Index
Y shift
X shift
Match-Index
Y shift
X shift

(i) (ii)



3.1.2. Using Velocities

Fennema and Thompson [FeT79] exploit the relationship, (see also [LiM75, CaR76]), between the spatial and temporal gradients of intensities to segment images. Using the relation

$$\frac{dF(x,y,t)}{dt} = -(F_x.u + F_y.v)$$

they show that if the direction of the velocity and intensity gradients at a point are ϑ_v and ϑ_g , respectively, and the velocity over a surface is constant, then values of v_g plotted against ϑ_g will lie along a cosine curve. Assuming sufficient variations in ϑ_g for the surface, the relationship between v_g and ϑ_g will uniquely define v. They use a clustering approach to determine the dominant velocities in the scene for the classification of each point.

Note that this approach is based on two assumptions: the linear relationship between the temporal and spatial gradient, and constant velocity of the object. To satisfy the first assumption, they require blurring of the images. The constant velocity component requirement means that this approach will not work in the presence of rotation. The results for the real world scenes containing translating objects are encouraging.

Thompson [Tho80] suggests that the velocity information be combined with the intensity information for segmentation. Starting with the velocity information one may extract regions of uniform velocity and then use region growing based on intensities for the extraction of the images of the moving objects. It appears that rotation will still cause problems for this approach. Velocity values at different points of a rotating object will be different in a frame sequence and may not form a good cluster for starting region growing based on intensity values. Occlusion of moving object will also give difficult time to the algorithm based on this approach.

Potter [Pot74] used template matching for segmentation of a dynamic scene. His approach is very sensitive to noise and may have difficulties even with laboratory scenes.

3.2. Moving Camera

If the camera is moving then every point in the image, with the exception of pathological case of points that are also moving with the same velocity, has non-zero relative velocity. The velocity of points depends on their distance from the camera and their own velocity. The difference picture based approaches may be extended for the segmentation of the scene, but if the aim is to extract the images of moving objects then more information will be required to decide whether the motion component at a point is due to its depth alone or is a combination of components due to it's depth and motion. The gradient based approach will also require additional information.

If the direction of the motion of the camera is known then the FOE with respect to the stationary component of the scene can be easily computed since the FOE will have coordintaes

$$x_f = \frac{dx}{dz}$$

and

$$y_f = \frac{dy}{dz}$$

in the image plane. As discussed in a later section, the velocity vectors for all stationary points of the scene project on the image plane so that they will intersect at the FOE. Jain [Jai82, Jai83b] proposed the Ego-Motion Polar transform of an image such that a frame $F(x,y,t)$ is transformed to $E(r,\vartheta,t)$ using

$$E(r,\vartheta,t) = F(x,y,t)$$

where

$$r = \sqrt{(x - x_f)^2 + (y - y_f)^2}$$

and

$$\vartheta = Tan^{-1}\left(\frac{y - y_f}{x - x_f}\right)$$

In the EMP space all stationary points show displacement along the ϑ axis, the points that belong to moving objects have a displacement component along the r axis also. Thus by using any technique for the stationary camera case, discussed above, we may determine the displacement component in the EMP space and segment a scene into its stationary and non-stationary components. The results of the experiments reported in [Jai83b] with real scenes are very encouraging. In Figure 9 we show 3 frames of a sequence acquired using a moving camera; the results of the segmentation are shown in Figure 10.

Figure 9

Three frames of a scene acquired using a moving camera are shown in Figure 9a, b, and c.

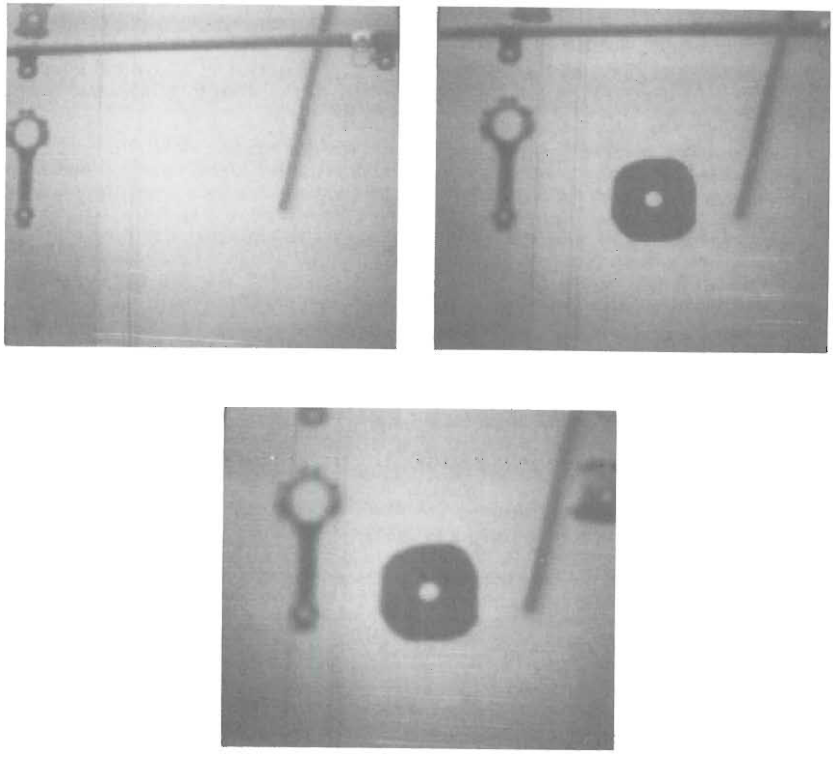
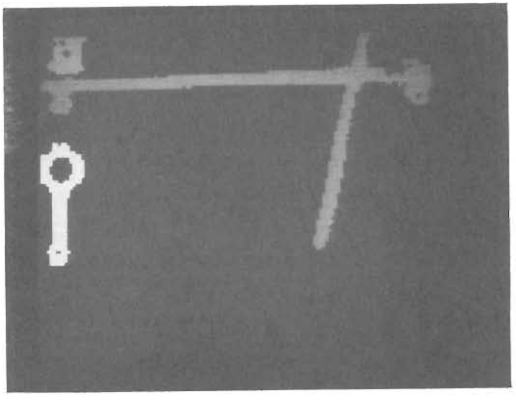


Figure 10

The segmentation of the frames shown in Figure 9 as obtained by [Jai83b] is shown here. The moving objects are brighter.



It appears that more information about moving as well as stationary objects may be extracted using a complex logarithmic mapping rather than the simple polar mapping about the FOE. This mapping will be discussed in a later section.

3.3. Edge-Based Methods

If moving edges are detected, then images of the moving objects in the case of the SCMO can be obtained easily. The segmentation of the static and dynamic component of the scenes may require techniques similar to that discussed in a previous section for the MCMO. No efforts have been made to extract masks of the moving objects from moving edges obtained using a moving edge detector such as [HaJ82, HaJ83]. The results of the moving edge detector for different real world scenes indicate the feasibility of the approach. A problem in this endeavor is likely to arise due to missing edges. The edge detectors fail on the points for which the motion is along the spatial gradient. The boundaries of an object obtained using a moving edge detector may, therefore, require some technique for filling such gaps in the boundary of the object.

4. Optical Flow

Gibson [Gib66, Gib'79] proposed the concept of optical flow in his theory of ecological optics. Optical flow is the distribution of velocity at each point of the image relative to the observer. It has been shown that optical flow carries valuable information for the analysis of dynamic scenes [Lee80, Clo80, Pra80, LoP80, Pra82]. Some methods have been proposed for the extraction of the information assuming that optical flow is available. Techniques developed for the computation of optical flow do not, however, result in the optical flow of the quality required by the information recovery processes. In this section we first discuss current methods for the computation of the optical flow and then consider the intrinsic information and its recovery.

4.1. Computation of the Optical Flow

The optical flow is determined by computing the velocity vector at each pixel of the image. Several schemes have been devised for obtaining optical flow from two or more frames of a sequence. These schemes can be classified in two general categories: feature based and gradient based. Interestingly, though optical flow is more useful for a moving camera, in most of the proposed methods for the computation of the optical flow the results are shown for scenes containing moving objects acquired by a stationary camera. When the camera is stationary, most points in a frame have zero velocity, since usually a very small subset of a scene is in motion. The main application of the optical flow is in the case of a moving camera; it is implicitly assumed that the techniques developed for the stationary camera scenes will be extensible to the mobile camera too.

4.1.1. Feature Based Methods

These methods compute optical flow in the form of disparity vectors by first selecting some features in frames and then using matching to solve the correspondence problem. Barnard and Thompson demonstrated that disparities may be computed using relaxation [BaT79]. This approach was discussed in section 2.2. As discussed there, the problem of selection of features and establishing correspondence is not an easy one. Moreover, this method gives velocity vectors only at sparse points. Nagel [Nag82b] and Nagel and Enkelmann [NaE82] have recently proposed a method based on the second order intensity variation to compute velocity vectors for corner points and then use relaxation to obtain vectors for the missing points. He obtains the displacement vector at a point by minimizing the function

$$MD = \sum [F(x,y,t_1) - F(x-\delta_x, y-\delta_y, t_2)]^2$$

where δ_x and δ_y are displacement components in the x and y directions, respectively.

He developed a formalism for iterative refinement of a displacement estimate for image locations around a corner. The displacement estimate obtained at the corner may then be used to compute the estimates in the neighborhood of the corner point. This approach, thus, starts at the corners and propogates the displacement field over the image. Hill and Jain [HiJ83] use intensity profiles in conjunction with a bilinear-recursive hill climbing approach to obtain the velocity vectors to sub-pixel precision and then used relaxation across several frames to refine the vectors obtained over a sequence. No effort was made to propogate the vectors to the neighborhood. The behavior of relaxation algorithms for the propagation of velocity vectors in images is not yet understood and hence efforts for propagation are usually sensitive to noise and occluding boundaries in images.

No experience is yet available for scenes obtained using a moving camera for the computation of optical flow using the feature based approach. It appears that the difficult problem in this case would be that of extrapolating, or filling, the disparity vectors to obtain the optical flow. Since most approaches obtain disparity vectors only to the resolution determined by the feature locations, the extrapolation may result in noisy

optical flow.

4.1.2. Gradient Based Methods

The methods in this class exploit the relationship between the spatial and temporal gradients of intensity at a point in a frame. Limb and Murphy[LiM75] and Cafforiao and Rocca [CaR76] use the relationship to compute velocity at points for reduction of bandwidth for transmitting TV images. As discussed earlier, Fennema and Thompson [FeT79] applied this to the segmentation of images using the velocity of points. These researchers use the relation:

$$F_x u + F_y v + F_t = 0$$

where $u = \frac{dx}{dt}$ and $v = \frac{dy}{dt}$ and F_x, F_y, and F_t are the partial derivatives of image intensity with respect to x, y, and t, respectively. This equation can be written in the following form:

$$(F_x, F_y).(u, v) = -F_t$$

Fennema and Thompson [FeT79] compute

$$V_g = -\frac{F_t}{\sqrt{F_x^2 + F_y^2}}$$

where V_g is the component of the velocity in the direction of the intensity gradient in an image. They assumed that surfaces have uniform velocity and computed the velocity component using Hough transform. Their method will not work in the presence of rotation due to the assumption of uniform velocity. Moreover, this approach is good for segmentation of images, as it gives velocity for an object rather than velocity at every point.

Horn and Schunck [HoS81] assume that a velocity field varies smoothly everywhere in an image. They developed an iterative approach for the computation of the optical flow using two or more frames. The following iterative equations were used for the computation of the optical flow:

$$u = u_{av} - f_x \frac{P}{D}$$

$$v = v_{av} - f_y \frac{P}{D}$$

where

$$P = f_x u_{av} + f_y v_{av} + f_t$$

$$D = \lambda^2 + f_x^2 + f_y^2$$

In the above equations f_x, f_y, f_t, and λ represent the spatial gradients in the x and y directions, the temporal gradient, and a multiplier, respectively. For the case of two frames the method was iterated on the same frames many times, in case of more frames each iteration used a new frame. They demonstrate their method with synthetic frame sequences. The smoothness constraint is not satisfied at the boundaries of objects because the objects may be at different depths. If the objects are moving then also the constraint will be violated. The abrupt changes in the velocity field at the boundaries cause problems in this approach. Schunck and Horn [ScH81] discuss some heuristics to solve these problems. Our experience in applying this approach to real scenes [Bor82] showed that the optical flow computed using this approach is very noisy.

An important fact about gradient based methods is that they assume linear variation of the intensities [HLS80] in images and compute the velocity at a point using this

assumption. Typically, it is expected that such an assumption will be satisfied at the edges in images and hence velocity can be computed at the edge points. Fennema and Thompson [FeT79] blur their images to satisfy this condition; Horn and Schunck [HoS81] use relaxation to fill in the regions of uniform intensity. Some efforts have been made to combine the feature matching with the gradient approaches for the computation of the optical flow [Gla81,Yac81]. Nagel [Nag82a, Nag82b] argues that the linear model of intensity variation at the edges is too simplistic; he proposes a quadratic model for the computation of the velocity. The intensity at a point in an image must be represented by

$$I(x,y)=I(x_o,y_o)+I_x(x-x_o)+I_y(y-y_o)$$

$$+\frac{1}{2}I_{xx}(x-x_o)^2+I_{xy}(x-x_o,y-y_o)+\frac{1}{2}I_{yy}(y-y_o)$$

where

$$I_{xx}=\frac{\partial^2 I}{\partial x^2} \quad I_{xy}=\frac{\partial^2 I}{\partial x\,\partial y} \quad I_{yy}=\frac{\partial^2 I}{\partial y^2}$$

The results obtained, using this approximation, for real world frame sequences are encouraging.

4.2. Information in Optical Flow

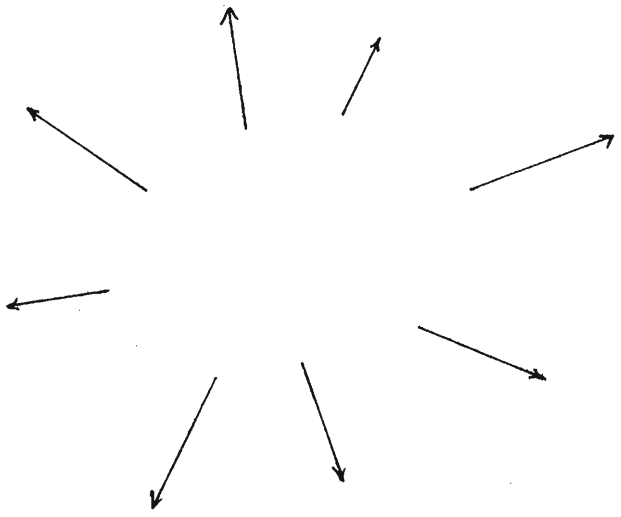
Assuming that somehow high quality optical flow has been computed, researchers have studied what kind of information is available in extractable form in the flow field. Clocksin [Clo80] and Prazdny [Pra80, LoP80, Pra81] addressed the computational aspect of this problem. They assume that the environment contains rigid stationary surfaces at known depths and that the observer, the camera, locomotes through this world. The optical flow can be derived from the known structure. They then show how to invert the process giving the structure of the environment from the computed optical flow field.

Clocksin [Clo80] argues that areas with smooth velocity gradients relate to a single surface and can give information about the structure of the surface. The areas with large gradients give information on occlusion and boundaries because only two different objects at different depths, can move at different speeds relative to the camera. Using an observer-based coordinate system, he derives a relationship to recover the surface orientation from the smooth velocity gradients. The orientation is specified with respect to the direction of the motion of the observer.

Longuet-Higgins and Prazdny [LoP79] allow for rotational motion of the observer. They used the fact that in this case the velocity field is the vector sum of a translational component and a rotational component. The translational component is directed towards a point, called the Focus of Expansion (FOE, for the approaching motion of the observer) or the focus of contraction (for the receding motion of the observer), in the image. This is shown in Figure 11. This point is the intersection of the direction of motion and the image plane. The structure of surfaces can be recovered from the first and second spatial derivatives of the translational component. The rotational component is fully determined by the angular velocity.

Figure 11

The velocity vectors due to the stationary components of the scene, for a translating observer, meet at the FOE.



The importance of the FOE for the recovery of the structure from the translational component of optical flow encouraged several researchers to develop methods for the determination of the FOE. Jain [Jai83a] proposes a method, using a geometric argument, based on the minimization of the disparities of tokens in individual frames. The FOE in an image is the point at which the function

$$M(\hat{O}) = \left[\sum_{k=1}^{k=n} L(\hat{O}P^1) - \sum_{k=1}^{k=n} L(\hat{O}P) \right]$$

is extremized. In the above equation $L(.)$ is the Euclidean distance between two points, P^1 and P are the tokens in the second and first frames, $\hat{O}$ is a point in the search space which may be considered a tentative choice for the FOE, and n is the number of tokens in frames. The attractive feature of this method is the fact that no correspondence is required. Another attractive feature of this approach is that the search space is gradual, allowing application of the irrevocable search methods such as gradient techiques. Figure 13 shows the function for an approaching observer for the two frames shown in Figure 12. Emerging and disappearing tokens may create problems for this approach. Efforts to apply this to real scenes have not been encouraging [JeJ83a]. Lawton [Law81, Law82] combines the determination of the FOE and the extraction of tokens and shows that some success can be achieved in real scenes. Prager [Pra79] has emphasized the importance of the known value of the FOE.

Figure 12

A composite frame showing the locations of the tokens in two frames. The features for different frames are shown marked x and o.

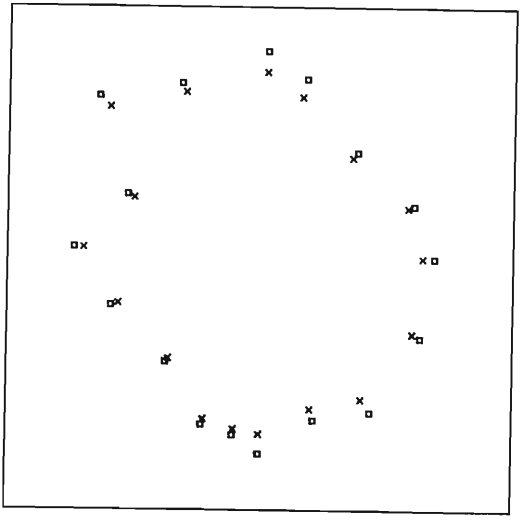
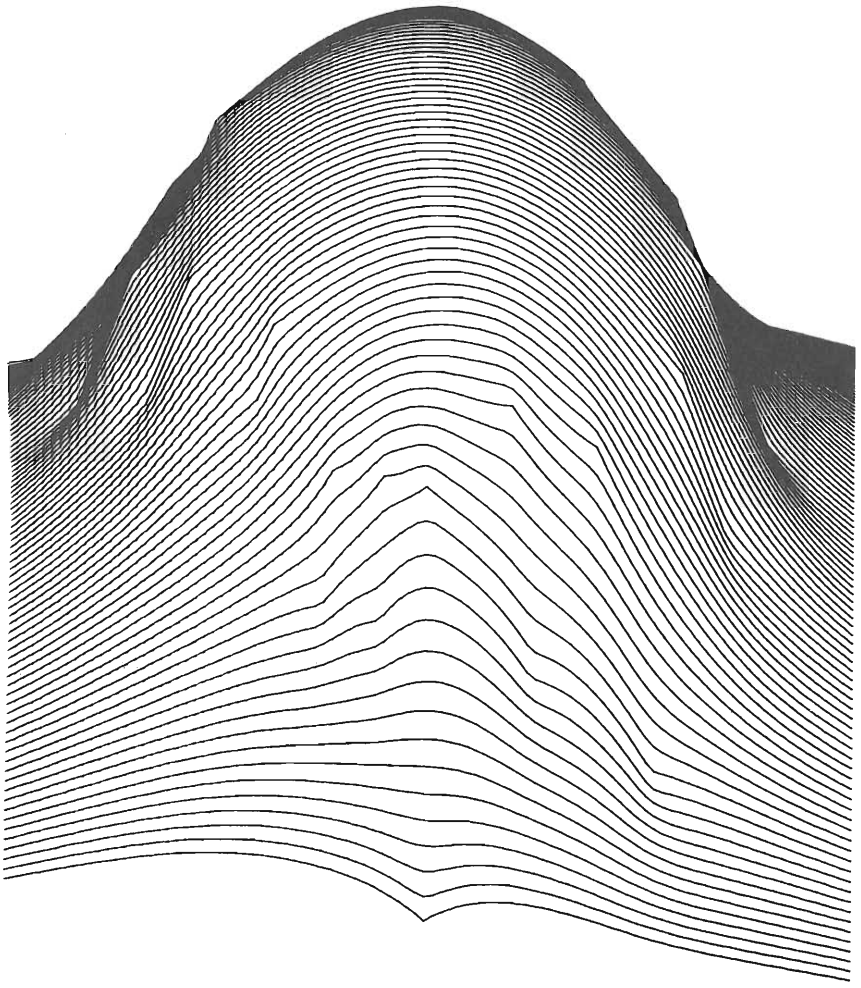


Figure 13

The function for computing the FOE for the frames shown in the figure 12. Note that the function is gradual and hence gradient techniques may be used for the determination of the FOE.



Jerian and Jain [JeJ83a] point out that after the FOE is determined using a direct method, it should be used to solve correspondence problem. The next step is to recompute the FOE using the matching thus obtained. It is expected that this type of approach may result in better correspondence and a more precise location of the FOE.

If the FOE is correctly determined then it may be used for the computation of the translational component of the optical flow. Since all flow vectors meet at the FOE, the direction of the flow vector is determined, only magnitude remains to be computed. Thus, the two dimensional problem of the computation of the optical flow is reduced to a one dimensional problem. This fact has been specified by many researchers, but not applied, possibly due to the uncertainty in the location of the FOE in real scenes using the proposed approaches.

By detecting edges in optical flow field, one may detect depth edges in frames. Thompson et. al. [TMB82] propose an approach for edge detection in the flow field. The flow field is computed in hierarchical structures using

5. Recovering 3-Dimensional Information

Most systems for the determination of motion parameters were concerned with the parameters in the image plane, i.e. in 2-D space. The interpretation of 2-D displacements in terms of 3-D motion is much more complicated due to the fact that the picture formation process results in a 2-D projection of the 3-D events leading to a loss of information. The recovery of the 3-D motion parameters and the 3-D structure of the objects has attracted much attention from researchers in the last few years. Ullman [Ull79] argues that the *rigidity* assumption about an object may help in the recovery of the structure. The rigidity assumption states that any set of elements undergoing a 2-D transformation which has a unique interpretation as a rigid body moving in space should be so interpreted. The research in human perception [Tod82, Joh76] suggests that the human visual system exploits this fact.

The research in the recovery of 3-D structure from image sequences may be divided in two general classes. Some researchers have been concerned with the problem of recovery of the structure and motion using a minimal number of points in a minimal number of frames. Recently, the trajectory based recovery has attracted some attention.

5.1. Recovering Structure Using Tokens

Suppose that we apply an interest operator to consecutive frames of a sequence and extract some interesting points or tokens, such as corners, then using some method discussed in an earlier section, manage to succeed in solving the correspondence problem. Ullman shows that if token correspondences have been established, it is possible to recover the 3-D location of four noncoplanar points from their 3 orthogonal projections. This gives an implicit 3-D structure of the object. He shows that if the points are noncoplanar then a unique structure can be recovered; in the case of coplanar points, the structure, to a reflection, can be recovered. Note that these results are obtained for the orthographic or parallel projection. For the case of perspective projections, two views of five non coplanar points are required. The equations for the recovery are nonlinear and require iterative methods for the solution. Williams uses a heuristic method [Wil80] for the recovery. Roach and Aggarwal [RoA80] present results of their efforts to solve the resulting nonlinear equations using standard techniques available under IMSL. In their studies they observed that in the presence of noise, the minimal solution does not give correct results; significant overdetermination (2 views of 15 points, or 3 views of 8 points) may be required.

Nagel and Neumann [NaN81] give a compact representation of the nonlinear equations required for the specification of 3-D rotations of rigid objects. They show that Ullman's polar equations are a special case of their solution. They derive the following

compact relation for the recovery of structure:

$$(X_{p21}\ X\ X_{p22}D')X'_{p11}(X_{pm1}\ X\ X_{pm2}D')DX'_{p12}=$$

$$(X_{p21}\ X\ X_{p22}D')DX'_{p12}(X_{pm1}\ X\ X_{pm2}D')X'_{p11}$$

where X_{pmn} denotes a vector representing image plane coordinates in the form $[x_p, y_p, f]$ where f is the focal length of the imaging device, of m-th object point in n-th frame, and D is the rotation matrix. For m =3,4,5 this equation gives a set of three equations for the determination of 3 rotation parameters.

Tsai and Huang [TsH81,THZ82] introduce eight *pure parameters* for the case of a rigid planar patch undergoing general 3-D motion. Using the Lie group theory of transformations, they show that for two given successive views the solution is unique. From computational point of view, a very attractive feature of their approach is that the parameters can be computed by solving a set of linear equations. They demonstrate that though theoretically 6 solutions are possible, practically the maximum number of solutions is two. In [THZ82] it is shown that the actual motion parameters can be estimated by computing the singular value decomposition of a 3 X 3 matrix consisting of the eight pure parameters and that the number of solutions is either one or two. This approach has been extended to rigid objects with curved surfaces in [TsH82], where it is shown that seven points in two frames are required to uniquely estimate the 3-D motion parameters. The points should not be traversed by two planes with one plane containing the origin, nor by a cone containing the origin. For the estimation of 3-D translation by solving a set of linear equations, derived using a set of 8 points in two frames, *essential parameters* were introduced.

By avoiding uncertain and time consuming iterative methods required for the solution of non linear equations, one can hope to recover the 3-D motion in realistic situations in real time. An effort [JeJ83b] to apply the approach using the IMSL package to recover the motion parameters in a real scene shows, however, that the proposed approach is very sensitive to the location of points. It was observed that a very high precision in the location of the tokens may be required to obtain reliable results.

The feature based methods for the recovery of the structure or for the estimation of the motion parameters require two difficult steps before they are applied: the precise location of the tokens or points, and the correspondence. If we apply interest operators based on small neighborhoods, then the number of tokens extracted in a real image is very large making correspondence a difficult problem. The operators based on a large neighborhood and higher order greylevel characteristics do result in a reasonable number of tokens, reducing the complexity of the correspondence, but their location may not be precise. Even if the location is obtained at the resolution of the pixel, it appears that results obtained using the methods discussed above may not be reliable.

5.2. Trajectory Based Methods

All above methods depend on a set of points in two or three frames. If a token is traced over several frames, by solving correspondences, then one obtains the 2-D trajectory of the point. It appears that the efforts to recover the structure and motion in 3-D from the trajectories may be more reliable than those based on a set of features in a few frames. A trajectory may be interpolated by using curve fitting techniques to obtain a better resolution in the 2-D path. Moreover, experiments by Dreschler and Nagel [DrH81] and Hill and Jain [HiJ83] show that the correspondence problem may be simplified by considering more than two frames and extending relaxation across frames.

Webb and Aggarwal [Web81, WeA82b] propose the use of several monocular views for recovering the 3-D structure of moving rigid and jointed objects. They assume that a general motion of objects may be considered, over at least a short interval, as a rotation about a fixed axis and a translation. The fixed axis assumption allows recovery of the structure, under parallel projection, for even two points. If a point on the object is fixed

then the other points trace out circles in planes normal to the axis of rotation. These circles are projected as ellipses under parallel projection. The structure of points can be recovered to within a reflection by finding the parameters of the ellipse.

Jointed objects are composed of two or more rigid objects. Webb and Aggarwal [WeA82b] present an algorithm, assuming that the feature selection and the correspondence problems have been solved and that the fixed axis assumption is satisfied for each rigid component, for identifying jointed components and for recovering their structure. The most attractive feature of this approach is that the recovery requires fitting an ellipse and hence the recovery method is not as sensitive as the methods based on 2 or 3 frames.

Recently, Sethi and Jain [SeJ83] have extended the trajectory based approach using perspective projections. They have shown that rotation of a point about a fixed axis results in an ellipse in the image plane. For the case of rotation about an axis parallel to the X or Y axis the 3-D coordinates can be recovered, to a scale, using one point only. Their work is inspired by some experiments of Todd [Tod82], who shows that humans appear to be interpreting rigid and non-rigid rotations based on trajectories. Sethi and Jain [SeJ83] shows that if a point is rotating about an axis parallel to the Y axis that passes through a point z_o on the Z axis, then the equation of the ellipse can be written in the general form:

$$\frac{(x_p - h)^2}{a^2} + \frac{(y_p - k)^2}{b^2} = 1$$

The following relations can be used to recover the structure from the parameters of the ellipse:

$$y = \frac{2b^2}{a^2 k}(1 + z_o)$$

$$x = y\frac{x_p}{y_p}$$

and the radius of the rotation is

$$r = \frac{b}{k}(1 + z_o)$$

It was shown further that if the axis of rotation passes through a point x_o, then the major and minor axes of the ellipse are oriented differently. By using transformations given by

$$\hat{x}_p = x_p - \frac{y_p}{y} x_o$$

and

$$\hat{y}_p = y_p$$

it is possible to transform the ellipse to the standard form and use the standard equations to recover the structure. The experiments with the synthetic data are encouraging.

The case of an arbitrary axis was not considered for the recovery of structure in [SeJ83]. Since the parameters of an ellipse can be obtained using only 5 points, it is possible to recover the structure from 5 frames. Note, however, that this method considers rotation about a fixed axis. It will be interesting to consider trajectories by relaxing the fixed axis assumption and see whether structure may be recovered.

6. Motion Understanding

The analysis of a frame sequence may result in the extraction of the images of moving objects, their 3-D structure, and their frame-to-frame displacements. In many applications the aim may be to name moving objects and to describe the events taking place in the scene. The recognition of the object may be performed using an image or the 3-D structure of the object. It appears that the motion characteristics of objects may also help in the recognition. Different objects have different kinds of motion: humans walk, cars run, aeroplanes fly. The motion characteristics of objects are difficult to obtain directly from the frame-to-frame displacement. Much of the recognition and analysis of things by humans does not require fine detail and precise analysis of small details and parts. Details are only brought into play when the object or the process is the focus of attention.

An object with complex motion in each of its several parts can be abstracted to a simple moving block undergoing rigid motion. The abstracted motion can be simple translation or rotation, whereas the real motion may be very complex. At the next level of analysis one may try to analyze the motion of parts of the object; the knowledge of the motion of the abstraction of the object may help in the analysis of the motion of individual components of the object. It appears that a correct approach is to *somehow* compensate for the known motion of a higher level abstraction of the object to get detailed motion of the lower level parts. Jerian and Jain present their approach for such a system in [JeJ83b].

6.1. Motion Representation

Asada, Yachida, and Tsuji [AYT81] present some interesting experiments in understanding motion of blocks in 3-D space. Blocks can be in complex configurations and may be undergoing coincidental motion, such as a block rotating around a joint attached to another moving block. Assuming orthogonal projection and exploiting knowledge about the blocks-world they solved the correspondence problem. The most interesting and the novel part in their system is the hierarchical repesentation for articulated motion. The complex rotation of objects is interpreted as the combination of simple rotations of individual blocks. Suppose that there are three blocks, A, B, and C. A is rotating about some fixed axis, B is connected to A and is rotating about some other axis, and C is connected to B and is also rotating about yet another axis. It is possible to recognize the motion of A; but motion of B and C becomes complex. They show that a motion comprising a translation and rotation between two frames may be represented as a rotation. They developed methods for obtaining axis of rotation and the rotation angle to describe an arbitrary motion. If the orientations of an axis as determined from several frames forms a close cluster then the object is assumed to be undergoing simple motion. In the above situation first the motion for A is determined; for B and C there is no close cluster. After the motion of A is determined, the motion of B is obtained by transforming its motion to the coordinate system fixed to the object A. The motion of C can be determined similarly by transforming its motion to a coordinate system fixed to B. The experiments show that this approach is successful in the analysis of complex motion. Using projections on the Gaussian sphere, this approach may be extended for perspective cases [AYT82].

A method for the representation of the motion of objects in images, acquired using a moving camera, is proposed by Jain[Jai82, Jai83b, Jai83c]. This approach is based on the fact that all velocity vectors corresponding to the stationary objects in a scene acquired using a translating observer intersect at the FOE. Using the FOE as the origin, we may convert a frame to a second frame in which the abcissa is r and the ordinate is ϑ. Using this transformation, as discussed in an earlier section, it is possible to segment a dynamic scene into moving and stationary objects. What is more interesting, is that by using complex logarithmic mapping (CLM) about the FOE some interesting properties may be obtained in the EMP space. Let us define

$$z = x + iy$$

$$w = u + iv$$

where

$$w = log(z)$$

Thus

$$u = log(r) \qquad v = \vartheta = Tan^{-1}\frac{y}{x}$$

Using this transformation it can be shown that

$$\frac{du}{dZ} = -\frac{1}{Z}$$

and

$$\frac{dv}{dZ} = 0$$

The above result states that if the observer is moving then for a stationary point the horizontal displacement depends only on its depth and the vertical component is zero. This fact appears to be very useful in not only the segmentation of a dynamic scene into moving and stationary components, but also in the determination of the depth of points using the known motion of the observer. Schwartz [Sch80a, Sch80b, Sch81, Sch82] and Cavanaugh [Cav78, Cav81] have studied this mapping in the context of biological vision systems. Schwartz has found that the retino-striate mapping can be approximated using a complex log function. This mapping is responsible for size, rotation and projection invariance in these systems. Cavanaugh argues that the mapping is justified only in limited cases. Jain [Jai83c] showed that some of the limitations, with respect to the projection invariance, may be removed if the mapping is obtained with respect to the FOE. The complex EMP space allows an observer-centered representation of the sequence by considering the ego-motion of the observer. This representation may play an important role in MCMO dynamic scenes.

6.2. Motion Understanding

The motion exhibited over a sequence of frames may be described using motion verbs. To describe the motion of an object in a frame sequence using a motion verb requires abstraction and recognition of motion concepts. Badler[Bad75] and Tsotsos[Tso77, Tso80] have developed methods for the representation of motion verbs and for the recognition of the motion in terms of the predefined verbs.

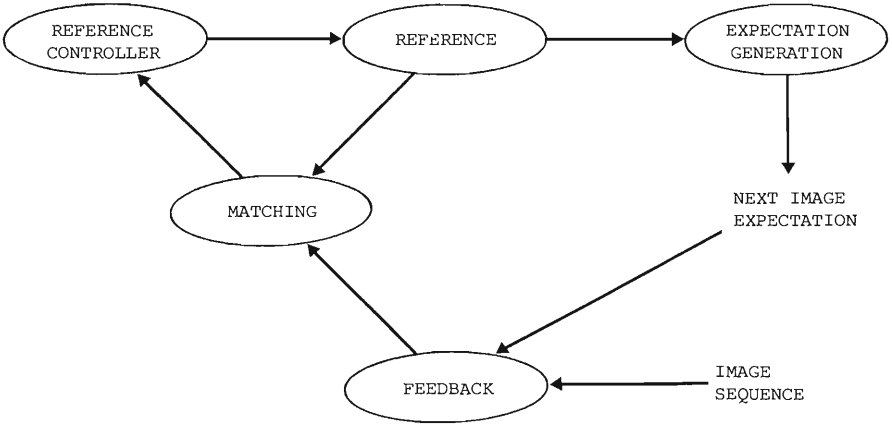
Badler [Bad75] was the first attempt to describe the events taking place in a frame sequence using natural language. He outlined a methodology for describing the events using motion verbs, adverbs, and directionals. The motion concepts are defined using lower level location, orientation, and spatial relation changes. It is assumed that lower level processes will be able to give such information and then the system will work mainly in bottom-up mode for recognizing the motion events taking place in the sequence. He developed a methodology for the representation of the motion concepts and for the recognition of the events in terms of the motion concepts from the low level data. No effort was made to consider a real vision system and hence synthetic data was used as the input to the system.

Tsotsos [Tso80] suggests the use of feedback and knowledge for motion understanding. He developed a motion understanding system for the left ventricular wall motion. By using feedback at different levels it is expected to overcome the problems due to poor quality images and the inherent ambiguity in the recognition of motion concepts based on low level processes. His scheme for the knowledge based analysis of a

frame sequence using feedback is shown in Figure 14. In this system, the objects in the first frame are identified and classified using static scene analysis techniques or manually. The analysis of motion of objects in the later frames is guided by expectations. Based on the previous motion of the objects, a new location in the next frame is predicted and verified. The changes detected using this approach are considered to be due to motion and are represented using low level vision constructs, such as axes, area, and arclength. Note that if a new object enters in the field of view in a later frame, the system will not be able to analyze its motion.

Figure 14

The knowledge based feedback model for the analysis of dynamic scenes as proposed by Tsotso in [Tso80].



The low level description of the motion is called the *essential trace* of the object. The next higher level of the representation is called the *essential kineses* and is obtained by combining pairs of adjacent essential traces. Four types of essential kineses are:

1. location changes,

2. length changes,

3. area changes,

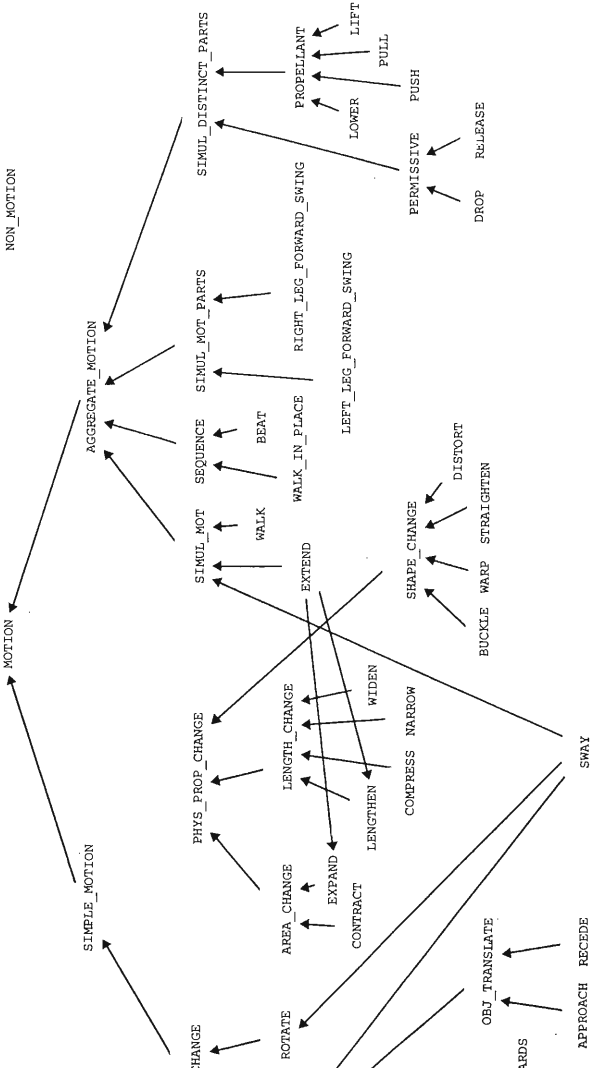


Figure 15

For the representation of motion using natural language, one requires methods to represent hierarchical motion concepts. The hierarchy proposed by Tsotso [Tso80] represents higher concepts in terms of lower concepts using an *isa* hierarchy.

4. shape changes.

These primitive kineses are used as a representation for relating quantitative changes to qualitative ones and are also used to match against the hypotheses which are active for a particular object's motion.

The objects and the concepts in this system are represented using frames. The motion concepts are organized hierarchically and are shown in Figure 15. This hierarchy is an extension of work done by Badler [Bad75]. The details of the representation of objects and other concepts used by Tsotsos are beyond the scope of this paper. It should be pointed out here that Tsotsos considered more aspects of a motion understanding system than considered by any other system. This system starts at the image level and uses *markers* to simplify the task of low level processing and analyzes the left ventricular wall motion and finally describes the motion in high level concepts related to the domain. A project for the natural language description of traffic scenes is described in [Neu82]. This project extends the methodologies developed by Tsotsos.

7. Discussion

The last few years have seen significant advances in the field of dynamic scene analysis. The early systems were strongly influenced by static scene analysis; most researchers now try to exploit the extra dimension offered by a frame sequence. Better techniques are available for change detection at pixel (or super pixel) level, a good understanding of methods for the extraction of interesting points, such as corners, has been achieved. Methods for the segmentation of dynamic scenes of all types perform reasonably well, though they are still not of the level required by higher level processes. Recovery of 3-D information has received significant attention leading to a good knowledge about the theoretical aspects, their performance in real scenes requires better location of features. Methods for the determination of the FOE for the recovery of information from optical flow show promise, at least in limited domains. The problem of how to represent motion is receiving increasing attention, both at the low-level and at the level of the description of motion concepts using motion verbs.

In the quest for solving problems in different phases of dynamic scene analysis, the techniques used are being influenced by different fields: such as differential geometry, natural language understanding, psychology, neuro-physiology, and photogrammetry. Many mathematical techniques are being applied to different aspects of the analysis leading to, generally, a better understanding of the problem and hence the development of better tools for the extraction of the information.

In many cases the assumptions made are too severe and result in techniques having applicability for only a very constrained domain. In some cases the problems arise due to the assumption that the input to the process, from a lower level process, will be perfect; and unfortunately the lower level process produces output, if at all, that is far from acceptable. A good example of this is the recovery of information from optical flow. Feature based recovery techniques are also very sensitive to the location of the tokens and fail to give acceptable results for the real images. By studying the sensitivity of the proposed approaches one could make stronger statements about their applicability.

We feel that the principle of least commitment and opportunistic procrastination [JaH82] have not been exploited to the desirable extent. Bnowing that the output of a process is going to be inherently imperfect, one should refrain from coming to faulty conclusions. Moreover, dynamic scenes allow the freedom to procrastinate, but not to sleep, until an appropriate time comes. As suggested in [JaH82] we may consider a dynamic scene analysis system in three phases and assume that the outputs of various processes may not be perfect. We should study the interaction of various processes and develop techniques that will be more tolerant to the noise coming from other processes. It appears that the paradigm of distributed problem solving [ErL75, MSC82] may help the integration of the imprecise information obtained from different sources.

Acknowledgment

The presentation and clarity of this paper was significantly enhanced by the comments and suggestions of Susan Haynes and Nancy O'Brien. I am thankful to them.

REFERENCES

[AgD75]
Aggarwal, J.K. and R.O. Duda, "Computer analysis of moving polygonal images," *IEEE Trans. on Computers*, vol. C-24, No.10, Oct. 1975, 966-976.

[AgJ82]
Agrawal, D.P. and Jain, R., "A multiprocessor system for dynamic scene analysis" *IEEE Trans. on Computer*, vol. C-31, pp.952-962, Oct. 1982.

[ALR75]
Arking,A.A., R.C. Lo, and A. Rosenfeld, "An evaluation of Fourier transform techniques for cloud motion estimation," TR-351, Dept. of Computer Science, University of Maryland, Jan. 1975.

[AYT81]
Asada, M., M. Yachida, and S. Tsuji, "Understanding of three-dimensional motions in block world", *Tech. Report no. 81-01*, Dept. Control Engin., Osaka University, 1981.

[AYT82]
Asada, M., M. Yachida, and S. Tsuji, "Representation of motions in time-varying imagery" *Proc. ICPR 82*, pp.306-309, 1982.

[BaB82]
Ballard, D.H. and C. M. Brown, *Computer Vision*, Prentice Hall, 1982.

[Bad75]
Badler, N.I., *Temporal Scene Analysis: Conceptual descriptions of object movements*, TR-80, Dept. of Computer Science, University of Toronto, 1975.

[BaT79]
Barnard, S.T. and W.B. Thompson, "Disparity analysis of images," *IEEE Trans. on PAMI*, vol. PAMI-2, 1980, pp. 333-340.

[Bor82]
Bornaee, Z., "Computation of optical flow", *M.S. Thesis*, Dept. of Computer Science, Wayne State University, 1982.

[CaR76]
Cafforio, C. and F. Rocca, "Methods for measuring small displacements of television images," *IEEE Trans. on Info. Theory*, vol. IT-22, pp. 573-579, Sept. 1976.

[Cav78]
Cavanagh, P., "Size and position invariance in the visual system," *Perception*, vol. 7, pp.167-177, 1978.

[Cav81]
Cavanagh, P., "Size invariance: reply to Schwartz," *Perception*, col. 10, pp.469-474, 1981.

[ChJ75]
Chien, R.T. and V.C. Jones, "Acquisition of moving objects and hand eye coordination," *Proc. 4th IJCAI, 1975, pp. 737-741.*

[ChW77]
Chien, Y.T. and M.O. Ward, "Representation and detection of position and shape change in time varying images," *Proc. IEEE Workshop on Picture Data Description and Management*, April 1977, pp.220-225.

[Clo80]
Clocksin, W.F., "Perception of surface slant and edge labels from optical flow: A computational approach," *Perception*, vol. 9, 1980, pp.253-269.

[DrN81]
Dreschler, L. and H.-H. Nagel, "Volumetric model and 3-D trajectory of a moving car derived from monocular TV-frame sequences of a street scene," *Proceedings of IJCAI*, 1981, pp.692-697.

[ErL75]
Erman, L and V. Lesser, "A multilevel organization for problem solving using many diverse cooperating sources of knowledge," *Proc. 4th IJCAI*, 1975, pp. 483-490.

[FeT79]
Fennema, C. L. and W. B. Thompson, "Velocity determination in scenes containing several moving objects," *Computer Graphics and Image Proc.*, vol 9, pp. 301-315, Apr. 1979.

[GrJ83]
Grosky, W.I. and R. Jain, "Region matching in pyramids for dynamic scene analysis", *in Multi-resolution image processing*, Ed. A. Rosenfeld, 1983.

[Gib66]
Gibson, J. J., *The senses considered as perceptual systems*, Boston: Houghton Mifflin, 1966.

[Gib79]
Gibson, J.J., *The ecological approach to visual perception*, Houghton Mifflen, Boston, 1979.

[GGF80]
Gilbert, A.L., M.K. Giles, G.M. Flachs, R.B. Rogers and Y. Hsun," "A real-time video tracking system using image processing," *IEEE Trans PAMI, vol. PAMI-2, 1980, pp.47-56.*

[Gla81]
Glazer, F., "Computing optical flow," *Proceedings IJCAI-81*, 1981, pp. 644-647.

[HiJ83]
Hill, Richard G. and R. Jain, "On determining optical flow", *Technical Report, Department of Computer Science, Wayne State University*, 1983.

[Hay82]
Haynes, Susan, "Detection of Moving Edges", *M.S. Thesis*, Dept. of Computer Science, Wayne State University, Detroit, 1982.

[HaJ82]
Haynes, S. and R. Jain, "Time varying edge detection", *Proc. ICPR*, Munich, pp. 754-756, 1982.

[HaJ83]
Haynes, S. and Jain, R., "Time varying edge detector," *Computer Graphics and IMage Processing*, (In Press) 1982.

[Hil82]
Hildreth, E.C., "The integration of motion information along contours," *Proc. of Computer Vision Workshop*, 1982, pp.83-91.

[HLS80]
Hirzinger, G., Landzettel and W.E. Snyder, "Automated TV tracking of moving objects," *Proc. IJCPR 1980, pp.1255-1261.*

[HNR82]
Hsu, Y.Z., H.-H. Nagel, and G. Rekers, "New likelihood test methods for change detection in image sequences", IFI-HH-M 104/82, University of Hamburg, Hambureg, West Germany, Nov. 1982.

[HoS81]
Horn, B. K. P. and B. G. Schunck, "Determining optical flow," *Proc. DARPA Image*

Understanding Workshop, pp. 144-156, Apr. 1981.

[Jai81a]
Jain, R., "Extraction of motion information from peripheral processes," *IEEE Trans on PAMI*, vol. PAMI-3, 1981, pp. 489-503.

[Jai81b]
Jain, R., "Dynamic scene analysis using pixel based processes," *IEEE Computer*, Aug 1981, pp.12-18.

[Jai82]
Jain, R., "Segmentation of moving observer frame sequences," *Pattern Recognition Letters*, vol. 1, pp. 115-120, 1982.

[Jai83a]
Jain, R., "Direct computation of the focus of expansion," *IEEE Trans. on PAMI*, vol. PAMI-5, pp.58-64, 1983.

[Jai83b]
Jain, R. "Segmentation of frame sequences obtained by a moving observer," *General Motors Research Publication*, Nov. 1982.

[Jai83c]
Jain, R., "Complex Logarithmic Mapping and the Focus of Expansion", *Proc. of Workshop on Motion: Representation and Control*, April 4-6, Toronto, 1983.

[Jai83d]
Jain, R. "On difference and accumulative difference pictures in dynamic scene analysis", *GMR publication*, 1983.

[JaJ82]
Jayaramamurthy, S. N. and R. Jain, "An approach for segmentation of textured dynamic scenes", *Proc. ICPR*, Munich, pp. 925-930, 1982.

[JaH82]
Jain, R. and S. Haynes, "Imprecision in computer vision," *IEEE Computer*, Aug 1982.

[JaJ83]
Jayaramamurthy, S.N. and Jain, R., "Segmentation of textured dynamic scenes" *Computer Graphics and Image Processing*, (in press) 1982.

[JaN79]
Jain, R., and H. H. Nagel, "On the analysis of accumulative difference pictures from image sequences of real world scenes," *IEEE Trans. PAMI*, vol. PAMI-1, pp. 206-214, Apr. 1979.

[JMA79]
Jain, R., W. N. Martin, and J. K Aggarwal, "Segmentation through the detection of changes due to motion," *Computer Graphics and Image Proc.*, vol. 11, pp. 13-34, Sept. 1979.

[JMN77]
Jain, R., D. Militzer, and H.-H. Nagel, "Separation of stationary from nonstationary components of a scene," *Proc. IJCAI*, 1977, pp.612-618.

[JeJ83a]
Jerian, C. P. and R. Jain, "Determining motion parameters for scenes with translation and rotation", *Proc. Workshop on Motion: Representation and Control*, April 4-6, Toronto, 1983.

[JeJ83b]
Jerian, C. P. and R. Jain, "On recovering 3-D information in dynamic scenes", *Technical report*, 1983.

[Joh76]
Johansson, G., "Spatio-temporal differentiation and integration in visual motion

perception", *Psych. Research*, vol. 38, pp.379-383, 1976.

[KoK80]
Korn, A. and R. Korris, "Motion analysis in natural scenes picked up by a moving sensor," *Proc. IJCPR, 1980, pp. 1251-1254.*

[Law81]
Lawton, D.T., "Optic flow field structure and processing image motion," *Proceedings IJCAI-81*, 1981, pp. 700-704.

[Law82]
Lawton, D. T., "Motion analysis via local translational processes," *Proc. Computer Vision Workshop, 1982, pp. 59-72.*

[Lee80]
Lee, D.N., "The optic flow field: The foundation of vision," *Phil. Trans. Royal Society of London*, vol. B290, 1980, pp. 169-179.

[LeG82]
Lenz, R and A. Gerhard, "Image sequence coding using scene analysis and spatio-temporal interpolation", *Proc. of NATO advanced studies institute on Image Sequence Processing* and Dynamic Scene Analysis, 1982.

[LeY82]
Legters, G.R. and T.Y.Young, "A mathematical model for computer image tracking", *IEEE Trans. on PAMI*, vol. PAMI-4, pp.583-594, 1982.

[LiM75]
Limb, J. O., and J. A. Murphy, "Estimating the velocity of moving images in television signals," *Computer Graphics and Image Proc.*, vol. 4, pp. 311-327, Dec. 1975.

[LNY82]
Levine, M.D., P. B. Noble, and Y.M. Youssef, "Understanding the dynamic behavior of a moving cell", *Proc. ICPR 82*, pp. 316-319, 1982.

[LoP80]
Longuet-Higgins, C. and K. Prazdny, "The interpretation of a moving retinal image", *Proc. Royal Society London*, vol. B-208, pp.385-397, 1980.

[MaA78]
Martin, W. N. and J. K. Aggarwal, "Dynamic Scene Analysis", *Computer Graphics and Image Processing*, vol. 7, pp. 356-374, 1978.

[MaA79]
Martin, W.N. and J.K. Aggarwal, "Computer analysis of dynamic scenes containing curvilinear objects," *Pattern Recognition*, vol. 11, 1979, pp.169-179.

[MaU79]
Marr, D. and S. Ullman, "Directional selectivity and its use in early visual processing," *MIT AI-Memo 524*, June 1979.

[Mor81]
Moravec, H.P., "Rover visual obstacle avoidance", *Proc. IJCAI 81*, pp. 785-790, 1981.

[MSC82]
McArthur, D., R. Steeb and S. Cammarata, "A framework for distributed problem solving," *Proc. AAAI*, 1982, pp.181-184.

[Nag78a]
Nagel, H.-H., "Formation of an image concept by analysis of systematic variations in the optically perceptible environment," *Computer Graphics and Image Processing*, vol 7, 1978, pp. 149-194.

[Nag78b]
Nagel, H.-H., "Analysis techniques for image sequences", *Proc. IJCPR*, pp. 186-211, 1978.

[Nag82a]
Nagel, H.-H., "On change detection and displacement vector estimation in image sequences," *Pattern Recognition Letters*, vol.1, pp.55-59, 1982.

[Nag82b]
Nagel, H.-H., "Displacement vectors derived from second order intensity variations in image sequences," Report IFI-HH-M-97/82, Fachbereich Informatik, Universitaet Hamburg, 1982.

[Nag82c]
Nagel, H.-H., "Overview on image sequence analysis", *Mitteilung No. 100*, Fachbereich Informatik, Universitaet Hamburg, Aug. 1982.

[NaE82]
Nagel, H.-H. and W. Enkelmann, "Investigation of second order greyvalue variations to estimate corner point displacements", *Proc. ICPR*, pp. 768-773, 1983.

[NaN81]
Nagel, H.-H. and B. Neumann, "On the derivation of 3-D rigid point configuration from image sequences," *Proc. IJCAI, 1981.*

[NaR82]
Nagel, H.-H. and G. Rekers, "Moving object masks based on an improved likelihood test", *Proc. ICPR*, pp. 1140-1142, 1982.

[Neu82]
Neumann, B., "Towards natural language description of real-world image sequences", *TR. No. Ifi-HH-M-101/82* Fachbereich Informatik, Universitat Hamburg, Nov. 1982.

[Neu80]
Neumann, B., "Exploiting image formation knowledge for motion analysis," *IEEE Trans. PAMI*, PAMI-2, 1980, pp. 550-554.

[NeR79]
Netravali, A. N. and J. D. Robbins, "Motion compensated television coding: part 1," *Bell Sys. Tech. Journal*, vol. 58, pp. 631-670, March 1979.

[NeR80]
Netravali, A. N. and J. D. Robbins, "Motion-compensated coding: Some new results", *BSTJ*, vol. 59, pp. 1735-1745, 1980.

[NeS79]
Netravali, A. N. and J. A. Stuller, "Motion compensated transform coding", *PRIP 79*, pp. 561-567, 1979.

[OHO73]
Onoe, R. M., N. Hammano, and K. Ohba, "Computer analysis of traffic flow observed by subtractive tevevision", *Computer Graphics and Image Processing*, pp. 377-399, 1973.

[Pot74]
Potter, J.L., "Motion as a cue to segmentation," *Milwaukee symposium on Automatic Control*, 1974, pp.100-104.

[Pra79]
Prager, J.M., "Segmentation of static and dynamic scenes," *COINS TR-79-7*, University of Massachusetts at Amherst, May 1979.

[Pra80]
Prazdny, K., "Egomotion and relative depth map from optical flow," *Biological Cybernetics*, vol. 36, 1980, pp. 87-102.

[Pra81]
Prazdny, K., "A simple method for recovering relative depth map in the case of a translating sensor", *Proc. IJCAI 81*, pp. 698-699,1981.

[Pra82]
Prazdny, K., "Computing motions of planar surfaces from spatio-temporal changes in image brightness," *Proc. IEEE Conf. Pattern Recog. and Image Processing*, June 14-17, 1982, Las Vegas, Nevada, pp. 256-258.

[PrR77]
Price, K. and R. Reddy, "Change detection and analysis in multispectral analysis", *Proc. IJCAI*, Cambridge, pp.619-625, 1977.

[ReJ83]
Rheaume, D. P. and R. Jain, "A visual tracking system", *General Motors Research Publication*, Feb. 1983.

[RoA79]
Roach, J.W. and J.K. Aggarwal, "Computer tracking of objects moving in space," *IEEE Trans PAMI*, vol PAMI-1, No.2, April 1979, pp. 127-135.

[RoA80]
Roach, J.W. and J.K. Aggarwal, "Determining the movement of objects from a sequence of images," *IEEE Trans. on PAMI*, vol. PAMI-2, No.6, Nov. 1980, pp. 554-562.

[Sch79]
Schalkoff, R. J., "Algorithms for a real-time automatic video tracking system," PhD dissertation, Univ. of Virginia, Charlottesville, Va., May 1979.

[ScM81]
Schalkoff, R. J., and E. S. McVey, "A model and tracking algorithm for a class of video targets," *IEEE Trans. PAMI*, vol. PAMI-4, pp. 2-10.

[SCH82]
Schalkoff, R. J., "Dynamic imagery via distributed parameters systems: Characteristics, discontinuities, weak solution and shocks", *PRIP 82*, pp. 119-125, 1982.

[Sch80a]
Schwartz, E. L., "Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to coding," *Vision Research*, 20, 1980, pp.645-669.

[Sch80b]
Schwartz, E.L., "A quantitative model of the functional architecture of human striate cortex with application to visual illusion and cortical texture analysis", *Biological Cybernetics*, vol.37, pp.63-76, 1980.

[Sch81]
Schwartz, E. L., "Cortical anatomy, size invariance, and spatial frequency analysis," *Perception*, vol.10, pp.455-468, 1981.

[Sch82]
Schwartz, E.L., "Columnar architecture and computational anatomy in primate visual cortex: Segmentation and feature extraction via spatial frequency coded difference mapping," *Biological Cybernetics*, vol. 42, pp.157-168, 1982.

[ScH81]
Schunck, B.G. and B.K.P. Horn, "Constraints on Optical flow computations," *Proc. IEEE Conf. on PRIP*, 1981, pp.205-210.

[SeJ83]
Sethi, I. K. and R. Jain, "Determining three dimensional structure of rotating objects using a sequence of monocular views", *Technical Report, Dept. of Computer Science, Wayne State University, Detroit*, 1983.

[SNR80]
Stuller, J. A., A. N. Netravali, and J. D. Robbins, "Interframe television coding using gain and displacement compensation", *BSTJ*, vol. 59, pp.1227-1240, 1980.

[ThB81]
Thompson, W. B. and S. T. Barnard, "Lower-level estimation and interpretation of visual motion," *Computer*, vol. 14, pp. 20-28. Aug. 1981.

[Tho80]
Thompson, W. B., "Combining contrast and motion for segmentation", *IEEE Trans. on PAMI*, PAMI-2, 1980.

[TMB82]
Thompson, W.B., K.M. Mutch, and V. Berzins, "Edge detection in optical flow fields," *Proc. AAAI*, 1982, pp.

[Tod82]
Todd, J. T., "Visual information about rigid and nonrigid motion: A geometric analysis", *J. Experimental Psychology: Human Perception and Performance*, vol. *, pp.238-252, 1982.

[TsH81]
Tsai, R. Y. and T. S. Huang, "Estimaing three-dimensional motion parameters of a rigid planar patch", *Proc of PRIP*, pp. 94-97, 1981.

[TsH82]
Tsai, R. Y. and T. S. Huang, "Uniqueness and estimation of three-dimensional motion parameters or rigid objects with curved surfaces," *Proc. IEEE Conf. Pattern Recog. and Image Processing*, 1982, pp. 112-118.

[THZ82]
Tsai, R. Y., T. S. Huang, and W.L. Zhu, "Estimating three-dimensional motion parameters of a rigid planar patch, II: Singular value decomposition", *IEEE Trans. Acoustics, Speech and SIgnal Processing*, vol. ASSP-30, pp.525-534.

[TSR82]
Tang, I, W. E. Snyder, and S. A. Rajala, "Extraction of moving objects in dynamic scenes", *Proc. ICPR*, Munich, pp. 1143-1146, 1982.

[Tso77]
Tsotsos, J.K., "Some notes on motion understanding," *Proc. of IJCAI*, 1977, p611.

[Tso80]
Tsotsos, J.K., *A framework for visual motion understanding*, TR114, Dept. of Computer Science, University of Toronto, 1980.

[TsY79]
Tsuji, S., M. Osada, and M. Yachida, "Tracking and segmentation of moving objects in dynamic line images," *IEEE Trans. on PAMI*, PAMI-2, 1980, pp.516-522.

[WaC80]
Ward, M.O. and Y.T. Chien, "Analysis of time-varying imagery through the representation of position and shape changes," *Proc. IJCPR*, 1980, pp.1236-1238.

[WeA82]
Webb, J.A. and J.K.Aggarwal, "Structure from motion of rigid and jointed objects", *Artificial Intelligence*, vol. 19, pp.107-130, 1982.

[Web81]
Webb, J., *Shape and structure from motion of objects*, Ph. D. Thesis, University of Texas, Dec. 1981.

[Wil80]
Williams, T.D., "Depth from motion in real world scenes," *IEEE Trans. PAMI*, PAMI-2, 1980,pp.511-516.

[Ull79]
Ullman, S., *The interpretation of visual motion* Cambridge, Mass, MIT Press, 1979

[Yac81]
Yachida, M., "Determining velocity map by 3-D iterative estimation," *Proceedings IJCAI-81* 1081, pp.716-718.

[YAT78]
Yachida, M., Asada, M., and S. Tsuji, "Automatic motion analysis system of moving objects from the records of natural processes", *Proc. IJCPR*, Kyoto, pp.726-730, 1978.

[YMA80]
Yalamanchili, S., W.N. Martin, and J. K. Aggarwal, "Differencing operations for the segmentation of moving objects in dynamic scenes", *Proc. of 5 ICPR*, pp. 1239-1242, 1980.

[ZuH81]
Zucker, S.W. and R.A. Hummel, "A three-dimensional edge detector," *IEEE Trans. PAMI*, PAMI-3, 1981, pp. 324-330.

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

The Estimation of the Bayes Error by the k-Nearest Neighbor Approach

Keinosuke Fukunaga
School of Electrical Engineering
Purdue University
Lafayette, Indiana 47907

I. INTRODUCTION

In pattern recognition, the estimation of the Bayes error (the probability of error due to the Bayes classification rule) plays a very important role. In order to see it, let us present a process of designing a pattern recognition system as in Fig. 1.

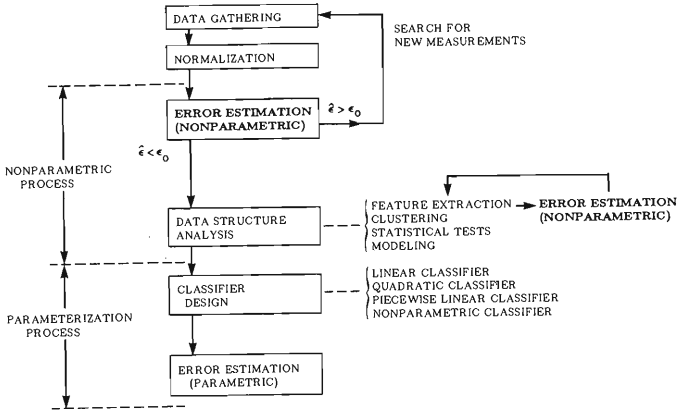


Fig. 1. Process of Designing a Pattern Recognition System

The figure shows the flow of the operations. First, the data is gathered and properly normalized. Then, the Bayes error among different distributions is estimated. This error represents the amount of overlap in the current measurement space, and the selection of features and a classifier in the later stages always results in increased error. Therefore, if the error is unacceptably high at this stage, there is no use in processing the data further. We must return to DATA GATHERING and find alternative measurements in order to obtain an acceptable performance. Also, the estimated Bayes error could serve as a reference for future operations. For example, when features are extracted, the Bayes error in the feature space must be estimated and compared with the Bayes error in the original space in order to determine whether the extracted features are acceptable. The same is true for the error of the final classifier.

Once the estimated Bayes error is acceptable, we can move to DATA STRUCTURE ANALYSIS which includes many operations such as feature extraction, clustering, statistical tests and modeling. Based on these studies, we may choose a proper classifier for the given data set. The final stage is the evaluation of the classifier.

The estimation of the Bayes error appears in Fig. 1 twice, as is shown by bald faces. One is the estimation in the original measurement space to determine whether these measure-

This work was supported in part by the National Science Foundation under Grant ECS-80-05482.

ments guarantee an acceptable performance. The other is in the feature space after the features are selected. The Bayes error in the feature space, in comparison with the Bayes error in the original measurement space, gives the evaluation of the selected features. Both are crucial operations. In addition, both require a nonparametric procedure, since any assumption of the mathematical forms for the distributions of the data is not appropriate at these stages.

There are two nonparametric approaches to the Bayes error estimation. One is the k-nearest neighbor (k-NN) approach which is based on the k-NN density estimate [FI] [LO] [CO1] [CO2]. The other is the one based on the Parzen density estimate [PA] [CA]. There are some differences between them. But, in a simplified term, the k-NN density estimate may be considered as the Parzen one with a uniform kernel whose volume varies locally, depending on the locations of nearest neighbors. Therefore, if one can do something by the Parzen approach, the same performance can be achieved by the k-NN approach with a proper modification. Because of this, this paper discusses only the k-NN approach. Similar discussions could be developed for the Parzen approach.

Besides the Bayes error estimation, the k-NN approach has many other applications in pattern recognition. The most obvious one is to use it for classification of an unknown sample, and many works have been done to make the procedure more efficient [HA] [FU10] [FR] [YU], to make the error smaller [WI] and so on. An estimation problem is formulated based on the k-NN approach [CO3] [ST]. Also, the k-NN approach provides an excellent tool to analyze the data structure in a nonparametric way [KO1] [KO2] [FU9]. The density estimation is another application [FU5]. Also, there are many mathematical problems which are related to the convergence of the estimator to the asymptotic value [CO2] [WA]. These subjects need a survey paper, but are outside the scope of this paper. This paper deals only with the Bayes error estimation.

Finally it should be pointed out that this is not a survey paper but an assembly of the author's past work on the subject. There are many other papers in the area, but they are not mentioned in this paper, only because they are not directly related to the discussion of the paper.

II. k-NN Error Bounds

The k-NN classification rule is theoretically based on the Bayes decision rule, combined with the density estimation of the distributions by the locations of the nearest neighbors.

Let us look at the problem in which a density function $p(X)$ is estimated locally at a point X which is an n-dimensional vector. There are two ways to do this as follows:

(1) Set a local fixed region $\Gamma(X)$ around X, and count the number of the neighboring samples of X. Then, the density is estimated by

$$\hat{p}(X) = \frac{\mathbf{k}}{N\mathbf{v}} \tag{1}$$

where $\mathbf{k}$ is the number of samples in $\Gamma(X)$, v is the volume of $\Gamma(X)$, and N is the total number of samples. $\mathbf{k}$ is a random variable and so is $\hat{p}(X)$. Throughout the paper, bold faces will be used to indicate random variables. The density function of $\mathbf{k}$ is known as

$$p_{\mathbf{k}}(k) = \binom{N}{k} u^k (1-u)^{N-k} \tag{2}$$

where u is the probability of a sample falling in the $\Gamma(X)$; that is, $u = \int_{\Gamma(X)} p(X)dX$. It can be easily shown that $\hat{p}(X)$ of (1) is an asymptotically unbiased and consistent estimate of $p(X)$ if

$$\lim_{N\to\infty} k = \infty \quad \text{and} \quad \lim_{N\to\infty} k/N = 0 \tag{3}$$

are satisfied [PA].

(2) The other way of estimating a density is to fix the number of the neighbors k and extend $\Gamma(X)$ until the k-NN is found. The $\mathbf{v}$ for the $\Gamma(X)$ becomes a random variable. Then

$$\hat{p}(X) = \frac{k-1}{N\mathbf{v}} \tag{4}$$

gives an asymptotically unbiased and consistent estimate of p(X) under the conditions of (3) [LO]. This time, the density function of $\mathbf{u}$ is known as

$$p_{\mathbf{u}}(u) = \frac{N!}{(k-1)!(N-k)!}\, u^{k-1}(1-u)^{N-k} \tag{5}$$

The Bayes decision rule is stated as

$$P_1 p_1(X) \gtrless P_2 p_2(X) \quad \rightarrow X\epsilon \begin{cases} \omega_1 \\ \omega_2 \end{cases} \tag{6}$$

where P_i and $p_i(X)$ are a priori probability and the density function of class i (ω_i) respectively. Counting the numbers of neighboring samples from ω_1 and ω_2 as $\mathbf{k}_1$ and $\mathbf{k}_2$ in a fixed $\Gamma(X)$, (6) may be replaced by

$$\frac{N_1}{N}\frac{\mathbf{k}_1}{N_1 v} \gtrless \frac{N_2}{N}\frac{\mathbf{k}_2}{N_2 v} \tag{7}$$

where N_i is the number of class i samples and P_i is estimated by N_i/N. Or, eliminating all common terms,

$$\mathbf{k}_1 \gtrless \mathbf{k}_2 \tag{8}$$

That is, the procedure calls for the classification of X according to the majority of the classes of neighboring samples.

The k-NN decision rule is very simple, and knowledge about the density functions is not required. Furthermore, the probability of error (performance) due to this procedure is very close to that of the Bayes decision rule. In order to see it, the asymptotic analysis of the NN decision rule is presented as follows [CO1].

As the simplest case, let us pick the nearest neighbor of X, $\mathbf{X}_1$, and classify X by the class of the neighbor. Then, the risk of this decision is

$$r_1(X,\mathbf{X}_1) = \eta_1(X)\eta_2(\mathbf{X}_1) + \eta_2(X)\eta_1(\mathbf{X}_1) \tag{9}$$

where $\eta_i(X)$ is the a posterior probability of ω_i given X, and $\eta_i(X) = P_i p_i(X)/p(X)$ with the mixture density $p(X) = P_1 p_1(X) + P_2 p_2(X)$. For asymptotic analysis, in which $\eta_i(\mathbf{X}_1) = \eta_i(X)$ may be assumed, (9) becomes

$$r_1(X) = 2\,\eta_1(X)\eta_2(X) \tag{10}$$

This is compared with the Bayes risk given X, $r^*(X)$, which is expressed by

$$\begin{aligned} r^*(X) &= \min\left[\eta_1(X), \eta_2(X)\right] = \frac{1}{2} - \frac{1}{2}\sqrt{1 - 4\eta_1(X)\eta_2(X)} \\ &= \sum_{i=1}^{\infty} \frac{1}{i}\binom{2i-2}{i-1}\left[\eta_1(X)\eta_2(X)\right]^i \end{aligned} \tag{11}$$

Eq. (10) can be easily extended to the k-NN case. Under the same assumption of $\eta_i(X) = \eta_i(\mathbf{X}_1) = \cdots = \eta_i(\mathbf{X}_k)$, the asymptotic risk becomes [FU4]

$$\begin{aligned} r_{2k-1}(X) &= \eta_1(X)\sum_{j=0}^{k-1}\binom{2k-1}{j}\eta_1^j(X)\eta_2^{2k-1-j}(X) + \eta_2(X)\sum_{j=0}^{k-1}\binom{2k-1}{j}\eta_2^j(X)\eta_1^{2k-1-j}(X) \\ &= \sum_{i=1}^{k}\frac{1}{i}\binom{2i-2}{i-1}\left[\eta_1(X)\eta_2(X)\right]^i + \frac{1}{2}\binom{2k}{k}\left[\eta_1(X)\eta_2(X)\right]^k \end{aligned} \tag{12}$$

Note that all $r_k(X)$'s are expressed as the functions of $\eta_1(X)\eta_2(X)$. Using $\xi(X) = \eta_1(X)\eta_2(X)$, Fig. 2 shows the relations among $r_k(X)$ as the functions of $\xi(X)$. It is seen from the figure that

$$\frac{1}{2}r_1(X) = r_2(X) < r_4(X) < \cdots < r^*(X) < \cdots < r_5(X) < r_3(X) < r_1(X) < 2r^*(X) \tag{13}$$

where r_{2k} indicates the first term of the last line of (12) for the corresponding k. These inequalities can be verified also from (11) and (12) directly. $r_{2k}(X)$ could be obtained as the risk when we choose an even number of neighbors and reject the case where the number of the ω_1 neighbors is equal to the number of the ω_2 neighbors [HE] [DE].

