

A MICROCOMPUTER BASED
IMAGE ANALYSIS SYSTEM
FOR FLEXIBLE AUTOMATION

by

George Alexander Cartounis, B.Sc., M.Sc.

March 1983

A thesis submitted for the degree of Doctor of Philosophy
of the University of London and for the Diploma
of Membership of the Imperial College.

Department of Mechanical Engineering
Imperial College of Science and Technology
University of London

This work is dedicated to my daughter Flora.

ABSTRACT

The new generation of sophisticated robots combine sensory devices and computer technology to achieve adaptability to a changing working environment, a feature which renders them particularly attractive for small batch production lines. Real-time computer vision systems have already been used for pick-and-place operations, offering an efficient source of sensory information. Their cost, however, still constitutes a high portion of the total system cost, raising it to unattractive levels for this application area.

The aim of this project is to investigate the feasibility of applying efficient image processing and pattern recognition techniques on an 8-bit microcomputer, with the scope of designing a low-cost robotic vision system able to perform simple recognition and two-dimensional position determination tasks, in a non-structured environment at a reasonable speed and efficiency.

The thesis describes a model system incorporating recognition, training and other operating routines. The system's hardware was built around a Motorola 6809 microcomputer development system, reinforced by a floating-point processor card, and interfaced to a column digitizer, both purposely designed.

A 128x96 image array of 16 gray-levels is buffered in the memory. Each point is represented by an 8-bit number, the 4-bits of which are reserved for a 'mirror image' to be composed by meaningful traces left behind by the various processing stages.

Both gradient and light-dark information is used by a new contour follower which performs image segmentation. A set of features is extracted for each contour and compared to a series of model sets, obtained during a training-by-showing stage. Holes are described in a new way by a set of features related to a unified pattern of

internal points. All recognition software was written in assembly language for real-time operation.

The performance of the system under normal lighting conditions and for essentially flat, non-touching objects was satisfactory, so proving that further work, directed to the development of vision systems based on low cost microcomputers, is worthwhile.

ACKNOWLEDGEMENTS

I wish to express my gratitude to my supervisor Dr. C.B. Besant for suggesting the idea of this project and for his continuous help and encouragement.

Sincere thanks are also due to Prof. N. Chrysochoides for giving me the opportunity to carry out this work.

I wish to acknowledge the friendly assistance of all my colleagues, especially that of Dr. S.K. Khurmi, T.Jin, Dr. H.A. Pak, C. Balasubramaniam, S. Ahmad, P. Niccolls, N. Zarimnas, and L. Daneshmed.

I would also like to express my deepest feelings of gratitude to my wife and my daughter for their patience and understanding, my parents for their moral and financial support, and my friends Stefano Furlis, John Caloumenos, Maria Klitsa, and Gerti Filis for their moral support and their help.

The financial support provided by the International Atomic Energy Agency during the first two years of this project is deeply appreciated.

CONTENTS	Page
ABSTRACT	ii
ACKNOWLEDGEMENTS	iv
CONTENTS	v
CHAPTER 1: INTRODUCTION	1
1.1 Recent developments in industrial automation.	
1.2 The advent of the 'industrial robot' - Types of robots	2
1.3 Future systems - The need for sensor-based robots	5
CHAPTER 2: REVIEW OF SENSORS - VISION SYSTEMS	8
2.1 Applications of sensors in flexible automation	
2.2 Industrial Vision Systems	11
2.2.1 Industrial requirements	12
2.2.2 Functional description of a typical vision system	
2.2.3 Major function blocks of a typical vision system	17
2.2.4 Industrial Applications of Vision Systems - Review of existing systems	28
CHAPTER 3: DESIGN PHILOSOPHY	43
3.1 General Introduction	
3.2 System Specifications	44
3.3 The Basic Decisions	46
CHAPTER 4: IMAGE ANALYSIS AND DESCRIPTION	50
4.1 Basic Elements	
4.1.1 Introduction	
4.1.2 Data Structure. Digital representation of picture elements	
4.1.3 The problem of Image Segmentation	54
4.1.4 The Raster-Scan Search	56
4.2 CONTOUR FOLLOWING (Edge Following)	59
4.2.1 Introduction	
4.2.2 Basic Steps	60

	<i>Page</i>
4.2.3 STEP A: Definition of Candidate Links - Directional Grading	61
4.2.4 STEP B: Selection of the most probable link.	69
4.2.5 STEP C: CONTROL FUNCTIONS	74
4.3 Primitive Features of the Outline	81
4.3.1 Introduction	
4.3.2 Calculation of Primitive Features	83
4.4 Internal Features	90
4.4.1 Introduction	
4.4.2 Search for points of the internal pattern.	92
4.4.3 Internal Segmentation	102
4.4.4 Description of the Internal Pattern (Primitive Features)	117
4.5 Orientation	120
CHAPTER 5: CLASSIFICATION AND LEARNING PROCEDURES	123
5.1 Theoretical Background (Statistical Pattern Recognition)	
5.2 Data Organisation for Classification and Learning	132
5.2.1 The Compound Features Set	133
5.2.2 Models and Model Repertoires	135
5.3 The Classifier	137
5.4 The Learning Process	140
CHAPTER 6: DESCRIPTION OF SYSTEM HARDWARE AND STRUCTURE OF SYSTEM SOFTWARE	142
6.1 System Hardware	
6.2 System operation - ICAROS Operating System	146
6.3 System Software	154
6.3.1 Software Modules	155
6.3.2 Database Structure - Image Files	160

	<i>Page</i>
CHAPTER 7: PERFORMANCE - EVALUATION	162
7.1 Positional Accuracy, Directional Accuracy, Timing	
7.1.1 Positional Accuracy	
7.1.2 Directional Accuracy	166
7.1.3 Timing Performance	
7.2 Detection and Identification Accuracy	169
7.2.1 Test Case	170
7.3 Comparison with existing systems	174
 CHAPTER 8: CONCLUSIONS - SUGGESTIONS FOR FUTURE WORK	
8.1 Conclusions	181
8.2 Suggestions for future work	185
8.2.1 Immediate improvements	186
8.2.2 Expansion of the system	187
8.2.3 Hardware realization	188
 REFERENCES	192
 APPENDIX A: COLUMN DIGITISER HARDWARE	198
APPENDIX B: VISION UTILITY COMMANDS	203
APPENDIX C: FEATURE LISTS OF TEST OBJECTS	207

CHAPTER 1 : INTRODUCTION

1.1. Recent Developments in Industrial Automation

Flexible or programmable automation is a term used to define a new form of industrial automation that has made its first cautious but impressive steps in industry in the last 20 years.

A combination of computer programmability, use of sensors, and novel methods of mechanical handling, this new implementation of computer control techniques on the shop floor, offers amazing possibilities and in the same time creates new problems for industry and the society in general.

The existing style of industrial automation, that has developed for so many years after the industrial revolution, is now called "fixed" or "hard" automation.

Machines were at first built to utilise steam and then electric power to replace human muscles. Mass production techniques were introduced to increase productivity, and special purpose machines were built that can produce a particular component or sub-assembly with minimum human intervention.

These machines are still economically justifiable only for large scale production. Most of the workforce in such big factories is assigned to load and unload the same machine, or go through one specific routine of operations on an assembly line.

The industrial scene remained essentially the same, until the advent of numerical controlled (NC) machines in the 1950's. The new concept of numerical control offered a certain degree of flexibility and programmability, that made these machines very popular, mainly for metal cutting.

At about the same time (early 1950's) digital computers began to

be commercially available, and the first patents for programmable material handling machines were issued.

The early 1960's saw the first developments in the field of powered artificial limbs and braces and the first "robots" being delivered to industry.

1.2. The advent of the "industrial robot" - Types of robots

The word "robot" was first used by Karel Capek, a famous Czech writer, in his novel "Rossum's Universal Robots". In his understanding "robot" means a humanoid system, capable of performing intellectual, as well as physical work.

It is our opinion, that the choice of the term "industrial robot" to describe a flexible automation system, was not the best, mainly because it gives a rather false impression about the potential capabilities, the appearance and the structure of such a system. An anthropomorphic image has been already built around this term, which can in a way influence the minds of both potential users and future designers, as to what these systems can do, what are the jobs that suit them best and how they should be designed to meet the needs of a particular application.

The way a human being performs a given task is not always the easiest or the most suitable way for a machine to perform the same task.

The industrial robot has been developed as part of the automation in industry; it should be viewed as an integrated part of a production system, and not as an independent miracle performer.

The earliest manipulators that bore some resemblance to today's industrial robots were the teleoperators, mechanical devices guided and powered by human beings.

These devices were eventually equipped with their own electric or hydraulic power and some means of electronic or hydraulic feedback. The next major step was to provide them with some means of storing the sequence of motions that they were expected to follow.

A new concept of "teaching" the manipulator emerged out of this innovation.

A numerically controlled manipulator with some storage capabilities and at least 2-3 degrees of freedom meets the minimum requirements of being termed as an industrial robot. In fact most of the existing simple robots are of this type.

The control schemes that are used in this type of robots are entirely point-to-point or finite-state, for each of the axes, and the overall movement is achieved in an asynchronous sequential mode.

The information provided to the controller consists of a sequence of points, stored in memory during training, together with timing information.

During operation, each axis actuator takes its predefined position in sequence, and the whole system moves to a new position only after all axes have attained their desired values.

There are two types of these elementary programmable robots, according to their degree of flexibility:

- 1) The "pick-and-place" devices which have usually very limited memory capacity and computational capabilities, 3-4 degrees of freedom and a binary choice of possible positions for each axis, i.e. up/down, right/left, in/out. Mechanical stops are used to set the amount of travel for each axis.

They are more suitable for rather simple operations in medium and long-run production industries. Their prices range from \$ 3000 to \$ 5000.

- 2) The multiple-point, multiple sequence or medium-range robots, offer greater flexibility through a higher degree of programmability and increased storage capacity. They normally have 4-6 degrees of freedom and some or all of their axes can be positioned by servo-control to a series of programmable orientations.

They can follow up to 256 steps, but they do not possess any branching or interrupt capabilities. Their price range is \$ 15000 to \$ 35000.

3) Sophisticated robots : While point-to-point control proved practical and sufficient in many industrial applications, there are other needs in industry which can be satisfied only by a more precisely controlled robot, in which a continuous path for all joints is specified. Continuous path control implies synchronous operation in space, as well as in time. Such a requirement can only be met with greater storage capacity, or computational power, or both. More sophisticated software tools, comprising branching and interrupts handling, and low-level decision making routines, are necessary to enable the controller to adapt to a more complex environment in a smoother and more flexible way.

Until recently, this kind of storage and computational requirements could only be fulfilled by the use of slow magnetic storage, and expensive digital computers.

The tremendous breakthrough emerged the last 5 years, as a direct result of the boom in the microelectronics industry. During this period, there was a 10:1 reduction in the cost of semiconductor memories and an almost 15:1 increase in their packaging densities.

The microprocessor, the central processing unit (CPU) of a digital computer, was first realised on an integrated circuit of approx. 5 sq.mm in area, in 1971. Today a whole range of similar products is available, ranging from 8-bit microcomputers and complete microcontrollers on a "chip", to 32 bit microprocessor components with mainframe capabilities. An equally wide variety of interface and support circuits is available.

Complete microcomputer systems, with similar features to the ones found in large and expensive computers, can be now purchased at a fraction of the cost.

Distributed-processing systems employing several inter-connected

processing centres in various configurations, are now used for a wide range of applications, which require high-speed and design modularity. A few years ago, these systems were only considered for plant-process control and similar applications, where the extremely high cost was justifiable.

All these hardware innovations, backed by accumulated human knowledge in various fields, such as digital control, computer science and artificial intelligence constitute the framework for the development of the sophisticated robots.

Continuous path-control, automatic program change, depending on the state and character of components arriving at a workstation, training by showing, by simulation on a terminal, or by off-line programming in special high-level languages, and moving line tracking capabilities, render these robots capable of performing tasks as complex as paint-spraying, polishing, grinding and even part-assembly related tasks. Sophisticated robots cost in the range of \$ 40,000 to \$ 150,000 depending on configuration.

1.3. Future systems - The need for sensor-based robots

The results of a recent survey conducted among robot users in Britain (Refs 1,2), showed that the most desirable developments in robot technology from the user's point of view relate to the flexibility and adaptability of these devices in the industrial environment (see Fig.1.1).

The majority of today's robots are almost entirely preprogrammed, memory-controlled machines.

It is the control side, which is the major source of their weaknesses, and it is therefore the control side which needs drastic improvements, if flexible automation is going to prove a viable and efficient alternative for even more new important applications.

A brief reflection on the results of the above mentioned survey will show that the limited adaptive control capabilities of today's

machines are responsible for almost all their limitations, even when this is not clearly evident.

Let us for instance consider the desired improvement on the mechanical side and, more specifically, those related to higher speed, more compact size, increase of carrying capacity and greater precision.

A human arm can lift up to ten times its own weight. Most industrial robots are not able to lift even one tenth of their weight. Is it the mechanical structure the only, or even the main reason of this 1:100 disadvantage?

The human arm is adaptively guided to its destination by tactile and/or visual feedback, whereas existing industrial robots rely solely on structural rigidity in order to transform accurately external-target coordinates to internal-joint coordinates. Structural rigidity means excess weight and size, which in turn mean speed limitations.

From the above short discussion it becomes clear that extending the present capabilities of industrial robots will require above all improvements in two major directions:

- a) Real-time adaptive control.
- b) Utilisation of sensory facilities.

Sensory devices are needed to provide feedback information on:

- a) The state variables of the robot itself
(internal sensing of arms position and velocity
- tactile, force and torque information
- measurement of heat dissipation, etc.).
- b) The task related environment
(workpiece position and orientation -
- relative positions of ancillary equipment, etc.).

Sensory devices can also play an important role in the prevention of accidents and the detection of malfunctions.

A. GREATER MECHANICAL FLEXIBILITY
. More degrees of freedom . Higher speed . More compact size . Larger assortment of attachments . Increase of carrying capacity . Larger scope of movement . Greater precision (locational accuracy) . Walking capacity
B. GREATER PROGRAMMABILITY AND ADAPTABILITY
. Easier change of program . Increase of memory capacity . Addition of tactile sense . Addition of visual sense and pattern recognition capacity

Fig. 1.1 - Desirable improvements in robot technology

CHAPTER 2 : REVIEW OF SENSORS - VISION SYSTEMS

2.1. Applications of sensors in flexible automation

Mechanical and electronic sensory devices have been employed in industrial automation for many years for various purposes. Limit switches, shaft encoders, proximity sensors, temperature sensors, strain gauges and various other transducers are employed in the majority of "fixed" automation machines.

Most of these devices are purposely designed for each application area, in order to provide the necessary information in the most straightforward and reliable way.

Recent technological developments, however, have enabled a remarkable increase in the degree of sophistication and standardisation of sensory systems combined with a remarkable decrease of their cost.

Depending on the way they interact with their environment in order to acquire the necessary information, sensors can be classified in two categories:

- a) Contact sensors, operating by physical contact.
- b) Non-contact sensors, functioning at a distance from the information source.

The brief survey that follows is focused mainly on devices that have been applied, or can be applied in flexible automation systems, either as internal sensing elements, or as sources of information about the work environment.

Vision systems are treated separately in more detail, being the main subject of this thesis.

Contact Sensors : Contact sensing of force, torque and touch can be usefully combined with remote sensing for material handling and assembly tasks.

The basic operations needed for these tasks include grasping, handling, orienting, inserting, aligning, fitting, screwing, etc.

The best scheme is to employ remote sensing for coarse examination and monitoring, and contact sensing for fine resolution, which is needed in the grasping or fitting stages.

Force and torque sensors can also be used for internal sensing, which is necessary for the control of the joint movements.

Measuring the forces acting on the joints of the manipulator, can be done simply by measuring the armature current of an electrical motor, or the back pressure of a hydraulic motor, without employing any additional sensors.

A combination of such measurements can also be used to calculate the forces acting on the end effector (the "hand"). But the accuracy and the resolution of such a method would be adversely affected by the variability in the inertia of the arm and its load and by the non-uniform friction of the individual joints.

A more accurate method is to measure the 3 components of force and the 3 components of torque, using a compliant structure with force transducers (strain gauges) mounted on it. This structure could be based either on the wrist of the manipulator hand, or mounted on a pedestal which holds the workpiece.

A 6-axis force sensor has been used by Draper Laboratories in Cambridge Massachusetts, for round peg-hole insertion tasks (Ref. 3)

A similar combination of force-feedback and a compliant wrist has been used by Hittachi for assembly purposes.

The Olivetti SIGMA robot has a force sensitive wrist in which displacement is measured against springs by position sensors (Ref.4)

If flexible automation systems are to progress to more complex assembly tasks, some kind of touch sensors have to be employed, so that smaller and more delicate parts can be handled.

Touch (or tactile) sensors are applicable to a variety of tasks, such as reaching a target, preventing collision damage, following a path, centering the gripper on an object, measuring dimensions, or even recognising an object by touching it. An array of touch sensors can be used to provide information about the shape of an object and the way it is handled. Existing arrays, however, have insufficient compliance and coarse spatial resolution.

Simple touch sensors have already been implemented successfully for path following in arc-welding. The sensing device can be a simple copper stylus mounted in a water cooled heat sink (Ref. 5), or a needle displacement sensor with photodiodes to measure the displacement of the needle which tracks the seam.

The situation is far more complicated in applications which require array sensors. The basic problem here is the sensing material which has to be sensitive, stable, and in the same time mechanically strong. A common material which has been used up to now is carbonised conductive foam mounted between two conducting surfaces. Its electrical conductivity varies with compression, but it becomes unstable after long use.

Two new, more promising materials have been tried recently, conductive silicon rubber (Ref. 6), and carbon fibre (Ref. 7). They offer increased mechanical strength, higher dynamic range and better stability.

Non-contact sensors : Non-contact sensors are particularly useful in locating and indentifying parts. They can also be used for automatic inspection. Proximity sensors, range-finding sensors and electro-optical imaging sensors (vision systems), have been used for various applications in flexible automation.

The most popular type of proximity sensors is the photoelectric transmitter-receiver pair, consisting of a solid-state light-emitting diode (LED), and a solid-state photodiode, both integrated in a small package.

They can be used for effective collision avoidance and for detecting the presence of an object within the hand area.

Range-finding sensors have been tried mainly in moving robot experiments, but they can be of great importance for three-dimensional recognition of parts and for visual feedback control.

They measure the distance from themselves to a raster of points in the scene. The calculation of distance is based either on the trigonometry of triangulation, or on the time of flight of light (or sound).

Triangulation is based on the comparison of two images of the same scene obtained from different angles. Time of flight calculation is based on transmitting a laser pulse or an ultrasonic burst and measuring the arrival time of the reflected signal.

Scanning laser rangefinders have been also implemented recently in a configuration that finds the three-dimensional orientation of work - pieces (Ref. 8)

2.2. Industrial Vision Systems

The most general and efficient method of acquiring information about the working environment, is probably the use of visual sensors.

A visual sensor is a non-contact remote sensor that detects and processes information in the form of images carried by radiation at frequencies falling in the visual range (light waves). The two major distinctive characteristics of vision systems are:

- a) The structure of light information that they process (one-two or three dimensional images).
- b) The image processing capabilities that such systems possess.

Simple photocell detectors do not constitute vision systems according to this definition, unless they are arranged in an image array.

2.2.1. Industrial requirements

High performance image processing systems have been used in many applications (radar, medical diagnostics, satellite data analysis, etc.). These systems are customly designed to meet specific needs in the relevant fields, and they are therefore built in small series only. This makes them very costly and/or very slow in operation to be accepted as industrial vision systems.

For most industrial applications the entire vision system must cost from £ 3000 to £ 5000 or, in more general terms, its cost must represent a small fraction (10 % to 20 %) of the total cost for a complete flexible automation work-cell (manipulator, controller, tools).

Operating speed requirements vary, according to the application, from a fraction of a second, which is necessary for real-time visual servoing, to a few seconds (4-5 secs) for assembly operations, where the cycle time is determined by the nature of the assembly task and the speed of the manipulator.

The main functions that industrial vision sensors are expected to perform, are the following:

- a) Determination of position and orientation angle of a known stationary or moving object.
- b) Recognition of known, thoroughly predefined objects.
- c) Measurement of predefined object parameters for inspection purposes.
- d) Location of salient features of an object, in order to establish a spatial reference for visual servoing.

2.2.2. Functional description of a typical vision system

An industrial vision system could be described macroscopically as a "black-box", that is looking at a scene and produces at its output a certain amount of information concerning the scene.

A typical minimal set of output information will include a position vector (x, ϕ, θ) and an identification code I for each object in the scene.

So basically this black box could be described as an "intelligent" programmable data acquisition and data reduction system, which receives an analog electrical signal from a suitable image transducer (TV camera), and operates on it in order to extract exactly the information that is needed for a particular application.

Image processing can be achieved either by analog optical techniques or by digital techniques. Optical systems offer high speed parallel processing and they are particularly useful for the implementation of processes involving Fourier transforms of the original image, or other analytical transformations used in image enhancement. Digital processors, on the other hand, are capable of applying a wide range of algorithms, and can be easily reconfigured to meet various requirements.

They offer high precision and good repeatability of calculations, and there is the possibility of interactive processing, which is extremely important in the development stages.

These advantages have led to a marked preference for digital techniques in image processing and pattern recognition. However, it seems that special purpose optical systems will probably find some place in certain application areas.

The first major problem in the design of a digital image processor is the selection of a suitable digital representation of the image.

A black-and-white picture can be thought of as a real-valued function of two variables x and y , with $0 \leq x \leq L$ and $0 \leq y \leq W$ (see Fig. 2.1.). At each point (x, y) , the value of the function $I(x, y)$ is proportional to the light intensity at this point. The intensity of an image at a point is also called the gray level or the brightness.

Having defined the analog image function, the next step is to find a digital approximation of this function, that will retain the necessary information for subsequent digital processing. This is usually accomplished by sampling the image function at a discrete number of

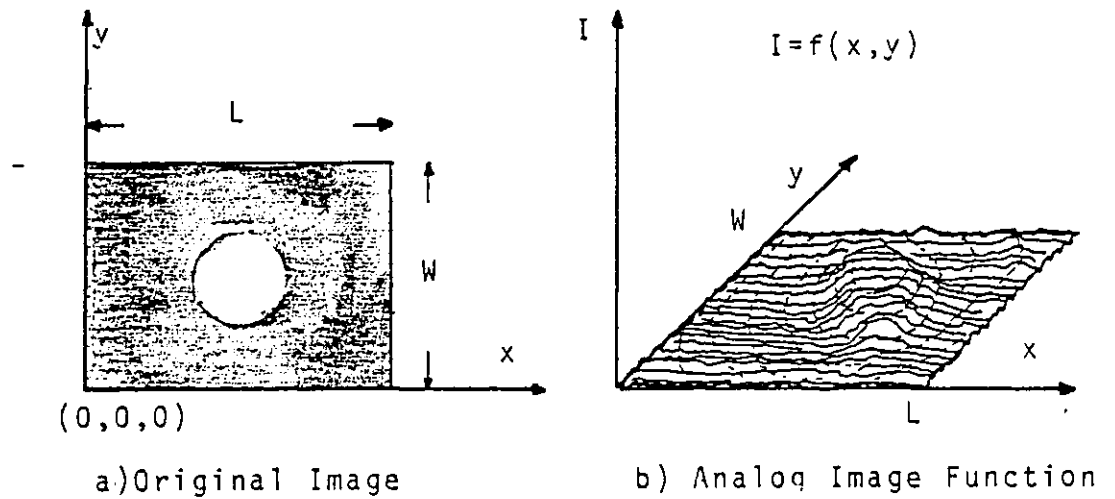
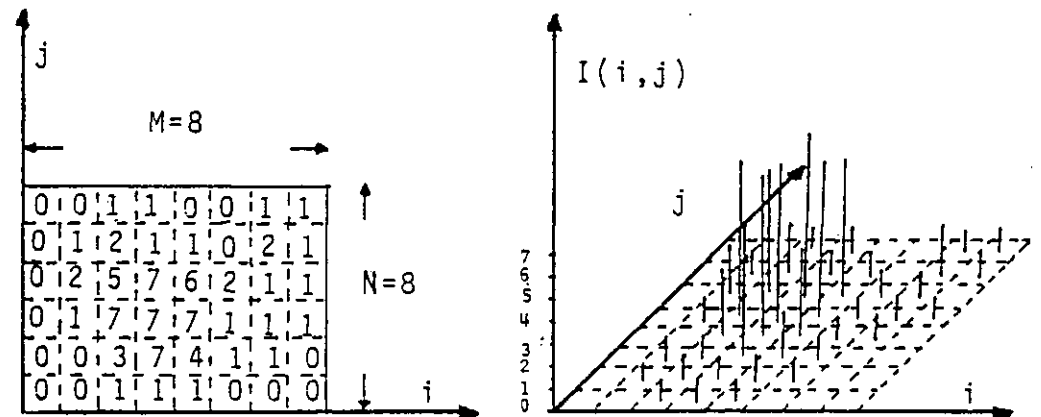


Fig. 2.1 Description of a picture as an analog Image function.



points in the X-Y plane, and storing the resulting array of digital numbers. The image is partitioned by a grid and the image function is sampled at the centre of each cell of the grid. This process is called sampling or quantization. A quadrupled grid is the most simple choice because it results in straightforward two entry addressing scheme for referencing the image points.

Hexagonal grids have also been used, because they possess some interesting connectivity properties.

In any case the most important factor is the size of the grid, i.e. the total number of grid points, which specifies the spatial resolution of the digital representations.

The minimum sample spacing that preserves the information content of the original image is given by Shannon's sampling theorem, which merely states that, if the image function is a bandlimited two dimensional function with bandwidths W_x and W_y , in the x and y directions, then the minimum sample spacing must be $1/2 W_x$ and $1/2 W_y$ respectively. These limits are not critical for many vision applications, where the main objective is not to preserve the information content of the whole image, but to preserve as much information as it is necessary, in order to extract the relevant features that will lead to recognition and position determination for the objects in the scene.

In most of the cases therefore, the minimum spatial resolution is imposed by the accuracy requirements of the particular application, and not by the quality of the digitized image.

Spatial resolutions of industrial vision systems can vary from 50 x 50 picture elements (pixels) to 512 x 512 pixels.

The image function can assume any value between some minimum (black) and maximum (white), while a digital processor operates only on a finite number of discrete values.

We must therefore also partition the range of intensity values into a finite number of discrete gray levels.

This is called level quantization or gray scale quantization. The simplest way of achieving it, is obviously to choose a suitable threshold value of intensity and assign a binary value of 1 (white) to all intensity values that are above this threshold, and a binary value of 0 (black), to all intensity values below the threshold. This results in a "binary" image, i.e. an array of 0's and 1's with 0's representing the background and 1's representing the object areas (or vice versa).

This simple quantization scheme provides a drastic reduction in image data and processing time, but it requires a high contrast between the objects of interest and the background, and/or a complicated process of threshold selection, which can vary within the image area.

Most of the existing vision systems operate on binary images, only a few of them use a multiple gray-level representation, with the number of gray levels ranging from 8 to 64.

Fig. 2.2. illustrates the effect of sampling the original image in Fig. 2.1. at the centres of 64 cells of an 8 by 8 grid, at 8 discrete gray levels. The resulting array of integers constitutes a digital image function.

A typical image array of, say 128 by 128 integers in the range of 0-15 (16 gray-levels) contains a total information of 65536 bits and requires 8192 memory locations in an 8-bit per word organized memory, for storage. If we compare this to the amount of information that we are actually interested in getting at the output of the vision system, typically 4 16-bit words per object in the scene (2 words for the position coordinates x-y, one word for the orientation angle θ , and one word for the identification number assigned to the object), we can see that a huge data reduction is involved in the operation of any vision system.

The most common *approach* to the effective reduction of data is to apply a sequence of operations on the image array, each operation resulting in a new reduced set of data, that contains the necessary

information in a more compact form (bottom-up method).

A typical sequence of such operations is listed below:

- 1) Image segmentation: The areas of the image that represent the objects of interest are located and isolated from the background. A compact form of description is created for each of the objects in the scene. A common description is a connected chain of points on the outline, or the silhouette of the object.
- 2) Feature extraction : A set of features representative of the object of interest is extracted from the set of outline points. Some features relate to the shape of the outline, other features are used to calculate the position and the orientation of the object on the x-y plane.
- 3) Recognition : The set of measured features that relate to the shape of the outline, or the shape and the relative positions of internal features (such as holes), is compared to a similar set of prototype features, which have been obtained through a teaching process. During this process the objects are shown to the system at various positions and orientations. The system accumulates statistics of these features and creates a generalised prototype feature set (a model).

Fig. 2.3. illustrates the various steps in data reduction.

2.2.3. Major function blocks of a typical vision system

After the discussion of the functions of a typical vision system, we can now give a block diagram of the major function blocks of such a system. In the block diagram of Fig. 2.4. we have split our "black box" into two major function blocks:

- a) The image acquisition block, which contains an image sensor that converts light intensities to electrical signals, and a sensor interface, that converts the electrical signals to an array of binary numbers for subsequent digital processing.