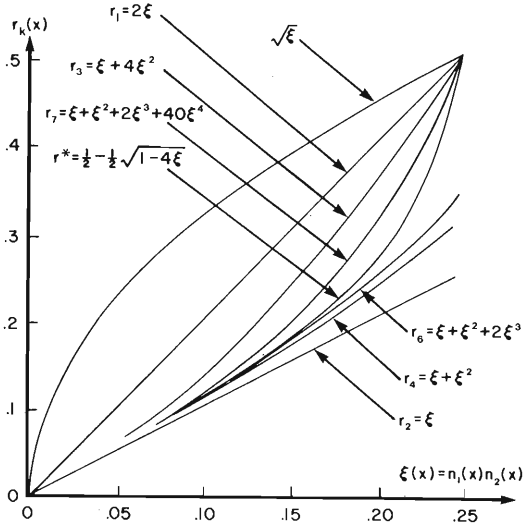


Fig. 2. Comparison of ϵ_i for Various i

The overall error can be obtained by taking the expectation of $r(\mathbf{X})$ with respect to $\mathbf{X}$ as

$$\epsilon = E\{r(\mathbf{X})\} = \int r(X)p(X)dX \tag{14}$$

Thus, the inequalities of (13) hold for the errors as

$$\frac{1}{2}\epsilon_1 = \epsilon_2 < \epsilon_4 < \cdots < \epsilon^* < \cdots < \epsilon_5 < \epsilon_3 < \epsilon_1 < 2\epsilon^* \tag{15}$$

Fig. 2 also includes $\sqrt{\xi}$ because $E\{\sqrt{\xi(\mathbf{X})}\}$, known as the Bhattacharyya number, is another upper bound of the Bayes error [FU1].

The above discussion suggests the following for the asymptotic case:

(1) The k-NN classification rule with an odd k gives the upper bound of the Bayes error.

(2) An even k with reject option gives the lower bound of the Bayes error.

(3) ϵ_2 is supposed to be equal to $1/2\,\epsilon_1$. This relation may provide us a means to check whether or not the asymptotic assumption is met.

(4) If $\xi(X) = (r_2(x)$ or $1/2\,r_1(X))$ can be estimated for every X, (11) and (14) provide a way to estimate the Bayes error. That is,

$$\hat{\epsilon}^* = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{1}{2} - \frac{1}{2}\sqrt{1 - 4\hat{\xi}(\mathbf{X}_i)}\right] \tag{16}$$

where the expectation of (14) is replaced by a sample mean over available samples $\mathbf{X}_1,\ldots,\mathbf{X}_N$.

Table 1 shows the actual operation to estimate the upper and lower bounds of the Bayes error by the k-NN decision rule. Given are the left two columns; samples $X_1, \cdots, X_N$ and their true classes. Then, we must find the neighbors of each X_i and fill out the table. The classification is made, the result is compared with the true class, and they are labeled as "Correct," "Reject," or "Error." Only the "Errors" are counted and the summation divided by N gives the estimate of ϵ.

Table 1. A Procedure for the k-NN Error Estimation

GIVEN SAMPLES		1st NN		2nd NN		3rd NN		NN		2NN		3NN	
	ω		ω		ω		ω	Class-ification	Correct or Error	Class-ification	Correct or Error	Class-ification	Correct or Error
X_1	1	X_3	1	X_{10}	1	X_{23}	2	1	Correct	1	Correct	1	Correct
X_2	2	X_{18}	1	X_{25}	2	X_{36}	2	1	Error	?	Reject	2	Correct
X_3	2	X_{53}	1	X_{32}	2	X_{65}	1	1	Error	?	Reject	1	Error
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.
X_N	1	X_{125}	2	X_{82}	2	X_{91}	2	2	Error	2	Error	2	Error

$$\hat{\epsilon}_1 = \frac{\#\text{ of Errors}}{N} \qquad \hat{\epsilon}_2 = \frac{\#\text{ of Errors}}{N} \qquad \hat{\epsilon}_3 = \frac{\#\text{ of Errors}}{N}$$

Alternatively, the bounds of the Bayes error may be estimated, based on the disjoint training and test sets as follows:

(1) the available samples are divided into two groups, the training set and the test set.

(2) Each sample from the test set is classified by the k-NN procedure with the neighbors selected only from the training set. The number of the misclassified test samples is counted to estimate the bound.

This procedure is designed to achieve independence between the training and test sets. However, the effective sample size is reduced to one half. This reduction is considered to be more harmful than the dependency of the training and test sets. In fact our experience has shown that the error bounds obtained from the divided sets with 2N samples is very close to the ones obtained by the procedure of table 1 with N samples. Therefore, the procedure of table 1 is, in general, used preferably.

Although the k-NN Bayes error estimation is very simple and does not require any knowledge of the density functions, there are many factors which produce erroneous results, mostly due to the fact that available data are finite. In order to minimize them, we must understand the performance of the k-NN estimator for a finite sample case. The following is a list of problems which most practicians face when they try to use the k-NN approach.

(1) Sample size: How do we determine whether the given sample size N is adequate for the Bayes error estimation?

(2) Optimal k: How do we determine the optimal value for k?

(3) Optimal metric: What metric must we use to measure the nearness between samples?

(4) Dimensionality and density: What effect do we have from the dimensionality and the density functions?

(5) Unbiasness and smaller variance: Does an unbiased estimator exist for the Bayes error? Is there any way to reduce the variance and bias of the estimator?

(6) k-NN and Parzen: Which is a better estimate of the Bayes error--the k-NN approach or the Parzen one?

(7) Different sample sizes for different classes: If the sample sizes are different from class probabilities, what can we do? This problem is also related to the k-NN Neyman-Pearson test and the k-NN minimax test.

In Section 3, a different type of estimate will be discussed, resulting in a smaller variance (Item 5). Section 4 shows how to handle the case where the sample sizes from ω_1 and ω_2 are not proportional to P_1 and P_2 (Item 7). Section 5 deals with the optimal metric (Item 3). So, the remainder of this section will be devoted to the sample size problem (Item 1), and the optimal k problem (Item 2).

Sample Size Problem:

It is very difficult to determine theoretically how many samples are needed to obtain the

reliable values for the upper and lower bounds of the Bayes error. But, in practice, the plots of $\hat{\epsilon}_{2k-1}$ (the upper bound) and $\hat{\epsilon}_{2k}$ (the lower bound) as the functions of N will give a reasonable feeling as to where the Bayes error is located and whether the available N is adequate.

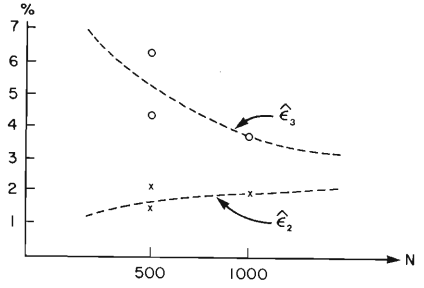


Fig. 3. Experimental Result for Hand-Written Numerals

Fig. 3 shows the plots for one extreme example. The data is selected from handwritten numerals, 0 through 9 (10 classes). A rectangular frame is set to cover each numeral most compactly. The frame is divided into 16x16 meshes and each mesh is assigned to either 0 or 1 depending on the occupancy rate of the numeral on the mesh. Thus, each numeral is expressed by a 256-dimensional binary vector. We had 1000 samples, 100 for each class. The nearness of samples is measured by the Hamming distance, the number of discrepancies in the corresponding mesh positions. $\hat{\epsilon}_3$ and $\hat{\epsilon}_2$ are estimated by using all available 1000 samples according to the procedure shown in table 1. Next, the 1000 samples are divided into two groups of 500 samples. $\hat{\epsilon}_3$ and $\hat{\epsilon}_2$ are estimated by using these 500 samples. Thus, there are two $\hat{\epsilon}_3$'s and two $\hat{\epsilon}_2$'s. Fig. 3 shows the reasonable two curves bounding the supposed-to-be-Bayes error. Also, the figure gives an impression that, even if we add another samples, say 1000 more, probably we cannot expect a much more accurate estimate of the Bayes error.

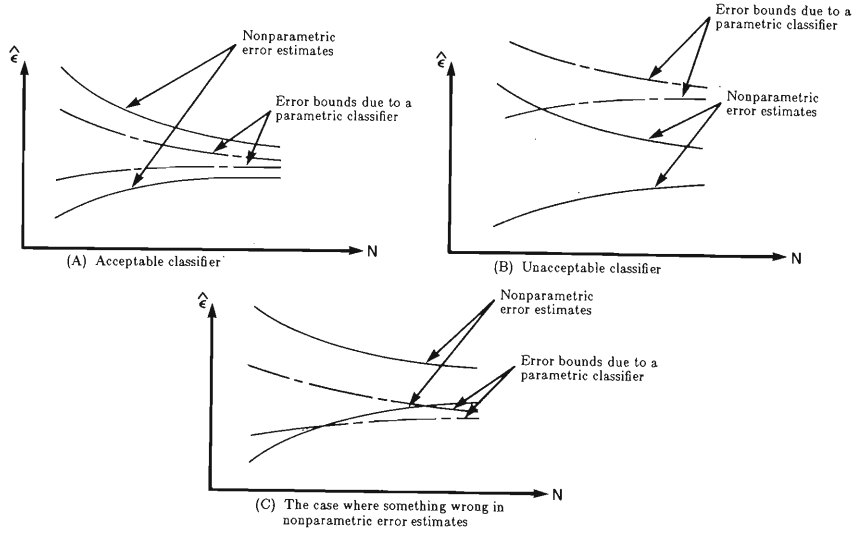


Fig. 4. Parametric and Nonparametric Error Bounds

The plots of $\hat{\epsilon}_{2k-1}$ and $\hat{\epsilon}_{2k}$ vs. N may serve later as the reference when a classifier is designed and evaluated. Fig. 4 shows the plots of both nonparametric and parametric error

estimates. The upper and lower bounds of a parametric classifier may be estimated by using the leaving-one-out and resubstitution methods respectively [FU2]. Normally, classifiers are characterized by a small number of parameters such as the means and covariances, and these parameters are estimated by all available samples. On the other hand, nonparametric estimates are based on a relatively small number of local samples. Therefore, the bias between the upper and lower bounds is smaller for the parametric case. But, the disadvantage of the parametric classifier is that the performance depends on the assumption of a parametric form for the distribution. Fig. 4 (a) shows the case of an acceptable classifier where the error of the classifier looks close to the Bayes error which is guessed from the nonparametric error estimates. Fig. 4 (b) is the case where the classifier error is much higher than the guessed Bayes error. In this case, with better knowledge of the distribution, a better classifier could be designed. Fig. 4 (c) is a case where something is wrong, because the classifier error cannot be smaller than the Bayes error. If this happens, the entire operation must be reevaluated, but the problem, most likely, concerns the non-parametric operations. Because we are dealing with the k-NN decision rule for a finite sample size, the effects of many factors such as the metric, the sample size, dimensionality and density functions are less understood.

Finally, it should be pointed out that, as Fig. 4 (a) shows, the performance of a well designed classifier is better than the one of the k-NN classifier with a smaller error and more stability.

Optimal k:

Again, theoretical treatment to find the optimal k for the Bayes error estimation is very difficult and very little is known on this subject. But, in practice, particularly for high dimensional cases, there are very limited choices left other than selecting k to be less than 5 or so, because we normally do not have enough samples to fill up the space to assume $\eta_i(X) \simeq \eta_i(\mathbf{X}_j)$ for large j's. Probably, the most practical way is to compute the k-NN errors for all possible k's up to 5 or so, and see whether these errors are related reasonably. For example, we can check whether $\hat{\epsilon}_1 \simeq 2\hat{\epsilon}_2$ or whether $\hat{\epsilon}_1 > \hat{\epsilon}_3 > \hat{\epsilon}_5$ and $\hat{\epsilon}_2 < \hat{\epsilon}_4$ are satisfied.

As a related subject, the optimal k for density estimation is known as a function of N, n and p(X) [FU5]. However, it is unknown how or whether the optimal k for density estimation is related to the optimal k for the k-NN classification.

III. Estimation of $r^*(X)$

In the previous section, we discussed the case where the number of the misclassified samples is counted to estimate the upper and lower bounds of the Bayes error. We will call this procedure the k-NN error count. In this section, we treat the Bayes error estimation in a different way.

Reduction of Variance:

Let us look at (14). If $r^*(X)$ is known as a function of X, $r^*(X)$ has two significant properties as follows [FU3] [FU4]:

(1) The Bayes error can be estimated by the sample mean of $r^*(\mathbf{X}_i)$ for N_t test samples as

$$\hat{\epsilon} = \frac{1}{N_t} \sum_{i=1}^{N_t} r^*(\mathbf{X}_i) , \tag{17}$$

where the $\mathbf{X}_i$ is the i-th sample, drawn from the mixture density p(X). Note that $\mathbf{X}_i$ meant the i-th NN of X in the previous section and that the same notation is used differently in this section. Eq. (17) suggests that, since the $\mathbf{X}_i$'s are drawn from p(X), the class assignments of the $\mathbf{X}_i$'s are not needed to compute $\hat{\epsilon}$. The estimate of (17) is unbiased as

$$E\{\hat{\epsilon}\} = \frac{1}{N_t} \sum_{i=1}^{N_t} E\{r^*(\mathbf{X}_i)\} = \epsilon^* . \tag{18}$$

(2) The variance of $r^*(\mathbf{X})$,$\sigma^2(r^*(\mathbf{X}))$, is

$$\sigma^2(r^*(\mathbf{X})) = E\{r^{*2}(\mathbf{X}))\} - \epsilon^{*2}$$
$$= \epsilon^* - \epsilon^{*2} - E\{r^*(\mathbf{X})\,(1-r^*(\mathbf{X}))\}$$
$$\leq \epsilon^*(1-\epsilon^*) - \epsilon^*/2\,. \tag{19}$$

Thus, the variance of $\hat{\epsilon}$ is given by

$$\text{Var}\,[\hat{\epsilon}] = \sigma^2(r^*(\mathbf{X}))/N_t \tag{20}$$

On the other hand, if class identification were available for the N_t test vectors and an empirical error count were made, the error count would also given an unbiased estimate $\hat{\epsilon}'$ of the Bayes error. The variance of this estimate is known to be

$$\text{Var}\,[\hat{\epsilon}'] = \epsilon^*(1-\epsilon^*)/N_t\,. \tag{21}$$

Thus, we can achieve a reduction in variance of at least $0.5\epsilon^* N_t^{-1}$ by using $\hat{\epsilon}$.

In many practical instances the probability densities $p_i(X)$ needed to specify the function $r^*(X)$ are not known. However, because of the reduction in variance and the effective use of unclassified samples, it is reasonable to search for nonparametric approximations to $r^*(X)$, thus providing improved nonparametric estimates of the Bayes error.

Estimation of $r^*(X)$:

In this section a nonparametric estimate of the risk function $r^*(X)$ will be derived by using the volume information of the given design data set.

Let us consider the design set to be grouped into $\mathbf{N}_1$ class 1 samples and $\mathbf{N}_2$ class 2 samples. We then specify that the k_1–nearest neighbors of X from class 1 and the k_2–nearest neighbors of X from class 2 be found. All the available information about $r^*(X)$ is thus embodied in $\mathbf{N}_1$, $\mathbf{N}_2$ and in the distances out to, or equivalently the volumes of the regions out to, the j_i–nearest neighbors ($j_1 = 1,2,...,k_1$ and $j_2 = 1,2,...,k_2$). We then find the joint density function of all the available information given k_1, k_2, and the total number of design samples, N_d, as follows [FU7].

$$p(\mathbf{v}_{11},...,\mathbf{v}_{1k_1},\mathbf{v}_{21},...,\mathbf{v}_{2k_2},\mathbf{N}_1,\mathbf{N}_2/N_d,k_1,k_2)$$
$$= \frac{(N_1+N_2)!}{(N_1-k_1)!(N_2-k_2)!}\,P_1^{N_1}P_2^{N_2}\,p_1(X)^{k_1}p_2(X)^{k_2}\left[1-p_1(X)v_{1k_1}\right]^{N_1-k_1}\left[1-p_2(X)v_{2k_2}\right]^{N_2-k_2}$$

$$0 < v_{i1} < \cdots < v_{ik_i} < 1/p_i(X) \quad (i=1,2) \text{ and } N_1 = 0,1,...,N_d;\quad N_2 = N_d - N_1 \tag{22}$$

where $\mathbf{v}_{ij}$ is the volume of the region out to the j-th nearest neighbor of X from class i. Eq. (22) can be derived as the product of $p_{\mathbf{u}}(u/\omega_1)$, $p_{\mathbf{u}}(u/\omega_2)$ and $p_{N_1}(N_1)$, where $p_{\mathbf{u}}(u/\omega_i)$ is given in (5) and $p_{N_1}(N_1)$ is a binomial distribution. The conversion of the density from u to v is carried out under the linearity assumption of $u = vp(X)$. Eq. (22) shows that only three random variables $\mathbf{v}_{1k_1}$, $\mathbf{v}_{2k_2}$, and $\mathbf{N}_1$ provide the complete sufficient statistics. Knowing the joint density function of these random variables, we can derive the maximum likelihood estimates of P_i, $p_i(X)$ and $r^*(X)$. The results are

$$\hat{\mathbf{P}}_i = \mathbf{N}_i/N_d \tag{23}$$

$$\hat{\mathbf{p}}_i(X) = k_i/N_i\mathbf{v}_{ik_i} \tag{24}$$

$$\hat{\mathbf{r}}_g(X) = \min\left[\frac{k_1/\mathbf{v}_{1k_1}}{k_1/\mathbf{v}_{1k_1} + k_2/\mathbf{v}_{2k_2}}\,,\ \frac{k_2/\mathbf{v}_{2k_2}}{k_1/\mathbf{v}_{1k_1} + k_2/\mathbf{v}_{2k_2}}\right] \tag{25}$$

We call this estimate of $r^*(X)$ the grouped maximum likelihood estimate.

Knowing the density function of $\mathbf{v}_{1k_1}$, $\mathbf{v}_{2k_2}$ and $\mathbf{N}_1$ as (22), it is possible to evaluate the performance of the grouped estimate $\hat{\mathbf{r}}_g(X)$ of (25) [FU7]. Figs. 5 and 6 show the expected value and variance of $\hat{\mathbf{r}}_g(X)$ as the functions of $r^*(X)$.

In another set up, we can pool together all design samples from both classes and specify that the k-NN of X from this pooled set be found. This varies from the previous approach in

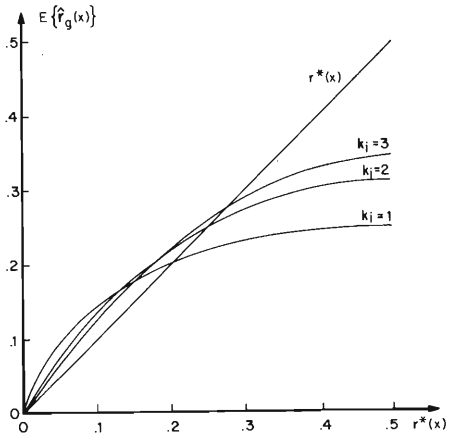


Fig. 5. Expected Value of Grouped Estimate $\hat{r}_g(X)$

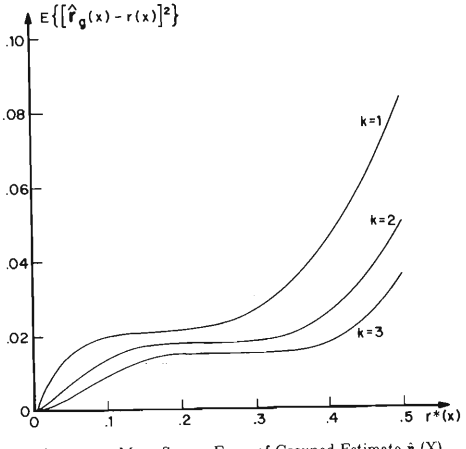


Fig. 6. Mean-Square-Error of Grouped Estimate $\hat{r}_g(X)$

that here the numbers of class 1 neighbors and class 2 neighbors found are random variables. The joint density function of those random variables can be found, as [FU7].

$$p(v_1,\cdots,v_k,\theta_1,\cdots,\theta_k,N_1,N_2/N_d,k)$$

$$= \binom{N_d}{N_1} p_1(X)^{k_1} p_2(X)^{k_2} P_1^{N_1+k_1} P_2^{N_2+k_2} \exp\left[-N_d v_k p(X)\right]$$

$$0 < v_1 < v_2 < \cdots < v_k < \infty, \text{ and } N_1 = 0, \cdots, N_d \tag{26}$$

where $\boldsymbol{\theta}_i$ is the class of the i-th NN which is a random variable this time. Eq. (26) shows that only three random variables $\mathbf{k}_1(\mathbf{k}_2 = k - \mathbf{k}_1)$, $\mathbf{v}_k$, and $\mathbf{N}_1(\mathbf{N}_2 = N_d - \mathbf{N}_1)$ provide the complete sufficient statistics. The maximum likelihood estimate of $r^*(X)$ becomes

$$\hat{\mathbf{r}}_p(X) = \min\left[\frac{\mathbf{k}_1}{k}, \frac{\mathbf{k}_2}{k}\right] \tag{27}$$

Eq. (27) is equivalent to estimating the posterior probability $\eta_i(X)$ by $\mathbf{k}_i/k$ in (11). Also, it has been pointed out that this approach gives the same performance as ϵ_k when k is even, thus it is a lower bound of the Bayes error [FU7].

The expected value and variance of $\hat{\mathbf{r}}_p(X)$ can be also calculated and they are shown in Figs. 7 and 8. The fact that the pooled estimate gives a lower bound of the Bayes error is evident in Fig. 7.

From Figs. 5-8 we see that, depending upon the use to be made of the Bayes risk estimate, one might prefer either the grouped estimate $\hat{\mathbf{r}}_g(X)$ or the pooled estimate $\hat{\mathbf{r}}_p(X)$.

If one is interested in obtaining estimates of $r^*(X)$ that are representative of its true value or in obtaining confidence interval estimates for $r^*(X)$, then one would choose the grouped estimate, since it always has the smaller mean-square-error of the two as a comparison between Figs. 6 and 8 shows.

This can be thought of as resulting from the fact that $\hat{\mathbf{r}}_g(X)$ takes on a continuum of values between zero and one-half while $\hat{\mathbf{r}}_p(X)$ takes on only discrete values. In addition to providing continuous estimates of the true Bayes risk and smaller asymptotic mean-square-error, the grouped estimate also has the advantage that for finite design set sizes, its ability to use different distance functions $d_i(X,Y)$ for each class i should result in better estimates of $r^*(X)$.

On the other hand, the pooled estimate gives the lower bound of the Bayes error. Also, if k is selected to be two, the pooled estimate gives an estimate of $\xi(X) = \eta_1(X)\eta_2(X)$, which could be used to estimate the Bayes error by using (16).

To demonstrate the performance of the pooled and grouped approaches in obtaining esti-

mates of the Bayes risk and their use in estimating the Bayes error from unclassified test samples, the following experiment was performed.

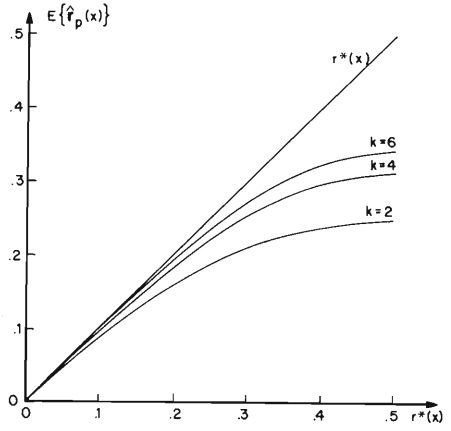


Fig. 7. Expected Value of Pooled Estimate $\hat{r}_p(X)$

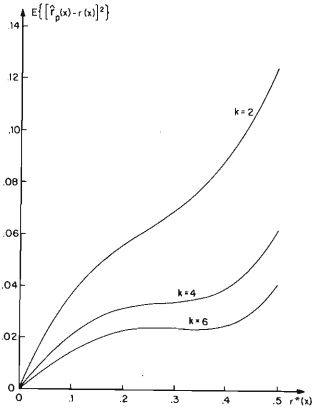


Fig. 8. Mean-Square-Error of Pooled Estimate $\hat{r}_p(X)$

Two dimensional random vectors were generated from equal covariance Gaussian distributions with each class being chosen with equal probability and their means chosen to give a Bayes error of 11%. Euclidean distance functions were used for both the grouped and the pooled estimates. For each trial $N_d = 500$ classified design samples and $N_t = 500$ unclassified test samples were generated.

To investigate the mean-square-error of the Bayes risk estimates $\hat{\mathbf{r}}_g(X)$ and $\hat{\mathbf{r}}_p(X)$, we calculated the true $r^*(X)$ at each test sample and compared it with its pooled and grouped estimate. The sample mean-square-error for a typical trial is shown in Fig. 9. This clearly demonstrates the theoretically predicted smaller mean-square-error of the grouped estimate.

To demonstrate the performance of the grouped and pooled estimates in estimating the Bayes error, we evaluated (17) using $\hat{\mathbf{r}}_g(X)$ and $\hat{\mathbf{r}}_p(X)$ averaged over the unclassified test samples. In addition, an error count estimate of ϵ^* was calculated using the k-NN decision rule. This was repeated for ten trials with the average value of $\hat{\epsilon}$ and the average square error $(\hat{\epsilon} - \epsilon^*)^2$ of $\hat{\epsilon}$ being shown in Figs. 10 and 11.

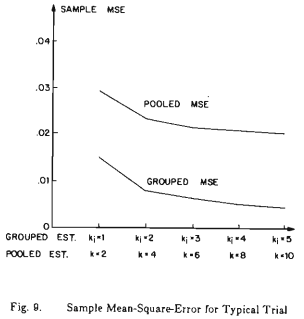


Fig. 9. Sample Mean-Square-Error for Typical Trial

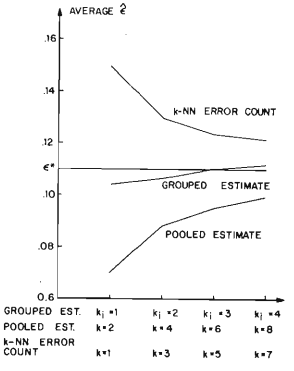


Fig. 10. Average Value of Bayes Error Estimate $\hat{\epsilon}^*$

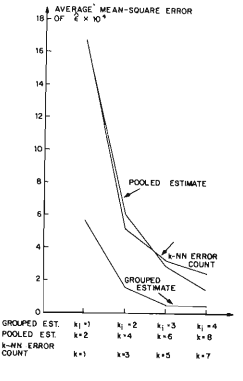


Fig. 11. Average Mean-Square-Error of Bayes Error Estimate $\hat{\epsilon}^*$

These indicate that for this experiment the grouped estimate of the Bayes error provides a good measure of the true error ϵ^*, while the pooled estimate provides a lower bound on ϵ^*,

and the error count estimate provides an upper bound on ϵ^*. This can be thought of as resulting from the fact that the expected value of the error count estimate is always greater than $r^*(X)$, the pooled estimates expected value is always below $r^*(X)$, and the grouped estimate's expected value can be either above or below the true $r^*(X)$ allowing its average over the testing set to be near the true Bayes error ϵ^*.

Finally, it should be pointed out that, if an unbiased estimate of $r^*(X)$ is obtained, ϵ^* can be estimated without bias as (17) indicates. Unfortunately, up to now there is no unbiased estimate of $r^*(X)$ known. This is due to the nonlinearity of the function $r^*(X) = \min [\eta_1(X), \eta_2(X)]$.

IV. k-NN Data Display

Another application of utilizing the volume information of the k-NN is the display of the given data set. This leads us to a handy algorithm to apply the k-NN approach to the Neyman-Pearson test and the minimax test.

Data Display and the Likelihood Ratio Classifier:

Let us look at the Bayes decision rule for minimum risk:

$$\frac{p_1(X)}{p_2(X)} \gtrless \frac{P_2(c_{21} - c_{22})}{P_1(c_{12} - c_{11})} \rightarrow X\epsilon \begin{cases} \omega_2 \\ \omega_1 \end{cases} \tag{28}$$

where the c_{ij} are the cost of deciding $X\epsilon\omega_j$ when $X\epsilon\omega_i$. The other tests such as the Neyman-Pearson test and the minimax test use the same density ratio, but different thresholds instead of $P_2(c_{21} - c_{22})/P_1(c_{12} - c_{11})$.

Let us use the density estimate

$$\hat{p}_i(X) = \frac{k}{N\mathbf{v}_i(X)} \tag{29}$$

with a fixed k and variable volume $\mathbf{v}_i(X)$. The volume is related to the Euclidean distance from X to the k-NN from ω_i, $\mathbf{d}_i(X)$, by $2\mathbf{d}_i^n(X)\pi^{n/2}/n\Gamma(n/2)$ where n is the dimensionality of X and Γ is the gamma function. Taking the logarithm of (29),

$$-\ln \hat{p}_i(X) = n \ln \mathbf{d}_i(X) + \ln \frac{2N\pi^{n/2}}{kn\Gamma(n/2)} \tag{30}$$

Compare (30) with $-\ln p_i(X)$ for a Gaussian distribution with the mean vector M_i and the covariance matrix Σ_i,

$$-\ln p_i(X) = 1/2\,(X - M_i)^T\Sigma_i^{-1}(X - M_i) + 1/2 \ln(2\pi)^n \left| \Sigma_i \right| \tag{31}$$

we realize that $n \ln \mathbf{d}_i(X)$ of (30) is the nonparametric version of the Mahalanobis distance $1/2\,(X-M_i)^T\Sigma_i^{-1}(X-M_i)$. Using (30), (28) can be rewritten as

$$n \ln \mathbf{d}_2(X) \gtrless n \ln \mathbf{d}_1(X) + \ln \frac{N_1P_2(c_{21} - c_{22})}{N_2P_1(c_{12} - c_{11})} \rightarrow X\epsilon \begin{cases} \omega_2 \\ \omega_1 \end{cases} \tag{32}$$

Thus, X can be mapped to two feature functions $y_1(X) = n \ln d_1(X)$ and $y_2(X) = n \ln d_2(X)$, and they may be plotted in this feature space.

In Fig. 12 the display is presented using classes ω_1 and ω_2 of the eight-dimensional standard data of Fukunaga [FU1]. There are 100 samples per class plotted. It was assumed $P_1 = P_2 = 0.5$, and $c_{12} - c_{11} = c_{21} - c_{22}$. For this figure $k = 1$ was used. The display indicates that the two distributions have approximately a 5% overlap, which is reasonable given that the theoretical Bayes error is 1.9%. Fig. 13 shows the same data plotted with $k = 3$. The distributions of samples in Figs. 12 and 13 are very similar, as we expect. The use of $k = 3$ provides a more accurate error estimate of 3%.

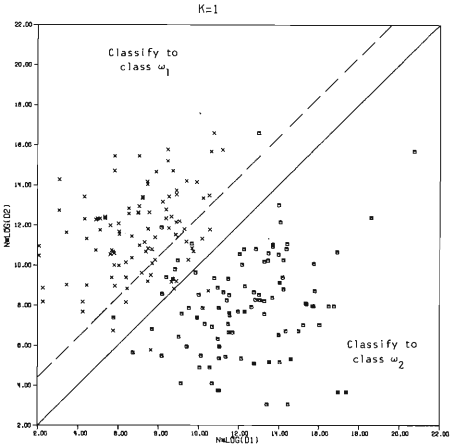


Fig. 12. Nonparametric Display with k=1 Using Standard Data (Solid line is Bayes minimum error classifier. Dashed line is Neyman-Pearson type classifier.): x - class ω_1 and □ - class ω_2

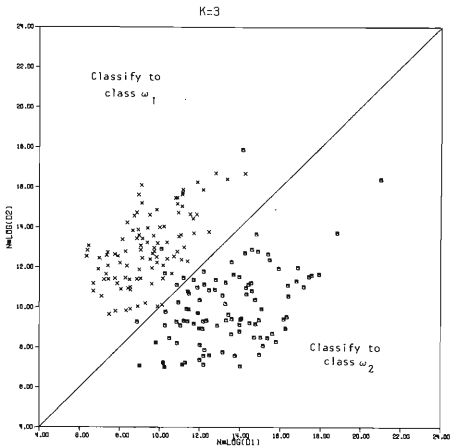


Fig. 13. Nonparametric Display Using Standard Data with k=3: x - class ω_1 and □ - class ω_2

Case When $P_i \neq N_i/(N_1 + N_2)$:

The standard k-NN majority vote classifier assumes that $P_1 = N_1/N_1+N_2$ and $P_2 = N_2/N_1+N_2$. When this is not the case, the standard k-NN classifier cannot be directly used. As an example, if $P_1 < N_1/N_1+N_2$, then $d_1(X) < d_2(X)$ will occur more frequently than it should, increasing the overall error rate.

One approach would be to discard samples to produce equality. This suffers from the obvious drawback of discarding preclassified samples. Since nonparametric techniques typically require a large number of samples, this may cause a significant deterioration in the performance of the classifier.

As a second approach we could generate enough artificial samples to produce equality. These samples would hopefully smooth the data. However, it is not clear how to artificially generate random samples from a distribution that is modeled using k-NN density estimates.

Fortunately, neither of these approaches are necessary, if we use (32) as a basis. Observe that (32) allows the specification of the N_i without concern for the P_i. By manipulating (32) a form that has an appealing interpretation results:

$$d_2(X)\left[\frac{N_2}{P_2N}\right]^{\frac{1}{n}} \gtrless d_1(X)\left[\frac{N_1}{P_1N}\right]^{\frac{1}{n}} \rightarrow X\epsilon\begin{cases}\omega_2\\ \omega_1\end{cases}, \tag{33}$$

where $N = N_1 + N_2$ and we have assumed $c_{21}-c_{22}=c_{12}-c_{11}$. It is seen that the distances for class ω_i are weighted by the multiplicative constant $\left[\frac{N_i}{P_iN}\right]^{\frac{1}{n}}$.

On the display this distance weighting is reflected in the position of the 45° boundary line. As an example, refer to Fig. 14, which was computed using the standard data of Fig. 12 with k=1, $N_1 = 50$, $N_2 = 150$, and $P_1 = P_2 = 0.5$. It was assumed that $c_{21}-c_{22}=c_{12}-c_{11}$.

The properties of this form of distance weighting were studied in [BR]. It was shown that the distance weighted classifier had improved finite sample classification performance when compared to the standard NN classifier. Furthermore, the distance weighted classifier was shown to have the same asymptotic performance as the standard NN classifier.

Another important use of the display is in minimax and Neyman-Pearson type nonparametric classifier design. In the minimax case if $c_{11} = c_{22} = 0$ and $c_{12} = c_{21}$, the design procedure consists of positioning the 45° line so that the number of class ω_1 samples in the ω_2 region equals the number of class ω_2 samples in the ω_1 region, i.e. the errors from both classes

are equal. For Neyman-Pearson type classifiers the 45° line is placed to provide a prespecified class error rate. By using the display one can readily determine the error rate for the other class. This makes tradeoff analysis much easier.

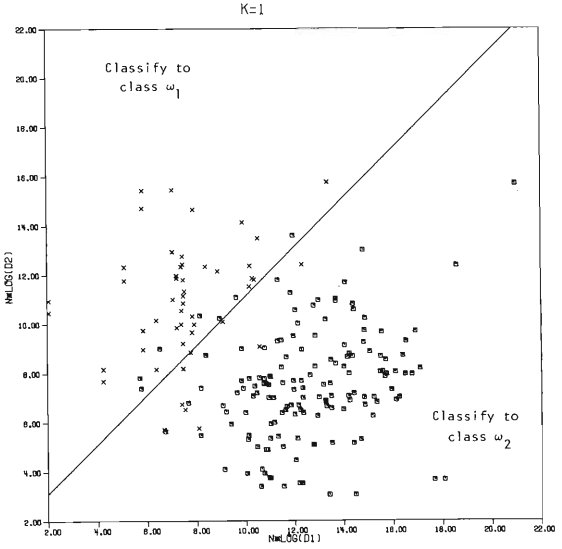


Fig. 14. Nonparametric Display with k=1 for Distance Weighting Given $P_1 = P_2 = 0.5$ with $N_1 = 50$ and $N_2 = 150$: x - class ω_1 and □ - class ω_2

As an example, suppose that a nonparametric classifier was to be designed for the standard data that had a class ω_2 error rate of approximately 2%. This is easily accomplished by translating the 45° line so that only two class ω_2 samples lie in the class ω_1 region. The dashed line in Fig. 12 corresponds to this design. Observe that a class ω_1 error rate of approximately 15% will result.

It should be noted that this is not the only design procedure for nonparametric Neyman-Pearson type classifiers. Most other nonparametric procedures are based on distribution-free tolerance regions [QU][AN][BE]. One of the major problems with this approach is determining the ordering functions that are used to construct the tolerance regions. The experimental results reported in [AN] and [BE] indicate this problem has not yet been completely solved. In [BE] the standard NN rule outperformed the distribution-free tolerance region classifier in both false alarm and miss probabilities in nearly all cases. This suggests that the Neyman-Pearson classifier design problem might be better handled in a k-NN framework. The display discussed in this Section allows this extension.

Error Estimation on Display:

It is frequently desirable to have risk information about the points in a data set. For the two class case the Bayes risk at a point X, $r^*(X)$, is given by

$$r^*(X) = \frac{P_1 p_1(X)}{P_1 p_1(X) + P_2 p_2(X)} \tag{34}$$

for $P_1 p_1(X) < P_2 p_2(X)$.

Upon substituting the k-NN density estimate into (34) and taking logarithms the result is

$$n \ln d_2(X) = n \ln d_1(X) + \ln \frac{N_1 P_2}{N_2 P_1} + \ln \frac{r^*(X)}{1 - r^*(X)} . \tag{35}$$

Hence for a given level of risk $r^*(X)$, a contour can be drawn on the display. The resulting contour is obviously a translated 45° line. Similarly, for $P_1p_1(X) > P_2p_2(X)$ it can be shown that the form

$$n \ln d_2(X) = n \ln d_1(X) + \ln \frac{N_1P_2}{N_2P_1} - \ln \frac{r^*(X)}{1 - r^*(X)} \tag{36}$$

results. Thus, the constant risk lines are symmetric about the Bayes decision line ($r^*(X) = 0.5$), and as $r^*(X)$ is decreased the constant risk lines move farther from the Bayes decision line. This result indicates that points mapped near the Bayes decision line on the display do in fact have a high degree of risk associated with them, while those points mapped far from the Bayes decision line have a low degree of risk. This desirable property will be true regardless of the underlying distribution. In Fig. 15 the 45° constant risk lines are drawn for values of $r^*(X) = 0.1$ to $r^*(X) = 0.5$ in increments of 0.1.

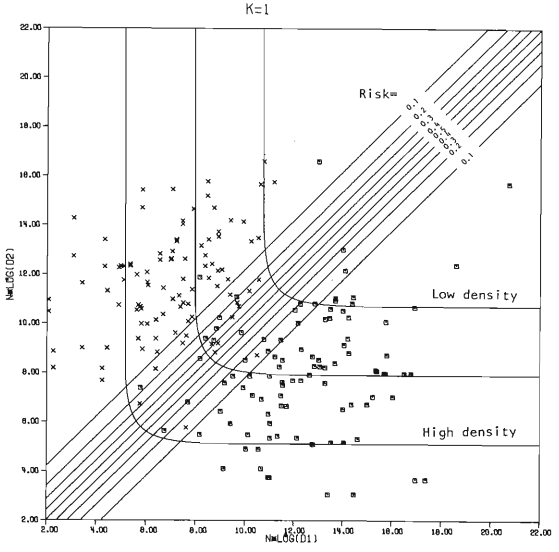


Fig. 15. Nonparametric Display with Constant Risk and Density Contours: x - class ω_1 and □ - class ω_2

These constant risk lines allow the user to easily specify a reject region [FU3]. All points mapped inside the region on the display specified by the reject threshold are rejected.

Past attempts to implement reject regions in a k-NN framework have been primarily based on class membership as in Tomek and Devijver [TO] [DE]. Unfortunately, large values of k were potentially required to provide a desired reject threshold. The proposed display easily allows the selection of any value of reject threshold for any choice of k. It also allows the user to see the effect of different choices of reject thresholds.

The reason for the increase in flexibility is simple. Standard k-NN classifiers use only voting (class membership) information. The display, however, includes distance information, which provides a basis for distance weighting. It is the creative use of distance weighting that provides the additional flexibility.

In a similar fashion it is possible to plot a contour of constant mixture density. The constant contours of mixture density D, after manipulation, can be expressed as

$$D = \frac{k}{c}\left[\frac{P_1}{N_1}\exp(-n \ln d_1(X)) + \frac{P_2}{N_2}\exp(-n \ln d_2(X))\right], \tag{37}$$

where $c = 2\pi^{n/2}\Gamma(n/2)$. Three curves of constant mixture density are plotted in Fig. 15.

An obvious method of error estimation is to count the classification errors present on the display. That is, count the number of □'s above and x's below the 45° boundary line. The error count obtained using the display constructed with a given k is identical to the error count of a standard (2k-1)-NN classifier [FU8]. The only assumption is that $P_1 = N_1/N$, $P_2 = N_2/N$, and $c_{21} - c_{22} = c_{12} - c_{11}$.

A second method of error estimation is based on (17) with the grouped estimate of $r^*(X)$ as (25).

Since the $\hat{r}_g$ may be directly obtained using the display, as is seen in the risk lines of Fig. 15, this error estimation technique is directly related to the display. It is also possible to solve (35) or (36) for $r^*(X)$ to provide the risk estimate.

Using this approach a user can compute and plot an error-reject graph [FU3], which can then be used for a more accurate specification of a reject region on the display.

V. Optimal Metric

The error bounds of Section II are derived, based on the assumption that $\eta_j(X) = \eta_j(\mathbf{X}_i)$ where $\mathbf{X}_i$ is the i-th NN of X. This is true with infinite samples at a continuity point of the conditional density functions. Also, these asymptotic bounds are independent of the particular metric to measure the nearness. However, when the number of samples becomes finite, the k-NN classification performance is no longer independent of the choice of the metric. The variance of the finite sample estimate may be minimized by the proper choice of the metric.

In this section we address the problem of selecting the best distance measure for minimizing the difference between the finite sample NN classification error and the asymptotic NN error.

Let us look at the NN risk, which was given in (9). Eq. (9) can be rewritten as

$$r(X,\mathbf{X}_1) = 2\eta_1(X)\eta_2(X) + [\eta_1(X)-\eta_2(X)][\eta_1(X)-\eta_1(\mathbf{X}_1)] \tag{38}$$

Note that, when $\eta_1(\mathbf{X}_1) = \eta_1(X)$ is satisfied, (38) becomes (10). Thus, the mean-square error between $r(X,\mathbf{X}_1)$ and its asymptotic counter part $2\eta_1(X)\eta_2(X)$ for a given X is

$$\overline{\epsilon^2}(X) = [\eta_1(X) - \eta_2(X)]^2 \underset{\mathbf{X}_1}{E}\{[\eta_1(\mathbf{X}_1) - \eta_1(X)]^2\} \tag{39}$$

In order to find the optimal metric, (39) is minimized with respect to metric.

Eq. (39) can be approximated by using a linear approximation of $\eta_1(\mathbf{X}_1)$ as

$$\eta_1(\mathbf{X}_1) = \eta_1(X) + \nabla\eta_1(X)^T(\mathbf{X}_1-X) \tag{40}$$

where $\nabla\eta_1(X)^T$ is the gradient vector of $\eta_1(X)$, transposed. Substitution of (40) into (39) gives

$$\overline{\epsilon^2}(X) = \underset{\mathbf{X}_1}{E}\{|\, V(X)^T(\mathbf{X}_1-X)\,|^2\} \tag{41}$$

where V(X) is $[\eta_1(X)-\eta_2(X)]^2\nabla\eta_1(X)$. This equation suggests that $\overline{\epsilon^2}(X)$ can be minimized by projecting all neighboring samples of X into the V vector and selecting the closest neighbor to X on the projected line as the NN [SH1]. It means that only the direction of V, not the magnitude, is relevant to find X_{NN}. However, it must be reminded that the above algorithm may be applied only to the samples close to X, because the local linearity assumption of (40) will be violated by the far away samples. It is a pleasing conclusion that the closeness among samples can be measured in one-dimensional space, rather than in the original high dimensional space. It suggests that the smaller variances may result.

The remaining question now is how to estimate $\nabla\eta_1(X)$. However, before discussing the problem, let us study $\nabla\eta_1(X)$ for Gaussian distributions. Two Gaussian distributions with the expected vectors M_i and the covariance matrices Σ_i (i=1,2) give

$$\nabla\eta_1(X) = \eta_1(X)\eta_2(X)[\Sigma_1^{-1}(X-M_1)-\Sigma_2^{-1}(X-M_2)] \tag{42}$$

Since both $\eta_1(X)\eta_2(X)$ and $[\eta_1(X)-\eta_2(X)]$ are numbers, the direction of the vector V(X) is determined by $[\Sigma_1^{-1}(X-M_1)-\Sigma_2^{-1}(X-M_2)]$. In particular, if $\Sigma_1=\Sigma_2=\Sigma$, (42) becomes

$$\nabla\eta_1(X) = \eta_1(X)\eta_2(X)[\Sigma^{-1}(M_2-M_1)] \tag{43}$$

which is independent of X in its direction. $\Sigma^{-1}(M_2-M_1)$ is known as the vector which is perpendicular to the Bayes classifier (hyperplane) for Gaussian distributions with equal covariance matrix. Thus, in this case we can project all samples onto the $\Sigma^{-1}(M_2-M_1)$ vector and measure the nearness on that line. On the other hand, if $\Sigma_1 \neq \Sigma_2$, the V vector varies, depending on X as (42) suggests.

$\nabla\eta_1(X)$ is estimated in general as follows. Set a fixed local spherical region $\Gamma(X)$ around X, and calculate the expected vector of the random vector in $\Gamma(X)$.

$$E\{(\mathbf{Y}-X)/\Gamma(X)\} = \frac{2d^2}{n+2}\frac{\nabla p(X)}{p(X)} \tag{44}$$

where d is the radius of $\Gamma(X)$. Eq. (44) is obtained under the linearity assumption of $p(Y) = p(X) + \nabla p(X)^T(Y-X)$ [FU6] [SH1]. Thus

$$\nabla\eta_1(X) = \frac{n+2}{2d^2}\,\eta_1(X)[E\{(\mathbf{Y}-X)/\Gamma(X),\omega_1\} - E\{(\mathbf{Y}-X)/\Gamma(X)\}] \tag{45}$$

In estimation, the expected values are replaced by the sample means. Also, note that finding $\mathbf{X}_1$ is not affected by the magnitude of V. Thus, the suggested procedure to estimate $\nabla\eta_1(X)$ is as follows:

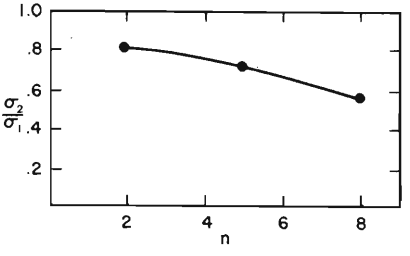
(1) Find the m NN's of X in the conventional euclidean metric.
(2) Calculate the sample mean of these m NN's, $\hat{M}_0$, and the sample mean of the class one samples among these m NN's, $\hat{M}_1$.
(3) Calculate $|(\hat{M}_1-\hat{M}_0)^T(\mathbf{X}_i-X)|$ $(i=1,\ldots,m)$ and select the smallest. This vector becomes the NN of X.

The above technique was applied to a two class distribution with $P_1 = P_2 = 1/2$ Class one and two are each normal with identity covariances and means separated by $||M_1-M_2|| = 2.6$. This gives a Byes error of .10 and an asymptotic NN error of .14. The experiment was run for n = 2, 5, and 8 using equal numbers of design and independent test data. In this experiment we compare the finite sample NN errors using d_e, the euclidean distance, and d_0, the optimal local distance. For a given design set, the NN errors, $\hat{\epsilon}_e$ and $\hat{\epsilon}_0$, are computed. Table 2 shows the sample mean ($\bar{\epsilon}_e$ and $\bar{\epsilon}_0$) and the sample variance (σ_1 and σ_2) of $\hat{\epsilon}_e$ and $\hat{\epsilon}_0$ for 15 trials at each dimension, with m = 5. It is interesting to note the improvement in $\hat{\epsilon}_0$ over $\hat{\epsilon}_e$ as n increases. This is further illustrated in Fig. 16 where σ_2/σ_1 is plotted as a function of n. There we see that the variance of $\hat{\epsilon}_0$ decreases more rapidly than that of $\hat{\epsilon}_e$.

Table 2 Sample Statistics of $\hat{\epsilon}_e$ and ϵ_o

n	N	$\bar{\epsilon}_e$	$\bar{\epsilon}_o$	σ_1	σ_2
2	200	.155	.145	.0299	.0255
5	200	.176	.140	.0224	.0162
8	500	.199	.154	.0238	.0133

Fig. 16. Sample Statistics as a Function of n



Although the detail discussions are out of the scope of this paper, it may be worth pointing out that various extensions of the above discussion are possible. They are:

(1) The local optimal metric for M-class problems is to minimize [SH2]

$$\left|\left[\sum_{i=1}^{M}\eta_i(X)\nabla\eta_i(X)\right]^T(\mathbf{X}_{NN}-X)\right| \tag{46}$$

That is, the NN of X is found on the projection into a vector $\sum_{i=1}^{M}\eta_i(X)\nabla\eta_i(X)$. Note that the

two class problem is a special case of (46) as $\eta_1(X)\nabla\eta_1(X) + \eta_2(X)\nabla\eta_2(X) = [\eta_1(X)-\eta_2(X)]\nabla\eta_1(X)$.

(2) The local optimal metric for the k-NN classification is to minimize [SH2]

$$\left| \nabla\eta_1(X)^T[\frac{1}{k}\sum_{j=1}^{k}(\mathbf{X}_j-X)] \right| \tag{47}$$

Again, we project samples into the $\nabla\eta_1(X)$ vector. But, this time the k-NNs of X must be selected to minimize the average distance on the projected line.

(3) If one wants to get the optimal metric which is independent of X, the following mean-square error could be used instead of (39)

$$\bar{\epsilon}^2 = \mathop{E}_{X}\{\bar{\epsilon}^2(\mathbf{X})\} \tag{48}$$

Assuming that the quadratic metric $(\mathbf{X}_1-X)^TA(\mathbf{X}_1-X)$ is sought, the optimal matrix A is

$$A = E\{\frac{[\eta_1(\mathbf{X})-\eta_2(\mathbf{X})]^2}{p(\mathbf{X})^{n/2}}\nabla\eta_1(\mathbf{X})\nabla\eta_1(\mathbf{X})^T\} \tag{49}$$

A can be interpreted as a nonparametric between class scatter matrix since A can be rewritten as

$$A = P_1E\{w(\mathbf{X})\nabla\eta_2(\mathbf{X})\nabla\eta_2(\mathbf{X})^T/\omega_1\} + P_2E\{w(\mathbf{X})\nabla\eta_1(\mathbf{X})\nabla\eta_1(\mathbf{X})^T/\omega_2\} \tag{50}$$

where $w(\mathbf{X}) = [\eta_1(\mathbf{X})-\eta_2(\mathbf{X})]^2/p(\mathbf{X})^{n/2}$ and the relationship of $\nabla\eta_1(\mathbf{X}) + \nabla\eta_2(\mathbf{X}) = 0$ is used. The lower dimensional feature space can be found by selecting a set of eigenvectors of A whose eigenvalues are large [FU9].

In the previous discussion, we saw that we can treat the effect of the sample size by approximating $\eta_1(\mathbf{X}_1)$ by (40). The same treatment can be applied to discuss the sensitivity of the sample size to the Bayes error bounds.

The upper bound due to the NN classification is given in (38) which shows the deviation of $[\eta_1(X)-\eta_2(X)]\nabla\eta_1(X)^T[\mathbf{X}_1-X]$ from the asymptotic case in the first order approximation. On the other hand, the lower bound of the 2-NN classification can be expressed as

$$r(X,\mathbf{X}_1,\mathbf{X}_2) = \eta_1(X)\eta_2(\mathbf{X}_1)\eta_2(\mathbf{X}_2) + \eta_2(X)\eta_1(\mathbf{X}_1)\eta_1(\mathbf{X}_2) \tag{51}$$

Applying the same approximation of (40), (51) becomes

$$r(X,\mathbf{X}_1,\mathbf{X}_2) = \eta_1(X)\eta_2(X) + 2\nabla\eta_1(X)^T[\mathbf{X}_1-X]\nabla\eta_1(X)^T[\mathbf{X}_2-X] \tag{52}$$

where $\nabla\eta_1(X) = -\nabla\eta_2(X)$ is used. The second term of (52) is the second order term and the first order term disappeared in (52). Therefore, we can conclude that the 2-NN error is less sensitive to the sample size. Since the NN error is exactly twice the 2-NN error in the asymptotic case, and the effects of sample size are different in these two cases, a comparison of these two errors may suggest whether the available sample size is adequate.

VI. Conclusion

The Bayes error of a given data set is one of the most important parameters in pattern recognition. Yet, the estimation of the Bayes error does not allow us to use any assumption on the mathematical structure of density functions and calls for nonparametric estimation techniques. Thus, the k-NN approach is one of the most promising ways to do the job.

The author has tried to clarify some of the problems on the subject with his former coworkers, Drs. D. L. Kessell, L. D. Hostetler, R. D. Short, J. M. Mantock and T. E. Flick. This paper is an assembly of their work.

Unfortunately, there are still many unanswered questions. The author has experienced, on many occasions, strange phenomena which could not be understood, particularly for high dimensional data. It is the author's hope that these unsolved problems will be answered in the near future.

References

AN M. W. Anderson, and R. D. Benning, "A distribution-free discrimination procedure based on clustering," *IEEE Trans. Information Theory,* Vol. IT-16, pp. 541-548, September 1970.

BE G. W. Beakley, and F. B. Tuteur, "Distribution-free pattern verification using statistically equivalent blocks," *IEEE Trans. Computers,* Vol. C-21, No. 12, pp. 1337-1347, December 1972.

BR T. A. Brown and J. Koplowitz, "The weighted nearest neighbor rule for class dependent sample sizes," *IEEE Trans. Information Theory,* Vol. IT-25, pp. 617-619, September 1979.

CA T. Cacoullos, "Estimation of a multivariate density," *Ann. Inst. Stat. Math.,* Vol. 18, pp. 179-189, 1966.

CO1 T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Information Theory,* Vol. IT-13, pp. 21-27, January 1967.

CO2 T. M. Cover, "Rates of convergence of nearest neighbor decision procedures," *Proc. 1968 Hawaii International Conference on Systems Theory,* pp. 413-415.

CO3 T. M Cover, "Learning in a pattern recognition," in *Methodologies in Pattern Recognition,* M. Watanabe, Ed., New York: Academic Press, 1969.

DE P. A. Devijver, "New error bounds with the nearest neighbor rule," *IEEE Trans. Information Theory,* Vol. IT-25, pp. 749-753, November 1979.

FI E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination," USAF School of Aviation Medicine, Randolph Field, Texas, Project 21-49-004, Rept. 4, Contract AF-41-(128)-31, February 1951.

FR J. H. Friedman, F. Baskett and L. J. Shustek, "An algorithm for finding nearest neighbors," *IEEE Trans. Computers,* Vol. C-24, pp. 1000-1006, October 1975.

FU1 K. Fukunaga, *Introduction to Statistical Pattern Recognition,* Academic Press, New York, 1972.

FU2 K. Fukunaga and D. L. Kessell, "Estimation of classification errors," *IEEE Trans. Computers,* Vol. C-20, pp. 1521-1526, December 1971.

FU3 K. Fukunaga and D. L. Kessell, "Application of optimum error and reject tradeoff," *IEEE Trans. Information Theory,* Vol. IT-18, pp. 814-817, November 1972.

FU4 K. Fukunaga and D. L. Kessell, "Nonparametric Bayes error estimation using unclassified samples," *IEEE Trans. Information Theory,* Vol. IT-19, pp. 434-440, July 1973.

FU5 K. Fukunaga and L. D. Hostetler, "Optimization of k-nearest neighbor density estimates," *IEEE Trans. Information Theory,* Vol. IT-19, pp. 320-326, May 1973.

FU6 K. Fukunaga and L. D. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Information Theory,* Vol. IT-21, pp. 32-40, January 1975.

FU7 K. Fukunaga and L. D. Hostetler, "K-nearest neighbor Bayes risk estimation," *IEEE Trans. Information Theory,* Vol. IT-21, pp. 258-293, May 1975.

FU8 K. Fukunaga and J. M. Mantock, "Nonparametric two-dimensional display," *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. PAMI-4, pp. 427-436, July 1982.

FU9 K. Fukunaga and J. M. Mantock, "Nonparametric discriminant analysis," to appear in *IEEE Trans. Pattern Analysis and Machine Intelligence,* Vol. PMMI-6, 1984.

FU10 K. Fukunaga and P. M. Narendra, "A branch and bound algorithm for computing k-nearest neighbors," *IEEE Trans. Computers,* Vol. C-24, pp. 750-753, July 1975.

HA P. E. Hart, "The condensed nearest neighbor rule," *IEEE Trans. Information Theory,* Vol. IT-14, pp. 515-516, May 1968.

HE M. E. Hellman, "The nearest neighbor classification rule with a reject option," *IEEE Trans. Systems, Man, Cybernetics,* Vol. SMC-6, pp. 179-185, July 1970.

KO1 W. L. G. Koontz and K. Fukunaga, "A nonparametric valley-seeking technique for cluster analysis," *IEEE Trans. Computers,* Vol. C-21, pp. 171-178, February 1972.

KO2 W. L. G. Koontz, P. M. Narendra and K. Fukunaga, "A graph-theoretic approach to nonparametric cluster analysis," *IEEE Trans. Computers,* Vol. C-25, pp. 936-944, September 1976.

LO D. O. Loftsgaarden and C. P. Quesenberry, "A nonparametric estimate of a multivariate density function," *Ann. Math. Statist.,* Vol. 36, pp. 1049-1051, 1965.

PA E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Stat.,* Vol. 33, pp. 1065-1076, 1962.

QU C. P. Quesenberry, and M. P. Gessaman, "Nonparametric discrimination using tolerance regions," *Ann. Math. Stat.,* Vol. 39, no. 2, pp. 664-673, 1968.

SH1 R. D. Short and K. Fukunaga, "The optimal distance measure for nearest neighbor classification," *IEEE Trans. Information Theory,* Vol. IT-27, pp. 622-627, September 1981.

SH2 R. D. Short and K. Fukunaga, "A new nearest neighbor distance measure," *Proc. 5th International Conference on Pattern Recognition,* pp. 81-86, December 1980.

ST C. J. Stone, "Consistent nonparametric regression," *The Annals of Statistics,* Vol. 3, No. 4, pp. 595-645, 1977.

TO I. Tomek, "A generalization of the k-NN rule," *IEEE Trans. Systems, Man, Cybernetics,* Vol. SMC-6, No. 2, pp. 121-126, February 1976.

WA T. J. Wagner, "Convergence of the nearest neighbor rule," *IEEE Trans. Information Theory,* Vol. IT-17, pp. 566-571, 1971.

WI D. L. Wilson "Asymptotic properties of nearest neighbor rule using edited data," *IEEE Trans. Systems, Man, and Cybernetics,* Vol. SMC-2, pp. 408-421, July 1972.

YU T. P. Yunck, "A technique to identify nearest neighbors," *IEEE Trans. Systems, Man, Cybernetics,* Vol. SMC-6, pp. 678-683, October 1976.

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

Decision Trees in Pattern Recognition

G.R. Dattatreya and L.N. Kanal

Machine Intelligence and Pattern Analysis Laboratory
Department of Computer Science
University of Maryland
College Park, MD 20742

1 Introduction

An intuitively appealing and aesthetically elegant method in decision making is to proceed in multiple stages, taking partial decisions in the course. Such procedures have several attractive features, the most important of which, perhaps, is understandability. The decision maker gets the comfortable feeling that he or she (or the algorithm, if implemented on a machine) is in control of the situation so that some errors in the partial decisions can be corrected at later stages or even by backtracking. In many instances, taking such partial decisions is conceptually simpler as they are required to examine the information relevant to the present stage only. This would also save the expense of gathering information not required for the present stage. These points have contributed to the popularity of multistage decision making in several engineering problems, pattern recognition being a major one.

The appropriate definition of pattern recognition in the context of multistage methods is *the assignment of a physical object or an event to one of the prespecified categories.* The main problem in multistage systems is the design of a classification scheme. Other important problems are the definition of proper information bearing elements for classification, specification of methods for their extraction, data modeling and validation, and estimation of performance of the designed scheme. Very often, only a limited set of data is given to aid the design and this may necessitate making sweeping assumptions on the statistical structure of the data. Decision trees offer distinct advantages when multivariate representation is unrealistic for the limited amount of data. If a decision tree is well designed, the result is a classification scheme which is accurate, flexible, and computationally efficient. The flexibility of the decision making model enhances its

applicability in a wide range of problems. However, this less restrictive representation opens up so many possibilities in tailoring a multistage scheme that there is no straightforward method to design decision trees. The designer can allow frameworks which have extremely complex decision functions at each node or very simple ones like binary decisions based on threshold comparison of single variables. Furthermore, a tree which is simple to design may not be simple in operation on test samples and vice versa. These points tend to confuse the designer who might finally end up with a decision tree without being able to specify why or how the particular classifier is well suited to the problem at hand.

The objectives of this review are (i) to consolidate the major methodologies for decision tree design, (ii) to bring out their commonalities, (iii) to provide insight into the mechanism of multistage classification, (iv) to explode certain myths such as the idea that decision trees are always simple to design and to use, (v) to mention the areas of their applications, and in general (vi) to aid the designer in selecting an appropriate technique or in evolving a suitable one for the particular problem of interest. The need for a fresh review arises from three counts. First, the increasing recognition that globally optimal multistage schemes are neither necessary nor beneficial for practical applications of short duration. Second, the reluctant acceptance of the fact that the training sample set size is almost always smaller than desirable. And finally, as a consequence, the development of widely varying techniques in decision tree design.

The main difficulty in achieving the goals set above is the virtual impossibility of briefly describing the works of different authors in a common notation. Section 2, after presenting the main notation, establishes the need and describes a method for multistage decision making for certain pattern recognition problems. Section 3 categorizes a host of decision tree frameworks into three main groups. The difficulty even in merely using a tree, given a multistage structure with specified features and classes, is clearly brought out. A tour of the major design methods is taken in Section 4. Not much work has been reported on the performance analysis of decision trees. Most design methods rely on training and test set accuracies. Section 5 touches upon this aspect. Section 6 is devoted to a comparative discussion of all the methods and some specific problems not otherwise covered. Several applications are indicated in Section 7 and Section 8 summarizes major conclusions.

A convenient reference to begin discussion on pattern recognition trees is Kanal [35], which surveys interactive approaches to pattern analysis. At that time, it was noted that the performance on independent test data of self learning and other new approaches fell far short of expectation. This led to the recognition that the key to pattern classification problems does not lie wholly in learning machines, statistical approaches, spatial filtering, heuristic programming, formal linguistic approaches, or in any other particular solution vigorously advocated by one or another group at that time. Interactive pattern analysis is a way of allowing examination of the structure in the pattern data. Hierarchical structures form a very important category of data representation, and decision trees are naturally suited for pattern recognition with such a data representation. Wu et al. [63] describe the decision tree approach to the classification of multivariate normally distributed multispectral data. The major advantages considered there are the improvement in classification accuracy with finite samples, computational efficiency, and convenience in applications. They provide insight into the dimensionality problem and propose a tree approach to overcome it. The histogram approach, the sequential clustering, and the optimization approach to the design are described.

Kulkarni [40] investigates theoretical properties of a general model of multistage multiclass recognition schemes. The model allows a large class of parametric and nonparametric schemes within its framework. He proposes two classes of search strategies for obtaining optimal decisions. Hierarchical classifiers are more practical special types of multistage recognition schemes. A three phase approach to the design of hierarchical classifiers is proposed. The individual phases are the tree skeleton design, feature selection at its nodes, and the decision function design at each node. These optimal design methods are computationally cumbersome for moderate to large problems. Several heuristic methods are explored to reduce the design complexity. Some of these are (i) clustering decision rules and rejecting sets of suboptimal rules without evaluating them individually and (ii) feature ranking and branch and bound methods. Relationship between performance, sample size, and the number of quantization levels is also examined. Anderson and Fu [1], in developing linear binary trees for leukocyte classification, review other methods such as the state space, the game tree, and the partitioning approaches.

In describing earlier work, Bell [4] proposes arriving at a decision table with initial seed rules and interactively changing the rules from *don't care*

to *yes* or *no* and vice versa. The decision table so obtained is converted to binary trees by one of several suggested methods. Payne and Preece [53] attempt a synthesis of a large and widely dispersed literature on identification keys and diagnostic tables. They define a *key* as a two column table. The first column is a set of tests. The second is the identification or the index to the next test to be conducted. On the other hand, a diagnostic table is defined to have as many rows as the number of identifications (classes). Entries corresponding to second column onwards are the properties representing the identifications. The authors discuss the relative merits of the two kinds of representations and cite pattern recognition as an application for their use. They touch upon optimality concepts and probabilistic and decision theoretic approaches for selection of tests (features). They also cite botanical, coding, electronic, medical, microbiological, and zoological applications as well as applications to decision tables and prime implicants. Moret [48] surveys decision trees and diagrams having widespread applications wherever discrete functions need to be evaluated sequentially. He establishes a common framework of definitions and reviews contributions from several fields of applications like databases, decision table programming, concrete complexity theory, switching theory, and pattern recognition. The recent book by Breiman et al. [6] develops classification and regression trees as a tool for statistical data analysis. Their main methodology is top-down splitting to structure and categorize the data. This is suited more for data analysis and nonlinear regression than for current pattern recognition problems. The present article emphasizes decision trees and their applications to pattern recognition.

2 A Formal Basis for Decision Trees

A convenient way of introducing decision trees is by considering a problem for which the optimal classifier works out to be a multistage decision making scheme. Incorporating the costs of measured features as part of the total cost for minimization provides such a basis.

The fundamental problem in statistical pattern recognition is the design of the Bayes minimum risk classifier. This scheme minimizes the expected cost of classification when general cost values are associated with possible combinations of correct and wrong classifications. An excellent

treatment of this problem is available in Duda and Hart [21]. If the cost of each misclassification is unity and that of each correct classification is zero, the decision scheme is identical to that of the minimum probability of error. An extension of the minimum risk classification problem is in allowing the measurement of features to incur costs.

In such a pattern recognition environment, the decision maker or the classification scheme would have to assess the worth of the information to be revealed by a feature measurement in relation to its cost. The possible reduction in the classification risk by using the outcome of a measurement may be negligible in comparison with the cost of measuring the feature. The best example to illustrate this is in medical diagnosis. To diagnose among any set of suspected disorders, there is usually a large set of tests available. Some of them may be very costly, injurious, or even potentially fatal to the patient. Thus, unless necessary, not all tests will be conducted. Intuitively, it is better to start with simple tests. Based on the outcomes of these tests, it may be possible, in some instances, to make a decision. Even if more tests are needed, the outcomes previously observed may indicate which of the costly tests is likely to provide the information required for diagnosis. Thus, decision making proceeds in several stages. At every stage the scheme considers two main options: Deciding on a class label and terminating the process, or measuring a feature to gather more information. It should be noted that if the measurements do not cost anything, there is nothing gained by not measuring a feature, so the best scheme would conduct all the measurements and decide on a class label in a one shot manner. This reduces the scheme to the minimum risk classifier.

2.1 The minimum cost pattern recognition problem.

The above qualitative arguments are developed further by studying the properties of the optimal scheme for the following problem. Given

(i) A set Ω of M classes ω_i with their *a priori* probabilities $P(\omega_i)$, $i=1, \ldots, M$,

(ii) A vector of N features, $F=(f_1, \ldots, f_N)$ within which the features are arbitrarily ordered, with their corresponding costs of measurement c_i, $i=1, \ldots, N$,

(iii) M completely known class conditional probability density functions (or mass functions) $p(x \mid \omega_i)$, $i=1, \ldots, M$, where $x=(x_1, \ldots, x_N)$ is an N-variate outcome of the N features, and

(iv) The $M \times M$ loss matrix Λ with $\lambda_{ij} \geq 0$ representing the cost incurred if a pattern sample from a true class ω_i is decided as belonging to ω_j,

it is required to arrive at a decision making scheme with the minimum total expected cost, where the total cost is the sum of the measurement cost and the classification risk.

Obviously, on any pattern sample, using more features will decrease the risk of misclassification but increase the cost of measurement. The extreme possibilities are (i) to conduct no measurement at all and classify the sample on a minimum *a priori* risk basis and (ii) to conduct all the feature measurements (incurring their costs) and minimizing the *a posteriori* risk. The best choice lies somewhere in between, and the difficulty is that it varies from sample to sample, making the design of the optimal scheme exceedingly complex. Earlier arguments provide some ideas about the behavior of the optimal scheme. These ideas are refined further by reviewing some elementary but interesting facts about pattern recognition.

Most important of all is the fact that the optimal classifier for the minimum cost problem is a multistage decision making scheme. By proceeding in multiple stages, the scheme retains the option of dispensing with a costly feature not measured yet, if measurements already conducted indicate (statistically) its suboptimality. If, however, the observed outcomes indicate that further measurements are required, one or more of the remaining features can be measured. Extending this argument further, it is easy to show that measurement of only one feature per stage is optimal. Formal proofs are available in Dattatreya [18]. Wald [62] considers an even more general problem wherein the costs of feature measurements are functions of their outcomes and the sequences in which they are measured. In some such cases, measurement of more than one feature at a stage turns out to be optimal. However, for practical pattern recognition, feature costs are best modeled as constants allowing one feature per stage to be optimal.

If features are required to be measured one at a time, a straightforward solution is an optimal sequence of feature measurements. The best sequence can be arrived at for a given problem. The decision scheme picks

up the next specified feature for measurement if the expected accuracy (or the classification risk) is to be improved. This is the sequential pattern recognition extensively studied by Fu [25]. A sequence of features that does not change as measurements are taken is not optimal as the following example shows; at any intermediate stage, the best feature for the next measurement is a function of all the measurements available.

Example: A two class pattern classifier measures a feature f_1 at the first stage. The costs of measuring two more available features f_2 and f_3 are equal and are high enough to stop further measurement, after measuring one of them at the second stage. All the features are binary and are class conditionally statistically independent. The class conditional probability mass functions of f_2 and f_3 are given in Table 1. The updated probabilities of class ω_1 at the end of the first stage, i.e., $P(\omega_1 \mid x_1)$ are given in Table 2. The objective of this example is to show that f_2 is better than f_3 for one outcome of f_1, and f_3 is better than f_2 for the other outcome of f_1. The probabilities of errors for the two schemes are tabulated in Table 2 as functions of the outcomes of f_1. Clearly f_2 yields a lower error rate if $x_1=0$ and f_3, if $x_1=1$. Thus the decision scheme shown in Fig. 1(c) is better than those in Figs. 1(a) and 1(b). In the figure, nonterminal nodes conduct specified feature measurements and let the decision scheme enter the descendant nodes based on the outcome of the measured feature. The outcomes are indicated on the relevant branches.

2.2 Dynamic programming solution.

Note that we have logically shown the theoretical need for decision trees for a class of pattern recognition problems (those involving feature measurement costs), rather than resorting to decision trees as an alternative to some other scheme. The example is trivial since it is given that using all the three measurements is prohibitively costly. Had there been more candidate features of tolerable costs, we would have to ask two questions: (i) What is the expected cost if further measurement is abandoned in favor of deciding on the least risky class label? (ii) What is the expected cost (sum of the measurement cost and the classification risk) if the scheme measures a candidate feature f_i and uses the optimal policy thereafter? To answer these questions, we need a notation for multistage decision making

Table 1
Probability mass functions

Class	$P(x_i \mid \omega_j)$			
	x_2		x_3	
	0	1	0	1
ω_1	0.6	0.4	0.2	0.8
ω_2	0.1	0.9	0.7	0.3

Table 2
$P(\text{error} \mid x_1)$ as function of feature used at the II stage

x_1	$P(\omega_1 \mid x_1)$	Error	
		f_2	f_3
0	0.4	0.22	0.26
1	0.6	0.28	0.24

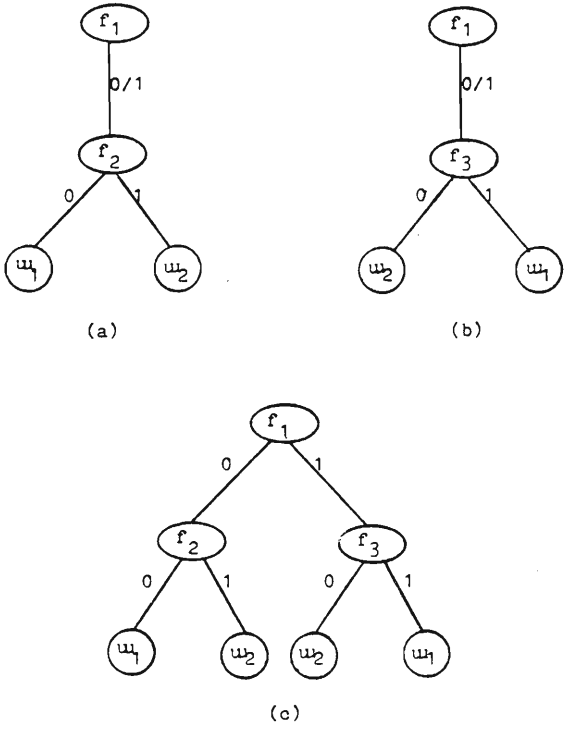


Fig. 1 Alternate schemes for the example.

structures. Many terms, e.g., nodes, branches, and direct descendants, are assumed to be unambiguous.

n_i is the i-th nonterminal node in the decision making structure. Nonterminal, in the sense that there is a feature measurement at n_i. The way these nodes are numbered is not specified yet. A terminal node is a direct descendant of a nonterminal node and corresponds to some class decision ω_j. In general n_i has both terminal and nonterminal sets of direct descendant nodes.

$D(n_i)$ is the set of nonterminal direct descendant nodes (if any) of the nonterminal node n_i.

If n_j is a nonterminal descendant of n_i, $f(n_i n_j)$ is the feature to be measured at n_j, if the decision scheme descends from n_i to n_j. Naturally, $f(n_i n_j) \in \{f_1, \ldots, f_N\}$. $c(n_i n_j)$ is the cost of measuring $f(n_i n_j)$. $z(n_i n_j)$ is the outcome observed by measuring $f(n_i n_j)$.

$F(n_i)$ is a subset of the feature set F and comprises all the features measured up to and inclusive of the node n_i. $y(n_i)$ is the outcome observed after measuring all of $F(n_i)$. $y(n_i)$ is $|F(n_i)|$-variate within which individual outcomes are ordered as in $x = (x_1, \ldots, x_N)$.

After feature measurement at a node n_i, the risk or the expected classification cost (to be incurred if it is decided to classify the sample as belonging to ω_j) is given by

$$R(\omega_j \mid y(n_i)) = \sum_{k=1}^{M} \lambda_{kj} P(\omega_k \mid y(n_i)) \tag{2.1}$$

using the above notation and the terminology of the problem defined earlier.

$$R^*(y(n_i)) = \min_{\omega_j \in \Omega} [R(\omega_j \mid y(n_i))] \tag{2.2}$$

is the minimum risk (conditioned on $y(n_i)$, the observations made) if it is decided to terminate feature measurement and make a classification after node n_i.

However, if the nonterminal node n_i has descendants which are themselves nonterminal, further feature measurement is possible. To decide between further feature measurement and classification, we need the expectation of the cost yet to be incurred if further measurement is to be conducted. Let n_j be a direct descendant of n_i; $E^*(y(n_j))$, the optimal expected cost yet to be incurred after observing $z(n_i n_j)$ at node n_j. Note that $z(n_i n_j)$ and $y(n_i)$ together (ordered appropriately) constitute $y(n_j)$. Also $E^*(y(n_j))$ itself depends on the optimal decisions at n_j and its descendants. The expected cost yet to be incurred after n_i upon deciding to descend to n_j is the expected cost to be incurred at and after n_j. This is the cost of measuring the feature at n_j, i.e., $c(n_i n_j)$, plus the optimal expected cost to be incurred thereafter. This should be properly averaged over all outcomes of $f(n_i n_j)$, since we have not observed $z(n_i n_j)$, but are only contemplating moving to n_j from n_i. Thus

$$E(n_i \mid y(n_i)) = c(n_i n_j) + \int_{z(n_i n_j)} E^*(y(n_j))p(z(n_i n_j) \mid y(n_i))dz(n_i n_j) \tag{2.3}$$

is the optimal expected cost, conditioned on the observed outcomes $y(n_i)$, yet to be incurred if a decision to descend to node n_j from n_i is taken. But descending to n_j in particular may not be optimal if there are other elements in $D(n_i)$. Therefore

$$E^*(y(n_i)) = \min \{ R^*(y(n_i)), \min_{\omega_j \in D(n_i)} [E(n_j \mid y(n_i))] \} \tag{2.4}$$

is the minimum expected cost of all possible decisions at node n_i conditioned on all observations made up to that stage, $y(n_i)$.

At some nodes in the feature graph, there will be no more features for further measurement. Such nodes have only terminal nodes as descendants. If n_k is such a node, $F(n_k)=F$, the total feature set and $y(n_k)=x$, the N-variate outcome. The minimum expected cost (yet to be incurred) at such a node is given by the minimum risk of classification itself.

$$E^*(n_k \mid x) = \min_{\omega_j \in \Omega} \sum_{i=1}^{M} \lambda_{ij} P(\omega_i \mid x) \tag{2.5}$$

Therefore, given a multistage decision making structure such as the one in Fig. 1(c), it is possible by the above recursive dynamic programming procedure to compute the minimum expected cost of the overall tree. A byproduct of such a computation is the optimal decision rules at each stage, for each outcome. But if we are given just the statistics and costs, how do we proceed? The properties of the optimal classifier discussed earlier help us. At any stage, consider measuring one feature, at a time, of all the features not yet measured. This is too unwieldy to be of practical use, especially if the number of features is large and the features are continuous. Nevertheless it is useful to study such a mechanism of optimal decision making scheme and its design.

2.3 Representation of the decision graph.

The decision making structure (to be called the feature graph) over which dynamic programming equations can be written comprises

(i) A root node corresponding to the zeroth stage with no feature assignment,

(ii) N levels of nodes below the root node, corresponding to N possible stages of feature measurement,

(iii) $N-i$ direct descendant nodes to every i-th stage node (connected by unidirectional links); each direct descendant node measures a feature not already measured at the first i stages on a path leading to the i-th stage node under consideration, and

(iv) $N!$ distinct but overlapping paths from the zeroth stage, corresponding to the $N!$ orderings of the features.

Two or more paths merge at a node at the i-th stage if they measure the same set of i features up to and inclusive of the merging node. Thus there is a node corresponding to every subset of features. Totally there are 2^N nodes including the root node corresponding to the null feature set. A

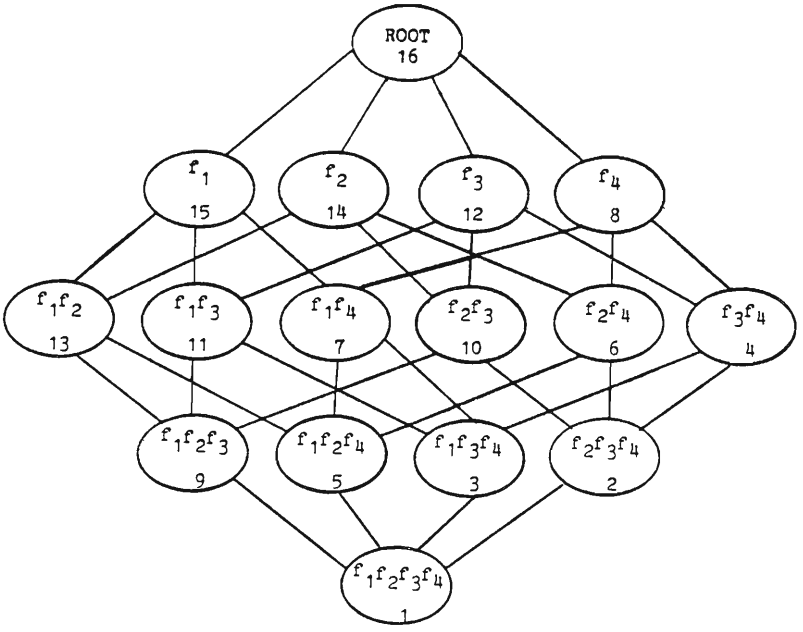


Fig. 2 The feature graph of a four feature problem.

feature graph with four features is drawn in Fig. 2. The nodes are numbered in a particular order. Notice that if n_j is a descendant of n_i, then $n_i > n_j$. With such a numbering, the optimal costs during the dynamic programming procedure are evaluated starting from the first node and going upwards according to the node numbers, till the root node. This ensures the availability of the optimal costs of all the descendants of a node when they are required to evaluate the optimal cost of the node under consideration.

The node numbering in Fig. 2 appears peculiar and does not advance one row at a stretch. This is done to simplify the computation of the node numbers of all direct descendants of an arbitrary node while computing its

optimal cost. Mathematically, let

$$L(n_i) = (\gamma_1(n_i), \ldots, \gamma_N(n_i)) \tag{2.6}$$

be a binary word corresponding to every node n_i in the feature graph such that

$$\begin{aligned} \gamma_j(n_i) &= 0, \text{ if } f_j \in F(n_i) \\ &= 1, \textit{ otherwise.} \end{aligned} \tag{2.7}$$

$L(n_i)$ denotes the features measured up to and inclusive of the node n_i. Then the node number (or the node index) i of the node n_i is given by

$$i = 1 + \sum_{j=1}^{N} \gamma_j(n_i)\, 2^{j-1} \tag{2.8}$$

This allows computation of a node number given the features measured and vice versa. Note that the index number of the root node (zeroth stage) is 2^N, at which no feature is measured. The number of direct descendants of the node n_i is just the number of *ones* in the binary word $L(n_i)$, since $\gamma_j = 1$ if f_j has not been measured. Thus

$$|D(n_i)| = \sum_{j=1}^{N} \gamma_j(n_i) \tag{2.9}$$

Let $d_k(n_i)$ be the k-th direct descendant of node n_i, i.e.,

$$D(n_i) = \{d_k(n_i)\},\ k = 1, \ldots, \sum_{j=1}^{N} \gamma_j(n_i) \tag{2.10}$$

We are interested in computing the node number of $d_k(n_i)$. Let $d_k(n_i) = n_j$. Note that $L(n_i)$ is obtained by replacing the k-th *one* by *zero* in $L(n_i)$. Therefore

$$j = i - 2^{m-1} \tag{2.11}$$

where m is the least integer satisfying

$$\sum_{t=1}^{m} \gamma_t(n_i) = k \tag{2.12}$$

It is now possible to systematically work out the dynamic programming procedure over the node numbered (or indexed) feature graph starting from the first node at which all the features would have been measured. The complete methodology is outlined below.

Step 1: For node n_1 evaluate

$$E^*(y(n_1)) = \min_{\omega_j \in \Omega} \sum_{k=1}^{M} \lambda_{kj} P(\omega_k \mid x)$$

This also yields appropriate decision rules for all $y(n_1)$. Note: $y(n_1) = x$.

Step 2: For n_i, $i = 2,\ldots,(2^N - 1)$, evaluate

$$R^*(y(n_i)) = \min_{\omega_j \in \Omega} \sum_{k=1}^{M} \lambda_{kj} P(\omega_k \mid y(n_i))$$

For $n_j = d_k(n_i)$, $k = 1,\ldots, |D(n_i)|$, evaluate

$$E(n_j \mid y(n_i)) = c(n_i n_j) + \int_{z(n_i n_j)} E^*(y(n_j)) p(z(n_i n_j) \mid y(n_i)) dz(n_i n_j)$$

$$E^*(y(n_i)) = \min \{ R^*(y(n_i)), \min_{n_j \in D(n_i)} E(n_j \mid y(n_i)) \}$$

This also yields the decision rules for all $y(n_i)$.

Step 3: At root node, n_i, $i = 2^N$, $y(n_i) = \phi$ is empty; evaluate

$$R^*(n_i \mid \phi) = \min_{\omega_j \in \Omega} \sum_{k=1}^{M} \lambda_{kj} P(\omega_k \mid \phi)$$

For $n_j = d_k(n_i)$, $k = 1, \ldots, \mid D(n_i) \mid$, evaluate

$$E(n_j \mid \phi) = c(n_i n_j) + \int_{z(n_i n_j)} E^*(y(n_j)) p(z(n_i n_j) \mid \phi) dz(n_i n_j)$$

$$E^*(\phi) = \min \{ R^*(n_i \mid \phi), \min_{n_j \in D(n_i)} E(n_j \mid \phi) \}$$

The optimal decision rule is just the optimal feature to be measured at the first stage, unless the problem turns out to be pathological, in which case the optimal policy is deciding a class label being based on the *a priori* risk.

The infeasibility of the above procedure for practical pattern recognition problems with even a moderate number of features is immediately obvious. First of all, for continuous features, it is not possible to obtain closed form expressions for $E(n_j \mid y(n_i))$ even with the simplest of probability density functions. This, and the fact that different decision regions yield different functions for integration over the measured feature space, makes averaging to obtain the optimal costs and the decision rules for all values of the continuous variable $y(n_i)$ virtually impossible. Class conditional statistical independence of features does not simplify the procedure, since the conditional density $p(z(n_i n_j) \mid y(n_i))$ used in eqn. (2.3) is a mixture over all classes. Mixture densities can never have statistically

independent components. If

$$p(x_1x_2) = P(\omega_1)p(x_1x_2 \mid \omega_1) + P(\omega_2)p(x_1x_2 \mid \omega_2) \tag{2.13}$$

can be factored as $g_1(x_1)g_2(x_2)$, then $p(x_1x_2 \mid \omega_1)$ and $p(x_1x_2 \mid \omega_2)$ need to have the same factors (since these are nonnegative functions integrating to the same value., i.e., unity) causing the features to degenerate into the same density function for both the classes.

For discrete features, the optimal procedure can be refined for machine design of decision trees, but only for a modest number of features with at most a few discretization levels for each. The method described in [19] and [20] orders all the outcomes of all the features measured up to the present stage (in the feature graph, e.g., in Fig. 2) in a particularly convenient way. A recursive computation of optimal costs is then implemented on the computer.

Another drawback of this theoretical framework is that it requires the complete specification of the underlying probabilistic structure. In practice, a pattern recognition problem is usually defined from a set of training (labeled) samples. The number of samples will be inadequate to model or estimate the dependence of features purely by statistical means. Other peculiarities like the inability of the formal model to incorporate computational burden have led to alternate models for multistage pattern recognition.

3 Models for Multistage Pattern Recognition

Fu [25], viewing the minimum cost problem as a generalized version of sequential pattern recognition, proposed an algorithm for both feature ordering and classification. But the resulting scheme was not interpreted as or simplified to a practically applicable decision tree. Subsequently, several frameworks of decision trees, which essentially try to exploit the favorable properties and alleviate some of the problems associated with the minimum cost decision scheme, have evolved. Individual models concentrate on subsets of aspects of the attractive properties and also use techniques for overcoming specific disadvantages. Though such an approach is undesirable

from the point of view of developing a general theory, it should be borne in mind that for a given problem with a finite set of training samples and limited computational resources, it is neither possible nor necessary to attempt very general solutions. Concentrating on problem specific peculiarities aids in achieving the best returns.

The specific peculiarities and regularities of a particular pattern recognition problem are arrived at by pattern analysis. Interactive pattern analysis (Kanal [36], [37]) is a way of allowing examination of the structure of data in a high dimensional space. Humans are superior in recognizing certain types of structures, as for example in distinguishing clusters even in the presence of outliers. Human examination is generally facilitated by graphic displays of various data transformations. The procedure consists of using whatever is known about the problem at hand to guide the gathering of data about the patterns and pattern classes which may exist in the environment being examined, and then subjecting the data to a variety of procedures for inferring deterministic and probabilistic structures that are present in the data. This is also called exploratory data analysis and includes Histogram plots, Scatter plots, Cluster analysis routines, Linear discriminant analysis, Nonlinear mapping, Analysis of variance, Regression analysis, etc. A relevant example of an interactive pattern analysis system is the ISPAHAN (Gelsema [26]). This system allows the user to interactively investigate the structure of multidimensional data using a menu of options including supervised and unsupervised classification, mapping algorithms, and feature transformations. It also supports the interactive analysis and design of multistage classification systems. In another recent study, Friedman and Stuetzle [24] report some *projection pursuit* schemes for detection, description, and validation of structure in p-dimensional point clouds. These and many other data analysis methods help in selecting an existing multistage model or in developing a new model for the pattern recognition problem at hand.

3.1 State space models.

A designed decision structure for the minimum cost problem typically looks like the one shown in Fig. 3. The feature to be measured initially at the first stage is specified. Thereafter, the nodes to be traversed depend on the observed outcomes. As explained in Section 2, it is very difficult to keep track of or specify the optimal decision rules, since these involve averaging

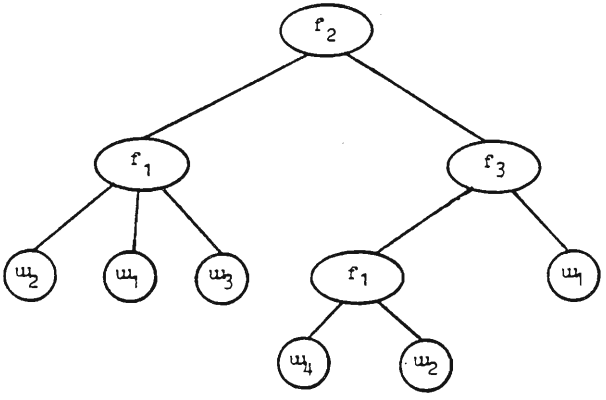


Fig. 3 A designed decision structure (hypothetical).

over all possible costs at descendant nodes. Since it is not possible to conform to globally optimal decision rules in practice, the problem can be modified to search for a goal node which minimizes the sum of the costs of measurements at all the nodes leading to the particular goal node, and the risk conditioned on the measurements. The procedure involves feature measurement not merely along the path to the finally decided terminal node, but also at a few other nodes in an attempt to reduce the total cost as much as possible. If we are given the optimal expected cost yet to be incurred beyond the present stage, only those nodes along the path to the final node to be decided need be traversed, reducing the scheme to that of the minimum cost. This suggests the use of lower bound estimates similar to those used by Hart et al. [28] in graph searching, to reduce *going out of the way* from the path leading to the *unknown* goal node. This is the philosophy behind the state-space models proposed and studied by Kanal's group [38]-[43]. The notation introduced in Section 2 is used here rather than that used in the original references.

A state-space graph G is similar to a subgraph (Fig. 3) of the feature graph (Fig. 2) except that the nonterminal nodes in G can measure more than one feature at a node and each nonterminal node n_i keeps track of a subset of classes which appear as its descendant (not necessarily direct

descendant) terminal nodes. The terminal nodes with class labels are also explicitly included in the state-space graph, which is not the case in the feature graph. Mathematically G is a seven-tuple

$$G = \{S, E, F, \Omega, C, R, T\}$$

where

(i) $S = \{n_i\}$ is the set of nodes (states).

(ii) $E = \{n_i n_j\}$ is the set of directed edges. There is an edge from node n_i to n_j only if n_j is a direct descendant of n_i.

(iii) F is the total feature set.

(iv) Ω is the total class set.

(v) C is a non-negative real valued cost function on the edges; $c(n_i n_j)$ is the cost of features measured with a transition from n_i to one of its direct descendants n_j.

(vi) R is a real-valued function on the goal (terminal) nodes representing the expected value of the cost or loss incurred in making the classificatory decision associated with the goal node, and

(vii) T is the strategy used in deciding the nodes to be traversed in the graph.

The essential difference between this formulation and the decision graph of Fig. 3 is that $f(n_i n_j)$, the feature measured in a single transition from n_i to n_j is not necessarily single. $f(n_i n_j)$ is a subset of F. Similarly $c(n_i n_j)$ is the sum of the costs of all features in $f(n_i n_j)$ and $z(n_i n_j)$ is a vector variable rather than a scalar. These definitions are extended also to the case when n_j is not a direct descendant but some descendant of n_i. That is, $f(n_i n_j)$ is the set of features measured beyond n_i up to and inclusive of n_j. Also, $\Omega(n_j)$, a subset of Ω, is the set of classes under consideration at n_j. If n_j is a terminal node, $|\Omega(n_j)| = 1$. Apart from these adjustments, the notation is essentially the same as in Section 2. A few more terms are introduced as needed. It is unfortunate that the complexity

of possible situations in multistage pattern recognition does not allow the use of a simple notation.

An important class of schemes within the state-space model is the hierarchical classifier in which groups of classes form a hierarchy. Such schemes have the advantage of being able to use *a priori* knowledge concerning the physical and biological relationships among the subgroups of the class set. Even in situations without such relationships, an exploratory data analysis may reveal the ease of discriminating different subsets of classes using different subsets of features. This approach will also reduce the complexity of the resulting classifier. The longest path in the state-space graph will not be greater than the number of classes, since at least one class is rejected at a stage. Thus, $\Omega(n_j)$ is a proper subset of $\Omega(n_i)$ if n_j is a descendant of n_i. At the root node (or the initial state) no feature is measured and the set of all classes is considered. Also, in hierarchical classifiers each node has a unique parent.

A classification process is a strategy for searching in the graph for a terminal node with the least cost. This process depends on the cost formulation. The cost of a goal node n_i is the sum of the costs of measurements and the risk of classification. Risk could be a function of observations only along the path to the accepted goal node or on all the measurements. The search strategy naturally depends on this. In the first case the total cost of the terminal node n_i is

$$s(n_i) = C(n_i) + R(\Omega(n_i) \mid y(n_i)) \tag{3.1}$$

where $y(n_i)$ represents the observations along the path to n_i only. In the second case

$$s(n_i) = C(n_i) + R(\Omega(n_i) \mid x) \tag{3.2}$$

where x is the observation due to all possible measurements. The following algorithms, Algorithm-S and Algorithm-B, correspond respectively to these cases.

Algorithm-S

Step 1: Put the root node in a set *OPEN*. Empty a set called *CLOSED*.

Step 2: For each $n_i \in OPEN$, evaluate

$$s(n_i) = C(n_i) + h(n_i) + r(n_i)$$

where $h(n_i) = 0$ if n_i *is a goal node*

$$\leq \min_{n_j \in T(n_i)} C(n_i n_j) \text{ if } n_i \text{ is not a goal node}$$

$$r(n_i) = R(\Omega(n_i) \mid y(n_i)) \text{ if } n_i \text{ is a goal node}$$

$$= \min_{n_j \in T(n_i)} \min_{z(n_i n_j)} [R(\Omega(n_j), z(n_i n_j)] \text{ if } n_i \text{ is not a goal node}$$

Step 3: Let n_k be the node among all nodes in *OPEN* with minimum $s(\,)$. Remove n_k from *OPEN* and add it to *CLOSED*. Put all the direct descendants of n_k in *OPEN*. If n_k is a terminal node, accept the corresponding class label and exit. Else Go To Step 2.

In the above algorithm, the $s(n_i)$ function computed is a lower bound on the total cost which would be incurred by a pattern sample given that it has entered node n_i with feature observation $y(n_i)$. If this lower bound is tight enough to equal the actual cost, the sample pattern will reach the appropriate goal node without making unnecessary measurements. Else the scheme will conduct measurement not only on the path to the finally accepted goal, but at a few more nodes. However, what is minimized is the sum of classification risk at the goal node and the cost of only those feature measurements along the path to the goal node.

The assumption that the risk is a function of the observations along the path to the goal node (and not at other nodes) may be valid in problems wherein different feature sets are known to be useful for different

classes. In extreme cases class conditional densities of certain features may not be defined or may degenerate to be equal for certain classes. In general, however, risk is a function of all measurements (see eqn. 3.2) for which the following procedure is relevant.

Algorithm-B

Step 1: Put the initial state in *OPEN*. Empty a set called *CLOSED*.

Step 2: For each node $n_i \in OPEN$, evaluate

$$s(n_i) = C(n_i) + h(n_i) + r(n_i)$$

where $h(n_i) = 0$ if n_i *is a goal node*

$$\leq \min_{n_j \in T(n_j)} C(n_i n_j), \text{ if } n_i \text{ is not a goal node}$$

$$r(n_i) \leq \min_{v} R(\Omega(n_i) \mid y(n_i), v), \text{ if } n_i \text{ is a goal node}$$

$$\leq \min_{v} \min_{n_j \in T(n_i)} R(\Omega(n_j) \mid y(n_i), v) \text{ if } n_i \text{ is a non goal node}$$

where v is the outcome of all features not in $F(n_i)$

Step 3: Let n_k be the node in *OPEN* with the minimum $s(\,)$. Move n_k from *OPEN* to *CLOSED* and put all the direct descendants of n_k (if any) in *OPEN*.

Step 4: For each *goal node* $n_i \in CLOSED$, evaluate the upper bound $b(n_i)$ where

$$b(n_i) = C(n_i) + \max_{v} R(\Omega(n_i) \mid y(n_i), v)$$

Let n_k be the terminal node in *CLOSED* for which $b(\ )$ is minimum. If for all nodes $n_j \in OPEN \cup CLOSED$, $n_j \neq n_k$, $b(n_k) \leq s(n_j)$, then exit with n_k as the B optimal goal; Else Go To Step 2.

When risk is a function of all the measurements, one can terminate the search only when it is certain that additional measurements will not cause any other goal to be rated higher than a goal node already in the *CLOSED*. This is accomplished by using the upper bounding function $b(n_i)$. Formal proofs that these algorithms are admissible in their respective senses are available in Kulkarni [40]. He also discusses state space models for nonparametric pattern recognition like the nearest neighbor schemes wherein the costs of feature measurement again give rise to multistage schemes. Typically, the cost to be minimized is the sum of measurement cost and the distance from the test pattern to the nearest neighbor. The recognition scheme can be implemented as a search strategy similar to those discussed above. These search strategies assume the existence of the state space graph appropriate to the problem at hand. The design of such decision structures is dealt with in Section 4.

3.2 Independent subrecognition models.

One of the ways of simplifying pattern recognition is by splitting the problem into several smaller ones. If one looks for a globally optimal scheme, the interaction of these subschemes gets intolerably complex, and designing them, even worse. A practical trade off would be to design the subschemes ignoring their interdependence. A logical combination of the results of these subschemes should then be worked out for the final categorization of the input pattern sample.

Ichino [33]-[35] studies a few models of this kind. Major specifications of such models are the type of subschemes to be used and the method of combining their results. The first type is the dual class recognition system. In this method, a classifier is designed for each pair of classes using a subset of features for each pair. The output of each subclassifier is the label of one of the two classes considered. Let there be M classes. To arrive at the final

class label, a count of each class from the output of $M(M-1)/2$ subsystems is taken. The maximum count for any class is $M-1$, since $M-1$ classifiers consider any particular class. Also, if one class yields a count $M-1$, no other class can do so, since any other class would have been rejected by at least the classifier considering that class and the one which yielded a count of $M-1$. Therefore Ichino suggests that a sample be finally decided as belonging to a class if and only if that class yields a count $M-1$. Otherwise, the pattern is rejected as unclassified. However, it should be noted that other combinations like majority count decisions are possible. This method has some advantages when *a priori* knowledge or data analysis indicates the possibility of discriminating different class pairs with small feature subsets. But if the number of classes is large, the number of subrecognition schemes, $M(M-1)/2$, tends to be impractical.

Alternately, the subschemes can consider one class against the rest, in which case there will be only M subschemes. This is called the symmetric classwise subrecognition. Final classification is done only if one subscheme accepts the particular class. There are two types of rejections in this method. One, when two or more subrecognition schemes accept their respective classes. The other, when no scheme accepts its class. One method of avoiding the first type of rejection is to accept the first class a subrecognition scheme accepts. But it is then important to order the classes in a particular way.

When a large number of classes form a homogeneous cloud of points in some metric space, the best method is to design a good classifier for a small set of class labels drawn at random from the original large set. This classifier can be used as the last stage of a multistage scheme in which many classes are rejected with a high accuracy at initial stages. This model is proposed and used for speaker recognition in large populations by Dante [12], and Dante and Sarma [13], [14]. The technique they use to simplify the model is to assume, for analytical purposes, that the class label itself is a continuous variable along the continuous feature variables. The state of the classification system at any stage is defined as the number of speakers still under consideration. The problem of classifier design then works out to be the allocation of features at appropriate states and the definition of policies for state transitions. The problem is solved by using stochastic optimal control techniques.

3.3 Top down models

The discussion so far makes it clear that the only way to achieve optimal decisions in multistage pattern recognition is to use the optimal costs of possible courses of action beyond the present stage. This bottom up procedure requires enormous computation and memory. In practice the user may be more interested in simple schemes leading to a compromise between the complexity of operation and the performance in terms of costs. An obvious model for this purpose is the one wherein the decision scheme explicitly specifies the action as a function of the feature measured at the stage of interest. This is an advantage from the point of view of the operation of the classifier only, and adds to the complexity of design. Dattatreya and Sarma [18] propose one such method and demonstrate it for a very small problem with three Gaussian features. The method involves designing the minimum cost scheme and modifying it to a decision tree which takes decisions by threshold comparisons with the measured feature at any stage. Let n_j be a direct descendant of the first stage node n_i, and n_k be a direct descendant of n_j. With the usual terminology (consider only one feature per stage), the optimal cost yet to be incurred if a decision to descend to node n_k is taken,

$$E^*(n_k \mid y(n_j))$$

is a function of the observation $z(n_i n_j)$ at n_j, as well as the previous single observation $y(n_i)$. The best way to get rid of $y(n_i)$ is by averaging over $y(n_i)$.

$$E^*(n_k \mid z(n_i n_j)) = \int E^*(n_k \mid y(n_i), z(n_i n_j))\, p(y(n_i))\, dy(n_i) \tag{3.3}$$

where the region of integration extends over only those values of $y(n_i)$ for which a pattern sample reaches the node n_j. This, done for all direct descendants n_k of n_j, facilitates decision making at n_j based on the function of only one variable $z(n_i n_j)$. The resulting decision scheme can be implemented using threshold comparisons. Averaging should be done top down, after designing the minimum cost scheme, and involves eliminating only one variable at a time. This is a technique to modify the complex decision regions into hyper rectangular regions to arrive at a very simple scheme for

operation. However, beyond providing some insight into the mechanism of multistage classification, the method is of little use for practical problems. In general, simplicity of design is also very important.

Top down methods in decision tree design usually involve repeatedly selecting a feature and segmenting its region. Usually these methods design classification schemes from a relatively small set of labeled samples and use nonparametric splitting rules. Two distinct classes of these methods are

(i) Partitioning the entire feature space and converting the resulting decision scheme to a tree. Conversion of these tables to trees may be top down or bottom up. Henrichon and Fu [30] present such a procedure which generally does not correctly classify all the design samples. Meisel and Michalopoulos [47] present another for one hundred percent correct classification.

(ii) Partitioning features one at a time and repeating the same on resulting subregions to dynamically develop a tree.

Recall that a classifier can always be designed to correctly classify all the given labeled samples. Since this will have an adverse effect on the test samples to be classified (see Hughes [31]), many of the above methods incorporate stopping rules to prevent too fine a splitting of the feature space.

Some schemes use a one step look ahead or simple evaluation functions for costs beyond the present stage, rather than being purely top down. Breiman et al. [6], [7] use a tree complexity function in addition to the accuracy of classification. These top down tree design models do not consider the costs of feature measurements as defined in Section 2. The obvious question, therefore, is why do we use decision trees at all? Wu et al. [63] bring out the need for decision tree classifiers to overcome the problems of dimensionality and small sample size. Other advantages like computational efficiency during operation of decision trees and understandability of the classification process justify their use. The next section discusses design methodologies for decision trees.

4 Decision Tree Design

This section consolidates a large number of tree design techniques proposed for multistage pattern recognition. These are categorized into three groups. Section 4.1 discusses conversion of decision tables to trees. This problem is also relevant to a variety of other topics in computer science. Only those methods directly related to pattern recognition are discussed in some detail. Other references are merely mentioned. Section 4.2 is concerned with hierarchical classifiers and Section 4.3 combines a large body of methodologies into a dynamic segmentation framework.

4.1 Conversion of decision tables to trees.

The problem here is to design a decision tree to efficiently evaluate the value of a multiple input, multiple output function given the values of independent variables. Each of the independent variables has a finite domain. In pattern recognition, the range of the output is the set of classes. In general, one can think of multiple outputs each of which takes a value from a finite set.

Stoffel [60] proposes a prime events theory to deal with pattern recognition problems involving discrete variables. The procedure arrives at a set of distinct templates or prototypes for each pattern category. Such a complete set of prototypes for all the categories can be viewed as a decision table. The set of prime events covers all the labeled design set samples but not the entire domain of the set of features. Therefore there is a reject class label.

Sethi and Chatterjee [57] propose a procedure for converting such a set of prototypes to a decision tree scheme with the following properties: a multiple outcome feature test is used only once as a binary question along a path to a terminal node; there are as many terminal nodes as the number of prime events (prototypes). Using an estimate of the average number of further comparisons required, the procedure selects a feature for binary splitting at every node till it reaches a unique prototype. The estimate assumes that the prototypes under consideration at a node will reach terminal nodes in a binary subtree, as balanced as possible. The tree will not have a reject option. Also, it is not possible to design a binary tree with a multivalued feature used only once along a path to a terminal node. An example to illustrate this is a single key binary search tree. In general, trees

will be more efficient if we allow the same prototype to occur at more than one terminal node since the prototypes allow *don't care* entries. Sethi and Chatterjee [57] also demonstrate their method in the recognition of hand printed characters.

Meisel and Michalopoulos [47], and Payne and Meisel [52] study the problem of converting hyper rectangular decision regions to binary trees with each node making a threshold comparison of the ordinal scale of a feature outcome. Let each feature take a finite number of values x_{ij}, $j=1, \ldots, m_j$. Because these could be obtained by discretizing a metric space, the condition $x_{ij} < x_{i(j+1)}$ is imposed. The nonterminal nodes in the decision tree to be developed are binary comparisons of the form $y_i \leq x_{ij}$. y_i is the outcome of the feature f_i for the pattern sample under consideration. Terminal nodes are class labels corresponding to the decisions. The data is given in the form of class decisions for all the combinations of the outcomes of the (discretized) feature set and the probabilities of such outcomes. Tree design is posed as an optimization problem with a general class of optimization criteria which includes minimizing the average number of comparisons, the total number of comparisons, and the maximum possible number of comparisons in the decision tree. A feature can be used many times with different thresholds of comparisons along the path to a terminal node. Let

$$\alpha = \{x_{ij}\},\ j=1, \ldots, m_i;\ i=1, \ldots, N \tag{4.1}$$

be the *lattice* of hyper cuboids in the N dimensional feature space. A node with a threshold comparison of y_t with x_{tk} yields two sublattices.

$$\{x_{ij}\},\ j=1, \ldots, m_i \text{ for } i \neq t;\ j=1, \ldots, x_{tk} \text{ for } i=t$$

is the sublattice confirming $y_t \leq x_{tk}$ and

$$\{x_{ij}\},\ j=1, \ldots, m_i \text{ for } i \neq t;\ j=x_{t(k+1)}, \ldots, m_t \text{ for } i=t$$

is the complement sublattice, i.e., that corresponding to $y_t > x_{tk}$. Thus every nonterminal node splits a lattice into *left* and *right* sublattices. A

sequence of such binary questions on resulting sublattices ultimately ends up with a single point sublattice or a sublattice with all points having the same class label. The cost of a sublattice is the same as that of the tree used to classify a sample lying within the sublattice. Note that the cost of a lattice is a componentwise monotonically nondecreasing function of the costs of the left and right sublattices obtained with any split. Therefore, if the optimal costs of all possible pairs of sublattices are available, it is easy to choose the best split. There are

$$\sum_{i=1}^{M} (m_i - 1)$$

comparisons and hence as many pairs of sublattices for the total lattice α.

Using the principle of optimality, the cost of optimally splitting a lattice can be worked out by a bottom up procedure of computing the optimal splitting costs of all sublattices starting from single point sublattices. In [47] and [52], functional equations for dynamic programming are developed and an ordering of the set of all sublattices is given to facilitate recursive computation. The sublattices are ordered in a one dimensional array with a one to one correspondence between a sublattice and its index in the array. To achieve a simple correspondence, they define the *size* and *position* of a sublattice. The size is the vector of the number of points along each direction, e.g., the size of the sublattice in eqn. (4.1) is $(m_1, \ldots, m_N)$. Since we need to consider all sublattices of all sizes, a *position* of the sublattice needs to be defined. This is defined as the vector of coordinates of a point in the sublattice with its minimum coordinates along all directions. The set of all sublattices is partially ordered, with sublattices of the same size ordered according to their positions, and groups of sublattices ordered according to their sizes. Fig. 4 shows a two dimensional decision space, and Fig. 5, the corresponding tree with minimum number of nonterminal nodes.

There are also many other table conversion methods, most of them useful in applications other than pattern recognition. Bayes [3], and Schumacher and Sevick [56] use dynamic programming to convert limited entry tables to trees. The procedure is also extended to *extended entry* tables in [56]. Cerny et al. [9] realize multiple output Boolean functions by binary decision programs. Hartman et al. [29] define an upper bound on the efficiency criterion of a given decision tree using information theoretic

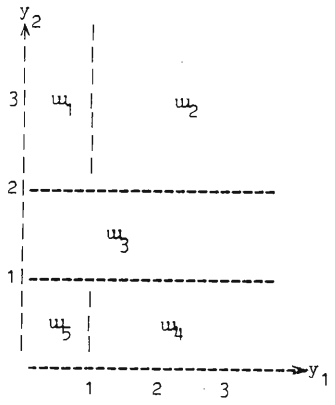


Fig. 4 Decision regions for a hypothetical problem.

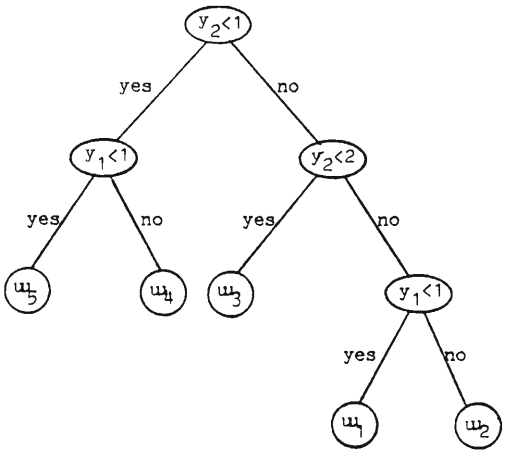


Fig. 5 Binary tree with minimum number of nonterminal nodes for the decision space shown in Fig. 4.

concepts. Then, a decision tree is developed so as to minimize this upper bound at each step of its construction. This is a systematic procedure and leads to near optimal trees.

The discussion on decision table to tree conversion is not complete without some mention of the computational complexity of optimal tree design. Hyafil and Rivest [32] consider the following problem. There is a set of classes, elements of which are uniquely represented by Yes/No answers on a set of tests. The external path length of a terminal node is defined as the number of questions required to reach the terminal node, plus one. They show that minimization of the sum of external path lengths of all terminal nodes is an NP-complete problem. This implies that as the size of the problem increases, computation required to design an optimal tree increases to unmanageable amounts. Comer and Sethi [10] study several related problems in *trie* construction for information storage and retrieval and show them to be NP-complete. A distinction is required here between the computational complexity and the cost criterion of pattern recognition. When trees or tries are designed for information storage and retrieval, they are usually required to be modified frequently, with insertions and deletions of data. It is therefore not profitable to look for optimal structures during every insertion or deletion. On the other hand, a pattern recognition tree can have a cost criterion not related to the computational complexity. For example, in medical diagnosis, it is very important to minimize the cost of tests required to detect the disorder. These costs are related to the injuries and risks to the human body due to clinical or other tests. It might therefore be desirable to design an optimal tree for practical problems of modest sizes. Furthermore, such trees would not require frequent redesigning.

4.2 Hierarchical classifier design.

Hierarchical classifiers are multistage pattern recognizers in which classes are sequentially rejected along a path to a finally accepted class label. In general, the hierarchical subgrouping of classes, the features required at nonterminal nodes of the hierarchy and the decision rules are interdependent. Given a class hierarchy, it is useful to think of individual nodes themselves as pattern classifiers. However the node classifiers can not be designed independently of each other. Kulkarni [40] brings out this problem by deriving the total probability of error as a function of the

individual node error probabilities and concludes that (i) decision rules which maximize the individual node performances do not maximize the overall performance. (ii) features which maximize the individual node performances do not maximize the overall performance. But arriving at a globally optimal tree by searching for a tree skeleton (class hierarchy) with the best feature assignment and decision rules is not feasible even for moderate problems. Let us assume that every node splits a class set into two non overlapping subsets. Then a class set may be recursively split by finding the optimal cost of splitting all subsets of the class set. In keeping with our earlier terminology, let n_1 be the node at which a set of classes $\Omega(n_1)$ needs to be split into two non overlapping descendant node sets $\Omega(n_2)$ and $\Omega(n_3)$. Let $E^*(n_i)$ be the optimal cost at and beyond the node n_i. The cost of a tree is defined as the cost of features measured at the top node plus the error rate at the top node plus the costs of the two descendant subtrees. Let $c(F(n_1))$ be the cost of all features used at the node n_1, $P_e(n_1 \mid \omega_i)$ the probability of error at node n_1 if the true class of a sample is ω_i. Then,

$$E^*(n_i) = \min_{F(n_1)} \{ \min_{\Omega(n_2)} \{ \sum_{\omega_i \in \Omega} [P(\omega_i) c(F(n_1)) + P_e(n_1 \mid \omega_i)] + E^*(n_2) + E^*(n_3) \} \} \tag{4.2}$$

where $\Omega(n_2)$ is any subset of $\Omega(n_1)$ and $\Omega(n_3) = \Omega(n_1) - \Omega(n_2)$.

Kulkarni [40] shows that such a formulation leads to the optimal tree when node error rates are much smaller than unity. It is clear that the method of building up the skeletons from smaller and smaller subsets of classes is computationally very complex. Kulkarni also proposes schemes for reducing the complexity. One such strategy is to cluster the decision rules and represent each cluster by a prototype. Then only the prototypes would need searching, rather than the entire set of decision rules.

An alternative approach is to design a good tree skeleton, i.e., a tree structure with only class sets at its nodes, by agglomerative clustering. By defining a suitable distance function over the entire feature space, a distance matrix between all the labeled samples may be computed. This facilitates hierarchical merging of subgroups to obtain the tree skeleton. A branch and bound method can then be used for feature allocation at

different nodes. Whenever the assignment of a feature reduces the correct recognition rate below a lower bound on the optimal performance, that sequence is abandoned. The algorithm backtracks to the previous stage and tries a new sequence.

Once the tree skeleton and feature allocation are obtained, decision rules may be worked out in a bottom up procedure as described in Section 2. Since keeping track of decision rules is again complex, search strategies given in Section 3 may be used for operating the classification process.

4.3 Dynamic tree development.

Dynamic tree development methods select a feature and split its region at every stage (node). Later stages handle smaller regions of the feature space and repeat the procedure. They are essentially top-down procedures. Swain and Hauska [61] develop an evaluation function to optimize decision trees. Every node n_i in a tree is taken as a classifier and a performance measure $E(n_i)$ is defined for the node. Since it is not feasible to keep track of all the nodes below (or beyond) the present node, a one step look ahead is used. The present node is optimized by assuming that the direct descendant nodes are conventional one stage classifiers. The node performance is defined as an appropriate linear combination of computation time, classification error, and descendant node performances.

$$E(n_i) = [t_o(n_i) - t(n_i)] + K[e_o(n_i) - e(n_i)] + \sum_j P_j E_o(n_j) \tag{4.3}$$

where $t(n_i)$ is the computation time, $e(n_i)$ is the error, P_j is the probability of reaching the possible descendant n_j from n_i. The subscript o is used for the conventional one shot classifier. Thus the equation denotes the improvement in the expense of the present node over the conventional classifier. Specification of the designed node involves a feature assignment, class subsets for the different descendants, and the probabilities of reaching these descendants. All possible specifications are searched to find the best node configuration. Note that $E_o(n_j)$ is the performance of a possible node n_j when the class set under consideration is classified by the conventional one shot maximum likelihood classifier. The procedure at the node n_i is also maximum likelihood but does not attempt to identify all the classes

individually. Several experiments are reported in [61] using this method for machine design of classifiers for remotely sensed data.

You and Fu [65] design binary trees by splitting a set of classes into two nonoverlapping subsets at every stage. The two subsets are arrived at by defining a measure of separability for different pairs of subsets over different feature spaces of fixed size sets. The separability measure is equivalent to the squared Mahalanobis distance between the two subgroups of labeled samples. The mean vectors used for computing this distance are the subgroup averages and the covariance matrix is the average of the matrices for the subgroups. The decision rules at each node are linear discriminant functions over the selected feature spaces. Mui and Fu [50] use quadratic decision rules at each node in a similar framework with feature selection performed by solving the eigenvalue-eigenvector problem and choosing the first two canonical variables. Argentiero et al. [2] utilize a set of two dimensional canonical transformations and Bayes table look up decision rules. They demonstrate the scheme with LANDSAT data.

Casey and Nagy [8] develop a binary tree for optical character recognition using information theoretic concepts. Characters are treated as a grid of *black* and *white* pixels. *A priori* class probabilities and class conditional frequencies of individual pixels are estimated from labeled samples. Examination of any pixel reduces the uncertainty (information theoretic entropy) of the character class. The tree is designed from the root node to the leaves by selecting that pixel for examination which maximizes the reduction in the current entropy. At a leaf node, if the probability of correct recognition is less than a prespecified value, samples (characters) reaching the leaf are rejected as unsuitable for machine classification.

A variety of other techniques use simple dynamic data splitting to design binary trees. Friedman [23], Breiman et al. [6], [7], Fielding [22], and Rounds [55] develop trees with relatively small training sample sets. In [23] and [55], the feature which maximizes the Kolmogorov-Smirnov distance between two estimated cumulative distributions is used. The distribution is estimated as the fraction of the number of samples which lie below or at the point x at which the distribution is required. This yields a stepped function. The feature with the largest difference between the distributions of the two classes is selected and partitioned at points at which the distributions cross each other. A feature used once can be used again. Therefore, stopping rules are required for terminating splitting. Those suggested are to stop when there is less than a prespecified number of samples to be split or

when the confidence level of the Kolmogorov-Smirnov test is not acceptable. For multiclass problems, Friedman [23] suggests building M trees, one for each class considered against the rest, and using majority logic for the final class decision. Gustafson et al. [27] use the same splitting rule but consider all pairs of classes and use the best among them at every stage of splitting.

Fielding [22] discusses binary segmentation in AID, an automatic interaction detector for exploring data structures. The idea is to develop a tree type of representation of the effect of independent variables (predictors) on the dependent variables. If there is a single dependent variable taking nominal values, the procedure is directly useful in pattern recognition. The strategy is to examine each independent variable and partition the same into two subsets such that a distance function between the dependent variables of the two subgroups is maximized. The independent variable (or the feature) with the maximum of these maxima is chosen for partitioning. Successive subgroups are further partitioned. The distance function used is the frequency weighted sum of squares of deviations of subgroup means from the parent mean of the dependent variable. If the independent variables are nominal, it may be appropriate to consider all pairs of mutually exclusive and collectively exhaustive subsets as possible splits. However, if they are interval or ordered, only splits which separate smaller values from larger values of the independent variable are considered.

For the nominal dependent variable, Morgan and Messenger [49] have developed algorithms called THAID for binary segmentation. The dependent variable which takes a value ω_j among $\omega_1, \ldots, \omega_M$ is represented as a M-long binary word with only the j-th bit being a *one* and the rest *zeros*. The centroid of a group of samples is a vector with its individual elements representing the frequency of occurrence of each ω_j in the group. Let there be m_1 samples from one group and m_2 in the other, totaling to m. If p_{1j} and p_{2j} are the frequencies of ω_j in groups one and two respectively, THAID defines a distance function between the two groups as

$$\frac{m_1 m_2}{2} \sum_{j=1}^{M} | p_{1j} - p_{2j} | .$$

This helps in computing the best split.

Breiman et al. [6], [7] develop binary segmentation based on maximum improvement of classification. Let $\{t_i\}$ be the set of terminal nodes of an existing binary tree. The tree uses *allowable* binary splitting functions to land a data point at one of the terminal nodes at which it is classified as $\omega(t_i)$. The allowable splitting functions are usually partitioning a single feature into two ranges. Let $P(t_i \mid \omega_j)$ be the probability that a sample from class ω_j lands at t_i. With other usual terms (defined in Section 2), the risk of misclassification with the tree is

$$\sum_{t_i} \sum_{j=1}^{M} \lambda_{jk} P(\omega_j) p(t_i \mid \omega_j) \tag{4.4}$$

where $\omega(t_i)=\omega_k$, decided by minimizing the node risk. The outer summation is carried over all terminal nodes t_i.

If a node t_i is now further split by an allowable function, to yield left and right nodes t_l and t_r respectively, the total risk will reduce by

$$\min_k \sum_{j=1}^{M} \lambda_{jk} P(\omega_j) P(t_l \mid \omega_j) + \min_k \sum_{j=1}^{M} \lambda_{jk} P(\omega_j) P(t_r \mid \omega_j)$$

A method for dynamic segmentation is to start from the entire data set and choose the splitting function which maximizes the improvement. Given a split, note that $P(t_l \mid \omega_j)$ and $P(t_r \mid \omega_j)$ are very easy to compute from labeled samples. It is possible that at some stage, the available splits do not yield any improvement at all. But on further splitting any chosen split, a good tree might be obtained. Therefore, Breiman and Stone [7] suggest an alternate splitting criterion as follows. A diversity of a terminal node t_i is defined as

$$\beta(t_i) = \sum_j P(\omega_j) P(t_i \mid \omega_j) \sum_k P(\omega_k \mid t_i)(1-P(\omega_k \mid t_i)) \tag{4.5}$$

where $P(t_i \mid \omega_j)$ is the probability that a sample from class ω_j reaches the terminal node t_i, and $P(\omega_k \mid t_i)$ is the probability that a sample belongs to ω_k given that it arrived at the terminal node t_i. Eqn. (4.5) represents the

expected error rate when a sample arriving at a terminal node is *randomly* decided based on the *a posteriori* probabilities (rather than deterministically deciding based on the *maximum a posteriori probability).* Even if the best class decisions for the subgroups of a split are the same, the differing probabilities of all the classes in the subgroups let the improvement in diversity be different for different splits. With any split, the error rate will improve to

$$\beta(t_i) - \beta(t_l) - \beta(t_r)$$

where t_l and t_r are the left and right successors induced on t_i by any candidate split. The split which gives the maximum improvement is chosen.

The procedure can be continued till all the training samples get correctly classified. Since this is not desirable, a criterion for terminating splitting is necessary. The suggested way to terminate splitting is to minimize a linear combination of risk and the number of terminal nodes in a tree. In [7], a simple sufficient condition is given for optimal termination.

5 Performance Evaluation

Evaluation of error rates, risk of misclassification, total expected cost, or computational complexity per decision of multistage schemes is much more difficult than the performance evaluation of single stage classifiers. If the tree is designed from a limited set of samples, the only reasonable methods are the design set, the test set, and the leave n out estimation of performance. For the minimum cost decision tree with discrete features, a byproduct of the design procedure is the overall expected cost at the root node. But again, if the probability mass functions used are estimated from a design set, the computed cost will be the same as the average cost determined by running the classifier with the design set samples.

Ichino [33] analyzes independent subrecognition schemes to arrive at bounds on performance. He shows that when there are $M(M-1)/2$ pairwise subrecognition schemes for the M class problem, minimizing the subsystem error probabilities will minimize an upper bound on the sum of the total error and rejection probabilities. He concludes that it is possible to design

such a system in which the error probability is less than the corresponding Bayes error probability. However, this is at an expense of higher rejection probability. He also notes the possibility of designing a system with a given tolerable error rate, by trading off with rejection rate.

Nadler [51] studies a model in which several types of classifiers operate in a sequence. At each level, there are M classifiers, one for each class. Successive levels consider only those classes accepted and passed on from the immediate preceding level. A pattern sample is rejected if no class under consideration is accepted at some level. On exhausting all available levels of classification, if more than one class is present, such a sample is also rejected. Nadler proposes such a scheme for a possible combination of different types of classifiers like the statistical and the syntactic type. Defining the various probabilities of error, empty decision, and multiple decision, he derives overall expressions for reject and error rates when at each level, rates of mistakes of both kinds are small. Roughly, the error rate is the product of individual error rates, while the reject rate is the sum of individual reject rates. This offers individual control of overall rates.

The analysis of performance of search strategies is very complicated since they expand nodes also along paths not leading to the finally accepted goal. In general, if direct decision rules rather than search strategies are used, one can derive the overall performance by appropriately averaging the terminal node performances. Given any decision tree with terminal nodes t_i having class decisions $\omega(t_i)$ and samples reaching them after feature observation belonging to a region $\{y(t_i)\}$, the overall probability of correct recognition is

$$P_c = \sum_{t_i} P(\omega(t_i), \{y(t_i)\}) \tag{5.1}$$

Note that $y(t_i)$ takes values over a subregion in a feature space corresponding to the measured features along the path to the terminal node t_i. Also,

$$\bigcup_{t_i} \{y(t_i)\} = \{x\}, \tag{5.2}$$

the entire feature space, since every sample pattern yielding any

observation in the total feature space does reach some terminal node. Eqn. (5.1) can be expanded as

$$P_c = \sum_{t_i} P(\omega(t_i)) \int_{\{y(t_i)\}} p(y(t_i) \mid \omega(t_i)) dy(t_i) \tag{5.3}$$

If we are interested in the expectation of the total cost (including classification and measurement cost), it is given by

$$E = \sum_{t_i} \sum_{j=1}^{M} \lambda_{jk} P(\omega_j, \{y(t_i)\}) + C(t_i) P(\{y(t_i)\})$$

$$= \sum_{t_i} \{ \sum_{j=1}^{M} \lambda_{jk} \int_{\{y(t_i)\}} p(y(t_i) \mid \omega_j) P(\omega_j) dy(t_i) + C(t_i) \int_{\{y(t_i)\}} p(y(t_i)) dy(t_i) \} \tag{5.4}$$

where k is given by $\omega_k = \omega(t_i)$. If error rates are low, probability of $\{y(t_i)\}$ is the same as that of the joint event $\{y(t_i)\}$ and the corresponding decision. Therefore,

$$\int_{\{y(t_i)\}} p(y(t_i)) dy(t_i) = \int_{\{y(t_i)\}} p(y(t_i), \omega_j) dy(t_i) \tag{5.5}$$

Substituting this in eqn. (5.4),

$$E = \sum_{j=1}^{M} P(\omega_j) \sum_{t_i} (\lambda_{jk} + C(t_i)) \int_{\{y(t_i)\}} p(y(t_i) \mid \omega_j) dy(t_i) \tag{5.6}$$

When we are interested in error rates only, even if the features are class conditionally statistically independent, the integral in eqn. (5.3) cannot be split into a product of integrals, as the limits of one of them will depend on others due to the complex boundary of $\{y(t_i)\}$. However, if along the path

to t_i, node decisions are independent (i.e., decisions taken on only the currently observed single variables), the integral can be evaluated as a product of independent integrals. Each integral over the variable at the corresponding node is just the probability of going down to the node leading to $\omega(t_i)$. Thus, for a hierarchical classifier with a class label appearing only at one of the terminal nodes, we have

$$P_c = \sum_{t_i} P(\omega(t_i)) \prod_{n_i} P_c(n_i) \tag{5.7}$$

where n_i are the nodes along the path to t_i and $P_c(n_i)$ are the probabilities of correct decisions at intermediate nodes. Kulkarni [41] analyzes such a scheme further and shows that for balanced binary trees, there is an optimum number of quantization levels. For a fixed sample size and tree depth, the mean accuracy (averaged over all classifiers) first increases with quantization levels and then decreases. Furthermore, the optimum number of levels increases with sample size for a fixed tree depth (the number of nodes up to a terminal node) and is a nondecreasing function of the tree depth for a fixed training sample size.

6 Discussion

Multistage classification frameworks and design methodologies described so far are not completely exhaustive. But even these generally tend to confuse the designer when it is necessary to choose one for a particular problem. The model relevant to the problem generally depends on the relative importance of three factors: the computational complexity of design; the complexity of operation; and the resulting performance. If only measurement and classification costs are important, and if they are not related to the complexity of computation, the minimum cost scheme (which might have to be designed by numerical methods) is suggested. If the number of features is large, it is better to select a subset of features with which to design the tree. The problem is not as straightforward if the design data is in the form of a small set of labeled samples. A reasonable procedure then is to discretize the features by any of the methods. With discrete features, a one step look ahead procedure suggested by Slagle and

Lee [59] is preferred over the minimum cost scheme if the feature set is large. In this method, the risk of nodes to be expanded is evaluated assuming that the next step is the last stage. The node with the least evaluated risk is chosen for further expansion.

Usually, it is only in medical diagnosis that an optimum balance of measurement and classification costs is very important. For other problems, computational costs are also important and it is very difficult to model these in the classification process. Thus other methods like the phased approach and data splitting are used. In remote sensing, often only groups of classes are required to be classified. For example, it may be sufficient to distinguish only between crop and forest in some applications. If further classification like the type of crop is required, the decision process may be continued in the decision tree. This also yields a natural hierarchy of classes so that hierarchical classifiers are more useful than others.

With small design sets, dynamic data splitting methods are simple and computationally feasible. It is also possible to design several trees and select the best among them. One reason why such schemes are useful (even when feature measurements do not involve costs) is that the clouds of data points viewed in high dimensional feature space are almost always sparse and reasonably separable. Attempts at finding good splits in several projected spaces are easily carried out on the computer.

Search strategies are applicable when a state space model is available or can be designed. Even if a good tree skeleton with class labels and features at its nodes can be arrived at by *a priori* problem knowledge or by data analysis and clustering, it may still be very complex to specify *direct decision rules* at any nonterminal node, which will be a function of all observations up to that node. Search strategies are useful in that case. One problem with these strategies is that features need to be discretized and the histogram approach for this will force several of these discrete feature values to occur with zero probabilities. This would require the expansion of a large number of nodes, forcing the search strategy to be inefficient. Dalton [11] proposes that rather than using the raw histograms to estimate the probabilities of the discretized regions, estimated Gaussian densities be used to compute these probabilities. This will force nonzero risks and help in reducing the number of nodes to be expanded.

When the sum of measurement cost and classification risk is required to be minimized, all the classes should be under consideration at every stage [18]. This is true under the assumption that the computational complexity

required for considering all the classes does not enter the cost to be minimized. However, on designing the minimum cost decision structure, it is still possible that only a proper subset of classes occur as terminal nodes at some stage beyond a node under consideration. Therefore it may be computationally attractive to reject the classes which would never be accepted beyond a nonterminal node. But to arrive at these classes to be rejected at any stage is not easy. This is where the *a priori* problem knowledge and data analysis like clustering help in designing tree skeletons. Thus, what is common to all multistage classification schemes is that nonterminal nodes have feature subsets and class subsets assigned to them. Terminal nodes simply accept a class label. The course of action to be taken at any nonterminal node could depend on the new information(current measurement) only or on current as well as past measurements. In the former case the decision rules will be simple.

With several independent methodologies available for designing decision trees, a pertinent question is "can a decision tree designed by any method be adaptively tuned to improve its performance?" Dattatreya and Kanal [16], [17] study this problem and develop a scheme towards this end. If *a priori* and class conditional statistics are available, the expected cost a sample incurs after classification, together with its *a posteriori* class probabilities can be used to modify the decision schemes at all the nodes along its path. This method also provides an alternate design model. A very desirable feature of this model is that it views all the descendant nodes of an intermediate node as a set of available decisions. A cost matrix can be assigned for an intermediate node. The updated class probabilities at that node together with the cost matrix provide a linear decision making mechanism. The elements of the cost matrix are very complicated functions of the decision tree skeleton and the problem statistics. However, these elements can be estimated (with assured convergence) either by simulation or by an adaptive procedure using unlabeled samples.

7 Applications

Decision trees have been used for a variety of practical pattern recognition problems. Predominant among these are medical diagnosis and classification of remotely sensed multispectral data. Other applications are

reported in speech processing, radar profile classification, and identification with chemical spectra. Sethi and Chatterjee [57] and Casey and Nagy [8] report good results on their decision trees developed specifically for optical character recognition.

Wu et al. [63], Swain and Hauska [61], You and Fu [65], and Argentiero et al. [2] utilize decision trees for classification of remotely sensed data. An optimization procedure is developed in [63] and [61] using Gaussian maximum likelihood node decisions. The authors also note the possibility of manual design for small problems. The requirement of merging remote sensing information with collateral data, the possibility of occurrence of singular overall covariance matrices, missing data etc. render the tree approach very attractive. In [65], an error bound is set and a fixed size feature subset is selected for each node. The problem considers ten classes and twelve spectral band features. An accuracy of 84% with three features at each tree node is reported as against 94.5% for the single stage (twelve features) maximum likelihood classifier.

Anderson and Fu [1], and Mui and Fu [50] examine decision tree application for blood cell classification. In [1], a binary tree structure representing the natural class relationships is obtained by clustering. Feature subsets are then allocated to different nodes based on the accuracy obtained with Fisher's [21] linear classifier. With the resubstitution method, accuracies of 80.5% and 75% are quoted in [1] and [50] respectively. Landweerd et al. [44] examine several statistical methods for representation and recognition of normal white blood blood cells. Through a sequence of procedures including feature selection by Student's T-test, ranking by covariance analysis, and agglomerative unsupervised clustering, they design a binary tree. Landweerd et al. [44], [45] also analyze monocyte blood cells to classify them as normal, Hodgkin's disease, or non Hodgkin Lymphoma. Parameters from the geometry and the densitogram of the blood cells are used as features. Using the interactive system ISPAHAN [26], they develop a hierarchical structure of the data set useful for designing decision trees. Bernard [5] experiments with several one step and hierarchical classification schemes for leukocyte classification and concludes that whereas the accuracies are always within about 4% of each other, hierarchical methods help in reducing the number of features used. Xiong et al. [64] conduct experiments on hierarchical classification of leukocytes and report similar results. Zahniser et al. [66] design decision trees in a prescreening system for cervical smears. The criterion is to allow a low false alarm rate while

maintaining a reasonably high detection rate for abnormal cells. Cut off thresholds at each node are chosen by comparing frequency distributions of normal and abnormal cells that reach the node under consideration.

Meisel [46] applies the optimization technique developed by his group ([47], [52]) to some medical diagnosis problems and notes the importance of human directed and data directed development of decision trees. Peters [54] designs decision trees for discrete feature problems in the classification of hematologic diseases. The framework is the minimum cost scheme with measurement and classification costs.

In speech processing, Dante [12] and others ([13] and [14]) apply multistage classification techniques for speaker recognition in large populations. Dattatreya [15] and others ([18], [19], and [20]) develop decision trees for voiced, unvoiced, and silence classification of speech segments and for vowel recognition. The model uses computational complexity as a measure of cost of extracting features from the raw data and arrives at decision trees which turn out to be very simple to operate and yield acceptable accuracies. Siegel and Bessey [58] refine the initial classification of voiced, unvoiced excitation of speech to include the mixed type in the second stage. Ichino [33] uses independent subrecognition schemes for vowel classification using the data from a channel vocoder (twenty six channels).

Breiman et al. [6] use dynamic data segmentation for several problems like radar range profile classification to identify ship types, prediction of pollution levels, and identification of certain elements in chemical compounds using mass spectra. He notes that the schemes are powerful in using conditional information for feature selection, easy for nonstatisticians to interpret, and simple to use. But due to small sample set effects, they tend to give biased estimates. Rounds [43] applies the scheme developed using the Kolmogorov-Smirnov criterion to split the data, for a two class radar target recognition.

The AID [22] and THAID [49] algorithms are developed mainly for modeling nonlinear regression and prediction problems. A typical example is the study of patterns of housing flats as a function of variables like daylight standards, location, plan types etc. It is also possible to fit certain pattern recognition problems into these frameworks by treating the class set as the range of a nominal dependent variable.

8 Conclusion

Decision trees are useful in pattern recognition for several reasons. Important among these are the following: (i) Computational efficiency provided by the multistage methods, (ii) The possibility of obtaining a class decision with a fewer number of features, thus reducing the average cost of feature measurement, (iii) Inadequate design set size posing difficulties in the estimation of multivariate densities for single stage recognition, (iv) Reduction of storage requirements for class prototypes (templates) or statistics, and (v) the ability to make use of features differently for different subgroups of classes.

Very often, available data for designing decision schemes is not sufficient for global modeling. The designer then resorts to approaches free from restrictions of full specifications of a model for multiple features and classes, and conducts an examination of the data in a formal though flexible way. This article has attempted to consolidate several of these methodologies to bring out their similarities and differences. Details of individual methods are not given to avoid undue digression and to limit the size of the article. The most important conclusion, perhaps, is that there is no universal method of designing trees for all problems. But a comparative study of the methods given here is expected to aid the designer in selecting an existing technique or in developing a suitable one.

References

[1] A.C. Anderson and K.S. Fu, "Design and development of a linear binary tree classifier for leukocytes," Purdue Univ. Tech. Rep. TR-EE 79-31, Aug. 1979.

[2] P. Argentiero, R. Chin, and P. Beaudet, "An automated approach to the design of decision tree classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. PAMI-4, pp. 51-57, Jan. 1982.

[3] A.J. Bayes, "A dynamic programming algorithm to optimize decision table code," *Austral. Comput. J.,* vol. 5, No. 2, pp. 77-79, 1973.

[4] D.A. Bell, "Decision trees, tables, and lattices," In B.G. Batchelor, Ed., *Pattern Recognition: Ideas in Practice.* New York: Plenum Press, 1978, Chap. 5, pp. 119-141.

[5] E.D.A.S. Bernard, "On the tree structure design and hierarchical classifiers," Tech. Rep., Information Theory Group, Dept. of Elec. Engg., Delft Univ. of Technology, Netherlands, Nov. 1980.

[6] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, *Classification and Regression Trees.* Belmont, CA: Wadsworth, 1984.

[7] L. Breiman and C.J. Stone, "Parsimonious binary classification trees," Technology Service Corporation, Santa Monica, Calif. Tech. Rep. TSC-CSD-TN-004, July 1978.

[8] R.G. Casey and G. Nagy, "Decision tree design using a probabilistic model," *IEEE Trans. Inform Th.,* vol. IT-30, pp. 93-99, Jan. 1984.

[9] E. Cerny, D. Mange, E. Sanchez, "Synthesis of minimal binary decision trees," *IEEE Trans. Comput.,* vol. C-28, pp. 472-482, July 1979.

[10] D. Comer and R. Sethi, "The trie index construction," *J. ACM,* vol. 24, pp. 428-440, July 1977.

[11] R.E. Dalton, "S-admissible search of hierarchical classifiers with continuous feature distributions," *Pattern Recognition Letters,* vol. 2, pp.213-217, May 1983.

[12] H.M. Dante, "Multistage pattern recognition schemes for automatic speaker identification and verification," Ph.D. Thesis, Dept. of Elec. Comm. Engg., Indian Institute of Science, Bangalore, India, May 1979.

[13] H.M. Dante and V.V.S. Sarma, "Automatic speaker identification for a large population," *IEEE Trans. Acoust. Speech. Signal Processing,* vol. ASSP-27, pp. 255-263, June 1979.

[14] H.M. Dante, V.V.S. Sarma, and G.R. Dattatreya, "Multistage decision schemes for speaker recognition," Proc. IEEE Int. Conf. Acoust. Speech, Signal Processing, Washington, D.C., Apr. 1979, pp. 797-800.

[15] G.R. Dattatreya, "Pattern recognition schemes involving feature measurement and random classification costs," Ph.D. Thesis, School of Automation, Indian Institute of Science, Bangalore, India, Nov. 1980.

[16] G.R. Dattatreya and L.N. Kanal, "Adaptive pattern classification with random costs and its application to decision trees," Dept. of Computer Science TR-1348, University of Maryland, 1984.

[17] G.R. Dattatreya and L.N. Kanal, "Adaptive improvement of pattern recognition trees," Proc. IEEE Int. Conf. Syst., Man, Cybern., Bombay and New Delhi, India, Dec. 1983 and Jan. 1984, pp. 393-397.

[18] G.R. Dattatreya and V.V.S. Sarma, "Bayesian and decision tree approaches for pattern recognition including feature measurement costs," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. PAMI-3, pp. 293-298, May 1981.

[19] G.R. Dattatreya and V.V.S. Sarma, "Decision tree design for pattern recognition including feature measurement cost," Proc. Fifth Int. Conf. Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 1212-1214.

[20] G.R. Dattatreya and V.V.S. Sarma, "Decision tree design and applications in speech processing," *IEE Proceedings-F, Communications, Radar, and Signal Processing,* vol. 131, pp. 146-152, Apr. 1984.

[21] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis.* New York: John Wiley, 1973.

[22] A. Fielding, "Binary segmentation: The automatic interaction detector and related techniques for exploring data structures," In C.A. O'Muircheartaigh and C. Payne, Eds., *The Analysis of Survey Data, vol.1: Exploring Data Structures.* Chichester: John Wiley, 1977, Chapter 8, pp. 221-257.

[23] J.H. Friedman, "A recursive partitioning decision rule for nonparametric classification," *IEEE Trans. Comput.* vol. C-26, pp. 404-408, Apr. 1977.

[24] J.H. Friedman and W. Stuetzle, "Projection pursuit methods for data

analysis," In R.L. Launer and A.F. Siegel, Eds., *Modern Data Analysis.* NY: Academic Press, 1982, pp. 123-147.

[25] K.S. Fu, *Sequential Methods in Pattern Recognition and Machine Learning.* New York: Academic Press, 1968.

[26] E.S. Gelsema, "ISPAHAN; An interactive system for pattern analysis: Structure and capabilities," In E.S. Gelsema and L.N. Kanal, Eds., *Pattern Recognition in Practice.* Amsterdam: North Holland, 1980, pp. 481-491.

[27] D.E. Gustafson, S. Gelfand, and S.K. Mitter, "A nonparametric multiclass partitioning method for classification," Proc. Fifth Int. Conf. Pattern Recognition, Miami Beach, Florida, Dec. 1980, pp. 654-659.

[28] P.E. Hart, N.J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.,* vol. SSC-4, pp. 100-107, July 1968.

[29] C.R.P. Hartman, P.K. Varshney, K.G. Mehrotra, and C.L. Gerberich, "Application of information theory to the construction of efficient decision trees," *IEEE Trans. Inform. Th.,* vol. IT-28, pp. 565-577, July 1982.

[30] E.G. Henrichon and K.S. Fu, "A nonparametric partitioning procedure for pattern classification," *IEEE Trans. Comput.,* vol. C-18, pp. 614-624, July 1969.

[31] G.F. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Trans. Inform. Th.,* vol. IT-14, pp. 55-63, Jan. 1968.

[32] L. Hyafil and R.L. Rivest, "Constructing optimal binary decision trees is NP-complete," *Information Processing Letters,* vol. 5, pp. 15-17, May 1976.

[33] M. Ichino, "Multiclass pattern recognition systems based on independent subrecognition systems," *IEEE Trans. Syst. Man, Cybern.,* vol. SMC-6, pp. 256-259, Apr. 1976.

[34] M. Ichino, "A nonparametric multiclass pattern classifier," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-9, pp. 345-352, June 1979.

[35] M. Ichino, "Nonparametric feature selection method based on local interclass structure," *IEEE Trans. Syst. Man, Cybern.* vol. SMC-11, pp. 289-296, Apr. 1981.

[36] L.N. Kanal, "Interactive pattern analysis and classification systems: A survey and commentary," *Proc. IEEE,* vol. 60, pp. 1200-1215, Oct. 1972.

[37] L. Kanal, "Patterns in pattern recognition: 1968-1974," *IEEE Trans. Inform. Th.,* vol.IT-20, pp. 697-721, Nov. 1974.

[38] L.N. Kanal, "On hierarchical classification theory and interactive design," In P.R. Krishnaiah, Ed., *Applications of Statistics.* North Holland, 1977, pp. 301-322.

[39] L.N. Kanal, "Problem-solving models and search strategies for pattern recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. PAMI-1, pp. 193-201, Apr.1979.

[40] A.V. Kulkarni, "Optimal and heuristic synthesis of hierarchical classifiers," Ph.D dissertation, Univ. Maryland Comput Sci. Tech. Rep. TR-469, 1976.

[41] A.V. Kulkarni, "On the mean accuracy of hierarchical classifiers," *IEEE Trans. Comput.,* vol. C-27, pp. 771-776, Aug. 1978.

[42] A.V. Kulkarni and L.N. Kanal, "An optimization approach to hierarchical classifier design," Proc. Third Int. Joint Conf. Pattern Recognition, San Diego, Calif. 1976, pp.459-466.

[43] A.V. Kulkarni and L.N. Kanal, "Admissible search strategies for parametric and nonparametric hierarchical classifiers," In Proc. Fourth Int. Joint Conf. Pattern Recognition, Kyoto, 1978.

[44] G.H. Landweerd, M. Bins, and E.S. Gelsema, "Interactive pattern recognition of white blood cells using ISPAHAN," In E.S. Gelsema and L.N. Kanal, Eds., *Pattern Recognition in Practice.* Amsterdam: North Holland, 1980, pp. 493-503.