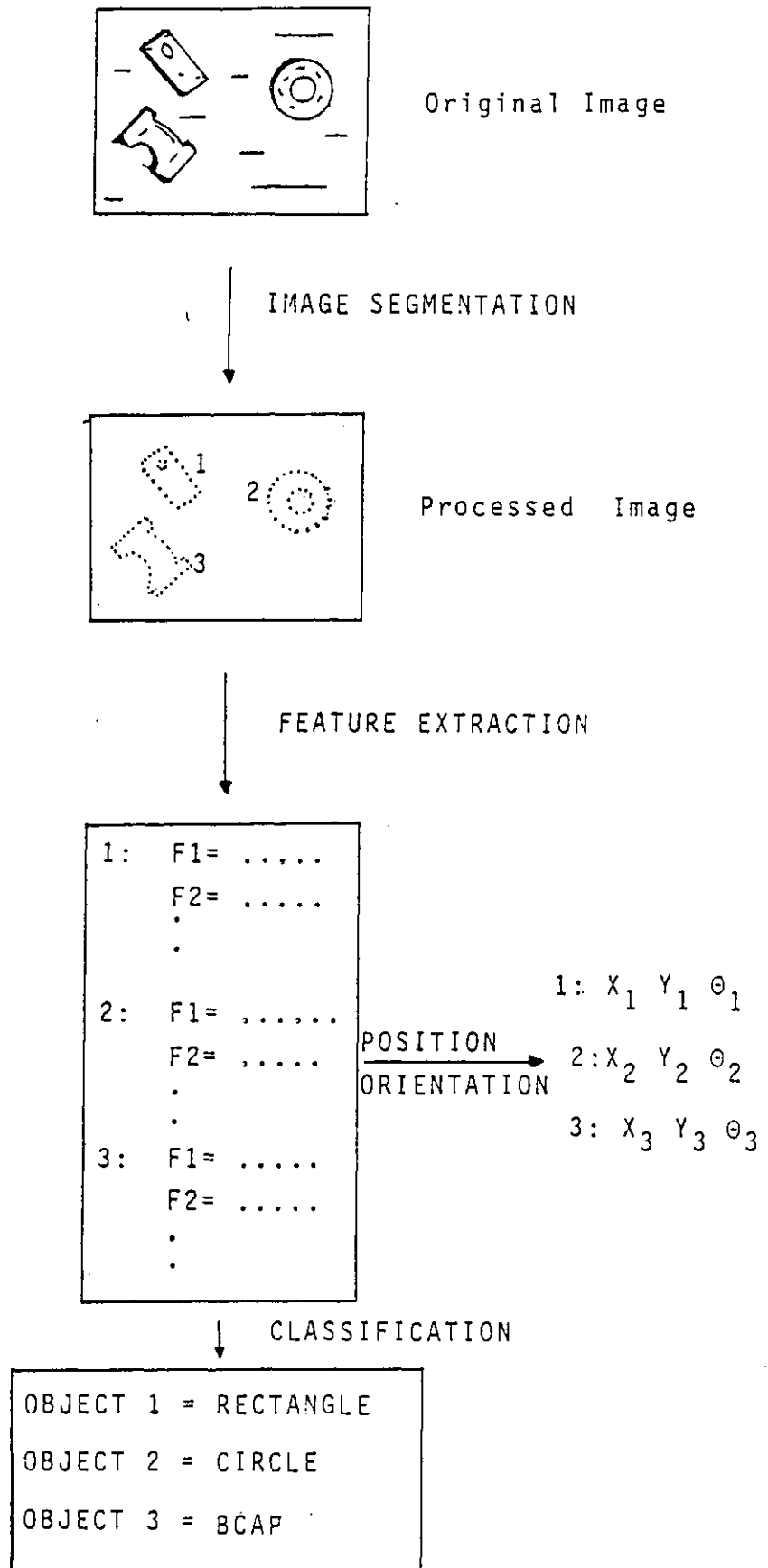
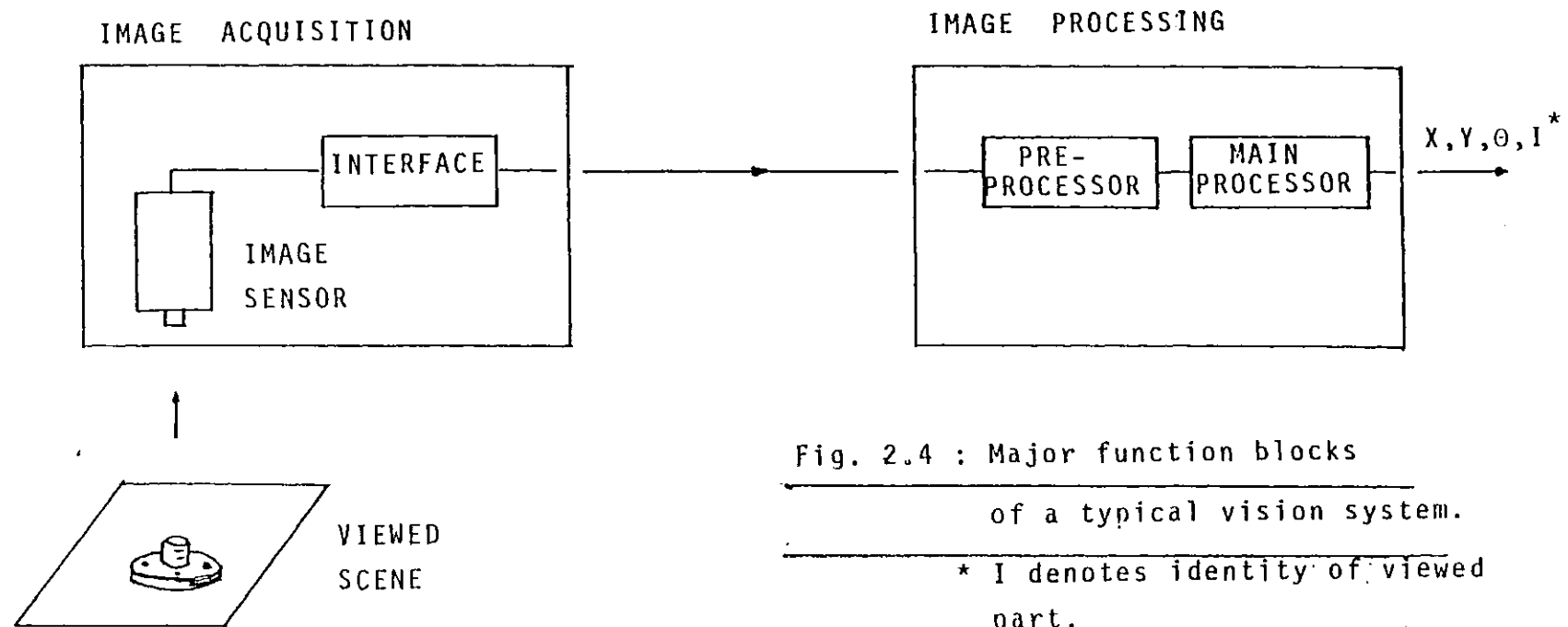


Fig. 2.3 : Basic steps of the bottom-up method of image description.



- b) The image processing block, which performs the digital operations on the image array, in order to extract the required informations.

Very often some of the initial processing stages include some kind of image enhancement, or filtering operations. A special purpose preprocessor is often assigned to perform these operations. The same term however is used sometimes for digital processors that perform other operations, such as coding or even image segmentation.

Hardware Components

A) **IMAGE SENSORS** : A variety of visual sensors have been used as inputs to industrial vision systems. The most popular and readily available low-cost device is an ordinary Vidicon camera, working in the raster scan mode.

The image is focused on a target area inside the camera with a system of lenses. The target can be thought of as a continuous matrix of elementary photosensitive capacitors charging proportionally to the incident light intensity. The target is scanned by a narrow electron beam that discharges the elementary capacitors producing a continuous stream of electrical current, which after amplification constitutes the video signal.

Normal video signals are analog voltage waveforms with values ranging from 0 V, corresponding to low values of light intensity (black areas), to 1 V, which corresponds to high values of light intensity (white areas). A series of synchronization pulses, which control the movement of the scanning beam, is mixed with the video signal. The whole image is scanned in two consecutive interlaced fields, i.e. two sets of horizontal lines with each line of one field positioned between two consecutive lines of the other field (see Fig. 2.5). A pair of interlaced fields constitutes a frame; each frame consists of 625 horizontal lines and is scanned at a rate of 25 frames per second. Low-cost vidicon cameras have a horizontal resolution of around 400 picture points along a line; high resolution cameras have an effective resolution of up to 1000 or more picture points.

Vidicon cameras have certain disadvantages, that make them undesirable for industrial applications. Most of them are rather bulky and fragile, and they have a rather short tube life. They are sensitive to electro-magnetic fields and they need constant readjustment for drift and aging.

Solid-state cameras : A major objective for the semiconductor industry the last few years was to produce a solid-state equivalent to the conventional electron-beam camera tube. Today a variety of such

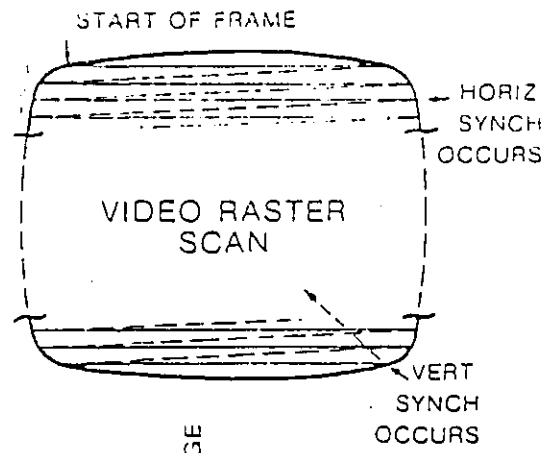


Fig. 2.5 : Interlaced scan mode of a normal CCTV Vidicon camera, and resulting video signal.

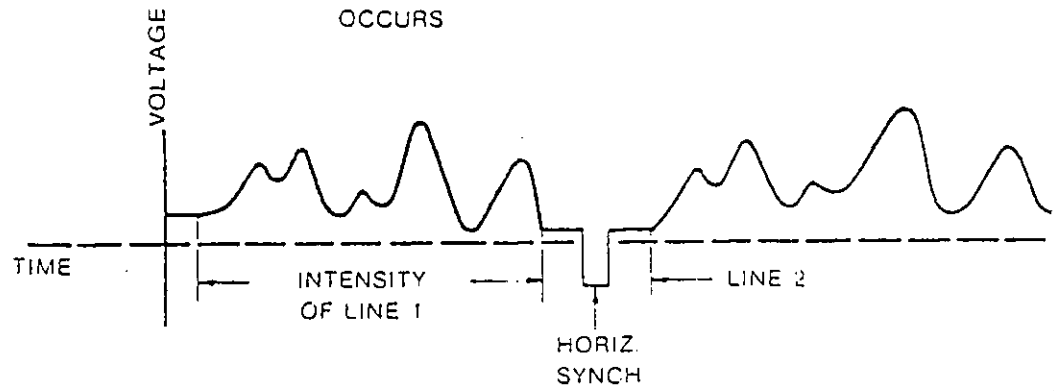
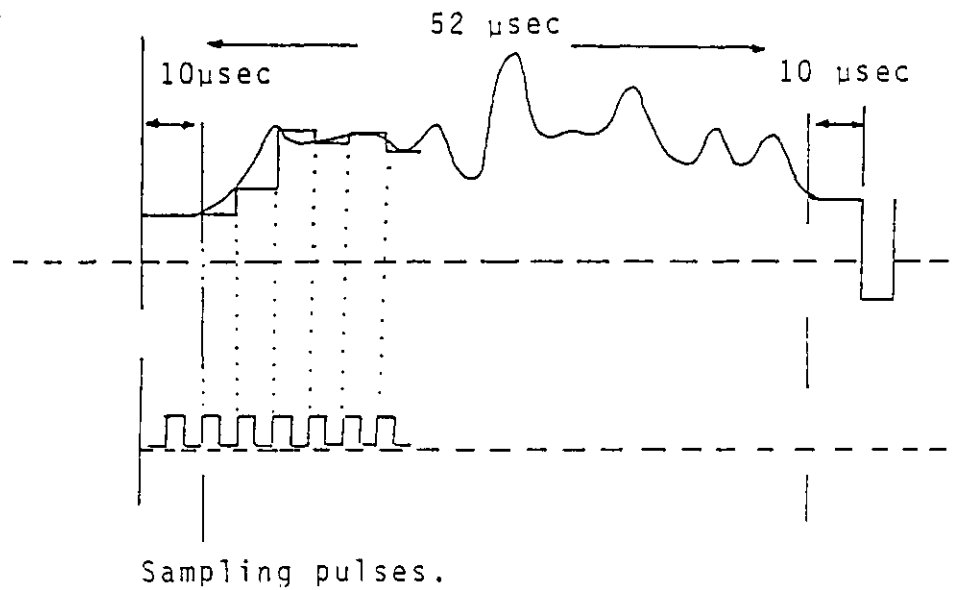


Fig. 2.6 : Sampling along a horizontal TV line.



devices exist, with their performance characteristics improving rapidly.

Solid-state imaging devices are manufactured using photodiodes or light-sensitive charge coupled devices (CCD). The photosensitive elements are discrete and they are arranged in linear arrays or matrices. Thousands of such tiny elements are precisely integrated on a small chip area together with the necessary clocking circuitry.

Linear arrays with resolutions of up to 2048 elements are particularly useful for moving line applications, in vision systems recognizing parts transferred by conveyor belts. The camera scans a horizontal line across the conveyor; the vertical scan is provided by the movement of the part itself. Two-dimensional array cameras with spatial resolutions matching those of ordinary vidicons (576 x 385), are already available.

Their main advantages over vidicons are small size, low power consumption, low operating voltages, high resistance to mechanical shock, vibration and high levels of radiation, high geometrical linearity, and the fact that some of them are purposely designed to be used in conjunction with digital processors, so they include facilities such as programmable scan modes and digital outputs.

Laser scanners have been used in some applications. Usually a system of rotating mirrors moves the laser beam in one direction; the scan in the other direction is provided by the movement of the object. Several photodetectors placed in various positions measure the reflected light. Systems employing laser scanners can operate at very high speeds, and they are mainly used to monitor the production of materials such as cloth or paper.

Image Sensor Interface - T.V. Digitizers

The function of the sensor interface is to convert the analog video signal to a matrix of binary numbers corresponding to the digital gray levels of the quantized picture elements (pixels).

To do that, the time scale of a horizontal scan line is divided into a number of time intervals and the video signal is sampled during each consecutive interval. The number of samples on a horizontal line corresponds to the horizontal resolution of the digital image. The samples thus acquired must then be converted to binary numbers. Referring to Fig. 2.6 we note that the total acquisition and conversion time for one pixel must not exceed 400nsecs for a line sampled at 128 points ($52\mu\text{s}/128$). Most of mini and microcomputers cannot accept and process data at such high rates. Some form of data buffering is therefore necessary. The simplest scheme is to sample only one pixel at a time at each consecutive horizontal line (i.e. at a rate of $60\mu\text{s}/\text{pixel}$). This is the principle of a column digitizer, which takes 20 msec (a field period) to complete a column across the image, using a relatively slow A/D converter.

For higher transfer rates a fast A/D converter is needed, which can follow the horizontal line rate, triggered by a clock circuit that is synchronized to a multiple (say 128 times) of the horizontal scan frequency. A few horizontal lines can be buffered at a time, or alternatively a complete dual-port fast video memory can be used, one port facing the digitizer, the other facing the computer. Fig. 2.7 shows a block diagram of such an arrangement. (Ref. 9)

The video signal is sampled and converted at time instances defined by the clock circuitry which is driven by the horizontal synchronization pulses of the TV camera. The same sync pulses drive two displacement counters (X and Y) and the vertical sync drives a field counter. The digital outputs of these counters are converted to the appropriate address numbers to access the video memory. The data input to this memory is the digital output of the A/D converter, which corresponds to the value of the current sample. A two-dimensional image matrix is thus created in the memory, which can be randomly accessed from the computer. TV digitizers of various configurations are commercially available, but their prices range from £ 2000 to £ 4000.

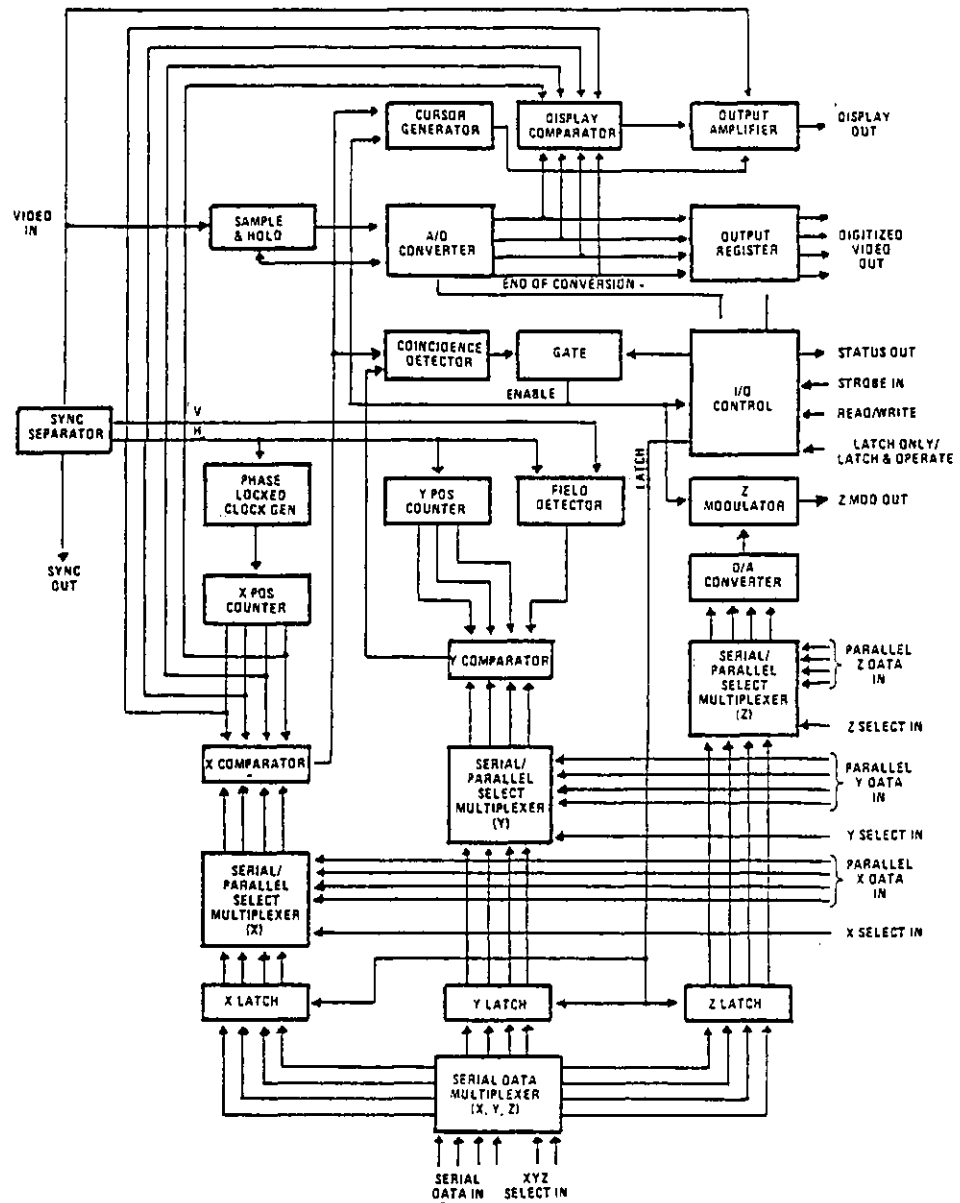


Fig. 2.7 : Block diagram of a fast TV digitiser
(Ref. 9)

DIGITAL IMAGE PROCESSORS : The heart of any industrial vision system is the image processor. Real-time digital image processing is a challenging and difficult area, because it involves very delicate trade-offs between accuracy, efficiency and speed.

Efficient processing under such circumstances involves the implementation of complicated algorithms with computational requirements that are beyond the capabilities of many general-purpose processors. For example, a very common set of algorithms are those that operate on small subimages scanning the image area with the purpose of reducing the amount of information. Implementation of such an algorithm on a moderately sized image of 128 x 128 picture elements, requires operation on 16384 subimages of say 3 x 3 pixels each. If we only consider the memory access times for such operations, under the most favourable conditions, we are talking about 4 to 6 operations per pixel, which results in approx. 50 operations (instructions) for each subimage, a total of 819200 instructions. Assuming an average execution time of 1 μ s per instruction, we end up with a total execution time of 0.8 seconds only for memory access. Of course improvements can be made at various levels, but still the most dramatic improvements can be expected from new technology.

With the existing technology today, probably the best compromise is a general-purpose minicomputer or an advanced microcomputer, or a combination of general purpose microprocessors and other special purpose processors.

In most of the industrial applications, cost and operational simplicity are the main factors favouring a general-purpose processor. Another important factor is the trend in the development of microprocessors and microcomputer systems.

New, faster, more powerfull and more versatile processors appear almost every year. Special purpose processors with architectures that are more suitable for vision applications than the classical von Neuman architectures, can now be implemented in integrated circuit form, using the new VLSI (Very Large Scale Integration) manufacturing

technology, which offers a 100 fold increase in the number of active elements per chip.

Most of the early industrial vision systems employed minicomputers, mainly of the DEC PDP-11 series. With the advent of the LSI-11 series of high performance 16-bit microcomputers that are descendants from the PDP-11 series, most of these systems, and in fact most of the subsequent vision systems, employed these microcomputers, because they offered an easy transfer of applications software that had been already tested on a minicomputer.

The main drawback of general-purpose processors is that they use a single Arithmetic Logic Unit (ALU) to perform simple arithmetic operations on binary numbers with a fixed number of bits (8-bits, 16-bits, 32-bits, etc.) and a fixed format. Larger systems employ special purpose ALU'S with optimised architectures, operating on either fixed-point or floating-point numbers of variable length.

Such special purpose ALU'S, which can be easily interfaced to microprocessors, are now available at relatively low prices.

Other solutions of the speed problem based on hardware improvements include:

- a) The use of bit-slice processors which achieve high execution speeds (in the order of 200ns per instruction) at the expense of higher cost and less flexibility, especially in the development stages (Ref. 10)
- b) The use of CCD processors that are basically analog devices controlled by a clock. They constitute a potentially viable alternative for many signal processing applications, but their capabilities are still limited (Ref. 11)
- c) The use of analog microprocessors that appeared lately (1979), promising digital processing of real-time analog signals in a programmable mode. Their processing rates are still within the audio range (up to 20 KHZ), but they offer a substantial

flexibility (Ref. 12) and it is highly probable that they will reach real time video processing rates within the next few years.

A different approach to real-time pattern recognition has been adopted by Prof. Aleksander (Ref. 13) which makes use of RAM (Random Access Memory) elements, instead of processing elements, and it is based on the comparison of acquired images to generalised sets of patterns, instead of describing them in terms of characteristic features.

2.2.4. Industrial Applications of Vision Systems - Review of Existing Systems

A thorough investigation of the industrial processes, where vision sensors could play an important role was reported by C.A. Rosen of Stanford Research Institute in a symposium which was held at the General Motors Research Laboratories in September 1978 (Ref.14)

According to this report, the potential application fields of vision systems are grouped in two broad classes.

The first class includes applications in which vision systems could be used to enhance the capabilities of industrial manipulators (robots) in manipulative tasks. In the second class of applications, vision systems can be used for inspection, increasing yield and improving safety.

Vision systems for sensor-controlled manipulation

A summary list of representative task areas within this class of applications is given in Fig. 2.8. Although the functions of a vision system are basically the same for all manipulative tasks (positioning, orienting, recognition), the conditions, and hence the requirements of each task area, differ considerably.

The following is a brief discussion of the problems involved in each task area, accompanied by examples of existing experimental

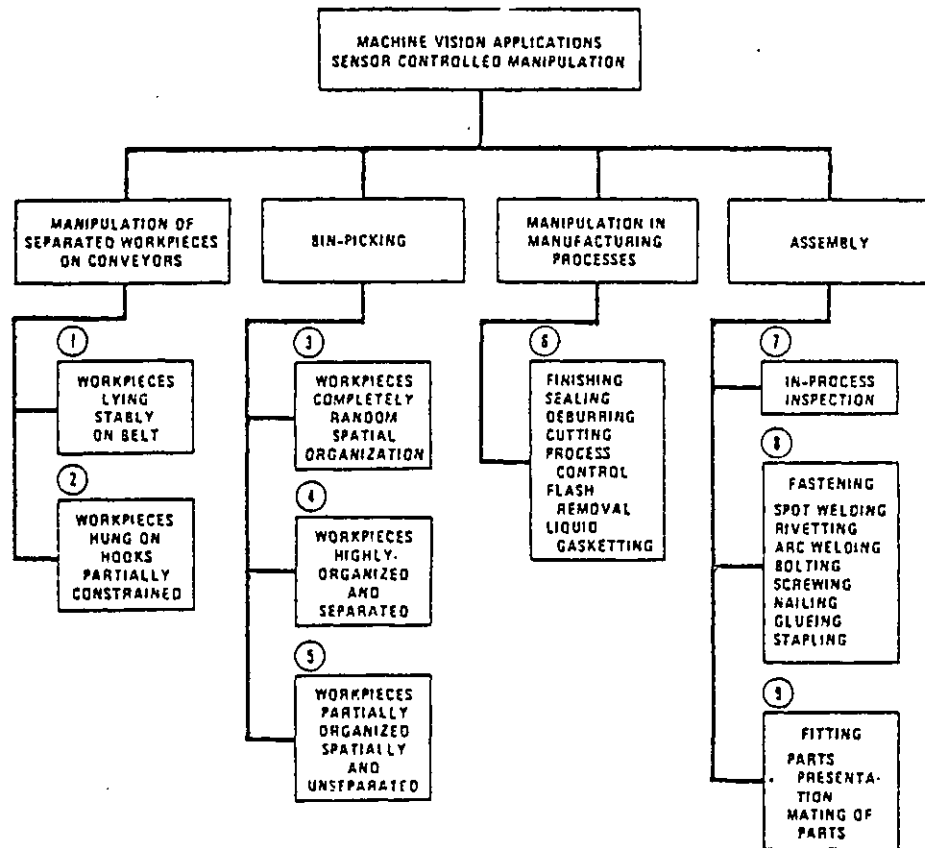


Fig. 2.8 : Machine vision applications - sensor controlled manipulation.

or production systems.

Manipulation of Separated Workpieces on Conveyors

In many manufacturing processes individually separated parts are being transported by means of systems that do not control part position or orientation (conveyors). Automated part-handling under these conditions is impossible.

The use of intricate feeding devices, on the other hand, can lead to uneconomical, unreliable and inflexible operation.

Vision systems offer a much more flexible solution, especially in batch production, and for frequent model changes. For parts lying stable on a belt conveyor, two dimensional images of the parts provide sufficient information about their position and their orientation, as well as about the features that specify one of the stable states in which they can be.

The degree of complexity of a vision system for this application will be decided by two factors:

- a) The degree of physical separation of the parts from each other.
Clearly separated, non-overlapping parts compose simple scenes with simpler image analysis requirements, than those imposed by scenes including overlapping or touching parts.
- b) The degree of visual separation of each individual part from the background, in other words the light contrast between the parts and the background (the conveyor).

The first problem can be solved by using simple mechanical means to separate the parts. Parts that are touching or overlapping are ignored by the vision system and recycled.

The contrast between the parts and the conveyor can be enhanced either by back lighting the conveyor, or by using structured lighting (projection of simple geometrical light patterns into the workpiece).

If the parts are being transported by an overhead chain conveyor, the requirements are even more stringent. Three-dimensional information will be needed to determine the position and the orientation of the parts in space, unless special supporting devices are used. A combination of structured lighting and range sensors can provide information on the third dimension.

A typical example of a system that recognizes and locates parts moving on a belt conveyor is the CONSIGHT, designed by S.W. HOLLAND, L. ROSSOL and M.R. WARD of General Motors (Ref.15)

The main feature of CONSIGHT is its clever lighting arrangement. Two light planes are projected at different angles on one narrow line of light across the belt surface. A solid-state line camera is focused on the projected line. As the object is moving under the camera, these two projections are displaced and the camera is looking at a broken line. The missing part corresponds to a line that scans the object area as the part is passing in front of the camera. A two dimensional binary image of the part is thus obtained, and the white-to-black (1 to 0) transitions define the outline of the part.

CONSIGHT hardware consisted of a DEC 11/34 minicomputer, a RETICON RL256C linear-array camera, a Stanford manipulator and a conveyor with a built -in encoder for position and speed measurements. The prototype production system operates at a cycle time of 5 seconds, at belt speeds of up to 20 cm/s.

A different system approach to the problem of handling separated parts has been adopted by P. Saraga and J.A. Weaver of Phillips Research Laboratories. (Ref.16). Their system consists of a simple 3-degrees of freedom manipulator of the pick-and place type, a rotating X-Y table, and a conveyor belt on top of the table. Parts arrive on the table at random positions and orientations. Two TV cameras, one over the table and one at the side, send information to a minicomputer which recognizes the parts, calculates their position and orientation and sends the appropriate commands to the table to rotate and translate a part, so that it is presented to the manipulator at a fixed posi-

tion and orientation angle. High contrast is achieved by backlighting provided by light within the conveyor. The operation cycle is 5 secs of which at least 3 secs are taken by the mechanical movements of the arm and the table. Although the system uses 2 cameras, it does not possess true three dimensional capabilities.

Bin-Picking

Automatic handling of parts in containers can be very difficult, except when they are highly organized.