[45] G.H. Landweerd, E.S. Gelsema, M. Bins, and M.R. Hale, "Interactive

pattern recognition of white blood cells in malignant lymphomas," *Pattern Recognition,* vol. 14, pp. 239-244, 1981.

[46] W.S. Meisel, "Design of decision structures in medical diagnosis," In Proc. Int. Conf. Cybern. Society, Calif., 1975, pp. 469-472.

[47] W.S. Meisel and D.A. Michalopoulos, "A partitioning algorithm with applications in pattern recognition and optimization of decision trees," *IEEE Trans. Comput.,* vol. C-22, pp. 93-103, Jan. 1973.

[48] B.M.E. Moret, "Decision trees and diagrams," *ACM Computing Surveys,* vol.14, pp. 593-623, Dec. 1982.

[49] J.N. Morgan and R.C. Messenger, *THAID, A Sequential Analysis Program for the Analysis of Nominal Scale Dependent Variables.* Ann Arbor: Institute for Social Research, University of Michigan, 1973.

[50] J.K. Mui and K.S. Fu, "Automated classification of nucleated blood cells using a binary tree classifier," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. PAMI-2, pp. 429-443, Sept. 1980.

[51] M. Nadler, "Error and reject rates in a hierarchical pattern recognizer," *IEEE Trans. Comput.,* vol. C-20, pp. 1598-1601, Dec. 1971.

[52] H.J. Payne and W.S. Meisel, "An algorithm for constructing optimal binary decision trees," *IEEE Trans. Comput.,* vol. C-26, pp. 905-916, Sept. 1977.

[53] R.W. Payne and D.A. Preece, "Identification keys and diagnostic tables," *J. Royal Statistical Soc. A,* vol. 143, Part 3, pp. 253-292, 1980.

[54] R.J. Peters, "Zero order and nonzero order decision rules in medical diagnosis," *IBM J. Res. Develop.,* vol. 21, pp. 449-460, Sept. 1977.

[55] E.M. Rounds, "A combined nonparametric approach to feature selection and binary decision tree design," *Pattern Recognition,* vol. 12, pp. 313-317, 1980.

[56] H. Schumacher and K.C. Sevick, "The synthetic approach to decision

table conversion," *Comm. ACM,* vol. 19, pp. 343-351, June 1976.

[57] I.K. Sethi and B. Chatterjee, "Efficient decision tree design for discrete variable pattern recognition problems," *Pattern Recognition,* vol. 9, pp. 197-206,1977.

[58] L.J. Siegel and A.C. Bessey, "Voiced/unvoiced/mixed excitation classification of speech," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-30, pp. 451-460, June 1979.

[59] J.R. Slagle and R.C.T. Lee, "Application of game tree searching techniques to sequential pattern recognition," *Comm. ACM,* vol. 14, pp. 103-110, Feb. 1971.

[60] J.C. Stoffel, "A classifier design technique for discrete variable pattern recognition problems," *IEEE Trans. Comput.,* vol. C-23, pp. 428-441, Apr. 1974.

[61] P.H. Swain and H. Hauska, "The decision tree classifier: Design and potential," *IEEE Trans. Geoscience Electron.,* vol. GE-15, pp. 142-147, July 1977.

[62] A. Wald, *Statistical Decision Functions.* New York: John Wiley, 1950.

[63] C. Wu, D. Landgrebe, and P. Swain, "The decision tree approach to classification," Tech. Rep. TR-EE 75-17, Purdue Univ., 1975.

[64] F.-L. Xiong, G.R. Dattatreya, L.N. Kanal, "Hierarchical methods for leukocyte classification," Proc. of IEEE Int. Conf. Syst., Man, Cybern., Bombay and New Delhi, India, Dec. 1983 and Jan. 1984, pp. 403-407.

[65] K.C. You and K.S. Fu, "An approach to the design of a linear binary tree classifier," Proc. Symp. Machine Processing of Remotely Sensed Data, Purdue Univ., 1976, pp. 3A-1 to 3A-10.

[66] D.J. Zahniser, P.S. Oud, M.C.T. Raaijmakers, G.P. Vooys, and R.T. Van de Walle, "Decision tree optimization in a prescreening system for cervical smears," In E.S. Gelsema and L.N. Kanal, Eds., *Pattern Recognition in Practice.* Amsterdam: North Holland, 1980, pp. 453-462.

Progress in Pattern Recognition 2
L.N. Kanal and A. Rosenfeld (Editors)

A NEW APPROACH TO PATTERN RECOGNITION

Lev Goldfarb

School of Computer Science
University of New Brunswick
Fredericton, N.B.
Canada E3A 5A3

"Notions of quantity are possible only when there already exists a general concept which admits particular instances. These instances form either a continuous or a discrete manifold, depending on whether or not a continuous transition of instances can be found between any two of them; individual instances are called points in the first case and elements of the manifold in the second. Concepts whose particular instances form a discrete manifold are so numerous that some concept can always be found, at least in the more highly developed languages, under which any given collection of things can be comprehended (and consequently, in the study of discrete quantities, mathematicians could unhesitatingly proceed from the principle that given objects are to be regarded as all of one kind). On the other hand, opportunities for creating concepts whose instances form a continuous manifold occur so seldom in everyday life that colour and the position of sensible objects are perhaps the only simple concepts whose instances form a multiple extended manifold. More frequent opportunities for creating and developing these concepts first occur in higher mathematics."

"... in a discrete manifold the principle of metric relations is already contained in the concept of the manifold, but in a continuous one it must come from something else. Therefore, either the reality underlying Space must form a discrete manifold, or the basis for the metric relations must be sought outside it, in binding forces acting upon it."

"Thus arises the problem of seeking out the simplest data from which the metric relations of Space can be determined, a problem which by its very nature is not completely determined, for there may be several systems of simple data which suffice to determine the metric relations of Space... ."

G.B. Riemann, "On the Hypotheses which lie at the Foundations of Geometry"

FOREWORD

It is well known that the principal reason for the distinction between discrete and continuous distributions lies in the difference between mathematical tools used for the relevant estimations. In the event of completely unknown distributions (as in most of the cases in pattern recognition) the above distinction is not of great help. On the other hand, the more fundamental notion of a random vector plays a unifying and simplifying role, providing us with a common and analytically convenient space (Euclidean space $\mathbb{R}^n$) in which to work.

However, during the last several decades researchers in the frontiers of different fields of data analysis (particularly in pattern recognition) began to move away from this "convenient" representation space to the more abstract representation "spaces". The availability of the appropriate representational structures (graphs, grammars, etc.) facilitated these transitions. As a result, one gains a considerably greater representational freedom, but loses *the universality and convenience* of the vector space. "The convenience" should be understood in the practical sense: the other "spaces" do not have the analytical tools available in a vector space, and, therefore, decision-theoretic algorithms in such "spaces" are more primitive and more costly (in computer terms). Still, the transitions were justified on the following ground: in many cases, the gains resulting from the possibility of reflecting in pattern representation different "structural" features of the data seem to outweigh the convenience of the universal representation space, particularly in the face of absence of any information about the underlying distributions. Thus, the "structural" approach in pattern recognition appeared.

Is there a way to unify the "two approaches" to pattern recognition? This work is an attempt to answer this question, which has been raised many times during the last two decades. The monograph is based on my Ph.D. thesis (Department of Systems Design, University of Waterloo, Canada). The main problem it is addressed to is the development of a framework for a domain-independent theory for pattern recognition and data analysis in general, i.e., the principal motivation of the proposed approach is to develop a rigorous analytical basis for making decisions which depend only on a finite set of collected patterns independent of the chosen form of the pattern representation.

Since no restrictions on the form of data representation are imposed, all the advantages of the "structural" approach are automatically incorporated in the proposed approach. The difference between the two approaches is that the latter, unlike the former, constructs a vector representation of the patterns, and thus allows one to use the machinery available in a vector space for decision algorithms. The practical advantages are obvious: for example, linear and non-linear decision surfaces can be used for supervised classification of "syntactic" patterns, or different scattering criteria can be used for clustering of "structural" patterns, and, of course, many efficient nearest neighbour decision algorithms for vector patterns [24] can now be applied to any "syntactic" patterns.

The proposed approach reduces to the classical "geometric" approach if the input data are a finite set of Euclidean vectors with the Euclidean distance. In view of both the immediate advantages and the similarity with the classical techniques, I was at this *initial stage* more concerned with the development of the basic theoretical tools necessary for applications, rather than with testing the approach on concrete data. I also firmly believe that "the final criterion for deciding whether a model is 'good' is whether it yields *useful* information", and "our motivation in considering mathematical models lies in their utility rather than in any notion of 'correctness'." Consequently, "judging models in terms of their utility gives rise to the possibility of using several different models for the same phenomenon. This is not an uncommon situation in science" ([56], p. 2). The reader is advised to keep these points in mind throughout the entire work. In fact, one of the main driving motivations for the development of the proposed approach to pattern recognition came from the realization of a simple fact that many *useful, concrete (numerical) characteristics of a class of complex patterns obtained within the proposed framework cannot be obtained by any other means.*

It turns out that to "accommodate" data of any generality in a vector space the class of Euclidean vector spaces is not sufficient, i.e., one has to consider the class of vector spaces with not necessarily positive "inner products". This class obviously contains all Euclidean spaces. Since the linear algebra in such spaces is by no means a standard part of any undergraduate or graduate curriculum, the largest chapter (chapter 3) is devoted to an independent exposition (with illustrations) of the necessary material.

I would like to thank my Ph.D. thesis advisor Andrew K.C. Wong for rousing interest in me to problems of pattern recognition, and for his constant encouragement, enthusiasm, and patience during the course of my work on the dissertation.

My sincere thanks to Maurice Chayet and Luc Devroye for numerous discussions, and to the editors L. Kanal and A. Rosenfeld for their encouragement. The monograph was written during the tenure of NSERC Postdoctoral Fellowship 1980-81, without which the work may not have been completed, and since then several copies have been circulated.

Preliminary notations and terminology

The Greek alphabet is used, as a rule, to denote mappings (functions).

$(a) \Longrightarrow (b)$	from (a) follows (b)
$(a) \Longleftrightarrow (b)$	(a) is equivalent to (b)
$x \in X$ $(x \notin X)$	element x belongs (does not belong) to the set X
$\forall x \in X$ $(\forall x, y \in X)$	for all x from X (for all x,y from X)
$\exists x$	there exists x
$A \stackrel{def}{=} B$	the expression on the left is defined to be equal to the expression on the right
$\{a_i\}_{1 \le i \le n}$	the set $\{a_1, \ldots, a_n\}$
$X \subseteq Y$	X is a subset of Y
$X \subset Y$	$X \subseteq Y$ and $X \neq Y$
$X/\sim$	the quotient set of X by the equivalence relation $\sim$ (the set of equivalence classes)
$\{x \in X \mid B\}$	the set of all x from X satisfying property B
$\alpha : A \rightarrow B$	a mapping from A to B, A is the domain of α, B is the codomain of α; for $A_1 \subseteq A$ the set $\alpha(A_1) = \{b \in B \mid \exists a \in A_1$ such that $b = \alpha(a)\}$ is the image of A_1 under α; the set $\alpha(A)$ is the range of α

$\alpha \circ \beta$ or $\alpha\beta$	the composition of the two mappings
$\imath_A$ or $\imath$	the identity mapping of the set A
injection	a one-to-one mapping whose range does not have to be equal to the codomain
surjection	a mapping whose range coincides with the codomain
bijection	an injection which is also a surjection
$\alpha \mid A_1$	the restriction of the mapping $\alpha: A \longrightarrow B$ to $A_1 \subseteq A$, i.e., the mapping $\alpha_1: A_1 \longrightarrow B$ such that $\alpha_1(a) = \alpha(a) \quad \forall a \in A_1$
the extension of α	if $\alpha: A \longrightarrow B$, then the extension of α from A to $C \supseteq A$ is a mapping $\beta: C \longrightarrow B$ such that $\beta \mid A = \alpha$
$X \times Y$	the cartesian (direct) product of the sets X and Y ($X \times Y \stackrel{def}{=} \{(x,y) \mid x \in X, y \in Y\}$)
$X - Y$	the difference of the two sets ($X - Y \stackrel{def}{=} \{x \in X \mid x \notin Y\}$)
$\lvert X \rvert$	the number of elements of a finite set X
$\mathbb{R}$ ($\mathbb{R}_+$)	the set of real (non-negative real) numbers
homomorphism	a linear mapping from one vector space to another
endomorphism	a linear mapping from a vector space to itself
End(V)	the set of all endomorphisms of vector space V
monomorphism	a homomorphism which is an injection
isomorphism	a homomorphism which is a bijection
$V_1 \approx V_2$	vector space V_1 is isomorphic to vector space V_2
automorphism	an endomorphism which is an isomorphism
$\text{Null}(\psi)$	if $\psi: V_1 \longrightarrow V_2$ is a homomorphism then $\text{Null}(\psi) = \{x \in V_1 \mid \psi(x) = 0\} = \text{Ker}(\psi)$
linear form on V	a homomorphism from V to $\mathbb{R}$

V^*	the dual vector space of V, i.e., the set of all linear forms on V
$\dim(V)$	dimension of vector space V
$U + W$	the sum of the two subspaces of a vector space
$(a_i)_{1\le i\le n}$ or simply (a_i)	a basis of a vector space
$(x^i) \in \mathbb{R}^n$	an n-tuple of real numbers
$U \oplus W$	the direct sum of the two subspaces of a vector space, i.e., a sum for which $U \cap W = \{0\}$
$\bigoplus_{i\in I} U_i$ (or simply $\bigoplus_i U_i$)	the direct sum of the subspaces U_i, where $i\in I$
$\langle a_1,\ldots,a_n\rangle$	the subspace generated (spanned) by the vectors a_i $(1\le i\le n)$
$\langle A\rangle$	the subspace generated by the set A
complementary subspace	if $V = U \oplus W$, then V and W are mutually complementary subspaces of V
I_n (or simply I)	the identity matrix of size $n\times n$
${}^tA$	the transpose of the matrix A
W^0	the annihilator of the subspace W of V, i.e., the set $\{\psi\in V^* \mid \psi(x) = 0\ \forall x\in V\}$
linear variety or affine subspace	the subset of a vector space V of the form $x + W$, where x is a fixed vector and W is a subspace of V
$V(c;\gamma)$	the characteristic subspace of $\gamma: V \longrightarrow V$ corresponding to the characteristic value c of γ; i.e., the subspace $\{x\in V \mid \gamma(x) = cx\}$
$(a\,\vert\,b)$	the Euclidean inner product of vectors a and b

TABLE OF CONTENTS

1 INTRODUCTION

Statistical theory has long been a major tool for general data analysis. Partly because of this and partly because of the tradition which goes back to the last century (and which in its own turn has conditioned the particular development of statistical theory), it has become a common practice to *represent data under investigation in a Euclidean vector space* of n dimensions. The strength of this tradition in pattern recognition can be seen in some recent theoretical surveys on clustering ([1], p. 19; [2], p. 47). Of course, other forms of representation also exist. In fact, with the development of set theory, abstract algebra, and graph theory the arsenal of representation tools has been considerably enlarged (this does not mean that they are fully utilized in the various fields of data analysis). However, besides tradition, the main reason behind a bias toward representing data in $\mathbb{R}^n$, in the author's opinion, lies in the indisputable fact that more powerful classical *analytical tools* for data analysis are not available for other forms of representation.

Traditionally it has been assumed in data analysis (and pattern recognition) that *any* measurable "variables" can be represented by "the axes" in a Euclidean space of appropriate dimension, i.e., in the vector space in which the distance is calculated by the well known generalized Pythagorian formula: $d(x,y) = \sqrt{(x^1-y^1)^2 + (x^2-y^2)^2 + \cdots + (x^n-y^n)^2}$. It should be realized, however, that this model of the "variables", because it imposes the Euclidean distance, may (and, as we shall see, in most cases will) considerably distort the information contained in the original data. The realization of this fact was fundamental to the development of the special relativity theory, where the Euclidean distance in the four-dimensional space representing space and time is replaced by the Minkowski distance, in order to arrive at a better mathematical model for space-time. In other words, the moral of this physical theory for us is: do not assume that the vector space generated by *any* "variables" is always Euclidean.

On the other hand, during the last twenty years the importance of introducing structural information in data representation has become increasingly clear. This can be supported, for example, by the emergence of "structural pattern recognition" (and "syntactic

pattern recognition" in particular) ([3], [4]). The new trends were *in no way motivated by any dissatisfaction with the mathematical tools available in* $\mathbb{R}^n$, but rather by the *representational limitations* forced on the data under the classical approach.

The two main approaches, "structural" and traditional, have not been integrated into one general framework:

"The many different mathematical techniques used to solve pattern recognition problems may be grouped into two general approaches. They are the decision-theoretic (or discriminant) approach and the syntactic (or structural) approach. In the decision-theoretic approach, a set of characteristic measurements, called features, are extracted from the patterns. Each pattern is represented by a feature vector, and the recognition of each pattern is usually made by partitioning the feature space. On the other hand, in the syntactic approach, each pattern is expressed as a composition of its components, called sub-patterns and pattern primitives. This approach draws an analogy between the structure of patterns and the syntax of a language. The recognition of each pattern is usually made by parsing the pattern structure according to a given set of syntax rules" ([5], p. 13; see also [3], p. 3).

Moreover, we have the following evidence (important for us) from a well known book by Duda and Hart:

"No single theory of pattern recognition embraces all of the important topics because each domain of application has unique characteristics that mold and shape the appropriate approach. The most prominent domain-independent theory is classification theory ... . Based on statistical decision theory, *it provides formal mathematical procedures for classifying patterns once they have been represented abstractly as vectors.*

Attempts to find domain-independent procedures for constructing these vector representations have not yielded generally useful results. Instead every problem area has acquired a collection of procedures suited to its special characteristics" ([6], p. vii, the italics are mine).

It should be stressed that "a collection of procedures" is in many cases a more or less skillful dismantling and fitting of the structural information present in the data into the n-tuple of real numbers. This dismantling and fitting has *the following consequences*: 1) the destruction of the structural information; 2) an arbitrary coalescing of the newly created (artificial) features in the vector form; 3) the necessity to go to high dimensional Euclidean spaces to

represent the data; and as a consequence of this, 4) the necessity of reducing the dimensionality, which in its own turn further contributes to the distortion of the true picture of the phenomenon and significantly complicates the analysis itself.

Before proceeding further, one question should be answered at this point: Why is it at all desirable to have a vector representation of the data being analyzed? The answer to this question is exactly the same as the answer to the following question: Why do we need the notion of a random variable? And the answer is: *the analytical convenience and the standardization of the sample space.* In fact, the analogy between the above two questions should be kept in mind throughout the entire work.

The fundamental problem is thus "... to find domain-independent procedures for constructing these vector representations" when the data are not immediately "Euclidean".

To be satisfactory the solution to the stated problem should be, first of all, general enough to be applicable to *any* data representation. An immediate question in connection with this is: What should constitute a basis of the transition from any data representation to a vector representation? I suggest that the answer to this question has already been provided by the historical development of pattern recognition itself, as well as some other fields of knowledge, which contributed to the emergence of pattern recognition, such as biological taxonomy and psychology. To be specific, the answer lies in the notion of a dissimilarity (equivalently, a pairwise similarity) measure, which is responsible for the very origin of pattern recognition: "Similarity plays a fundamental role in theories of knowledge and behaviour. It serves as an organizing principle by which individuals *classify* objects, form concepts, and make generalizations" ([7], p. 327, italics are mine).

At the same time the notion of a distance plays the fundamental role either explicitly or implicitly in all of the classical decision-theoretic algorithms; even the notion of measure, and therefore of an integral, can and should be introduced as a secondary one with respect to that of distance (when the distance is introduced at all).

A formal notion of a metric (distance) in an abstract set, that is a notion of a metric space was introduced in mathematics in 1906 by M. Frechet ([8]), and since the appearance in 1914 of Hausdorff's book ([9]) it has begun to play a fundamental and unifying role in mathematics.

Considering this, the author has adopted the point of view that

has already been accepted in mathematical taxonomy: as far as the problems of pattern recognition (or general data analysis) are concerned, the basic information about a given set of data can satisfactorily be represented by an appropriate dissimilarity measure (or measures) ([10], pp. 5-6) on the given set. By analogy with the corresponding terminology in mathematics, we shall call such a dissimilarity measure a pseudometric function, and a set together with a pseudometric function on it, a pseudometric space (see Definition 2.1). Another reason for introducing this new terminology is the desire to somehow stress a formal rather than an heuristic way of introducing a pseudometric function. More precisely, by this term it is intended to underline the mathematical way in which a pseudometric function should be introduced: instead of defining it *directly* on a given set of data (as has been practiced, for example, in biological taxonomy and psychology), one should first try to find an appropriate formal (mathematical) representation of the data, and only then, again in a formal manner, to introduce an appropriate pseudometric function (which, if necessary, could be constructed from several pseudometric functions, see Chapter 2).

In mathematics the notion of a pseudometric space has not received attention for the simple reason that whereas a metric generates topology in the metric space, a pseudometric function does not generate one in a natural way. Nevertheless, as far as pattern recognition is concerned, the purpose of the pseudometric function is to reflect as wide a spectrum of the properties of the data as possible. And I believe that the notion of a pseudometric function does give one sufficient freedom to express in a numerical form a dissimilarity between two objects. Another reason for admitting a pseudometric distance function is that although more reliable theory could be built only under the assumption of a metric distance function, more efficient solutions of some concrete problems even under this more general assumption become available within the proposed framework.

One of the most important reasons for adopting a dissimilarity measure as the basis of data analysis is the uniformity it affords. Thus, as we shall see, on this basis the syntactic approach can be integrated with the classical; some steps in this direction have already been made (see, for example, [11] - [13]). On the other hand, in no way is the notion of a distance function too general to be useful. In fact, the sole purpose of this work is to show that the natural ground for pattern recognition (and data analysis in

general) is not only the Euclidean vector space $\mathbb{R}^n$, but also the notion of an abstract distance space, formally represented by that of a metric space.

Once the discussed basis is accepted, one can proceed to the solution of the problem stated above: "... find domain-independent procedures for constructing ... vector representations". Since the adopted framework implies that the only information to be preserved is the matrix of interdistances, it is natural to understand "*vector representation*" as a *distance preserving* representation in some vector space.

Strangely enough some (and from the author's experience a considerable number) of the specialists in data analysis believe that a solution to the problem, at least in the case of a metric space, is quite simple. Thus, in the book [14], p. 54, we find:

"Once such a distance measure (metric - L.G.) has been defined, it is easy to derive a set of coordinates for each point that have the required set of $\frac{1}{2}N(N-1)$ distances (N is the number of points - L.G.), but in general it requires N-1 dimensions. A possible algorithm for doing so would be to place the first point at the origin, the second at the appropriate distance along the axis of x_1, the third in the x_1x_2 plane at the assigned distance from the other two, and so on. The addition of each new point involves the introduction of a new coordinate, and the placing of the m^{th} point involves the solution of m-1 equations in m-1 unknowns. There remains one ambiguity, which can be resolved by specifying that the $(m-1)^{th}$ coordinate of the m^{th} point shall be positive (otherwise the fourth point, for example, could be placed above or below the plane of the first three).

A rule of this sort makes it possible to transform the $\frac{1}{2}N(N-1)$ distances into a set of coordinates."

However, the situation is not quite that simple, since the outlined construction algorithm can in general be realized only for $N \leq 3$. In fact, even among *four-point metric spaces*, most cannot at all be represented in any Euclidean space. An example of such a four-point space is given below. The reason behind this fact is that m-1 equations, mentioned in the above quotation, are the equations of (m-2)-dimensional spheres, and the intersection of the m-1 (m-2)-dimensional spheres (which is the solution set) for $m \geq 4$ may quite well be empty, even though their radii satisfy the triangle inequality. To illustrate this statement consider a four-point *metric space* $\{p_1, p_2, p_3, p_4\}$ with the distance matrix

	p_1	p_2	p_3	p_4
p_1	0	1	1	0.55
p_2	1	0	1	0.55
p_3	1	1	0	0
p_4	0.55	0.55	0.55	0.55

The points p_1, p_2, p_3, which form an equilateral triangle, are easily representable in the Euclidean plane $\mathbb{R}^2$. But the fourth point p_4 cannot be added to them even in $\mathbb{R}^3$, since the three spheres with centers at p_1, p_2, p_3 and with radii equal to 0.55 have no point in common. On the Figure 1.1 the projection of the configuration onto the plane (where p_1, p_2, p_3 are represented) is sketched.

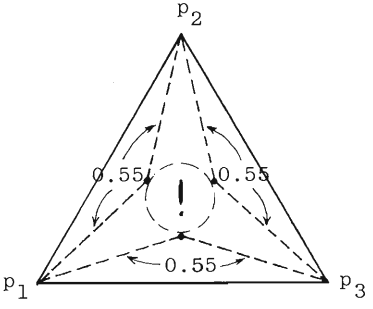


Figure 1.1

Fortunately, a simple vector representation algorithm does exist; only the class of Euclidean spaces is insufficient for the purpose, and sometimes the addition of one element to the sample may result in an increase of the dimension of the previously constructed vector representation by a larger number and not by one, as is the case when the population can be represented in a Euclidean space. But this will become clear later on (see chapter 4).

It may be useful to note here that in mathematics non-Euclidean spaces were introduced in the first half of the last century, and their theory was given particular impetus by the work of B. Riemann ([16]), which has changed the face of mathematics. On the other hand, the discoveries of Einstein in physics have permanently brought the notion of a non-Euclidean space into physics. A not so well known fact is that less than 40 years ago Dirac [17] proposed a new

method of field quantization, where an infinite-dimensional non-Euclidean space serves as the space of quantum states*). At approximately the same time Sobolev [18], studying concrete problems of dynamics, was led to consider a similar class of spaces. Non-Euclidean spaces were also suggested in the theory of binocular vision [19], as well as in psychological theory [20], where they are becoming increasingly popular.

Returning to the question of constructing "domain-independent ... vector representations", we see that it can be answered in a positive way. This construction, as was mentioned, cannot always be performed in a Euclidean vector space, necessitating the consideration of pseudoeuclidean spaces (V,Φ)'s, i.e., vector spaces V's in which the scalar products Φ's are given by any symmetric bilinear forms, and not by positive forms, as in the case of Euclidean spaces. A well known example of such a space is the Minkowski space of special relativity theory.

However, the fundamental difference between the classical and the proposed approaches to vector representation is not just the change of one space by another, but the fact that in many cases the classical approach "squeezes" the patterns in the fixed Euclidean space *without determining whether or not it is an appropriate space for pattern representation*. So that at the very beginning of the analysis, instead of finding structure in the data, one would be imposing structure on it. In other words, while the classical "vector representation" consists in lining up some numerical characteristics of *a single object*, the suggested approach is based directly on a chosen dissimilarity measure defined on *pairs of objects*. As a consequence, in the classical approach one is not preoccupied with the question of finding an appropriate formal representation of the object, since in most cases it is not clear under that approach how the chosen representation can be used to construct vector representation. This also explains why we now have "two general approaches" (see the first quotation).

The above mentioned vector representation algorithm (see chapter 4) can be implemented as the so-called QR-algorithm, which "is one of the most satisfactory in all numerical analysis" ([21], p. 108). The output from the algorithm consists of the set of coordinate vectors and the inner product on the corresponding vector

*)In the later development of the theory Pauli, Lee, Heisenberg and many others have participated.

space, where the distances between the vectors (computed with respect to the inner product) are those between the original points. The dimension of the obtained vector space is minimal, i.e., there is no vector representation (of the input set of patterns together with the distance function) in a vector space of a smaller dimension (Theorem 4.1).

Unfortunately, in reality *the measurements* (and therefore the distances) *corresponding to continuous parameters* in the data representation contain systematic and random errors. And since the vector representation algorithm reproduces the interdistances precisely (up to round-off errors), these errors may result in a "swelling" of the minimal representation space. Although this problem can be solved by the application of generalized principal component analysis to the constructed set of vectors, there are computationally more satisfactory ways of dealing with the problem (see chapter 6), i.e., the embedding algorithm can actually construct a reduced vector representation with the "noisy" dimensions removed. It is important to note that the latter techniques can also be applied in the classical situation, when one wants to obtain the reduced vector representation.

Obviously, an application of the corresponding orthogonal transformation and translation to the constructed vector representation yields another vector representation. Thus, we quite naturally obtain a group of transformations, the group of motions of (V,Φ), with respect to which the entire theory that is to be developed in the pseudoeuclidean representation space (V,Φ) should be invariant.

Most important, in a pseudoeuclidean vector space all of the classical notions related to the decision-theoretic techniques can be introduced, and thus many efficient "geometric" algorithms become applicable to "structural" data, or for that matter to any data (chapters 5-7). To see the importance of this it suffices to mention that one can use linear decision surfaces for the supervised classification of any syntactic pattern, or to use different scattering criteria ([6], section 6.8.3) for unsupervised classification of syntactic patterns. It is also very important to stress that *all these criteria become more reliable under the proposed approach than they are under the traditional.* The reason for this is that the distances between the vectors can now represent any desirable dissimilarity measure, in particular a measure inducing *well separated* images:

"Most papers give no reasons for the choice of features. In

fact most features in pattern recognition work are chosen on the grounds that the choice is intuitively reasonable. Once the features have been chosen authors apply sophisticated statistical methods in order to minimize errors. *In most cases, however, the game has been lost with the choice of features. Unless the features separate the patterns no amount of statistical analysis can untangle the errors that necessarily result when pattern images overlap*" ([22], pp. 37-38, the italics are mine; see also [23], section 4.2).

Another merit of the vector representation is that the computational efficiency of some algorithms, such as different versions of the nearest-neighbor rule, can significantly be improved in the presence of a vector space structure (see [24]).

It was only in 1973 that Duda and Hart wrote:

"We feel obliged to mention that fundamental issues in measurement theory are involved in the use of any distance or similarity function. The calculation of the similarity between two vectors always involves combining the values of their components. Yet, in many pattern recognition applications the components of the feature vector measure seemingly noncomparable quantities ... How does one treat vectors whose components have a mixture of nominal, ordinal, interval, and ratio scales? Ultimately, there is no methodological answer to these questions" ([6], pp. 216-217).

The proposed approach seems to be a good candidate for a realistic solution of the mixed-mode problem (see section 7.5). In fact, the main strength of the suggested framework is the possibility of representing data of any complexity in some low dimensional vector space where it is much easier to find and utilize the underlying analytical dependence.

When the decision about a "new" object has to be made on the basis of the information provided by a collected set of data (as in supervised classification), it is not necessary to construct the vector representation again. The coordinates of a projection of a new object on the space where the collected data have been represented are easily calculated by solving a system of n linear equation, where n is the dimension of the representation space (section 7.2). If the original distance function is appropriate for the problem, and the collected data is "sufficient", this projection of a new object, as we shall see, is also a reliable vector representation for all decision-theoretic purposes. Of course, the classical approach also becomes unreliable if one of the above two conditions are violated, although this is not as apparent in the

traditional setting.

Mathematically the suggested approach to pattern recognition can be described in the following example. Suppose that two classes from a conceptual population P generate two probability distributions (P,B,μ_1) and (P,B,μ_2). To make possible a concrete analytical description of this situation, in the traditional approach one introduces a random vector $\delta : (P,B) \longrightarrow \mathbb{R}^n$, which is usually obtained by putting together n random variables $\delta_i : (P,B) \longrightarrow \mathbb{R}$, $i = 1,2,\ldots,n$. In the proposed approach one first introduces a distance function Π on the sample space P which is supposed to "move" the two densities away from each other, and only then constructs a random vector $\alpha : ((P,\Pi),B) \longrightarrow (\mathbb{R}^n,\Phi)$ with values in the pseudoeuclidean space $(\mathbb{R}^n,\Phi)$, where Φ is an "inner product". Random vector (or vector representation) α is constructed in such a way as to preserve as much as possible the shapes of the "new" distributions obtained from the original by introducing the distance function. Since α preserves the distances between the elements of collected finite samples, the better these samples represent the corresponding distributions, the more accurate will be the images of the "new" distributions in $(\mathbb{R}^n,\Phi)$. It is not difficult to see that any random variable (and vector) in the classical approach can be obtained in this way by introducing some pseudometric distance function on the sample space. Indeed, taking, for example, an experiment "a fair coin is tossed three times", we obtain the following sample space P:

$$p_1 = TTT, \quad p_2 = TTH, \quad p_3 = THT, \quad p_4 = HTT,$$

$$p_5 = THH, \quad p_6 = HTH, \quad p_7 = HHT, \quad p_8 = HHH.$$

Let the random variable be "the number of heads". Then the reader can check easily that the finite pseudometric space (P,Π), where Π is a pseudometric function defined as

$$\Pi(p_i,p_j) = |(\text{the number of heads in } p_i) - (\text{the number of heads in } p_j)|,$$

is isometric (Def. 2.4) to the subspace $\{0,1,2,3\}$ of the real numbers, which is the set of values of the above random variable.

We can summarize the Introduction in the following points:

1) Obviously, the proposed approach becomes redundant when a *natural* representation in a Euclidean space can be found directly,

i.e., when suitable *comeasurable* real variables can be found immediately.

2) The two preliminary steps in the approach which are not present in the traditional setting are: a) to choose an appropriate formal representation, and b) to define an appropriate distance function (a dissimilarity measure).

3) The suggested framework for data analysis explicitly exposes the relativistic side of any analysis: figuratively speaking, the choice of a pseudometric space corresponds to the choice of a direction from which the picture of the data is going to be made; and, as always, for a given problem there are more illuminating and less illuminating directions.

4) To the dissatisfaction of some no *universal* guidance can exist as to which pseudometric function one should use, just as there is no universal guidance as to which data representation or probability measure to adopt (or spouse to choose). At the same time, the preliminary stage in pattern recognition is simplified in the sense that one now has an immeasurably larger arsenal of representation tools and hence does not have to make painful decisions as to how to code numerically non-numerical (structural) information.

5) The resulting vector space is not any longer a static universal space where any object from the fixed conceptual population can be represented (as in the classical case), but one determined by the training (finite) set of pattern, and thus can be compared with the physical notion of a field which may be altered by introducing a new object.

This space is constructed for a specific (mainly decision-theoretic) problem, and thus the coordinate axes in it many not have any simple interpretation. In other words, the approach offers the ultimate coordinate free framework for pattern recognition.

6) Most important, the approach allows us to use the power of the decision-theoretic "geometric" algorithms in a vector space to classify patterns in which structural information is of importance. In this sense one can say that *the unification of the two main existing approaches in pattern recognition is achieved.*

2 PSEUDOMETRIC SPACES

In this chapter we define the class of pseudometric spaces and some of its important subclasses. The definitions are illustrated by six interesting examples, which should demonstrate the universality of the notion of a pseudometric function. Then, we introduce and analyze the notion of a cartesian product of pseudometric spaces, which is important for analysis of the, so-called, mixed-mode data (see section 7.5). And finally, the notions of isometry and isometric embedding are defined.

It is interesting to note that the way people usually tend to compare a pair of objects (from the same conceptual universe) is in terms of their similarity, which has a simple interpretation through the intersection of the two *sets* of features. It seems, that this fact is the main reason behind the popularity of the notion of similarity in psychology. However, in mathematics the inverse notion of a dissimilarity, or a distance, is fundamental. This is simply because the original object becomes a point (rather than a set) in some abstract space.

Definition 2.1 By a pseudometric space*) we mean a set P together with a non-negative real-valued mapping

$$\Pi : P \times P \longrightarrow \mathbb{R}_+$$

satisfying the following two conditions

a) $\forall p_1, p_2 \in P \qquad \Pi(p_1, p_2) = \Pi(p_2, p_1)$ (symmetry)

b) $\forall p \in P \qquad \Pi(p, p) = 0$ (reflexivity)

Both notations (P, Π) and P (for short) will be used and the mapping Π will be called a pseudometric function (or simply a pseudometric).

In another terminology a pseudometric function Π is called a dissimilarity coefficient, DC ([10], p. 5-6).

A pseudometric space of finite cardinality will be called a

---

*)The term is not standard.

finite pseudometric space.

Definition 2.2 A pseudometric space (P,Π) which satisfies the following additional axiom

c) $\forall p_1,p_2 \epsilon P \qquad \Pi(p_1,p_2) = 0 \Longrightarrow p_1 = p_2$ (definiteness)

is called a semimetric space and the mapping Π will be called a semimetric function (or simply a semimetric).

Again, in another terminology a semimetric Π is called a definite dissimilarity coefficient, DCD ([10], p. 78).

It is important to note that the condition c) is not a necessary restriction on every distance function, since, in general, one wants to allow for the possibility of the two objects being completely similar (w.r.t. the chosen distance function) without being identical (see examples below).

Definition 2.3 A semimetric space (P,Π) which satisfies the following axiom

d) $\forall p_1,p_2,p_3 \epsilon P \qquad \Pi(p_1,p_2) + \Pi(p_2,p_3) \geq \Pi(p_1,p_3)$ (triangle inequality)

is called a metric space and the mapping Π is called a metric function (or simply a metric).

There are, of course, other subclasses of pseudometric spaces, e.g., premetric (a),b),d)), ultrametric (see [10], p. 78), etc. However, not all subclasses are of equal importance. Thus, for example, we have the following result which shows that a premetric space can be quite naturally transformed into a metric space.

Lemma 2.1. Let (P,Π) be a premetric space. Define an equivalence relation $\sim$ on P as follows

$$s_1 \sim s_2 \quad \text{if} \quad \Pi(s_1,s_2) = 0,$$

and define a mapping $\tilde{\Pi} : \tilde{P} \times \tilde{P} \longrightarrow \mathbb{R}_+$, with $\tilde{P} = P/\sim$, as

$$\tilde{\Pi}(\tilde{p}_1,\tilde{p}_2) \stackrel{def}{=} \Pi(p_1,p_2) \qquad p_1 \epsilon \tilde{p}_1,\ p_2 \epsilon \tilde{p}_2.$$

Then $(\tilde{P},\tilde{\Pi})$ is a metric space.

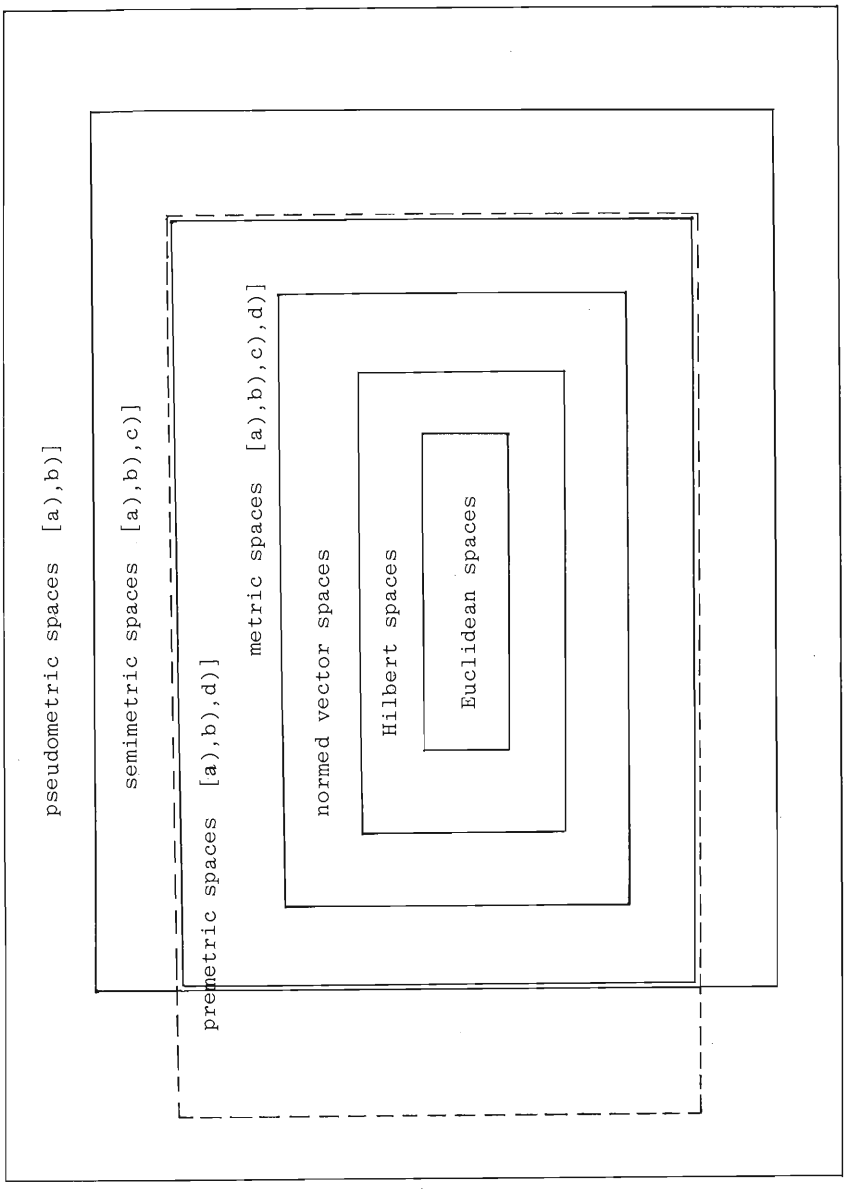

pseudometric spaces [a),b)]
semimetric spaces [a),b),c)]
premetric spaces [a),b),d)]
metric spaces [a),b),c),d)]
normed vector spaces
Hilbert spaces
Euclidean spaces

Figure 2.1

Proof of the lemma follows from the fact, that if $\Pi(p_1,p_2) = 0$, $\Pi(p_1,p_3) = a$, $\Pi(p_2,p_3) = b$, then $a + 0 \geq b$ and $b + 0 \geq a$, so that $a = b$. ∎

The set theoretical interrelationship between some classes of spaces is represented diagrammatically in Fig. 2.1.

For any of the above mentioned spaces the number $\Pi(p_1,p_2)$ will be called the distance between p_1 and p_2 in the space (P,Π) and the function Π will be called a distance function.

To give some idea about the variety of properties that could be represented by a symmetric distance function several examples are presented, only one of which is well known (Example 2.5). We begin with an example clearly demonstrating an important idea (also illustrated by the next three examples) that one should keep in mind when defining a distance function on a set of patterns for which a notion of a "symmetric difference" could *naturally* be introduced.

Example 2.1. Denote by Map(S,T) the set of mappings from a finite set $S = \{s_i\}_{1\leq i\leq n_1}$ into a finite set $T = \{t_i\}_{1\leq i\leq n_2}$. Define a pseudometric function $\Theta : \text{Map}(S,T) \times \text{Map}(S,T) \longrightarrow \mathbb{R}_+$ as follows (see Fig. 2.2)

$$\Theta(\mu_1,\mu_2) = \sum_{i=1}^{n_2} |\mu_1^{-1}(t_i) \Delta \mu_2^{-1}(t_i)|,$$

where μ_1,μ_2 Map(S,T) and $A\Delta B \stackrel{\text{def}}{=} (A\cup B) - (A\cap B)$ is a symmetric difference of sets A and B. One can show that Θ is metric.

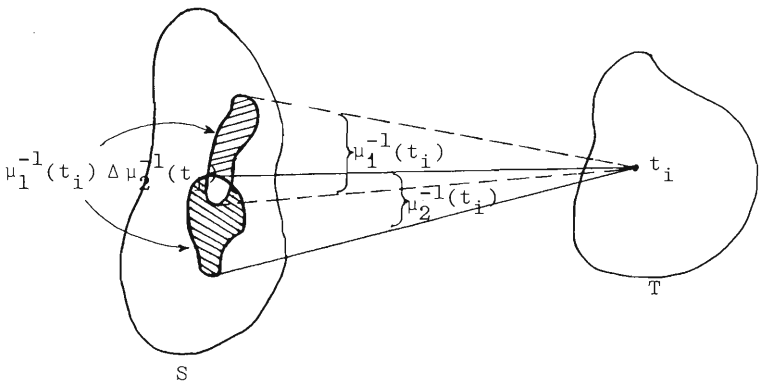


Figure 2.2

The above distance function may be useful in situations similar to those found in the learnability theory [25], where objects to be learned are assumed to be the above mappings, or in the analysis of the classifications of elements of S_1 into the classes labelled by elements of S_2 (see also [57]).

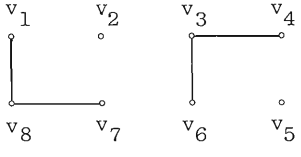
In the next example we consider a finite set of elements which are suited, for instance, for the representation of states of a system consisting of n objects and binary relations between them (e.g. a scene changing in time).

Example 2.2. Let G_n be the set of all graphs on n labelled vertices (there are $2^{\binom{n}{2}}$ of them), and let g_1, g_2 be elements of G_n. Define the graph $g_1 \Delta g_2$ as the symmetric difference of the graphs g_1 and g_2, i.e., the graph obtained from the union of the two graphs by removing common edges. For example, if $n = 8$ and g_1, g_2 are as in Fig. 2.3

g_1 g_2

Figure 2.3

then $g_1 \Delta g_2$ is shown in Fig. 2.4



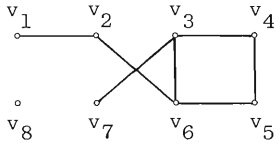
$g_1 \Delta g_2$

Figure 2.4

Define a mapping $Z : G_n \times G_n \longrightarrow \mathbb{R}_+$ as

$$Z(g_1,g_2) \stackrel{\text{def}}{=} \text{the number of proper cycles}^{*)} \text{ in } g_1 \Delta g_2.$$

One can see immediately that (G_n,Z) is a pseudometric space. Let us show that the symmetric function Z does not satisfy axioms c) and d) from Definitions 2.2 and 2.3 correspondingly, that is, (G_n,Z) is an example of a pseudometric space which is not semimetric nor premetric (and, therefore, cannot be transformed into a metric space). Choose g_1 and g_2 as in Fig. 2.2 and g_3 as in Fig. 2.5.



g_3

Figure 2.5

Then we have

$$Z(g_1,g_2) = 0 \quad (\text{but } g_1 \neq g_2),$$
$$Z(g_2,g_3) = 0, \quad Z(g_1,g_3) = 1$$

so that $Z(g_1,g_2) + Z(g_2,g_3) < Z(g_1,g_3)$. It is interesting to note that $Z(g_1,g_2) = 0$ if and only if $g_1 \Delta g_2$ is a forest (graph whose connected components are trees), and that $Z(g_1,g_2)$ numerically represents the dissimilarity between the cycle spaces of the two graphs.

The above example could be generalized in many ways, but the important point here is that an appropriately chosen pseudometric function can represent a *dissimilarity which is based on the presence of a particular structural element in the pattern representation.*

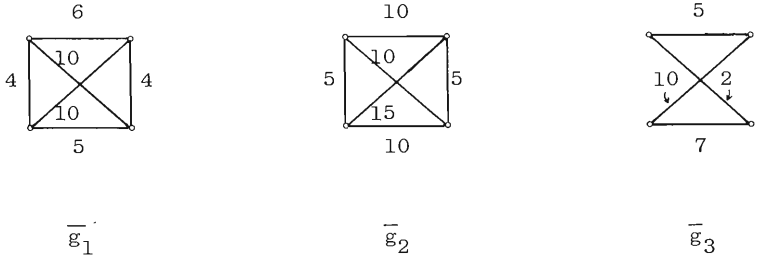
Example 2.3. Let $WG_n(M)$ be the set of all weighted graphs on the

*) An edge can appear in a cycle only once.

set V of n labelled vertices, where $(M, \|\cdot\|)$ is a normed vector space of generalized measurements (weights of the edges). In particular, M could be the n-dimensional Euclidean space $\mathbb{R}^n$ with the usual norm, or the set of all bounded continuous functions $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ with the norm $\|f\| = \sup_{\mathbb{R}^n} |f(x)|$. Define a mapping $\Psi : WG_n(M) \times WG_n(M) \longrightarrow \mathbb{R}_+$ as follows

$$\Psi(\bar{g}_1, \bar{g}_2)^{*)} \stackrel{def}{=} \begin{cases} \dfrac{\sum_{e \in E_{g_1 \Delta g_2}} \|w_e\|}{\sum_{e \in E_{g_1 \cap g_2}} \|w_e^1 - w_e^2\|}, & \text{if } g_1 \cap g_2 \neq g_0 \\ \sum_{e \in E_{g_1 \Delta g_2}} \|w_e\|, & \text{if } g_1 \cap g_2 = g_0, \end{cases}$$

where $E_{g_1 \Delta g_2}$ is the edge set of the graph $g_1 \Delta g_2$ defined in Example 2.2, $E_{g_1 \cap g_2}$ is the edge set of the graph $g_1 \Delta g_2$ which is the intersection graph of the labelled graphs g_1 and g_2, and g_0 is the graph with the empty set of edges; w_e^i is the weight of the edge e in the weighted graph $\bar{g}_i$ (i = 1,2), and w_e is that of e in $E_{g_1 \Delta g_2}$. Thus we obtain a pseudometric space $(WG_n(M), \Psi)$. It can be seen immediately, that mapping Ψ does not satisfy axioms c) and d) from Definitions 2.2 and 2.3 respectively, that is, Ψ is not semimetric nor premetric. Indeed, if n = 4, M = $\mathbb{R}$ and $\bar{g}_1, \bar{g}_2, \bar{g}_3$ are as in Fig. 2.6,



$\bar{g}_1$ $\bar{g}_2$ $\bar{g}_3$

Figure 2.6

*) If $\bar{g}$ denotes a weighted labelled graph, then g will denote the corresponding (underlying) graph.

then

$$\Psi(\bar{g}_1,\bar{g}_2) = 0 \quad (\text{but } \bar{g}_1 \neq \bar{g}_2),$$

$$\Psi(\bar{g}_1,\bar{g}_3) = \frac{4+4}{1+2+0+8} = \frac{8}{11}, \quad \Psi(\bar{g}_2,\bar{g}_3) = \frac{5+5}{5+3+5+8} = \frac{10}{21},$$

so that $\Psi(\bar{g}_1,\bar{g}_2) + \Psi(\bar{g}_2,\bar{g}_3) < \Psi(\bar{g}_1,\bar{g}_3)$.

It is not difficult to see how the ideas expressed in the above two examples could be put together. Thus, a pseudometric function (which is not always a metric) can be constructed that measures dissimilarity between two objects in terms of the *total weight of specific structural elements* which the two objects do not share: in the above example one can set the distance between $\bar{g}_1$ and $\bar{g}_2$ to be equal to the sum of the weights of the proper cycles in graph $g_1 \Delta g_2$.

Example 2.4. Let $CB(\mathbb{R}^n)$ be the set of all n-dimensional compact sets (bodies) in the real Euclidean space $\mathbb{R}^n$, and let $B_1, B_2 \in CB(\mathbb{R}^n)$. Set $\bar{B}_1 = c_1 B_1 = \{x \in \mathbb{R}^n \mid x = c_1 y, y \in B_1\}$, $\bar{B}_2 = c_2 B_2$, where c_1, c_2 are such real numbers that the following identity for the volumes is satisfied: $v(\bar{B}_1) = v(\bar{B}_2) = 1$. Next, translate and rotate $\bar{B}_2$ in such a way that the new convex body $\bar{B}_2'$ satisfies the following condition: $v(\bar{B}_1 \Delta \bar{B}_2')$ is minimal, where $\bar{B}_1 \Delta \bar{B}_2' = (\bar{B}_1 - \bar{B}_2') \cup (\bar{B}_2' - \bar{B}_1)$ is a set-theoretic symmetric difference (see Fig. 2.7).

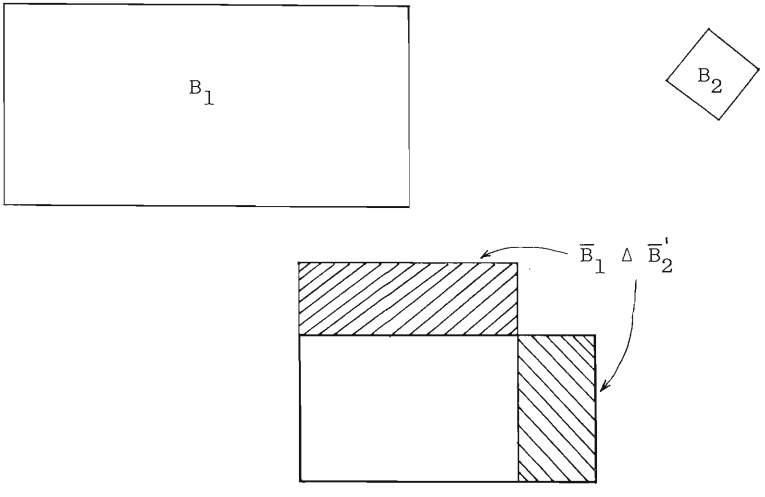


Figure 2.7

Define a function $\Delta : CB(\mathbb{R}^n) \times CB(\mathbb{R}^n) \longrightarrow \mathbb{R}_+$ as

$$\Delta(B_1, B_2) = v(\bar{B}_1 \Delta \bar{B}_2').$$

Obviously, $(CB(\mathbb{R}^n), \Delta)$ is a pseudometric space, and Δ measures a shape dissimilarity. It is not difficult to show that Δ is a metric.

The next example demonstrates another way (see Example 2.2) in which a distance could be introduced on a set of syntactic patterns.

Example 2.5. ([26]-[28], [11]) For a grammar

$$G = (V_N, V_T, P, S)$$

Levenshtein in [26] defined a distance between two words $x, y \in V_T^*$ as the minimal number of substitutions, deletions, or insertions required to transform y into x, where substitution, deletion and insertion are the transformations defined as follows ($s_1, s_2 \in V_T^*$):

i)	$s_1 a s_2 \longrightarrow s_1 b s_2$	$\forall a, b \in V_T \ (a \neq b)$	(substitution)
ii)	$s_1 a s_2 \longrightarrow s_1 s_2$	$\forall a \in V_T$	(deletion)
iii)	$s_1 s_2 \longrightarrow s_1 a s_2$	$\forall a \in V_T$	(insertion).

One can show that this distance function is a metric.

The last example could be generalized in many ways. We give one which combines the above idea with the notion of a transformation group (syntactic representation of a closed curve).

Example 2.6. Let V be a set of elements (alphabet), and let the patterns to be studied be finite circular arrangements of these elements (see Fig. 2.8).

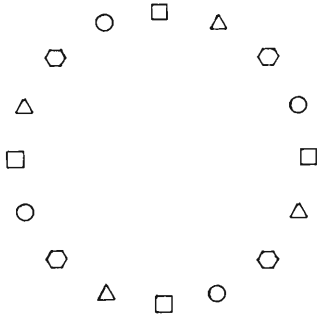


Figure 2.8

Define on the above set of patterns V^* the following equivalence relation: $v_1^* \sim v_2^*$ if v_2^* can be obtained from v_1^* by a rotation. Denoting the quotient set of the equivalence classes $V^*/\sim$ by $\tilde{V}^*$, we can define on it a distance function $\Omega : \Omega(\tilde{v}_1^*, \tilde{v}_2^*)$ is equal to the minimal number of substitutions, deletions, or insertions required to transform an element $v_1^* \epsilon \tilde{v}_1^*$ into an element $v_2^* \epsilon \tilde{v}_2^*$. Obviously, we obtain a metric space $(\tilde{V}^*, \Omega)$. For an application of this distance function see [58].

The readers interested in seeing how a contextual distance function could be introduced are referred to [29]-[31].

The above examples can be supplemented, of course, by a large number of the classical examples of metric spaces, as well as by others which are scattered throughout numerous papers in mathematics and other disciplines.

Next, we consider the situation where it is not clear how several qualitatively different groups of features in the object representation can be directly combined into one dissimilarity measure. For example, consider the case, where several different distance functions are defined on the same set of objects (but possibly on *different representation sets*). Can we put these several images of the same data together into one image? The answer is yes, if we are prepared to accept an *a priori* weighting scheme which may not have an absolute justification. It is quite possible, however, that in the future some formal goal oriented procedures to deal with this fundamental problem (weighting scheme) will be developed.

Thus, let (P_i, Π_i), $1 \leq i \leq r$, be a finite set of pseudometric spaces, where some or possibly all of the P_i might be the same. A <u>weighted Cartesian product</u> $\underset{i=1}{\overset{r}{\times}} (P_i, \Pi_i)$ of (P_i, Π_i) is a pseudometric space (P, Π) with $P = \underset{i=1}{\overset{r}{\times}} P_i$ and Π defined as

$$\Pi((p_1^1, \ldots, p_1^r), (p_2^1, \ldots, p_2^r)) = [w_1^2 \Pi_1^2(p_1^1, p_2^1) + \ldots + w_r^2 \Pi_r^2(p_1^r, p_2^r)]^{\frac{1}{2}},$$

where $p_1^i, p_2^i \epsilon P_i$, and w_i is a relative weight of Π_i w.r.t. $\{\Pi_j\}_{1 \leq j \leq r}$ (see Fig. 2.9).

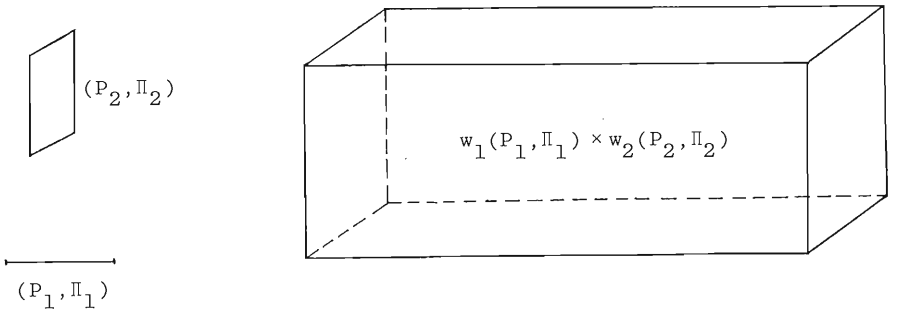


Figure 2.9

There are alternative ways of constructing a weighted distance function on the product of several pseudometric spaces:

1) $\Pi((p_1^1,\ldots,p_1^r),(p_2^1,\ldots,p_2^r)) = w_1\Pi_1(p_1^1,p_2^1) + \ldots + w_r\Pi_r(p_1^r,p_2^r)$,

or

2) $\Pi((p_1^1,\ldots,p_1^r),(p_2^1,\ldots,p_2^r)) = \max(w_1\Pi_1(p_1^1,p_2^1),\ldots,w_r\Pi_r(p_1^r,p_2^r))$.

However, these are useful in a small number of cases, and even then only because of the computational simplicity.

In practice, an obvious simplification in the construction of the weighted Cartesian product can be achieved by proceeding in two stages. First, one groups together those (P_i,Π_i) which have the same P_i (and different Π_i), and for each such group $\{(p_j,\Pi_{j_k})\}_{1\le k\le t}$ define a weighted Cartesian product (P_j,Π_j) by a simpler looking formula

$$\Pi_j(p_1,p_2) = [w_1^2\Pi_{i_1}^2(p_1,p_2) + \ldots + w_t^2\Pi_{i_t}^2(p_1,p_2)]^{\frac{1}{2}} \qquad p_1,p_2\epsilon P_j.$$

And only then, using the former formula, define a complete weighted Cartesian product by taking a weighted Cartesian product of (P_j,Π_j) constructed in the first stage. Obviously, the weighting scheme in each stage is easier to construct, than the one for the entire product.

If one does not have any idea as to which weighting scheme to adopt in the case of finite P_i (the case we are mainly concerned with), the following two simplest schemes could be (a priori) tried

out:

1) $w_i = 1$ $\quad 1 \leq i \leq r,$

2) $w_i = [\text{diam}(P_i)]^{-1}$

where $\text{diam}(P_i)$ is the largest of all the distances $\Pi_i(p_1^i, p_2^i)$ $p_1^i, p_2^i \in P_i$. Obviously, the w_i's in 2) reduce the configuration to a standard size. A still better weighting scheme, the author thinks, should be goal oriented, and should be obtained by comparing the geometric configuration of (P_i, Π_i) with that of a natural superspace (of which it is a subspace), as well as with the geometric configurations of the other (P_j, Π_j). In other words, since the geometric structure of the sets spanned by the points in $\bigtimes_{i=1}^{r} (P_i, \Pi_i)$ are invariant w.r.t. the scaling, the information contained in them could be used for *a posteriori* weighting scheme that is best suited for a particular problem. For a discussion of the weighting issue the reader is referred to [10], Chapter 4, and [32].

Thus, one possible way the preliminary stage in pattern analysis could be organized is represented schematically in Fig. 2.10.

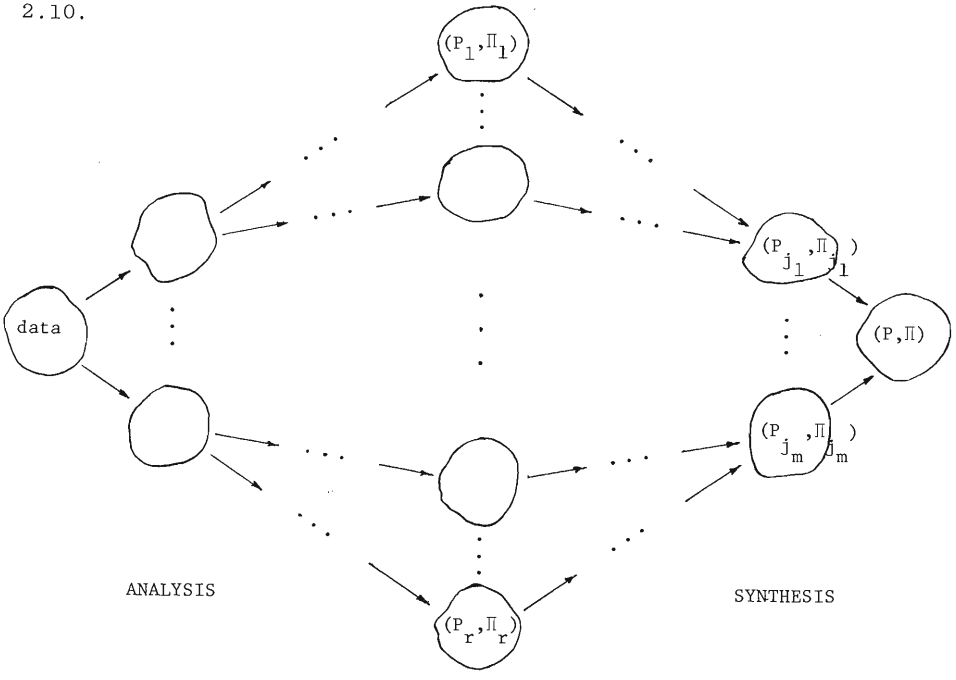


Figure 2.10

Next, we give a definition of which spaces, from the metric point of view, should be considered as identical.

Definition 2.4. A pseudometric space (P,Π) is called isometric to a pseudometric space (R,Ω) if there exists a surjection $\beta : P \longrightarrow R$ such that

$$\forall p_1,p_2 \epsilon P \qquad \Omega(\beta(p_1),\beta(p_2)) = \Pi(p_1,p_2),$$

in other words, if there exists a distance preserving onto mapping β (called isometry) between the spaces. A distance preserving mapping α is called an isometric embedding (see Fig. 2.11).

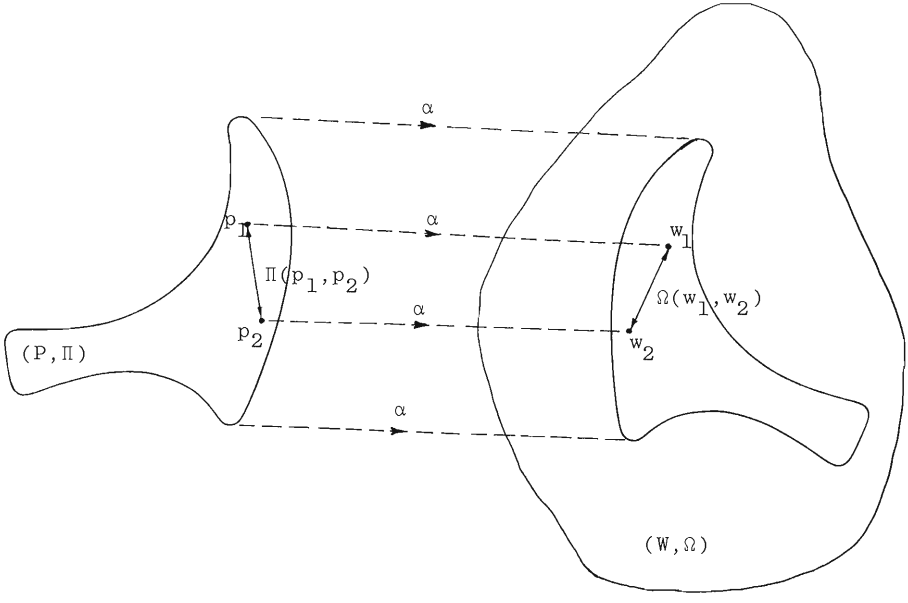


Figure 2.11

By a subspace of a pseudometric space (P,Π) we mean a subset $P_1 \subseteq P$ together with the same distance function: $\Pi_1 = \Pi|P_1 \times P_1$. We denote this by $(P_1,\Pi_1) \subseteq (P,\Pi)$ or $P_1 \subseteq P$ (for short).

Obviously, an isometric embedding $\alpha : (P,\Pi) \longrightarrow (R,\Omega)$ establishes an isometry between (P,Π) and the subspace of (R,Ω) which is the range of α. It is also easy to see that any isometry of a semi-metric space must be a bijection (one-to-one).

3 PSEUDOEUCLIDEAN SPACES

In this chapter almost all the technical facts needed for the development of the approach will be given. They are generalizations of the corresponding facts from Euclidean spaces to non-Euclidean spaces. Adjective "non-Euclidean" means that the distance defined in the vector space is not necessarily measured by the Pythagorean formula, as in the case of Euclidean spaces. Thus, a greater generality is achieved. Since the material is not a part of any standard course, the presentation is self-contained and well illustrated. In several places in section 3.1 I used §36 of [33], which is generally an excellent textbook (see also other textbooks: [34], App. II; [35], Ch. IX; [36], 5.1-5.4; [37], 6.6-6.7; [38], Ch. 1; [39], Ch. 6, 7). Unfortunately, the above books do not contain all the material needed.

3.1 Symmetric bilinear forms on finite-dimensional vector spaces*)

During the last century it became increasingly clear that the concept of a symmetric bilinear form (equivalently, a quadratic form, see App. I) on a vector space was the most convenient one for introducing the notion of generalized distance in a vector space. Thus, it is a natural starting point for us.

Definition 3.1. Let V be a vector space over the field $\mathbb{R}$. A symmetric bilinear form on V is a mapping

$$\Phi : V \times V \longrightarrow R$$

satisfying the following conditions:

$\forall x, x_1, x_2, y \in V$, $\forall c \in \mathbb{R}$

1) $\Phi(x_1+x_2, y) = \Phi(x_1, y) + \Phi(x_2, y)$

2) $\Phi(cx, y) = c\Phi(x, y)$

3) $\Phi(x, y) = \Phi(y, x)$

One should think of a symmetric bilinear form as a generalized

*) All vector spaces are assumed to be finite-dimensional and over the field $\mathbb{R}$ of real numbers.

inner product.

The following are the corollaries of the above definition:

$$\forall x \in V \qquad \Phi(x,0) = \Phi(0,x) = 0$$

$$\forall x_1, x_2, y \in V \qquad \Phi(y, x_1+x_2) \overset{\text{by 3)}}{=} \Phi(x_1+x_2, y) \overset{\text{by 1)}}{=} \Phi(x_1,y) + \Phi(x_2,y) \overset{\text{by 3)}}{=} \Phi(y,x_1) + \Phi(y,x_2),$$

and

$$\forall x, y \in V,\ \forall c \in \mathbb{R} \qquad \Phi(x,cy) \overset{3)}{=} \Phi(cy,x) \overset{2)}{=} c\Phi(y,x) \overset{3)}{=} c\Phi(x,y);$$

so that

$$(3.1) \qquad \Phi\Big(\sum_{j=1}^{n} c_j x_j,\ \sum_{k=1}^{m} d_k y_k\Big) = \sum_{j=1}^{n}\sum_{k=1}^{m} c_j d_k \Phi(x_j, y_k)$$

for every set $\{x_1,\ldots,x_n,y_1,y_2,\ldots,y_m\} \subset V$ and every set $\{c_1,\ldots,c_n,d_1,d_2,\ldots,d_m\} \subset \mathbb{R}$.

The Euclidean inner (scalar) product is a symmetric bilinear form satisfying the following additional axiom:

$$\forall x \in V, \quad x \neq 0 \quad \Phi(x,x) > 0.$$

Another interpretation of a symmetric bilinear form comes from the next definition.

Definition 3.2. Let V be a real vector space, and Φ be a symmetric bilinear form on it. The square of the distance between vectors x and y of V with respect to the form Φ is the number

$$\|x-y\|^2 \overset{\text{def}}{=} \Phi(x-y,x-y),$$

and the square of the length of x (w.r.t. Φ) is the square of its distance from 0, i.e., the number

$$\|x\|^2 \overset{\text{def}}{=} \Phi(x,x).$$

Next, we show how to evaluate $\Phi(x,y)$. For this purpose let us choose any basis $(a_i)_{1\le i\le n}$ of V ($\dim(V) = n$), and let $x = \sum_{i=1}^{n} x^i a_i$, $y = \sum_{i=1}^{n} y^i a_i$ be any two vectors of V. Then from (3.1)

$$\Phi(x,y) = \sum_{i=1}^{n} \sum_{j=1}^{n} x^i y^j \Phi(a_i, a_j), \tag{3.2}$$

and therefore the numbers $\Phi(a_i,a_j)$ completely determine the bilinear form Φ (under the fixed basis $(a_i)_{1\le i\le n}$).

Definition 3.3. Numbers $\Phi(a_i,a_j)$ are called the coefficients of the symmetric bilinear form Φ with respect to the given basis, and the square matrix

$$M(\Phi) \overset{\text{def}}{=} (\Phi(a_i,a_j))_{1\le i,j\le n} = (m_{ij})_{1\le i,j\le n}$$

is called the matrix of Φ with respect to the basis $(a_i)_{1\le i\le n}$.

From axiom 3) of Definition 3.1 it follows that the matrix of a symmetric bilinear form with respect to any basis $(a_i)_{1\le i\le n}$ is symmetric: ${}^t[M(\Phi)] = M(\Phi)$.

It is also easy to see that (3.2) can be written as

$$\Phi(x,y) = \begin{pmatrix} y^1 \\ \vdots \\ y^n \end{pmatrix}^t \cdot M(\Phi) \cdot \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix}, \tag{3.3}$$

from which one immediately obtains that if $\bar{M}(\Phi)$ is the matrix of Φ with respect to another basis $(\bar{a}_i)_{1\le i\le n}$, and T is the matrix of transition $\tau : V \longrightarrow V$ from the basis (a_i) to the basis $(\bar{a}_i)$, then

$$\bar{M}(\Phi) = {}^tT \cdot M(\Phi) \cdot T. \tag{3.4}$$

Indeed, it is enough to substitute in (3.3) $x = a_i$ and $y = a_j$, remembering that

$$\begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} = T \cdot \begin{pmatrix} \bar{x}^1 \\ \vdots \\ \bar{x}^n \end{pmatrix},$$

where $\bar{x}^1$, $1\le i\le n$, are the coordinates of x with respect to the new basis $(\bar{a}_i)_{1\le i\le n}$.

The notion of the orthogonality between two vectors can also be introduced.

Definition 3.4. Vectors x and y of a vector space V, with a symmetric bilinear form Φ on it, are said to be orthogonal to each other w.r.t. Φ, or Φ-orthogonal, if $\Phi(x,y) = 0$. This fact will be denoted $x \perp y$.

Using (3.2) it is easy to show that for every set $U \subseteq V$ the set of all vectors orthogonal to every vector from U is a subspace of V, called the orthogonal complement of U (w.r.t. Φ), and is denoted $U^{\perp}$.

Similarly, from (3.2) it follows that if $x \in V$ is orthogonal to a set of vectors $\{a_i\}_{i \in I} = A$ of V, then x is orthogonal to a subspace $\langle A \rangle$ generated by A. In particular, for a subspace U of V to show that $x \in U^{\perp}$, it is enough to check that $\Phi(a_i, x) = 0$, $1 \leq i \leq m$, for a basis $(a_i)_{1 \leq i \leq m}$ of U.

Next, let $x \perp y$ (w.r.t. Φ), i.e., $\Phi(x,y) = 0$, then

$$\begin{aligned} &\Phi(x+y,x+y) = \Phi(x,x) + \Phi(y,x) + \Phi(x,y) + \Phi(y,y) = \Phi(x,x) + \Phi(y,y) \\ &\Phi(x-y,x-y) = \Phi(x,x) - \Phi(y,x) - \Phi(x,y) + \Phi(y,y) = \Phi(x,x) + \Phi(y,y). \end{aligned} \tag{3.5}$$

The second statement, in view of Def. 3.2, may be called the generalized theorem of Pythagoras (see Fig. 3.1).