In all other cases a vision system would have to cope with poor contrast, partial views of parts, multiple reflections, variable geometric transformations of shapes etc. Although the subject has already attracted a lot of research interest, there are still those who would argue about the necessity of such an effort in the industrial applications area, mainly because in an automated factory provisions can be made so that this situation (randomly jumbled parts) would very rarely, if ever, occur.

The work of W.A. Perkins at General Motors Research Laboratories (Ref. 17) is an excellent example of highly sophisticated image analysis techniques devised primarily for the solution of the problem of overlapping parts. It is worth noting however, that the proposed method was implemented on a IBM 370/168 mainframe computer for the recognition of overlapping steering knuckles on a conveyor belt, and average execution times for a pile of 5-6 workpieces were in the range of 20 seconds.

A lateral solution of the bin-picking problem is to divide it into two easier stages. First acquire a part in random from the bin, and then try to recognize it, either by depositing it randomly on a flat surface, or by showing it to the vision system while it is still being held by the robot. This second approach has been tried in the University of Rhode Island (Ref. 18) Success rates ranged from 60 to 100 percent, depending on the nature of the workpiece, and the estimated time for part acquisition and transfer to one of the intermediate goal

sites was about 20 seconds on average.

Vision-Controlled Manipulation in Manufacturing Processes

A large number of important manufacturing processes (see Fig. 2.8) cannot be automated without the use of specialized work-holding and tool-guiding devices (jigs). Jigs are generally expensive and therefore not justified unless a sufficiently large volume exists. Flexible automation systems can be programmed to adapt to varying workpiece shapes, but they still have to rely on a fixed presentation of workpieces.

A typical and widespread application of robots is for automated paint spraying. Most of the existing paint-spraying robots can be programmed to follow exactly the same sequence of operations that a human worker performs, simply by memorising all the steps during a guide-by-hand teaching process. During operation they will repeat this sequence exactly, even if, for any reason, the workpiece is missing, or is slightly misplaced.

Since the majority of spray-painting lines do not have any kind of specialized control on the position and orientation of the workpieces, there is clearly a need for non-contact sensors to provide information for adaptive corrections.

Simple photocell detectors could be used to sense the presence or absence of the surface to be painted, but they were found unsatisfactory in practice, the reason being that signal delays, irregular surfaces and contamination of the sensors complicate the design.

The outline of a simple vision system, incorporated on a conventional reciprocating spray gun mechanism, was presented by E. Johnston of Carrier Drysys Ltd., in the 1st International Conference on Robot Vision and Sensory Controls (Ref. 19)

The function of the system is to provide visual information so that the spray gun is opened when there is a product to be painted and closed when there is not.

There are other manufacturing processes that have common requirements on vision systems, like sealing, deburring, removing the metal that squeezes out of the impression between the die blocks in die casting (flash removal), and liquid gasketing. These applications require following edges, seams or other features of the workpiece, with the ability to cope with slight path variations. For most of these applications two-dimensional image analysis is sufficient, but in the general case the addition of range information would be useful.

Vision-Controlled Manipulation in Assembly

Most of the research and development activities in the field of flexible automation have been directed the last few years towards applications related to batch-assembly processes.

Vision systems have an important role in automated assembly. There are 3 major groups of functions that they can perform in this application area:

- a) They can provide information for parts presentation and mating in fitting and insertion processes.
Information on the identity of parts and their relative position and orientation is necessary, so that they can be sequentially presented and fitted to the rest of the assembly at each workstation. Similar information is needed for the correct alignments of parts in insertion operations such as bolting and rivetting.
- b) They can provide visual feedback for path control, which is necessary in many fastening operations (spot and arc-welding, screwing, nailing, stapling and gluing), in situations where the location and the trajectory of the line of fastening is variable.
- c) They can perform in-process visual inspection after the completion of each assembly step .

In an ambitious project partly funded by the National Science Foundation of the USA, a team of researchers from Westinghouse Electric Corp. tried to build a complete Adaptable Programmable Assembly System (APAS) using techniques developed at Universities and Research Institutes in the US the last few years. APAS is a combination of robots, programmable fixtures, universal end effectors and tools, vision and other sensory system, backed by a network of minicomputer and microcomputers. (Ref. 20)

VISION SYSTEMS FOR INSPECTION

Most of the existing industrial vision systems are designed to work together with an automatic manipulator in applications involving some kind of parts handling.

The same principles, governing the design of these systems, however, can be applied for many inspection orientated applications. The main difficulty here is the wide variability of inspection processes, which makes the design of a general-purpose visual inspection system conceptually very complex, and therefore economically non-justifiable for most of the applications. On the other hand, special purpose systems have been designed and implemented successfully in some areas, especially in the electronics industry.

Visual inspection can be either highly quantitative (measurement of critical dimensions), or semi-quantitative, primarily qualitative (label reading, sorting, integrity and completeness inspection, inspection of surface finish properties, inspection for safety and monitoring).

A broad classification of industrial inspection processes that could be automated by the utilisation of vision sensors appears in Fig. 2.9

In many applications, other sensory techniques may be less expensive, but also less flexible.

For instance, optical character recognition (OCR) devices and bar-code scanners are already being used extensively for label and bar-code reading, but can only operate within narrow limits in the position and the orientation of the products bearing the labels or bar-codes.

There are different requirements for each class of inspection applications. High-resolution and optimal optical coupling between sensor and workpiece is essential in most quantitative inspection applications.

The existence of an appropriate objective and quantifiable visual test for each particular part is a prerequisite for qualitative

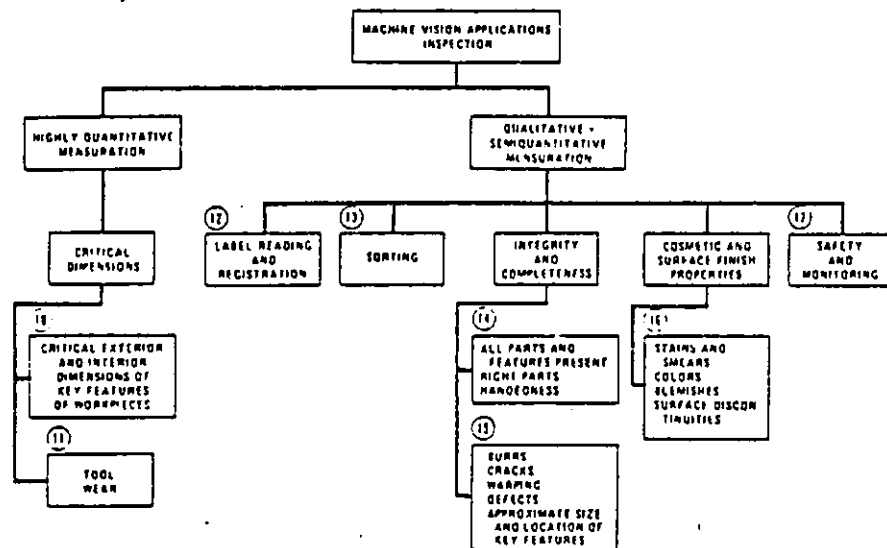


Fig. 2.9 : Machine vision applications - Inspection.

inspection. Flaws or defects must be definable in terms of shape and size criteria. In most of the cases it is practically impossible to create a list of all good patterns, in terms of combinations of such criteria. An inherent error probability is therefore always present.

Perhaps the most decisive general factor in all inspection applications, is the amount of irrelevant image variability that exists in the inspected scene. This can be sometimes reduced by:

- a) Controlling the position and orientation of the workpieces under inspection, in an earlier manipulative stage (see p.28)
- b) Employing specialized lighting techniques. For applications such as inspection for surface finishing properties, the choice of appropriate spectral bands and filters, and the choice of the directions of incident light is essential. (Ref. 21)
- c) Employing special signal processing techniques, such as digital filtering, programmable gray-level windowing, etc.

Integrated circuit (IC) manufacture, a highly automated, computer oriented industry was one of the first to employ vision systems for inspection purposes. SIGHT-1 is the name that was given to such a system, which was developed by the Electronics Division of General Motors in 1975. The system was used for quality inspection and automatic alignment of IC chips, and operated on low-resolution (50 x 50) images digitized at 16 levels of intensity. (Ref. 22)

A similar vision system was developed by Hittachi in 1975 (Ref. 23) for transistor wire-bonding inspection. This system is a typical example of the processing speeds that can be achieved through special-purpose hardwired logic at the expense of limited scope (lack of programmability).

We conclude this chapter with a general description of two vision systems of particular interest.

A) The SRI Vision-Module

This system was developed by the Stanford Research Institute in USA as a part of substantial research and development in the field of flexible automation, dated as far back as 1973 (Ref. 24)

The main attraction of the SRI system is the modularity and the generality of its design, which makes it easily applicable in a wide range of applications, as a series of vision-library subroutines is stored in its memory.

It is mainly intended for parts handling applications, but it can also be used for some semi-quantitative inspection applications, for assembly operations and for limited visual servoing.

The Vision Module consists of three major components, a General-Electric solid-state camera with 128 x 128 elements, a DEC LSI-11 16-bit microcomputer and an interface processor between the camera and the computer.

The preprocessor interface performs analog to binary conversion of the TV signal, stores the resulting 128 by 128-bit binary image in a 16K-bit frame buffer and encodes the image in an appropriate coding scheme (run-length coding).

The LSI-11 microcomputer has a 28K 16-bit memory which stores all the vision library subroutines and some of the application programs.

The software design of the system is based on a series of standard independent modules that perform the various sequential algorithms for image segmentation, feature extraction and feature comparison. Various special purpose routines are also available for other applications, including fast routines that locate features within specified windows in an image for high speed visual servoing, and other routines that evaluate relationships of local features to one another for assembly and inspection applications. Total execution time for a program that identifies and locates separated workpieces

is around 1.4 seconds.

8) The Brown-Boveri Vision System

This system was developed in the Brown Boveri Research Centre in Switzerland, and it was presented at the 8th International Symposium on Industrial Robots in 1978. It is a general-purpose modular system with applications in parts handling, adaptive visual control, and inspection based on global properties (Ref.25)

The main attraction of this system is its high processing speed, which is a result of the utilisation of a very fast hardware processor.

Its main components are:

- a) A video-processor which performs the analog to binary conversion of the TV signal (at a 256 by 256, maximum 512 by 512 spatial resolution).
- b) Two microprocessors, a DEC LSI-11, which is responsible for system management, communication with the peripherals and all high-level arithmetic and logical operations, and a specially designed fast bit-slice processor, which performs most of the standardized, time consuming, low-level processing tasks.
- c) A normal vidicon TV camera.
- d) Two separate image memories for fast superposition of two images which is used for some inspection applications.

The procedure followed for object identification is the typical procedure of image segmentation (using an adaptive frame), feature extraction and feature comparison. Various orientation independent features are extracted for each object with the use of a series of circular scans, centered at the centre of gravity of the object area.

The system has been used in various applications, including unloading pallets, providing limited visual feedback for the control

of the movement of a robot, reading coded labels, and performing global visual inspection (an overall estimate of resemblance of an object with a reference pattern).

Overall processing time for one object range between 30 and 200 msecs, depending on the required resolution and the complexity of the recognition procedure.

Conclusion : Vision systems offer the most effective solution compared to all other sensory devices in a wide range of applications. Their main advantages over other sensors include:

- a) Quick simultaneous acquisition of all the necessary information.
- b) Straightforward human interpretation of the acquired and processed information, and above all
- c) Increased flexibility and programmability.

The same image can be analysed in numerous ways at a widely varying degree of complexity, in order to reveal exactly the kind of information which is needed for each particular application.

A general-purpose modular vision system can be built to cover a wide range of applications, simply by changing software modules.

The price that one has to pay for such flexibility is high as expected. As the complexity of the processing techniques involved increases, hence the investment of human knowledge. Research in fields like image processing, pattern recognition, and artificial intelligence constitutes the background for the design of successful industrial vision systems.

CHAPTER 3 : DESIGN PHILOSOPHY

3.1. General Introduction

The development of this system is part of a major project involving the development of a number of constituent parts of a flexible automation centre.

The centre consists of an integrated CAD (Computer Aided Design)-CAM (Computer Aided Manufacturing) system, a six-axis sophisticated robot, and a flexible machining centre.

The main tasks were to develop an operating system for image analysis, based on an 8-bit microcomputer, and to devise the techniques and the algorithms for a stand-alone, low cost industrial vision system for real time recognition of mechanical parts and determination of their position and orientation on a plane.

At the beginning of the project a general feasibility study showed that an acceptable compromise between speed and effectiveness on the one side, and cost on the other, could be achieved, provided that:

- a) Special importance was given to the selection of the method and the algorithms to conform with the limited capabilities of an 8-bit microcomputer.
- b) All the software for real-time operations was written in low-level language (assembly).

In the previous chapter it was shown that the main problems encountered in the design of such a system relate to the wide variability of the data, and the amount of spurious or irrelevant information inherent in the information source (the image).

An industrial vision system should be able to cope with:

- a) A wide variability of shapes.
- b) A considerable 'distortion' of these shapes due to various geometrical transformations (translation, rotation, scaling, perspective transformation).
- c) A non-linear distortion due to the spatial quantization of the image.
- d) Various 'noise' sources, ranging from undesirable lighting effects (shadows, reflections, non uniform illuminations) to electronic noise generated within the preprocessing devices, and processing noise due to the accuracy limitations of the processor itself.

3.2. System Specifications

Bearing in mind the above mentioned problems, the speed limitations and the main objective of low cost, both in development and for the end product, the following specifications were set:

- a) The system must operate on two dimensional information. Three-dimensional relations either between the camera and the objects of the scene, or among the objects themselves are not considered.

This imposes a limitation on the range of objects that can be handled successfully, namely objects that have a finite number of possible stable positions on a plane and also have a low height-to-width ratio, so that they can be represented reliably by a finite set of two dimensional silhouettes. Flat objects are ideal.

- b) The system must operate satisfactorily under normal lighting conditions. Structured lighting should not be necessary. The system should be able to operate on objects with varying contrasts in the same scene, provided that a reasonably good contrast exists between the objects and the background.

c) The objects should be adequately distinguishable from their silhouettes and their internal features (holes). Objects with very similar outlines demand extra information based on local templates.

A provision should be made for this kind of object sets, but the overall system operation should be tuned to distinguish among a wide variety of essentially dissimilar shapes.

d) No special provisions should be made for touching, overlapping or otherwise partially occluded objects. The processing requirements for the analysis of scenes including partial views of objects can only be met at high cost, both in terms of processing resources and of speed. For applications involving objects transported by conveyors or otherwise lying on a plane, simple mechanical devices can separate them efficiently. Objects that are not thereby separated should be recycled.

e) The system should be designed to cover primarily application areas involving sorting of parts according to their shapes (sorting and assembly operations). The basic structure of the system should however allow future expansion for inspection applications.

f) The speed of operation requirements for the basic recognition routine, excluding image acquisition time (which depends on the digitiser) was set in the range of a few seconds (5-10 sec).

g) Last, but not least, the cost of the prototype system, excluding software development costs and special devices used for the initial setup of the system, was set at less than £5000, which probably means that the production cost could be somewhere in the range of £2000 to £3000 per unit, or even less than this, if we take into consideration the downward trend in the prices of microcomputer hardware.

3.3. The Basic Decisions

Obviously the first decision that had to be made related to the choice of hardware.

A high-resolution vidicon CCTV camera allowing external control, was already available. The purchase of a solid-state lightweight camera was considered, but was not found necessary for the development of the system.

The same applies to the TV digitiser. An existing design for a column digitiser was modified to meet the interfacing specifications of the microcomputer system, the digitiser was built, and the necessary interface software was written in assembly language. The choice of the microcomputer was primarily based on cost-efficiency considerations and availability. The system that was chosen is a Motorola 6809 based 8-bit microcomputer development system by South West Technical Corporation.

The 6809 microprocessor, essentially an 8-bit processor, offers some 16-bit processing capabilities and can be readily interfaced to a number of important components of the 6800 family, to form an integrated microcomputer system. The price of such a system incorporating a faster version of the 6809 microprocessor (6809 B, with a 2MHZ cycle), 64K of RAM, a floppy-disk controller, parallel and serial interface boards and a ROM monitor was £1800.

Obviously the cost of a stand alone processor, with 64K of RAM and a parallel interface to the digitiser will be less than this figure.

Our opinion was that a 6809 based system is a cost effective compromise between 8-bit and 16-bit microprocessors, which can just about provide the processing power required within the specified operating speed.

In order to enhance the computational capabilities of the system, a special purpose floating-point processor (AM9511) was used. A board

incorporating this processor and supporting circuitry was built at the College. The use of this processor would only add a minimal amount to the total price of a stand alone system, offering in the same time considerable computational enhancement.

A more detailed description of the hardware can be found in Chapter 6.

Having chosen the hardware, the next step was to decide about the general outlines of the database and the software package.

Two important decisions had to be made at this stage. The first decision was about the digital representation of the image. More specifically the question was whether we should use a binary or a gray-level representation. By using only two intensity levels after thresholding, we can achieve a considerable data reduction. Simpler hardware is required for digitization and simpler techniques can be used for image segmentation. The availability of gray-level information, on the other hand, permits the application of a great variety of image processing techniques and renders the system capable of expansion to other more demanding applications, or applications of different nature (quality inspection, 3-D scene analysis, bin sorting). Considering the downward trend in the prices of RAMs it seems that such an approach is justifiable. After all, one can always work with binary levels at any stage once a gray level representation is available, while the opposite is not possible. The decision therefore was to represent each picture element with a 4-bit binary number, accommodating a range of 16 possible gray levels.

The second question was whether to buffer a whole picture frame in memory, or only part of it at a time (e.g. 2 or 3 consecutive horizontal lines). For a simple binary system the presence of the whole digital image in memory is not necessary, since the connectivity information required for segmentation can be acquired on the 'fly' from the consecutive lines. Buffering the whole picture frame, however, offers greater flexibility in the selection of image segmentation algorithms. It also permits a future hardware implementation, employing

a number of processors operating on the whole image or on different parts of it, in any serial-parallel combination, by sharing the same image memory.

The potentiality of such an approach is further enhanced by the introduction of a new concept of a mirror image memory, that is discussed below. A possible classification of vision systems could be based on the balance between processing capabilities and memory requirements.

Some systems rely totally on fast processors, that operate on the incoming image data on a sequential purely algorithmic basis. Other systems make extensive use of memory-orientated techniques (Ref. 13)

The basic philosophy of our system is a combination of both techniques. The whole image is buffered in the user's memory. Each pixel is represented by an 8-bit word. The lower 4-bit part (lower nibble) represents the gray level of the pixel. The upper 4-bit part is reserved for the accomodation of a range of 16 code numbers that are assigned to some pixels during processing.

The idea behing the use of this seemingly redundant memory space is to 'paint' certain areas of the image with a set of code numbers, thus effectively creating a mirror image array that contains information obtained during the processing of the original image.

We think that this concept can be applied for many purposes in a variety of ways, offering substantial flexibility, as a form of data base organisation, that is particularly convenient for image analysis.

It appears also to be an attractive solution for the communication between processors in a multi-processor environment. Each processor leaves a meaningfull trace during its operation on the common image memory, which in the same time retains in a compact way the original information and the spatial relationships inherent in the original image.

This concept has not been exploited to our knowledge in anything more than very trivial ways (such as complementing binary levels).

Having established the form of organisation of the data-base, the next step was to choose a suitable segmentation technique and also to decide about the representation of shapes (shape description).

A contour follower based on gradient information was chosen as a suitable way of isolating the objects in the scene from the background, mainly because of its speed and the convenience of the resulting numerical representation of the outlines (chain encoding).

The description of the shape of the outlines is based on a set of compound features, i.e. features that result from a combination of informative primitive features of the outline (perimeter, area, etc).

The compound features were defined on a heuristic basis and are invariant to geometrical transformations such as translation, rotation and scaling. The orientation of the objects is specified in terms of the directions of the principal axes of the contour. Finally a new representation was introduced for internal features, such as holes. A set of holes is treated as a unified pattern which is described with a set of global parameters. These parameters constitute a subset of the overall feature vector.

Classification of the objects in the scene is achieved by comparison of the measured feature vector with a set of models representing the objects of interest.

The models are created during training, whereby the same object is shown in various positions, orientations and magnifications.

CHAPTER 4 : IMAGE ANALYSIS and DESCRIPTION

4.1. Basic Elements

4.1.1. Introduction

Having established all the necessary background information and the outline of the proposed method, we now proceed to the detailed description of the system, its structure and its constituents. This chapter gives an analysis of the underlying concepts and the algorithms that form the basis of the system.

The bottom-up method of description was adopted, in other words a sequence of information reduction steps is applied, each step resulting in a higher, more compact level of representation. The principles of this method were given in Chapter 3.

4.1.2. Data Structure - Digital representation of picture elements

A typical image array $R(x,\phi)$ consists of 128 by 96 pixels quantized at 16 intensity levels. The origin of the image coordinate system (top left of the TV image) corresponds to the first entry of the image array $P(0,0)$.

Each point of the image array is represented by an 8-bit binary word. The 4-bit least significant (LS) part corresponds to the digitized intensity level $I(x,\phi)$ of the image at the specified point (x,ϕ) .

The 4-bit most significant part (MS) is reserved as a trace field accommodating up to 16 different code numbers. The image array can thus

be viewed as a digital representation of the original image, superimposed by a 'mirror' image, which reflects the traces left behind by the various processing stages (see Fig. 4.1 and 4.2).

A detailed description of the significance of each code number, representing a post-processing trace is given in Fig. 4.3. Ten code numbers are used for inter-process identification and communication purposes (informative traces). The remaining 5 code numbers (1,2,3, 6,7) are reserved either for display purposes (display traces), or as spare informative traces for future use. The use of special characters to display the different codes proved a valuable development and debugging tool, since it offers a straightforward way of examining the effect of an operation on the image array.

However, the main significance of this coding scheme is that the image array is used not only as the source of raw information, but also as an intermediate buffer between the various processing stages. To illustrate the potentiality of this representation, let us consider the situation whereby the outline of an already traced object is chain-coded, in other words each outline point is assigned an integer from 0 to 7 representing one of the 8 possible links to the next outline point in a quadrupled grid

The resulting chain list could be saved in a buffer in the system memory with a pointer to the start point of the chain. Instead of using a separate buffer for the chain list, by marking all the outline points in the image array with code numbers that include the chain links, we can create a more compact and efficient source of communication between the processing stages. At any stage during the processing of the image, the history of any point can be easily traced. If the same outline is intercepted again, it will be a very easy task to avoid retracing it, or to retrace it very fast if the purpose is to extract more information out of it.

This scheme should also provide a unique communication buffer in a hardware implementation involving a number of processors that are assigned parts of the whole process, or parts of the image area in any serial or parallel combination.

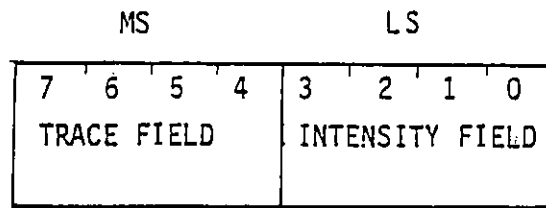


Fig. 4.1 : Digital representation of a pixel.

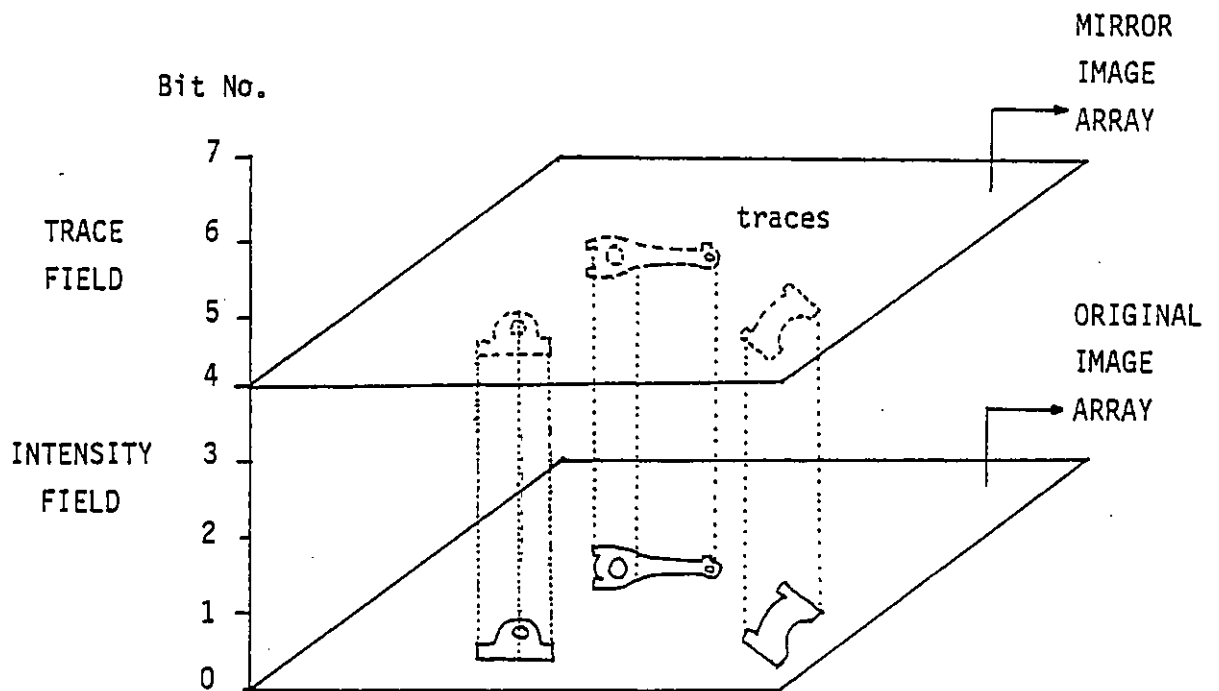


Fig. 4.2 : Original and Mirror Image relationship.

Code Number	Meaning
0 (0000)	'Clean' pixel - No trace
1,2,3 (00XX)	Reserved for display purposes.
4 (0100)	Point of a closed and processed (sealed) outline, or Point of part of an outline, result of an unsuccessful following attempt, not worthy of a second trial
5 (0101)	Point of part of an outline, result of an unsuccessful following attempt, due to reasons that justify another trial
6,7 (011X)	Reserved for display purposes
8-15 (1XXX)	Point of an outline that is currently being followed, or not yet sealed. Bit 7 signifies above meaning, Bits 6,5 and 4 represent chain link of the point (Chain-code numbers 0 to 7 - 3 bits)

Fig. 4.3 - List of trace codes with their corresponding meanings.

4.1.3. The problem of Image Segmentation

The problem of image segmentation, more specifically the isolation of the areas of interest (objects) from the background, is fundamental in any scene analysis procedure that follows the bottom-up method of description, since it is the first major step in the information reduction process. The effect of an incorrect determination of the object boundaries can be viewed as an additional noise source, introduced at the first stage of a serial process, thus resulting in a severe impairment of the performance of the whole system. On the other hand, this first stage is responsible for the highest proportion of the total information reduction, operating consequently on large amounts of data.

Considerable effort has already been devoted to this problem in two major directions; one involves applying intensity thresholding techniques, followed by run-length coding of the image and subsequent connectivity analysis of the 0 to 1 transitions (binary techniques) the other involves detection of edges, followed by edge linkage and possibly chain-coding, to form closed contours corresponding to the outlines of the objects (gray-level techniques). The last method implies in most of the cases the use of gray-level information and is applied in two steps.

First the whole image area is scanned by an edge detection operator, then an edge linkage algorithm attempts a selection of the most prominent edges and a grouping of those edges in closed contours. Edge linkage is based on the continuity assumption for the outline of an object expressed in terms of the edge intensities, their relative positions and their directions.

An important factor in all gradient-based systems is the selection of the edge detection operator. The simplest model of an edge is a step discontinuity in the image function, perhaps obscured by noise, reflectance variability, and blurring. For sharp images with little noise, the edge is disclosed by a step change in the image function, i.e. by large derivative

values. By replacing the first derivative (the gradient) by the finite differences, one can obtain discrete analogs of the gradient. At higher noise levels, instead of working very locally one must difference some kind of weighted average taken over windows on either side of the edge. The choice of the window size however presents a problem (Ref. 26)

A good survey of edge detection techniques can be found in Ref. 27

The selection of both the edge detection technique and the edge linkage algorithm was considered critical timewise, therefore some compromises had to be made. A simple gradient operator (Sobel's operator) applied on a 'basic cell' of 3 by 3 pixels was used. Edge detection is not performed on the whole image area, but is carried out in a search-for-a-starting point mode, followed by a contour follower.

Various search patterns were considered (circular search, radial search, rectangular search, etc.). The objective was to find an efficient search pattern that will locate all objects in the scene in the minimum time. A raster-like search was finally chosen, because it was thought to be more efficient for the localization of objects moving on conveyors. Searching in the image memory along horizontal lines is the fastest way, involving minimum computation of the addresses of search points.

The second stage in objects isolation, the contour following stage is described in details later in this chapter. A few general remarks will be made here. The first concerns the choice of the gradient operator and the size of the related window (basic cell). Larger windows and weighted averages taken over windows around the edge points were tried, but did not contribute sufficiently to the resolution of difficult cases (corners, shadows, etc.) to justify the time penalty that they impose.

The use of a more complicated gradient operator (for example Hueckel's operator, Ref.28) was excluded due to the same reason.

The choice of contour following, i.e. simultaneous edge detection and edge linkage was also mainly influenced by time considerations. Nevertheless, the contour following algorithm was given considerable attention to improve its efficiency.