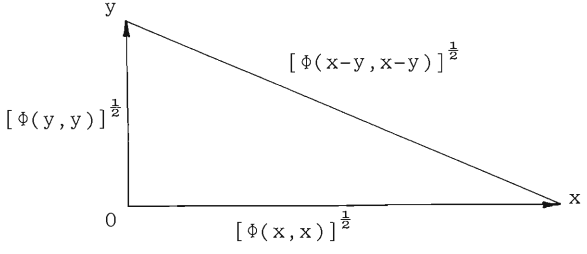


Figure 3.1

Theorem 3.1. For every symmetric bilinear form Φ on V
$\dim(V^{\perp}) = \dim(V) - \operatorname{rank}(M(\Phi))$.

Proof. Let us fix a basis $(a_i)_{1 \leq i \leq n}$ in V ($\dim(V) = n$). Then, $x \in V^{\perp} \iff \forall\, y \in V \quad \Phi(x,y) = 0$, which using (3.3) can be written as

$$\forall (y^1,\ldots,y^n)\in \mathbb{R}^n \qquad {}^t\begin{pmatrix} y^1 \\ \vdots \\ y^n \end{pmatrix}\cdot M(\Phi)\cdot \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} = 0.$$

The last statement is equivalent to

$$M(\Phi)\cdot \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix},$$

which means that $x\in \mathrm{Null}(\mu)$, where $\mu : V \longrightarrow V$ is an endomorphism of V whose matrix in the fixed basis is $M(\Phi)$. Thus,

$$V^{\perp} = \mathrm{Null}(\mu).$$

The statement of the theorem follows now from the well known fact that

$$\dim(\mathrm{Null}(\mu)) = \dim(V) - \mathrm{rank}(\mu). \quad \blacksquare$$

Next, several important notions will be introduced, although complete clarification of them is postponed until later (see Corollary to Theorem 3.12).

Definition 3.5. A symmetric bilinear form Φ on a vector space V of dimension n is said to be non-degenerate, if the rank of its matrix with respect to some basis of V is equal to n, and degenerate otherwise; it is called positive if it is non-degenerate and $\Phi(x,x) \geq 0 \quad \forall\, x\in V$, negative if it is non-degenerate and $\Phi(x,x) \leq 0 \quad \forall\, x\in V$, and indefinite if $\exists x,y\in V$ such that $\Phi(x,x) < 0$, $\Phi(y,y) > 0$.

Obviously, the definition of the degeneracy of a form Φ is correct, since for any other basis $(\bar{a}_i)_{1\leq i\leq n}$ using (3.4) we obtain

$$\det(\bar{M}(\Phi)) = [\det(T)]^2\cdot\det(M(\Phi)), \tag{3.6}$$

and since for a transition matrix T $\det(T) \neq 0$, we also have

$$\det(M(\Phi)) = 0 \iff \det(\bar{M}(\Phi)) = 0.$$

As was mentioned above, positive symmetric bilinear forms are usually called inner (scalar) products, and vector spaces with inner products are widely known. The most known example of an indefinite symmetric bilinear form is the following, defined on $\mathbb{R}^4$, called in the theory of relativity the Lorentz form,

$$\Phi(x,y) = x^1y^1 + x^2y^2 + x^3y^3 - cx^4y^4$$

where c is the speed of light.

From Theorem 3.1 it follows that the notion of the non-degeneracy of a symmetric bilinear form Φ is equivalent to the statement:

$$V^{\perp} = \{0\}.$$

In other words, a form Φ is non-degenerate if and only if 0 is the only vector Φ-orthogonal to all vectors of V.

The following definition will be useful in the sequel.

Definition 3.6. Let Φ be a symmetric bilinear form on V, then the mapping

$$\ell_{\Phi} : V \longrightarrow V^*$$

defined as $\forall x \in V \ \ell_{\Phi}(x) \stackrel{\text{def}}{=} {}_x\Phi$, where ${}_x\Phi$ is a linear form on V such that $\forall y \in V \ {}_x\Phi(y) \stackrel{\text{def}}{=} \Phi(x,y)$, is called left-associated with Φ. Similarly, the mapping

$$r_{\Phi} : V \longrightarrow V^*$$

defined as $\forall x \in V \ r_{\Phi}(y) \stackrel{\text{def}}{=} \Phi_y$, where Φ_y is a linear form on V such that $\forall y \in V \ \Phi_y(x) \stackrel{\text{def}}{=} \Phi(x,y)$, is called right-associated with Φ.

Theorem 3.2. The mappings ℓ_{Φ} and r_{Φ} are homomorphisms, and $\ell_{\Phi} = r_{\Phi}$.

Proof follows from Definitions 3.6, 3.1. ∎

The next theorem gives two more alternative definitions of the non-degeneracy of a form Φ.

Theorem 3.3. For a symmetric bilinear form Φ on a vector space V the following statements are equivalent:

a) Φ is non-degenerate, or $\det(M(\Phi)) = \det((m_{ij})_{1 \le i,j \le n}) \neq 0$.

b) The homomorphism $\ell_\Phi (= r_\Phi)$ is a monomorphism, or

$$\Phi(x,y) = 0 \quad \forall y \in V \quad \Longrightarrow \quad x = 0;$$

c) The homomorphism $\ell_\Phi (= r_\Phi)$ is an isomorphism, or, for every linear form $\phi \in V^*$ there exists a unique vector $x_\phi \in V$ such that

$$\phi(y) = \Phi(x_\phi, y) \quad \forall v \in V.$$

Proof. b) $\Longleftrightarrow$ a). By definition, ℓ_Φ is a monomorphism if and only if $\mathrm{Null}(\ell_\Phi) = \{0\}$. But

$$\mathrm{Null}(\ell_\Phi) = \{x \in V \mid \Phi(x,y) = 0 \quad \forall y \in V\} = V^\perp,$$

and the sought equivalence follows from the fact stated just before Definition 3.6.

a) $\Longleftrightarrow$ c). Let us choose any $\phi \in V^*$, and set $\phi(a_i) = c_i$ for a basis $(a_i)_{1 \le i \le n}$ of V. Obviously, ϕ is uniquely determined by $(c_i)_{1 \le i \le n}$. So, setting $x = \sum_{i=1}^{n} x^i a_i$, $y = \sum_{j=1}^{n} y^j a_j$ we have

$$\Phi(x,y) = \phi(y) \quad \forall y \overset{(3.2)}{\Longleftrightarrow} \sum_{j=1}^{n} \sum_{i=1}^{n} m_{ij} x^i y^j = \sum_{j=1}^{n} y^j c_j \quad \forall (y^1, \ldots, y^n)$$

$$\Longleftrightarrow \sum_{j=1}^{n} y^j \Big(\sum_{i=1}^{n} m_{ij} x^i - c_j \Big) = 0 \quad \forall (y^j)_{1 \le j \le n} \Longleftrightarrow \sum_{i=1}^{n} m_{ij} x^i - c_j = 0 \quad (1 \le j \le n).$$

Therefore, the set of all x satisfying $\Phi(x,y) = \phi(y) \quad \forall y \in V$ coincides with the solution set of the linear system

$$\sum_{i=1}^{n} m_{ij} x^i - c_j = 0 \quad (1 \le j \le n),$$

and the sought equivalence follows from a well known theorem on the uniqueness of the solution of a linear system with a non-zero determinant. ∎

As a corollary we obtain the following important fact: if the form Φ is non-degenerate, then $\ell_\Phi (= r_\Phi)$ gives *canonical* (independent of a basis) *isomorphism between* V and V^*.

The next theorem gives two equivalent conditions for a form Φ, defined on a vector space V, to induce a non-degenerate form on a subspace $U \subseteq V$.

Theorem 3.4. If Φ is a symmetric bilinear form on V, then for every subspace U of V the following statements are equivalent:

a) $\Psi \overset{\text{def}}{=} \Phi|U \times U$ is a non-degenerate symmetric bilinear form on U,

b) $U \cap U^{\perp} = \{0\}$,

c) $V = U \oplus U^{\perp}$.

Proof. Denoting by $U_{\Psi}^{\perp}$ the vector subspace of U which is the orthogonal complement of U w.r.t. Ψ, we have (see Fig. 3.2)

$$U \cap U^{\perp} = U_{\Psi}^{\perp}$$

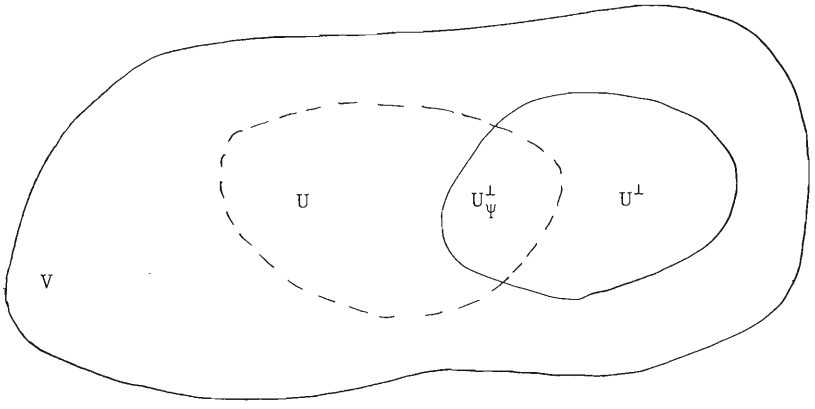


Figure 3.2

Indeed, $U_{\Psi}^{\perp} \subseteq U$, $U_{\Psi}^{\perp} \subseteq U^{\perp}$, and $x \in U \cap U^{\perp} \Longrightarrow x \in U_{\Psi}^{\perp}$. But a) is equivalent to $U_{\Psi}^{\perp} = \{0\}$ (see the remark before Def. 3.6), and the equivalence a) $\Longleftrightarrow$ b) follows.

Obviously, c) $\Longrightarrow$ b), so that in view of the above c) $\Longrightarrow$ a). Let us prove the implication a) $\Longrightarrow$ c).

Since a) implies b), it is enough to show that $V = U + U^{\perp}$, or, that every vector $v \in V$ can be expressed as a sum of two vectors from the corresponding subspaces. To this end consider a linear form ϕ on U defined by

$$\phi(u) = \Psi(u,v) = \Phi(u,v) \qquad \forall u \in U.$$

From Theorem 3.3 a), c) it follows that there exists a vector $u_{\phi} \in U$ such that

$$\phi(u) = \Psi(u,u_{\phi}) = \Phi(u,u_{\phi}) \qquad \forall u \in U.$$

Thus,

$$\Phi(u,v) = \Phi(u,u_\phi) \qquad \forall u \in U,$$

whence

$$\Phi(u,v-u_\phi) = 0 \qquad \forall u \in U,$$

i.e., $v-u_\phi \in U^\perp$. Finally,

$$v = u_\phi + (v-u_\phi)$$

where $u_\phi \in U$, $v-u_\phi \in U^\perp$. ∎

To prove the next theorem we need one simple but important fact.

Lemma 3.5. Let U^0 be the annihilator of a subspace U of a finite-dimensional real vector space V, i.e.,

$$U^0 = \{\psi \in V^* \mid \psi(x) = 0 \quad \forall x \in U\}.$$

Then

$$\dim(U) + \dim(U^0) = \dim(V).$$

Proof. Let us complete a basis $(a_i)_{1 \le i \le m}$ of U to a basis $(a_i)_{1 \le i \le n}$ of V, and denote by $(\psi_i)_{1 \le i \le n}$ the dual basis, that is, a basis of the dual space V^* defined by

$$\psi_i(x) = x^i \qquad \forall x \in V,$$

where $x = \sum_{i=1}^{n} x^i a_i$. To prove the theorem, it is enough to show that $(\psi_i)_{m+1 \le i \le n}$ is a basis of U^0. Since $(\psi_i)_{m+1 \le i \le n}$ are linearly independent, it is enough to show that they generate U^0. Take any $\psi \in U^0$, then, since $(\psi_i)_{1 \le i \le n}$ is a basis of V^*, we have

$$\psi = \sum_{i=1}^{n} c_i \psi_i,$$

where $c_i \in \mathbb{R}$ $(1 \le i \le n)$. But $\psi(a_i) = 0$ $(1 \le i \le m)$, whence $c_i = 0$ $(1 \le i \le m)$, and therefore

$$\psi = \sum_{i=m+1}^{n} c_i \psi_i. \quad ∎$$

Now we are ready to prove the following important result.

Theorem 3.6. If Φ is a non-degenerate symmetric bilinear form on V, then for every subspace U of V

$$\dim(U) + \dim(U^{\perp}) = \dim(V).$$

Proof. Since $\dim(U) + \dim(U^{0}) = \dim(V)$ (Lemma 3.5), it is enough to prove that

$$\dim(U^{\perp}) = \dim(U^{0}).$$

Take $\forall \phi \in U^{0}$; Φ is non-degenerate, therefore by Theorem 3.3 a), c) there exists unique $y_{\phi} \in V$ such that $\phi(x) = \Phi(x, y_{\phi})$ $\forall x \in V$. Whence $\Phi(x, y_{\phi}) = 0$ $\forall x \in U$, i.e., $y_{\phi} \in U^{\perp}$. But (see Def. 3.6)

$$\phi = r_{\Phi}(y_{\phi}),$$

therefore $U^{0} \subset r_{\Phi}(U^{\perp})$.

On the other hand, if $y \in U^{\perp}$, then $\Phi(x,y) = 0$ $\forall x \in U$, and $\Phi_{y} = r_{\Phi}(y) \in U^{0}$ (see Def. 3.6), i.e., $r_{\Phi}(U^{\perp}) \subseteq U^{0}$. Thus,

$$r_{\Phi}(U^{\perp}) = U^{0},$$

and the equality of their dimensions follows from the fact that $r_{\Phi} : V \longrightarrow V^{*}$ is an isomorphism (Theorem 3.3). ∎

It is interesting to note that in the course of the proof of the above theorem we have actually proved two facts:

1) for every form Φ

$$\ell_{\Phi}(U^{\perp}) = r_{\Phi}(U^{\perp}) \subseteq U^{0};$$

2) for a non-degenerate Φ the isomorphism $r_{\Phi} (= \ell_{\Phi})$ establishes an isomorphism between $U^{\perp}$ and U^{0}.

Theorem 3.7. For a non-degenerate symmetric bilinear form Φ each of the three (equivalent) statements in Theorem 3.4 is equivalent to

$$\text{d)} \quad V = U + U^{\perp}.$$

Proof. By the previous theorem $\dim(V) = \dim(U) + \dim(U^{\perp})$, which combined with d) is equivalent to the statement c) of Theorem 3.4. ∎

The next two theorems give some basic properties of operation $^{\perp}$.

Theorem 3.8. Suppose Φ is a non-degenerate symmetric bilinear form on V, then for every subspace U of V

$$(U^{\perp})^{\perp} = U.$$

Proof. Applying Theorem 3.6 first to $U^{\perp}$ and then to U, we obtain

$$\dim((U^{\perp})^{\perp}) = \dim(V) - \dim(U^{\perp}) = \dim(V) - [\dim(V) - \dim(U)] = \dim(U).$$

And the theorem follows from a simple fact that

$$U \subseteq (U^{\perp})^{\perp}.$$

To prove the last inclusion, let u be an arbitrary vector in U, then, by definition of $U^{\perp}$ $\Phi(x,u) = 0 \quad \forall x \in U$, and hence $u \in (U^{\perp})^{\perp}$.

Theorem 3.9. Let Φ be a non-degenerate symmetric bilinear form on a vector space V, and let U, W be subspaces of E. Then

a) $(U + W)^{\perp} = U^{\perp} \cap W^{\perp}$

b) $(U \cap W)^{\perp} = U^{\perp} + W^{\perp}$.

Proof. a) $x \in (U + W)^{\perp} \Longleftrightarrow \Phi(u+w,x) = 0 \quad \forall u \in U, \ \forall w \in W \Longleftrightarrow$

$\Phi(u,x) + \Phi(w,x) = 0 \quad \forall u \in V, \quad \forall w \in W \Longleftrightarrow$

$\Phi(u,x) = 0 \quad \forall u \in U$ and $\Phi(w,x) = 0 \quad \forall w \in W$,

where the last $\Longleftarrow$ is obvious, and the last $\Longrightarrow$ is obtained by setting first $w = 0$, then $u = 0$.

b) Using Theorem 3.8 and a) of this theorem we have

$$U^{\perp} + W^{\perp} = [(U^{\perp} + W^{\perp})^{\perp}]^{\perp} = [(U^{\perp})^{\perp} \cap (W^{\perp})^{\perp}]^{\perp} = [U \cap W]^{\perp}. \quad \blacksquare$$

The following proposition clarifies the relation between degenerate and non-degenerate forms.

Lemma 3.10. Let Φ be a symmetric bilinear form on V, then for every direct decomposition of V

$$V = V^{\perp} \oplus V_1$$

the form induced by Φ on V_1 is non-degenerate. Moreover, if for some U, $\Phi | U \times U$ is non-degenerate, then $\dim(U) \leq \dim(V_1)$.

Proof. We prove the first statement by contradiction. Suppose $\Phi | V_1 \times V_1$ is degenerate, i.e., there exists a nonzero $x \in V_1$ such that

$$\Phi(x_1,y) = 0 \qquad \forall y \in V_1.$$

Then

$$\Phi(x_1,y) + \Phi(x_1,z) = 0 \qquad \forall y \in V_1, \ \forall z \in V^{\perp},$$

whence

$$\Phi(x_1,x) = 0 \qquad \forall x \in V,$$

i.e., $x \in V^{\perp}$, and since $V^{\perp} \cap V_1 = \{0\}$, we obtain the contradiction.

To prove the second statement, let us write V as a direct sum

$$V = U \oplus U^{\perp}$$

(Theorem 3.4). Obviously, $V^{\perp} \subseteq U^{\perp}$, and thus

$$\dim(V_1) = \dim(V) - \dim(V^{\perp}) \geq \dim(V) - \dim(U^{\perp}) = \dim(U). \quad \blacksquare$$

The above lemma allows us to reduce the problem of finding the matrix of a symmetric bilinear form to that of finding the matrix of the non-degenerate form. Before solving the latter we introduce the following fundamental definition.

Definition 3.7. Let Φ be a non-degenerate symmetric bilinear form on a real vector space V of dimension n. A basis $(e_i)_{1 \leq i \leq n}$ of V is called orthonormal with respect to Φ, or Φ-orthonormal, if the matrix of Φ with respect to it has the form

$$M(\Phi) = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix}_{n \times n},$$

where I_m denotes an $m \times m$ identity matrix. The ordered pair of integers (n^+, n^-) is called the signature of the form Φ; the vector space V together with the form Φ is called a pseudoeuclidean (or Minkowski) vector space of signature (n^+, n^-).

The following theorem shows that to every pseudoeuclidean vector space corresponds a unique signature.

Theorem 3.11. For every non-degenerate symmetric form Φ on V there exists a Φ-orthonormal basis. The number n^+ in the above definition, and therefore the number $n^- = n - n^+$ remain the same for any Φ-orthonormal basis of V.

Proof. We prove the existence of the basis by induction on $n = \dim(V)$. The statement is obvious for $n = 0$. Assume that it is true for any vector space U with $\dim(U) \leq n-1$. Since Φ is non-degenerate, we can find $x \in V : \Phi(x,x) \neq 0$. Indeed, if $\Phi(x,x) = 0 \quad \forall x \in V$, then from (3.5)

$$\Phi(x,y) = \frac{1}{2}[\Phi(x,x) + \Phi(y,y) - \Phi(x-y,x-y)] = 0 \qquad \forall x,y,$$

and Φ is degenerate.

Let us normalize the chosen vector x:

$$e_1 = \Phi|(x,x)|^{-\frac{1}{2}} \cdot x,$$

$|\Phi(e_1,e_1)| = 1$, and consider a Φ-orthogonal complement of a one-dimensional subspace generated by e_1 : $<e_1>^{\perp}$. By Theorem 3.6

$$\dim(<e_1>^{\perp}) = n - \dim(<e_1>) = n-1.$$

Since $e_1 \notin <e_1>^{\perp}$, by Theorem 3.4

$$V = <e_1> \oplus U,$$

where $U = <e_1>^{\perp}$ and $\Psi \stackrel{\text{def}}{=} \Phi|V \times V$ is non-degenerate. Thus we can apply the inductive hypothesis to U: there exists an orthogonal w.r.t. Ψ basis $(e_i)_{2\le i\le n}$. Adding to it e_1 we obtain Φ-orthonormal basis $(e_i)_{1\le i\le n}$.

To prove the second statement of the theorem, let us set

$$I_+ \stackrel{\text{def}}{=} \{i|\Phi(e_i,e_i) = 1\}, \quad I_- \stackrel{\text{def}}{=} \{i|\Phi(e_i,e_i) = -1\},$$

$$V_+ \stackrel{\text{def}}{=} \bigoplus_{i\in I_+} <e_i>, \quad V_- \stackrel{\text{def}}{=} \bigoplus_{i\in I_-} <e_i>,$$

where $(e_i)_{1\le i\le n}$ is an orthonormal w.r.t. Φ basis of V. If $x\in V_+$, $x \neq 0$, then

$$\Phi(x,x) = \Phi(\sum_{i=1}^{n} x^i e_i, \sum_{i=1}^{n} x^i e_i) = \sum_{i\in I_+} (x^i)^2 > 0,$$

and if $x\in V_-$, $x \neq 0$, then

$$\Phi(x,x) = \Phi(\sum_i x^i e_i, \sum_i x^i e_i) = -\sum_{i\in I_-} (x^i)^2 < 0.$$

Let $(\bar{e}_i)_{1\le i\le n}$ be another orthonormal w.r.t. Φ basis of V, and $\bar{V}_+$, $\bar{V}_-$ are constructed as before. For any $x\in V_+$, $x \neq 0$, $\Phi(x,x) > 0$, and for any $x\in \bar{V}_-$, $x \neq 0$, $\Phi(x,x) < 0$, so that $V_+ \cap \bar{V}_- = \{0\}$. Whence $\dim(V_+) + \dim(\bar{V}_-) = \dim(V_+ + \bar{V}_-) \le \dim(V) = \dim(\bar{V}_+) + \dim(\bar{V}_-)$, and $\dim(V_+) \le \dim(\bar{V}_+)$. Similarly, $\dim(\bar{V}_+) \le \dim(V_+)$, and thus $\dim(\bar{V}_+) = \dim(V_+)$. ∎

It is important to note that in the course of the proof we have constructed a direct decomposition of V

$$V = V_+ \oplus V_-.$$

Although the decomposition is tied to a chosen Φ-orthonormal basis

$(e_i)_{1\le i\le n}$, we always have (assuming that $V = \bar{V}_+ \oplus \bar{V}_-$ is another such decomposition)

$$V_+ \cap \bar{V}_- = \{0\}, \quad V_- \cap \bar{V}_+ = \{0\},$$

$$V_+ - \{0\} \subset \{x\in V \mid \Phi(x,x) > 0\}, \quad V_- - \{0\} \subset \{x\in V \mid \Phi(x,x) < 0\}.$$

The author's interpretation of this fact is that a vector space V with a non-degenerate symmetric bilinear form Φ is constructed from two non-commeasurable vector subspaces V_+ and V_- (see also Def. 3.9) i.e., the homogeneity of the euclidean spaces is lost (compare with the situation in special relativity theory, where one has space-like and time-like vectors).

Theorem 3.12. For every symmetric bilinear form Φ on a vector space V of dimension n, there exists a basis of V with respect to which the matrix of Φ is of the form

$$M(\Phi) = \begin{pmatrix} I_{n^+} & & O \\ & -I_{n^-} & \\ O & & O \end{pmatrix}_{n\times n},$$

where the integers n^+, n^-, and therefore the components of the entire matrix, are uniquely determined by Φ. The pair (n^+,n^-) is called the signature of Φ, and the number $n^+ + n^-$ is called the rank of Φ.

Proof follows from Lemma 3.10 and the previous theorem. ∎

Corollary. a) Φ is positive if and only if $n^+ = n$;
b) Φ is negative if and only if $n^- = n$;
c) Φ is indefinite if and only if $n^+ \ge 1$, $n^- \ge 1$.

Proof. By Theorem 3.12 there exists a basis $(e_i)_{1\le i\le n}$ of V in which the matrix of Φ is of the above form. Thus, if $x = \sum_{i=1}^{n} x^i e_i$, then by (3.3)

$$\Phi(x,x) = (x^1)^2 + \ldots + (x^{n^+})^2 - (x^{n^++1})^2 - \ldots - (x^{n^++n^-})^2,$$

and all three statements follow. ∎

The decomposition $V = V^\perp \oplus V_1$, where $\Phi|V_1 \times V_1$ is non-degenerate

(Lemma 3.10), shows that without loss of generality we can restrict our attention to the case when the given symmetric bilinear form is non-degenerate.

Thus, from now on it is assumed that we are given some fixed non-degenerate symmetric bilinear form Φ of signature (n^+,n^-) on a finite-dimensional real vector space V, which will be called pseudoeuclidean.

The crucial difference between the Euclidean and pseudo-euclidean vector spaces is the existence in the latter of non-zero vectors v such that $\Phi(v,v) = 0$, in other words, the existence of non-zero vectors which are orthogonal to themselves.

Definition 3.8. Let (V,Φ) be a pseudoeuclidean vector space of signature (n^+,n^-). A vector $v \in V$ is called an isotropic vector (relative to Φ) if $\Phi(v,v) = 0$, and the set of all isotropic vectors, which is obviously a union of the lines passing through 0 (i.e., one-dimensional subspaces of V), is called the isotropic cone of Φ.[†] Correspondingly, a vector subspace U of V is said to be an isotropic subspace if it contains a non-zero vector orthogonal to U, non-isotropic otherwise, and totally isotropic if it consists only of isotropic vectors. A totally isotropic subspace is called maximal if it is not properly contained in any totally isotropic subspace. The number $r = \min(n^+,n^-)$ is called the index of Φ.

Theorem 3.13. For a pseudoeuclidean vector space (V,Φ) the following statements are equivalent:

a) U is an isotropic subspace;

b) $\Phi|U \times U$ is degenerate;

c) $U \cap U^\perp \neq \{0\}$;

d) $U + U^\perp \neq V$;

e) $U \oplus U^\perp \neq V$.

Proof follows from Definition 3.8 and Theorem 3.7. ∎

[†] Isotropic vectors and the isotropic cone are sometimes called "light-vectors" and the "light cone" respectively (see for example [35] p. 281). This terminology was adapted from the special theory of relativity, where "space-time" is represented (locally) as R^4 with the Lorentz form (see the remark after Definition 3.5), and the trajectories of the photons in "space-time" are represented by the isotropic lines.

Lemma 3.14. For a pseudoeuclidean vector space (V,Φ) the following statements are equivalent:

a) U is a totally isotropic subspace;

b) $\Phi(x,y) = 0 \quad \forall x,y \in U$;

c) $U \subseteq U^{\perp}$.

Proof follows from Definition 3.8 and the formula

$$\Phi(x,y) = \frac{1}{2}\,[\Phi(x+y,x+y) - \Phi(x,x) - \Phi(y,y)]. \qquad \square$$

Obviously, an isotropic line is a totally isotropic subspace, and a totally isotropic subspace is a subspace contained in the isotropic cone. It is also easy to see that for every subspace U of V the subspace $U \cap U^{\perp}$ is totally isotropic. Indeed, $U \cap U^{\perp} \subseteq U^{\perp} + U$, but $U^{\perp} + U = (U \cap U^{\perp})^{\perp}$ (Theorems 3.9 b), 3.8), and the statement follows from Lemma 3.14 a), c).

One can show (see [34], p. 152) that every maximal totally isotropic subspace has dimension equal to the index of the form Φ.

Finally, in this section we shall consider the notions of a Θ-direct product and a Θ-direct product of two pseudoeuclidean vector spaces (V_1,Φ_1) and (V_2,Φ_2).

Let $V = V_1 \times V_2$ be the direct product of the vector spaces V_1 and V_2, i.e., the set of all pairs (x_1,x_2), where $x_1 \in V_1$, $x_2 \in V_2$, with the following operations

$$(x_1,x_2) + (y_1,y_2) = (x_1+y_1, x_2+y_2),$$

$$c(x_1,x_2) = (cx_1, cx_2).$$

Define the mapping $\Phi^+ : V \times V \longrightarrow \mathbb{R}$ as

$$\Phi^+((x_1,x_2),(y_1,y_2)) \stackrel{\text{def}}{=} \Phi_1(x_1,y_1) + \Phi_2(x_2,y_2) \tag{3.7}$$

for every $(x_1,x_2),(y_1,y_2) \in V$. It is easy to see that Φ^+ is a symmetric bilinear form on V. Indeed, the symmetry is obvious and the linearity can be proved as follows

$$\Phi^+(c(x_1,x_2) + d(y_1,y_2),(z_1,z_2)) = \Phi^+((cx_1+dy_1, cx_2+dy_2),(z_1,z_2)) =$$
$$= \Phi_1(cx_1+dy_1,z_1) + \Phi_2(cx_2+dy_2,z_2) = c\Phi_1(x_1,z_1) + d\Phi_1(y_1,z_1) +$$
$$+ c\Phi_2(x_2,z_2) + d\Phi_2(y_2,z_2) = c[\Phi_1(x_1,z_1) + \Phi_2(x_2,z_2)] +$$

$$+ d[\Phi_1(y_1,z_1) + \Phi_2(y_2,z_2)] = c\Phi^+((x_1,x_2),(z_1,z_2)) +$$
$$+ d\Phi^+(y_1,y_2),(z_1,z_2)).$$

Similarly, we can define a mapping $\Phi^- : V \times V \longrightarrow \mathbb{R}$ as

$$(3.7') \qquad \Phi^-((x_1,x_2)(y_1,y_2) \stackrel{\text{def}}{=} \Phi_1(x_1,y_1) - \Phi_2(x_2,y_2)$$

for every $(x_1,x_2),(y_1,y_2) \in V$. As Φ^+, the form Φ^- is a symmetric bilinear form on V.

Definition 3.9. Vector space $V = V_1 \times V_2$ with the symmetric bilinear form Φ^+ [Φ^-] defined as in (3.7) [(3.7')] will be called the $\oplus$-direct [$\ominus$-direct] product of the pseudoeuclidean vector space (V_1,Φ_1) and (V_2,Φ_2).

It is quite easy to see that if

$$(e'_i)_{1\le i\le n_1} \quad \text{and} \quad (e''_j)_{1\le j\le n_2}$$

are orthonormal bases of V_1 and V_2 respectively, then vectors

$$(e'_1,0),\ldots,(e'_{n_1},0),(0,e''_1),\ldots,(0,e''_{n_2})$$

form a Φ^+ [Φ^-]-orthonormal basis of $V = V_1 \times V_2$. Thus, $\dim(V) = n_1+n_2$, and if the forms Φ_1 and Φ_2 have signatures (n_1^+,n_1^-) and (n_2^+,n_2^-) respectively, then Φ^+ [Φ^-] has signature $(n_1^++n_2^+,n_1^-+n_2^-)$[$(n_1^++n_2^-,n_1^-+n_2^+)$], so that the form Φ^+ [Φ^-] is non-degenerate, i.e., (V,Φ^+) [(V,Φ^-)] is a pseudoeuclidean space. Obviously, the first n_1 vectors of the above basis generate the subspace V' of V isomorphic to V_1, and the last n_2 vectors generate the subspace V'' of V isomorphic to V_2. Thus $V = V' \oplus V''$, where $V' \approx V_1$ and $V'' \approx V_2$.

It is important to note that every pseudoeuclidean space is either Euclidean or a $\ominus$-direct product of two Euclidean spaces. As the reader will see in the next chapter, the latter construction allows one to accommodate in the vector space a much larger variety of metric configurations.

3.2 Geometric interpretation

In this section a geometric illustration of the concepts introduced so far will be given. To this end we consider a pseudo-

euclidean vector space (V,Φ) of signature (2,1), and take a Φ-orthonormal basis $\{e_1,e_2,e_3\}$ of V. Then, if $x = x^1e_1 + x^2e_2 + x^3e_3$, $y = y^1e_1 + y^2e_2 + y^3e_3$, we have

$$\Phi(x,y) = x^1y^1 + x^2y^2 - x^3y^3,$$

so that the equation of the isotropic cone is $(x^1)^2 + (x^2)^2 - (x^3)^2 = 0$.

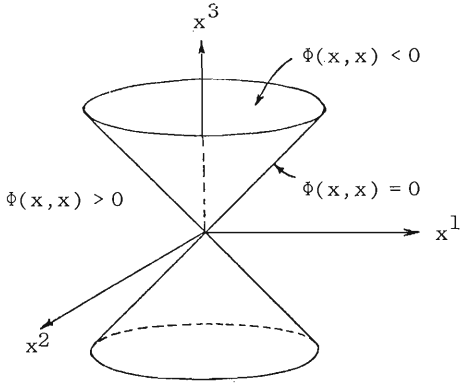


Figure 3.3

The inside of the cone consists of the vectors whose lengths squared are negative ($\|x\|^2 = (x^1)^2 + (x^2)^2 - (x^3)^2 < 0$), and the outside of the isotropic cone consists of those vectors whose lengths squared are positive. The cone itself, of course, consists of vectors of length zero.

What does the orthogonality of several vectors mean geometrically? Here the distinction between the isotropic and non-isotropic vectors is fundamental. Non-isotropic pairwise orthogonal vectors are linearly independent. Indeed, if

$$c_1x_1 + c_2x_2 + \ldots + c_mx_m = 0,$$

then

$$c_1\Phi(x_1,x_j) + c_2\Phi(x_2,x_j) + \ldots + c_m\Phi(x_m,x_j) = \Phi(0,x_j), \qquad 1\le j\le m,$$

and since $\Phi(x_i,x_j) = 0$ for $i \neq j$, we have

$$c_j\Phi(x_j,x_j) = 0.$$

By assumption $\Phi(x_j,x_j) \neq 0$, therefore $c_j = 0 \quad \forall j$, and $\{x_i\}_{1\le i\le m}$ are linearly independent.

From the example below the reader will see that there exists an

infinite set of non-zero orthogonal vectors, but the number of non-isotropic vectors in it is finite (by the above statement).

Let us consider in detail the geometric interpretation of orthogonality for a pair of non-isotropic vectors in the above example. Since we are now interested only in the direction of orthogonal vectors x and y, their lengths could be assumed to be equal to one. In other words, we assume that x and y each lie on one of the two unit "spheres" in $\mathbb{R}^{(n^+,n^-)}$, which are the hyperboloids

$$H_+ = \{x \in V \mid \Phi(x,x) = 1\}, \quad H_- = \{x \in V \mid \Phi(x,x) = -1\}.$$

In Figure 3.4 these are drawn separately for convenience.

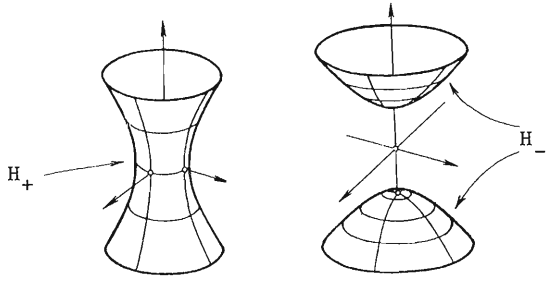


Figure 3.4

The "positive sphere" is a hyperboloid of one sheet that asymptotically approaches the isotropic cone from the outside, while the "negative sphere" is a hyperboloid of two sheets that asymptotically approaches the isotropic cone from the inside. It is easy to see that for the two chosen vectors x and y, $x \perp y$, $x \neq 0$, $y \neq 0$, only two cases are possible: both of them lie on H_+, or they lie on different "spheres". Indeed, the case when $x,y \in H_-$ is impossible: by the above statement x and y are linearly independent and hence cannot be in V_-, since $\dim(V_-) = n^- = 1$.

In the first case, when $x,y \in H_+$, geometric interpretation of their orthogonality is the same as that in the Euclidean space: the corresponding segments-vectors are perpendicular.

In the second case, when, say, $x \in H_+$ and $y \in H_-$, we have

$$(x^1)^2 + (x^2)^2 - (x^3)^2 = 1$$
$$(y^1)^2 + (y^2)^2 - (y^3)^2 = -1$$
$$x^1y^1 + x^2y^2 - x^3y^3 = 0.$$

Multiplying the last equation by -2 and then adding the three equations we obtain

$$(x^1-y^1)^2 + (x^2-y^2) - (x^3-y^3)^2 = 0.$$

The last equation says that the vector x-y lies on the isotropic cone, which means that the two orthogonal non-isotropic vectors $x \in H_+$ and $y \in H_-$ are necessarily situated *symmetrically with respect to the isotropic cone.*

If we take a subspace U of $R^{(2,1)}$ to be an isotropic line (totally isotropic subspace)

$$U = \{x \mid x^2 = 0,\ x^1 = x^3\},$$

then the following equation is obtained for $U^\perp = \{y \mid \Phi(x,y) = 0 \ \forall x \in U\}$:

$$x^1y^1 + x^2y^2 - x^3y^3 = 0 \qquad \forall x \in U.$$

Substituting x^1, x^2, x^3 from the equation of U we get

$$x^1(y^1-y^3) = 0 \qquad \forall x^1 \in \mathbb{R},$$

or $U^\perp = \{y \mid y^1 = y^3,\ y_2 \in \mathbb{R}\}$, i.e., the plane tangent to the isotropic cone at the line U. Thus the plane $U^\perp$ is an isotropic subspace (since $U \subset U^\perp$, see Fig. 3.5).

If we take a subspace W of $\mathbb{R}^{(2,1)}$ to be a non-isotropic line

$$W = \{x \mid x^1 = x^3 = 0,\ x^2 \in \mathbb{R}\},$$

then for W we have the equation

$$x^2y^2 = 0 \qquad \forall x^2 \in \mathbb{R},$$

i.e., $W^\perp = \{y \mid y^2 = 0,\ y^1, y^3 \in \mathbb{R}\}$ is the x^1x^3-plane. This plane is a non-isotropic subspace ($W \cap W^\perp = \{0\}$) containing two isotropic lines:

$$\ell_1 = \{x \mid x^1 = x^3,\ x^2 = 0\}, \quad \ell_2 = \{x \mid x^1 = -x^3,\ x^2 = 0\}.$$

Two dimensional subspaces P of this type (for which $\Phi \mid P \times P$ has signature (1,1) are called hyperbolic planes and play an important role in the linear algebra of a symmetric bilinear form. A "sphere"

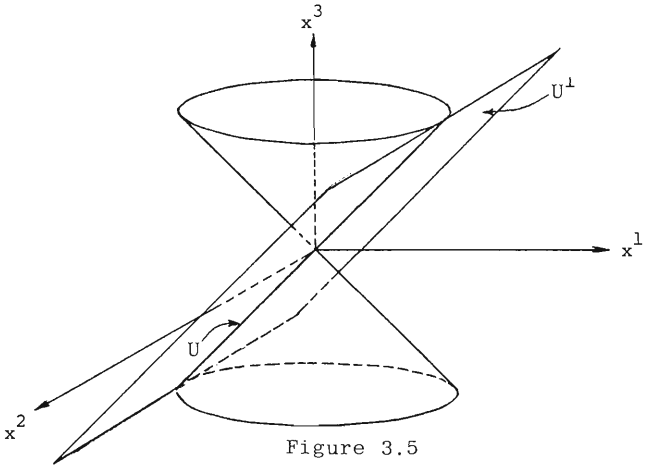


Figure 3.5

of a positive radius in this plane is a horizontal pair of hyperbolas, and a "sphere" of a negative radius is a vertical (conjugate) pair of hyperbolas (see Fig. 3.6).

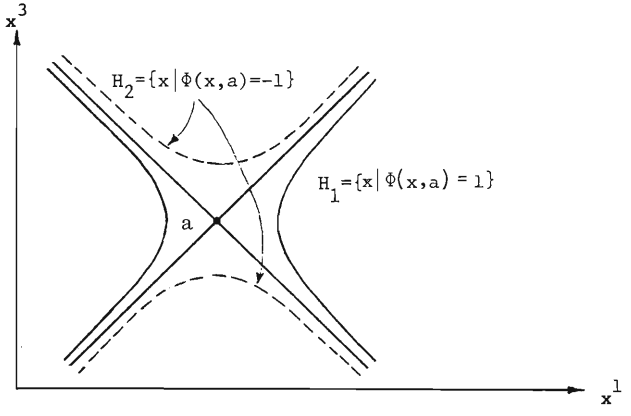


Figure 3.6

3.3 Orthogonal projections and Gram matrices

Let U be a subspace of a pseudoeuclidean vector space (V, Φ). Then, in order to have a decomposition

$$V = U \oplus U^{\perp},$$

the subspace U must be non-isotropic (Theorem 3.13). In this case

every vector $x \in V$ can be uniquely represented as

$$x = x_U + y,$$

where $x_U \in U$ and $y \in U^{\perp}$. It is easy to see that the mapping

$$\pi_U : V \longrightarrow V$$

defined by

$$\pi_U(x) = x_U,$$

is, in fact, an endomorphism of V (Fig. 3.7). From the definition of π_U it follows that

$$\pi_U(V) = U, \quad \pi_U(U^{\perp}) = \{0\}.$$

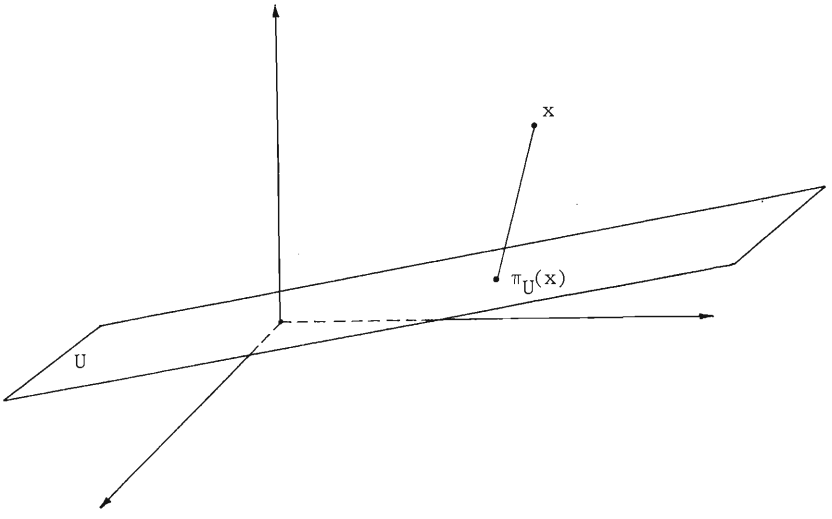


Figure 3.7

Definition 3.10. The endomorphism π_U is called the orthogonal projection onto the non-isotropic subspace U, and the vector $\pi_U(x)$ is called the orthogonal projection of x on U.

One should remember that in a pseudoeuclidean space $\|x-x_U\|^2$ could be negative, in which case the projection x_U has a greater length that that of the vector x itself.

Assume now that the dimension of a non-isotropic subspace U is

equal to m, and that $(e_i)_{1\le i\le m}$ is an orthonormal (w.r.t. $\Phi|U\times U$) basis of U. Then

$$\pi_U(x) = \sum_{i=1}^{m} \Phi(e_i,e_i)\Phi(x,e_i)e_i = \sum_{i\in I_+} \Phi(x,e_i)e_i - \sum_{i\in I_-} \Phi(x,e_i)e_i,$$

where $I_+ = \{1\le i\le m \mid \Phi(e_i,e_i) = 1\}$, $I_- = \{1\le i\le m \mid \Phi(e_i,e_i) = -1\}$.

Indeed, $x = \sum_{i=1}^{m} x^i e_i$ implies that $\Phi(x,e_j) = \Phi(\sum_{i=1}^{m} x^i e_i, e_j) = \sum_{i=1}^{m} x^i \Phi(e_i,e_j) = x^j \Phi(e_j,e_j)$, whence $x^j = \Phi(e_j,e_j)\Phi(x,e_j)$.

To obtain an expression for the orthogonal projection in the event $\Phi(x,e_i)$'s are not known, but only the $\Phi(x,a_i)$'s are given (where $(a_i)_{1\le i\le m}$ is an arbitrary basis), the following definition will be useful.

Definition 3.11. Let $x_1,x_2,\ldots,x_m$ be vectors of a pseudoeuclidean vector space (V,Φ). The matrix

$$G(x_1,x_2,\ldots,x_m) \stackrel{\text{def}}{=} (\Phi(x_i,x_j))_{1\le i,j\le m}$$

is called the Gram matrix of $x_1,\ldots,x_m$, and its determinant

$$g(x_1,\ldots,x_m) \stackrel{\text{def}}{=} \det(G(x_1,\ldots,x_m))$$

is called the Gram determinant of $x_1,\ldots,x_m$.

Obviously, if $(a_i)_{1\le i\le n}$ is a basis of V, the Gram matrix $G(a_1,\ldots,a_n)$ is the matrix of the form Φ with respect to this basis. Hence, the fact that, in general, the Gram matrix $G(x_1,\ldots,x_m)$ contains the complete metric information about the vectors $x_1,\ldots,x_m$ should not be surprising. In particular we have

Theorem 3.15. Vectors $x_1,\ldots,x_m$ form a basis of an m-dimensional non-isotropic subspace if and only if $g(x_1,\ldots,x_m) \neq 0$.

Proof. Part $\Longrightarrow$ of the statement follows from the definition of a non-isotropic subspace, so we have to prove the part $\Longleftarrow$ only.

Let $(e_i)_{1\le i\le n}$ be an orthonormal basis of (V,Φ), and let $x_j = \sum_{i=1}^{n} x_j^i e_i$ $(1\le j\le m)$. Then from (3.3) it follows that

$$G(x_1,\ldots,x_m) = {}^tX \cdot M(\Phi) \cdot X,$$

where the j^{th} column of the matrix X is the coordinate column of x_j $(1 \le j \le m)$, and $M(\Phi)$ is the matrix of Φ w.r.t. (e_i). Since $M(\Phi)$ is invertible,

$$\operatorname{rank}(M(\Phi) \cdot X) = \operatorname{rank}(X).$$

But the rank of the product is not greater than the minimum of the factor ranks, so that

$$\operatorname{rank}(G(x_1,\ldots,x_m)) \le \min(\operatorname{rank}({}^tX),\ \operatorname{rank}(M(\Phi) \cdot X)) = \operatorname{rank}(X).$$

By the assumption $g(x_1,\ldots,x_m) \neq 0$, and thus

$$m = \operatorname{rank}(G(x_1,\ldots,x_m)) \le \operatorname{rank}(X) \le m.$$

Therefore, $\operatorname{rank}(X) = m$, and x_j, $1 \le j \le m$, are linearly independent. The given condition, $g(x_1,\ldots,x_m) \neq 0$, means that the form $\Psi = \Phi | U \times U$, where $U = \langle x_1,\ldots,x_m \rangle$, is non-degenerate, i.e., U is non-isotropic. ∎

Now we are ready to establish formulas promised before Definition 3.11.

Theorem 3.16. Let U be an m-dimensional non-isotropic subspace of a pseudoeuclidean space (V,Φ), and let (a_k) be a basis of U (not necessarily orthonormal). Then $\forall x \in V$ the coordinate of the orthogonal projection $\pi_U(x) = u$ w.r.t. a fixed orthonormal basis (e_i) of U are

$$u^i = -\frac{\det \begin{pmatrix} & & a_1^i \\ & G(a_1,\ldots,a_m) & a_2^i \\ & & \vdots \\ & & a_m^i \\ \Phi(x,a_1) \ldots \Phi(x,a_m) & & 0 \end{pmatrix}}{g(a_1,\ldots,a_m)} \qquad (1 \le i \le m),$$

where a_k^i, $1 \le i \le m$, are the coordinates of a_k w.r.t. $(e_i)_{1 \le i \le m}$.

Proof. Since x-u is orthogonal to U

$$\Phi(u-x,\ a_j) = 0 \qquad (1 \le j \le m),$$

or, substituting $u = \sum_{k=1}^{m} c^k a_k$, we obtain

$$\sum_{k=1}^{m} \Phi(a_k, a_j)c^k - \Phi(x, a_j) = 0 \qquad (1 \le j \le m).$$

Adding to the above system equation $u^i = \sum_{k=1}^{m} c^k a_k^i$, we get the following system

$$\begin{cases} \Phi(a_1, a_1)c^1 + \ldots + \Phi(a_m, a_1)c^m + \Phi(x, a_1)(-1) = 0 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \Phi(a_1, a_m)c^1 + \ldots + \Phi(a_m, a_m)c^m + \Phi(x, a_m)(-1) = 0 \\ a_1^i c^1 + \ldots + a_m^i c^m + u^i(-1) = 0. \end{cases}$$

Consider this system as a system of m+1 linear homogeneous equations in m+1 unknowns $y_1, \ldots, y_{m+1}$, which replace in the above system $c^1, \ldots, c^m, -1$. Since the system has a non-zero solution $(c^1, \ldots, c^m, -1)$, its determinant must be zero:

$$\det \begin{pmatrix} \Phi(a_1, a_1) \ldots \Phi(a_m, a_1) & \Phi(x, a_1) \\ \cdots\cdots\cdots\cdots\cdots & \cdots \\ \Phi(a_1, a_m) \ldots \Phi(a_m, a_m) & \Phi(x, a_m) \\ a_1^i \quad \ldots \quad a_m^i & u^i \end{pmatrix} = 0$$

The last row in the above matrix can be decomposed as

$$a_1^i + 0, \ldots, a_m^i + 0, \; 0 + u^i,$$

and thus the above equation can be rewritten as

$$\det \left(\begin{array}{ccc|c} & G(a_1, \ldots, a_m) & & \Phi(x, a_1) \\ & & & \vdots \\ a_1^i & \ldots & a_m^i & \Phi(x, a_m) \\ \hline 0 & \ldots & 0 & u^i \end{array} \right) = 0$$

The determinant on the right is equal to $\det(G(a_1, \ldots, a_m)) \cdot u^i = g(a_1, \ldots, a_m) \cdot u^i$, so that the sought formula follows from the above equation, by replacing the matrix on the left by its transpose. ∎

It is also easy to devise another formula for the above vector ${}^t u = (u^1, \ldots, u^m)$:

(3.8) $$u = A_0 \cdot [G(a_1,\ldots,a_m)]^{-1} \cdot b,$$

where ${}^t b = (b^1,\ldots,b^m)$, $b^j = \Phi(x,a_j)$, $1 \le j \le m$, and A_0 is the square matrix, whose k^{th} column is formed by the coordinates of a_k w.r.t. (e_i). Indeed, as in the above proof

$$\sum_{k=1}^{m} \Phi(a_k,a_j)c^k = \Phi(x,a_j) \qquad (1 \le j \le m),$$

or

$$G(a_1,\ldots,a_n)\cdot c = b,$$

where ${}^t c = (c^1,\ldots,c^m)$, $u = \sum_{k=1}^{m} c^k a_k$. Whence, $c = [G(a_1,\ldots,a_m)]^{-1} \cdot b$, and the required formula follows from the last one by transition to the Φ-orthonormal basis $(e_i)_{1 \le i \le m}$.

One should note that the above formulas are valid for any vector x from V: x may belong to U, or it may be an isotropic vector of V.

By analogy with the Euclidean vector spaces we introduce the following fundamental definition.

Definition 3.12. Let U be a non-isotropic subspace of a pseudo-euclidean vector space V. The square of the distance from a vector $x \in V$ to the subspace U, d(x,U), is the squared norm of the vector x-u, where $u = \pi_U(x)$:

$$[d(x,U)]^2 \stackrel{def}{=} \|x-u\|^2 = \Phi(x-u,x-u).$$

Contrary to the Euclidean case,

$$\|x-u\|^2 \neq \inf\{d^2(x,u) \mid u \in U\} = -\infty,$$

but, as in the Euclidean space, this is a distance measured from x to U in the direction orthogonal to U.

Having obtained above two closed form expressions for $u = \pi_U(x)$ it is not difficult to find corresponding expressions for the introduced distance as well.

Theorem 3.17. Under the conditions of Theorem 3.16

$$d^2(x,U) = \frac{g(a_1,a_2,\ldots,a_m,x)}{g(a_1,a_2,\ldots,a_m)} .$$

Proof. Let as in the last theorem $u = \sum_{i=1}^{m} u^i e_i$, then

$$d^2(x,U) = \Phi(x-u,x-u) = \Phi(x-u,x) - \Phi(x-u,u) = \Phi(x-u,x) =$$
$$= \Phi(x,x) - \Phi(u,x) = [g(a_1,\ldots,a_m)]^{-1}\cdot[g(a_1,\ldots,a_m)\Phi(x,x) -$$
$$- g(a_1,\ldots,a_m)\cdot\Phi(\sum_{i=1}^{m} u^i e_i, x)].$$

Using Theorem 3.16 and remembering that $a_k = \sum a_k^i e_i$ the expression in the last square brackets can be written as

$$\Phi(x,x)\cdot g(a_1,\ldots,a_m) - g(a_1,\ldots,a_m)\sum_{i=1}^{m} u^i\cdot\Phi(e_i,x) = \Phi(x,x)\cdot g(a_1,\ldots,a_m)$$

$$+ \sum_{i=1}^{m} \det\begin{pmatrix} G(a_1,\ldots,a_m) & \begin{matrix} a_1^i\cdot\Phi(e_i,x) \\ \vdots \\ a_m^i\cdot\Phi(e_i,x) \end{matrix} \\ \Phi(x,a_1)\ \ldots\ \Phi(x,a_m) & 0 \end{pmatrix} = \Phi(x,x)\cdot g(a_1,\ldots,a_m) +$$

$$+ \det\begin{pmatrix} G(a_1,\ldots,a_m) & \begin{matrix} \sum_{i=1}^{m} a_1^i\cdot\Phi(e_i,x) \\ \vdots \\ \sum_{i=1}^{m} a_m^i\cdot\Phi(e_i,x) \end{matrix} \\ \Phi(x,a_1)\ \ldots\ \Phi(x,a_m) & 0 \end{pmatrix} = \Phi(x,x)\cdot g(a_1,\ldots,a_m) +$$

$$+ \det\begin{pmatrix} G(a_1,\ldots,a_m) & \begin{matrix} \Phi(a_1,x) \\ \vdots \\ \Phi(a_m,x) \end{matrix} \\ \Phi(x,a_1)\ \ldots\ \Phi(x,a_m) & 0 \end{pmatrix} = g(a_1,a_2,\ldots,a_m,x),$$

and the theorem is proved. ∎

The second expression for the distance (in the notation of Theorem 3.16 and the statement following it) is

$$d^2(x,U) = \Phi(x,x) - {}^t b\cdot[G(a_1,\ldots,a_m)]^{-1}\cdot b. \tag{3.9}$$

To obtain the formula it is enough to substitute $c = [G(a_1,\ldots,a_m)]^{-1}\cdot b$ in the expression

$$d^2(x,U) = \Phi(x,x) - \Phi(u,x) = \Phi(x,x) - \Phi(\sum_{k=1}^{m} c^k a_k, x) =$$

$$= \Phi(x,x) - \sum_{k=1}^{m} c^k \Phi(a_k,x) = \Phi(x,x) - {}^t b \cdot c.$$

3.4 Self-adjoint, orthogonal, and positive endomorphisms of a pseudoeuclidean space

The main purpose of this section is to establish basic properties of the endomorphisms (linear operators) of $\mathbb{R}^{(n^+,n^-)}$ which are analogues of orthogonal and positive endomorphisms (having positive matrices) of a Euclidean space.

First, we introduce a fundamental notion of the adjoint endomorphism of an endomorphism in $\mathbb{R}^{(n^+,n^-)}$.

Definition 3.13. Let $\gamma : V \longrightarrow V$ be an endomorphism of a pseudoeuclidean vector space (V,Φ). Take any $y \in V$, then the mapping $\psi_y(x) \stackrel{\text{def}}{=} \Phi(\gamma(x),y)$ is a linear form on V, therefore by Theorem 3.3 there exists a unique $\bar{y} \in V$ such that $\psi_y(x) = \Phi(x,\bar{y})$. It is easy to see that the mapping $\gamma^* : V \longrightarrow V$ defined as $\gamma^*(y) = \bar{y}$, or equivalently as

$$\Phi(\gamma(x),y) = \Phi(x,\gamma^*(y)) \qquad \forall x,y \in V \tag{3.10}$$

is an endomorphism of V. It is called the adjoint of the endomorphism γ with respect to the form Φ, or simply Φ-adjoint of γ.

Denoting by $M(\gamma)$ the matrix of γ w.r.t. some basis of V, we have the following relationship between the matrices of γ and γ^*.

Theorem 3.18. An endomorphism $\delta : V \longrightarrow V$ is the adjoint of γ w.r.t. Φ if and only if

$$M(\delta) = [M(\Phi)]^{-1} \cdot {}^t[M(\gamma)] \cdot M(\Phi),$$

where all the matrices are computed in the same basis $(a_i)_{1 \le i \le n}$.

Proof. By (3.10) δ is the Φ-adjoint of the endomorphism γ if and only if

$$\Phi(\gamma(x),y = \Phi(x,\delta(y)) \qquad \forall x,y \in V,$$

which by (3.3) is equivalent to

$$ {}^t\begin{pmatrix} y^1 \\ \vdots \\ y^n \end{pmatrix} M(\Phi)M(\gamma) \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} = {}^t\left[M(\delta) \begin{pmatrix} y^1 \\ \vdots \\ y^n \end{pmatrix} \right] \cdot M(\Phi) \cdot \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} \qquad \forall (x_i),(y_i) \in \mathbb{R}^n, $$

or

$$ {}^t\begin{pmatrix} y^1 \\ \vdots \\ y^n \end{pmatrix} M(\Phi)M(\gamma) \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} = {}^t\begin{pmatrix} y^1 \\ \vdots \\ y^n \end{pmatrix} {}^t[M(\delta)] \cdot M(\Phi) \begin{pmatrix} x^1 \\ \vdots \\ x^n \end{pmatrix} \qquad \forall (x_i),(y_i) \in \mathbb{R}^n. $$

The last statement is true if and only if

$$ M(\Phi) \cdot M(\gamma) = {}^t[M(\delta)] \cdot M(\Phi), $$

or

$$ {}^t[M(\gamma)] \cdot {}^t[M(\Phi)] = {}^t[M(\Phi)] \cdot M(\delta). $$

But ${}^t M(\Phi) = M(\Phi)$, therefore the last equality becomes

$$ {}^t[M(\gamma)] \cdot M(\Phi) = M(\Phi) \cdot M(\delta), $$

from which the theorem follows. ∎

Since in the Euclidean vector space (V,Φ) the matrix $M(\Phi)$ w.r.t. an orthonormal basis is an identity matrix, we obtain, as a corollary, a well known result:

$$ M(\gamma^*) = {}^t[M(\gamma)], $$

where γ^* is the adjoint of γ w.r.t. a positive form Φ, and the two matrices are computed in the same Φ-orthonormal basis.

Main algebraic properties of the operation * are given in the theorem below.

Theorem 3.19. Let γ_1 and γ_2 be endomorphisms of a pseudoeuclidean vector space (V,Φ), then

a) $(c_1\gamma_1 + c_2\gamma_2)^* = c_1\gamma_1^* + c_2\gamma_2^* \qquad \forall c_1, c_2 \in \mathbb{R}$

b) $(\gamma_2 \circ \gamma_1)^* = \gamma_1^* \circ \gamma_2^*$

c) $(\gamma_1^*)^* = \gamma_1$.

Proof. Since the proofs of all three statements are similar, we shall give the proof of b) only. By (3.10) we have

$$\Phi(x,(\gamma_2\circ\gamma_1)^*(y)) = \Phi(\gamma_2(\gamma_1(x)),y) = \Phi(\gamma_1(x),\gamma_2^*(y)) = \Phi(x,\gamma_1^*(\gamma_2^*(y))). \quad \blacksquare$$

Next, we introduce another fundamental notion, that of an orthogonal endomorphism of $\mathbb{R}^{(n^+,n^-)}$.

Definition 3.14. An endomorphism γ of a pseudoeuclidean vector space (V,Φ) is called orthogonal with respect to the form Φ, (Φ-orthogonal), or a linear isometry of the space (V,Φ), if

$$\Phi(\gamma(x),\gamma(y)) = \Phi(x,y) \qquad \forall x,y\in V.$$

The term linear isometry is explained by the fact that a linear operator γ is orthogonal w.r.t. Φ if and only if it preserves all the distances in (V,Φ) (see Def. 3.2). Indeed,

$$\Phi(\gamma(x)-\gamma(y),\gamma(x)-\gamma(y)) = \Phi(\gamma(x),\gamma(x)) + \Phi(\gamma(y),\gamma(y)) - 2\Phi(\gamma(x),\gamma(y)),$$

so that

$$\|\gamma(x)-\gamma(y)\|^2 = \|x-y\|^2 \quad \forall x,y\in V \iff \Phi(\gamma(x),\gamma(y)) = \Phi(x,y) \qquad \forall x,y.$$

From the definition it also follows that a Φ-orthogonal endomorphism is, in fact, an automorphism (one-to-one). Indeed, if $\gamma(x) = \gamma(y)$, then $\gamma(x-y) = 0$, so that $\Phi(x-y,z) = \Phi(\gamma(x-y),\gamma(z)) = 0$ for any $z\in V$, and since Φ is a non-degenerate form we conclude that $x-y = 0$, i.e., $x = y$.

It is also easy to see that for a linear isometry

$$\begin{aligned} \|x\|^2 = \Phi(x,x) > 0 &\implies \|\gamma(x)\|^2 > 0, \\ \|x\|^2 < 0 &\implies \|\gamma(x)\|^2 < 0, \\ \|x\| = 0 &\implies \|\gamma(x)\| = 0. \end{aligned}$$

Characteristic value c of a Φ-orthogonal endomorphism γ whose corresponding characteristic subspace contains at least one non-isotropic vector v satisfies: $|c| = 1$. Indeed, from

$$\gamma(v) = cv$$

it follows that

$$\Phi(\gamma(v),\gamma(v)) = c^2\Phi(v,v),$$

and since $\Phi(v,v) \neq 0$ we get $c^2 = 1$.

Totally isotropic characteristic subspaces of a Φ-orthogonal endomorphism (and only they) may correspond to a characteristic value c such that $|c| \neq 1$, as shown by the example below.

Theorem 3.20. The following statements are equivalent:

a) automorphism $\gamma : V \longrightarrow V$ is orthogonal with respect to Φ;
b) ${}^t[M(\gamma)]\cdot M(\Phi)\cdot M(\gamma) = M(\Phi)$, where all matrices are computed in the same basis;
c) $\gamma^* = \gamma^{-1}$.

Proof. a) $\Longleftrightarrow$ b) follows from Definition 3.14 and (3.3).
a) $\Longleftrightarrow$ c) follows from the formula $\Phi(\gamma(x),\gamma(y)) = \Phi(x,\gamma^*(\gamma(y)))$. ∎

Corollary. The set of all Φ-orthogonal automorphisms denoted as $O(V,\Phi)$, is a subgroup of the group GL(V) of all automorphisms of V.

Proof. To prove that $O(V,\Phi)$ is a subgroup of GL(V) it is enough to show that if $\gamma,\delta\in O(V,\Phi)$ then $\gamma\delta^{-1}\in O(V,\Phi)$. By Theorem 3.20 this is equivalent to $(\gamma\delta^{-1})^* = (\gamma\delta^{-1})^{-1}$, which is true by Theorems 3.19b), 3.20:

$$(\gamma\delta^{-1})^* = (\delta^{-1})^*\gamma^* = (\delta^{-1})^{-1}\gamma^{-1} = (\gamma\delta^{-1})^{-1}.$$ ∎

Definition 3.15. The group $O(V,\Phi)$ is called the orthogonal group of the pseudoeuclidean vector space (V,Φ), and is also denoted $O(\mathbb{R}^{(n^+,n^-)})$ or simply $O(n^+,n^-)$.

It is easy to see that $\gamma\in O(V,\Phi)$ if and only if

$$\Phi(\gamma(a_i),\gamma(a_j)) = \Phi(a_i,a_j) \qquad 1\leq i,j\leq n$$

for a basis $(a_i)_{1\leq i\leq n}$ of V. In particular, $\gamma\in O(V,\Phi)$ if and only if $(\gamma(e_i))_{1\leq i\leq n}$ is a Φ-orthonormal basis for a Φ-orthonormal basis $(e_i)_{1\leq i\leq n}$.

Definition 3.16. Let $M'(\gamma)$ and $M''(\gamma)$ be the matrices of an endomorphism $\gamma : V \longrightarrow V$ w.r.t. bases (a_i') and (a_i'') correspondingly. Then $M''(\gamma) = T\cdot M'(\gamma)\cdot T^{-1}$, where T is the matrix of transition from the basis (a_i') to the basis (a_i''), so that $\det(M'(\gamma)) = \det(M''(\gamma))$. The number $\det(\gamma) = \det(M(\gamma))$ (which is independent of the basis) is

called the determinant of the endomorphism γ.

Theorem 3.21. If γ is an endomorphism of a pseudoeuclidean space (V,Φ), then

$$\det(\gamma) = \det(\gamma^*),$$

where γ^* is the adjoint of the endomorphism γ with respect to Φ.

Proof follows from Theorem 3.18. ∎

Corollary. If γ is a Φ-orthogonal endomorphism, then

$$(\det(\gamma))^2 = 1.$$

Proof. We have $\gamma\gamma^{-1} = \iota_V$. But, by Theorem 3.20 $\gamma^* = \gamma^{-1}$, so that substituting γ^* instead of γ^{-1} in the former equation, equating the determinants of both sides, and using the above theorem, we obtain the desired result. ∎

Next, to illustrate some of the above notions we describe the orthogonal group of $\mathbb{R}^{(1,1)}$. According to Theorem 3.20b) $\gamma\in O(\mathbb{R}^{(1,1)})$ with $M(\gamma) = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ if and only if

$$\begin{pmatrix} a & c \\ b & d \end{pmatrix}\cdot\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\cdot\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

or

$$\begin{pmatrix} a^2-c^2 & ab-cd \\ ab-cd & b^2-d^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

This matrix equation is equivalent to the system

$$\begin{cases} a^2-c^2 = 1 \\ ab-cd = 0 \\ b^2-d^2 = -1. \end{cases}$$

Since from the first equation it follows that $a \neq 0$, we substitute $b = cd/a$ into the last equation:

$$\frac{c^2d^2}{a^2} - d^2 = -1,$$

or

$$\frac{d^2}{a^2}(c^2-a^2) = -1,$$

which using the first equation becomes

$$\frac{d^2}{a^2} = 1,$$

i.e., $d = \pm a$. If $d = a$, then $b = c$, and $\det(\gamma) = a^2-c^2 = 1$, which means that γ preserves the orientation; if $d = -a$, then γ does not preserve orientation. Thus, the above system is equivalent to the following one

$$\begin{cases} a^2-c^2 = 1 \\ d = \pm a, \ b = \pm c. \end{cases}$$

The general solution of the equation $a^2-c^2 = 1$ is

$$(*) \qquad \begin{aligned} a &= \pm \cosh r \\ c &= \pm \sinh r, \end{aligned}$$

where $r \in \mathbb{R}$. Obviously, these four solutions for a and c are, in fact, only two solutions: if $a > 0$ we take

$$\begin{aligned} a &= \cosh r \\ c &= \sinh r \end{aligned} \qquad r \in \mathbb{R},$$

and if $a < 0$ we take

$$\begin{aligned} a &= -\cosh r \\ c &= -\sinh r \end{aligned} \qquad r \in \mathbb{R}.$$

The other two combinations of signs in (*) reduce to the above two cases by taking the real parameters to be $-r$. Thus, we have found matrices of all automorphisms from $O(\mathbb{R}^{(1,1)})$.

Let us take a closer look at the subgroup $O_+(\mathbb{R}^{(1,1)})$ of those linear isometries which preserve the orientation: $d = a$, $b = c$. To this end we choose a Φ-orthonormal basis $\{e_1, e_2\}$ of $\mathbb{R}^{(1,1)}$ such that e_1 is perpendicular to e_2 in the Euclidean plane. The possibility of such a choice follows from the fact that any two vectors that are symmetric w.r.t. the isotropic cone (consisting in this case of two perpendicular lines: $x^1 = x^2$, $x^1 = -x^2$) are orthogonal w.r.t. the form of signature (1,1) (see Section 3.2). "Rotating" the chosen basis by acting on it by an orthogonal automorphism $\gamma \in O_+(\mathbb{R}^{(1,1)})$ we obtain Φ-orthonormal basis $\{\gamma(e_1), \gamma(e_2)\}$:

$$\gamma(e_1) = ae_1 + ce_2$$
$$\gamma(e_2) = be_1 + de_2,$$

and since d = a, b = c we have

$$\gamma(e_1) = ae_1 + ce_2$$
$$\gamma(e_2) = ce_1 + ae_2.$$

It is easy to see that if $a > 0$, then the Euclidean angle between e_1 and $\gamma(e_1)$ (which equals the angle between e_2 and $\gamma(e_2)$) is acute, and if $a < 0$, then it is an obtuse angle. To show that, it is enough to compute the Euclidean inner product of e_1 and $\gamma(e_1)$:

$$(e_1 \mid \gamma(e_1)) = (e_1 \mid ae_1 + ce_2) = a.$$

Similarly, one can interpret number c:

$$(e_1 \mid \gamma(e_2)) = (e_2 \mid \gamma(e_1)) = c,$$

i.e., the angle between e_1 and $\gamma(e_2)$ (which equals the angle between e_2 and $\gamma(e_1)$) is acute or obtuse according to the sign of the number c. Thus, all orientation preserving linear isometries of the hyperbolic plane can be represented pictorially (by action on the orthogonal basis e_1, e_2) as in Fig. 3.8.

To get a still better idea about the linear isometries of the hyperbolic plane we find the characteristic values and vectors of the top two linear isometries in Figure 3.8. The matrix of the first one is (see above).

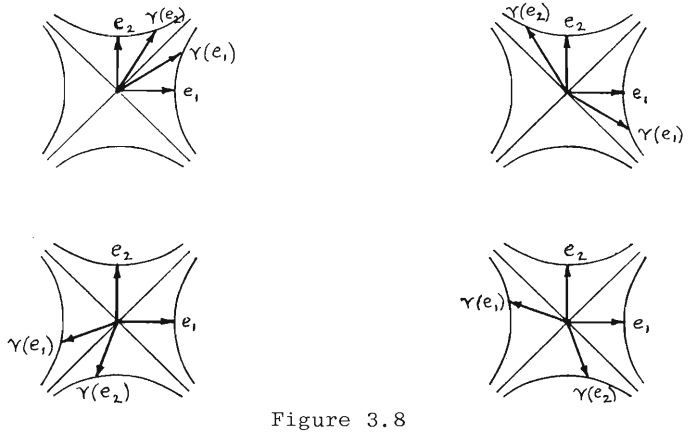


Figure 3.8

$$M(\gamma) = \begin{pmatrix} \cosh r & \sinh r \\ \sinh r & \cosh r \end{pmatrix},$$

where the number r is called the hyperbolic angle; the characteristic values are

$$c_1 = \cosh r + \sinh r$$

$$c_2 = \cosh r - \sinh r,$$

and the characteristic vectors lie on lines $x^1 = x^2$ and $x^1 = -x^2$ respectively. Since $c_1 c_2 = \det \gamma = 1$, we see that in the first case (Fig. 3.9) the linear isometry of $\mathbb{R}^{(1,1)}$ "hyperbolically" (along the hyperboles) stretches the plane along the line $x^1 = x^2$ by a factor of $c_1 > 1$, and "hyperbolically" compresses the plane by the same factor to the line $x^1 = -x^2$ (which is the same, "stretches" by a factor $c_2 = \frac{1}{c_1} < 1$).

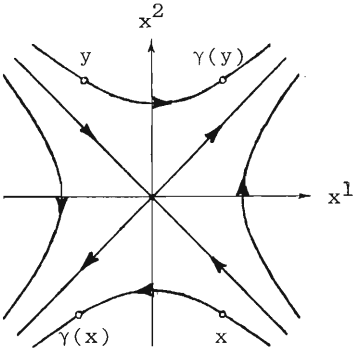


Figure 3.9

In the second case the linear isometry γ has the same matrix, but with negative hyperbolic angle: $r < 0$. Now, of course, $c_1 < 1$, $c_2 > 1$, so that the verbs "stretches" and "compresses" above interchange, and the directions of the arrows in the Fig. 3.9 change to the opposite.

Visualizing an orientation preserving linear isometry in $\mathbb{R}^{(n^+,n^-)}$ when $n^+, n^- > 2$, it is useful to remember that for every

$a \in \mathbb{R}$, $a > 0$ each of the two dual hyperboloids

$$\sum_{i=1}^{n^+} (x^i)^2 - \sum_{i=n^++1}^{n^++n^-} (x^i)^2 = a$$

$$\sum_{i=1}^{n^+} (x^i)^2 - \sum_{i=n^++1}^{n^++n^-} (x^i)^2 = -a$$

is a connected set, invariant under the Φ-orthogonal endomorphisms. These sets are analogues of the spheres when Euclidean rotations are considered, but contrary to the latter they are not compact sets, which creates a fundamental difference between the Euclidean and non-Euclidean theories, in spite of the fact that "events" in both of them "take place" in the same vector space.

Concluding the geometric description we note that with some exceptions sets that are congruent in $\mathbb{R}^{(n^+,n^-)}$ are not congruent in $\mathbb{R}^{(s,t)}$, where $n^+ + n^- = s+t$. Thus, for example, the two plane figures in Fig. 3.10 are congruent in $\mathbb{R}^{(1,1)}$, but, obviously, are not congruent in the Euclidean plane $\mathbb{R}^2$.

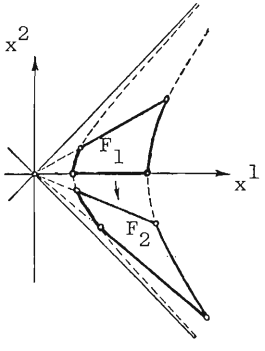


Figure 3.10

Definition 3.17. An endomorphism γ of a pseudoeuclidean vector space (V, Φ) is called self-adjoint with respect to the form Φ, or simply Φ-self-adjoint, if

$$\gamma^* = \gamma.$$

A Φ-self-adjoint endomorphism γ is called non-negative (positive) w.r.t. Φ if

$$\Phi(\gamma(x),x) \geq 0 \qquad x\epsilon V$$

$$(\Phi(\gamma(x),x) > 0 \qquad x\epsilon V,\ x \neq 0).$$

We shall use the term non-negative (positive)[†)] without indicating the form Φ only when the Euclidean form is meant.

Every Φ-positive endomorphism γ is an automorphism, since $\text{Null}(\gamma) = \{0\}$. An important property of Φ-positive endomorphisms is that their sets of characteristic vectors do not contain non-zero isotropic vectors. To show this, it is enough to note, that if $x\epsilon V(c;\gamma)$, then $\Phi(\gamma(x),x) = \Phi(cx,x) = c\Phi(x,x) > 0$.

The following statements allow one, given all the necessary matrices, to check if an endomorphism γ is Φ-self-adjoint, Φ-non-negative (Φ-positive).

Theorem 3.22. An endomorphism $\gamma : V \longrightarrow V$ is self-adjoint with respect to the form Φ if and only if

$$M(\Phi)\cdot M(\gamma) \quad \text{is a symmetric matrix,}$$

where the two matrices are computed in the same basis.

Proof. From Theorem 3.18 it follows that γ is self-adjoint w.r.t. Φ if and only if

$$M(\Phi) \cdot M(\gamma) = {}^{t}[M(\gamma)] \cdot M(\Phi).$$

But $M(\Phi) = {}^{t}[M(\Phi)]$, so that the expression on the right equals ${}^{t}[M(\Phi) \cdot M(\gamma)]$, and the statement follows. ∎

A well known statement (Euclidean case) is obtained by setting $M(\Phi) = I$.

Theorem 3.23. An endomorphism $\gamma : V \longrightarrow V$ is Φ-non-negative (Φ-positive) if and only if

$$M(\Phi) \cdot M(\gamma) = P,$$

where P is a non-negative (positive) matrix[†)], and all the matrices are computed in the same basis.

Proof. From Definition 3.17, the last theorem, and formula (3.3) it follows that γ is non-negative, w.r.t. Φ if and only if $M(\Phi) \cdot M(\gamma)$

[†)]Often the terms positive (positive-definite) are used.

is symmetric and

$$ {}^{t}\begin{pmatrix} x^1 \\ x^2 \\ \vdots \\ x^n \end{pmatrix} \cdot M(\Phi) \cdot M(\gamma) \cdot \begin{pmatrix} x^1 \\ x^2 \\ \vdots \\ x^n \end{pmatrix} \geq 0 \qquad \forall (x^1, \ldots, x^n) \in \mathbb{R}^n $$

This means that matrix $M(\Phi) \cdot M(\gamma) = P$ is non-negative, and the theorem follows. For a Φ-positive endomorphism γ the proof is identical (with obvious modifications). ∎

Corollary. Under conditions of the above theorem, γ is non-negative (positive) w.r.t. Φ of signature (n^+, n^-) if and only if

$$ J \cdot M(\gamma) = P, $$

where $M(\gamma)$ is the matrix of γ in a Φ-orthonormal basis, P is a non-negative (positive) matrix, and

$$ J = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix}. $$

∎

The next theorem, as well as Theorem 3.26, are of special importance to the development of the proposed theory (see chapter 5).

Theorem 3.24. All characteristic values of a non-negative w.r.t. Φ endomorphism $\gamma : V \longrightarrow V$ are real numbers.

Proof. Let us choose a Φ-orthonormal basis $(e_i)_{1 \leq i \leq n}$. Then, from the above corollary it follows that w.r.t. this basis

$$ M(\gamma) = J \cdot P. $$

By a well known theorem ([33], p. 605) there exists an orthogonal (with respect to the Euclidean form) matrix U, such that

$$ U^{-1} \cdot P \cdot U = D, $$

where D is the diagonal matrix with the characteristic values of P on the main diagonal. P is a non-negative matrix, so the elements of D are non-negative numbers, and without loss of generality we can

assume that $D = \begin{pmatrix} O & O \\ O & D_1 \end{pmatrix}$, where D_1 is a diagonal matrix with positive entries on the diagonal. Then, matrix $M(\gamma) = J \cdot P$ is similar to the matrix $M = (U^{-1}JU) \cdot (U^{-1}PU) = A \cdot D$, where $A = U^{-1}JU$ is symmetric. So that

$$M = A \cdot D = \begin{pmatrix} A_1 & A_2 \\ {}^tA_2 & A_3 \end{pmatrix} \begin{pmatrix} O & O \\ O & D_1 \end{pmatrix} = \begin{pmatrix} O & A_2 \cdot D_1 \\ O & A_3 \cdot D_1 \end{pmatrix}.$$

Thus, all non-zero characteristic values of $M(\gamma)$ are the characteristic values of matrix A_3D_1, which is similar to the matrix $(D_1^{\frac{1}{2}}A_3D_1^{\frac{1}{2}}) \cdot (D_1^{-\frac{1}{2}}D_1D_1^{-\frac{1}{2}}) = D_1^{\frac{1}{2}}A_3D_1^{\frac{1}{2}}$. The last matrix is symmetric, and therefore has real characteristic values. ∎

Theorem 3.25. Suppose that all characteristic values of a self-adjoint with respect to Φ endomorphism $\gamma : V \longrightarrow V$ are real numbers. Then, all distinct characteristic subspaces of $\gamma : V(c_1;\gamma), \ldots, V(c_r;\gamma)$, where $c_i \neq c_j$ $(1 \leq i,j \leq r)$ are pairwise orthogonal with respect to Φ.

Proof. Let $x_i \in V(c_i;\gamma)$, $x_j \in V(c_j;\gamma)$, then

$$(c_1 - c_j)\Phi(x_i,x_j) = c_i\Phi(x_i,x_j) - c_j\Phi(x_i,x_j) = \Phi(c_ix_i,x_j) -$$

$$- \Phi(x_i,c_jx_j) = \Phi(\gamma(x_i),x_j) - \Phi(x_i,\gamma(x_j)) = \Phi(x_i,\gamma^*(x_j)) -$$

$$- \Phi(x_i,\gamma(x_j)) = 0,$$

and since $c_i - c_j \neq 0$ it follows that $\Phi(x_i,x_j) = 0$. ∎

Theorem 3.26. Let γ be an endomorphism of (V,Φ). Then a subspace W of V is stable under γ (i.e., $\gamma(V) \subseteq V$) if and only if $V^{\perp}$ is stable under γ^*.

Proof. $\Longrightarrow$. Take any $y \in W^{\perp}$, then

$$\Phi(x,\gamma^*(y)) = \Phi(\gamma(x),y) = 0 \qquad \forall x \in W,$$

and therefore $\gamma^*(y) \in W^{\perp}$.

$\Longrightarrow$. Take any $x \in W$, then

$$\Phi(\gamma(x),y) = \Phi(x,\gamma^*(y)) = 0 \qquad \forall y \in W^{\perp},$$

and therefore $\gamma(x) \in W$. ∎

Finally, we establish an important fact which together with Theorem 3.24 gives a restricted analogue of the corresponding result in the real Euclidean spaces, that every self-adjoint endomorphism could be "diagonalized (over $\mathbb{R}$) in an orthonormal basis".

Theorem 3.27. Suppose that γ is a positive with respect to Φ endomorphism of V. Then there exists a Φ-orthonormal basis of V, consisting of the characteristic vectors of γ.

Proof. The proof is by induction on $n = \dim(V)$. Since the theorem is trivially true for $n = 0$, assume now that it is true for V with $\dim(V) = n-1$. Let us take any vector space V, $\dim(V) = n$, with non-degenerate symmetric bilinear form Φ on it, and let γ be a Φ-positive endomorphism of V. Then by Theorem 3.24 it has at least one characteristic vector $a_1 \in V(c_1;\gamma)$. Vector a_1 is non-isotropic (see the remark after Def. 3.17), therefore $\langle a_1 \rangle \cap \langle a_1 \rangle^{\perp} = \{0\}$, and by Theorem 3.4 $V = \langle a_1 \rangle \oplus \langle a_1 \rangle^{\perp}$ and $\Phi | \langle a_1 \rangle^{\perp} \times \langle a_1 \rangle^{\perp}$ is non-degenerate. Subspace $\langle a_1 \rangle$ is stable under γ and $\gamma^* = \gamma$, thus by Theorem 3.26 so is $\langle a_1 \rangle^{\perp}$. Obviously, $\gamma | \langle a_1 \rangle^{\perp}$ is Φ-positive, and $\dim(\langle a_1 \rangle^{\perp}) = n-1$. Applying the induction hypothesis to $\langle a_1 \rangle^{\perp}$, we obtain an orthonormal basis $(e_i)_{2 \le i \le n}$ of $\langle a_1 \rangle^{\perp}$, consisting of characteristic vectors of $\gamma | \langle a_1 \rangle^{\perp}$, and therefore characteristic vectors of γ. If we set

$$e_1 = \sqrt{|\Phi(a_1, a_1)|}^{-1} a_1,$$

the basis $(e_i)_{1 \le i \le n}$ is the sought basis, because $e_1 \perp \langle a_1 \rangle^{\perp}$. ∎

4 VECTOR REPRESENTATION OF FINITE PSEUDOMETRIC SPACES

In this chapter we are going to put the two halves - pseudometric and pseudoeuclidean spaces - together. The chapter is based on almost half a century old result due to I.J. Schoenberg ([15], Theorem 1'), and the algorithm following from it. Surprisingly enough the result has remained largely unknown to research workers in the fields of general data analysis, in spite of the considerable efforts exerted, for instance, in multidimensional scaling. Since the corresponding proof in [15] is only sketched, and the author does not know of any other reference containing a proof of the result, a complete and somewhat modified proof is included (which is necessary anyway because our terminology is quite different). It is interesting to note that the quadratic form which plays the fundamental role in the theorem (Theorem 4.1) came from number theory, and is associate with such names as Gauss, Dirichlet, and Minkowski (see [40], p. 50).

The algorithm following from the proof of the theorem is remarkably simple, particularly considering the theoretical subtlety of the problem. Some practical improvements of the algorithm will also be discussed in chapter 6.

The last several pages of this chapter are crucial to understanding the essence of the proposed approach.

As above, $\mathbb{R}^{(n^+,n^-)}$ denotes the real n-dimensional ($n = n^+ + n^-$) pseudoeuclidean vector space, i.e., a vector space V together with a non-degenerate symmetric bilinear form Φ of signature (n^+,n^-) measuring innder products in V (see Def. 3.3, 3.7). The squared distance between vectors $x = \sum_{i=1}^{n} x^i e_i$ and $y = \sum_{i=1}^{n} y^i e_i$, where $(e_i)_{1\le i\le n}$ is a Φ-orthonormal basis, can be computed by the formula (see Def. 3.2)

$$\|x-y\|^2 = (x^1-y^1)^2 + \ldots + (x^{n^+}-y^{n^+})^2 - (x^{n^++1}-y^{n^++1})^2 - \ldots - (x^n-y^n)^2.$$

If L is a finite set of vectors from $\mathbb{R}^{(n^+,n^-)}$ satisfying the property $\|x-y\|^2 \ge 0 \quad \forall x,y\in L$, then L may be considered as a subspace

of some pseudometric space (S,Σ), where

$$S = \{v \in \mathbb{R}^{(n^+,n^-)} \mid L \subset S, \ \|v_1-v_2\|^2 \geq 0 \quad \forall v_1,v_2 \in S\}$$

and

$$\Sigma(v_1,v_2) = \|v_1-v_2\| .$$

It is precisely this situation we have in mind when we speak below of an isometric embedding (see Def. 2.4) of a finite pseudometric space into $\mathbb{R}^{(n^+,n^-)}$.

The next definition introduces for a finite pseudometric space an analogue of a notion of the dimension of a set of vectors.

Definition 4.1. A pair of non-negative numbers (n^+,n^-) will be called the vector signature of a finite pseudometric space (P,Π) if there exists an isometric embedding

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)},$$

where $\Pi(p_1,p_2) = \|\alpha(p_1)-\alpha(p_2)\| \qquad \forall p_1,p_2 \in P,$

such that for any other similar isometric embedding of (P,Π) into $\mathbb{R}^{(n_1,n_2)}$ we have $n_1 \geq n^+$, $n_2 \geq n^-$. The above mapping α will be called in this case a vector representation of (P,Π).

In other words, (n^+,n^-) is the vector signature of a finite pseudometric space (P,Π) if $\mathbb{R}^{(n^+,n^-)}$ is a "minimal" pseudoeuclidean vector space, where (P,Π) can be isometrically represented (see Fig. 4.1).

When (P,Π) lies in a euclidean space $\mathbb{R}^m$, a mapping $\alpha : P \longrightarrow \mathbb{R}^n$, $\alpha(p_i) = p_i$, where $\mathbb{R}^n$ is a subspace of $\mathbb{R}^m$ spanned by P, would obviously be a vector representation.

It is not evident at all if the vector signature exists for every finite pseudometric space, but this fact follows from the next theorem. To better understand the statement of the theorem it may be helpful to see where the quadratic form appearing in it comes from. Let $\{v_i\}_{0\leq i\leq k}$ be a set of vectors in $\mathbb{R}^{(n^+,n^-)}$ such that $v_0 = 0$ and $\|v_i-v_j\| = d_{ij}^2 \quad \forall i,j$. Then the Gram matrix of $\{v_i\}_{1\leq i\leq k}$ (Def. 3.11) is $(\Phi(v_i,v_j))_{1\leq i,j\leq k}$, and

$$\Phi(v_i,v_j) = \frac{1}{2}\,[\Phi(v_i,v_i) + \Phi(v_j,v_j) - \Phi(v_i-v_j,v_i-v_j)] =$$

$$= \frac{1}{2}\,[\|v_i-0\|^2 + \|v_j-0\|^2 - \|v_i-v_j\|^2] = \frac{1}{2}\,(d_{0i}^2 + d_{0j}^2 - d_{ij}^2).$$

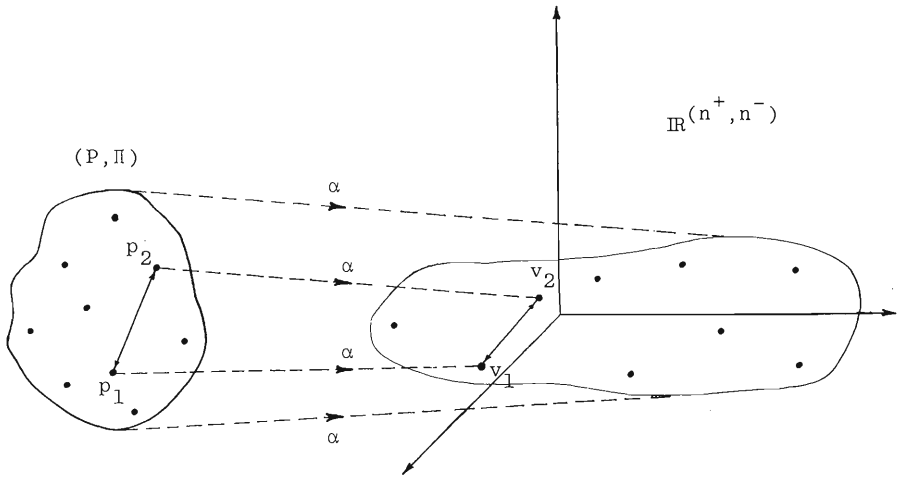


Figure 4.1

Now it should not come as a surprise that the following theorem holds. Set $a_i = (0,\ldots,0,\underset{(i)}{1},0,\ldots,0) \in \mathbb{R}^k$ †) $(1 \le i \le k)$.

Theorem 4.1. A finite pseudometric space (P,Π), where $P = \{p_i\}_{0 \le i \le k}$, has the vector signature (n^+,n^-) if and only if the symmetric bilinear form Ψ on $\mathbb{R}^k$ †) whose matrix w.r.t. the standard basis $(a_i)_{1 \le i \le k}$ is

$$M(\Psi) = (m_{ij})_{1 \le i,j \le k} = \left(\frac{1}{2}\left(d_{0i}^2 + d_{0j}^2 - d_{ij}^2\right)\right)_{1 \le i,j \le k},$$

where $d_{ij} = \Pi(p_i,p_j)$, $0 \le i,j \le k$, has the signature (n^+,n^-) (see Def. 3.7). Bilinear form Ψ will be called the bilinear form of the pseudometric space (P,Π).

Proof. $\boxed{\Longrightarrow}$. Suppose that (P,Π) has the vector signature (n^+,n^-), and suppose that $\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}$ is its vector representation. Consider another vector representation of (P,Π)

$$\beta : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}$$

---

†) $\mathbb{R}^k$ denotes here not the Euclidean space, but just the real k-dimensional vector space of k-tuples $(x^1,\ldots,x^k)$, where $x^i \in \mathbb{R}$, without any inner product.

defined as $\beta(p_i) = \alpha(p_i) - \alpha(p_0)$ $(0 \le i \le k)$, i.e., obtained by the parallel translation of the vectors $\alpha(p_i)$. Then $\beta(p_0) = 0$, and the statement that β is indeed a distance preserving mapping is obvious, since a parallel translation preserves all the distances in the pseudoeuclidean space. Let $\beta(p_i) = z_i = \sum_{j=1}^{n} z_i^j e_j$ $(0 \le i \le k)$, where $n = n^+ + n^-$ and (e_j) is a Φ-orthonormal basis of $\mathbb{R}^{(n^+,n^-)}$, and let

$$Z = \begin{pmatrix} z_1^1 & z_2^1 & \cdots & z_k^1 \\ z_1^2 & z_2^2 & & z_k^2 \\ \vdots & \vdots & \vdots & \vdots \\ z_1^n & z_2^n & & z_k^n \end{pmatrix}_{n \times k}.$$

First of all, the rank of Z is equal to n: suppose the opposite, then by the known theorem vectors $z_1, \ldots, z_k$ generate the subspace of $\mathbb{R}^n$ of dimension less than n, and therefore (n^+, n^-) is not the vector signature of (P, Π), contrary to the assumption.

Since the rank of Z is equal to n, it follows that $n \le k$, and that the n vectors formed by the rows of Z are linearly independent in $\mathbb{R}^k$. Thus, if $n < k$ we can add to these n vectors such k-n vectors of $\mathbb{R}^k$ that together they form a basis of $\mathbb{R}^k$. Let L be the matrix obtained from Z by adding to Z from below k-n rows corresponding to the above k-n vectors. Obviously, L is an invertible matrix. To prove the part $\Longrightarrow$ of the theorem, it is enough to show that there exists a basis of $\mathbb{R}^k$ such that the matrix of the above form Ψ with respect to it is the matrix

$$J_0 = \begin{pmatrix} I_{n^+} & 0 & 0 \\ 0 & -I_{n^-} & 0 \\ 0 & 0 & 0 \end{pmatrix}_{k \times k}$$

By (3.4) this is equivalent to the existence of an invertible matrix T such that

$$J_0 = {}^tT \cdot M(\Psi) \cdot T.$$

In other words, it is enough to find matrix T such that

$${}^t(T^{-1}) \cdot J_0 \cdot T^{-1} = M(\Psi).$$

Set $T^{-1} = L$. Then the last equality follows from the definitions of L, z_i $(1 \le i \le k)$ and the remark preceding the statement of the theorem: denoting the product matrix ${}^tL \cdot J_0 \cdot L$ by $G = (g_{ij})_{1 \le i,j \le k}$ one can check immediately that $g_{ij} = \Phi(z_i, z_j) = \frac{1}{2}(d_{0i}^2 + d_{0j}^2 - d_{ij}^2)$.

$\boxed{\Longleftarrow}$. Suppose that the symmetric bilinear form Ψ of the pseudometric space (P, Π) has signature (n^+, n^-), $n^+ + n^- = n$. By Theorem 3.12 there exists a basis $(e_i)_{1 \le i \le k}$ of $\mathbb{R}^k$ such that the above mentioned matrix J_0 is the matrix of Ψ w.r.t. this basis. Then

$$\mathbb{R}^k = V \oplus V^{\perp},$$

where $V = \langle e_1, \ldots, e_n \rangle$ and $V^{\perp} = \langle e_{n+1}, \ldots, e_k \rangle$. Obviously, $\Phi = \Psi | V \times V$ is non-degenerate. Consider the projection

$$\pi_V : \mathbb{R}^k \longrightarrow V$$

defined as: $x = \sum_{i=1}^{k} x^i e_i \Longrightarrow \pi_V(x) = \sum_{i=1}^{n} x^i e_i$. Let $(a_i)_{1 \le i \le k}$ be, as above, the standard basis of $\mathbb{R}^k$, and let $a_0 = 0$. We shall show that the mapping

$$\alpha : (P, \Pi) \longrightarrow (V, \Phi),$$

where $\alpha(p_i) = \pi_V(a_i)$ $(0 \le i \le k)$, is a vector representation of (P, Π), and thus the part $\Longleftarrow$ of the theorem will be proved.

Set $v_i = \pi_V(a_i)$. Since for every i, j $(0 \le i, j \le k)$

$$(4.1) \quad \| v_i - v_j \|^2 = \Phi(v_i - v_j, v_i - v_j) = \Phi(v_i, v_i) + \Phi(v_j, v_j) - 2\Phi(v_i, v_j),$$

we should find $\Phi(v_i, v_j)$. So, denoting $x - \pi_V(x)$ by $\bar{x}$ $(\bar{x} \in V^{\perp})$ we have

$$\Phi(v_i, v_j) = \Phi(\pi_V(a_i), \pi_V(a_j)) = \Psi(a_i - \bar{a}_i, a_j - \bar{a}_j) =$$

$$= \Psi(a_i, a_j) + \Psi(\bar{a}_i, \bar{a}_j) - \Psi(a_i, \bar{a}_j) - \Psi(\bar{a}_i, a_j) = \Psi(a_i, a_j) =$$

$$= \frac{1}{2}(d_{0i}^2 + d_{0j}^2 - d_{ij}^2),$$

since $\quad \forall \bar{v} \in V^{\perp}, \quad \forall v \in V \quad \Psi(\bar{v}, v) = 0.$

Substituting into (4.1) we obtain

$$\|v_i - v_j\|^2 = \frac{1}{2}\,(d_{0i}^2 + d_{0i}^2 - d_{ii}^2) + \frac{1}{2}\,(d_{0j}^2 + d_{0j}^2 - d_{jj}^2) -$$

$$- 2 \cdot \frac{1}{2}\,(d_{0i}^2 + d_{0j}^2 - d_{ij}^2) = d_{ij}^2.$$

Finally, we have to prove that there is no isometry of (P,Π) onto a subset of $\mathbb{R}^{(n_1,n_2)}$, where $n_1 < n^+$ or $n_2 < n^-$. Indeed, if one supposes the opposite, then by the part $\Longrightarrow$ of the theorem it follows that the signature of the form Ψ is not (n^+,n^-), which contradicts the assumption. ∎

Corollary. A finite pseudometric space can be isometrically represented in the Euclidean n-dimensional space if and only if its bilinear form is positive and of rank less than n. ∎

First, we note that the uniqueness of the vector signature follows from the above theorem and Theorem 3.12.

Second, one should also note another *corollary of the above theorem*: all bilinear forms of a finite pseudometric space have the same signature. In other words, their signatures do not depend on the ordering (labelling) of the points.

The most important feature of the proof of Theorem 4.1 is the fact that it is constructive: in part $\Longleftarrow$ we explicitly constructed a vector representation of (P,Π). Indeed,

$$\alpha(p_i) = \pi_V(a_i) \qquad 0 \le i \le k$$

(i)

where $a_0 = 0$, and $a_i = (0,\ldots,0,1,0,\ldots,0)$. Obviously, it is preferable to obtain the coordinates of $\alpha(p_i)$ w.r.t. some Φ-orthonormal basis of $\mathbb{R}^{(n^+,n^-)}$. Let $(e_i)_{1\le i\le n}$ be such a basis, as in the above proof. Then using any algorithm†) for the reduction of a matrix of a symmetric bilinear form (or corresponding quadratic form, see Appendix I) to the canonical form J_0 (see Theorem 3.12), we can find the matrix $T = M(\tau)$ of the transition automorphism $\tau : \mathbb{R}^k \longrightarrow \mathbb{R}^k$ from the basis (e_i) to the basis (a_i). So that

$$^tT \cdot J_0 \cdot T = M(\Psi),$$

where T is the matrix of τ w.r.t. $(e_i)_{1\le i\le k}$, and $M(\Psi)$ is the matrix of Ψ w.r.t. $(a_i)_{1\le i\le k}$. Having found T, we can immediately find the

†)There are several of them.

coordinates of a_i w.r.t. the Φ-orthonormal basis (e_i):

$$a_i = \tau(e_i) = \sum_{j=1}^{k} a_i^j e_j,$$

where ${}^t(a_i^1,\ldots,a_i^k)$ is the i's column of T. Finally, the first $n = n^+ + n^-$ coordinates of a_i give the coordinates of the vector $\pi_V(a_i) = \alpha(p_i) \in \mathbb{R}^{(n^+,n^-)}$ w.r.t. the orthonormal basis $(e_i)_{1\le i\le n}$. Note that $\alpha(p_0) = 0$.

Thus, quite fortunately the problem of finding a vector representation of a finite pseudometric space (P,Π), $P = \{p_i\}_{0\le i\le k}$, reduces in practice to the quite simple classical algebraic problem of finding the automorphism $\tau : \mathbb{R}^k \longrightarrow \mathbb{R}^k$ whose inverse τ^{-1} transforms the matrix of a symmetric bilinear form Ψ of (P,Π) to the canonical form.

THE FIRST EMBEDDING ALGORITHM. One of the most reliable computational methods to obtain a vector representation of a finite pseudometric space (P,Π) is to find matrix $\tilde{T}$

$$\tilde{T} = L \cdot \bar{D}^{\frac{1}{2}} \tag{4.2}$$

where $\bar{D} = \mathrm{diag}(|d_1|,\ldots,|d_k|)$ is a diagonal matrix of the modules of the characteristic values of $M(\Psi)$ (positive first, then negative, then 0's), and L is the orthogonal matrix (${}^tL\cdot L = I$) whose columns are the corresponding orthonormal characteristic vectors. The vector representation space then is $\mathbb{R}^{(n^+,n^-)}$, where n^+ is the number of positive characteristic values and n^- is the number of negative ones. The coordinates of the vector representing p_i are the first $n = n^+ + n^-$ elements of the i^{th} row of $\tilde{T}$, and p_0 is represented by the origin.

To see this it is sufficient to remember, that by definition of the characteristic vectors of $M(\Psi)$

$$M(\Psi)\cdot L = L\cdot D, \quad \text{or} \quad M(\Psi) = L\cdot D\cdot {}^tL,$$

since ${}^tL = L^{-1}$. Whence, taking $J_0 = \begin{pmatrix} I_{n^+} & 0 & 0 \\ 0 & -I_{n^-} & 0 \\ 0 & 0 & 0 \end{pmatrix}_{k\times k}$ we obtain

$$M(\Psi) = L\cdot(\bar{D}^{\frac{1}{2}}\cdot J_0\cdot \bar{D}^{\frac{1}{2}})\cdot {}^tL = (L\cdot\bar{D}^{\frac{1}{2}})\cdot J_0\cdot {}^t(L\cdot\bar{D}^{\frac{1}{2}}).$$

So, the first $n^+ + n^-$ elements of the i^{th} row of the above $\tilde{T}$ give the coordinates of the $\alpha(p_i)$ w.r.t. some Φ-orthonormal basis $(e_j)_{1\leq j\leq n}$: the transition matrix from $(e_i)_{1\leq i\leq k}$ to the standard basic $(a_i)_{1\leq i\leq k}$ is ${}^t(L\cdot\overline{D}_*^{\frac{1}{2}})$, where $\overline{D}_*$ is obtained from $\overline{D}$ by replacing 0's on the main diagonal with any non-zero numbers. The algorithm based on (4.2) will be called the first embedding algorithm (see also chapter 6).

One of the most efficient methods for solving the complete eigen-problem is the so-called QR-algorithm (see, for example [41]). In numerical analysis it is known to be a stable algorithm, and the eigen-problem itself is always a well-conditioned problem for any symmetric matrix. Thus, from the numerical point of view the QR-algorithm is as nice a one as one could possibly hope to have for a solution of an algebraic problem. It is one of the most reliable numerical algorithms available for commercial and scientific uses.

It is important to note that since the characteristic values of $M(\Psi)$ can be obtained in the order of decreased magnitude, *one has complete control over the dimensionality of the representation space.* In other words, by omitting the "axes" corresponding to very small characteristic values of $M(\Psi)$, we do not change essentially the configuration of the initial finite pseudometric space (P,Π) (see chapter 6). The author believes that the described algorithm is, in general, the simplest algorithm one could produce to represent *faithfully* dissimilarity data in a vector space.

Since the "philosophical" issues involved were discussed in chapter 1, I shall mention now only the two most important points. First of all, one should note that the vector space of the representation is defined by the "geometry" of the sample, so that the "variables" involved may or may not have simple physical interpretation. In the traditional approach one finds the "variables" first and only then represents the sample in this *fixed* vector (linear) frame with a *fixed* Euclidean distance, without actually knowing whether it is legitimate to join these "variables" together in a Euclidean (Cartesian) framework. Thus, since the sample is squeezed into a very restrictive frame the fundamental "geometric" (metric) characteristics the sample may be considerably distorted.

The second point has to do with the fact that the structure of $\mathbb{R}^{(n^+,n^-)}$ is richer than that of $\mathbb{R}^n$, simply because of the existence of n+1 different symmetric bilinear forms on $\mathbb{R}^n$. In other words, the geometry of $\mathbb{R}^{(n^+,n^-)}$ is not uniquely determined by the

topology of $\mathbb{R}^n$, as in the Euclidean space, hence offering a more *accurate* tool for general data analysis.

Next, a construction of the vector representation will be demonstrated on several simple but illuminating examples. To reveal some general ideas the examples are presented in a direct numerical form, although one should keep in mind that behind the numbers may be one or another concrete distance measure. To facilitate geometric comprehension of the embedding process the reader is advised to test the method manually on several small size problems. The simplest numerical algorithm for *manual computation* of the vector representation is the one obtained by adapting the Lagrange's algorithm for the reduction of a quadratic form[†)] to the sum of the squares (see, for example, [39], p. 194). Another "geometric" advantage of the latter algorithm is that it always places the first two points - p_0 and p_1 - on the axis x^1. It is this consideration that suggested the choice of the embedding algorithm for the examples in this chapter. For a more definitive choice of the embedding algorithm see chapter 6.

Example 4.1. Let $P = \{p_i\}_{0 \le i \le 3}$, and let Π be given by the following matrix

$$D = (d_{ij}) = \begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix},$$

where $d_{ij} = \Pi(p_i, p_j)$ is the distance between p_i and p_j in (P,Π). (P,Π) is represented diagrammatically in Figure 4.2.

As one can clearly see, (P,Π) is constructed by taking three points p_0, p_1, p_2 on a straight line, and adding a point p_3 which is of equal distance to each of them. It is easy to check that (P,Π) is a metric space, and it is also easy to guess that it is not representable in a Euclidean space. Indeed, on the one hand, p_3 must lie in the intersection of the two spheres of radii one with centers at p_0 and p_2, and p_1 is the only point at which the spheres meet.

[†)] On the relation between quadratic and bilinear forms see Appendix I.

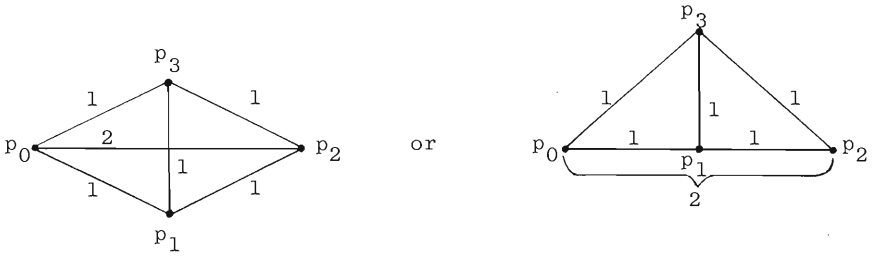


Figure 4.2

On the other hand, p_3 must be at distance 1 from p_1, so that no Euclidean space can accommodate this situation. Let us see what answer is provided by the embedding method.

The quadratic form of (P,Π) is (see Theorem 4.1 and Appendix I)

$$\Psi(x) = d_{01}^2(x^1)^2 + (d_{01}^2 + d_{02}^2 - d_{12}^2)x^1x^2 + (d_{01}^2 + d_{03}^2 - d_{13}^2)x^1x^3 +$$

$$+ d_{02}^2(x^2)^2 + (d_{02}^2 + d_{03}^2 - d_{23}^2)x^2x^3 + d_{03}^2(x^3)^2.$$

Substituting the data we get

$$\Psi(x) = (x^1)^2 + 4x^1x^2 + x^1x^3 + 4(x^2)^2 + 4x^2x^3 + (x^3)^2,$$

where $x = (x^1, x^2, x^3)$.

Using Lagrange's algorithm([39], p. 194), we can reduce Ψ to the following canonical form

$$\Psi(y) = (y^1)^2 + (y^2)^2 - (y^3)^2,$$

where $y = T \cdot x$ and

$$T \doteq \begin{pmatrix} 1 & 2 & 0.5 \\ 0 & 0.6030226 & 1.0552897 \\ 0 & 0.6030226 & -0.6030226 \end{pmatrix}.$$

So, the signature of the quadratic form is (2,1), and by Theorem 4.1 pseudoeuclidean vector space $\mathbb{R}^{(2,1)}$ is the "minimal" one, where (P,Π) can be isometrically represented. In fact, $\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(2,1)}$ given by

$$\alpha(p_0) = v_0 = (0,\ 0,\ 0)$$
$$\alpha(p_1) = v_1 = (1,\ 0,\ 0)$$
$$\alpha(p_2) = v_2 = (2,\ 0.6030226,\ 0.6030226)$$
$$\alpha(p_3) = v_3 = (0.5,\ 1.0552897,\ -0.6030226)$$

is a vector representation of (P,Π).

Remembering that the above are the coordinates of v_i with respect to some Φ-orthonormal basis (e_i) of $\mathbb{R}^{(2,1)}$, we can easily check the results of computations:

$$\|v_0-v_1\|^2 = 1 = d_{01}^2$$
$$\|v_0-v_2\|^2 = 2^2 + 0.6030226^2 - 0.6030226^2 = 4 = d_{02}^2$$
$$\|v_0-v_3\|^2 = 0.5^2 + 1.0552897^2 - (-0.6030226)^2 \doteq 1 = d_{03}^2$$
$$\|v_1-v_2\|^2 = (2-1)^2 + (0.6030226-0)^2 - (0.6030226-0)^2 = 1 = d_{12}^2$$
$$\|v_1-v_3\|^2 = (0.5-1)^2 + (1.0552897-0)^2 - (-0.6030226-0)^2 \doteq 1 = d_{13}^2$$
$$\|v_2-v_3\|^2 = (0.5-2)^2 + (1.0552897-0.6030226)^2 - (0.6030226 - 0.6030226)^2 \doteq 1 = d_{23}^2.$$

Concluding the example we should note that it also demonstrates another important fact: by adding one point (point p_3) to a finite pseudometric space (space $\{p_0,p_1,p_3\}$) the dimension of the vector space of the vector representation could be increased by more than one, contrary to the classical situation. Thus, noise in the data, as well as the outliers ("wild shots") may alter the vector representation. Quite fortunately, these problems can be relatively easily dealt with (see chapter 6).

Example 4.2. Let $P' = \{p'\}_{0\le i\le 7}$, and let Π' be given by the following matrix

$$D' = (d'_{ij}) = \begin{pmatrix} 0 & 1 & 2 & 1 & 4 & 5 & 6 & 3 \\ 1 & 0 & 1 & 1 & 3 & 4 & 5 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 & 4 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 4 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ 5 & 4 & 3 & 1 & 1 & 0 & 1 & 2 \\ 6 & 5 & 4 & 1 & 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 1 & 1 & 2 & 3 & 0 \end{pmatrix},$$

where $d'_{ij} = \Pi'(p'_i, p'_j)$ is the distance between p'_i and p'_j in (P', Π'). Diagrammatically we can represent this pseudometric space as follows

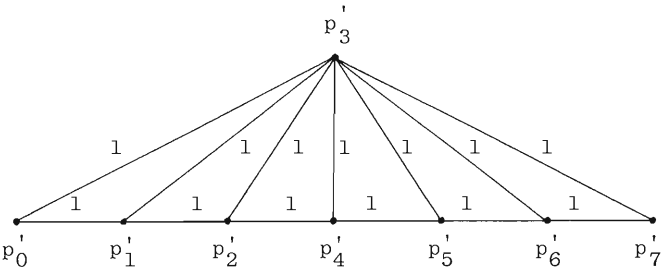


Figure 4.3

It is easy to see that the space (P, Π) of Example 4.1 is a subspace of (P', Π') (see Def. 2.4); (P', Π') is constructed from (P, Π) by adding four points p_4, p_5, p_6, p_7 which lie on the same line as the first three and all four at a distance 1 from p_3.

If the above mentioned program (see Example 1) is run with D' as an input, we get the following (rounded-off) vector representation:

$$\alpha' : (P', \Pi') \longrightarrow \mathbb{R}^{(2,1)},$$

where

$$\alpha'(p'_0) = v'_0 = (0,\ 0,\ 0)$$
$$\alpha'(p'_1) = v'_1 = (1,\ 0,\ 0)$$
$$\alpha'(p'_2) = v'_2 = (2,\ 0.603,\ 0.603)$$
$$\alpha'(p'_3) = v'_3 = (0.5,\ 1.055,\ -0.603)$$
$$\alpha'(p'_4) = v'_4 = (3,\ 1.809,\ 1.809)$$
$$\alpha'(p'_5) = v'_5 = (4,\ 3.618,\ 3.618)$$
$$\alpha'(p'_6) = v'_6 = (5,\ 6.03,\ 6.03)$$
$$\alpha'(p'_7) = v'_7 = (6,\ 9.045,\ 9.045).$$

All the points are plotted in Figure 4.4.

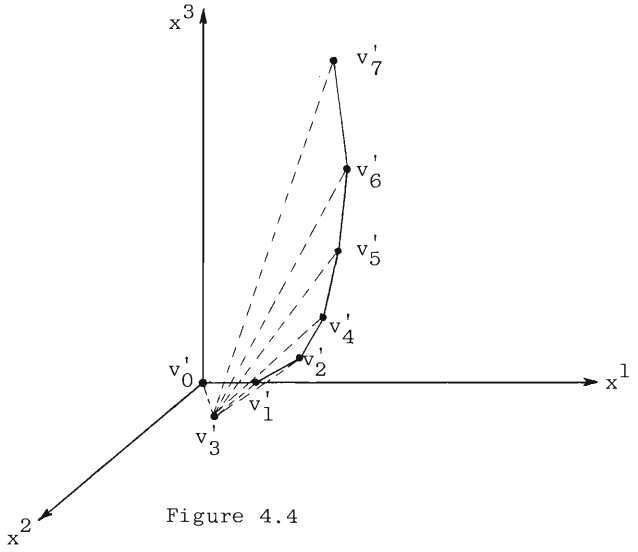


Figure 4.4

It is not difficult to see that points p_0, p_1, p_2, p_4, p_5, p_6, p_7 lie on the curve which could be called a metric (straight) line: for every three consecutive points x_1, x_2, x_3 which belong to it the distance between x_1 and x_2 plus the distance between x_2 and x_3 is equal to the distance between x_1 and x_3. In a Euclidean space there is only one kind of curve which satisfies the above property - straight lines, but the above example shows that this is not the case in a pseudoeuclidean space. Again, this illustrates why a pseudoeuclidean vector space can accommodate more general metric configurations than an Euclidean space.

As one may have already noticed, the coordinates of the first four points in Example 4.2 are the same as in the first example. This fact is a simplified illustration of *an important idea of the proposed approach*: if the sample one possesses is (more or less) sufficient to determine the vector space where the entire set under the analysis can be represented, then the vector representation of an enlarged sample does not (essentially) change the initial vector representation. Simply speaking, if the sample is big enough to determine the major geometric features of the group it represents, then the analysis could be justifiably performed in the vector space where the sample is represented.

A formal statement of the fact that we observed in the two examples is given below. The following important lemma will be used in the proof of the corresponding statement (as well as in another proof).

Lemma 4.2. For every basis $(a_i)_{1\le i\le n}$ of $\mathbb{R}^{(n^+,n^-)}$ there exists at most one vector $x\in\mathbb{R}^{(n^+,n^-)}$ such that for a given set of real numbers $\{d_0,d_1,\ldots,d_n\}$

$$\Phi(x-a_i,x-a_i) = d_i \qquad 1\le i\le n$$
$$\Phi(x,x) = d_0.$$

Proof. Let us assume that such vector x exists. Then, by (3.1)

$$\Phi(x-a_i,x-a_i) = \Phi(x,x) - 2\Phi(x,a_i) + \Phi(a_i,a_i),$$

so that subtracting from each of the first n given equations (in which the left-hand sides are replaced according to the above formula) the last equation, we obtain the following n equations:

$$-2\Phi(x,a_i) + \Phi(a_i,a_i) = d_i - d_0 \qquad 1\le i\le n\ ,$$

or

$$\Phi(x,a_i) = \frac{1}{2}\,[\Phi(a_i,a_i) + d_0 - d_i] \qquad 1\le i\le n\ .$$

Suppose that $x = \sum_{j=1}^{n} x^j a_j$, then again by (3.1) the above equations can be rewritten as

$$\sum_{j=1}^{n} x^j \Phi(a_j,a_i) = \frac{1}{2}\,[\Phi(a_i,a_i) + d_0 - d_i] \qquad 1\le i\le n$$

Since the last set of equations can be viewed as a system of n linear equations in n unknowns x^j with a non-zero determinant (see Theorem 3.15), the lemma follows now from a well known theorem. ∎

Theorem 4.3. Let a pseudometric space (P,Π) be a subspace of a finite pseudometric space (P',Π'), and let both of the pseudometric spaces have the same vector signature (n^+,n^-). Then, for every vector representation $\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}$ there exists a unique vector representation $\alpha' : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)}$ which is an extension of α:

$$\alpha'|P = \alpha.$$

Proof. Let $P = \{p_i\}_{0\le i\le m}$, $P' = \{p_i\}_{0\le i\le m+r}$, then we can assume, without loss of generality, that the given vector representation α of (P,Π) satisfies the following condition

$$\alpha(p_0) = 0$$

(see the beginning of the proof of Theorem 4.1). We can choose from the vectors $\alpha(p_i)$ $(1\le i\le m)$ a basis $(a_j)_{1\le j\le n^+ +n^-}$, $a_j = \alpha(p_{i_j})$, of $\mathbb{R}^{(n^+,n^-)}$. The possibility of such a choice follows from the fact that $\alpha(p_i)$ $(1\le i\le m)$ generate $\mathbb{R}^{(n^+,n^-)}$. Now, the existence of the vectors $x_\ell \in \mathbb{R}^{(n^+,n^-)}$ $(m+1\le \ell\le m+r)$ satisfying the conditions

$$\Phi(x_\ell - a_j, x_\ell - a_j) = [\Pi'(p_\ell, p_{i_j})]^2 \qquad (1\le j\le n^+ +n^-)$$

$$\Phi(x_\ell, x_\ell) = [\Pi'(p_\ell, p_0)]^2$$

follows from the given fact that the pseudometric space (P',Π') has the vector signature (n^+,n^-), i.e., from the fact that (P',Π') can be represented in $\mathbb{R}^{(n^+,n^-)}$. The uniqueness of the vectors x_ℓ $(m+1\le\ell\le m+r)$ follows from Lemma 4.2, and to find each of them we have to solve the system of $n^+ +n^-$ linear equations in $n^+ +n^-$ unknowns x_ℓ^j $(1\le j\le n^+ +n^-)$ with the determinant $g(a_1,\ldots,a_n)$ (see Def. 3.11). Setting

$$\alpha'(p_\ell) = x_\ell \qquad (m+1\le\ell\le m+r)$$

$$\alpha'(p_\ell) = \alpha(p_\ell) \qquad (0\le\ell\le m)$$

we obtain the sought vector representation of (P',Π'), the uniqueness of which follows from the uniqueness of the vectors x_ℓ $(m+1\le\ell\le m+r)$. ∎

Since the algorithm we used in the above examples constructs coordinates of the vectors $\alpha'(p_i)$ consecutively, vectors $\alpha'(p_i)$, $4\le i\le 7$, are determined uniquely, which explains the fact noted at the beginning of the paragraph preceding Lemma 4.2. The fact that Lagrange's algorithm constructs coordinates of the vectors consecutively is not essential at all by two reasons: 1) the first one will become clear after the theorem below, 2) once a vector representation of the "training" sample is constructed it is more economical to represent a new element using the idea of the orthogonal projection onto a subspace (see formula (3.8)); so that if a new element does not change the vector signature, its coordinates are determined

uniquely by Lemma 4.2. We will return to the last point in chapter 7.

The next example illustrates the vector representation for the case of a pseudometric space which is not semimetric (see Def. 2.2). In this case, as one should expect, the point p_i of the space, whose distance from a point p_j is zero, will be mapped by a vector representation α into the "isotropic cone with the centre at p_j"[†]. So that if $\Pi(p_i,p_j) = 0$ for several p_j, then $\alpha(p_i)$ lies in the intersection of the corresponding cones.

Example 4.3. Suppose that the distance matrix of a pseudometric space (P,Π) is

$$\begin{pmatrix} 0 & 0 & 2 & 2 & 3 \\ 0 & 0 & 1 & 1 & 2 \\ 2 & 1 & 0 & 0 & 2 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & 2 & 2 & 1 & 0 \end{pmatrix}$$

Then, using the "Lagrange's embedding algorithm" we obtain the following (round-off) vector representation

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(2,2)}$$

$$\alpha(p_0) = (0,\ 0,\ 0,\ 0)$$

$$\alpha(p_1) = (0.5669,\ 0,\ 0.5669,\ 0)$$

$$\alpha(p_2) = (2.079,\ 0,\ -0.5669,\ 0)$$

$$\alpha(p_3) = (2.079,\ 0.5267,\ -0.5669,\ 0.5267)$$

$$\alpha(p_4) = (2.646,\ 2.321,\ -1.764,\ -0.5267).$$

It is important to note that all the resulting sets of vectors in $\mathbb{R}^{(n^+,n^-)}$ for different pseudometric spaces and different n^+,n^- have one geometric property in common: for every j the "isotropic cone with the center at $v_j = \alpha(p_j)$"[†] does not have "inside" any other points $v_i = \alpha(p_i)$, and thus the resulting set of vectors is elongated along the subspace $V^+ = \mathbb{R}^{(n^+,0)}$ more than along the subspace $V^- = \mathbb{R}^{(0,n^-)}$ (see the discussion following Theorem 3.11).

[†] The image of the isotropic cone (see Def. 3.8) under the translation $\tau_{\alpha(p_j)}$, where $\tau_{\alpha(p_j)}(x) = x + \alpha(p_j)$.

This also follows from Corollary of Theorem 5.2.

For the statement of the next theorem we need the following definition.

Definition 4.2. A motion of a pseudoeuclidean space $(V,\Phi) = \mathbb{R}^{(n^+,n^-)}$ is a bijective mapping $\mu : V \longrightarrow V$ of the form $\mu = \tau_a \circ \gamma$, where $\gamma \in O(V,\Phi)$ (see Def. 3.14, 3.15), and τ_a ($a \in V$) is a translation of $V : \tau_a(x) = a+x$. It is easy to see that the set of all motions of (V,Φ) is a group, denoted $Is(\mathbb{R}^{(n^+,n^-)})$.

An important question which has not been answered so far is: *What is the relation between two vector representations of a finite pseudometric space?* The next theorem settles the matter.

Theorem 4.4. Let

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}$$

be a vector representation of a finite pseudometric space (P,Π). Then any vector representation β of (P,Π) can be represented in the form

$$\beta = \mu \circ \alpha$$

where $\mu \in Is(\mathbb{R}^{(n^+,n^-)})$ (see Fig. 4.5). The converse of this statement is also true.

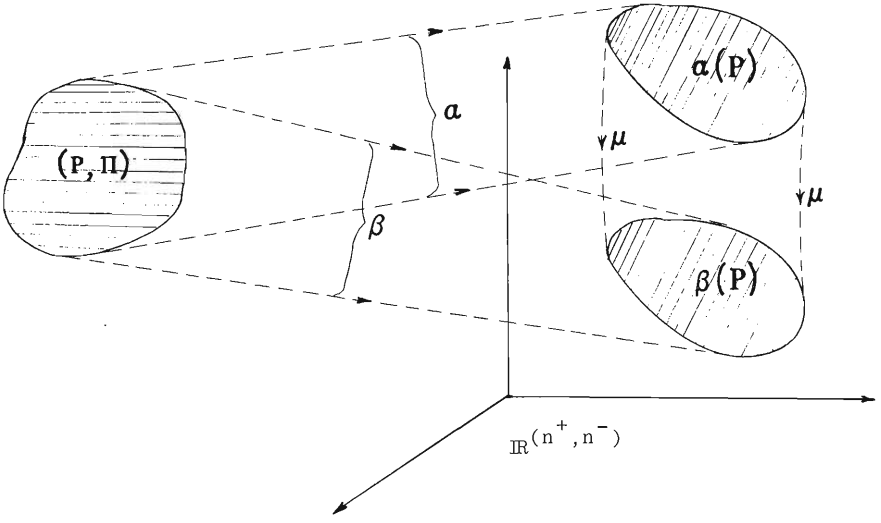


Figure 4.5

Proof. Suppose $P = \{p_i\}_{0\le i\le k}$, and α, β are given. Define translations τ_1 and τ_2 as $\tau_1(x) = x-\alpha(p_0)$, $\tau_2(x) = x-\beta(p_0)$, and set $\bar{\alpha} = \tau_1\circ\alpha$, $\bar{\beta} = \tau_2\circ\beta$. Then $\bar{\alpha}$ and $\bar{\beta}$ are vector representations of (P,Π). Let us choose (as we did in the beginning of the proof of Theorem 4.3) from the vectors $\bar{\alpha}(p_i)$ $(0\le i\le k)$ a basis $(a_j)_{1\le j\le n}$, $a_j = \bar{\alpha}(p_{i_j})$, of $\mathbb{R}^{(n^+,n^-)}$. By Theorem 3.15 the Gram determinant

$$g(a_1,\ldots,a_n) \neq 0.$$

Let $b_j = \bar{\beta}(p_{i_j})$ $(1\le j\le n)$, then,

$$g(b_1,\ldots,b_n) = \det((\Phi(b_s,b_j)_{1\le s,j\le n}),$$

and setting $p_{i_j} = p'_j$ we have

$$\Phi(b_s,b_j) = \Phi(\bar{\beta}(p'_s),\bar{\beta}(p'_j)) = \frac{1}{2}\,[\Phi(\bar{\beta}(p'_s),\bar{\beta}(p'_s)) + \Phi(\bar{\beta}(p'_j),\bar{\beta}(p'_j)) -$$

$$- \Phi(\bar{\beta}(p'_s)-\bar{\beta}(p'_j),\bar{\beta}(p'_s)-\bar{\beta}(p'_j))] = \frac{1}{2}\,[\Pi(p'_s,p'_0)^2 + \Pi(p'_j,p'_0)^2 -$$

$$- \Pi(p'_s,p'_j)^2] = \frac{1}{2}\,[\Phi(\bar{\alpha}(p'_s),\bar{\alpha}(p'_s)) + \Phi(\bar{\alpha}(p'_j),\bar{\alpha}(p'_j)) -$$

$$- \Phi(\bar{\alpha}(p'_s)-\bar{\alpha}(p'_j),\bar{\alpha}(p'_s)-\bar{\alpha}(p'_j))] = \Phi(\bar{\alpha}(p'_s),\bar{\alpha}(p'_j)) = \Phi(a_s,a_j).$$

Thus, $g(b_1,\ldots,b_n) = g(a_1,\ldots,a_n) \neq 0$, and by Theorem 3.15 $(b_j)_{1\le j\le n}$ is also a basis of $\mathbb{R}^{(n^+,n^-)}$.

Define an automorphism $\gamma : \mathbb{R}^{(n^+,n^-)} \longrightarrow \mathbb{R}^{(n^+,n^-)}$ as

$$\gamma(x) = \gamma\left(\sum_{j=1}^{n} x^j a_j\right) = \sum_{j=1}^{n} x^j b_j \qquad x\in\mathbb{R}^{(n^+,n^-)}.$$

Then $\gamma(a_j) = b_j$ $(0\le j\le n)$, and to prove that $\gamma(\bar{\alpha}(p_i)) = \bar{\beta}(p_i)$, $0\le i\le k$, (or $\gamma\circ\bar{\alpha} = \bar{\beta}$) it is enough to show the implication $\bar{\alpha}(p_i) = \sum_{j=1}^{n} x^j a_j \Longrightarrow \bar{\beta}(p_i) = \sum_{j=1}^{n} x^j b_j$, $n+1\le i\le k$. The last statement follows from the fact that the coordinates of $\bar{\alpha}(p_i)$ and $\bar{\beta}(p_i)$ are the unique solutions of the identical systems of n linear equations in n unknowns with the determinant $g(a_1,\ldots,a_n) = g(b_1,\ldots,b_n) \neq 0$ (see the end of the proof of Lemma 4.2).

From the statement preceding Definition 3.16 it follows that γ is an orthogonal automorphism. Substituting $\bar{\alpha} = \tau_1\circ\alpha$ and $\bar{\beta} = \tau_2\circ\beta$ into $\bar{\beta} = \gamma\circ\bar{\alpha}$, we obtain $\tau_2\circ\beta = \gamma\circ\tau_1\circ\alpha$ or $\beta = \tau_2^{-1}\circ\gamma\circ\tau_1\circ\alpha$, where τ_2^{-1} is a translation also: $\tau_2^{-1}(x) = x+\beta(p_0)$. It is easy to see that

$$\gamma \circ \tau_a = \tau_{\gamma(a)} \circ \gamma$$

for any endomorphism γ and any translation τ_a, and so

$$\beta = \tau_2^{-1} \circ \tau_3 \circ \gamma \circ \alpha,$$

where $\tau_3 = \tau_{-\gamma(\alpha(p_0))}$. At last,

$$\beta = \mu \circ \alpha,$$

where $\mu = \tau_2^{-1} \circ \tau_3 \circ \gamma$ is an element of $Is(\mathbb{R}^{(n^+,n^-)})$.

To prove the converse of the statement, suppose that a vector representation α of (P,Π) is given, and let us set

$$\beta = \mu \circ \alpha,$$

where $\mu \in OA(\mathbb{R}^{(n^+,n^-)})$. Then, since any motion of $\mathbb{R}^{(n^+,n^-)}$ is a distance preserving mapping (as a composition of two distance preserving bijections), β is also a vector representation of (P,Π). ∎

In particular, from the above theorem it follows that a renumbering of elements of P before the application of an embedding algorithm amounts to the motion of the image of the initial vector representation.

So, up to a motion of the pseudoeuclidean space $\mathbb{R}^{(n^+,n^-)}$, a vector representation

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}$$

is uniquely determined, and thus, one has sufficient justification to transfer the analysis of a finite pseudometric space (P,Π) from the original setting to $\mathbb{R}^{(n^+,n^-)}$. With this transition the metric pattern of (P,Π) is preserved by the vector representation of (P,Π). Thus, for example, if we take any number of points on a straight line and add one point equidistant to all of them the resulting pseudometric space is always representable in $\mathbb{R}^{(2,1)}$ and is always of the fixed configuration (see Examples 4.1, 4.2). This stability of a geometric pattern in a pseudoeuclidean space is not surprising, since this fundamental property has been implicitly assumed in the classical approach.

Now we are ready to state in a precise manner *one fundamental point* mentioned in chapter 1 and important for the entire future

development: *the only results in the space* $\mathbb{R}^{(n^+,n^-)}$ *relevant to the original data are those, that are invariant with respect to the group of motions* $\mathrm{Is}(\mathbb{R}^{(n^+,n^-)})$. This principle follows from the above theorem and a simple fact that all vector representations of a finite pseudometric space are equivalent.

Finally, I shall explain *the main idea of the proposed approach* as it relates to one of the most fundamental problems of pattern recognition. A typical example of such a problem is the problem of assigning an object o from a conceptual universe O to one of the several subclasses of O, for example, O_1 and O_2, forming the "images" generated by some abstract concept (see [23], p. 2). Of course, we wish to classify the object o on the basis of the information provided by a finite number of objects from each of the classes O_1 and O_2, even if the classes are infinite.

The suggested approach consists in the following. First, choose an appropriate (for the problem) representation set P corresponding to O. Second, define on P an appropriate (for the problem) distance function Π, which, in particular, would make P_1 and P_2, corresponding to O_1 and O_2, totally bounded sets in (P,Π). A set S in a pseudometric space P is <u>totally bounded</u> if $\forall \varepsilon > 0$ there exists a finite $\bar{S} \subseteq S$ such that

$$\forall_{s\in S} \quad \exists_{\bar{s}\in\bar{S}} \qquad \Pi(s,\bar{s}) \leq \varepsilon.$$

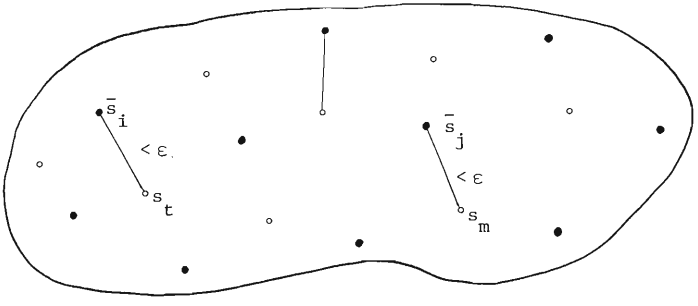


Figure 4.6

The above set $\bar{S}$ is called a finite <u>ε-net of S</u> (Fig. 4.6). The requirement of total boundedness of the sets P_1 and P_2 is quite natural, since every "recognizable" collection of sets is metrizable

in such a way as to become a collection of totally bounded sets. The relevant result from general topology is the following theorem ([42], p. 185): every separable metric space is metrizable by a totally bounded metric. Thus, for example, a compact set of a metric space is totally bounded.

Next, collect two finite sets of objects $\overline{O}_1$ and $\overline{O}_2$ (corresponding to finite ε-nets $\overline{P}_1$ and $\overline{P}_2$) which from the decision-theoretic point sufficiently well represent the images O_1, O_2. Then, construct a vector representation of a finite pseudometric space $\overline{P}_1 \cup \overline{P}_2 = \overline{P}$

$$\alpha : (\overline{P}, \overline{\Pi}) \longrightarrow \mathbb{R}^{(n^+, n^-)},$$

and eventually, produce a decision algorithm for $\alpha(\overline{P}_1)$ and $\alpha(\overline{P}_2)$ in $\mathbb{R}^{(n^+, n^-)}$.

Thus, vector representation α serves exactly the same purpose as a random vector - transplanting the original problem formulated in an abstract space to an analytically more convenient vector space. This convenience is immediately translated into power and efficiency of the relevant algorithms. The first two examples which come to my mind in connection with this are the Fisher's linear discriminant dimensionality reduction ([6], section 4.11) and the remarkable improvements in efficiency of the Nearest Neighbor classification algorithm achieved in the presence of the vector space structure [24].

The difference between the vector representation and a random vector is crucial, to the extent that the former is more flexible and can preserve finer information than the latter. In other words, as was already mentioned in chapter 1, the vector representation preserves *any* predefined dissimilarity measure and in this sense should be considered as a generalization of the concept of random vector (see also Appendix III).

Given a pattern $p \in P_1 \cup P_2$ to be classified, we know that it is sufficiently close to one of the elements in the training set $\overline{P}_1 \cup \overline{P}_2$, so that the projection of p onto the pseudoeuclidean space $\mathbb{R}^{(n^+, n^-)}$ should fall into one of the decision regions determined by $\alpha(\overline{P}_1)$ and $\alpha(\overline{P}_2)$ (Fig. 4.7).

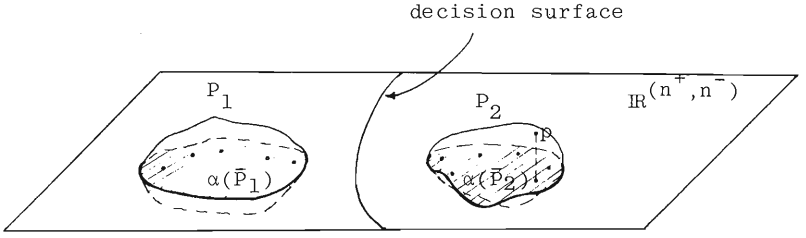


Figure 4.7

Thus, although the entire pseudometric space P may not be representable (isometrically) in a finite-dimensional pseudo-euclidean space, the totally bounded subspaces P_1 and P_2 can sufficiently accurately be represented in the finite-dimensional vector space. This fact is well known when P is a normed vector space. Determination of the "projection" of the element p onto $\mathbb{R}^{(n^+,n^-)}$ will be discussed in section 7.2.

In probabilistic language the above idea looks as follows (see also App. III). Let $H = (V,\Phi)$ be a vector space V (not necessarily finite-dimensional) together with some non-degenerate symmetric form defined on it. The theory of such spaces is relatively well developed (see [43]). Then one can show that for every pseudometric space P and a probability space (P,B,μ) there is a distance preserving Borel mapping

$$\beta : (P,B,\mu) \longrightarrow (H,B_H)$$

which can be considered as an H-valued random variable. In most of the problems of pattern recognition the range of β is a totally bounded set, therefore, we can approximate β by another random variable

$$\alpha : (P,B,\mu) \longrightarrow (\mathbb{R}^{(n^+,n^-)},B_{\mathbb{R}^n})$$

obtained from β by a projection onto a finite-dimensional subspace of H spanned by a finite ε-net of $\beta(P)$. Thus, it is the distance preserving property of the random variable β which makes it different from most of the classical random vectors.

5 GENERALIZED COVARIANCE MATRICES

In this chapter we introduce and study the notion of the covariance endomorphism of a subspace $(P,\Pi) \subseteq (P',\Pi')$ with respect to a vector representation of the finite pseudometric space (P',Π'). It is a true generalization of the corresponding classical notion, which is the single most fundamental notion in multivariate statistical data analysis. This generalization affords a better understanding of the notion itself and its connection with properties of the inner product *space where the data can be represented.* The covariance matrix, i.e., the matrix of the covariance endomorphism, is, in general, not a symmetric nor a non-negative matrix, but is a *symmetric and a non-negative matrix with respect to the form* Φ of the pseudoeuclidean space $\mathbb{R}^{(n^+,n^-)}$ where it is defined. In the case of a Euclidean space the latter notions reduce to the well known classical notions. The explicit dependence of the covariance matrix on the symmetric bilinear form Φ (representing the inner product in the vector space where the data can be represented) should further clarify the true meaning of this concept. In other words, in *a Euclidean space metric properties coincide with topological*, and this has created a deceptive situation, in which both of the above groups of properties (metric and topological) are identified. In a general pseudoeuclidean space one is forced to distinguish between the two groups of properties, and this should help, in particular, to better understand how *linear* and *metric* properties of the covariance matrix are blended.

It will be shown that all the metric properties of the classical covariance matrix with respect to the original finite sample remain true for its generalization. Moreover, the characteristic values of the generalized covariance matrix are real numbers, signs of which correspond to the signs of +1 and -1 in the canonical form of the matrix $M(\Phi)$, and which are invariant under any vector representation of the pseudometric space. Thus, according to a fundamental principle stated at the end of the last chapter, *these characteristic values are the true parameters of the initial finite pseudometric space, and hence of the original data.* Hence, as was mentioned in the Foreword, we now have at our disposal for analysis and classification of any pattern set one of the most powerful tools, *the tool which becomes available (for general patterns) only within*

the proposed approach. Of course, this is only one of the numerous bonuses.

As in the classical case, the "principal axes" also exist. The notion of a Gaussian inner product can be introduced as well, and thus the Mahalanobis distance may be defined.

Naturally, the generalization of the notions of the between and the within scatter matrices can also be introduced (and are introduced), and have the interpretation similar to their classical counterparts.

Several places in this and the next chapters will be understood better if the reader is familiar with the fundamental notion of the transpose of an endomorphism, for which we refer, for example, to reference [33], §16.5.

A somewhat more illuminating (from the theoretical point of view) general framework for introducing the covariance endomorphisms is via generalized random vectors (see Appendix III).

Since many problems in pattern recognition require simultaneous representation of several samples from some fixed population, the notions introduced in this chapter are general enough to be applied in these situations.

The fact that we have to consider simultaneously several samples is described formally as follows: the pseudometric space (P',Π') is formed by its several subspaces (P_j,Π_j), $1\leq j\leq m$. In the definitions below, when the introduced notion relates to one of the samples, index j is omitted.

So, let (P,Π), where $P = \{p_i\}_{1\leq i\leq k}$,[†] be a subspace of a finite pseudometric space (P',Π'), and let

$$\alpha : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)} \qquad (n^+ + n^- = n)$$

be a vector representation of (P',Π') (see Def. 4.1).

Definition 5.1. The mean vector (centre of gravity, barycentre) of (P,Π) with respect to the vector representation α is the vector

---

[†] Note, that the first index is now 1, and not 0, as in the last chapter. This notation will simplify some of the formulas in this chapter.

$$\bar{v} = \frac{1}{k} \sum_{i=1}^{k} v_i,$$

where $v_i = \alpha(p_i)$, $1 \leq i \leq k$.

When (P',Π') is a subset of the Euclidean space $\mathbb{R}^m$, a mapping $\alpha : P' \longrightarrow \mathbb{R}^n$, $\alpha(p_i) = p_i$, where $\mathbb{R}^n$ is an affine subspace of $\mathbb{R}^m$ spanned by P', is a vector representation of (P',Π'); so that our definition in this case gives the classical definition of the mean after the reduction of dimensionality of the given set of vectors P' is performed.

It is important to note the role of space (P',Π') in the above definition: if this space is replaced by another one, also containing (P,Π), then the vector representation α may, of course, change substantially, so that the mean of (P,Π) would change also (take P' to be the set $\{p_0,p_1,p_2,p_4,p_5,p_6,p_7\}$ in the Example 4.2, and P - any subset of the above set; then leaving P the same, change P' to the entire set of the same example). In other words, the notion of the mean depends on the universe with respect to which it is being considered. The vector representation α is not essential (see Theorem 4.4).

The mean vector of (P,Π) w.r.t. a vector representation α in some cases may not have any direct physical interpretation in terms of the original pseudometric space. But this fact should not perplex one, if one remembers, that, as in the classical case, *the entire process of transition from the patterns to the vector representation is always made for technical reasons only* (see chapter 1).

At the same time and what is more, the mean vector in a pseudo-euclidean space has the same metric properties as its classical counterpart. The first property of the classical mean can be stated as follows: it is the only point in $\mathbb{R}^n$, which has the minimal sum of the distances to the given k points. In our case this statement is also true, if a minimum of a function $\phi : \mathbb{R}^{(n^+,n^-)} \longrightarrow \mathbb{R}$ is defined as a point $x \in \mathbb{R}^{(n^+,n^-)}$, such that $\phi|V_+$ has minimum at $\pi_{V_+}(x)$ and $\phi|V_-$ has maximum at $\pi_{V_-}(x)$, where π_U is the orthogonal projection (section 3.3), and V_+ and V_- were introduced in Theorem 3.11. In other words, x is a minimum, if it is a saddle point of a special type, determined by the form Φ of $\mathbb{R}^{(n^+,n^-)}$. In terms of the first and second derivative the above condition is equivalent to the following: all the first derivatives at this point equal to zero, and the canonical form of the Hessian matrix at this point is the

same as that of the matrix $M(\Phi)$ of the form Φ of $\mathbb{R}^{(n^+,n^-)}$.

So, we want to prove that $\bar{v} = \frac{1}{k} \sum_{i=1}^{k} v_i$, $v_i \in \mathbb{R}^{(n^+,n^-)}$, is the only critical point of the function

$$\phi(x) = \sum_{i=1}^{k} \| x-v_i \|^2,$$

$\| x-v_i \|^2 = \Phi(x-v_i, x-v_i)$ (Def. 3.2), and the canonical form of the matrix $(D_{ij}\ \phi(\bar{v}))_{1\le i,j\le n}$ is

$$J = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix}.$$

Proof. Assuming that $v_i = \sum_{j=1}^{n} v_i^j e_j$, $x = \sum_{j=1}^{n} x^j e_j$, where $(e_j)_{1\le j\le n}$ is a Φ-orthonormal basis of $\mathbb{R}^{(n^+,n^-)}$, we have

$$\phi(x) = \sum_{i=1}^{k} \Big[\sum_{j=1}^{n^+} (x^j - v_i^j)^2 - \sum_{j=n^++1}^{n} (x^j - v_i^j)^2 \Big].$$

The critical points are the solutions of the system

$$D_j \phi = 0, \qquad 1\le j\le n,$$

which in our case is

$$\begin{cases} \sum_{i=1}^{k} 2(x^j - v_i^j) = 2k(x^j - \bar{v}^j) = 0, & 1\le j\le n^+, \\ -\sum_{i=1}^{k} 2(x^j - v_i^j) = -2k(x^j - \bar{v}^j) = 0, & n^+ + 1\le j\le n. \end{cases}$$

Thus, the only solution is $x^j = \bar{v}^j$, $1\le j\le n$, or $x = \bar{v}$. The Hessian matrix of ϕ is

$$2k \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix},$$

and the above statement is proved. ∎

Since other properties of the mean vector are connected with those of the covariance matrix we first introduce (using the notations of Definition 5.1) the following fundamental definition.

Definition 5.2. Let $(v_i-\bar{v})$, $1\le i\le k$, be the $n\times 1$ matrix of coordinates of the vector $v_i-\bar{v}$ with respect to a fixed Φ-orthonormal basis $(e_i)_{1\le i\le n}$ of $R^{(n^+,n^-)}$, and let us denote by A the $n\times k$ matrix whose i^{th} column is $(v_i-\bar{v})$. The $n\times n$ matrix

$$S = S(P) = S(P,\alpha) = A\cdot {}^tA\cdot J = [\sum_{i=1}^{k}(v_i-\bar{v})\cdot {}^t(v_i-\bar{v})]\cdot J,$$

where $J = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix}$, will be called the covariance matrix of the subspace (P,Π) with respect to the vector representation α of (P',Π').

As above, if $(P',\Pi')\subset R^m$, this definition gives the classical one after the reduction of dimensionality of the data has been performed (and α is a mapping that performs the reduction).