4.1.4. The Raster-Scan Search

The image area is scanned in a raster-like mode for edges suggesting the presence of an object.

The search routine rests at the top of the hierarchy of the system routines, in other words it constitutes the main operating loop. A diagram of the search pattern is given in Fig. 4.4.

The scan starts at the middle-left of the picture and proceeds from left to right at fixed two-unit steps. Before each new step, a tentative movement is attempted to check whether the new point lies within the image area. A gradient test (Sobel's algorithm) is then applied on a 3 by 3 window centered at the new point.

If the value of the gradient is larger than a predefined gradient threshold (usually a minimum of 2 or 3), the new pixel is assumed to be part of the outline of an object.

The trace fields of the 9 pixels of the basic cell are obtained and compared to the 3 possible outline 'colours'.

This test reveals the identity of the encountered outline, in other words it finds out whether it is a new outline, an already found and closed outline, an unfinished outline that is worth retracing, or an unfinished outline that is not worthy of a second trial.

There are 4 reasons that may have caused a previous failure to complete the closure of an outline:

1. Only part of the object is shown in the picture. Its outline cannot be closed within the picture area.
2. The object is touching another object, or two or more objects, overlap.
3. A loop condition has been encountered during a previous following

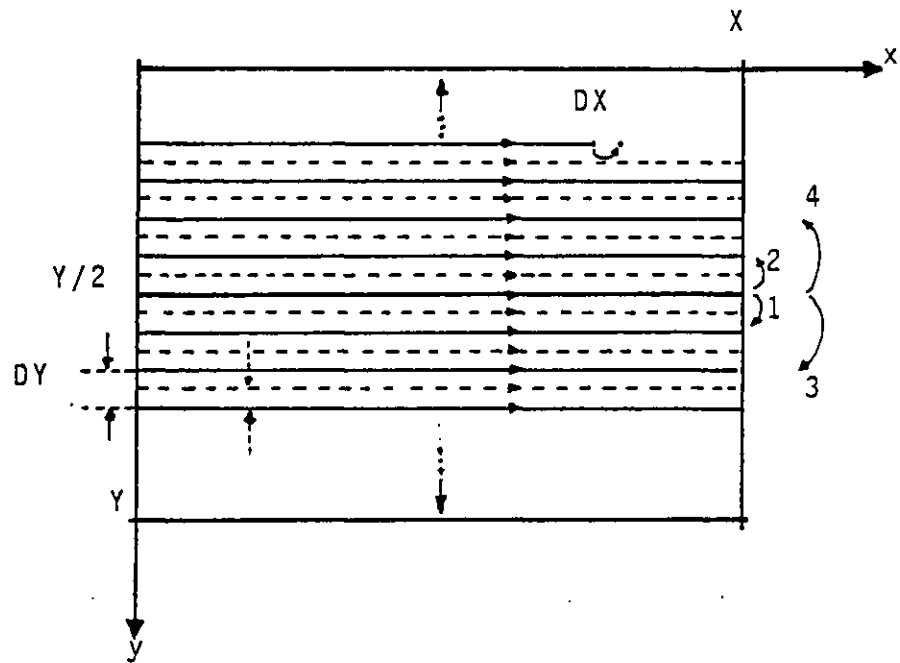


Fig. 4.4 : Raster-scan search
for edge points.

attempt.

4. Following has stopped because of low-contrast (local discontinuity of the reflectance difference between the object and the background along the outline, due to shadows, local blurring or any other kind of 'difficulty').

If the intercepted outline bears a trace signature signifying any of the two first reasons of a previous failure, a new attempt will not stand a better chance of succeeding.

However, if a previous following attempt has failed due to the third or fourth reason, a new attempt from a different starting point may prove successful, due to the nature of the following algorithm.

Having established the previous state of the candidate starting point, a decision can now be made whether to call the contour follower, or ignore this edge point and continue the search.

In any case, the search will continue to the right of the starting point until the right edge of the picture is met. The search bug is then moved back to the start of a new search line and the same process is repeated.

Search is carried out along equally spaced horizontal lines. The spacing among the search lines is defined by the operator, and depends on the size of the smaller possible projection of the image of any of the objects of the scene on the ϕ -axis.

New search lines are introduced in an alternating sequence of increasing positive and negative distances (above or below the middle line), until the upper or lower edge of the picture is reached or exceeded. The same pattern is repeated at a closer spacing (by halving the search distance) in a second pass.

A similar search pattern was used by R.S. Ledley (Ref. 29)

4.2. CONTOUR FOLLOWING (Edge Following)

4.2.1. Introduction

Contour following involves tracing out the boundary between a figure and its background. The contour following approach to image segmentation of binary images is commonly called bug following and it has been extensively applied for OCR (Optical Character Recognition) applications. A typical example is the contour analysis technique used by Grenias et al. (Ref. 30) for the recognition of handwritten numerals.

The same concept can be easily extended to gray scale imagery by using gradient information, so that the bug follower forms edges as it moves. Some algorithms accomplish this by combining a gradient operator with a contour follower. The gradient operator finds the local direction of the most prominent edge near a contour point. A small step in this direction is assumed to result in a new point near the figure boundary. The gradient operator is called again and the process is repeated. Contour following is an attractive solution to image segmentation, mainly because of its speed and simplicity. It suffers, however, from the same inherent disadvantage of all serial operations, namely the accumulation of errors. An error at any step in the process makes it more likely that succeeding steps will also be in error.

To minimize the probability of error, we developed a contour follower that makes use of various sources of local information at a given point. Gradient magnitudes, gradient directions and local average intensity levels at both sides of the edge are combined in a candidate selection process that bears some similarities to edge/border coincidence techniques (Ref. 31), and relaxation methods used for edge reinforcement (Ref. 32). The resulting scheme, however, remains a serial operation, conceptually different from relaxation techniques, that involve several parallel iterations. It can be defined as a combined gradient and binary follower, employing a candidate selection process.

4.2.2. Basic Steps

As soon as a valid starting point is found a following loop is initiated. The basic steps of this loop are:

STEP A : The slope of the edge at the current point is estimated. This defines three adjacent candidate links to the next point. A link is defined as a directed line segment (Ref. 33) The contour is traced in a clockwise sense. The 8 links of a cell are numbered from 0 to 7 in a counterclockwise sense (see Fig. 4,5).

STEP B : The tracing bug is moved tentatively to each one of the three candidate points. The three candidates are assigned a likelihood estimate evaluated in terms of the gradient value at each candidate point, the value of the edge slope, and the outcome of a binary test, which checks whether the left hand neighbour of the candidate is a background point, and the right hand neighbour is an object point. The most probable candidate is selected and the tracing bug is moved permanently to the new point.

STEP C : Each new point is checked for validity.

The coordinates of the new point are used to update running sums of primitive features.

Each new point is checked for a contour closure condition. If such a condition does not exist and the new point is valid, the process is repeated.

Otherwise a final report is issued stating the outcome of the following, either a succesfull closure or a failure, in which case the reason of failure is also reported.

4.2.3. STEP A : Definition of Candidate Links - Directional grading

Every new link in the following process is selected from 3 adjacent candidate links, picked out of the 8 links of the basic cell (Fig. 4.5). The selection of the candidate links is based on the slope continuity assumption. In other words, we assume that a link connecting two consecutive points on the contour, should have a direction close to the direction of the slope of the contour at the current point.

This assumption is not valid at points of slope discontinuities (sharp corners). By trying out 3 neighbour links, instead of one, and also by using only unitary links (of lengths 1 or $\sqrt{2}$), we can get around this difficulty, probably at the cost of introducing some spurious discontinuities (jaggedness).

Fig. 4.6 illustrates the following process.

$\vec{G}$ is the gradient vector, and $\vec{C}_{cen}$ is the centre candidate link. The outline is followed in a clockwise sense, therefore the centre candidate link is defined as the link with the closest direction to the perpendicular to the gradient vector, in an anticlockwise sense.

The directional angle ϕ , of the gradient vector, referenced to the x axis, is given by:

$$\phi = \tan^{-1} \left(\frac{V}{H} \right)$$

where $\vec{V}$ and $\vec{H}$ are the vertical and horizontal components of the gradient. The discrete analogs of these components are evaluated in terms of the 8 discrete intensity values of the basic cell (see Fig. 4.7). $\vec{V}$ is taken as positive, if it is pointing downwards; $\vec{H}$ is taken as positive, if it is pointing to the right. These discrete estimates are evaluated in accordance with the evaluation of the gradient magnitude by Sobel's operator (p.70)

The angular resolution within the basic cell is 45° .

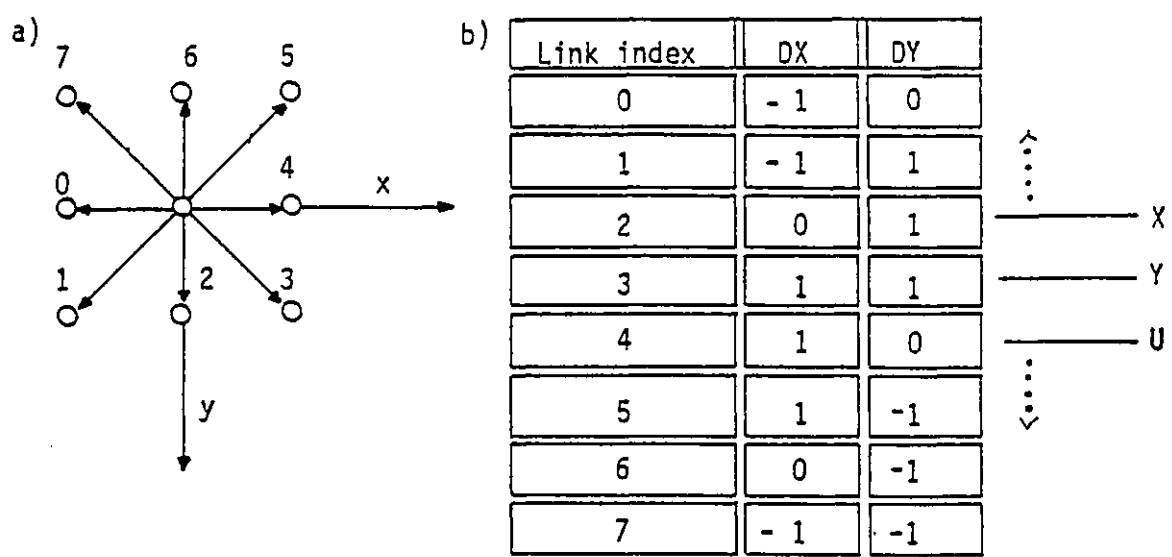


Fig. 4.5 :a) The 8 links of the basic cell
and b) Their components DX and DY

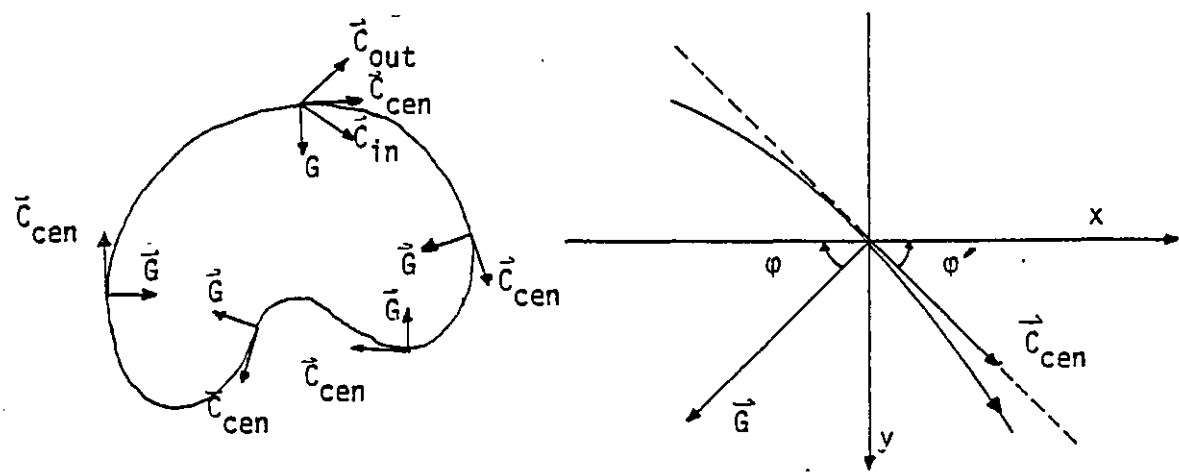


Fig. 4.6 : Definition of centre candidate link C_{cen}
relative to the gradient vector G .

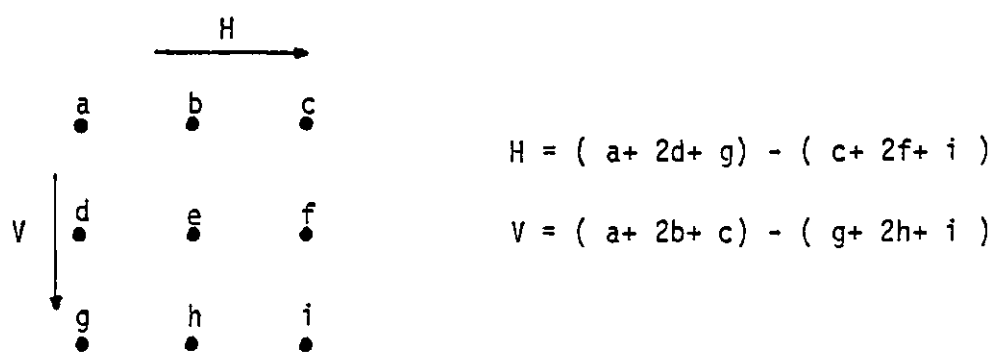


Fig. 4.7 : Evaluation of the vertical and horizontal
components of the gradient vector G .

Therefore, all directions must be quantized to a multiple of 45° . To achieve this, we define 4 decision sectors in each quadrant of the basic cell (Fig. 4.8b), each sector being 22.5° .

The resulting angular quantization rule is simple.

Any vector having a continuously valued angle ϕ is replaced by one of the 8 basic cell vectors, if it lies within a $\pm 22.5^\circ$ angular distance from that vector.

In other words, all directions are quantized to the nearest link direction.

Now, since $\tan(22.5^\circ) = 0.4142 \approx 0.4$

$$\text{and } \tan \phi = \frac{V}{H}$$

the angular quantization is achieved by a comparison among V , H and $2.5 H$, or H, V and $2.5 V$ ($1/2.5 = 0.4$).

These values can be easily obtained by simple binary operations.

The candidate selection algorithm proceeds as follows:

SUB-STEPS

- A_1 : From the signs of V and H we deduce the quadrant of the gradient vector, and from that, the quadrant of the centre candidate.
The three links within the specified quadrant are associated with three pointers X , Y and U , as in Fig. 4.9.
These pointers are pointing to three consecutive entries of a repetitive link list, one period of which is shown in Fig. 4.5b.
- A_2 : By comparing first the absolute values $|V|$ and $|H|$, and then, according to the outcome, the values of $|H|$ and $2.5|V|$, or $|V|$ and $2.5|H|$, we find the sector, where the centre candidate lies (Fig. 4.10).
According to the position of the specified sector with respect to X , Y and U , we select one of these as the centre candidate. The other two candidate links are labelled as C_{in} and C_{out} .
 C_{in} lies on the right of C_{cen} , and C_{out} on the left, therefore

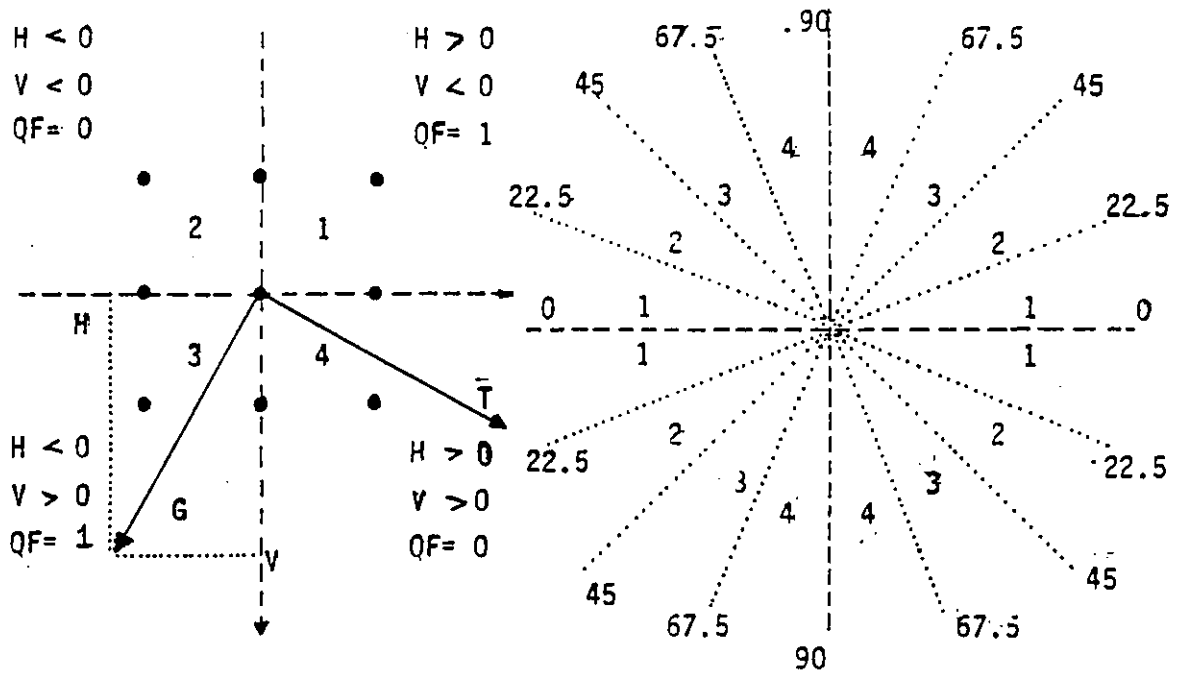


Fig. 4.8:

a) The 4 basic cell quadrants

G is the gradient vector

T is the tangent vector

H and V, the components of G

QF =Quadrant Flag

QF =1 for odd quadrants (H V 0)

QF =0 for even quadrants (H V 0)

b) Decision Sectors

for angular quantization

Angular resolution=22.5 deg.

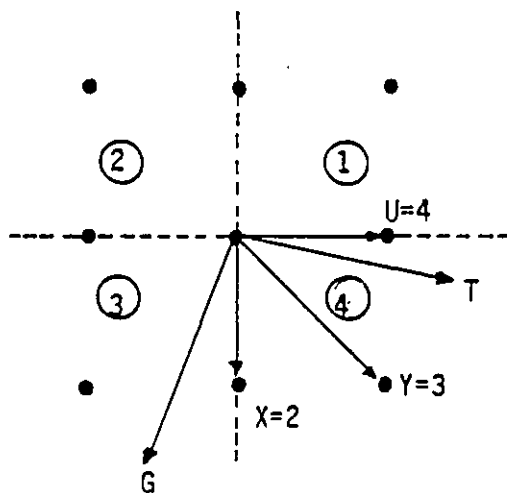
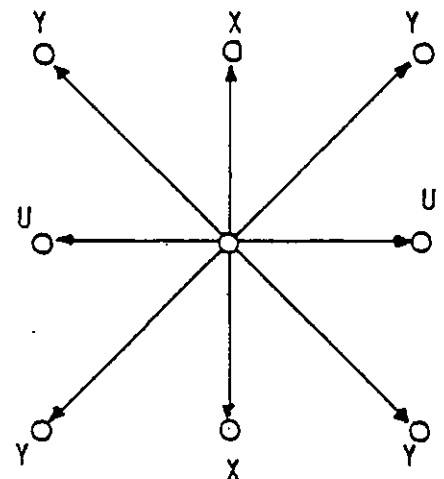


Fig. 4.9:

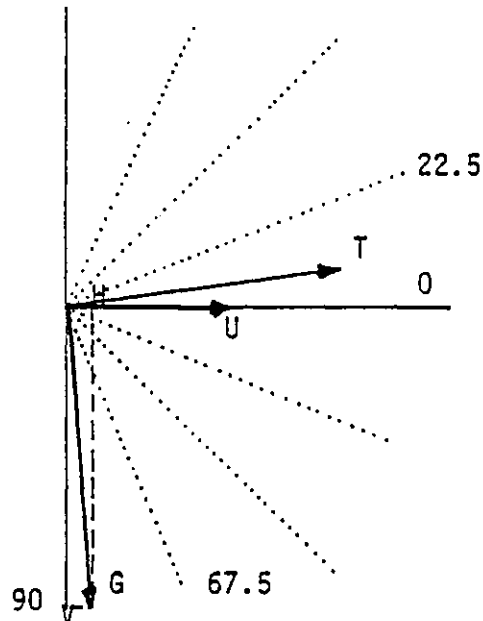
a) Gradient vector in quadrant 3,

Tangent vector in quadrant 4

X,Y and U are the candidate pointers.

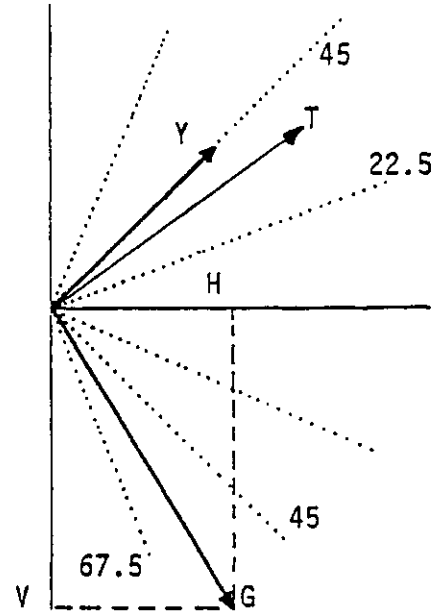


b) Arrangement of candidate pointers in the 4 quadrants



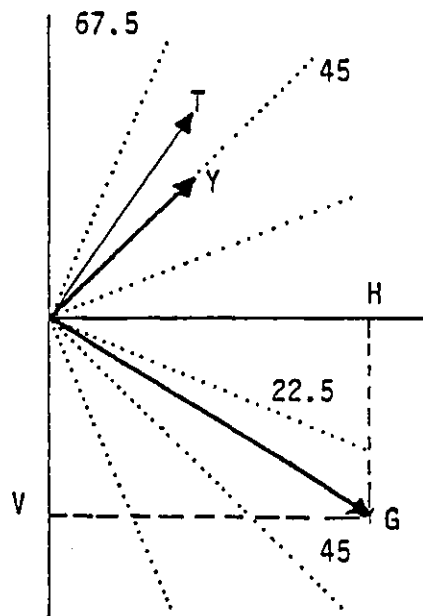
a) $|V| \geq 2.5|H|$
 G in sector 4 ($67.5 \leq \theta \leq 90$)

$$C_{cen} = U$$



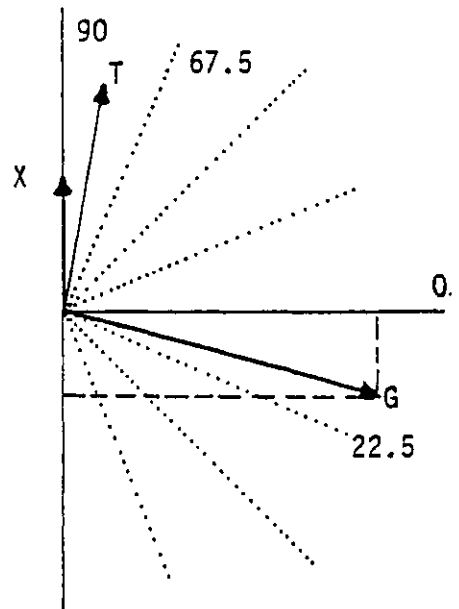
b) $|H| \leq |V| < 2.5|H|$
 G in sector 3 ($45 \leq \theta \leq 67.5$)

$$C_{cen} = Y$$



c) $|V| < |H| \leq 2.5|V|$
 G in sector 2 ($22.5 \leq \theta \leq 45$)

$$C_{cen} = Y$$



d) $|H| > 2.5|V|$
 G in sector 1 ($0 \leq \theta \leq 22.5$)

$$C_{cen} = X$$

Fig. 4.10 : Selection of centre candidate link (X,Y,U) according to the size relationships of the horizontal and vertical components of the gradient vector G .

in terms of the indices of the links (see also Fig. 4.5b) :

$$C_{in} = C_{cen} - 1$$

$$C_{out} = C_{cen} + 1$$

A₃ : Having found the 3 candidates, we now proceed in grading them, according to how close their directions are to the direction of the contour at the given point. This directional grading serves two purposes:

- a) It provides a discrimination of the candidates, according to their directions, in relation to the estimated slope of the contour.
- b) It introduces an angular resolution of 22.5° in a lateral way. In other words, although the links of the basic cell are spaced in angles of 45° , the 22.5° accuracy that is available from the evaluation of the slope, is incorporated in the selection process in terms of the directional grades. C_{cen} always gets the higher directional grade 3, C_{in} and C_{out} can get grades 1 or 2, depending on how close they are to the non-quantized tangent vector $\vec{T}$.

Fig. 4.11 illustrates the relative positions of $\vec{T}$, C_{in} , C_{cen} and C_{out} , and the resulting gradings.

Fig. 4.12 sums up the grading rules.

At the end of the candidate selection step, pointer Y is left pointing to the components of the centre candidate C_{cen} in the link list, the previous entry of the list being the components of the inner candidate C_{in} , and the next entry being the components of the outer candidate C_{out} .

VECTOR RELATIONSHIPS	DIRECTIONAL GRADES	Selection of Centre Candidate
		Position of Tangent Vector T
	CCEN = 3 CIN = 2 COUT = 1	CCEN = X T in 4 QF = 1
	CCEN = 3 CIN = 1 COUT = 2	CCEN = X T in 4 QF = 0
	CCEN = 3 CIN = 2 COUT = 1	CCEN = Y T in 2 QF = 1
	CCEN = 3 CIN = 1 COUT = 2	CCEN = Y T in 2 QF = 0
	CCEN = 3 CIN = 1 COUT = 2	CCEN = Y T in 3 QF = 1
	CCEN = 3 CIN = 2 COUT = 1	CCEN = Y T in 3 QF = 0
	CCEN = 3 CIN = 1 COUT = 2	CCEN = U T in 1 QF = 1
	CCEN = 3 CIN = 2 COUT = 1	CCEN = U T in 1 QF = 0

Fig. 4.11 : Directional Grading of the three candidate links (C_{in} , C_{cen} ,C_{out}), and selection of centre candidate obtained for each of the 8 possible positions of the tangent vector T in any of the 4 decision sectors, in an odd quadrant (QF=1) or an even quadrant (QF= 0).

DIRECTIONAL GRADING RULES

	Quadrant Flag	Centre Candidate	GRADES
$V \geq 2.5 H$	QF = 1	U	CIN = 1 COUT = 2
	QF = $\emptyset$		CIN = 2 COUT = 1
$ H \leq V < 2.5 H $	QF = 1	Y	CIN = 2 COUT = 1
	QF = $\emptyset$		CIN = 1 COUT = 2
$ V < H \leq 2.5 V $	QF = 1	Y	CIN = 1 COUT = 2
	QF = $\emptyset$		CIN = 2 COUT = 1
$ H > 2.5 V $	QF = 1	X	CIN = 2 COUT = 1
	QF = $\emptyset$		CIN = 1 COUT = 2

Fig. 4.12 - Directional Grading Rules

4.2.4. STEP B : Selection of the most probable link

Each of the three candidate points is now subjected to two more tests, a gradient test and a binary test. The outcomes of these two tests are expressed in terms of two additional marks. A weighted sum of all the marks constitutes a likelihood estimate for each candidate point. The link leading to the point with the highest total is selected as the most probable link.

Assuming that D is the directional mark obtained in STEP A, G is the gradient mark and B is the binary mark, then the total mark for each candidate link is given by:

$$M_i = \alpha \cdot D_i + b \cdot G_i + c \cdot B_i \quad (i = 1, 2, 3)$$

D_i can obtain any of the values 1,2 or 3

G_i can obtain any integer value from 0 to 15

and B_i can obtain any of the following values: -1, 0, 1

The weighting factors have the following values:

$$\alpha = 2, b = 8, c = 4$$

The selection of these values was somewhat arbitrary. The contribution of the gradient mark was significantly enhanced because it was proved the most reliable of the three. A candidate that satisfies the directional and the binary test will be at most enhanced by a little more than the equivalent of two gradient marks. This combination was found adequate to aid in the selection of an optimal route along the outline but any other set of values and weighting factors could have been tried.

SUB-STEPS

B₁ : The Gradient Test :

The gradient of the picture function can be estimated from the directional derivatives of the function along any two orthogonal directions. The main difficulty in this process is the existence of noise. We would like, therefore, to smooth the image function to eliminate most of the high-frequency noise, but not to smooth it so much as to blur the high-frequency jump at the vicinity of the edge, that we are trying to detect.

A simple gradient operator suggested by I. Sobel has been used for this purpose. The gradient value at the centre of a 3 x 3 basic cell is calculated, in terms of the pixel numbering convention of Fig. 4.7, as follows:

$$G = \sqrt{H^2 + V^2}$$

This operator attempts to estimate the partial derivatives H and V, by first forming two weighted averages to estimate the picture at points d, f, b and h, and then subtracting them along the two orthogonal directions x and y. Although the definition of this operator is based on purely intuitive grounds, its performance for pictures mildly corrupted by noise is acceptable.

In our case the gradient values obtained are normalized in the range 0-15, and the square root is replaced by an integer approximation of the real value, found from a look-up table.

The gradient is evaluated at every candidate point.

B₂ : The Binary Test :

The gradient operator is applied on a small window of the picture function, and it is therefore susceptible to localized undesirable variations of the light intensity (shadows, non-uniform reflectivity along the outline, etc).

To diminish the effect of these local deteriorations of edge sharpness, we introduced a topology test, which is based both on local intensity information, obtained from the vicinity on either sides of the edge, and on information obtained at the previous steps of the following process.

At each new point p_i we establish what we call a local intermediate intensity level M_i , evaluated as a middle intensity level between an estimated average local background intensity, and an average local object intensity (see Fig. 4.13).

The local intermediate level M_i is averaged with the previous intermediate level M_{i-1} , to obtain the new value of M_i .

Once the three candidate links for the next movement are established, the tracing bug is moved tentatively, to each of the candidate points and two samples of the local intensity on either side of the candidate link are taken.

For a candidate link to be successful the sampled intensities must have the right relationship to the intermediate intensity M_i . In other words an assumed background sample must be a 'black' sample (with intensity lower than M_i), if the background is 'black', and the assumed object sample must be 'white' (with intensity higher than M_i), or vice versa.

The binary grading rule is simple.

For each successful test a +1 mark is granted to the candidate. An unsuccessful test leads to a -1 mark. Therefore the binary test at each candidate point can lead to a +1, a -1, or 0 binary mark (for one successful and one unsuccessful sample).

This iterative way of defining the intermediate level is used in order to obtain a means of topological classification of pixels (inside and outside pixels with respect to the outline), that is not based only on local intensity levels, and it should result in a smoother and more accurate outline.

The binary grading rules are summarized in Fig. 4.14. BW is the black/white parameter, which is entered by the operator, and takes up the value BW = 1 for white (reflective) objects on black (non-reflective) background, or the value BW = 0 otherwise.

At this stage each candidate point is also checked for a loop condition that may occur especially at sharp turnings (Fig. 4.15a). Before granting the binary and the gradient mark, the trace field of each candidate is examined, to make sure that it is a genuinely new point (Bit 7 = 0).

If it has already been classified as an outline point, it means that the tracing bug is returned to a previous point and the following process may be trapped in an endless loop, therefore this candidate must be excluded from the selection (it gets a zero total). But before that, we must make sure that it is not the starting point, in which case a contour closure condition would occur.

As it will be seen in STEP C, when the tracing bug approaches the starting point at a distance of two units, a closure flag is raised.

By checking at this point the status of the closure flag we can detect a closure condition, in which case the suspected candidate not only is not excluded, but is also given absolute priority by getting the highest possible total mark (127), to ensure a proper closure of the contour.

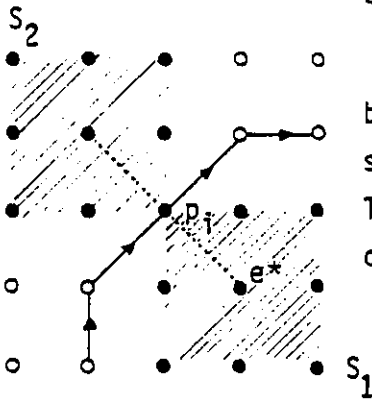
Finally, after all three candidates have been assigned a total mark, the candidate link with the highest mark is selected as the most probable subsequent link, provided that its total mark is also higher than a threshold value which is entered by the operator at the initialisation stage. This threshold, expressed in terms of gradient values (0-15), is used as an absolute quality criterion to ensure that the tracing bug is not going to get lost following shadows, or any other irrelevant regions with light intensities slightly different from that of the background, which could be stains, local variations

a b c
d (e) f
g h i

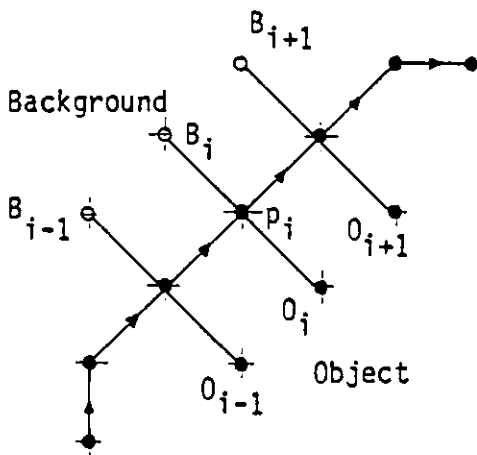
Fig. 4.13 : BINARY TEST

a) Intensity sample S obtained at point e by averaging the 8 neighbouring pixels of the basic cell centered at e .

$$S = (a+b+c+d+f+g+h+i) / 8$$



b) Two samples S_1 and S_2 taken on either side of the outline at point p_i . The dashed areas cover the two sample cells, centered at points e and e^*

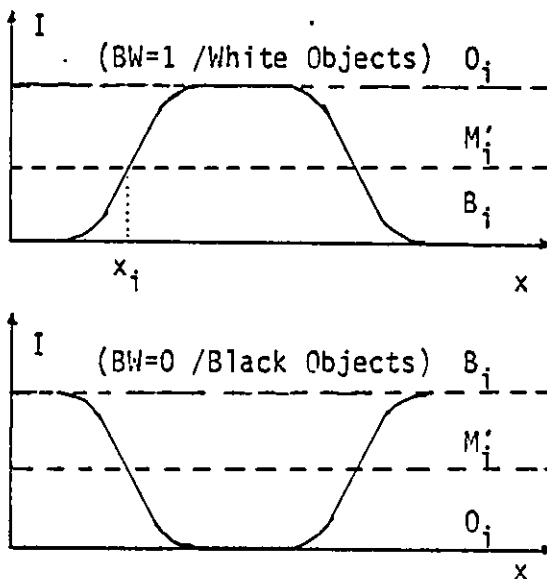


c) Evaluation of running average intermediate level M_i , at point p_i .

$$M'_i = (B_i + O_i) / 2$$

$$M_i = (M_{i-1} + M'_i) / 2$$

Fig. 4.14 : BINARY GRADING RULES



(S is the value of the intensity sample)

$S_1 \leq M \sim B_1 = -1$	Object
$S_1 > M \sim B_1 = 1$	Sample S_1
$S_2 \geq M \sim B_2 = -1$	Background
$S_2 < M \sim B_2 = 1$	Sample S_2
$S_1 \geq M \sim B_1 = -1$	Object
$S_1 < M \sim B_1 = 1$	Sample S_1
$S_2 \leq M \sim B_2 = -1$	Background
$S_2 > M \sim B_2 = 1$	Sample S_2

of background reflectivity due to non-uniform illumination, etc.

Typical threshold values are $T = 3$ or $T = 4$, but for sharp images even higher thresholds would do.

If at any point the best candidate is not good enough (its total mark is lower than T), the following process is stopped and a failure due to low contrast is reported (Report No 2). The same report is given for the above mentioned loop condition if all three candidates fail to lead to a new point. Failure caused by one of these two reasons is not considered fatal, and the outline will be retraced from a different starting point, when it will be intercepted again from the search routine (p.56)

4.2.5 STEP C : CONTROL FUNCTIONS

C_1 : Validity controls

The smooth continuation of the contour following may be interrupted due to one of the following reasons:

- a) The tracing bug is trapped in a loop.
- b) A low-contrast region is encountered.
- c) Only part of the outline appears in the image.
- d) One of two overlapping or touching outlines is being followed.
- e) The contour is closed.

The first two conditions have already been tested in STEP B. The closure condition is explained in C_3 . The detection of conditions c) and d) is explained below:

c) All the unit step movements attempted during the process of finding the best candidate point were tentative and not permanent movements. The tracing bug still rests at the previous point. A new tentative movement to the selected candidate is now attempted. The low-level routine that evaluates the new coordinates, checks their values against the limit coordinates of the image array and reports to the calling routine. If the new point lies outside the limits of

the array in memory, it means that the object that is being traced does not lie entirely in the picture. The following process is stopped and a fatal failure is reported.

d) The trace number of the new point is compared to the trace code assigned to the points of a sealed outline (code no 4 in fig. 4.3). If this trace is encountered, it means that the outline has touched a previously followed and sealed outline, or an unsuccessfully followed outline, that was stopped due to a fatal failure. This in turn signifies either the presence of another object, or that the currently followed outline has been already followed with no success.

Both conditions are considered as a cause for a fatal failure, and the following attempt is stopped. From this discussion it is apparent that the detection of overlapping outlines can only be achieved if one of them has already been followed.

There is still the possibility that two or more overlapping or touching outlines will be treated as an entity, but the resulting contour will most probably not resemble any of the outlines of the known objects, and consequently it will not be misclassified. Fig. 4.15 illustrates the various conditions that may cause an end to a contour following attempt.

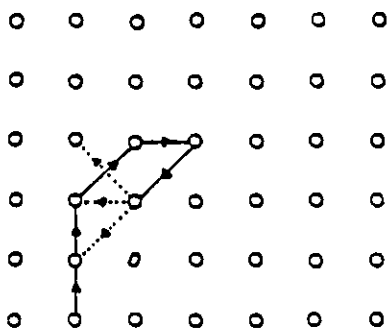
C₂ : Painting and recording the new points

Having established a valid new point, the next step is to paint it in the image memory by setting its trace field to the appropriate code, and then use it to update the running sums which are necessary for the calculation of the primitive features (p.81)

The 4-bit code used to paint new points at the contour following stage contains the following information:

- a) Bit-7 is set to 1, to denote that the point belongs to an outline that is still being processed.
- b) Bits 6,5 and 4 contain the number of the link that has led

a) Loop Condition

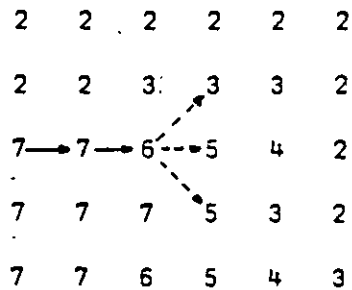


Two of the candidate links (dotted vectors) lead to previous points, the third gets a low mark. Following is stopped.

REPORT No 2

NON- FATAL FAILURE

b) Low Contrast



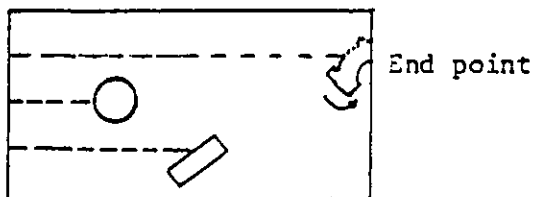
Blurred Edge.

The 3 candidate links get the following marks : $C_{out} = 26$, $C_{in} = 36$, $C_{cen} = 30$. If a quality threshold $TH=5$ was chosen, then all candidate marks are less than $5*8 = 40$.

REPORT No 2

NON- FATAL FAILURE

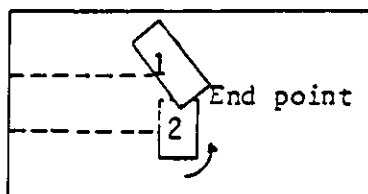
c) Object partly in the scene



REPORT No 3

FATAL FAILURE

d) Overlapping objects



Object no 1 has been traced.

Outline of object no 2 cannot be followed because it intersects an already followed outline.

REPORT No 4

FATAL FAILURE

Fig. 4.15 :Reasons of contour following failure

to the new point (Fig. 4.5b).

By this coding scheme all the points of the outline are connected in a chain, so that any outline can later be retraced by simply recording the starting point. With this structure there is no need to record the coordinates of all the points in a separate list in memory. However for reasons of flexibility, such a list is also constructed only for the current outline, so that if more information is required, two alternative representations of the outline are available. The chain-code is also useful for the internal scanning process, which is followed for the detection of surface features (p.102)

When the outline of an object has been used and no more information is needed, it is sealed by painting all its points with a different trace code, either a code no 4, or code no 5, according to the outcome, of the contour following (Fig. 4.3).

Fig. 4.16 is a printed output of a processed image array, showing the various codes that are being used to paint the outlines.

The following measurements are updated next:

1. The total number of outline points (N).
2. The coordinates of a vector connecting the starting point to the new point (TOTDX and TOTDY).

To update these coordinates we use the components DX_i and DY_i of the new link, in the following way:

$$TOTDX_i = TOTDX_{i-1} + DX_i$$

$$TOTDY_i = TOTDY_{i-1} + DY_i$$

(see also Fig. 4.17)

3. The absolute values of these coordinates ABSDX and ABSDY.

These measurements are then used to update a set of running sums, which at the end of a successful following will produce the primitive features of the outline (centre, perimeter, area, etc.).

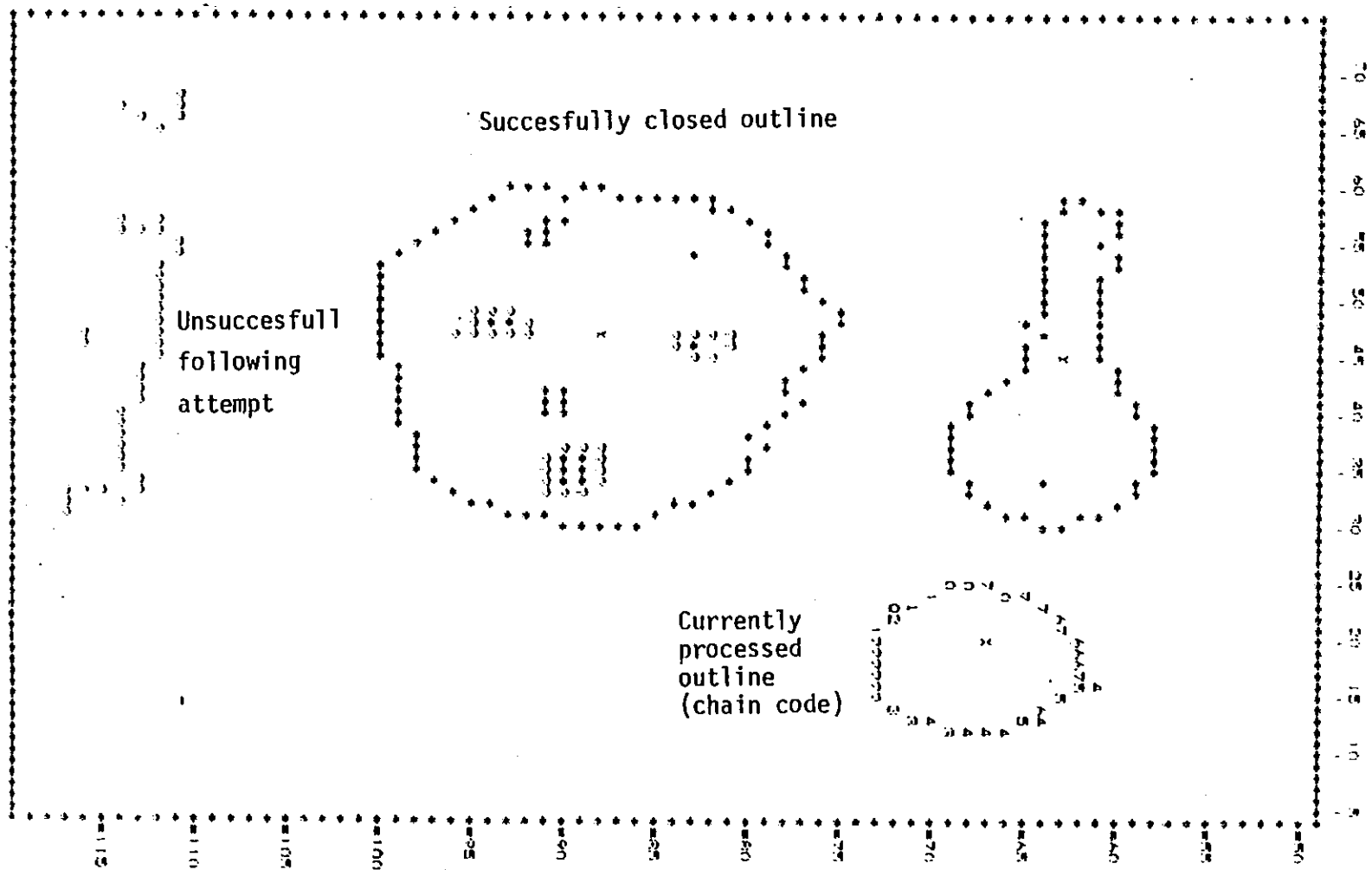


Fig. 4.16 : Processed image displaying the different trace codes.

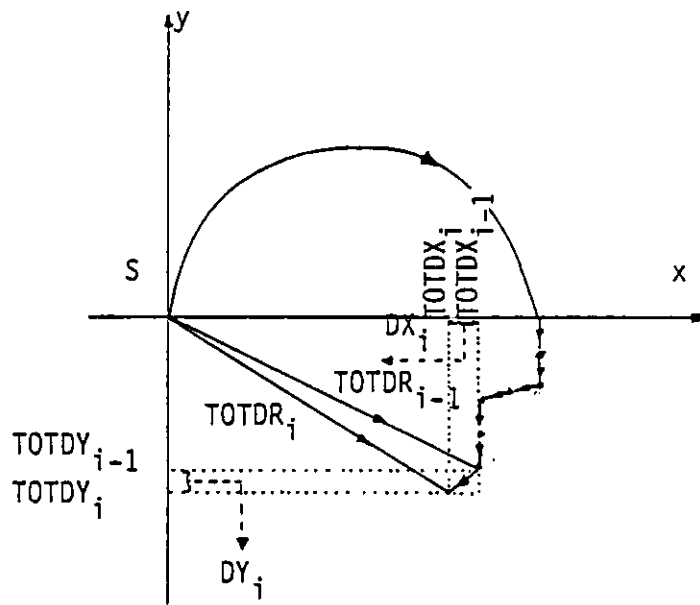


Fig.4.17: Measured coordinates of contour points.

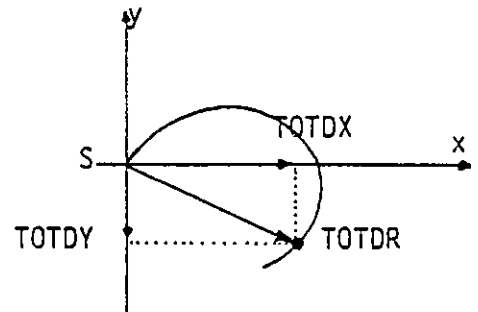
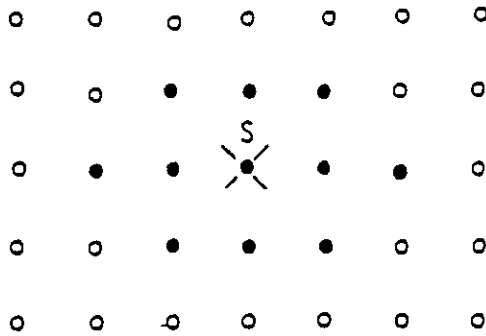


Fig. 4.18: Set of points (black points) that satisfy proximity condition for closure : $|TOTDR| \leq 2$.
S is the starting point.

C₃ : Closure Control:

A contour is assumed to be closing when either of the following conditions are met:

- a) The last point coincides with the starting point or a neighbouring point.
- b) The distance from the last point to the starting point is equal to the unit distance of the grid, provided that at least 5 points have been traced.

The measure of this distance TOTDR is defined as:

$$\text{TOTDR} = |\text{TOTDX}| + |\text{TOTDY}| = \text{ABSDX} + \text{ABSDY}$$

(see Fig. 4.17 and Fig. 4.18)

For every new point the distance TOTDR is evaluated and compared initially to 2.

If TOTDR = 0 or 1, the contour is considered closed.

If TOTDR = 2, a proximity flag is set, so that the next point will be checked bearing in mind that the starting point has already been reached at a distance of 2 units.

The contour will be considered closed at that point, even if a closer approach (TOTDR = 1) is not achieved.

This precaution is taken in order to avoid tracing part of the contour twice, a situation which may arise if the border region between the object and the background is wide, as for example in the case of shadows or blurring.

The objective here is to approach the starting point at the shortest possible distance (0 or 1), without risking retracing the contour if a distance less than 2 units cannot be achieved.

The best closure of course would be to meet the starting point, a case promoted by the loop checking routine (p.72).

4.3. Primitive Features of the Outline

4.3.1. Introduction

The next step of the low-level description stage involves the derivation of a set of attributes of the traced outline. The objective again is to reduce the amount of available information, from a list of chain encoded outline points, to a set of representative primitive features, from which the position, the orientation and the identity of the unknown object will be derived in the classification stage.

The selection of informative features is so important that, for some workers in the field, it solely defines the objective of pattern recognition (Ref.34)

Unfortunately, a general theory for the selection of features relevant for a particular problem does not exist. Metric properties (properties involving distances in the image array), topological properties, local properties, and analytic descriptions of the outline have been exploited in various combinations. The choice of features is influenced by the classification approach adopted, and the nature of the problem.

The two main categories of recognition strategies, namely the statistical approach (based on the classical decision theory, and the structural or linguistic approach (based on the creation of special purpose grammars to describe the structural relationships of a set of patterns), employ features of different nature.

In our case, simplicity of operation, speed, and generality considerations influenced the adoption of a statistical approach, based on global metric attributes.

Hence, the set of primitive features that was selected includes zero order, first order and second order position and orientation invariant moments of the outline (Ref. 35)

- List of Primitive Features

BASIC FEATURES

N : Number of outline points

P : Perimeter (total length) of the outline

A : Area (total area enclosed by the outline)

(XMAX, YMAX, XMIN, YMIN) : Dimensions of the minimum window
with sides parallel to the X, Y axes, that circumscribes
the outline.

CENX, CENY : Coordinates of the centre of gravity of the outline

SECOND CENTRAL MOMENTS

M_{20} (IX) : Moment of inertia about the X-axis

M_{02} (IY) : Moment of inertia about the Y-axis

M_{11} (IXY): Product of inertia

CHARACTERISTIC DISTANCES

DMAX : Maximum distance of outline points from centre

DMIN : Minimum distance of outline points from centre

DAV : Average distance of outline points from centre.

The measurements leading to the derivation of the BASIC FEATURES are updated at every point during the contour following, and they are therefore available at the end of the following, irrespective of the outcome. The same applies to the centre coordinates which determine the position of the object. The remaining features are evaluated after the contour following stage, and only for successfully followed outlines.

4.3.2. Calculation of Primitive Features

The numerical calculation of the first moments (centre coordinates) and the second moments (moments of inertia), can be based on discrete series approximations of Riemann integrals (Ref. 35)

Any closed region in the x,y plane can be represented by a density distribution function $\rho(x,y)$.

The two-dimensional $(p+q)$ th order moments of this function are defined in terms of Riemann integrals as:

$$m_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p y^q \rho(x,y) dx dy \quad (1)$$

For a digitized image the discrete analogs of these integrals can be defined, if we replace $\rho(x,y)$ by a binary function that has a value of 1 for all the elementary areas ΔA within the outline, and it is zero elsewhere.

The moment integrals above can now be replaced by the moment sums:

$$M_{pq} = \sum_{i=1}^n x_i^p \cdot y_i^q \cdot \Delta A \quad (2)$$

where: n = total no of points enclosed within the contour, and
 $\Delta A = 1$ or $1/2$ (see Fig. 4.19).

Using the Riemann formula (Ref. 35), the moment integrals, and consequently the moment sums can be evaluated along the borders of the region, as follows (Ref. 33)

$$\begin{aligned} M_{01} &= XCEN = \sum_{i=1}^N \frac{1}{2} \Delta x_i [y_{i-1}^2 + \Delta y_i (y_{i-1} + \frac{1}{2} \Delta y_i)] \\ M_{20} &= IX = \sum_{i=1}^N \frac{1}{3} \Delta x_i [y_{i-1}^3 + \frac{3}{2} \Delta y_i y_{i-1}^2 + \Delta y_i y_{i-1} + \frac{1}{4} \Delta y_i^3] \end{aligned} \quad (3)$$

where N represents the total number of contour points and $\Delta x_i, \Delta y_i$ are the components of the link that leads from point (x_{i-1}, y_{i-1}) to point (x_i, y_i) of the contour.

These formulae are rather complex and they involve second and third order terms for the evaluation of first and second moments respectively.

The method adopted in our case for the evaluation of the first and second order moments was based on the fact that the shape of an object is determined by its outline, and that, consequently, the same kind of information obtained from these moments of a 'solid' object could be obtained from an 'object' that consists only of the unit squares centred at the points of the outline (see Fig.4.20). In other words we are considering a density distribution function $p(x,y)$ only along a narrow path along the outline.

This scheme results in the following very simple formulae for the evaluation of the centroid coordinates and the moments of inertia:

$$XCEN = \frac{1}{N} \sum_{i=1}^N x_i, \quad YCEN = \frac{1}{N} \sum_{i=1}^N y_i \quad (4)$$

$$\text{and} \quad IX = \frac{1}{N} \sum_{i=1}^N x_i^2, \quad IY = \frac{1}{N} \sum_{i=1}^N y_i^2, \quad IXY = \frac{1}{N} \sum_{i=1}^N x_i y_i \quad (5)$$

involving only 1st order and 2nd order terms respectively. This simplification results of course in some loss of accuracy, especially for compact shapes, where the number of outline points is small compared to the total number of points enclosed by the outline; however it proved justifiable, considering the gain in execution time.

The second moments and the characteristic distances are evaluated in a second pass, if they are required. The coordinate system is translated from the starting point S to the centre C , and the new coordinates

x,y of the outline points are evaluated in terms of the stored coordinates TOTDX, TOTDY as in Fig. 4.21

For each new set of coordinates the terms x^2 , y^2 and xy are evaluated with 16-bits accuracy, and the sums $\sum x_i^2$, $\sum y_i^2$ and $\sum x_i y_i$ are updated with 32-bits accuracy.

To calculate the characteristic distances we need to evaluate the distances $d_i = \sqrt{x_i^2 + y_i^2}$ of all the points from the centre. The calculation of the square-root for every set (x_i, y_i) is time consuming and it is only really necessary for the evaluation of the average distance DAV, since:

$$DAV = \frac{\sum_{i=1}^N d_i}{N} = \frac{\sum_{i=1}^N (x_i^2 + y_i^2)^{1/2}}{N} \quad (6)$$

$$\text{whereas } DMAX = \max_{i=1 \text{ to } N} (d_i) = \left[\max_{i=1 \text{ to } N} (d_i^2) \right]^{1/2} \quad (7)$$

$$DMIN = \min_{i=1 \text{ to } N} (d_i) = \left[\min_{i=1 \text{ to } N} (d_i^2) \right]^{1/2} \quad (8)$$

In other words only one square-root calculation is needed for DMAX and DMIN at the end of the pass, which can be performed accurately by the floating-point processor (see p. 145) or not performed at all, since $\max (d_i^2)$ contains the same information as $\max (d_i)$.

However, because of the scheme adopted for the evaluation of the compound features (see fig. 5.2) where DAV plays an important role as a reference feature, the square-root metrics $\sqrt{x_i^2 + y_i^2}$ are evaluated at every point.

A Newton-Raphson iterative formula was first tried for the evaluation of the square-root, but it was later replaced by a look-up table occupying 256 bytes, because it was time-consuming.

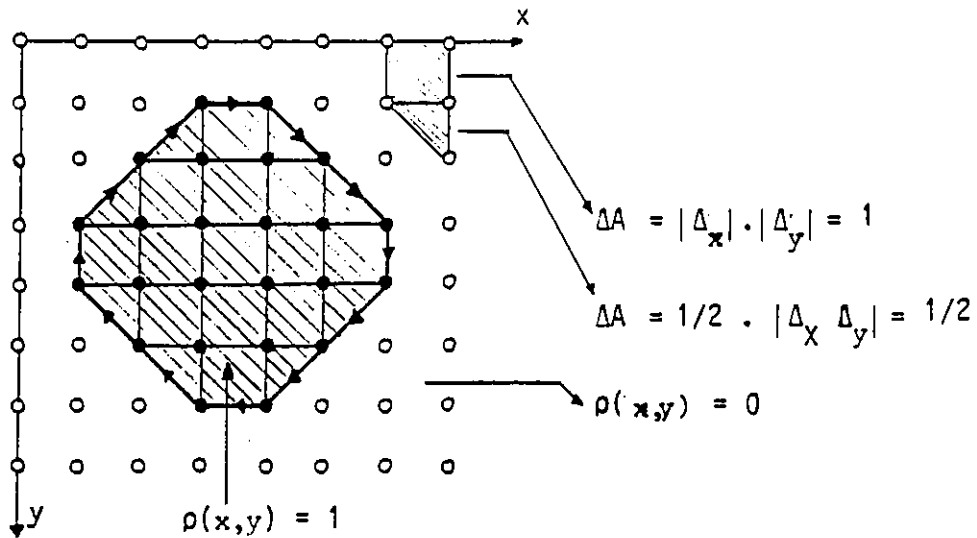


Fig 4.19 : Sub-division of an object area into elementary areas for the calculation of moment sums.