Again, it is important to note that the change of α is not essential at all, but the role of the space (P',Π') is important. *So that the above notion is not a function of the set P only, as it is in the classical case, but of the "universe" (P',Π') also.*

Let us write out the covariance matrix in full:

$$S=\sum_{i=1}^{k}\left(\begin{array}{cccc|ccc}
(v_i^1-\bar{v}^1)^2 & (v_i^1-\bar{v}^1)(v_i^2-\bar{v}^2) & \dots & (v_i^1-\bar{v}^1)(v_i^r-\bar{v}^r) & -(v_i^1-\bar{v}^1)(v_i^{r+1}-\bar{v}^{r+1}) & \dots & -(v_i^1-\bar{v}^1)(v_i^n-\bar{v}^n)\\
(v_i^2-\bar{v}^2)(v_i^1-\bar{v}^1) & (v_i^2-\bar{v}^2)^2 & \dots & (v_i^2-\bar{v}^2)(v_i^r-\bar{v}^r) & -(v_i^2-\bar{v}^2)(v_i^{r+1}-\bar{v}^{r+1}) & \dots & -(v_i^2-\bar{v}^2)(v_i^n-\bar{v}^n)\\
\dots & \dots & \dots & \dots & \dots & \dots & \dots\\
(v_i^r-\bar{v}^r)(v_i^1-\bar{v}^1) & (v_i^r-\bar{v}^r)(v_i^2-\bar{v}^2) & \dots & (v_i^r-\bar{v}^r)^2 & -(v_i^r-\bar{v}^r)(v_i^{r+1}-\bar{v}^{r+1}) & \dots & -(v_i^r-\bar{v}^r)(v_i^n-\bar{v}^n)\\
\hline
(v_i^{r+1}-\bar{v}^{r+1})(v_i^1-\bar{v}^1) & (v_i^{r+1}-\bar{v}^{r+1})(v_i^2-\bar{v}^2) & \dots & (v_i^{r+1}-\bar{v}^{r+1})(v_i^r-\bar{v}^r) & -(v_i^{r+1}-\bar{v}^{r+1})^2 & \dots & -(v_i^{r+1}-\bar{v}^{r+1})(v_i^n-\bar{v}^n)\\
\dots & \dots & \dots & \dots & \dots & \dots & \dots\\
(v_i^n-\bar{v}^n)(v_i^1-\bar{v}^1) & (v_i^n-\bar{v}^n)(v_i^2-\bar{v}^2) & \dots & (v_i^n-\bar{v}^n)(v_i^r-\bar{v}^r) & -(v_i^n-\bar{v}^n)(v_i^{r+1}-\bar{v}^{r+1}) & \dots & -(v_i^n-\bar{v}^n)^2
\end{array}\right),$$

where $v_i^j-\bar{v}^j$ $(1\le j\le n)$ are the coordinates of the vector $v_i-\bar{v}$ w.r.t. the basis (e_i) in the above definition, and $r=n^+$, $n=n^++n^-$. Obviously, the covariance matrix can be represented as

$$S = \begin{pmatrix} S_1 & -S_{12} \\ {}^tS_{12} & -S_2 \end{pmatrix},$$

where S_1 and S_2 are symmetric matrices, which are the classical covariance matrices of the orthogonal projections $\pi_V(\alpha(P))$ and $\pi_{\bar{V}}(\alpha(P))$. Whence, when $n^- = 0$ (Euclidean space), $S^+ = S_1$. The opposite signs on (off) the main diagonal may be interpreted as non-commensurability of the two groups of variables.

Now we are ready to state another property of the mean, which is a direct generalization of the corresponding classical fact.

Theorem 5.1. The sum of the characteristic values of the covariance matrix $S(P,\alpha)$ is equal to the sum of the squared distances (in $\mathbb{R}^{(n^+,n^-)}$) of points $v_i = \alpha(p_i)$, $1 \le i \le k$, from their mean vector. Since the trace of a matrix equals the sum of its characteristic values, we can express the statement as (see Fig. 5.1)

$$\mathrm{Tr}(S) = \sum_{i=1}^{k} \|v_i - \bar{v}\|^2.$$

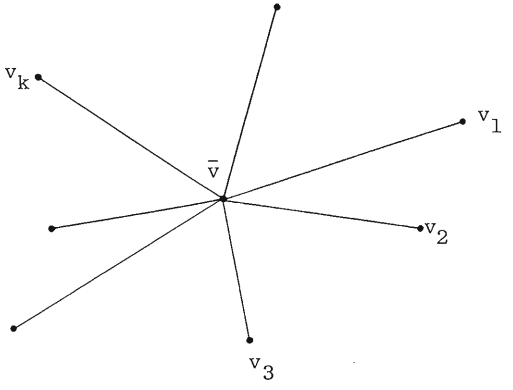


Figure 5.1

Proof. Indeed, from the above explicit expression for the covariance matrix we see that ($r = n^+$):

$$\mathrm{Tr}(S) = \sum_{j=1}^{r}\Big[\sum_{i=1}^{k}(v_i^j - \bar{v}^j)^2\Big] - \sum_{j=r+1}^{n}\Big[\sum_{i=1}^{k}(v_i^j - \bar{v}^j)^2\Big] = \sum_{i=1}^{k}\sum_{j=1}^{r}(v_i^j - \bar{v}^j)^2 -$$

$$- \sum_{i=1}^{k}\sum_{j=r+1}^{n}(v_i^j - \bar{v}^j)^2 = \sum_{i=1}^{k}\Big[\sum_{j=1}^{r}(v_i^j - \bar{v}^j)^2 - \sum_{j=r+1}^{n}(v_i^j - \bar{v}^j)^2\Big] =$$

$$= \sum_{i=1}^{k}\Phi(v_i - \bar{v}, v_i - \bar{v}). \quad \blacksquare$$

Moreover, the following generalization of the classical statement is also true.

Theorem 5.2. The sum of the characteristic values of $S(P,\alpha)$ is equal to the arithmetic average of the squared distances (in $\mathbb{R}^{(n^+,n^-)}$) between the points v_i $(1 \le i \le k)$:

$$\mathrm{Tr}(S) = \frac{1}{k} \sum_{\substack{i,j=1 \\ i<j}}^{k} \|v_i - v_j\|^2 .$$

Proof. By Theorem 5.1 and Def. 5.1

$$\mathrm{Tr}(S) = \sum_{i=1}^{k} \Phi(v_i-\bar{v}, v_i-\bar{v}) = \sum_{i=1}^{k} \Phi\left(v_i - \frac{\sum_{j=1}^{k} v_j}{k},\ v_i-\bar{v}\right) =$$

$$= \sum_{i=1}^{k} \Phi\left(\frac{\sum_{j=1}^{k} (v_i-v_j)}{k},\ v_i-\bar{v}\right) = \frac{1}{k} \sum_{i=1}^{k} \Phi\Big(\sum_{j=1}^{k} (v_i-v_j), v_i-\bar{v}\Big) =$$

$$= \frac{1}{k} \sum_{i=1}^{k} \sum_{j=1}^{k} \Phi(v_i-v_j, v_i-\bar{v}) = \frac{1}{k} \sum_{\substack{i=1 \\ i<j}}^{k} \sum_{j=1}^{k} [\Phi(v_i-v_j, v_i-\bar{v}) + \Phi(v_j-v_i, v_j-\bar{v})] =$$

$$= \frac{1}{k} \sum_{\substack{i,j=1 \\ i<j}}^{k} [\Phi(v_i-v_j, v_i-\bar{v}) - \Phi(v_i-v_j, v_j-\bar{v})] = \frac{1}{k} \sum_{\substack{i,j=1 \\ i<j}}^{k} \Phi(v_i-v_j, v_i-\bar{v}-v_j+\bar{v}) =$$

$$= \frac{1}{k} \sum_{\substack{i,j=1 \\ i<j}}^{k} \Phi(v_i-v_j, v_i-v_j). \quad \blacksquare$$

Corollary. The covariance matrix of a subspace of a finite pseudometric space with respect to a vector representation always has nonnegative trace. ∎

So far we have not shown that the characteristic values of the covariance matrix are real numbers. To prove this, we need several preliminary definitions and results, which are the generalizations of the corresponding classical notions and results.

Definition 5.3. An $n \times n$ matrix M will be called symmetric with

respect to a non-degenerate symmetric bilinear form Φ defined on a vector space V of dimension n, if there exists a Φ-orthonormal basis of V and a Φ-self-adjoint endomorphism γ (Def. 3.17), such that M is the matrix of γ w.r.t. this basis.

From Theorem 3.22 and the above definition it follows that the covariance matrix of (P,Π) w.r.t. a vector representation $\alpha : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)} = (V,\Phi)$ is a symmetric matrix w.r.t. the form Φ.

If the form Φ is positive (Def. 3.5), then the above notion coincides with that of a symmetric matrix (to check this, put $M(\Phi) = I$ in Theorem 3.22).

Definition 5.4. An n×n matrix M will be called non-negative (positive) with respect to a non-degenerate symmetric bilinear form Φ defined on a vector space V of dimension n, or simply Φ-non-negative (Φ-positive), if there exists a Φ-orthonormal basis of V and a non-negative (positive) w.r.t. Φ endomorphism γ (Def. 3.17), such that M is the matrix of γ w.r.t. this basis.

From the corollary to Theorem 3.23 and non-negativity of the classical covariance matrix it follows that the covariance matrix of (P,Π) w.r.t. a vector representation $\alpha : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)} = (V,\Phi)$ is a non-negative matrix w.r.t. the form Φ. From the same corollary it also follows, that if the vector signature of the subspace (P,Π) is the same as that of (P',Π'), then the covariance matrix (S,α) is a Φ-positive matrix. In particular, $S(P',\Pi)$ is always a Φ-positive matrix.

If the form Φ is positive, then the above notion coincides with the corresponding classical notion (set J = I in the corollary). However, we do not assume, that the covariance matrix of (P,Π) w.r.t. a vector representation α is of full rank.

Theorem 5.3. All characteristic values of the generalized covariance matrix are real numbers.

Proof. The theorem follows from Theorem 3.24 and the fact mentioned above, that the generalized matrix is a non-negative matrix w.r.t. Φ. ∎

Moreover, the following important proposition is true.

Theorem 5.4. Let (P,Π) be a subspace of a finite pseudometric space (P',Π') whose vector signature (n^+,n^-) is the same as that of (P',Π'). In particular, (P',Π') itself satisfies this condition. Then the covariance matrix $S(P,\alpha)$ has exactly n^+ positive and n^- negative characteristic values. Furthermore, there exists a Φ-orthonormal basis of $\mathbb{R}^{(n^+,n^-)}$ consisting of characteristic vectors of $S(P,\alpha)$.

Proof. The last statement follows from the fact mentioned above, that $S(P,\alpha)$ is a positive matrix w.r.t. Φ, and from Theorem 3.27.

Let us prove the first statement. Suppose that $(\bar{e}_i)_{1\le i\le n}$ is a Φ-orthonormal basis of $\mathbb{R}^{(n^+,n^-)}$ consisting of characteristic vectors of $S(P,\alpha)$, and suppose that γ is the positive endomorphism whose matrix is $S(P,\alpha)$. Then, denoting by c_i, $1\le i\le n$, the characteristic values of γ, we have

$$\Phi(\gamma(\bar{e}_i),\bar{e}_i) = \Phi(c_i\bar{e}_i,\bar{e}_i) = c_i\Phi(\bar{e}_i,\bar{e}_i) > 0,$$

where $\bar{e}_i \in \mathbb{R}^n(c_i;\gamma)$. Since $\Phi(\bar{e}_i,\bar{e}_i) > 0$, $1\le i\le n^+$, and $\Phi(\bar{e}_i,\bar{e}_i) < 0$, $n^+ + 1 \le i \le n^+ + n^-$, the required statement follows immediately. ∎

Again, if $(P',\Pi') \subset \mathbb{R}^m$, the covariance matrix $S(P,\alpha)$ is a classical covariance matrix for the vectors in $\mathbb{R}^{n^+} \subseteq \mathbb{R}^m$ (see the remark after Definition 5.2), so that in this case we have exactly n^+ positive characteristic values.

Since the most important information one can obtain from the covariance matrix is its characteristic values, the question of fundamental importance should be asked: Do the characteristic values of the covariance matrix of a subspace of a finite pseudometric space w.r.t. one vector representation coincide with those w.r.t. another vector representation? The answer is positive.

Theorem 5.5. Let

$$\alpha,\beta : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)}$$

be two vector representations of a finite pseudometric space (P',Π'), and let $S(P,\alpha)$, $S(P,\beta)$ be the covariance matrices of a subspace (P,Π) with respect to α and β respectively. Then the corresponding characteristic values of $S(P,\alpha)$ and $S(P,\beta)$ are equal.

Proof. By Theorem 4.4, vector representation β can be expressed as

$$\beta = \mu \circ \alpha \ ,$$

where $\mu = \tau \circ \gamma$, τ is a translation, and γ is a Φ-orthogonal endomorphism: $\gamma \in O(\mathbb{R}^{(n^+,n^-)})$. Since a translation does not change the covariance matrix, in order to prove the theorem it is enough to show that characteristic values of $S(P,\gamma\circ\alpha)$ are the same as those of $S(P,\alpha)$. Note, that $\gamma\circ\alpha$ is also a vector representation of (P',Π').

From Theorem 3.20b) it follows that

$$^{t}M(\gamma) \cdot J = J \cdot M(\gamma)^{-1},$$

where $M(\gamma)$ is the matrix of γ with respect to the fixed Φ-orthonormal basis (e_i) (see Definition 5.2). Denoting $\alpha(p_i)$ by v_i and using the last identity we have

$$S(P,\gamma\circ\alpha) = \{\sum_{i=1}^{k} [\gamma(v_i) - \gamma(\bar{v})] \cdot {}^{t}[\gamma(v_i) - \gamma(\bar{v})]\} \cdot J =$$

$$= \{M(\gamma) \cdot [\sum_{i=1}^{k} (v_i - \bar{v}) \cdot {}^{t}(v_i - \bar{v})] \cdot {}^{t}M(\gamma)\} \cdot J =$$

$$= M(\gamma) \cdot [\sum_{i=1}^{k} (v_i - \bar{v}) \cdot {}^{t}(v_i - \bar{v})] \cdot J \cdot M(\gamma)^{-1} =$$

$$= M(\gamma) \cdot S(P,\alpha) \cdot M(\gamma)^{-1},$$

so that $S(P,\gamma\circ\alpha)$ and $S(P,\alpha)$ are similar matrices, and therefore have identical characteristic values. ∎

The last formula in the above proof indicates that under an orthogonal endomorphism γ the covariance matrix S changes to $M(\gamma) \cdot S \cdot M(\gamma)^{-1}$, i.e., it changes as a matrix of an endomorphism of the vector space $\mathbb{R}^{(n^+,n^-)}$. This suggests that *the covariance matrix* $S(P,\alpha)$ *should be considered as a matrix of some endomorphism* σ, *which we shall call the covariance endomorphism induced by* (P,α) (see also App. III).

One important remark concerning the last notion is in order. The statistical tradition treats the covariance matrix *only* as a matrix of some inner product (see [44], Example 3.1.1 and section 7.2), and this is in spite of the role played by the corresponding characteristic values and vectors. Of course, all applied fields using statistical theory, pattern recognition in particular, repeat this inaccuracy. However, the proper setting for the vector spaces with inner products, which was unknown to the author until the very

last moment, has been developed since the 50's (see [45]), when the theory progressed to consider random variables with values in an infinite-dimensional Hilbert space. Random variables with values in vector spaces with indefinite symmetric bilinear forms have not been considered (to my knowledge). And, as one can see, the above definition, which for many reasons is the only legitimate analogue of the corresponding classical notion, absolutely excludes the option of treating the covariance matrix as a matrix of a symmetric bilinear form, simply because the matrix is not symmetric. Had the above mentioned developments influenced the exposition in the standard texts it would have saved the author a great deal of time and unnecessary doubts.†)

There are, of course, several reasons for the above misconception. The first reason comes from the fact that there are indeed two inner products - covariance (see App. III) and dual-normal (Def. 5.5 below) - induced by the covariance operator, and in the Euclidean space both of them have the same matrices as the covariance operator. The second reason is that in the classical case one is not, really, forced to make the choice between the two interpretations, simply because in the Euclidean space a matrix $M(\gamma)$ of an orthogonal endomorphism γ satisfies the identity ${}^{t}[M(\gamma)] = M(\gamma)^{-1}$, i.e., the classical covariance matrix changes (under γ) both as a matrix of an inner product and as matrix of an endomorphism. As was mentioned in the introduction to this chapter, this situation is explained by the fact that the topology of the Euclidean space is generated by its inner product.

Next, we shall show that, as in the classical case, the orthogonal transformation μ, which diagonalizes a full rank covariance matrix S, has matrix M^{-1}, where M is the matrix whose columns are the Φ-orthonormal basis formed by the characteristic vectors of S (see Theorem 4.4). Indeed, if D is the diagonal matrix of the characteristic values of S, then, as always, we have

$$S \cdot M = M \cdot D$$

or

$$M^{-1} \cdot S \cdot M = D.$$

---

†) When this work was almost finished the author found one book on multivariate statistics [53] which would have made a great difference. In fact, Appendix III is a direct generalization of the corresponding facts in [53]. Unfortunately, the book is now out of print. (There is a new edition [59]).

At the same time the matrix of the covariance endomorphism σ w.r.t. the Φ-orthonormal basis $(\mu(e_i))_{1\le i\le n}$, where (e_i) is the fixed basis (see Def. 5.2), is

$$\overline{S} = M^{-1}\cdot S\cdot M,$$

and the statement is proved.

Obviously, $\overline{S}$ is the covariance matrix of the finite pseudometric space (P,Π) with respect to the vector representation $\beta = \mu\circ\alpha$, where μ is the orthogonal automorphism with matrix M^{-1} (in the fixed basis $(e_i)_{1\le i\le n}$), and α is the original vector representation (see Theorem 4.4). Hence we can interpret the characteristic value of S as the total (generalized) variance of the sample in the corresponding direction, which is "uncorrelated" with the other n-1 directions (see App. III). Noting the identity $M^{-1} = J\cdot{}^tM\cdot J$, we see that the above statement is a complete analogue of the corresponding classical fact (see, for example, [46], p. 31).

The following three simple examples should give one some idea about the difference between the generalized and classical covariance matrices, as well as about their similarity.

Example 5.1.[†] Let us take a sample of ten points on the plane (see Fig. 5.2): four points (0,0), two points (1,0.1), and four points (4,0).

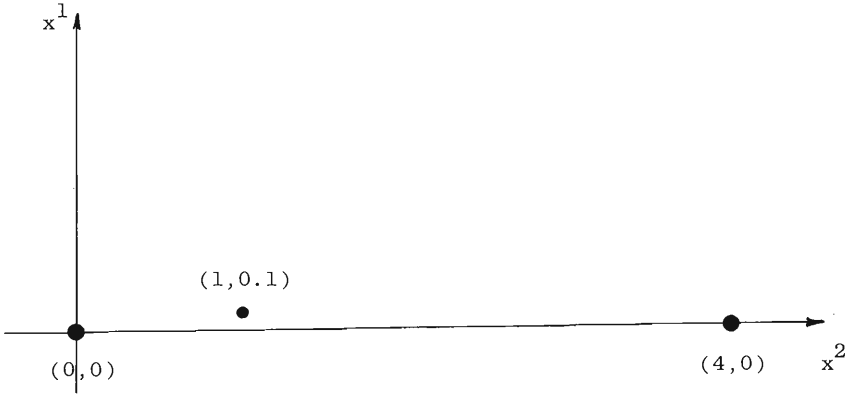


Figure 5.2

[†] To facilitate the geometric perception of the examples we consider the cases with coincident vectors.

Assuming first that these are vectors in the Euclidean plane, we compute the classical covariance matrix:

$$S' = \begin{pmatrix} 33.6 & 0.16 \\ 0.16 & 0.016 \end{pmatrix}.$$

Whence, the (rounded) characteristic values and the corresponding orthonormal characteristic vectors are

$$c_1' \doteq 33.6008, \qquad w_1' \doteq (0.9999884,\ 0.0047640);$$
$$c_2' \doteq 0.0152, \qquad w_2' \doteq (-0.0047640,\ 0.9999884).$$

Assuming now that the ten points are vectors in the pseudoeuclidean space $\mathbb{R}^{(1,1)}$, we also compute the covariance matrix (see Def. 5.2)

$$S'' = \begin{pmatrix} 33.6 & -0.16 \\ 0.16 & -0.016 \end{pmatrix}.$$

Whence, the (rounded) characteristic values and the corresponding orthonormal (w.r.t. Φ) characteristic vectors are

$$c_1'' \doteq 33.5992, \qquad w_1'' \doteq (1.0000113,\ 0.0047598);$$
$$c_2'' \doteq -0.0052, \qquad w_2'' \doteq (0.0047598,\ 1.0000113).$$

Comparing the results, we see that $c_1' > c_1''$ $c_2' > |c_2''|$. Of course, one should not forget that to "the same" vector sample in $\mathbb{R}^2$ and $\mathbb{R}^{(1,1)}$ correspond two different distance matrices, and, thus, two different original samples. A small variance in each of the "negative" directions indicates that the sample is "amost Euclidean", which guarantees the closeness of the variances in the "positive" directions to those computed in the Euclidean space: in the above case $|c_1' - c_1''|$ is small.

It is easy to see that if each of the "positive variables" is statistically independent of each of the "negative" ones, then the magnitudes of the characteristic values of the covariance matrix coincide with those computed in the Euclidean space. Indeed, in this case the generalized covariance matrix looks like

$$S'' = \begin{pmatrix} S_{n+} & O \\ O & -S_{n-} \end{pmatrix},$$

and the classical one looks like

$$S' = \begin{pmatrix} S_{n+} & 0 \\ 0 & S_{n-} \end{pmatrix},$$

whence the validity of the above statement follows immediately. Thus, the more symmetric with respect to each other the subspaces $\overline{V}_+$ and $\overline{V}_-$ (generated by the corresponding characteristic vectors of the covariance matrix)[†] are, the closer to each other are the variances computed in Euclidean and pseudoeuclidean spaces. The asymmetry can manifest itself in a remarkable way, as the following example shows.

Example 5.2. Let us take again a sample of ten points on the plane (see Fig. 5.3): four points (-1,0), four points (0,0), and two points (12,12).

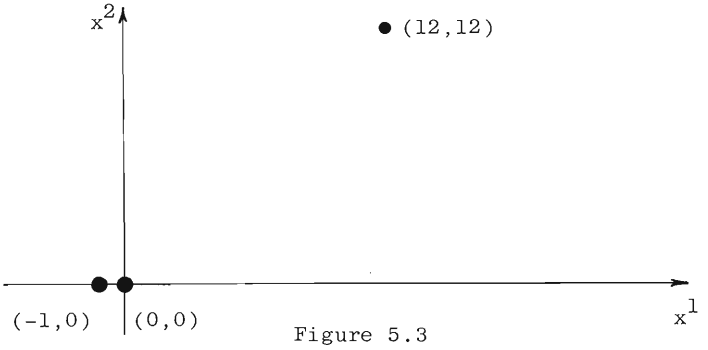


Figure 5.3

Assuming first that these are vectors in the Euclidean plane, we compute the classical covariance matrix:

$$S' = \begin{pmatrix} 252 & 240 \\ 240 & 230.4 \end{pmatrix}.$$

Whence, the (rounded) characteristic values and the corresponding orthonormal characteristic vectors are

$$c_1' \doteq 481.44288, \qquad v_1' \doteq (0.7228258,\ 0.6910302);$$
$$c_2' \doteq 0.95712, \qquad v_2' \doteq (-0.6910302,\ 0.7228258).$$

[†] See the remark following Theorem 3.11.

Next, assuming that the above ten points are vectors in $\mathbb{R}^{(1,1)}$, we also compute the covariance matrix

$$S'' = \begin{pmatrix} 252 & -240 \\ 240 & -230.4 \end{pmatrix}.$$

Whence, the (rounded) characteristic values and the corresponding orthonormal (w.r.t. Φ) characteristic vectors are (see Fig. 5.4)

$$c_1'' \doteq 34.829981, \qquad v_1'' \doteq (-2.3491969, -2.1257296);$$
$$c_2'' \doteq -13.229981, \qquad v_2'' \doteq (-2.1257296, -2.3491969).$$

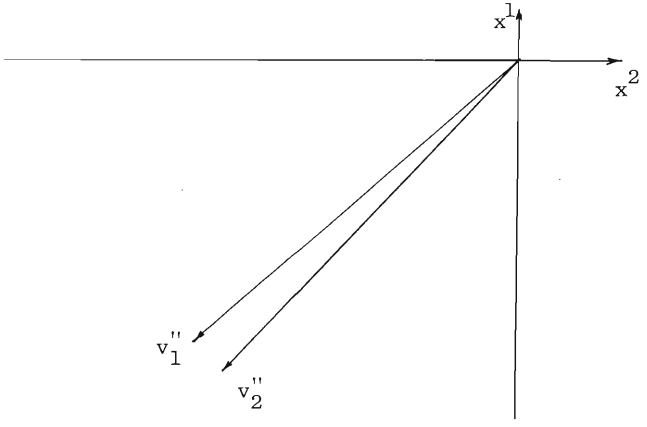


Figure 5.4

Comparing the results we see that

$$\frac{c_1'}{c_2'} \doteq 503, \qquad \frac{c_1''}{|c_2''|} \doteq 3,$$

which may at first seem puzzling. However, if we would have started the analysis from the very beginning, i.e., from the distance matrix, the result would appear quite natural. The point is that the distance in $\mathbb{R}^{(1,1)}$ between (0,0) and (12,12) is equal to 0, and the distance between (-1,0) and (12,12) is equal to 5, which explains the small ratio of the characteristic values.

The following example demonstrates that by means of the covariance matrix one can detect the intrinsic metric dimensionality of a set of vectors in $\mathbb{R}^{(n^+,n^-)}$, even when it does not coincide with the vector (Euclidean) dimension.

Example 5.3. Returning to Example 4.1 we compute the covariance matrix $S(\overline{P},\alpha)$, where $\overline{P} = \{p_0,p_1,p_2\} = P - \{p_3\}$,

$$S(\overline{P},\alpha) = \begin{pmatrix} 2 & 0.6030226 & -0.6030226 \\ 0.6030226 & 0.2424241 & -0.2424241 \\ 0.6030226 & 0.2424241 & -0.2424241 \end{pmatrix}.$$

This matrix has only one non-zero characteristic value 2, with the corresponding normalized characteristic vector

$$\begin{pmatrix} 1 \\ 0.3015113 \\ 0.3015113 \end{pmatrix}.$$

Similarly, we obtain only one non-zero characteristic value corresponding to the same characteristic vector, if in Example 4.2 we take $\overline{P}$ to be the set $P' - \{p_3\}$. Both of these results are in complete agreement with the fact that both of the original pattern sets are, indeed, of dimension one.

Thus, as in the Euclidean case, the generalized covariance matrix represents a very reliable tool for analysis of intrinsic (metric) geometry of the original pattern set, as well as its subsets, *regardless of the chosen form of pattern representation.*

Next, we consider the questions related to the generalized Mahalanobis distance. One way to introduce the Mahalanobis distance in the classical framework is *to declare the covariance matrix (when it is of full rank) to be a matrix of an inner product in the vector space generated by the chosen variable.* The last vector space can be identified with the vector space dual to the vector space where the data is represented (for example, if length ℓ is the first coordinate, then, given vector (2,3), we have $\ell(2,3) = 2$ and $\ell[(2,3) + (1,2)] = \ell(3,5) = 3 = \ell(2,3) + \ell(1,2)$, so that ℓ is a linear form). Thus, the above assumption that the covariance matrix is a matrix of an inner product in the dual space, is in complete agreement with the fact that it measures the dependence of the variables. By the theorem in Appendix II this inner product induces the dual inner product on the vector space where the data is represented; the distance in this new inner product space is the Mahalanobis distance. For a more detailed account along these lines see [44], Examples 2.1.1, 2.1.2; §§6.1, 7.2, 7.3.

The same construction can be carried out in a pseudoeuclidean vector space. However, some inaccuracy of the above approach must be corrected.

Definition 5.5. Let (P,Π), where $P = \{p_i\}_{1\le i\le k}$, be a subspace of a finite pseudometric space (P',Π'), let the vector signature of (P,Π) coincide with that of (P',Π')[†], and let

$$\alpha : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)} = (V,\Phi)$$

be a vector representation of (P',Π'). If ${}^t\sigma$ is the transpose of the covariance endomorphism induced by (P,α), then the inner product Ω^* on the dual space V^* defined as

$$\Omega^*(x,y) = \frac{1}{k}\,\Phi^*({}^t\sigma x, y) \qquad x,y\in V^*,$$

where Φ^* is the dual to Φ symmetric bilinear form (see Appendix II), will be called the dual-normal inner product induced by (P,Π) w.r.t. the vector representation α. The dual to the inner product Ω^*, inner product Ω on V, will be called the Gaussian (or normal) inner product induced by (P,Π) w.r.t. α.

So, when we pass to the dual space, we have to replace σ by ${}^t\sigma$, and to use ${}^t\sigma$ to define the above inner product. The fact that Ω^* is indeed an inner product is easily verified.

It is also easy to see that w.r.t. the orthonormal basis $(e_i^*)_{1\le i\le n}$ of V^* (dual to the fixed basis (e_i) of V (see Def. 5.2)) the matrix of Ω^* in the notations of Def. 5.2 is

$$M(\Omega^*) = \frac{1}{k}\,J\cdot {}^tS(P,\alpha) = \frac{1}{k}\,A\cdot {}^tA.$$

From the theorem in Appendix II it follows then that the matrix of Ω w.r.t. (e_i) is

$$M(\Omega) = \frac{1}{k}\,(A\cdot {}^tA)^{-1}.$$

Thus, the matrices $M(\Omega^*)$ and $M(\Omega)$ do not explicitly depend on the original form Φ, i.e., they coincide with their classical counter-

[†] The last assumption is made only to simplify the exposition, and will be in force in the rest of the chapter.

parts. However, there is a fundamental distinction, which can be seen from two angles. Informally, two distance matrices of a sample in a vector space V computed w.r.t. two different symmetric bilinear forms are different, so that although the two samples have the same normal inner products, they have very different geometric structure and the covariance matrices. By "geometric structure" we mean not only the structure in the vector space V, but the structure in the vector space V *together with* a non-degenerate symmetric bilinear form Φ defined on V. This brings us to the correct (formal) description of the situation: form Ω is defined on the pseudoeuclidean space (V,Φ), and thus cannot be considered without the original form Φ, determined by the data and defining the notion of orthogonality in the vector space. To put it in practical terms, *form Ω should be diagonalized (if at all) by Φ-orthogonal endomorphism.* The last is in a complete agreement with the fundamental point which was made in the discussion following the proof of Theorem 4.4: the "legitimate" linear transformations are those from the group $O(\mathbb{R}^{(n^+,n^-)})$. It is important to note, that the fixation of an orthogonal group acting on an inner product results in the uniqueness of the diagonal form of the matrix of the inner product.

As in the Euclidean case, the inverse of the transformation diagonalizing the covariance matrix S diagonalizes the matrix of Ω. Indeed, applying the transformation with the matrix M, where M is the matrix whose columns are the coordinates of the Φ-orthonormal basis formed by the characteristic vectors of S, we find the new matrix of Ω (see (3.4)):

$$\overline{M}(\Omega) = {}^tM \cdot M(\Omega) \cdot M = {}^tM \cdot (\frac{1}{k}\, {}^tS^{-1} \cdot J) \cdot M = \frac{1}{k}\, {}^tM \cdot {}^tS^{-1} \cdot {}^tM^{-1} \cdot J =$$

$$= \frac{1}{k}\, {}^t(M^{-1} \cdot S \cdot M)^{-1} \cdot J = \frac{1}{k}\, D^{-1} \cdot J,$$

where D is the diagonal matrix of the characteristic values of the covariance matrix.

Thus, as in the classical case, the matrix of the normal inner product w.r.t. the Φ-orthonormal basis formed by the characteristic vectors of the covariance matrix is diagonal.

Another way of introducing the Mahalanobis distance connected with the following problem.

Problem. Among all Φ-positive endomorphisms $\gamma : V \longrightarrow V$ of fixed

determinant, say $\det(\gamma) = 1$, find one that minimizes the function $\Phi : \mathrm{End}(V) \longrightarrow \mathbb{R}$

$$\phi(\gamma) = \frac{1}{2k^2} \sum_{i,j=1}^{k} \phi(\gamma(v_i - v_j), v_i - v_j)$$

where as above, v_i, $1 \le i \le k$, are the vectors representing (P, Π) in (V, Φ).

The above function ϕ is, in fact, the *mean sum of the squared distances between all* v_i computed in the pseudoeuclidean space (V, Ψ), where

$$\Psi(x,y) \overset{\text{def}}{=} \frac{1}{2k^2} \Phi(\gamma(x), y).$$

Solution. Let us fix some Φ-orthonormal basis of V. Then, by the Corollary to Theorem 3.23 the matrix of γ in this basis looks like $M(\gamma) = J \cdot X$, where X is a positive matrix. Thus, the whole problem reduces to the following one: find the constrained minimum

$\phi : \mathbb{R}^{\frac{n(n+1)}{2}} \longrightarrow \mathbb{R}$ in the open set $\{X \mid X > 0\}$

$$\phi(X) = \frac{1}{2k^2} \sum_{i,j=1}^{k} {}^{t}(v_i - v_j) X (v_i - v_j),$$

$$\det(X) = 1 \quad \text{(constraint)},$$

where v_i stands now for the coordinate column of the vector v_i. So, the original problem is equivalent to finding an inner product on V minimizing the sum of the squared distances computed w.r.t. this inner product.

We are going to use Lagrange's method of undetermined coefficients. Setting

$$\psi(X) = \phi(X) + m(1 - \det X),$$

we first compute the derivative of ϕ:

$$D\phi(X) = \frac{1}{2k^2} \sum_{i,j=1}^{k} (v_i - v_j)\, {}^{t}(v_i - v_j) = \frac{1}{2k^2} \sum_{i,j=1}^{k} (v_i\, {}^{t}v_i - 2 v_i\, {}^{t}v_j + v_j\, {}^{t}v_j) =$$

$$= \frac{1}{2k^2} \sum_{i,j=1}^{k} v_i\, {}^{t}v_i - \frac{1}{k^2} \sum_{i,j=1}^{k} v_i\, {}^{t}v_j + \frac{1}{2k^2} \sum_{i,j=1}^{k} v_j\, {}^{t}v_j = \frac{1}{2k^2} \sum_{i=1}^{k} k v_i\, {}^{t}v_j -$$

$$- [\frac{1}{k} \sum_{i=1}^{k} v_i {}^t\bar{v}] + \frac{1}{2k^2} \sum_{j=1}^{k} k v_j {}^t v_j = \frac{1}{k} \sum_{i=1}^{k} v_i {}^t v_i - [\frac{2}{k} \sum_{i=1}^{k} v_i {}^t\bar{v} - \frac{1}{k} \sum_{i=1}^{k} v_i {}^t\bar{v}] =$$

$$= \frac{1}{k} \sum_{i=1}^{k} (v_i {}^t v_i - 2 v_i {}^t\bar{v} + \bar{v} {}^t\bar{v}) = \frac{1}{k} \sum_{i=1}^{k} (v_i - \bar{v}) {}^t(v_i - \bar{v}) = M,$$

where M is the matrix of the dual-normal inner product of $\{v_i\}_{1 \le i \le k}$. Since

$$D[\det(X)] = (\det X)^{-1} \cdot {}^t X^{-1}$$

(see, for example [47], p. 72), and $\det X = 1$, we have

$$D\psi(X) = D\phi(X) - mD[\det(X)] = M - m {}^t X^{-1}.$$

Solving the equation $D\psi(X) = 0$ we find

$$X^{-1} = \frac{1}{m} M$$

or

$$X = mM^{-1}.$$

From condition $\det X = 1$ we can find m: $m = (\det M)^{\frac{1}{n}} > 0$. Now we shall outline the proof that point $X_0 = (\det M)^{\frac{1}{n}} \cdot M^{-1}$ is the sought point in $\mathbb{R}^{\frac{n(n+1)}{2}}$. Since, by Lagrange's theorem, the local constrained minimum (if it exists) must satisfy the equation $D\psi(X) = 0$, in order to prove the last statement it is enough to show that the global constrained minimum exists (X_0 is the only solution of the equation $D\psi(x) = 0$ which belongs to the set $K = \{X | X > 0, \det(X) = 1\}$). The last minimum exists because the set $\{X \mid X > 0, \det(X) \ge 1\}$, bounded by the cone $\{X > 0\}$, is closed and convex, and the positive linear function ϕ is monotone: $X_1 - X_2 > 0 \Longrightarrow \phi(X_1) - \phi(X_2) > 0$.

The restriction $\det(X) = 1$ in the above problem is of no special purpose, except to fix the volume of X. Thus, if we set the restriction $\det(X) = (\det M)^{n-1}$, with M as above, we would obtain that the minimum is achieved at $X_0 = M^{-1}$.

From the problem we conclude that *the Mahalanobis distance is the distance in the vector space* V *with the inner product determined by the symmetric bilinear form minimizing the sum of the squared distance between the given vectors.*

Finally, we shall generalize the notions of the within and between scatter matrices.

Let (P_j,Π_j), $1\le j\le m$, be subspaces of a finite pseudometric space (P',Π'), and let

$$\alpha : (P',\Pi') \longrightarrow \mathbb{R}^{(n^+,n^-)} \qquad (n^+ + n^- = n)$$

be a vector representation of (P',Π'). Denote by (P,Π) a subspace of (P',Π') such that $P = \bigcup_{j=1}^{m} P_j$. Suppose that $|P_j| = k_j$ $(1\le j\le m)$, $\sum_{j=1}^{m} k_j = k$, and that $\bar{v}_j$ and $\bar{v}$ are the mean vectors of (P_j,Π_j) and (P,Π) with respect to α correspondingly (Def. 5.1).

Definition 5.6. The matrix

$$S_w = S_w(\alpha) = S_w(\{P_j\}_{1\le j\le m},\alpha) \overset{\text{def}}{=} \sum_{j=1}^{m} S_j,$$

where S_j is the covariance matrix of (P_j,Π_j) with respect to α, will be called the within P_j scatter matrix (with respect to α). And the matrix

$$S_b = S_b(\alpha) = S_b(\{P_j\}_{1\le j\le m},\alpha) \overset{\text{def}}{=} \bar{A}\cdot {}^t\bar{A}\cdot J =$$

$$= \Big[\sum_{j=1}^{m} k_j(\bar{v}_j-\bar{v})\cdot {}^t(\bar{v}_j-\bar{v})\Big]\cdot J,$$

where $(\bar{v}_j-\bar{v})$, $1\le j\le m$, is the $n\times 1$ matrix of coordinates of the vector $\bar{v}_j-\bar{v}$ with respect to the fixed orthonormal basis (e_i) of $\mathbb{R}^{(n^+,n^-)}$ (see Def. 5.2), and $\bar{A}$ is the $n\times k$ matrix whose first k_1 columns coincide with $(\bar{v}_1-\bar{v})$ and so on, will be called the between P_j scatter matrix (with respect to α).

Theorem 5.6. Matrices S_b and S_w are positive w.r.t. Φ, and thus have real characteristic values.

Proof. From Definition 5.2 it follows that

$$S_j = B_j\cdot J \qquad 1\le j\le m;$$

where $B_j > 0$. Hence,

$$S_w = \sum_{j=1}^{m} S_j = \left(\sum_{j=1}^{m} B_j\right) \cdot J.$$

Since $\sum_{j=1}^{m} B_j > 0$, the theorem follows from Theorem 3.24 and the corollary preceding it. ∎

As in the case of the covariance matrix the following theorem is true.

Theorem 5.7. Characteristic values of S_w and S_b are invariant under any vector representation of (P', Π').

Proof. The proof of the theorem is similar to that of Theorem 5.5. ∎

Moreover, the well known decomposition of the covariance matrix remains true.

Theorem 5.8. If S is the covariance matrix of a subspace (P, Π) of a finite pseudometric space (P', Π') w.r.t. a vector representation α of (P', Π'), and $P = \bigcup_{j=1}^{m} P_j$, then

$$S = S_w + S_b,$$

where S_w is the within P_j scatter matrix (w.r.t. α), and S_b is the between P_j scatter matrix (w.r.t. α).

Proof. The statement follows from the corresponding classical equality by multiplying both sides of it on the right by J. ∎

6 DIMENSION AND DIMENSIONALITY REDUCTION

In this chapter some basic points related to the dimensionality reduction will be discussed. First, the generalization of classical principal component analysis is presented. Then, it will be shown that the first embedding algorithm (see chapter 4) can, in fact, be modified into the *main embedding algorithm*, which constructs a reduced vector representation in a vector space of any desirable dimension. This fact is of fundamental importance for most problems.

It is important to note that in many problems a reduced *representation in a Euclidean space* can be obtained, and two ways of constructing such representations are indicated.

In the last part of the chapter some comments regarding the so-called "non-metric multidimensional scaling" are made.

First of all, it should be stressed once more, that, to start with, a vector representation

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)} \qquad (n^+ + n^- = n,\ P = \{p_i\}_{0 \le i \le k})$$

of a finite pseudometric space (P,Π) (Def. 4.1) detects the *intrinsic "geometric" dimension* of (P,Π). So, some bothersome possibilities arising in the classical approach such as "small sample size" problems ($k < n$), or the possibility of "highly linearly dependent data" (the subspace generated by the sample is small) are automatically excluded:

"Current theories of statistical inference are not yet equipped to understand what can be learned from small samples on many variables. The author believes that such understanding, when it comes, will not be encouraging and that data collectors should not have high hopes from such data" ([44], p. 242).

In the proposed approach, however, in the event that (P,Π) is a (natural) subspace of some uncountable ("continuous") pseudometric space, the unavoidable imperfectness of the measurements, i.e., of the function Π, results (as it does in the classical case) in the possible "swelling" of the vector representation space $\mathbb{R}^{(n^+,n^-)}$. What we have in mind is the possibility conditioned by the following

statement, the proof of which can be found in [48], p. 4.

Theorem 6.1. Let $\{x_0, x_1, \ldots, x_k\}$ be a set of points in $\mathbb{R}^n$, $k \leq n$. Then, in an arbitrarily small neighbourhood of each point x_i there exists a point y_i, such that the set $\{y_0, y_1, \ldots, y_k\}$ generates a subspace of dimension k. ∎

Thus, for example, if we take $k+1$ different points on a straight line ℓ in Euclidean space $\mathbb{R}^n$, $k \leq n$, measure the distances between the points, and then try to reconstruct the original set of points from the distance matrix (using any method), it is very unlikely that the minimal dimension of the vector space where we can reconstruct the above points will be equal to one.

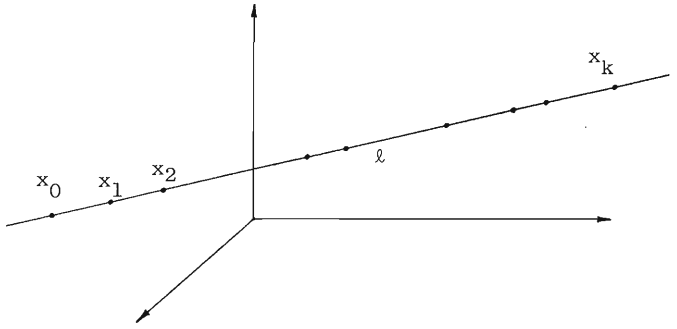


Figure 6.1

The reason behind this is very simple: since our measurements cannot be perfect, the distance matrix does not represent an *ideal* straight line ℓ. However, as far as we are concerned, the reconstructed line will be a straight line for all *practical* purposes, and there are, of course, a number of techniques to reveal this fact (see below).

On the other hand, the following statement is true.

Theorem 6.2. Let $\{x_0, x_1, \ldots, x_k\}$ be a set of points in $\mathbb{R}^m$ which generates a subspace of dimension n, $n \leq m$. There exists $\varepsilon > 0$ such that if $d(x_i, y_i) < \varepsilon$, $0 \leq i \leq k$, then the set $\{y_0, y_1, \ldots, y_k\}$ generates a subspace of dimension not less than n.

The proof is similar to that of Theorem 1 in [48], p. 7. ∎

For us the above two statements have the following implication: by increasing the accuracy of the measurements (i.e., of Π) we can achieve the situation, when the "geometric" dimension n of the finite pseudometric space (P,Π) is greater than or equal to the "ideal geometric" dimension, but we can never be guaranteed that by this process the exact dimension will be obtained.

Thus, in the cases involving continuous parameters the problem of dimensionality reduction is inevitable. In fact, the same problem confronts us in the classical approach. Next, we shall discuss the generalization of one of the classical techniques, principal component analysis, which can be used to solve the above problem, as well as any other problems requiring dimensionality reduction.

So, let

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)} \qquad P = \{p_i\}_{0\le i\le k}$$

be a vector representation of (P,Π) which is a finite subspace of some uncountable pseudometric space, and let $S = S(P,\alpha)$ be the covariance matrix of (P,Π) with respect to α (see Def. 4.2). We intend to use *invariant* quantitative information contained in S to detect the "true" intrinsic dimension of (P,Π). As we already know (Def. 5.5), the covariance endomorphism σ generates in the vector space dual to the vector representation space the dual-normal inner product Ω^*.

Let us diagonalize the matrix of Ω^*, remembering (see the discussion following Theorem 4.4) that the only "legitimate" linear transformations are the elements of the orthogonal group $O(n^+,n^-)$ (see Def. 3.15). By the theorem in Appendix II, the dual space $[\mathbb{R}^{(n^+,n^-)}]^*$ is isomorphic to $\mathbb{R}^{(n^+,n^-)}$. Thus, the matrix of Ω^* w.r.t. the basis $(e_i^*)_{1\le i\le n}$ dual to the fixed orthonormal basis $(e_i)_{1\le i\le n}$ (see Def. 5.2) is

$$M(\Omega^*) = \frac{1}{k+1}\, J \cdot {}^tS,$$

where $J = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix}$.

Let

$$\mu : \left[\mathbb{R}^{(n^+,n^-)}\right]^* \longrightarrow \left[\mathbb{R}^{(n^+,n^-)}\right]^*$$

be the transition endomorphism from the basis (e_i^*) to the basis $(\bar{e}_i^*)$ formed by the orthonormal characteristic vectors of ${}^t\sigma$, the transpose of the scatter endomorphism σ. By the remark just preceding Def. 3.16, μ is a Φ-orthogonal endomorphism, i.e., the matrix of μ w.r.t. the basis $(e_i^*)_{1\le i\le n}$, $M(\mu) = M$, by Theorem 3.20 satisfies the equation

$$(6.1) \qquad {}^tM \cdot J \cdot M = J.$$

Since M is also a matrix of characteristic vectors of ${}^tS$, we have

$$(6.2) \qquad {}^tS \cdot M = M \cdot D,$$

where D is a diagonal matrix of the corresponding characteristic values. Now, we can find the unique diagonal form of Ω^* w.r.t. the group $O(n^+,n^-)$. The matrix of Ω^* w.r.t. the basis $(\bar{e}_i^*)$ is (see 3.4)

$$\bar{M}(\Omega^*) = {}^tM \cdot \frac{1}{k+1}(J \cdot {}^tS) \cdot M = \frac{1}{k+1} J \cdot M^{-1} \cdot {}^tS \cdot M = \frac{1}{k+1} J \cdot D,$$

where from (6.1) ${}^tM \cdot J = J \cdot M^{-1}$, and the last equality follows from (6.2).

Since the dual-normal inner product, in fact, assigns weights to the directions in the vector space $\mathbb{R}^{(n^+,n^-)}$ according to the proportion of the variance of the sample in the corresponding direction, we conclude from the above formula that the numbers $|c_i|/k$, $1\le i\le n$, where c_i is the characteristic value of S, are these weights for the "uncorrelated" direction represented by the basis $(\bar{e}_i^*)_{1\le i\le n}$. It is of crucial importance that *these weights are independent of the original vector representation of the sample* (P,Π) (see Theorem 5.5).

So, removing the directions corresponding to the relatively negligible weights, we solve the above problem of eliminating "the noisy dimensions". Thus, for example, using this method one can find a true one-dimensional representation of the $k+1$ points considered in the example following Theorem 6.1; similarly, one can detect that the set of structural patterns in Fig. 6.2 (where the distance between the patterns is the ordinary Hamming distance) is essentially one-dimensional.

At this point one could raise the following *question*, related

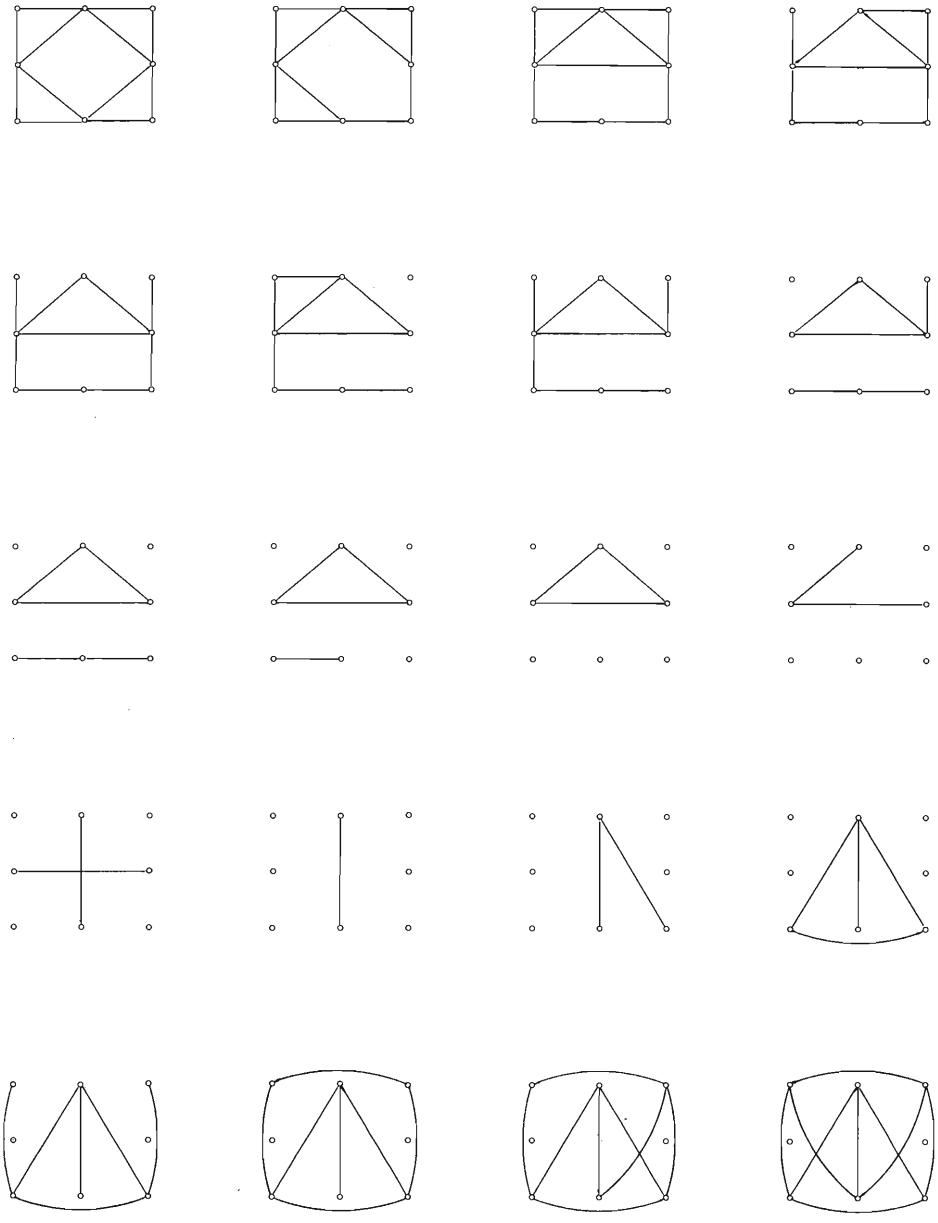


Figure 6.2

to the application of the above method: Is it computationally efficient first to embed the sample into the vector space most of the dimensions of which are created by the noise, and then, using this "swollen" high-dimensional vector representation, to detect the true vector representation of the data?

Quite fortunately, there is a very simple way to deal with the problem. The key is the following result.

Theorem 6.3. Let the mean vector of a finite pseudometric space (P,Π) w.r.t. a vector representation α (see Def. 5.1) be zero vector. Then, the non-zero characteristic values of the matrix of the bilinear form of (P,Π) (see Theorem 3.1) and those of the covariance matrix S of (P,Π) w.r.t. α (see Def. 5.2) are identical.

Proof. Suppose $P = \{p_i\}_{0\le i\le k}$, $\alpha(p_0) = 0 = \frac{1}{k+1}\sum_{i=0}^{k}\alpha(p_i)$, and let us go back to the first embedding algorithm described in the paragraph related to the expression (4.2). Using the corresponding notations, we can say that if the vector representation α in the statement of the theorem is constructed according to this algorithm, then the matrix

$$\tilde{T} = L \cdot \bar{D}^{\frac{1}{2}}$$

is equal to the matrix

$$({}^t\bar{A} \mid O_{k\times(k-n)}),$$

where $\bar{A}$ is obtained from A in Def. 5.2 by removing the first column of 0's, and where $O_{k\times(k-n)}$ is a zero matrix of the indicated size. So, we have

$$\begin{pmatrix} S & O \\ O & O_{k-n} \end{pmatrix} = \begin{pmatrix} \bar{A} \\ O_{(k-n)\times k} \end{pmatrix} \cdot \begin{pmatrix} {}^t\bar{A} & O_{k\times(k-n)} \end{pmatrix} \cdot \begin{pmatrix} J & O \\ O & O_{k-n} \end{pmatrix} =$$

$$= {}^t\tilde{T} \cdot \tilde{T} \cdot \begin{pmatrix} J & O \\ O & O_{k-n} \end{pmatrix} = {}^t(L\cdot\bar{D}^{\frac{1}{2}})\cdot(L\cdot\bar{D}^{\frac{1}{2}}) \cdot \begin{pmatrix} J & O \\ O & O_{k-n} \end{pmatrix} =$$

$$= \bar{D}^{\frac{1}{2}} \cdot {}^tL\cdot L\cdot\bar{D}^{\frac{1}{2}} \cdot \begin{pmatrix} J & O \\ O & O_{k-n} \end{pmatrix} = \bar{D}^{\frac{1}{2}}\cdot I_k \cdot \bar{D}^{\frac{1}{2}} \cdot \begin{pmatrix} J & O \\ O & O_{k-n} \end{pmatrix} = \bar{D} \cdot \begin{pmatrix} J & O \\ O & O_{k-n} \end{pmatrix} = D,$$

where D is the diagonal matrix of the characteristic values of the matrix of the bilinear form. Thus, the theorem is proved. ∎

From the above theorem it follows immediately, that if among the p_i's, $p_i \in P$, $0 \le i \le k$, there was a p_j with the property: $\alpha(p_j) = \bar{v}$, then by renumbering the p_i's in such a way as to have $p'_0 = p_j$, we would be able to use the characteristic values of the corresponding matrix of the symmetric bilinear form of (P,Π) as a very reliable guide in choosing the reduced vector representation, *without constructing the complete vector representation.* Obviously, the above assumption (about existence of p_j) is not true for most of the samples. However, it is not difficult to see, how the matrix of the squared distances of a given sample can be expanded into one corresponding to the new sample whose elements are the elements of the original sample plus one new element $p : [\Pi'(p,p_i)]^2 = \|\bar{v} - v_i\|^2$, where $v_i = \alpha(p_i)$, $\bar{v}$ is the mean vector of (P,Π) w.r.t. α. Indeed, using our standard notations (see Theorem 4.1)

$$[\Pi'(p,p_i)]^2 = \|\bar{v} - v_i\|^2 = \Phi(\bar{v} - v_i,\ \bar{v} - v_i) = \Phi(\bar{v},\bar{v}) +$$

$$\Phi(v_i,v_i) - 2\Phi(\bar{v},v_i) = \frac{1}{(k+1)^2} \sum_{i=0}^{k} \sum_{j=0}^{k} \Phi(v_i,v_j) + d_{0i}^2 - \frac{2}{k+1} \sum_{j=0}^{k} (v_j,v_i) =$$

$$= \frac{1}{(k+1)^2} \frac{1}{2} \sum_{i=0}^{k} \sum_{j=0}^{k} (d_{0i}^2 + d_{0j}^2 - d_{ij}^2) + d_{0i}^2 - \frac{2}{k+1} \cdot \frac{1}{2} \sum_{j=0}^{k} (d_{0i}^2 + d_{0j}^2 - d_{ij}^2) =$$

$$= \frac{1}{2(k+1)} \left(\sum_{i=0}^{k} d_{0i}^2 + \sum_{j=0}^{k} d_{0j}^2 \right) - \frac{1}{2(k+1)^2} \sum_{i=0}^{k} \sum_{j=0}^{k} d_{ij}^2 - \frac{1}{k+1} \sum_{j=0}^{k} (d_{0j}^2 - d_{ij}^2) =$$

$$= \frac{1}{k+1} \sum_{j=0}^{k} d_{ij}^2 - \frac{1}{2(k+1)^2} \sum_{i=0}^{k} \sum_{j=0}^{k} d_{ij}^2 .$$

Let $P' = P \cup \{p\}$, where p is as above, and let us index the elements of P' as follows: $p'_0 = p$, $p'_1 = p_0, \ldots, p'_{k+1} = p_k$. Then, obviously, the vector signature of (P',Π') (see Def. 4.1) is the same as that of (P',Π), and the matrix of the symmetric bilinear form of (P',Π') is

$$(6.3) \qquad M(\Psi') = (m'_{rs})_{1 \le r,s \le k+1} = \Big(\frac{1}{2} \Big[\frac{1}{k+1} \Big(\sum_{i=0}^{k} d_{ij}^2 + \sum_{j=0}^{k} d_{ij}^2 \Big) -$$

$$- \frac{1}{(k+1)^2} \sum_{i=0}^{k} \sum_{j=0}^{k} d_{ij}^2 - d_{ij}^2 \Big]\Big)_{0 \le i,j \le k},$$

where $d_{ij} = \Pi(p_i, p_j)$. The vector representation algorithm for (P,Π) obtained by application of the first embedding algorithm (see chapter 4) to the pseudometric space (P',Π') will be called the main embedding algorithm.

THE MAIN EMBEDDING ALGORITHM. To construct a vector representation of a finite pseudometric space (P,Π), $P = \{p_i\}_{0 \le i \le k}$,

1) compute the matrix $M(\Psi')$ of order $(k+1)\times(k+1)$ using (6.3), where $d_{ij} = \Pi(p_i, p_j)$;
2) find the characteristic values of $M(\Psi')$ and the corresponding orthonormal characteristic vectors; form a diagonal matrix $\bar{D}_{(k+1)\times(k+1)}$

$$\bar{D} = \begin{pmatrix} d_1 & & & & & \\ & \ddots & & & & \\ & & d_{n^+} & & & \\ & & & d_{n^++1} & & \\ & & & & \ddots & \\ & & & & & d_{n^++n^-} \\ & & & & & & 0 & \\ & & & & & & & \ddots \\ & & & & & & & & 0 \end{pmatrix},$$

where d_i, $1 \le i \le n^+$, are the positive characteristic values of $M(\Psi')$, d_i, $n^+ + 1 \le i \le n^+ + n^-$, are the magnitudes of the negative characteristic values; form a matrix $L_{(k+1)\times(k+1)}$ whose columns are the coordinates of the corresponding (to d_i) orthonormal characteristic vectors;
3) compute the matrix $\tilde{T}_{(k+1)\times(k+1)}$

$$\tilde{T} = L \cdot \bar{D}^{\frac{1}{2}};$$

finally, take the first $n^+ + n^-$ elements of the ith row of $\tilde{T}$ as the coordinates of $\alpha(p_{i-1})$, $1 \le i \le k+1$, w.r.t. a Φ-orthonormal basis (e_i) of $\mathbb{R}^{(n^+,n^-)}$ for a vector representation

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)};$$

4) since the characteristic values of $M(\Psi')$ can be determined in

the order of decreased magnitude, the matrices $\bar{D}$, L and $\tilde{T}$ can be computed for any number of the characteristic values and vectors, and, therefore, the embedding algorithm can be terminated, when the magnitudes of d_i's reach a predetermined threshold:

$$|d_i| \leq \delta;$$

then, the ith row of the corresponding $\tilde{T}$ would give the coordinates of $\beta(p_{i-1})$ for a reduced vector representation

$$\beta : (P,\Pi) \longrightarrow \mathbb{R}^{(\bar{n}^+,\bar{n}^-)};$$

the resulting vector representation β *is the projection of the exact vector representation* α *on the subspace spanned by the corresponding principal axes of the covariance matrix of* (P,Π) *w.r.t.* α.

Thus, one has complete control over dimension of the vector representation space.

As far as other features in the original data are concerned, the author believes that for some specific problems a *problem oriented* reduced vector representation can also be constructed, i.e., the necessary improvements could be incorporated into the main embedding algorithm.

Once a correct representation of the data has been obtained in the vector space, many existing analytical techniques (linear and nonlinear) for the detection of the "intrinsic" (topological) dimension of the surface spanned by the data could be successfully utilized. Thus, for example, one can use now a localized principal component analysis [49] to detect the intrinsic dimension of *any* data.

The author also believes that in many problems the following *two ways of representing the data in a Euclidean space* should be of importance. Both of them are legitimate, because of the specific structure of the embedded sample (see the remark following Example 4.3). In the first method the vectors in $\mathbb{R}^{(n^+,n^-)}$ representing the finite sample are simply projected orthogonally on the Euclidean subspace $\mathbb{R}^{n^+}$:

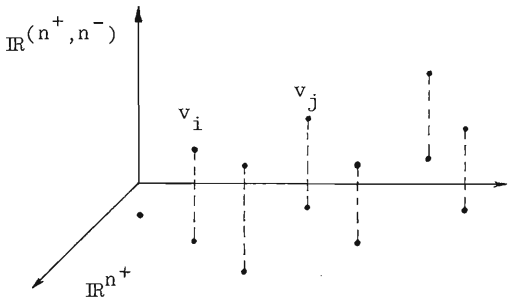


Figure 6.3

In the second method a projective mapping $\phi : \mathbb{R}^{(n^+,n^-)} \longrightarrow \mathbb{R}^{n^+}$ in Figure 6.4 is used for the projection of the vector sample on the Euclidean subspace.

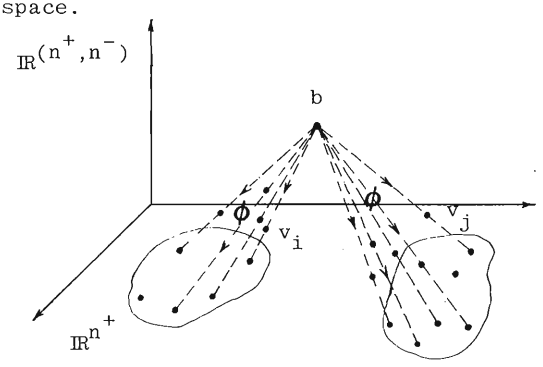


Figure 6.4

The choice of the corresponding point b (or, perhaps, several of them) should be motivated by the number and structure of the samples involved.

The rest of the chapter is devoted to *some comments about the* so-called method of non-metric multidimensional scaling, since superficially the latter may look similar to the approach suggested in this book. Strangely enough, some books on pattern recognition (see for example [6], 6.13) recommend this method, without indicating its fundamental shortcomings.

The corresponding algorithm converts *similarity* data into a set of points in a vector space. The algorithm does not preserve the actual matrix of similarities, but only their numerical ordering.

It can be briefly described as follows [50]:

1) In a space of some trial number of dimensions, a starting configuration of points x_i, $1 \le i \le k$, is first constructed, either at random or by a rational procedure, and the distances $d_{ij} = d(x_i, x_j)$ are computed.

On each iteration the following operations are performed.

2) The best-fitting w.r.t. some measure of discrepancy (stress), (for example, w.r.t.

$$S = \sqrt{\frac{\sum (d_{ij} - \hat{d}_{ij})}{\sum d_{ij}^2}} \qquad \text{(J. Kruskal))}$$

monotonic sequence $\hat{d}_{ij}$ is determined.

3) The partial derivatives

$$g_{im} = \frac{\partial S}{\partial x_{im}} \qquad \text{(gradient)}$$

are evaluated.

4) Small adjustments are made in the coordinates, x_{im}, of the points in the direction of the negative gradient or steepest descent by

$$x'_{im} = x_{im} - \alpha g_{im} \qquad \text{(adjustment)},$$

where α includes a step-size factor that is adaptively modified during the iterative process according to heuristic rules.

"The iterative process is terminated (a) when the components of the gradient have become small enough to indicate a close approach to a stationary configuration, and (b) when the residual stress is not so large as to suggest entrapment in a merely local minimum. If entrapment is suspected, another solution can be obtained using a different random or rational starting configuration. And, depending on whether the final stress seems large or small, the entire process can be tried again, in a space of one more or one less dimension, respectively. The choice among solutions of different dimensionality is then based on considerations of parsimony, goodness of fit, and especially, substantive interpretability" [50].

The idea of this heuristic algorithm was inspired by the intuitive geometric fact that configuration of a sufficiently large number of points in some Euclidean vector space remains reasonably stable under the perturbations preserving the rank order of the

interpoint distances. However, *it is not known how much of the original structure of the data is preserved by the rank order of the interpoint distances*, particularly if no information on the dimension of the representation space is available. Thus, for example, by an arbitrary small perturbation of $n+1$ vertices of the regular n-simplex in $\mathbb{R}^n$ one can obtain a set of $n+1$ points in $\mathbb{R}^n$ with any (admissible) given rank order of interpoint distances. So, while on the one hand a better fit can be achieved by increasing the dimension, on the other hand a high-dimensional representation loses the features of the original configuration.

This theoretical ambiguity results in *the following technical problems*:

a) the algorithm does not guarantee the convergence to the optimal configuration (because of the existence of local minimums);

b) the output depends on the order in which the points are processed;

c) the necessary computer time is excessive (since to get out of the local minimum one has to start anew);

d) in the end there still remains the making of the final choice from among the obtained solutions of different dimensions and configurations (of course, no one knows how many different configurations exist even in a Euclidean space of fixed dimensions).

Another fundamental objection to the multidimensional scaling comes from the fact (see Theorem 4.1) that a priori one does not even know whether the sample can be correctly represented in a Euclidean vector space, since the rank order of the intersample distances is not sufficient to make the appropriate conclusion. Moreover, even if one were to start with some configuration in a pseudoeuclidean space, the optimization of a stress function would be a much more cumbersome task (since the squared distances may be positive, zero, or negative), which results in the end in a greater variety of plausible dimensions and configurations.

The main goal of non-metric multidimensional scaling, created for the needs of psychology, was to give a "parsimonious", "interpretable", basically two-dimensional representation of the "behavioral data of arbitrary types such as confusion frequencies, or discrimination times, which were believed to vary with distance according to functions that are generally decreasing and highly nonlinear" [50]. We shall not discuss this motivation here, except to

point out an apparent contradiction (see also the first paragraph in 2.1). If *the dissimilarity measure* (distance function) *cannot be recovered or understood* from the data, what are the "considerations of parsimony, goodness of fit and, especially, substantive interpretability" on which "the choice among solutions of different dimensionality is then based on"? ([50], p. 11). And what is the connection between the original data and its Euclidean representation, the latter being always perceived as a *metric* configuration?

All of the above objections relate mainly to the use of multidimensional scaling for a metric representation of general dissimilarity data in a vector space, particularly now, when there is a very efficient algorithm to perform the task. On the other hand, incorporation of the corresponding ideas in some of non-linear dimensionality reduction techniques *applied to the samples in a vector space* might in some cases be advantageous (see [46], Ch. 10).

7 SOME CLASSICAL PROBLEMS OF PATTERN RECOGNITION

In this last chapter the solution of some typical pattern recognition problems within the proposed approach are briefly sketched. It is assumed that the reader is familiar (at least superficially) with the topics considered. Several pages of book [6] (which is often referred to in this chapter) should be enough to fill in the gaps in the reader's knowledge.

A small size of the chapter is explained by the author's belief in an almost total analogy between the analytical techniques in Euclidean and pseudoeuclidean spaces. In other words, once the geometry of the pseudoeuclidean spaces is understood, "the translation" of the classical version of an algorithm (in most cases) should not represent any *technical* problems. This is, of course, not to say that there are absolutely no problems connected with this transition, but rather that the present problems are not so much of "technical" as of "modelling" and "interpretive" nature. A more detailed treatment of the corresponding topics will be given in future articles.

7.1 Linear discriminant dimensionality reduction

Let $\{P_j\}_{1\le j\le m}$ be m finite classes in the problem of supervised classification (see [6], section 4.11). Then, having chosen an appropriate (for the problem) distance measure Π, we construct (using the embedding algorithm of chapter 6) a vector representation of (P,Π), $P = \bigcup_{j=1}^{m} P_j$:

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}, \qquad n^+ + n^- = n.$$

Now, in the pseudoeuclidean space $\mathbb{R}^{(n^+,n^-)} = (\mathbb{R}^n, \Phi)$ we can apply (if the classes are linearly separable) the *generalized* multiple discriminant method to reduce the dimension of the vector space to less than m.

Indeed, an orthogonal projection

$$\pi : \mathbb{R}^{(n^+,n^-)} \longrightarrow \mathbb{R}^{(\tilde{n}^+,\tilde{n}^-)}, \qquad \tilde{n}^+ + \tilde{n}^- = \tilde{n} \le m-1$$

can be given by $\tilde{n}$ coordinate projections

$$\pi_i(v) = \Phi(v,a_i) \qquad 1\le i\le \tilde{n},$$

where $(a_i)_{1\le i\le \tilde{n}}$ is a normalized basis of $\mathbb{R}^{(\tilde{n}^+,\tilde{n}^-)}$. So,

$$\pi_i(v) = {}^t a_i \cdot J \cdot v \qquad 1\le i\le \tilde{n} ,$$

where for simplicity the coordinate columns of vectors v and a_i are denoted by the same letters as the vectors themselves, and $J = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix}$. Combining the above $\tilde{n}$ equations we can define the projection π by a single expression

$$\pi(v) = {}^t A \cdot J \cdot v, \tag{7.1}$$

where A is an $n\times\tilde{n}$ rectangular matrix whose i^{th} column is the coordinate column of a_i.

Let us compute the within and the between scatter matrices of the projected samples $\tilde{P}_j = \pi(P_j)$ (see Def. 5.6):

$$\tilde{S}_w = \sum_{j=1}^{m} \tilde{S}_j = \sum_{j=1}^{m} \{ \sum_{v\in P_j} [\pi(v-\bar{v}_j)] \cdot {}^t[\pi(v-\bar{v}_j)] \} \cdot J =$$

$$= \sum_{j=1}^{m} \sum_{v\in P_j} [{}^t A\cdot J\cdot (v-\bar{v}_j)] \cdot {}^t[{}^t A\cdot J\cdot (v-\bar{v}_j)] \cdot J = {}^t A \cdot J \cdot S_w \cdot A \cdot J,$$

$$\tilde{S}_b = \sum_{j=1}^{m} k_j [\pi(\bar{v}_j-\bar{v})] \cdot {}^t[\pi(\bar{v}_j-\bar{v})] \cdot J = {}^t A \cdot J \cdot S_b \cdot A \cdot J.$$

As in the classical case, we take the criterion function

$$\phi(A) = \frac{\det(\tilde{S}_b)}{\det(\tilde{S}_w)} ,$$

which (taking the volume as a measure of the scatter) gives the ratio of the between-class scatter to the within-class scatter. Of course, we want to find a matrix A (that determines projection π) which maximizes the above criterion. Intuitively, this should be possible, in view of the identity $\tilde{S}_b + \tilde{S}_w = \tilde{S}$. Since

$$\phi(A) = \frac{\det({}^tA \cdot J \cdot S_b \cdot A)}{\det({}^tA \cdot J \cdot S_w \cdot A)} ,$$

the problem reduces to the classical case by introducing $B = J \cdot A$ and

$$\psi(B) = \frac{\det({}^tB \cdot S_b^0 \cdot B)}{\det({}^tB \cdot S_w^0 \cdot B)} ,$$

where S_b^0, S_w^0 are the classical between and within scatter matrices. Thus, the problem is reduced to the classical case, for which it is known ([6], p. 120) that the columns of the matrix B maximizing $\psi(B)$ are the characteristic vectors b_i corresponding to the $\tilde{n}$ maximal characteristic values d_i of the generalized eigenvalue problem

$$S_b^0 \cdot b_i = d_i \cdot S_w^0 \cdot b_i .$$

The rank of S_b^0 is less than or equal to $m-1$, so that at most $m-1$ of the above characteristic values d_i are non-zero. Taking the transpose of the found matrix B we get the matrix of *the sought projection* π (see (7.1)), *which turns out to be independent of the form* Φ.