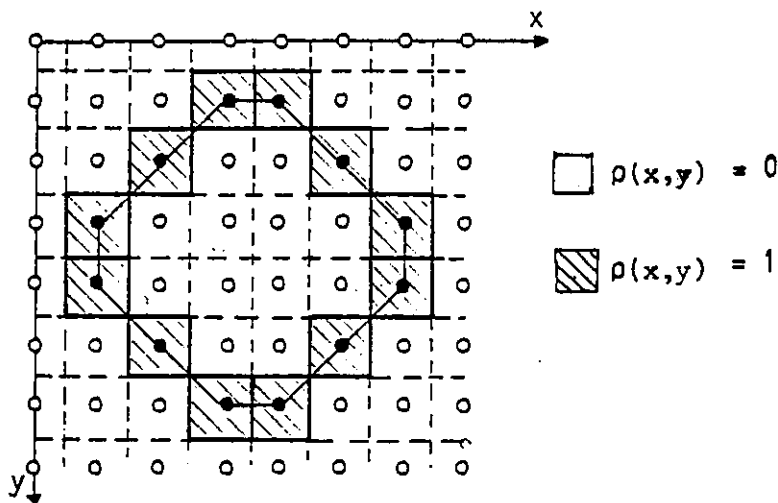


Fig 4.20 : Unit area rectangles at outline points from a "ring", which is used for the calculation of first and second moments.

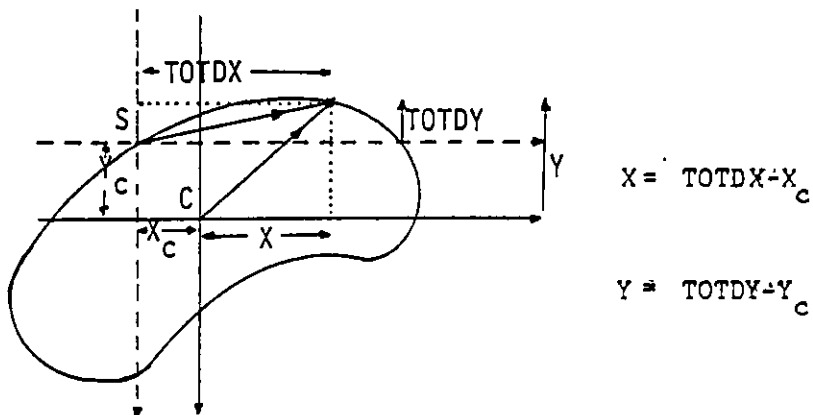


Fig 4.21 : Transformation of coordinates from the starting point coordinate system to the centre coordinate system.

Fig. 4.22 . Algorithm for the evaluation of the PERIMETER LENGTH

$$P = N_e + N_o \cdot \sqrt{2}$$

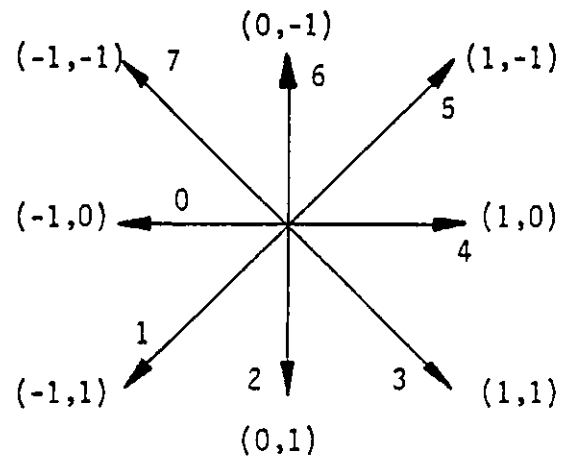
N_e = number of even links
(link nos 0,2,4,6)

N_o = number of odd links
(link nos 1,3,5,7)

$$P = N + N_o (\sqrt{2}-1)$$

N = total no of perimeter links
(points)

$$P = N + 0.4142 \cdot N_o \text{ (N available)}$$



Algorithm:

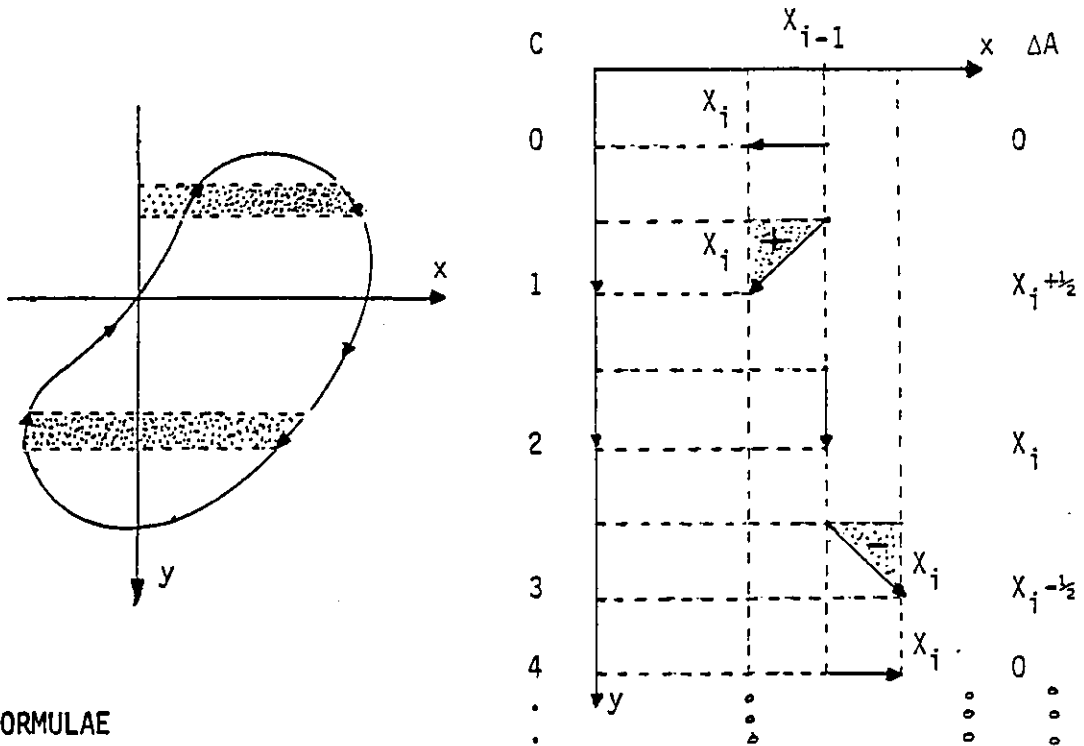
$b_2b_1b_0$ = bit representation of the chain-code

If $b_0 = 1$, increment N_o

Otherwise do nothing

* The constant $(\sqrt{2}-1)$ can be roughly approximated by $3/8$, or more accurately by $(106/256)$

Fig. 4.23 Algorithm for the evaluation of the AREA
(Numerical integration with respect to the y-axis)



FORMULAE

$$A = \sum_{i=1}^N x_i \cdot \Delta y_i - 1/2 \Delta x_i \cdot \Delta y_i = A_1 + 1/2 A_2$$

$$A_1 = \sum x_i \cdot \Delta y_i$$

$$A_2 = - \sum \Delta x_i \cdot \Delta y_i$$

$$\Delta A_1 = x_i \cdot \Delta y_i$$

$$\Delta A_2 = - \Delta x_i \cdot \Delta y_i$$

VARIABLES : C =Chain-code of ith link (C=0 to 7)

b_2, b_1, b_0 =Binary digits of C

ALGORITHM

a)Calculation of A_1 :

i) If $b_1, b_0=0$ (C=0 or 4), then $\Delta A_1 = 0$

ii) Otherwise: if $b_2=0$ (C=1,2,3), then $\Delta A_1 = X_i$

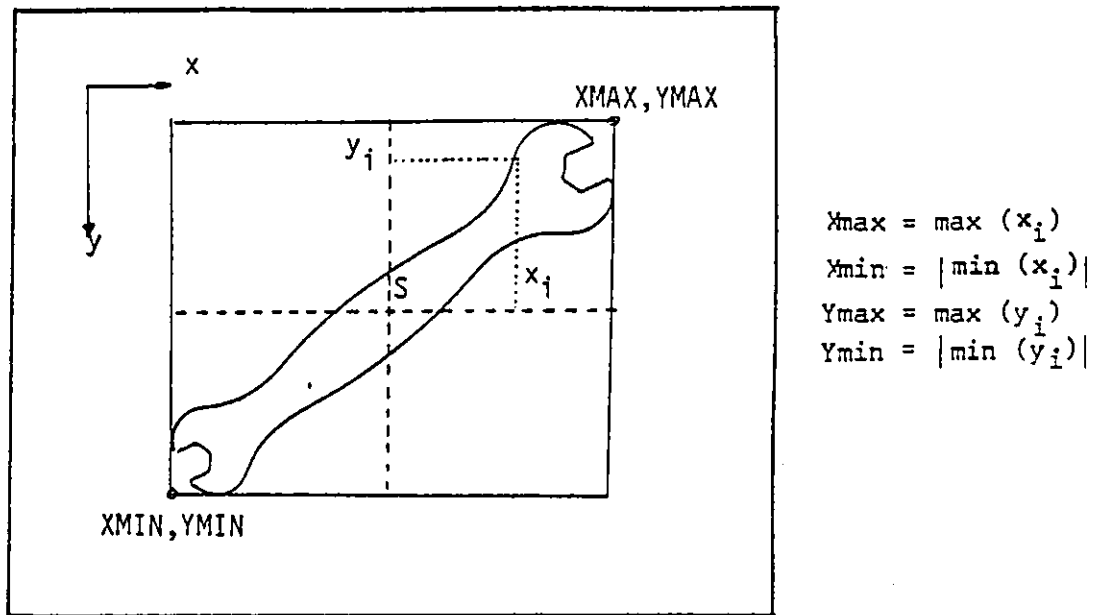
if $b_2=1$ (C=5,6,7), then $\Delta A_1 = -X_i$

b)Calculation of A_2 :

i) If $b_0=0$ (C=0,2,4,6), then $\Delta A_2 = 0$

ii) Otherwise: if $b_1=0$ (C=1,5), then $\Delta A_2 = 1$

if $b_1=1$ (C=3,7), then $\Delta A_2 = -1$



Algorithm XBOX : $X_i \geq 0$ $X_i < 0$ YBOX : $Y_i \geq 0$ $Y_i < 0$ Return	Update Xmax And Branch to YBox Update Xmin Update Ymin and return Update Ymax
--	--

Fig. 4.24 : Algorithm for the evaluation of the dimensions of the CIRCUMSCRIBED RECTANGLE.

A search routine provides the integer square root of any integer from 1 to 16384 ($\sqrt{16384} = 128$), with a round-off correction, in a maximum of 23 simple comparison steps.

The evaluation of the perimeter length, the area, and the dimensions of the circumscribed window involves the coordinates of the outline points TOTDX, TOTDY and their link numbers (chain code).

The algorithms used for these calculations are illustrated in Figs 4.22, 4.23 and 4.24. These calculations are also optimised time-wise, because they are repeated for all the points of the outline. The area and the perimeter are represented by 16-bit words.

4.4. Internal Features

4.4.1. Introduction

The description of the outline of an object provides a set of features which in many cases is sufficient for the successful classification of an object, by discriminating it from the other objects of interest, in terms of its shape and its size.

Sometimes, however, either because this description alone is not 'informative' enough (objects with close shape and size similarities), or because the main interest is focused on other details of the figure of a part (as for instance in assembly operations), there is a need to extract more information.

An important class of additional features can be derived in such cases from distinctive surface irregularities, such as holes, protrusions, or even the texture of the surface.

The most common surface features of mechanical parts are holes, and in fact many existing vision systems include in their feature sets measurements such as the area of the largest hole, the coordinates of the centroid of the largest hole, the count of holes in the shape, the sum of areas of all the holes, etc. (Ref.36)

The process of describing the internal features is normally carried out in two stages:

- a) Isolation of internal blobs (a blob is any connected component)
- b) Measurement of primitive features of the individual blobs (their relative positions, their areas, etc).

The first stage can present many problems..The edges of internal blobs in many cases are not as sharp as the edges of the objects. Non-informative irregularities of surface reflectivity (scratches, stains, etc.) and other sources of noise, may result in an inconsistent, therefore unreliable description.

Let us consider for example the variability that will occur in the measurement of the number of holes of a part, if one or two holes in a total of say four holes are not detected.

To improve the consistency of the description, a different description model was adopted that treats the set of internal blobs as a unified pattern of internal points.

The whole area enclosed by the outline is searched for internal points, and the resulting distribution of points is represented by a characteristic function $f(x,y)$, same as the one defined for enclosed figures (p. 83). Since this function is also bounded (by the outline of the figure), the moments of the distribution exist and can provide a description of the pattern.

A similar set of measurements as the one used for the description of the outline is now used to describe the internal pattern (zero, first and second moments, total perimeter, etc.).

The search for internal points is confined within the circumscribed window of Fig. 4.24. A topological analysis of all the points of the window, classifies their position relative to the outline (INSIDE-OUTSIDE).

The preceding contour following stage established connectivity for the object traced, in terms of the chain list representation of its outline. The painting scheme adopted proved its merits at this successive stage, where the traces left in the image array in terms of the chain-code, proved valuable as a means of establishing topological relationships.

An internal segmentation based mainly on binary techniques isolates those inside points, that constitute elementary connected componenets of the internal pattern.

4.4.2. Search for points of the internal pattern

A raster search routine, similar to the one used for the search for objects, is now used for the search for points of the internal pattern, only the search area is now confined within a rectangular window which coincides with the rectangle that is circumscribed around the outline (Fig. 4.25).

All movements within the image area are controlled by a low-level routine, that checks if the new point lies within the image frame. By changing four global variables that define the limits of the image array to new values corresponding to the coordinates of the corners of the specified window, we can easily select any sub-image, in such a way that the change is transparent to all the other routines.

The search is carried out along horizontal lines, point-by-point. Two sets of measurements are used; one incorporating the coordinates of the points found along the current line (line measurements), and one set of global measurements.

The following classification sequence is followed for every point.

- . First the point is classified as being INSIDE or OUTSIDE the outline (Topological analysis).

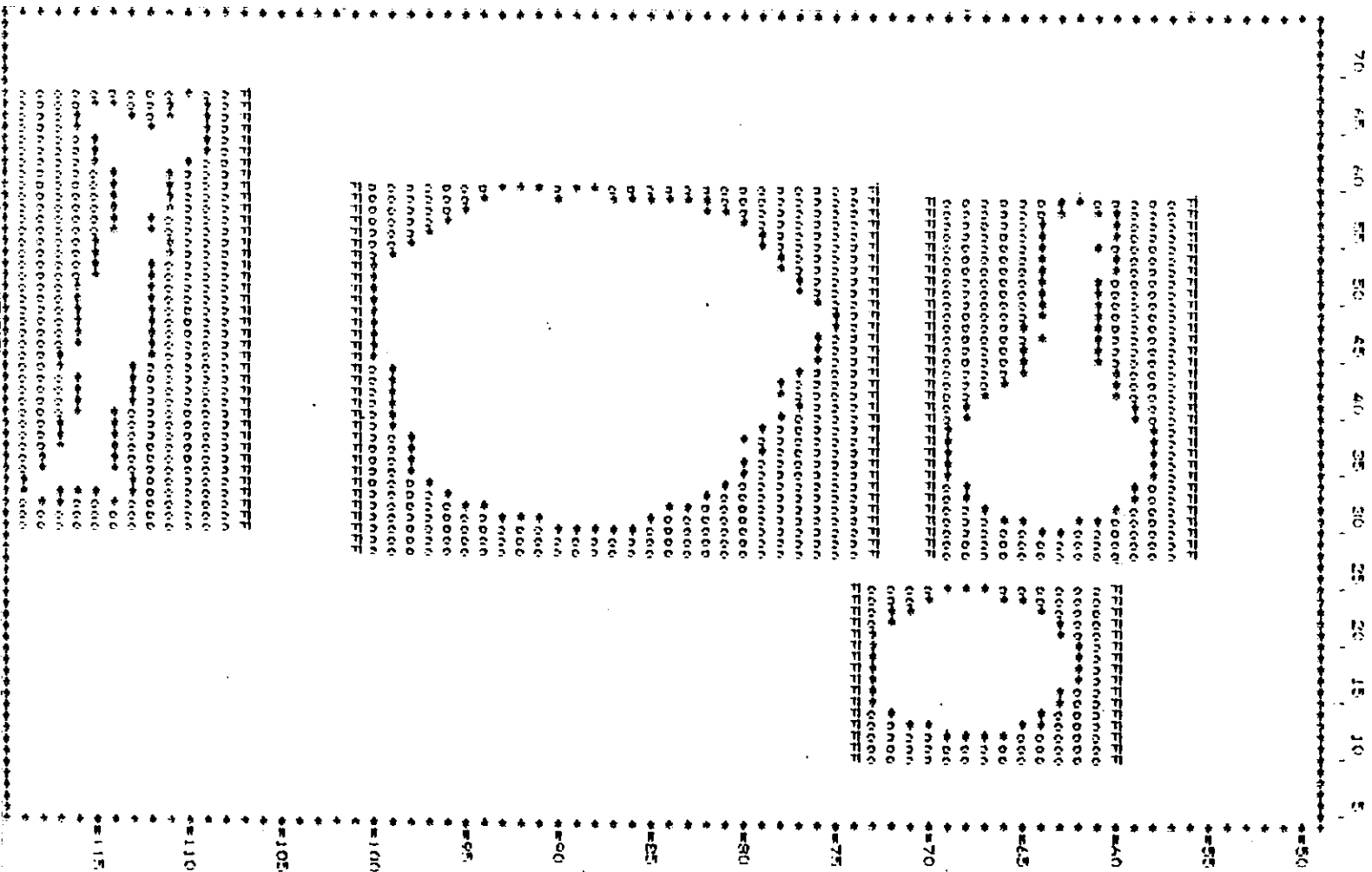


Fig. 4.25 : Search windows for internal points.

- . Next a set of tests are made to establish whether an inside point is a connected point of the internal pattern (Internal segmentation).

The positions of the internal points are expressed in terms of their coordinates (INTX, INTY), referenced to an orthogonal coordinate system centered at the centre of the outline.

Topological Analysis

During the search for points of the internal pattern, all the points visited by the search bug must firstly undergo a topology test, which will classify them as points inside the outline or outside it.

This process is similar in concept to the area-fill techniques used in computer graphics, except of course that the relevant techniques depend on the data-structure of each system, and the particular application (see for example Ref 37)

In our case, the available information consists of the chain list of the outline of the area to be 'filled'. The chain code resides in the image memory in the form of traces, left in the trace fields of the words representing the outline points (Fig.4.26).

This structure is particularly suitable for the extraction of topological information, since it preserves the two dimensional nature of the data.

An additional piece of information, apart from the inscribed chain list, is the clockwise direction of the contour, and the fact that the chain link associated with each point is pointing to the next point in the same direction.

A search bug, that moves always from left to right, and starts from a point outside the contour, should easily, with the aid of a topological flag (INOUT flag), classify all the points along a horizontal scan-line as inside points or outside points, making use of the above mentioned information.

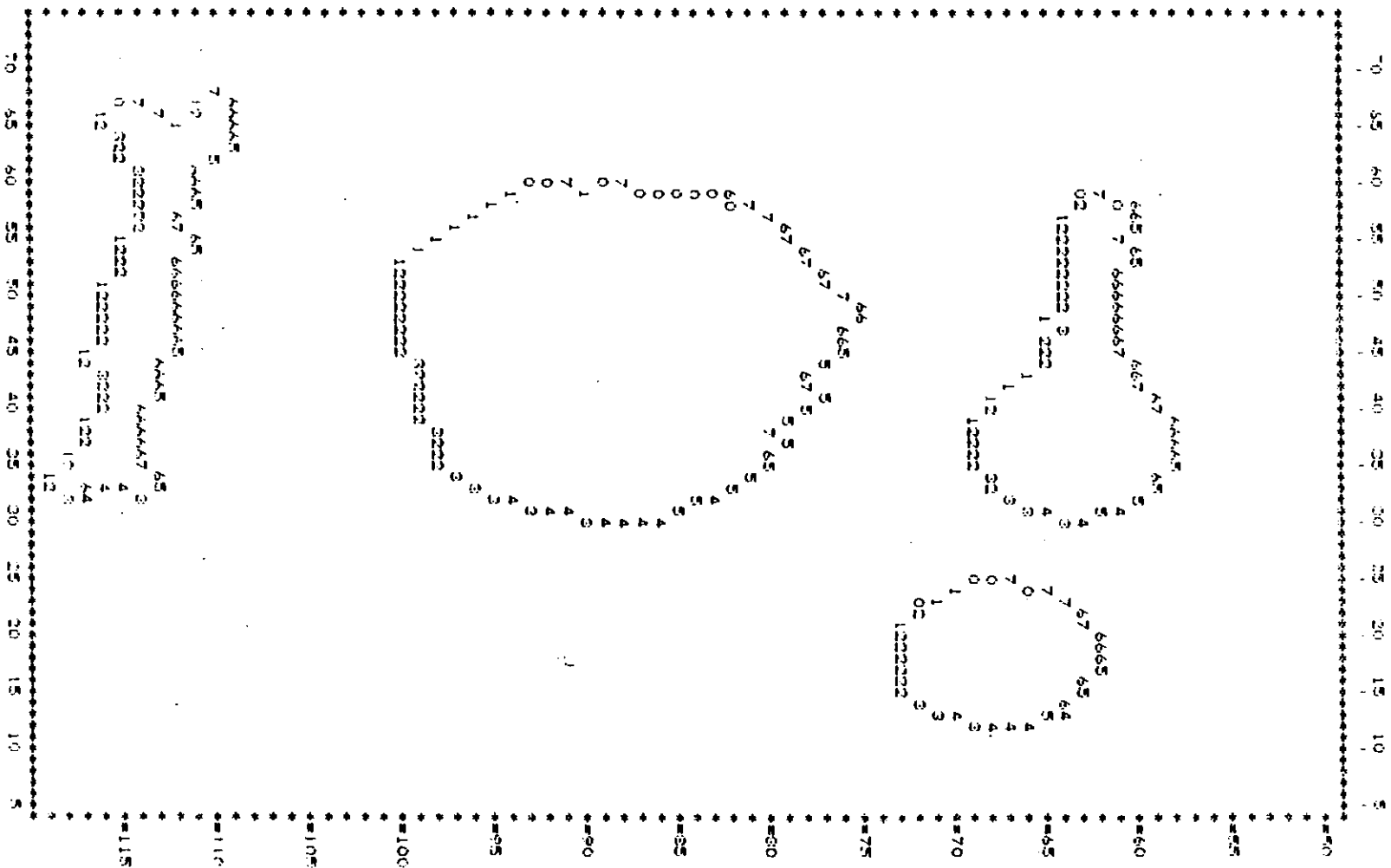


Fig. 4.26 : Chain - code traces used for topological analysis.

A simple method to achieve this would be the following:

- Set initially the INOUT flag to -1 (OUT), and change its status each time you cross the contour (use the flag as a toggle switch). However, a simple scheme like this would fail in cases where the scan line runs tangent to the contour, as in Fig.4.27.

The direction of the contour at the right end of the tangential segment is needed to resolve such cases.

The additional information is available, in terms of the link number of the end point. So, if the search bug does not consider all the points of horizontal contour segments, and only checks the trace field of the last point, it can rightly decide that the next point will be an inside point, if the link number of the last point is 5, 6, or 7, or that the next point will be an outside point, if the link number is a 1, 2, or 3.

Unfortunately, if the direction of the horizontal segment is opposite to the search direction as in Fig. 4.28, the link number of the last point will be 0, in which case the link number of the previous point in the chain will be needed as well, to complete the necessary information about the direction of the contour at that point. In general, for this method to cover all possible cases, the numbers of two successive links C_i , and C_{i-1} , meeting at each point of the contour, are needed.

This problem reveals one drawback of the data structure, namely that the adjacent links of the chain at a certain point are not easily traceable, as in the case of storing the chain in the form of a linear array. However, the major advantage of this structure, the fact that each of the 8 neighbouring points can be easily accessed and checked, can be exploited to provide the solution to the problem, in the following way:

All the 8 by 8 combinations of the directions of two successive links, C_i and C_{i-1} can be listed as in table 4.29; after the elimination of all the impossible combinations, and a suitable grouping, we obtain the following three general cases:

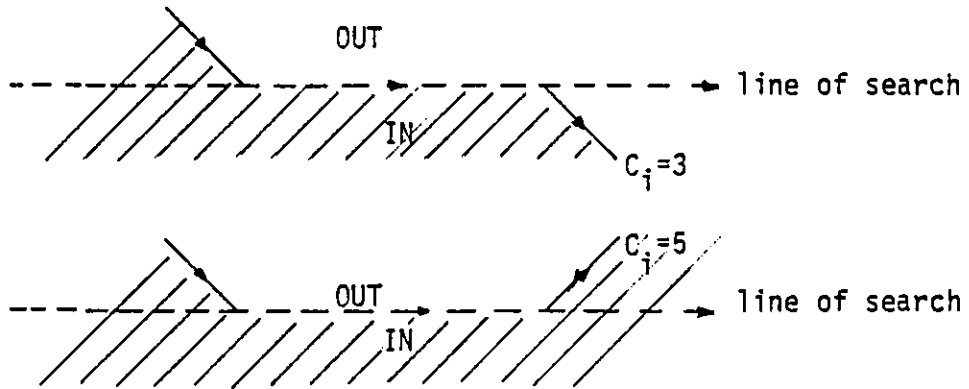
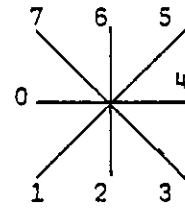


Fig. 4.27 : Search for internal points.

Line of search tangent to the contour in the same direction. The direction of the last link C_i of the tangential segment determines the topology.

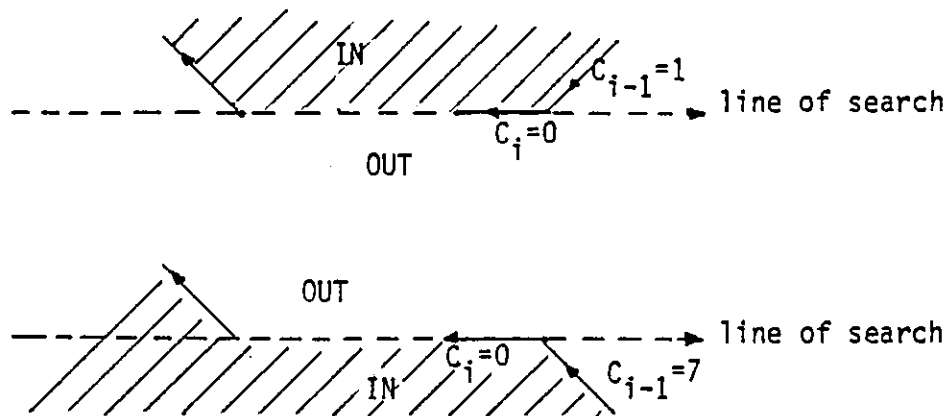


Fig. 4.28 : Search for internal points.

Line of search tangent to the contour in the opposite direction. The direction of the previous link C_{i-1} determines the topology.

$C_{i-1} \backslash C_i$	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

Fig. 4.29 : Grouping of all possible combinations of two successive outline links.

The shaded areas represent impossible combinations (opposite links).

NR signifies Normal Cases, T signifies Tangent Cases,

S signifies Sharp Angle Reflections

* If $C_{i-1} = 0$ or $C_i = 4$ the search will proceed to the next horizontal point.

- a) A 'normal' case, including all combinations of two successive links with similar directions.
- b) A case where the previous or the last link are horizontal (link no = 0 or 4), therefore tangent to the line of search.
- c) A case where the two vectors have almost opposite directions, in such a way that the last vector C_i seems to be a 'reflection' of the previous vector C_{i-1} on the line of search.

All these cases are illustrated in Fig. 4.30.

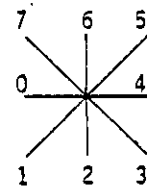
Further study of the resulting situations led to the development of the algorithm of Fig. 4.31, the basic concept of which has already been outlined. The only point that needs some clarification is the way previous links can be traced, as needed to resolve cases b and c.

Since the previous links in all these cases can only originate from the 3 points centered one line above or below the available link according to its direction, and always lead to the current point, a simple way to trace these cases is to check the trace fields of these points for link numbers leading to the current point (see Fig. 4.30c).

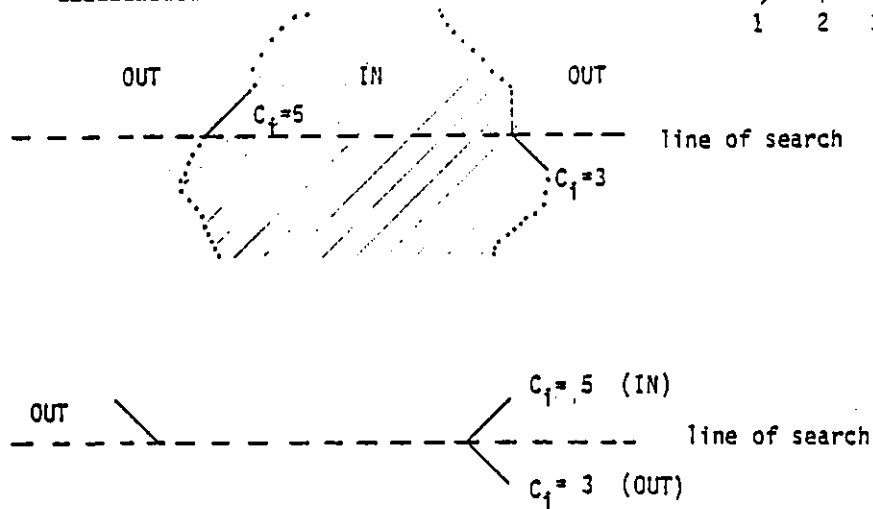
This way all points of the window surrounding the outline can be successfully topologically classified. A minor problem may arise, if the outline is not properly closed (see p.80), leaving a one link gap. In the worst case this gap may cause the loss of one set of line measurements.

A parity check is performed at the end of each line (the INOUT flag at the end of each search line should be at the OUT state), and if it fails, the measurements taken along that line are dismissed.

Finally, a comparison must be made among this technique and other similar techniques that make use of the chain list to locate internal points. By having incorporated the chain list within the image-array, instead of storing it sequentially in a linear array, we avoid the

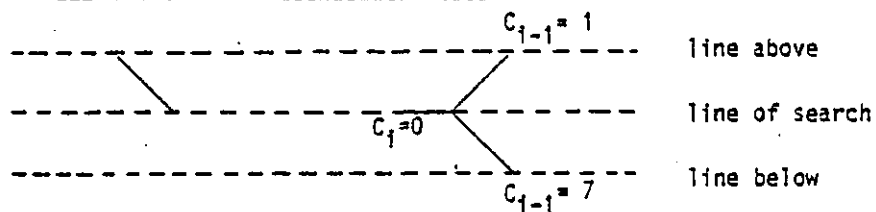


A) NORMAL CASES



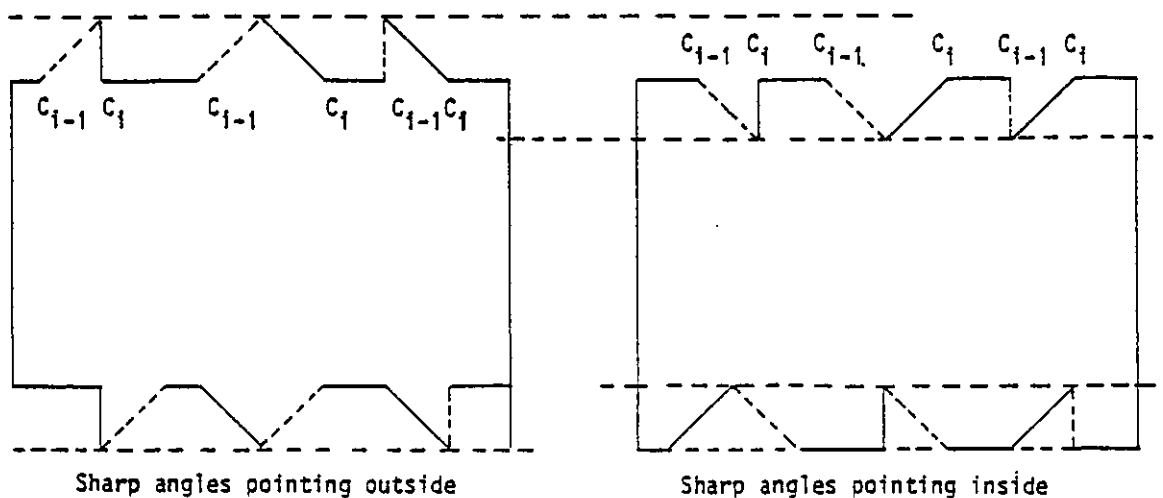
The topology is determined by the direction of the last link (C_i)

B) TANGENT CASES (Opposite direction)



The topology is determined by the direction of the previous link (C_{i-1})

C) SHARP ANGLES (Reflective cases)



The topology is determined by the directions of both links (C_i and C_{i-1})

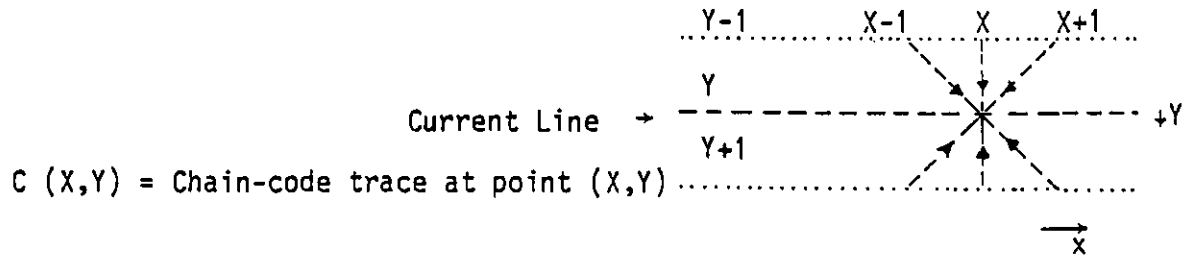
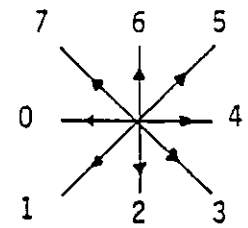
Fig. 4.30 : Examples of the grouped cases of Fig. 4.29

Fig. 4.31 . Algorithm for Topological Analysis

Variables: $C(X,Y)$

$C(X-1, Y-1), C(X, Y-1), C(X+1, Y-1)$

$C(X-1, Y+1), C(X, Y+1), C(X+1, Y+1)$



Step A: Move to next point of current line ($X = X+1$)
and check if it is an outline point $C(X,Y) = 1XXX$
If not ($C(X,Y) = 0XXX$), branch to EXIT1
else,

Step B: If $XXX = 5, 6, 7$ check next line ($Y = Y+1$)

for: a link 1 at $X+1$ $C(X+1, Y+1) = 1001$
or a link 2 at X $C(X, Y+1) = 1010$
or a link 3 at $X-1$ $C(X-1, Y+1) = 1011$

If you find one of the previous conditions, branch to EXIT1
else branch to EXIT2.

If $XXX = 1, 2, 3$ check previous line ($Y = Y-1$)
(If $XXX = 0$, set INOUT flag to +1-IN and follow:)

for: a link 5 at $X-1$ $C(X-1, Y-1) = 1101$
or a link 6 at X $C(X, Y-1) = 1110$
or a link 7 at $X+1$ $C(X+1, Y-1) = 1111$

If you find one of the previous conditions, branch to EXIT1
else branch to EXIT3

EXIT1: Do not change topology flag ($INOUT' = INOUT$)

EXIT2: Set INOUT to 1 (IN)

EXIT3: Set INOUT to -1 (OUT)

* INOUT is initially set to -1 (OUT)

need to go through the whole chain list each time we want to establish the position of a point of the window relative to the outline (Ref.38)

The later method involves a total number of $l \times m \times n$ additions and comparisons, whereas with our method the whole window area can be classified topologically with a number of simple operations of the order of $(l \times m) + n$; where l and m are the dimensions of the search window and n the number of points of the outline.

4.4.3. Internal Segmentation

The objective of this stage is to isolate the internal pattern from the rest of the surface of the object.

The minimum prerequisite for any inside point, in order to be classified as a point of the internal pattern, is that its intensity level must differ from a local intensity level representative of the average reflectivity of the surface.

In other words, a region of the surface can be detected as a separate entity, only if its reflectivity is different from the reflectivity of the surface. The detection of holes, for instance, is based on the fact that their reflectivity is similar to the reflectivity of the background.

So we can have 'black' holes on 'white' objects, or 'white' holes on 'black' objects (as for example in a backlitted scene).

The detection of protrusions can be achieved only if they look like holes (black or white respectively).

The main problem of all binary segmentation techniques is the selection of a suitable threshold. Various methods of automatic threshold selection have been studied. Most of them are based on some kind of histogram analysis (analysis of the distribution of intensity levels over the whole image array).

However, if the scene includes objects with varying reflectivities, the selection of a representative threshold is not always easy.

The situation can become even more complicated if the same global threshold is expected to isolate the objects both from their background and from their internal features.

A set of figures illustrate this difficulty. Fig.4.32 shows the image array of a digitized scene including a coin, a plug, a key and a small spanner. The reflectivity of the spanner is low compared to the reflectivity of the white plug.

The maximum binary threshold that would faithfully reveal the shape of the spanner is $T = 5$. All points with intensity levels less than 5 would thus be classified as 'black' points. This threshold, however, is not high enough to isolate the 'black' details on the surface of the plug (the pins and the screws), as it is illustrated in Fig. 4.33a. If the threshold is raised, say to 8, the spanner will almost disappear (Fig.4.33b), but some more 'black' points will emerge on the surface of the plug.

In our approach, the isolation of the objects from their background was based on gradient information (as described in section 4.2), and not on binary thresholding. This in turn has enabled us to specify a local instead of a global binary threshold, representative of the surface reflectivity of every individual object, that has been isolated by following its contour.

As the contour of the object is being traced, a sample of the average local internal intensity is taken at the vicinity of each new point, and the sum of these samples is divided by the total number of contour points to yield an average intensity level for the surface of the object (see Fig. 4.34).

The representative local levels found for the objects shown in Fig. 4.32, where $E(14)$ for the plug, for the coin and the key, and 5 for the spanner.

A binary threshold of 14 will reveal all the 'black' points on the surface of the plug. But, as Fig.4.35 shows, some more 'black'

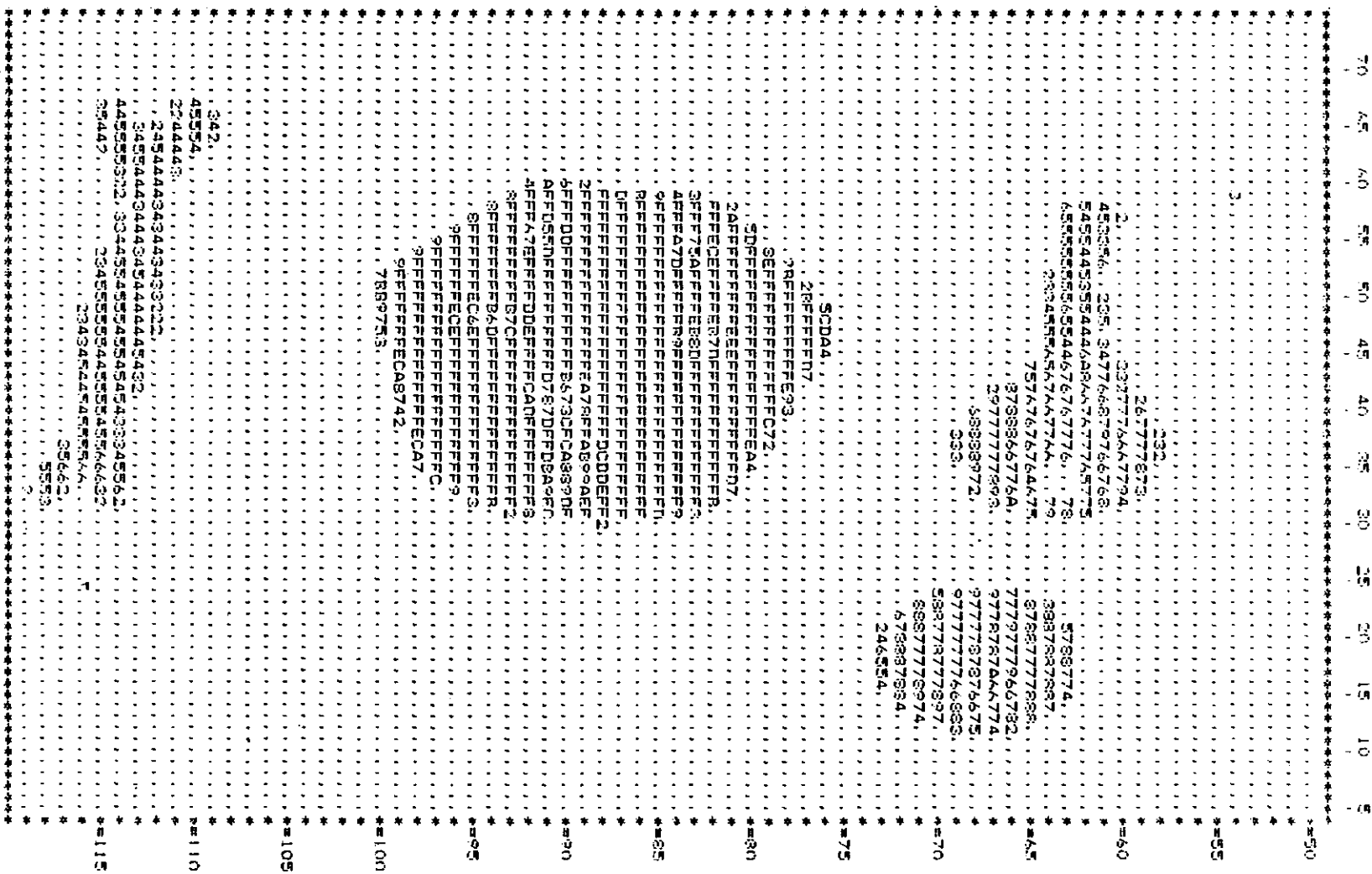


Fig. 4. 32: Image array of a digitised scene displaying the different reflectivities of the viewed surfaces (quantised average intensity levels range from 4 to 15 (F)).

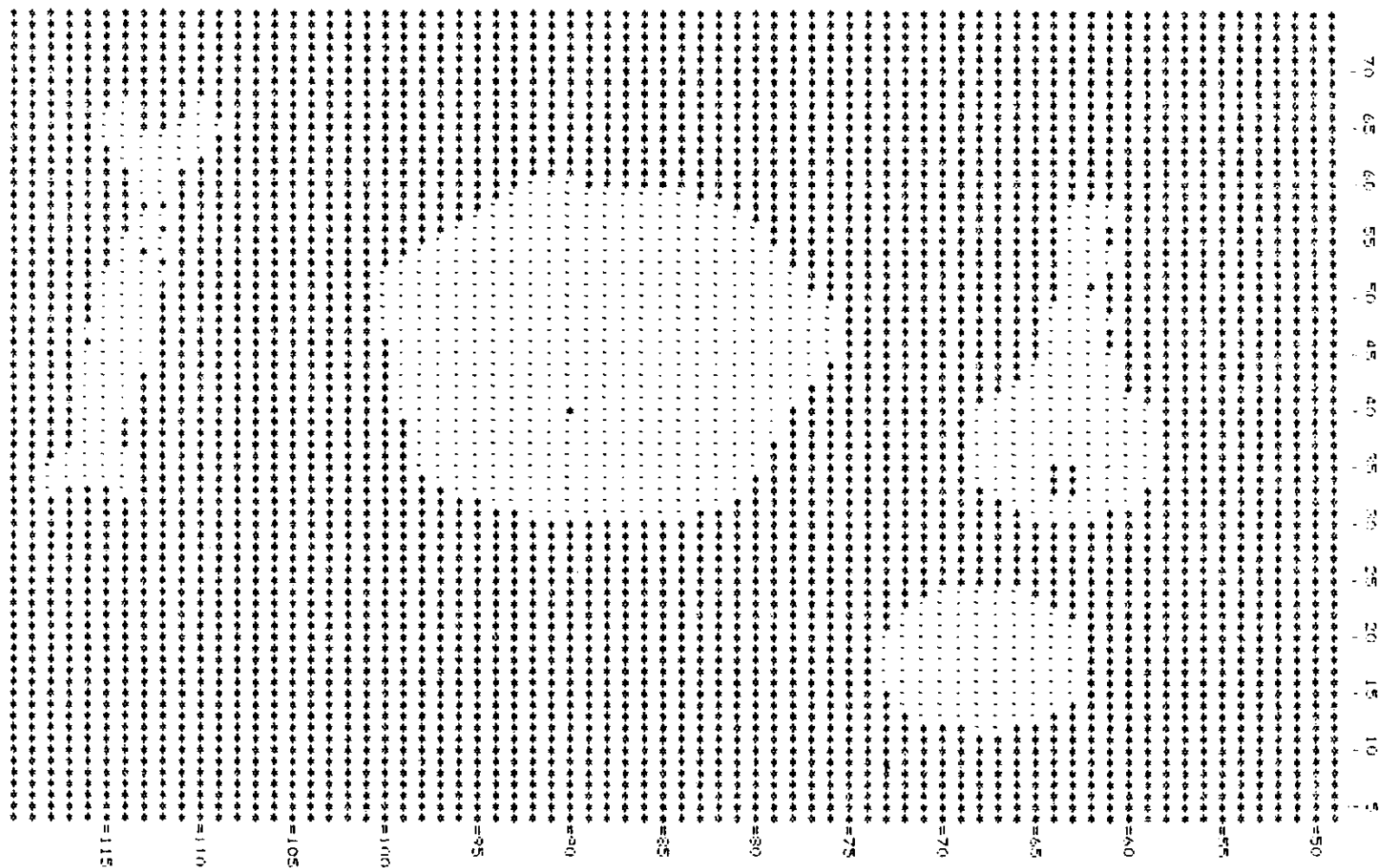


Fig. 4.33 a) Binary image array obtained after thresholding the image array of fig. 4.32 at a threshold of 5.

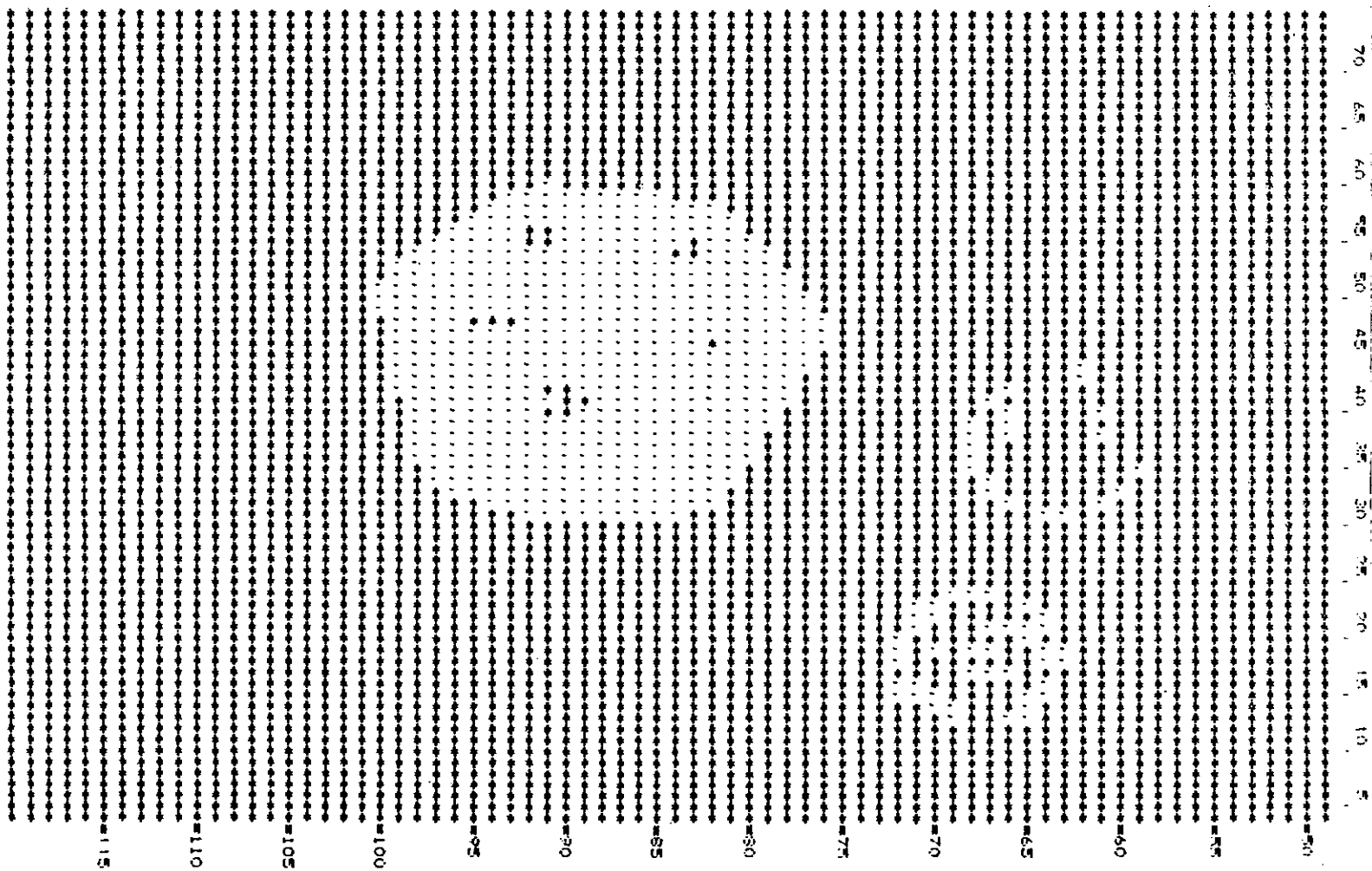


Fig. 4.33b : Binary image array obtained after thresholding the image array of fig. 4.32 at a threshold of 8.

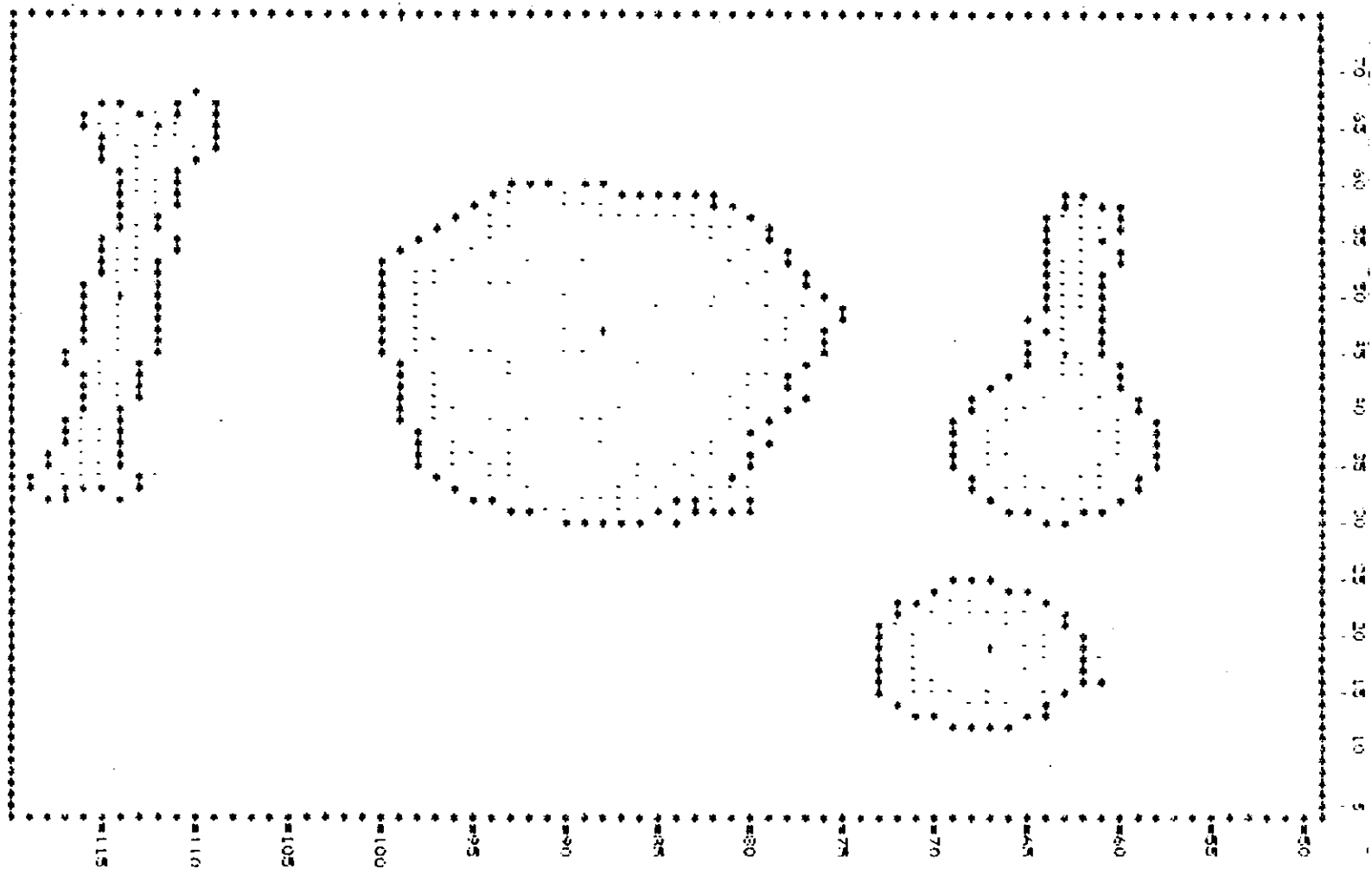


Fig. 4.34 : Sampling the vicinity of the outlines
for the evaluation of the average surface intensity levels.

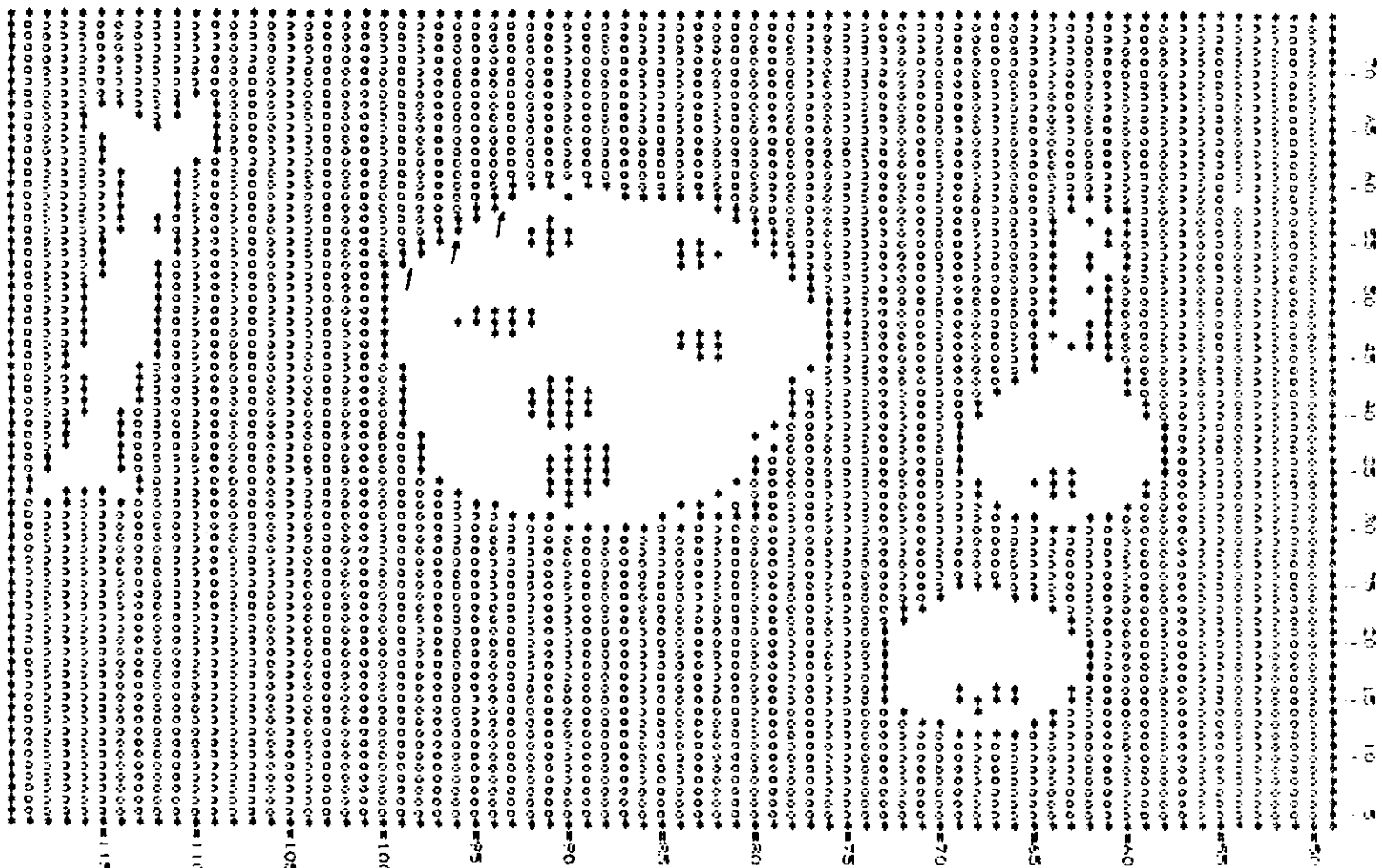


Fig. 4.35 : Binary thresholding of the internal areas at the evaluated surface intensity levels.
Emergence of border points.

points now emerge from the border regions of the objects.

The existence of a border region is based on the fact that the transition area from background intensity to surface intensity is extended to more than one pixels in a direction perpendicular to the contour. Two tests are used for the elimination of the border region. The first is a simple local test, incorporated within the connectivity analysis described below:

The 8 neighbours of each point that satisfies the previous binary test are visited in a circular clockwise sequence, and the trace field of each of them is examined for a code characteristic of an outline point. If such a neighbour is found, the centre point is considered as a border point, and it is excluded from further examination.

This test covers a single-point zone around the outline, and is sufficient for the elimination of the vast majority of border points. Sometimes, however, the border region may extend to more than one points in the vertical direction to the outline.

A second test based on gradient information is used to take care of these points.

The gradient vector is always pointing towards the object area, as shown in Fig.4.36

Therefore a step to the opposite direction of the gradient vector will lead to the outline, if the gradient was estimated at a border point, but it will lead away from the outline, if the gradient is estimated at the edge of a hole.

If the gradient test is applied at a point of the central area of a hole, it will lead nowhere, simply because the gradient is zero at these points. A search for outline points in the 8-neighborhood of the new point will thus discriminate a genuine internal point from a border point.