It is important to note, that the resulting projection does not depend essentially on the vector representation α. Indeed, it is easy to show that for another representation

$$\beta : (P, \Pi) \longrightarrow \mathbb{R}^{(n^+, n^-)} , \qquad n^+ + n^- = n ,$$

we have (see the proof of Theorem 5.5)

$$\psi(B) = \frac{\det({}^tB \cdot M \cdot S_b^0 \cdot {}^tM \cdot B)}{\det({}^tB \cdot M \cdot S_w^0 \cdot {}^tM \cdot B)} ,$$

where M is the matrix of the Φ-orthogonal automorphism γ transforming α into β. Hence the corresponding generalized eigenvalue problem

$$M \cdot S_b^0 \cdot {}^tM \cdot b_i' = d_i' \cdot M \cdot S_w^0 \cdot {}^tM \cdot b_i'$$

is equivalent to

$$S_b^0 \cdot {}^tM \cdot b_i' = d_i' \cdot S_w^0 \cdot {}^tM \cdot b_i' .$$

The characteristic values d_i' are the same as above, since

$$\det(S_b^0 \cdot {}^tM - d_i' S_w^0 \cdot {}^tM) = 0 \quad \Longleftrightarrow \quad \det(S_b^0 - d_i' \cdot S_w^0) = 0;$$

and the relation between the corresponding characteristic vectors is

$$b_i = {}^tM \cdot b_i' \qquad 1 \le i \le \tilde{n}.$$

In other words, the projection subspace obtained for β is the image under the Φ-orthogonal transformation (with the matrix ${}^tM^{-1}$) of the corresponding subspace obtained for α.

The following statement seems to be true: in most cases if $\tilde{n} \le n^+$ then the space $\mathbb{R}^{(\tilde{n}^+, \tilde{n}^-)}$ *where the original vectors are projected is always a Euclidean space,* i.e., $\tilde{n}^- = 0$.

Thus, for example, if the number of the classes is not greater than 3, one can actually "work" with a two-dimensional Euclidean representation of the patterns, *independent of the complexity of the original pattern representation!* Of course, as in the classical case, the success of this method depends on the degree of the linear separability of the classes. *However, as was mentioned in chapter 1, one now has a much better control over this separability, and therefore the above method becomes of greater significance.*

7.2 Representing a new object

In this section a problem whose successful solution is important for some applications will be briefly considered. Thus, for example, applying within the proposed framework some of the decision-theoretic algorithms for the supervised classification, we are confronted with the following question: How does one represent in the vector space a new object which has to be classified? An immediate answer to this question is to embed anew every time a new object is considered. However, this solution is not acceptable for a number of practical reasons, the most important of which is the necessity of changing parameters of the classification method every time a new object is not representable in the original vector space. Fortunately, there is a solution of the problem, that is quite acceptable. The key notion in the solution is the notion of the orthogonal projection defined in Section 3.3.

A general idea of the solution was mentioned at the end of chapter 4. Now, we shall go into the details of the computations.

The main difficulty is related to the fact that the dimension of the subspace spanned by the vectors $\beta(P)$, where P is a training sample, under the new vector representation

$$\beta : (P \cup \{p\}, \Pi_*) \longrightarrow \mathbb{R}^{(n_*^+, n_*^-)}$$

(p is the "new" element) may be in *some cases* greater than that of $\mathbb{R}^{(n^+, n^-)}$ in the original representation

$$\alpha : (P, \Pi) \longrightarrow \mathbb{R}^{(n^+, n^-)} \qquad n^+ \leq n_*^+, n^- \leq n_*^-$$

(see the remark at the end of Example 4.1). This, of course, cannot happen in the "Euclidean case". Again, one has to remember, that *the above fact is a reflection of the real situation*, and we can observe it only because the pseudoeuclidean spaces form a more general class of spaces than that of the Euclidean spaces.

As was mentioned above, the orthogonal projection seems to be the most natural tool to represent the new element p in the "basic" pseudoeuclidean space $\mathbb{R}^{(n^+, n^-)}$, obtained by means of the vector representation α. In fact, all the necessary formulas are given in section 3.3. Thus, having chosen a basis $v_{i_1}, \ldots, v_{i_n}$ of $\mathbb{R}^{(n^+, n^-)}$ among the elements $v_i = \alpha(p_i)$, $\{p_i\}_{0 \leq i \leq k} = P$, we can find the squared distance of the vector $\beta(P)$ from the subspace generated by the vectors $\beta(p_{i_1}), \beta(p_{i_2}), \ldots, \beta(p_{i_n})$ in $\mathbb{R}^{(n_*^+, n_*^-)}$, without constructing the vector representation β (assuming that the subspace is non-isotropic). To this end we shall assume, without loss of generality, that

$$\beta(p_0) = \alpha(p_0) = 0.$$

Let $\mathbb{R}^{(n_*^+, n_*^-)} = (\mathbb{R}^{n*}, \Phi_*)$, $\beta(p) = v$, and U be the subspace of $\mathbb{R}^{(n_*^+, n_*^-)}$ generated by the vectors $\beta(p_{i_1}) = a_1$, $\beta(p_{i_2}) = a_2, \ldots, \beta(p_{i_n}) = a_n$. Then, by (3.9) of section 3.3

$$d^2(v, U) = \Phi_*(v, v) - {}^t b \cdot [G_*(a_1, a_2, \ldots, a_n)]^{-1} \cdot b,$$

where G_* is the Gram matrix of $a_1, \ldots, a_n$ in $\mathbb{R}^{(n_*^+, n_*^-)}$, and ${}^t b = (\Phi_*(v, a_1), \Phi_*(v, a_2), \ldots, \Phi_*(v, a_n))$.

On the other hand, we find

$$\Phi_*(v,v) = \Phi_*(v-0,v-0) = \Pi^2_*(p_0,p),$$

$$\Phi_*(a_j,a_\ell) = \frac{1}{2}\,[\Phi_*(a_j,a_j) + \Phi_*(a_\ell,a_\ell) - \Phi_*(a_j-a_\ell,a_j-a_\ell)] =$$

$$= \frac{1}{2}\,[\Pi^2_*(p_0,p_{i_j}) + \Pi^2_*(p_0,p_{i_\ell}) - \Pi^2_*(p_{i_j},p_{i_\ell})] =$$

$$= \frac{1}{2}\,[\Pi^2(p_0,p_{i_j}) + \Pi^2(p_0,p_{i_\ell}) - \Pi^2(p_{i_j},p_{i_\ell})] = \Phi(a_j,a_\ell)$$

$$(7.2)\qquad b^j = \Phi_*(v,a_j) = \frac{1}{2}\,[\Phi_*(v,v) + \Phi_*(a_j,a_j) - \Phi_*(v-a_j,v-a_j)] =$$

$$= \frac{1}{2}\,[\Pi^2_*(p,p_0) + \Pi^2(p_{i_j},p_0) - \Pi^2_*(p,p_{i_j})].$$

Hence, *we can determine* $d^2(v,U)$ *without constructing the vector representation* β:

$$(7.3)\qquad d^2(p,U) \overset{\text{def}}{=} d^2(v,U) = \Pi^2_*(p_0,p) - {}^t b\cdot[G(a_1,\ldots,a_n)]^{-1}\cdot b,$$

where $G(a_1,\ldots,a_n)$ is the Gram matrix of $a_1,\ldots,a_n$ in $\mathbb{R}^{(n^+,n^-)}$, and b is the coordinate vector, components of which are computed by formula (7.2).

One is interested in choosing the basis (a_i) in $\mathbb{R}^{(n^+,n^-)}$ that minimizes $|d^2(p,U)|$, since then the distances between a_i, $1\le i\le n$, and the projection of p on U are close to those between a_i and p. Although, generally, the optimal basis (a_i) depends on p, *for each class of problems* one can give some simple guidelines for choosing an appropriate basis. Again, if the collected data is "sufficient" in the sense that it is ε-net (ε is small) for the corresponding population, these guidelines are not that essential.

For the orthogonal projection of $v = \beta(p)$ *on the non-isotropic subspace* U, $\pi_U(v)$, we can similarly find by (3.8) that

$$(7.4)\qquad u = A_0\cdot[G(a_1,\ldots,a_n)]^{-1}\cdot b,$$

where u is the vector of the coordinates of $\pi_U(v)$, A_0 is the matrix whose columns are the coordinate columns of a_i (w.r.t. the fixed orthonormal basis of $\mathbb{R}^{(n^+,n^-)}$), and b is the same as above.

Thus, since $[G(a_1,\ldots,a_n)]^{-1}$ and $A_0\cdot[G(a_1,\ldots,a_n)]^{-1}$ are computed once for all the new points p, the only computations that have to be done are those to find $b = (b^1,b^2,\ldots,b^n)$ by (7.2) and several simple computations in (7.3) and (7.4), i.e., it is

necessary to perform only $O(n^2)$ *on line* algebraic operations to find the distance $d^2(p,U)$ and the projection $\pi_U(v)$, where n is the dimension of the representation space. This is perfectly feasible, particularly considering the fact that one has control over the dimension of the representation space (see chapter 6).

It is important to note that the average magnitude of $d^2(p,U)$ for different p is, in general, a good measure of the reliability of the data collected at the main stage: a relatively small value indicates that the data collected sufficiently well represents the population involved.

It is also important to note that the reliability of the projection method depends on the accuracy ε with which the training sample represents the population (see above).

Finally, we note that if the above subspace U generated by the vectors a_i, $1 \le i \le n$, in $\mathbb{R}^{(n_*^+, n_*^-)}$ is isotropic, then we can still use the above formulas (7.3) and (7.4), only in this case (7.4) gives the projection of p on U assuming that U is non-isotropic.

7.3 Decision surfaces

As was mentioned in the Introduction, the possibility of constructing for the relevant problems decision surfaces for any data (for example, syntactic data of Example 2.5) is an extremely important advantage offered by the approach. Moreover, *the decision surface in the proposed approach is a more reliable tool* (than in the classical approach), since the objects are represented now not in an arbitrary absolute space, but in the space determined by the interobject distances, and if the distance function is appropriate (from the discriminant point of view), the existence of relatively simple decision surfaces becomes unquestionable.

The literature on discriminant functions constitutes such a large part of the entire literature on pattern recognition, that there is no need to repeat here the well know ideas (see, for example, [51] or [6], chapter 5). However, several remarks clarifying some technical differences between the classical and the pseudo-euclidean linear decision surfaces should be useful.

First of all, one has to remember that in a pseudoeuclidean space $\mathbb{R}^{(n^+,n^-)} = (\mathbb{R}^n, \Phi)$ if vector u belongs to a hyperplane U, and $v \neq 0$ is a "normal vector" of U ($v \perp U$), then, of course, for every $x \in U \Longrightarrow v \perp (x-u)$, i.e.,

$$\forall x \in U \qquad \Phi(v, x-u) = 0.$$

Thus, the coordinate form the equation of the plane becomes (see (3.3)):

$$(x^1-u^1, \ldots, x^n-u^n) \cdot \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix} \cdot \begin{pmatrix} v_1 \\ \vdots \\ v^n \end{pmatrix} = 0,$$

or

$$v^1(x^1-u^1) + \ldots + v^{\bar{n}}(x^{\bar{n}}-u^{\bar{n}}) - v^{\bar{n}+1}(x^{\bar{n}+1}-u^{\bar{n}+1}) - \ldots - v^n(x^n-u^n) = 0,$$

where $\bar{n} = n^+$. So, the general form of the equation is, of course, the same as in the Euclidean case, except for minus signs in front of the last n^- terms.

Let

$$v^1x^1 + \ldots + v^{\bar{n}}x^{\bar{n}} - v^{\bar{n}+1}x^{\bar{n}+1} - \ldots - v^nx^n + c = 0 \tag{7.5}$$

be an equation of the separating hyperplane U ($\bar{n} = n^+$), let $a \in \mathbb{R}^{(n^+,n^-)}$, and let the orthogonal projection of a on U be u. Denoting the coordinates of a and u as a^i and u^i, we have

$$v^1u^1 + \ldots + v^{\bar{n}}u^{\bar{n}} - v^{\bar{n}+1}u^{\bar{n}+1} - \ldots - v^nu^n + c = 0;$$

so that

$$v^1a^1 + \ldots + v^{\bar{n}}a^{\bar{n}} - v^{\bar{n}+1}a^{\bar{n}+1} - \ldots - v^na^n + c =$$

$$= v^1(a^1-u^1) + \ldots + v^{\bar{n}}(a^{\bar{n}}-u^{\bar{n}}) - v^{\bar{n}+1}(a^{\bar{n}+1}-u^{\bar{n}+1}) - \ldots - v^n(a^n-u^n) =$$

$$= \Phi(v, a-u).$$

But $v \perp U$ and $(a-u) \perp U$, hence $a-u = d \cdot v$, where d is a positive real number if vectors v, a-u are in the same half-space w.r.t. U, and negative, if v, a-u are in the different half-spaces. Substituting $a-u = d \cdot v$ in the last formula we obtain (see Def. 3.2)

$$v^1a^1 + \ldots + v^{\bar{n}}a^{\bar{n}} - v^{\bar{n}+1}a^{\bar{n}+1} - \ldots - v^na^n + c = d\|v\|^2.$$

Setting $\|v\|^2 = 1$, we obtain that $\|a-u\|^2 = \Phi(a-u, a-u) = \Phi(d \cdot v, d \cdot v) = d^2$. The last identity says, in fact, that the expression

$$v^1a^1 + \ldots + v^{\bar{n}}a^{\bar{n}} - \ldots - v^na^n + c,$$

where $\|v\|^2 = 1$, is positive in the one half-space, negative in the

other half-space, and its magnitude is equal to the magnitude of the distance from a to U (see Def. 3.12). In other words, if the normal vector v is of a positive norm, all the classical techniques are applicable. But the configuration of a finite pseudometric space embedded into a pseudoeuclidean space always has one specific property (see the remark following Example 4.3). This property ensures that the vector v normal to the plane U orthogonal to a segment connecting any two vectors representing the training sample, satisfy the above condition ($\|v\| \geq 0$), simply because v is parallel to the latter segment (Fig. 7.1).

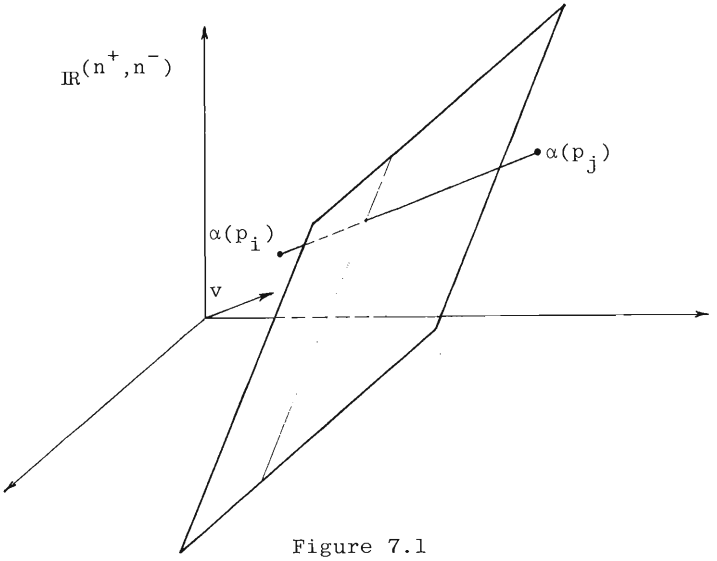


Figure 7.1

Of course, more general (than linear) surfaces, *quadratic decision surfaces*, can also be used now as decision surfaces. In particular, *very popular Mahalanobis distance* (see chapter 5) *becomes available for the supervised classification of syntactic patterns*.

It is also very important to note that almost all the algorithms developed within the field of computational geometry (see, for example, [24]) through the vector representation can now be adopted to any structural patterns, which results in crucial improvements in the efficiency of the relevant software.

7.4 Clustering

Three basic reasons (besides the traditional one which comes from taxonomy) for interest in clustering are given in [6], p. 189. The advantages of a vector representation of the data in clustering are well known: the choice of classification criteria is considerably wider in a vector space than in a more abstract space. In fact, the overwhelming majority of papers on clustering assume that the data are represented in a Euclidean vector space.

If the classes have more or less elliptical shapes and are pairwise disjoint, then good clustering criteria can be obtained by constructing some functions of the characteristic values of the within and between scatter matrices (see Def. 5.6). It is an important merit of the proposed approach that such criteria become available for the unsupervised classification of "structural" data or, for that matter, of any data.

So, let

$$\alpha : (P,\Pi) \longrightarrow \mathbb{R}^{(n^+,n^-)}$$

be a vector representation of a finite pseudometric space (P,Π), where (P,Π) represents the given (observed) data. Then, obviously, the scatter matrix S of (P,Π) with respect to α (Def. 5.2) does not depend on how the set P is partitioned into classes P_j, $\bigcup_{j=1}^{m} P_j = P$. On the other hand, the within P_j and between P_j scatter matrices do depend on the partition. Intuitively, one would like to obtain such a partition of P which minimizes the within P_j scatter and maximizes the between P_j scatter. As in the classical case, Theorem 5.8 implies that these two objectives are not contradictory; moreover, the achievement of one ensures the achievement of the other. And, of course, different measures of the intuitive notion of scatter give rise to distinct formal *optimal criteria.*

Thus, taking the sum of squares of the distances (in $\mathbb{R}^{(n^+n^-)}$) from the points in each P_j to the corresponding mean $\bar{v}_j$ as a measure of scatter, we are, according to Theorem 5.1, choosing $\mathrm{tr}(S_w)$ as optimally criterion. And since

$$\mathrm{tr}(S_b) = c - \mathrm{tr}(S_w),$$

where $c = \mathrm{tr}(S)$, minimizing $\mathrm{tr}(S_w)$ we are at the same time maximizing $\mathrm{tr}(S_b) = \sum_{j=1}^{m} k_j \|\bar{v}_j - \bar{v}\|$, which is a desirable feature.

Taking $|\det(S_w)|$ as the optimality criterion, we are, in fact, choosing the square of the volume as a measure of scatter. In the proposed approach the last criterion is more reliable than in the classical approach, since the covariance matrix S is always nonsingular.

As in the classical case, we now also have available for *any data* the clustering criteria obtained by constructing appropriate functions of the characteristic values of a non-negative matrix $S_w^{-1} \cdot S_b$. These characteristic values give the ratio of the between P_j to the within P_j scatter in the directions of the corresponding characteristic vectors.

Since all of the above criteria are invariant under the motions of $\mathbb{R}^{(n^+,n^-)}$, they are legitimate criteria for the finite pseudometric space (P,Π) (see the *fundamental point* made after Theorem 4.4).

It was point out in [52] that $\det(S_w)$ (equivalently $\frac{\det(S_w)}{\det(S)}$) is somewhat more sensitive to the local structure of data than some of the other criteria, but other criteria may shed some additional light on the configuration of the data.

If the metric interrelationship between the classes is more complex (than mentioned at the beginning of the section), the above criteria may not be appropriate. However, in any case, the presence of the vector space structure allows one to utilize for clustering a greater variety of analytical criteria.

7.5 Mixed-mode data

Mixed-mode data is data for representation of which several principally different abstract structures have been used. In other words, the representation set P is a Cartesian product $\underset{i=1}{\overset{r}{X}} P_i$, where some of the pairs P_i, P_j $(1 \le i,j \le r)$ are in some sense incompatible. The simplest (and typical) case of such data occurs when set P_1 is used to represent structural (or qualitative) characteristics of the data, and set P_2 is used to represent quantitative characteristics. This case is illustrated by Example 2.3, to which the following remark should be added. The pseudometric Ψ defined there is by no means always a satisfactory measure of dissimilarity. It was introduced to illustrate the point that sometimes one can *directly* define an appropriate distance function which would reflect the dissimilarity between both the

qualitative and the quantitative features of the data representation. Quite often, however, no apparent connections between these groups of features is known and it is this case that will be discussed in this section.

Two somewhat similar approaches can be suggested. In the first approach one constructs vector representations of each finite pseudometric space (P_i,Π_i) separately, and only then puts them together. Namely, let $(P,\Pi) \subseteq (P_1,\Pi_2) \times (P_2,\Pi_2)$, where (P_1,Π_1) and (P_2,Π_2) are finite pseudometric spaces[†]. Then we can construct two corresponding vector representations

$$\alpha_1 : (P_1,\Pi_1) \longrightarrow \mathbb{R}^{(n_1^+,n_1^-)} \qquad n_1^+ + n_1^- = n_1,$$
$$\alpha_2 : (P_2,\Pi_2) \longrightarrow \mathbb{R}^{(n_2^+,n_2^-)} \qquad n_2^+ + n_2^- = n_2.$$

Next, the analysis of each of the obtained vector pattern representations is performed. For some problems (such as supervised classification) this analysis may reveal that one can do much better by limiting the number of representation sets involved (for example, the decision surfaces might be simpler for a smaller number of representation sets). Removing the superfluous factors in the Cartesian product and performing optional transformations for each vector representation, we can construct a new mapping

$$\bar{\alpha} = \bar{\alpha}_1 \times \bar{\alpha}_2 : P_1 \times P_2 \longrightarrow \mathbb{R}^{(\bar{n}_1^+ + \bar{n}_2^+, \bar{n}_1^- + \bar{n}_2^-)}$$

where $\bar{\alpha}_1 : P_1 \longrightarrow \mathbb{R}^{(\bar{n}_1^+,\bar{n}_1^-)}$ and $\bar{\alpha}_2 : P_2 \longrightarrow \mathbb{R}^{(\bar{n}_2^+,\bar{n}_2^-)}$ are appropriately transformed (at the previous stage, if at all) vector representations α_1, α_2, and

$$\bar{\alpha}(p_1,p_2) = (\bar{\alpha}_1(p_1),\bar{\alpha}_2(p_2))$$

with necessary rearrangement of the coordinates in the right-hand side (i.e., those which enter with $+$ in one of the forms come first). The dimension of the vector space $\mathbb{R}^{(\bar{n}_1^+ + \bar{n}_2^+, \bar{n}_1^- + \bar{n}_2^-)}$ is not necessarily minimal, since it was obtained as the $\oplus$-direct product

[†]For simplicity we shall confine ourselves to the case of two "incompatible" pseudometric spaces.

of two "minimal" pseudoeuclidean spaces (see Def. 3.9). Taking the restriction $\bar{\alpha}|P$ (P is usually smaller than $P_1 \times P_2$) and applying, if desired, the main embedding algorithm (see chapter 6) one can obtain a reduced vector representation of P. Schematically the first approach is represented in Fig. 7.2.

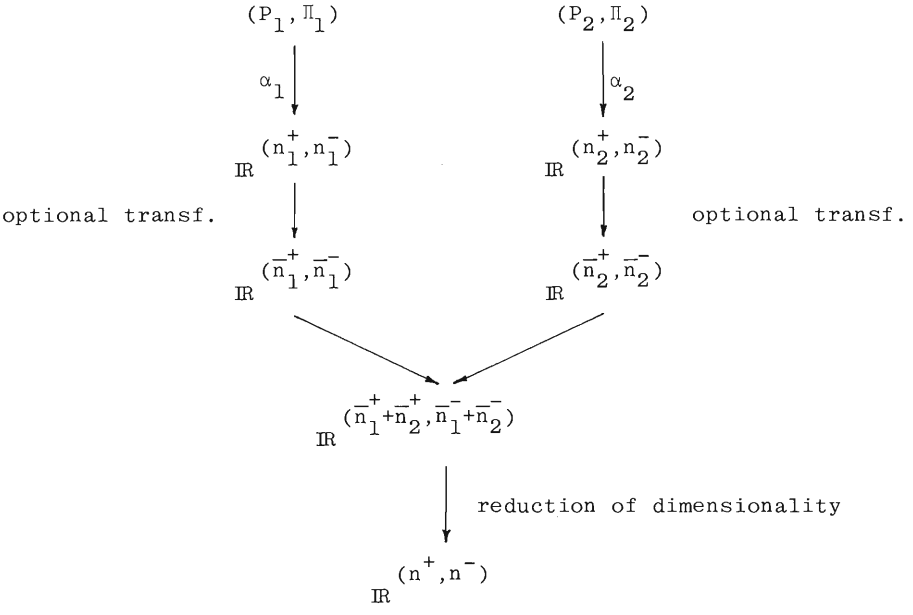


Figure 7.2

In the second approach one first constructs a weighted Cartesian product of the pseudometric spaces involved (see chapter 2), and then constructs the vector representation. Namely, let $(\bar{P}, \bar{\Pi})$ be a weighted Cartesian product of (P_1, Π_1) and (P_2, Π_2), so that the given (observed) data is considered now as a finite subspace (P, Π) of $(\bar{P}, \bar{\Pi})$. Then we can construct a vector representation

$$\alpha : (P, \Pi) \longrightarrow \mathbb{R}^{(n^+, n^-)},$$

and perform the consequent analysis in $\mathbb{R}^{(n^+, n^-)}$. Schematically the second approach is represented in Fig. 7.3.

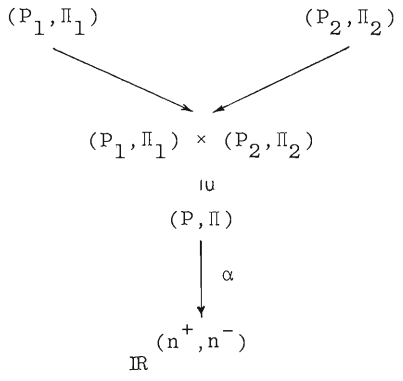


Figure 7.3

If in the first approach all originally chosen representation sets P_i are retained and no transformations are performed on the vector representations α_i, then the results of both approaches are identical (up to a choice of the weights). Thus, although the second approach seems to be shorter, in many cases the first approach is preferable (this is true, for example, when the number r of the initial representation sets is large, or when the dimension of the "direct" vector representation is large).

Having transformed the original mixed-mode problem into one in the pseudoeuclidean vector space, one now has much more powerful analytical tools to tackle the problem. Thus, for example, a new dissimilarity measure, Mahalanobis distance function, can now be introduced. This measure, then, incorporates *interdependence between different, originally incompatible features.*

Appendix I

Quadratic forms

Let V be a vector space over $\mathbb{R}$ of dimension n, and let $(a_i)_{1\leq i\leq n}$ be a basis of V. A quadratic form on V is by definition a mapping $\phi : V \longrightarrow \mathbb{R}$ given by an expression of the form

$$\phi(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x^i x^j,$$

where $c_{ij} \in \mathbb{R}$, x^i are the coordinates of x with respect to the basis (a_i), and $c_{ij} = c_{ji}$.

It follows from (3.2) that if Φ is a symmetric bilinear form on V, then the mapping $\phi : V \longrightarrow \mathbb{R}$ defined as

$$\phi(x) = \Phi(x,x)$$

is a quadratic form on V, called the quadratic form associated with Φ.

On the other hand, if ϕ is a quadratic form on V, then the mapping $\Phi : V \times V \longrightarrow \mathbb{R}$ defined as

$$\Phi(x,y) = \frac{\phi(x+y) - \phi(x-y)}{4}$$

is a symmetric bilinear form on V. Indeed,

$$\frac{1}{4}[\phi(x+y) - \phi(x-y)] = \frac{1}{4} \sum_{i,j} c_{ij}[(x^i+y^i)(x^j+y^j) - (x^i-y^i)(x^j-y^j)] =$$

$$= \frac{1}{4} \sum_{i,j} c_{ij}(2y^i x^j + 2x^i y^j) = \sum_{i,j} c_{ij} x^i y^j,$$

and the statement follows from (3.2). Obviously, if ϕ is the quadratic form associated with Φ, then the form Φ can be reconstructured from ϕ, by means of the above formula. The above shows that the notion of a symmetric bilinear form is in a sense equivalent to that of a quadratic form.

Appendix II

Duality for a vector space with a non-degenerate symmetric bilinear form

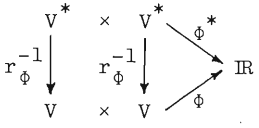
Let V be a finite-dimensional vector space over $\mathbb{R}$, Φ be a non-degenerate symmetric bilinear form on it, and V^* be the dual of V. The homomorphisms ℓ_Φ and r_Φ ($\ell_\Phi = r_\Phi$) were defined in Def. 3.6:

$$\ell_\Phi = r_\Phi : V \longrightarrow V^*.$$

By Theorem 3.3 it is an isomorphism. With the aid of this isomorphism we can define a non-degenerate symmetric bilinear form Φ^* on V^* as follows:

$$\Phi^*(x^*, y^*) \stackrel{\text{def}}{=} \Phi(r_\Phi^{-1}(x^*),\ r_\Phi^{-1}(y^*)),$$

which can be diagrammatically represented as



The form Φ^* is called the dual of the form Φ.

Theorem. Let $M(\Phi) = (f_{ij})_{1\le i,j\le n}$ be the matrix of Φ with respect to a basis $(a_i)_{1\le i\le n}$ of V, and let $M(\Phi^*) = (f^*_{ij})_{1\le i,j\le n}$ be the matrix of Φ^* with respect to the dual basis $(a^*_i)_{1\le i\le n}$ of V^*. Then

$$M(\Phi^*) = [M(\Phi)]^{-1}.$$

Proof. Suppose

$$r_\Phi(a_i) = \sum_{j=1}^{n} c_{ij} a^*_j.$$

Since both sides of the above equality are linear forms on V, we can equate their values at the vector $a_k \in V$. By Def. 3.6 we obtain for the left-hand side

$$[r_\Phi(a_i)](a_k) = \Phi(a_k,a_i) = f_{ki} = f_{ik},$$

and for the right-hand side we obtain

$$[\sum_{j=1}^{n} c_{ij}a_j^*](a_k) = \sum_{j=1}^{n} c_{ij}a_j^*(a_k) = c_{ik}.$$

Thus,

$$c_{ik} = f_{ik} \qquad (1\le i,k\le n),$$

i.e.,

$$M(r_\Phi) = M(\Phi),$$

where $M(r_\Phi)$ is the matrix of the isomorphism r_Φ with respect to the bases (a_i) and (a_i^*).

Next, suppose that

$$r_\Phi^{-1}(a_i^*) = \sum_{j=1}^{n} d_{ij}a_j.$$

Then,

$$f_{ik}^* = \Phi^*(a_i^*,a_k^*) = \Phi(r_\Phi^{-1}(a_i^*),r_\Phi^{-1}(a_k^*)) = \Phi(\sum_{j=1}^{n} d_{ij}a_j,r_\Phi^{-1}(a_k^*)) =$$

$$= \sum_{j=1}^{n} d_{ij}\Phi(a_j,r_\Phi^{-1}(a_k^*)) = d_{ik},$$

since if we set $r_\Phi^{-1}(a_k^*) = b$, then $r_\Phi(b) = \Phi_b = a_k^*$, so that

$$\Phi(a_j,r_\Phi^{-1}(a_k^*)) = \Phi(a_j,b) = a_k^*(a_j).$$

Thus,

$$M(\Phi^*) = M(r_\Phi^{-1}),$$

where $M(r_\Phi^{-1})$ is the matrix of the isomorphism r_Φ^{-1} with respect to bases (a_i) and (a_i^*). Since $M(r_\Phi^{-1}) = [M(r_\Phi)]^{-1}$, the theorem is proved. ∎

Appendix III

Random variables with values in a pseudoeuclidean space

If (P,B,μ) is a probability space, then, as always, a Borel mapping $\alpha : P \longrightarrow \mathbb{R}^n$ is called a random vector, or random variable with values in $\mathbb{R}^n$. By $\mathbb{R}^n$ we mean here the vector space with the ordinary topology. The above notion of a random vector *does not depend on any inner product* in $\mathbb{R}^n$, since the topology of a finite-dimensional vector space is determined uniquely by its vector structure (see, for example, [54], p. 199). Let us assume now that on the set P a distance function Π is defined, and B is the Borel σ-algebra generated by the open sets of (P,Π). It is well known, that the only purpose for introducing a random vector α is to transform the original problem in (P,B,μ) into the problem in another, analytically more convenient space $\mathbb{R}^n$. However, since with the transition one wants *to preserve* now *as much as possible the metric features of* (P,Π), the possibility of using different (of different signatures) symmetric bilinear forms on $\mathbb{R}^n$ to measure the distances becomes of great importance. The last fact to the author's knowledge, has not been realized in probability and statistical theories.

In this appendix some of the basic notions for random vectors in $\mathbb{R}^{(n^+,n^-)}$ are introduced. We follow the plan of [53],[†] which is highly recommended for the development of the multivariate theory. Although almost all notions and results are direct generalizations of those in [53], some readers may find it helpful to see first how the notions introduced below look in the case of a vector space with a positive inner product ([53], Ch. 2,3,7).

So let

$$\alpha : ((P,\Pi),B,\mu) \longrightarrow \mathbb{R}^{(n^+,n^-)} = (V,\Phi)$$

be a random vector in a pseudoeuclidean space $\mathbb{R}^{(n^+,n^-)}$. Then it is easy to see, that every vector $v \in V$ generates a real-valued random variable α_v defined by means of the projection of $\alpha(p)$ onto a vector line determined by v:

$$\forall p \in P \qquad \alpha_v(p) = \Phi(\alpha(p),v), \text{ i.e., } \alpha_v = r_\Phi(v) \circ \alpha$$

[†] There is a new edition [59].

(see Def. 3.6). If $\forall v \in V$ the expectation $E(\alpha_v)$ exists, then it is also easy to see that $\varepsilon(v) = E(\alpha_v)$ is a linear form on V. Indeed,

$$\varepsilon(c_1v_1+c_2v_2) = E(\alpha_{c_1v_1+c_2v_2}) = E[r_\Phi(c_1v_1+c_2v_2)\circ\alpha] =$$

$$= E\{[c_1r_\Phi(v_1)+c_2r_\Phi(v_2)]\circ\alpha\} = E[c_1r_\Phi(v_1)\circ\alpha+c_2r_\Phi(v_2)\circ\alpha] =$$

$$= c_1\varepsilon(v_1)+c_2\varepsilon(v_2).$$

By Theorem 3.3 there exists a unique $\bar{v}_\alpha = \bar{v} \in V$ such that

$$\varepsilon(v) = \Phi(v,\bar{v}) \qquad \forall v \in V.$$

Vector $\bar{v}$ is called <u>the mean vector of the random vector α</u>, and is denoted also $E(\alpha)$, since it does not depend on v. Obviously, if $\alpha'(x) = \alpha(x)+v_0$, then $E(\alpha') = E(\alpha)+v_0$.

<u>Theorem III.1</u>. Let $\gamma : (V,\Phi) \longrightarrow (W,\Psi)$ be a homomorphism of pseudo-euclidean spaces, then

$$E(\gamma\circ\alpha) = \gamma[E(\alpha)].$$

<u>Proof</u>. Denoting by γ^* the adjoint of γ w.r.t. the bilinear forms Φ, Ψ (see [33], §36.3), setting $E(\gamma\circ\alpha) = \bar{w}$, and $E(\alpha) = \bar{v}$, we have $\forall w \in W$

$$\Psi(w,\bar{w}) = E[(\gamma\circ\alpha)_w] = E[\Psi(\gamma\circ\alpha(p),w)] = E[\Phi(\alpha(p),\gamma^*(w))] =$$

$$= E[\alpha_{\gamma^*(w)}] = \Phi(\gamma^*(w),\bar{v}) = \Psi(w,\gamma(\bar{v})).$$

Thus, $\Psi(w,\bar{w}) = \Psi(w,\gamma(\bar{v}))$ $\forall w \in W$, and since Ψ is non-degenerate $\bar{w} = \gamma(\bar{v})$. ∎

As an important corollary we obtain the following result: the mean vector of α does not depend on the form Φ of V. Indeed, it is enough to set in the above lemma $W = V$ and $\gamma = \iota$ (identity mapping). This result is in a complete agreement with the fact that *the mean of a finite sample does not depend on the bilinear form* (see Def. 5.1).

Assume next, that for the random vector α and for $\forall v_1, v_2 \in V$ the expectations $E(\alpha_{v_1} \cdot \alpha_{v_2})$ and $E(\alpha_{v_1})$ exist. Then, we can

define on the vector space V a non-negative symmetric bilinear form Σ_α as follows

$$\Sigma_\alpha(v_1,v_2) = \mathrm{Cov}(\alpha_{v_1},\alpha_{v_2}) \qquad \forall v_1,v_2\in V,$$

where Cov is the usual covariance of two random variables. Indeed, the bilinearity follows from the bilinearity of Cov

$$\Sigma_\alpha(c_1v_1+c_2v_2,v) = \mathrm{Cov}(\alpha_{c_1v_1+c_2v_2},\alpha_v) = \mathrm{Cov}(c_1\alpha_{v_1}+c_2\alpha_{v_2}) =$$

$$= c_1\,\mathrm{Cov}(\alpha_{v_1},\alpha_v) + c_2\,\mathrm{Cov}(\alpha_{v_2},\alpha_v) = c_1\Sigma_\alpha(v_1,v) + c_2\Sigma_\alpha(v_2,v);$$

and the non-negativity also follows from the non-negativity of Cov:

$$\Sigma_\alpha(v,v) = \mathrm{Var}(\alpha_v) \geq 0.$$

We shall call the form Σ_α the covariance bilinear form of the random vector α.

Next, define an endomorphism $\sigma_\alpha : V \longrightarrow V$ as follows. If $(e_i)_{1\leq i\leq n}$ is a Φ-orthonormal basis of V, then set the matrix of σ_α w.r.t. (e_i) equal to $J\cdot M(\Sigma_\alpha)$, where J and $M(\Sigma_\alpha)$ are the matrices of Φ and Σ_α w.r.t. (e_i) respectively. By Theorem 3.23 σ_α is a Φ-non-negative endomorphism, and by the definition of σ_α

$$\Sigma_\alpha(u,v) = \Phi(\sigma_\alpha(u),v) \qquad \forall u,v\in V. \tag{III.1}$$

The uniqueness of σ_α can be proved as follows:

$$\Phi(\sigma^1_\alpha(u),v) = \Phi(\sigma^2_\alpha(u),v) \qquad \forall u,v \iff$$

$$\Phi((\sigma^1_\alpha-\sigma^2_\alpha)u,v) = 0 \qquad \forall u,v \iff$$

(since Φ is non-degenerate) $(\sigma^1_\alpha-\sigma^2_\alpha)u = 0 \quad \forall u$, or $\sigma^1_\alpha = \sigma^2_\alpha$. We shall call σ_α the covariance endomorphism of the random vector α. We shall see later that σ_α, *and therefore* Σ_α, *depend on the bilinear form* Φ, i.e., in general $\sigma^{\Phi_1}_\alpha \neq \sigma^{\Phi_2}_\alpha$, if $\Phi_1 \neq \Phi_2$. One can easily show that if $\alpha'(p) = \alpha(p) + v_0$, then $\sigma_{\alpha'} = \sigma_\alpha$.

Lemma III.2. Suppose that for a Φ-self-adjoint endomorphism $\gamma : V \longrightarrow V$ $\mathrm{Var}(\alpha_v) = \Phi(\gamma(v),v) \quad \forall v\in V$. Then $\gamma = \sigma_\alpha$.

Proof. By (III.1)

$$\Phi(\sigma_\alpha(v),v) = \Phi(\gamma(v),v) \qquad \forall v\in V$$

or

$$\Phi((\sigma_\alpha-\gamma)v,v) = 0 \qquad \forall v\in V.$$

Set $\sigma_\alpha-\gamma = \theta$, then since θ is Φ-self-adjoint, $\forall x,y\in V$

$$\Phi(\theta(x),y) = \frac{1}{2}[\Phi(\theta(x+y),x+y) - \Phi(\theta(x),x) - \Phi(\theta(y),y)] = 0;$$

and since Φ is non-degenerate $\theta(x) = 0 \qquad \forall x\in V.$ ∎

Theorem III.3. Let $\alpha : P \longrightarrow (V,\Phi)$ be a random vector, $\gamma : (V,\Phi) \longrightarrow (W,\Psi)$ be a homomorphism of pseudoeuclidean spaces, and the random vector $\beta : P \longrightarrow (W,\Psi)$ be defined as $\beta(p) = \gamma\circ\alpha(p) + w_0$. Then

$$\sigma_\beta = \gamma\circ\sigma_\alpha\circ\gamma^*,$$

where $\gamma^* : W \longrightarrow V$ is the adjoint of γ w.r.t. bilinear forms Φ,Ψ.

Proof. $\mathrm{Var}[\beta_w] = \mathrm{Var}[\Psi(\gamma\circ\alpha(p) + w_0,w)] = \mathrm{Var}[\Psi(\gamma\circ\alpha(p),w)] =$

$= \mathrm{Var}[\Phi(\alpha(p),\gamma^*(w))] = \mathrm{Var}[\alpha_{\gamma^*(w)}] = \Phi(\sigma_\alpha(\gamma^*(w)),\gamma^*(w)) =$

$= \Psi(\gamma\circ\sigma_\alpha\circ\gamma^*(w),w),$

and the theorem follows from Lemma III.2. ∎

Corollary. If in the above theorem Φ is a Euclidean inner product, $W = V$, and $\gamma = \iota$ (identity endomorphism), then $\sigma_\beta = \sigma_\alpha\circ\iota^*$.

One can check immediately that Def. 5.2 is in complete agreement with the above corollary.

Let a random vector $\alpha : P \longrightarrow (V,\Phi)$ satisfies: $\forall v_1,v_2\in V$ $\mathrm{Cov}(\alpha_{v_1},\alpha_{v_2})$ exists. Then we say that the distribution of α is weakly Φ-spherical if the following equality of the covariance forms holds

$$\Sigma_\alpha = \Sigma_{\gamma\circ\alpha} \qquad \forall\gamma\in O(V,\Phi)$$

(see Def. 3.15). Since the equality of the covariance forms implies the equality of the covariance endomorphisms, the distribution of α is weakly Φ-spherical if and only if

$$\sigma_\alpha = \gamma\circ\sigma_\alpha\circ\gamma^* \qquad \forall\gamma\in O(V,\Phi) \tag{III.2}$$

(see Theorem III.3).

Theorem III.4. The distribution of α is weakly Φ-spherical if and only if $\sigma_\alpha = \begin{pmatrix} c_\alpha \imath_{V_+} & 0 \\ 0 & -d_\alpha \imath_{V_-} \end{pmatrix}$, where $c_\alpha, d_\alpha \in \mathbb{R}_+$, and $\imath_{V_+}$, $\imath_{V_-}$ are the identity endomorphisms of the corresponding subspace (see the discussion following Th. 3.11).

Proof. From (III.2), (III.1) and Lemma III.2 it follows that the distribution of α is weakly Φ-spherical if and only if

$$\Phi(\alpha_\alpha(v),v) = \Phi(\sigma_\alpha \circ \gamma^*(v), \gamma^*(v)) \qquad \forall v \in V, \ \forall \gamma \in O(V,\Phi).$$

Since $V = V_+ \oplus V_-$, it is enough to find $\sigma_\alpha | V_+$ and $\sigma_\alpha | V_-$. Set $c_\alpha \stackrel{\text{def}}{=} \Phi(\sigma_\alpha(v_+^0), v_+^0)$ for $v_+^0 \in V_+$ such that $\|v_+^0\|^2 = 1$. Since for every $v_+ \in V_+$ there exists $\gamma \in O(V,\Phi)$ such that $\gamma^*(v_+^0) = \frac{v_+}{\|v_+\|}$, we have: $\forall v_+ \in V_+$

$$\Phi(\sigma_\alpha(v_+), v_+) = \|v_+\|^2 \, \Phi(\sigma_\alpha(\frac{v_+}{\|v_+\|}), \frac{v_+}{\|v_+\|}) =$$

$$= \|v_+\|^2 \, \Phi(\sigma_\alpha(\gamma^*(v_+^0)), \gamma^*(v_+^0)) = \|v_+\|^2 \, \Phi(\sigma_\alpha(v_+^0), v_+^0) =$$

$$= \|v_+\|^2 \cdot c_\alpha = \Phi(c_\alpha \cdot \imath_V(v_+), v_+).$$

It is easy to show that if $\Phi(\sigma_\alpha(v_+), v_+) = \Phi(\omega(v_+), v_+)$ for $\forall v_+ \in V_+$, then $\sigma_\alpha | V_+ = \omega | V_+$. Indeed, it is enough to substitute in the given identity the Φ-orthonormal basis vector of V_+.

From the last statement and the above computations it follows that

$$\sigma_\alpha | V_+ = c_\alpha \cdot \imath_{V_+}.$$

Let $d_\alpha = \Phi(\sigma_\alpha(v_-^0), v_-^0)$ for $v_-^0 \in V_-$, $\|v_-^0\|^2 = -1$.

Similarly to the above one can show that $\forall v_- \in V_- \quad \exists \gamma \in O(V,\Phi)$ such that

$$\Phi(\sigma_\alpha(v_-), v_-) = -\|v_-\|^2 \cdot d_\alpha = \Phi(-d_\alpha \cdot \imath_{V_-}(v_-), v_-),$$

from which similarly follows that

$$\sigma_\alpha | V_- = -d_\alpha \cdot \imath_{V_-}.$$

Finally we note $c_\alpha, d_\alpha \in \mathbb{R}_+$, since Σ_α is a non-negative bilinear form. ∎

If Φ is the Euclidean inner product, then for a weakly spherical distribution α we obtain: $\sigma_\alpha = c_\alpha \cdot \iota_V$.

As in the classical case, we can define for a random vector $\alpha : ((P,\Pi),B,\mu) \longrightarrow (V,\Phi)$ the characteristic function $\chi_\alpha : V \longrightarrow \mathbf{C}$

$$\chi_\alpha(v) \overset{\text{def}}{=} E(e^{i\alpha_v}) = E[\cos(\alpha_v)] + iE[\sin(\alpha_v)] =$$

$$= E[\cos\,\Phi(\alpha(p),v)] + iE[\sin\,\Phi(\alpha(p),v)].$$

Thus, *the characteristic function depends on the bilinear form* Φ, i.e., the mapping $\alpha \longrightarrow \chi_\alpha$ changes with the change of Φ.

It is easy to show (exactly as in the classical case, see [55], p. 43) that $\chi_\alpha(v) = \int_V e^{i\Phi(x,v)} dF_\alpha(x)$, where F_α is the distribution function of α.

Lemma III.5. Let $\alpha : P \longrightarrow (V,\Phi)$ be a random vector, and $\gamma : (V,\Phi) \longrightarrow (W,\Psi)$ be a homomorphism. Then, if the random vector $\beta : P \longrightarrow (W,\Psi)$ is defined by $\beta(p) = \gamma\circ\alpha(p) + w_0$ for some $w_0 \in W$, we have

$$\chi_\beta(w) = e^{i\Psi(w_0,w)} \chi_\alpha(\gamma^*(w)),$$

where γ^* is the adjoint of γ w.r.t. Φ, Ψ.

Proof. Indeed,

$$\chi_\beta(w) = E(e^{i\beta_w}) = E[e^{i\Psi(\beta,w)}] = E[e^{i\Psi(\gamma\circ\alpha,w)+i\Psi(w_0,w)}] =$$

$$= e^{i\Psi(w_0,w)} E[e^{i\Phi(\alpha,\gamma^*(w))}] = e^{i\Psi(w_0,w)} \chi_\alpha(\gamma^*(w)). \quad \blacksquare$$

Using the above lemma and the classical "inversion theorem" ([55], p. 51) one can show the validity of the inversion theorem for a pseudoeuclidean space (V,Φ). Indeed, if we set in Lemma III.5 $w_0 = 0$, Φ to be the Euclidean inner product, $W = V$, $\Psi = \Phi$, and $\gamma = \iota_V = \iota$ (identity mapping), then for $\alpha : P \longrightarrow V$

$$\chi_\alpha^\Phi = \chi_\alpha^{\text{Eucl.}} \circ \iota^*,$$

where ι^* is the adjoint of the identity endomorphism w.r.t. the above two forms. So, $\chi_\alpha^\Phi = \chi_\beta^\Phi \Longrightarrow \chi_\alpha^{\text{Eucl.}} = \chi_\beta^{\text{Eucl.}} \Longrightarrow$ by the classical inversion theorem that the two corresponding distributions are identical. In fact, since the matrix of ι^* w.r.t. the

Φ-orthonormal basis (which is also orthonormal w.r.t. the Euclidean inner product) is

$$J = \begin{pmatrix} I_{n^+} & O \\ O & -I_{n^-} \end{pmatrix},$$

where (n^+, n^-) is the signature of Φ, one can immediately write down one of the characteristic functions χ_α^Φ, $\chi_\alpha^{\text{Eucl.}}$ knowing the other.

Next, let $\alpha^1 : P \longrightarrow (V_1, \Phi_1)$ and $\alpha^2 : P \longrightarrow (V_2, \Phi_2)$ be two random vectors, and let $V = V_1 \times V_2$ be the direct product of V_1 and V_2. Define $\beta : P \longrightarrow (V, \Phi^\pm)$ for $\oplus$ ($\ominus$)-direct product as

$$\beta(p) = (\alpha^1(p), \alpha^2(p))$$

(see the end of section 3.1). Then β is a random vector, and α^1 and α^2 are independent if and only if

$$F_\beta(B_1 \times B_2) = F_{\alpha^1}(B_1) \cdot F_{\alpha^2}(B_2),$$

where F_δ is the distribution function of δ, B_1 and B_2 are arbitrary Borel sets from V_1 and V_2 respectively.

Theorem III.6. α^1 and α^2 are independent if and only if

$$\chi_\beta(v_1, v_2) = \chi_{\alpha^1}(v_1) \cdot \chi_{\alpha^2}(v_2) \qquad \text{for} \quad \oplus\text{-product},$$

$$\chi_\beta(v_1, v_2) = \chi_{\alpha^1}(v_1) \cdot \chi_{\alpha^2}(-v_2) \qquad \text{for} \quad \ominus\text{-product}.$$

Proof. ($\Longrightarrow$).

$$\chi_\beta(v_1, v_2) = E(e^{i\beta(v_1, v_2)}) = \int_{V_1 \times V_2} e^{i\Phi^\pm(x, (v_1, v_2))} dF_\beta(x) =$$

$$= \int_{V_1 \times V_2} e^{i[\Phi_1(x_1, v_1) \pm \Phi_2(x_2, v_2)]} dF_\beta(x_1, x_2) =$$

$$= \int_{V_1} \Big[\int_{V_2} e^{i\Phi_1(x_1, v_1)} e^{i\Phi_2(x_2, \pm v_2)} dF_{\alpha^2}(x_2) \Big] dF_{\alpha^1}(x_1) =$$

$$= \chi_{\alpha^1}(v_1) \cdot \chi_{\alpha^2}(\pm v_2),$$

where the equality before the last one follows from the independence and Fubini's theorem.

($\Longleftarrow$). We have

$\int_{V_1 \times V_2} e^{i\Phi^{\pm}(x_1,(v_1,v_2))} dF_\beta(x)$ = by the assumption and definition of the characteristic function

$$= \int_{V_1} e^{i\Phi_1(x_1,v_1)} dF_{\alpha^1}(x_1) \cdot \int_{V_2} e^{i\Phi_2(x_2,\pm v_2)} dF_{\alpha^2}(x_2) =$$

$$\int_{V_1}\int_{V_2} e^{i[\Phi_1(x_1,v_1)\pm\Phi_2(x_2,v_2)]} dF_{\alpha^1}(x_1) dF_{\alpha^2}(x_2) =$$

$$= \int_{V_1 \times V_2} e^{i\Phi^{\pm}(x,(v_1,v_2))} d\{F_{\alpha^1}(x_1) \cdot F_{\alpha^2}(x_2)\}.$$

The first expression is the characteristic function of the distribution function F_β and the last expression is that of $F_{\alpha^1} \cdot F_{\alpha^2}$. By the uniqueness theorem (see above) from the equality of characteristic functions follows the equality of distribution functions, i.e., $F_\beta(x_1,x_2) = F_{\alpha^1}(x_1) \cdot F_{\alpha^2}(x_2)$, which gives the independence of α^1 and α^2. ∎

Let $\bar{v}_1 = E(\alpha^1)$ and $\bar{v}^2 = E(\alpha^2)$. Then, if $\beta = (\alpha^1,\alpha^2)$ and $\bar{v} = E(\beta)$, from Theorem III.1 (setting γ to be the appropriate projection) it follows that

$$\bar{v} = (\bar{v}_1,\bar{v}_2),$$

i.e., from the existence of $\bar{v}_1$ and $\bar{v}_2$ follows the existence of $\bar{v}$.

Let σ_1 and σ_2 be the covariance endomorphisms of α^1 and α^2 respectively. It is easy to show that from the existence of σ_1 and σ_2 follows the existence of $\sigma_\beta = \sigma$, the covariance endomorphism of β. In order to describe the structure of σ, we shall need one more important notion.

Define a function $\Gamma : V_1 \times V_2 \longrightarrow \mathbb{R}$ by

$$\Gamma(v_1,v_2) = \mathrm{Cov}(\alpha^1_{v_1},\alpha^2_{v_2}) \qquad \forall v_1 \in V_1, \quad \forall v_2 \in V_2$$

Obviously, Γ is a bilinear function. As in the case of the covariance bilinear form, it is easy to show that for the function Γ there exists a unique homomorphism $\sigma_{12} : V_2 \longrightarrow V_1$ such that

$$\Gamma(v_1,v_2) = \Phi_1(v_1,\sigma_{12}(v_2)) \qquad \forall v_1 \in V_1, \quad \forall v_2 \in V_2.$$

The homomorphism σ_{12} is called the cross-covariance homomorphism of α^1 and α^2.

Theorem III.7. If V is a Θ-direct product of V_1 and V_2, then

$$\sigma = \begin{pmatrix} \sigma_1 & \sigma_{12} \\ \sigma_{12}^* & \sigma_2 \end{pmatrix}.$$

where σ_{12}^* is the adjoint of σ_{12} w.r.t. Φ_1 and Φ_2, and for a Θ-direct product

$$\sigma = \begin{pmatrix} \sigma_1 & -\sigma_{12} \\ \sigma_{12}^* & -\sigma_2 \end{pmatrix}.$$

Proof. $\forall (v_1,v_2) \in V_1 \times V_2$ we have

$$\mathrm{Var}(\beta_{(v_1,v_2)}) = \mathrm{Var}[\Phi^{\pm}(\beta(p),(v_1,v_2))] = \mathrm{Var}[\Phi_1(\alpha^1(p),v_1) \pm$$

$$\Phi_2(\alpha^2(p),v_2)] = \mathrm{Var}[\alpha^1_{v_1} \pm \alpha^2_{v_2}] = \mathrm{Var}(\alpha^1_{v_1}) + \mathrm{Var}(\alpha^2_{v_2}) \pm 2\,\mathrm{Cov}(\alpha^1_{v_1} \cdot \alpha^2_{v_2}) =$$

$$= \Phi_1(\sigma_1(v_1),v_1) + \Phi_2(\sigma_2(v_2),v_2) \pm 2\Phi_1(v_1,\sigma_{12}(v_2)) =$$

$$= \Phi^{\pm}\left(\begin{pmatrix} \sigma_1 & \pm\sigma_{12} \\ \sigma_{12}^* & \pm\sigma_2 \end{pmatrix}(v_1,v_2),(v_1,v_2)\right),$$

where $\dim(V_i) = n_i$, $i = 1,2$. We have to check only the last equality:

$$\Phi^{\pm}\left(\begin{pmatrix} \sigma_1 & \pm\sigma_{12} \\ \sigma_{12}^* & \pm\sigma_2 \end{pmatrix}(v_1,v_2),(v_1,v_2)\right) = \Phi^{\pm}((\sigma_1(v_1)\pm\sigma_{12}(v_2),\sigma_{12}^*(v_1)\pm\sigma_2(v_2)),$$

$$(v_1,v_2)) = \Phi_1(\sigma_1(v_1)\pm\sigma_{12}(v_2),v_1)\pm\Phi_2(\sigma_{12}^*(v_1)\pm\sigma_2(v_2),v_2) =$$

$$= \Phi_1(\sigma_1(v_1),v_1) \pm \Phi_1(\sigma_{12}(v_2),v_1) + \Phi_2(\sigma_{12}^*(v_1),v_2) + \Phi_2(\sigma_2(v_2),v_2),$$

from which the required equality follows, since $\Phi_2(\sigma_{12}^*(v_1),v_2) = \Phi_1(v_1,\sigma_{12}(v_2))$ by definition of σ_{12}^*. The theorem follows from Lemma III.2. ∎

The two random variables α^1 and α^2 are called uncorrelated, if $\mathrm{Cov}(\alpha^1_{v_1},\alpha^2_{v_2}) = 0 \quad \forall v_1 \in V_1, \ \forall v_2 \in V_2$. Obviously, α^1 and α^2 are uncorrelated if and only if the cross-covariance endomorphism σ_{12} is zero, equivalently

$$\sigma = \begin{pmatrix} \sigma_1 & 0 \\ 0 & \pm\sigma_2 \end{pmatrix}.$$

Theorem III.8. Let $\alpha : P \longrightarrow (V,\Phi)$ be a random vector and $\gamma_1 : (V,\Phi) \longrightarrow (W_1,\Psi_1)$, $\gamma_2 : (V,\Phi) \longrightarrow (W_2,\Psi_2)$ be two homomorphisms. Then $\gamma_1 \circ \alpha$ and $\gamma_2 \circ \alpha$ are uncorrelated $\iff \gamma_1 \circ \sigma_\alpha \circ \gamma_2^* = 0$, where σ_α is the covariance endomorphism of α.

Proof follows directly from the above definition (see [53], p. 2.6). ∎

A random vector $\alpha : P \longrightarrow (V,\Phi)$ has a normal distribution if $\forall v \in V \quad \alpha_v$ is normally distributed (on $\mathbb{R}$).

Theorem III.9. If the above α is normally distributed and $\gamma : (V,\Phi) \longrightarrow (W,\Psi)$ is a homomorphism, then $\beta : P \longrightarrow (W,\Psi)$ defined as $\beta(p) = \gamma \circ \alpha(p) + w_0$ is also normally distributed.

Proof. We have $\forall w \in W$

$$\beta_w(p) = \Psi(\beta(p),w) = \Psi(\gamma \circ \alpha(p) + w_0,w) = \Psi(\gamma \circ \alpha(p),w) + \Psi(w_0,w) =$$

$$= \Phi(\alpha(p),\gamma^*(w)) + c_0 = \alpha_{\gamma^*(w)}(p) + c_0,$$

where γ^* is the adjoint of γ w.r.t. Φ, Ψ, and $c_0 = \Psi(w_0,w)$. ∎

As in a Euclidean space, if α^i, $1 \le i \le n$, are independent random variables each having $N(0,1)$ distribution on $\mathbb{R}$, and $(e_i)_{1 \le i \le n}$ is Φ-orthonormal basis of (V,Φ), then a random vector α defined as

$$\alpha(p) = \sum_{i=1}^{n} \alpha^i(p) e_i$$

has a normal distribution. Indeed,

$$\alpha_v(p) = \Phi(\alpha(p), v) = \sum_{i=1}^{n} \Phi(e_i, v) \cdot \alpha^i(p)$$

is normally distributed, as a linear combination of independent normal random variables α^i. For the above α we have:

$$\forall v \in V \qquad \Phi(v, \bar{v}_\alpha) = E(\alpha_v) = \sum_{i=1}^{n} \Phi(e_i, v) E(\alpha^i) = 0,$$

and since Φ is non-degenerate it follows that $E(\alpha) = \bar{v}_\alpha = 0$. We also have

$$\Phi(\sigma_\alpha(v), v) = \mathrm{Var}(\alpha_v) = \sum_{i=1}^{n} \Phi^2(e_i, v) \cdot \mathrm{Var}(\alpha^i) = \sum_{i=1}^{n} \Phi^2(e_i, v) =$$

$$= \Phi(\nu(v), v),$$

where the matrix of ν w.r.t. (e_i) is equal to the matrix of Φ w.r.t. (e_i). By Lemma III.2 the matrix of σ_α w.r.t. (e_i) is J.

Let $\beta : P \longrightarrow (V, \Phi)$ be a random vector defined as

$$\beta(p) = \gamma \circ \alpha(p) + v_0,$$

where $v_0 \in V$ and γ is an endomorphism of V. Then, by Theorem III.9 β has a normal distribution, by Theorem III.1 and the remark preceding it, the mean of β is $\bar{v}_\beta = E(\beta) = \gamma[E(\alpha)] + v_0 = v_0$, and by Theorem III.3 the covariance endomorphism of β is $\sigma_\beta = \gamma \circ \sigma_\alpha \circ \gamma^* = \gamma \circ \nu \circ \gamma^*$, where ν is as above. The matrix of σ_β w.r.t. the above basis (e_i) is $M(\gamma) \cdot J \cdot M(\gamma^*)$. Substituting the expression for $M(\gamma^*)$ from Theorem 3.18 we obtain the expression for the matrix of σ_β w.r.t. to the above orthonormal basis:

$$M(\sigma_\beta) = M(\gamma) \cdot {}^t M(\gamma) \cdot J,$$

where $M(\gamma)$ is the matrix of γ w.r.t. (e_i). If (V, Φ) is a Euclidean space, then $J = I$, and we obtain the familiar expression for the covariance matrix. As in the classical case, every Φ-non-negative endomorphism σ_β can be the covariance endomorphism for some random vector β, since by varying γ in the above formula, we can obtain any Φ-non-negative σ_β. The notation $\alpha \sim N(\bar{v}, \sigma)$ will

be used to denote the fact that the random vector $\alpha : P \longrightarrow (V, \Phi)$ has a normal distribution with the mean vector $\bar{v}$ and the covariance endomorphism σ.

If β is a real-valued random variable which has a normal distribution $N(m,c)$ $(m \in \mathbb{R}, c>0)$, then the characteristic function of β is

$$\chi_\beta(t) = e^{imt - \frac{1}{2}ct^2}.$$

If $\alpha \sim N(\bar{v},\sigma)$, then, by definition,

$$\forall v \in V \qquad \alpha_v \sim N(\Phi(v,\bar{v}), \Phi(\sigma(v),v)),$$

since $E(\alpha_v) = \Phi(v,\bar{v})$ and $Var(\alpha_v) = \Sigma_\alpha(v,v) = \Phi(\sigma(v),v)$ (see above). Hence

$$\chi_{\alpha_v}(t) = e^{i\Phi(v,\bar{v})t - \frac{1}{2}\Phi(\sigma(v),v)t^2}.$$

Since $\chi_\alpha(v) = E(e^{i\alpha_v})$ and $\chi_{\alpha_v}(t) = E(e^{it\alpha_v})$, we get the characteristic function of α:

$$\chi_\alpha(v) = \chi_{\alpha_v}(1) = e^{i\Phi(v,\bar{v}) - \frac{1}{2}\Phi(\sigma(v),v)}. \tag{III.3}$$

Let $\alpha^1 : P \longrightarrow (V_1, \Phi_1)$ and $\alpha^2 : P \longrightarrow (V_2, \Phi_2)$ be random vectors such that

$$\beta = (\alpha^1, \alpha^2) \sim N(\bar{v},\sigma).$$

Then α^1 and α^2 have normal distributions and

$$\sigma = \begin{pmatrix} \sigma_1 & \pm\sigma_{12} \\ \sigma_{12}^* & \pm\sigma_2 \end{pmatrix}$$

(depending on the type of the direct product), where $\sigma_1 = \sigma_{\alpha^1}$, $\sigma_2 = \sigma_{\alpha^2}$, and σ_{12} is the cross-covariance endomorphism.

Theorem III.10. Under the above condition (that β is normal) α^1 and α^2 independent $\Longleftrightarrow$ α^1 and α^2 are uncorrelated.

Proof. Assuming (without loss of generality) that $E(\beta) = (\bar{v}_1, \bar{v}_2) = 0$ we have (use (III.3))

$$\chi_\beta(v_1,v_2) = e^{-\frac{1}{2}\Phi(\sigma(v),v)} =$$

$$= e^{-\frac{1}{2}[\Phi_1(\sigma_1(v_1),v_1) + \Phi_2(\sigma_2(v_2),v_2) \pm 2\Phi_1(v_1,\sigma_{12}(v_2))]} =$$

$$= \chi_{\alpha_1}(v_1) \cdot \chi_{\alpha_2}(v_2) \cdot e^{\pm\Phi_1(v_1,\sigma_{12}(v_2))}.$$

The last expression is equal to $\chi_{\alpha_1}(v_1) \cdot \chi_{\alpha_2}(v_2) \quad \forall v_1 \in V_1, \forall v_2 \in V_2$ if and only if $\sigma_{12} = 0$. Since α^2 is normally distributed, $\chi_{\alpha_2}(v_2) = \chi_{\alpha_2}(-v_2)$ (see (III.3)), and the theorem is proved. ∎

Theorem III.11. If an addition to the conditions of Theorem III.8 α is normally distributed, then, $\gamma_1 \circ \alpha$ and $\gamma_2 \circ \alpha$ are independent $\iff$ $\gamma_1 \circ \sigma_\alpha \circ \gamma_2^* = 0$.

Proof follows from Theorems III.8 and III.10.

Finally, a fundamental role in the proposed approach should play the notion of an invariant (or stable) family of probability measures (see [53], Chapter 7). The family of probability measures on a vector space V, $\{P_x | x \in X\}$, is called stable under the action of a subgroup G of endomorphisms of V, if $\forall x \in X$, $\forall \gamma \in G$ the probability measure γP_x defined as $\gamma P_x(B) = P_x(\gamma^{-1}(B))$, where B is a Borel set, belongs to the family. According to the fundamental point stated at the end of Chapter 4, in the proposed approach only those family of probability measures should be used that are invariant (stable) under the group of motions of (V,Φ), i.e., $G = Is(V,\Phi)$. The normal family is, obviously, stable under any group of motions (see Theorem III.9). The entire multivariate theory could be built within the framework of the invariance (see [59]).

References

[1] Simon, J.C., Some Current Topics in Clustering, in Relation with Pattern Recognition, Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, 1978, pp. 19-29.

[2] Diday, E., Simon, J.C., Clustering Analysis, in "Digital Pattern Recognition", ed. by Fu K.S., Springer-Verlag, 1976, pp. 47-94.

[3] Pavlidis, T., Structural Pattern Recognition, Springer-Verlag, 1977.

[4] Fu, K.S., Syntactic Methods in Pattern Recognition, Academic Press, 1974.

[5] Fu, K.S., Recent Advances in Syntactic Pattern Recognition, Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, 1978, pp. 13-18.

[6] Duda, R.O. and Hart, P.E., Pattern Classification and Scene Analysis, J. Wiley & Sons, 1973.

[7] Twersky, A., Features of Similarity, Psychological Review, Vol. 84, No. 4, 1977, pp. 327-352.

[8] Frechet, M., Sur quelques points du calcul fonctionnel, Rendiconti del Circolo Matematico di Palermo, 22, 1906, pp. 1-74.

[9] Hausdorff, F., Grundzüge der Mengenlehre, Leipzig, 1914.

[10] Jardin, N., Sibson, R., Mathematical Taxonomy, J. Wiley & Sons, 1971.

[11] Fu, K.S., Lu, S.Y., A Clustering Procedure for Syntactic Patterns, IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-7, No. 10, 1977, pp. 734-742.

[12] Bribiesca, E., Guzman, A., How to describe pure form and how to measure differences in shapes using shape numbers, Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, 1979, pp. 427-436.

[13] Burr, D.J., A technique for comparing curves, Proc. IEEE Comp. Soc. Conf. on Pattern Recognition and Image Processing, 1979, pp. 271-276.

[14] Marriott, F.H.C., The Interpretation of Multiple Observations, Academic Press, 1974.

[15] Schoenberg, I.J., Remarks to Maurice Frechet's article ..., Annals of Math. (2), 36, 1935, pp. 724-732.

[16] Riemann, B., Ueber die Hypothesen, welche der Geometrie zu Grunder lisgen (Three English translations: a) in Vol. 2 of Smith's "Source book in Mathematics", Dover; b) in Clifford's "Mathematical Papers", Chelsea; c) in Vol. 2 of Spivak's "A Comprehensive Introduction to Differential Geometry", Chapter 4A, Publish or Perish, Boston, 1970.) (I used the last translation.)

[17] Dirac, P.A.M., The physical interpretation of quantum mechanics, Proc. Roy. Soc. London, Ser. A 180, 1942, pp. 1-40.

[18] Sobolev, S.L., The motion of a symmetric top containing a cavity filled with a liquid, Z. Prikl. Meh. i Tech. Fiz. No. 3, 1960, pp. 20-55 (Russian).

[19] Luneburg, R.K., Mathematical analysis of binocular vision, Princeton University Press, 1947.

[20] Torgerson, W.S., Theory and methods of scaling, J. Wiley & Sons, 1958.

[21] Lawson, C.L., Hanson, R.J., Solving Least Squares Problems, Prentice-Hall, 1974.

[22] Bremermann, H.J., What Mathematics Can and Cannot do for Pattern Recognition, in "Pattern Recognition in Biological and Technical Systems", Springer-Verlag, 1971, pp. 31-45.

[23] Arkadev, A.G., Braverman, E.M., Computers and Pattern Recognition, Thompson, 1966.

[24] Toussaint, G.T., Pattern Recognition and Geometrical Complexity, 5th International Conf. on Pattern Recognition, 1980, pp. 1324-1347.

[25] Uesaka, Y., Aizawa, T., Ebara, T., Ozeki, K., A theory of learnability, Kibernetik 13, 1973, pp. 123-131.

[26] Levenstein, V.I., Binary Codes Capable of Correcting Deletions, Insertions, and Reversals, Soviet Physics-Doclady, Vol. 10, No. 8, 1966, pp. 707-710.

[27] Wagner, R.A., Fisher, M.J., The String to String Correction Problem, J - ACM, Vol. 21, 1974, pp. 168-173.

[28] Kashyap, R.L., Oomen, B.J., An Effective Algorithm for String Correction Using a Generalized Distance, Proc. IEEE Comp. Soc. Conference on Pattern Recognition and Image Processing, 1979, pp. 184-191.

[29] Calude, C., Asupra distantelor contextuale in lingvistica matematica (On contextual distances in mathematical linguistics). Studii si cercetari matematice, 1, 1976.

[30] Dinca, A., Distante contextuale in lingvistica algebrica (Contextual distances in algebraic linguistics). Studii si cercetari matematice, 2, 1973.

[31] Markus, S., Introduction mathematique a linguistique structurale, Dunod, Paris, 1967.

[32] Sokal, R.R., Sneath, H.A., Principles of Numerical Taxonomy, Freeman, 1963.

[33] Godement, R., Algebra, Herman, Paris, 1968.

[34] Dieudonné, J., Linear Algebra and Geometry, Herman, Paris, 1969.

[35] Greub, W., Linear Algebra, Springer, 1974.

[36] Gruenberg, K.W., Weir, A.J., Linear Geometry, Springer-Verlag, 1977.

[37] Jacobson, N., Basic Algebra I, Freeman, 1974.

[38] Kaplansky, I., Linear Algebra and Geometry, Alyn & Bacon, 1969.

[39] Mal'cev, A., Foundations of Linear Algebra, Freeman, 1963.

[40] Schoenberg, I.J., Regular simplices and quadratic forms, J. of the London Math. Soc., Vol. 12, 1937, pp. 48-55.

[41] Wilkinson, J.H., Reinsch, C., (ed. by F.L. Bauer, et al.) Handbook for Automatic Computation, II, Linear Algebra, Springer-Verlag, 1971.

[42] Engelking, R., Outline of General Topology, North-Holland, 1968.

[43] Bognar, J., Indefinite Inner Product Spaces, Springer-Verlag, 1974.

[44] Dempster, A.P., Elements of Continuous Multivariate Analysis, Addison-Wesley, 1969.

[45] Vakhania, N.N., Probability distributions on linear spaces, Elsevier, North Holland, 1981.

[46] Fukunaga, K., Introduction to Statistical Pattern Recognition, Academic Press, 1972.

[47] Rao, C.R., Linear Statistical Inference and Its Applications, Second Edition, J. Wiley & Sons, 1973.

[48] Pontryagin, L.S., Foundations of Combinatorial Topology, Graylock Press, 1952.

[49] Fukunaga, K., Olsen, D.R., An alogirthm for finding the intrinsic dimensionality of data, IEEE Trans. Computers C-20, 1971, pp. 176-183.

[50] Shepard, R.N., Multidimensional Scaling and Related Nondimensional Representation, Prepared for US/USSR Interacademy Seminar on Descriptive and Normative Models for Decision Making, March 1979, pp. 1-44.

[51] Nilsson, N.J., Learning Machines, McGraw-Hill, 1965.

[52] Friedman, H.P., Rubin, J., On some invariant criteria for grouping data, J. Amer. Stat. Assn., 62, 1967, pp. 1159-1178.

[53] Eaton, M.L., Multivariate Statistical Analysis, Institute of Math. Stat., Univ. of Copenhagen, 1972.

[54] Royden, H.L., Real Analysis, Second Edition, MacMillan, 1968.

[55] Tucker, H.C., A graduate course in Probability, Academic Press, 1967.

[56] Barr, D.R., Zehna, P.W., Probability, Brooks/Cole, 1971.

[57] Goldfarb, L., An Outline of a New Approach to Pattern Recognition, Second IASTED International Symposium on Robotics and Automation, Lugano (Switzerland), Acta Press, 1983, pp. 129-132.

[58] Goldfarb, L., A Unified Approach to Pattern Recognition, Pattern Recognition (accepted for publ.), 1984.

[59] Eaton, M.L., Multivariate Statistics, J. Wiley & Sons, 1983.

ISBN : 0 444 87723 1