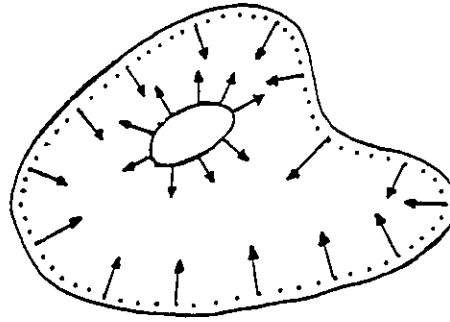


Fig. 4.36: Gradient test for the elimination of border points.

The gradient vectors point to opposite directions. A step to the opposite direction to the gradient vector leads to the outline in the case of border points, and away from it in the case of points on the edge of a hole. The dashed line marks the limit of the border region.

This test is effective but costly timewise, so it is performed after all the other tests, to validate only the otherwise valid points.

The only problem that now remains due to the border region, is the problem of holes that are very close to the border region. The preceeding two tests will exclude the border points as points of interest, but they do not detect them as 'bad' neighbours of other non border points.

This can mislead the detection of points of the perimeter of the internal pattern, but it will not affect the rest of the measurements that are not based on neighborhood considerations.

There are two possible remedies for this situation. One is to 'clean' all the border region during the top-down search, but this may result in eliminating some parts of the internal pattern as well. The other solution is to apply the local test previously mentioned, to all the 8 neighbours of every point of interest. This solution is effective, but it was considered as an unjustifiable time burden.

The final step is to perform a connectivity analysis for each up to now succesful point.

The objectives of this stage are the following:

- a) To eliminate all the remaining 'spurious' points, and consider only the points that are connected in the form of internal blobs,
- b) To provide a means of locating the points on the perimeter of the blobs, and the perimeter links associated with them.
The total perimeter length and the total area of the internal pattern can be evaluated this way, although the individual blobs that form the internal pattern are not located and considered as such.

Our definition of the connectivity of a point is based on the number and the arrangement of its similar neighbours within the basic

3 by 3 cell.

We assume that two points are connected, if they are similar (both 0's or both 1's), and if they are horizontally, vertically, or diagonally adjacent (8-connectivity as defined in Ref.44)

We define a neighborhood of 8 points as a basic loop, if all the similar points of the neighborhood can be connected in a closed path, advancing in a clockwise direction and visiting each point only once.

Some examples of basic loops are illustrated in Fig.4.37
This definition is introduced in order to distinguish internal points that are connected in blobs from noise points that are loosely connected.

An internal point is considered valid, if it is the centre of a neighborhood that forms a basic loop.

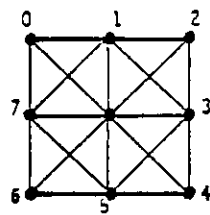
Furthermore, if the centre point is also a member of the loop, then it is a perimeter point, and the link leading from it to the next member of the loop is a perimeter link (see Fig. 4.37).

This selection rule automatically excludes:

- a) All isolated points, or points with only one similar neighbour (at least two neighbours are needed to form a loop).
- b) All loosely connected points, that are considered as 'noise' points (some examples of such points are illustrated in Fig. 4.38).
- c) Parts of lines or curves, that are considered irrelevant, because they may represent scratches or other non 'solid' internal features.

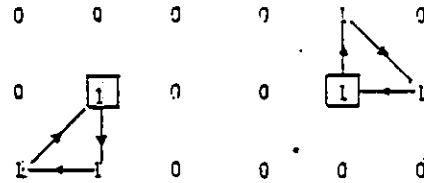
The elimination of linear segments from the pattern contradicts to a certain extent to the inclusion of small holes in the pattern. If small holes (holes consisting of 3-5 points) are to be considered consistently, i.e. independent of their orientation, then parts of lines at 45° to the x-axis, will also be considered.

Fig. 4.37 : Examples of basic loons

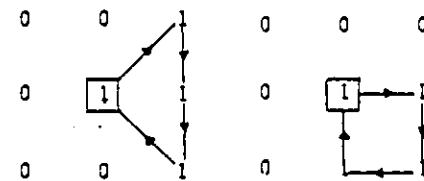


8-neighborhood
 =====

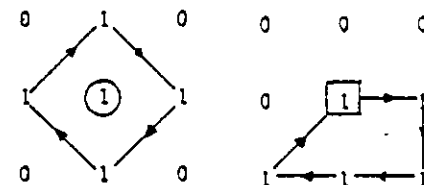
The 8 neighbours are enumerated in a clockwise sense. All possible links are shown.(8-connectivity)



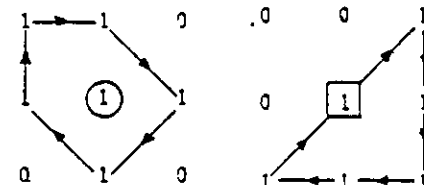
2 similar neighbours
 Centre point always
 a perimeter point.



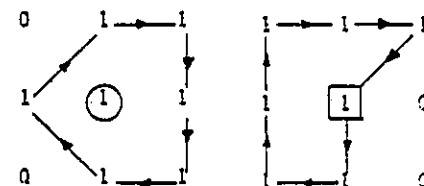
3 similar neighbours
 Centre point always
 a perimeter point.



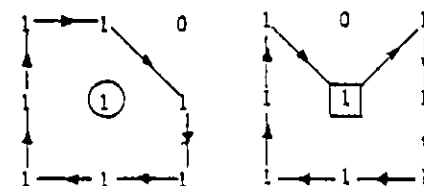
4 similar neighbours



5 similar neighbours



6 similar neighbours



7 similar neighbours

○ = Core point
 □ = Perimeter point

This ambiguity stems from the rectangular quantization of the image, and is illustrated in Fig. 4.39.

To distinguish between a small slanted rectangular hole and a line segment, a larger neighborhood must be investigated, but this is beyond the scope of a simple noise cleaning algorithm as the one described.

The actual algorithm that performs the connectivity analysis is explained diagrammatically in Fig. 4.40.

The basic idea is to count the number of 0 to 1 and 1 to 0 transitions as you scan the 8 neighbours in a loop around the centre point ($0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7$).

An 8-bit word representing the arrangement of the 8 neighbours in the loop is formed by shifting into it the binary values of the respective points of the loop.

This neighborhood word is then read back by shifting it into the carry bit. The carry bit C represents pixel p_{i-1} ; bit 7 of the word represents p_i , bit 6 represents p_{i+1} , and so on.

In order to include the 0 to 6, 6 to 4, 4 to 2 and 2 to 0 links, that are not provided by a peripheral path like this, if a 1 to 0 transition occurs at points 1, 3, 5, 7 the next pixel p_{i+1} (bit 6 of the neighborhood word) is tested for a 1, and a link is formed accordingly (no transitions are recorded).

Finally, if the recorded number of transitions is higher than 2, the point is rejected as loosely connected. If the number of transitions (TRCN) is 2, the point is classified as a perimeter point. If no transitions were recorded (TRCN = 0), the point is classified as an interior point of a blob (a core point).

The algorithm also returns the total number of similar neighbours (CON), and the link number of the point where the first 0 to 1 transition was recorded (POSIND).

We must stress at this point that this noise cleaning algorithm, formulated on the assumption of connectivity, has a more general

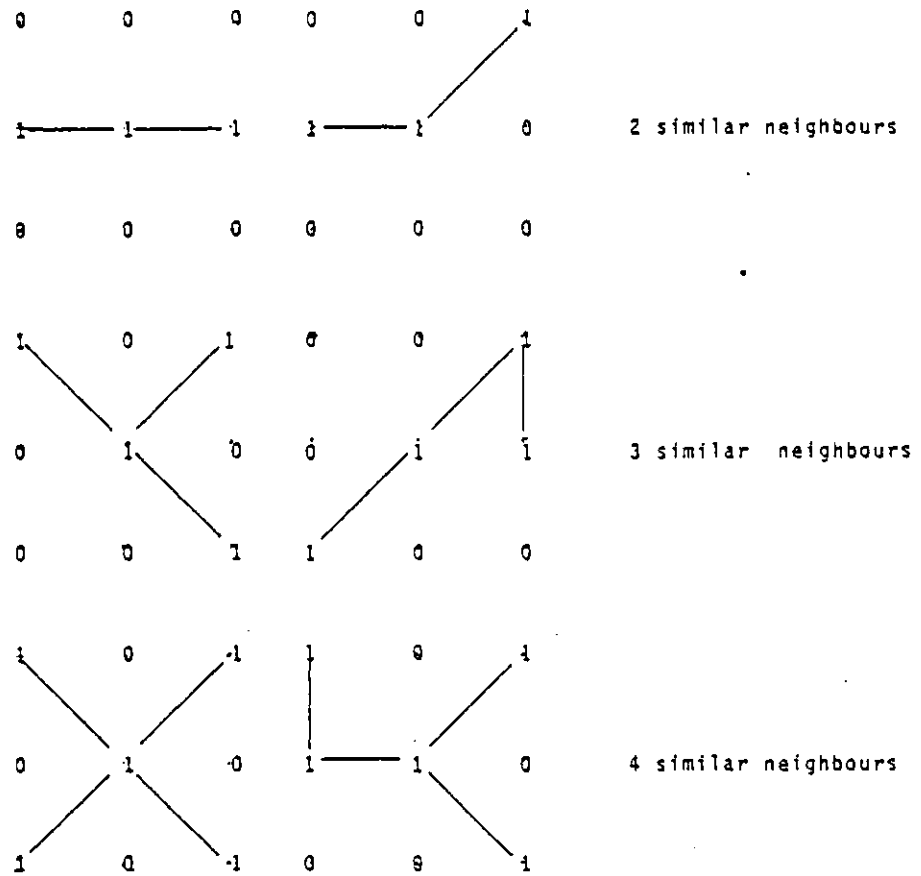


Fig. 4.38 : Examples of loosely connected points.

Note that closed paths cannot be formed.

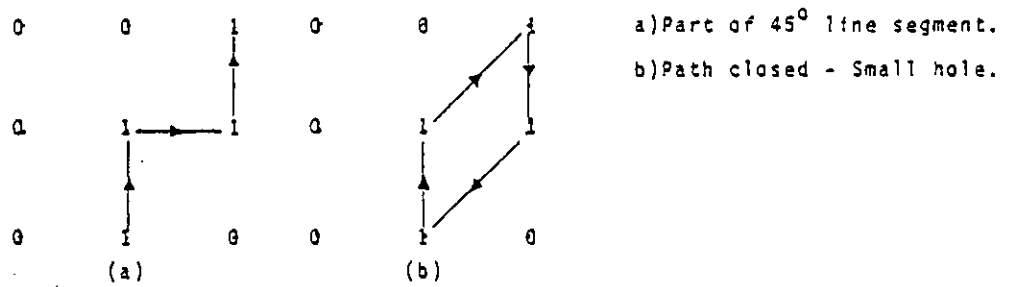
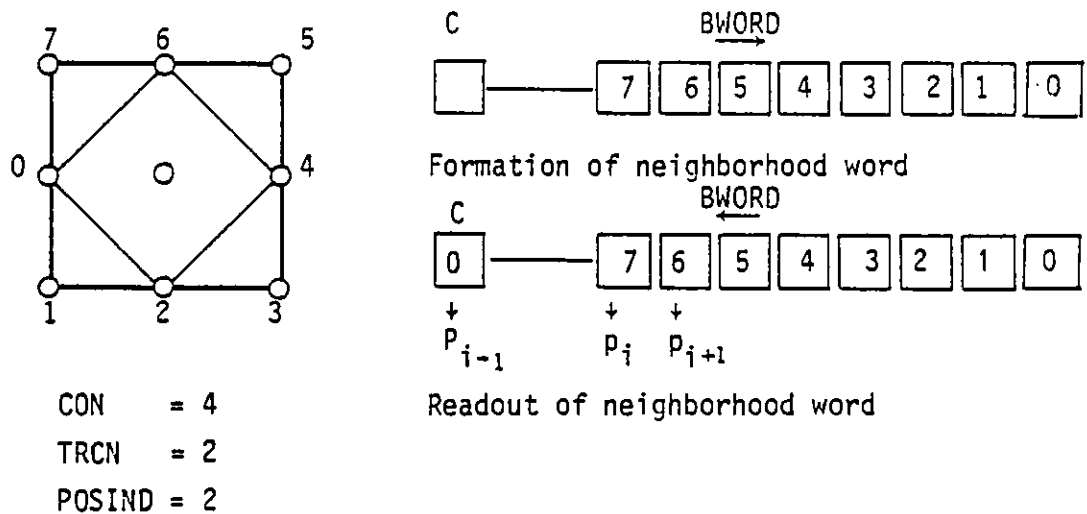


Fig. 4.39 : Ambiguity arising from considering small holes.

Note that the same neighborhood can be interpreted either as a part of a line (loosely connected component) or as a small hole (connected component).

Fig. 4.40 . Connectivity Analysis Algorithm



- Algorithm (Block Description)

Variables: P (I) / I = 0, 1, 2, ..., 7 Input binary variable
BWORD Local binary variable
CON, TRCN, POSIND Output variables

STEP A: Form neighborhood word by shifting binary thresholded values of 8 neighbours into it.
If the trace field of any of the neighbours = 1XXX, then the centre point is touching the outline → Report touching.
As you shift, count the number of similar neighbours (CON).

STEP B Read back BWORD counting the number of 0 to 1 and 1 to 0 transitions (TRCN)
For i = 1, 3, 5, 7
check p_{i+1} (bit 6 of BWORD)
If $p_{i-1} = 1$, $p_i = 0$ and $p_{i+1} = 1$
do not record two successive transitions
(form 0 to 6, 6 to 4, 4 to 2 and 2 to 0 links)
Record position i of first 0 to 1 transition (POSIND)

significance in terms of speed and efficiency, compared to other existing regularization or logical smoothing operations (see for example Ref. 39,p.276).

4.4.4. Description of the Internal Pattern (Primitive Features)

For each point of the internal pattern the following measurements are taken:

- . Its position (INTX, INTY) with respect to an orthogonal coordinate system centered at the centre of the outline.
- . The number of transitions in its neighborhood (TRCN), the total number of its similar neighbours (CON), and the link number of the first 0 to 1 transition (POSIND), which coincides with the perimeter link at the current point (if it is a perimeter point).

The following primitive features of the internal pattern can be evaluated from these measurements:

- A) Total number of points (INNUM)
Total perimeter length (PERH)
- B) Total Area of the pattern (AREAH) - Zero order moment.
Coordinates of the centre of the pattern (INCX, INCY).
Moments of Inertia (IXH, IYH, IXYH) - Second moments.
- C) Average radius from the centre of the outline to all the points of the pattern (DSH).

The calculation of INNUM, INCX, INCY, IXH, IYH, IXYH, and DSH is straightforward.

The algorithms for the calculation of total perimeter length (PERH) and total area (AREAH) are described in fig's 4.41 and 4.42.

Fig.4.41: Algorithm for the evaluation of total perimeter length
of the internal pattern

VARIABLES : TRCN = Number of 0 to 1, or 1 to 0 transitions

POSIND = Link number of next perimeter link

PERH = Total Perimeter Length

Values: TRCN = 0 or 2 POSIND = 0 to 7

ALGORITHM

STEP A : If TRCN = 2, then PERH = PERH+8, else RETURN

STEP B : If POSIND= even, then PERH= PERH+3

RETURN

Finally PERH= 11/8 PERH

The value of $\sqrt{2}$ is approximated by the fraction 11/8

EXAMPLE

8 neighborhood

0	1	2	1	0	0	TRCN= 2
7	*	3	1	*	0	POSIND= 4
6	5	4	1	1	1	PERH= PERH + 11/8

Fig. 4.42 . Algorithm for the evaluation of the total AREA of the internal pattern

Basic Idea:

- A point inside a hole (a core point) contributes a unit square to the total area, irrespective of the number of its neighbours (7 or 8).
- A point at the perimeter contributes a number of octants ($1/8$) to the total area, depending on the number of its neighbours (C).

Since all neighbours are ordered in a loop and every pair of consecutive links from the centre point to two adjacent neighbours creates a new octant, the number of octants is equal to the number of neighbour pairs (C-1)

Therefore, the contribution of a perimeter point, having C similar neighbours, will be:

$$\Delta A = C-1/8$$

Algorithm

Variables: TRCN = Number of transitions (0 or 2)
 CON = Number of similar neighbours
 (2 to 8)

If TRCN = 0, then AREA = AREA+8
 else AREA = AREA+(CON-1)

Finally: AREA_H = AREA_H/8

0	0	0	0	0	0
0	0	1	1	0	0
0	1	1	1	1	0
0	1	1	1	1	0
0	0	1	1	0	0
0	0	0	0	0	0

Binary image of a hole

		2	2		
	2	0	0	2	
	2	0	0	2	
		2	2		

0 → core point
 2 → perimeter point

Same image showing numbers of transitions at each point

		4	4		
	4	7	7	4	
	4	7	7	4	
		4	4		

Same image showing numbers of similar neighbours

C = 4
 $\Delta A = \frac{C-1}{8} = \frac{3}{8}$

Contribution of a perimeter element to the total area

4.5. Orientation

The orientation of an object resting on a plane can be defined in several ways. Two simple and popular definitions are the following:

- a) The orientation is defined by the relative positions of an ordered set of characteristic points. An ordered pair of such points constitutes the minimum required set.
The absolute positions of the intersections of a predefined circle with the outline of the object can form such a set (Ref. 25)
- b) Another definition of the orientation could be formed in terms of the direction of a characteristic axis of the object, usually one of the principal axes.

Various other definitions have been proposed, such as those based on the relative frequency of occurrence of all the directions of the edge points (Ref. 22), or those based on the coefficients of a Fourier analysis of the outline (Ref. 40)

The direction of the axis of least second moment was chosen in our case, as an orientation identifier.

If we assume that the object is in a reference position (orientation angle $\phi = 0^\circ$), when its axis of least moment of inertia coincides with the x-axis of the image coordinate system, then $I_{x_p} = \text{minimum}$, $I_{y_p} = \text{maximum}$, and $I_{x_p y_p} = 0$.

(I_{x_p} , I_{y_p} , $I_{x_p y_p}$ are the moments of inertia calculated with reference to the principal axes).

If the object is now rotated from this reference position at an angle ϕ (see Fig. 4.43), the new values of I_x , I_y and I_{xy} will depend on the values of I_{x_p} , I_{y_p} , $I_{x_p y_p}$, and the angle of rotation ϕ , in the following manner:

$$\begin{aligned}
 I_y &= I_{y_p} \cos^2 \phi + I_{x_p} \sin^2 \phi + 2 I_{x_p y_p} \sin \phi \cos \phi \\
 I_x &= I_{y_p} \sin^2 \phi + I_{x_p} \cos^2 \phi - 2 I_{x_p y_p} \sin \phi \cos \phi \quad (9) \\
 I_{xy} &= (I_{y_p} - I_{x_p}) \sin \phi \cos \phi - I_{x_p y_p} (\cos^2 \phi - \sin^2 \phi)
 \end{aligned}$$

or, since $2 \sin \phi \cos \phi = \sin 2\phi$, $\cos^2 \phi - \sin^2 \phi = \cos 2\phi$, and $I_{x_p y_p} = 0$ we get:

$$\begin{aligned}
 I_y - I_x &= (I_{y_p} - I_{x_p}) \cdot \cos 2\phi \\
 2 I_{xy} &= (I_{y_p} - I_{x_p}) \cdot \sin 2\phi \quad (10)
 \end{aligned}$$

and finally:

$$\boxed{\tan 2\phi = \frac{2 I_{xy}}{I_y - I_x}} \quad (11)$$

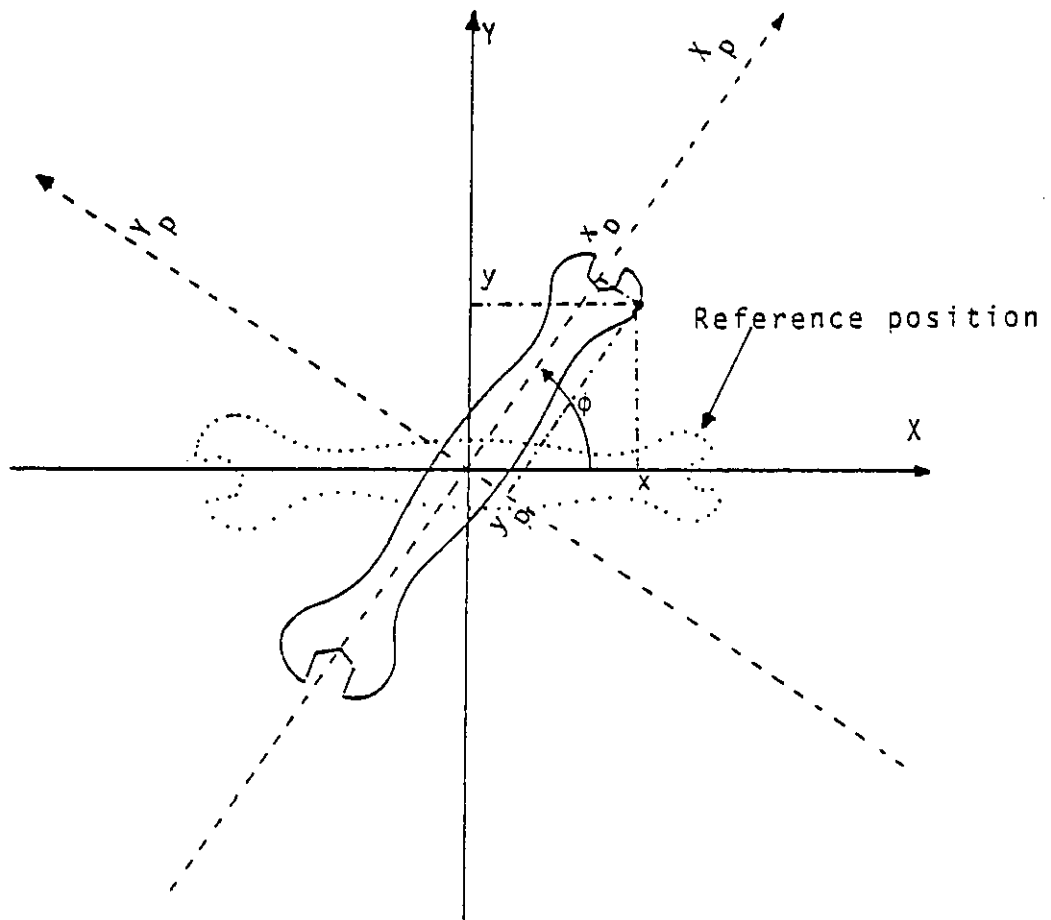
$$\text{yielding } \phi = \frac{1}{2} \operatorname{atan} \frac{2 I_{xy}}{I_y - I_x} \pm k \frac{\pi}{2}, \quad k = 0, 1, 2, \dots \quad (12)$$

This formula implies a 90° ($\pi/2$) ambiguity in the definition of the orientation. However, by making use of the sign relationships in (10), ($I_{y_p} - I_{x_p} > 0$ by definition), we can reduce this ambiguity to 180° , which is the inherent ambiguity of the method (the sense of direction does not enter in the definition of the reference position), as illustrated in Fig. 4.43.

For objects with at least two perpendicular axes of symmetry, this ambiguity is irrelevant, as far as the orientation of their outlines is concerned.

The position of the centre of the internal pattern for objects that have internal features, can resolve this ambiguity, provided that it does not coincide with the centre of the outline.

The orientation angle is evaluated within the range $[-\pi/2, \pi/2]$.



$$x = x_p \cos \phi + y_p \sin \phi$$

$$y = x_p \sin \phi - y_p \cos \phi$$

Fig.4.43 : Definition of Orientation by the direction of the axis of Least Moment of Inertia (LMI axis).

CHAPTER 5 : CLASSIFICATION AND LEARNING PROCEDURES

5.1. Theoretical Background (Statistical Pattern Recognition)

The objective of automatic classification of a part viewed by an industrial vision system is a common objective of many pattern recognition applications that draw their data from 2-D image arrays.

Of the many areas of interest, the pictorial domain has dominated the pattern recognition scene since its early days. Contributions to its growth have come from many disciplines, including statistics, communications theory, ~~control~~ theory, operational research, biology psychology, linguistics and computer science. Early research in pattern recognition related fields in the 1960's was mainly directed to theoretical aspects, particularly the modelling of perception and intelligence found in living systems. The exciting 'bionic' models continued to attract attention through the 1960's, but it was quickly realised that succesful recognition techniques can be applied by machines that have nothing in common with biological systems. The lack of a complete theory of perception has not prevented people from trying to solve more modest problems and actually achieving reasonable and usefull solutions.

The most commonly used abstract pattern recognition model is shown in Fig. 5.1 (Ref.41 ch. 20)

Classical pattern recognition systems are designed to classify an input pattern into one of several classes, by following the so called bottom-up method of successive data reduction stages. The feature extractor extracts relevant information from the input pattern. Sets of these features can be viewed as points in some N-dimensional feature space. The objective of the classifier is to further reduce the dimensionality of the problem by partitioning the feature space into regions, one region for each class.

This objective can be accomplished during the learning phase where the system extracts features from a statistical sample set.

If the measured features during this stage (prototype data) tend to cluster into well defined and separated groups according to the specified classes, then an effective classifier can be designed.

When a perfect partitioning cannot be accomplished, the classifier is designed so as to minimize the probability of error. The actual classification takes place during normal operation when a similar set of features is extracted for every input pattern, compared to the prototype data, and a decision is taken about the identity of the input pattern.

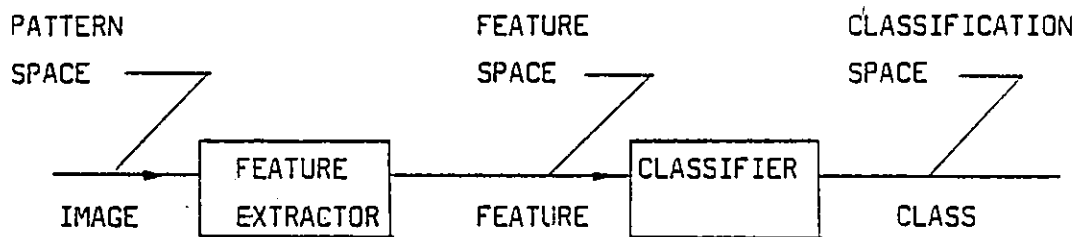


Fig. 5.1 - Pattern Recognition Model.

The problem of classification thus formulated becomes a problem in statistical decision theory, itself a handsome field with applications in many different areas.

The classical Bayesian decision theory provides a good framework for the theoretical formulation of many pattern recognition problems (Ref. 39 , ch. 2). The basic assumption underlying statistical classification is that there exists a multivariate probability distribution for each class (class conditional probability). Members of a pattern class are then treated as samples from a population, distributed in the n -dimensional feature space, according to the assumed distribution.

If the forms of the underlying class conditional density functions are known or can be estimated from the training sample set, then a

conditional risk $R(a_i/x)$, i.e. the probability of making a wrong decision a_i , based on an observation x (a measured feature vector), can be evaluated as a function of x , and a decision rule that minimizes the conditional risk can be found.

Statistical knowledge of the distributions range from:

- a) Known distributions.
- to b) Families of distributions with known functional forms, but with a finite set of unknown parameters.
- to c) Unknown distributions.

Complete knowledge of the distributions rarely if ever occurs in pattern recognition applications.

In a typical case the available information consists of some general knowledge of the situation, which can be expressed in terms of general assumptions and a number of samples that can be used for the design of the classifier (learning procedure). If the samples are labelled, then we call this learning procedure supervised learning, or learning with a teacher. If unlabelled samples are available, an unsupervised learning procedure must be followed.

If general assumptions about the forms of the distributions can be made, and furthermore these forms can be parametrized, then several procedures can be followed to estimate the parameters from the available samples (parametric classification), from which the most common is the maximum likelihood estimation. Density estimation can also be achieved in certain cases without any knowledge about the forms of the distributions (non parametric, or distribution free classification).

This category of classification procedures has attracted a lot of attention in recent years. (Ref. 42). A well-known family of non parametric techniques is the family of nearest neighbour classifiers. (NNC).

All the above mentioned techniques are based on the assumption that all the components of the feature vector are measured before a

decision can be made. Another alternative is to use a decision tree, evaluating the features in a predetermined sequence until a decision can be made. The most common procedure is to follow a sequence of binary decisions (two choices at each step) based on selected features that provide a good separation of the specified choices (binary decision trees).

Most of the proposed industrial vision systems that employ statistical classifiers use either a simplified NNC, or a binary decision tree (Ref. 36)

The nearest-neighbour rule can be formulated as follows (Ref. 39, ch. 4).

Let $S^n = \{x_1, \dots, x_n\}$ be a set of n labelled samples, obtained in a supervised learning phase (the whole set of training samples).

Let x be an observed vector and $x_k \in S^n$ the sample nearest to x .

The decision rule is to assign x the label associated with x_k . It follows immediately from this formulation that the complete set of training samples must be stored, and must be searched each time an observed feature vector is to be classified. The storage requirements and the cost of applying a NNC in its original form are prohibitive in the case of industrial vision systems. Several techniques for reducing the problem have been suggested (see for example Ref. 43).

The general idea is to find a small reference set representative of the sample data.

This simplification can still result in reliable classification, if the various classes are well defined and well separated, which among others involves the selection of discriminative and consistent features.

However, if the set of objects of interest includes similar objects, described in terms of features that are evaluated from a low-resolution grid, then the probability of classification errors can

increase to unacceptable values.

In our case an attempt was made to tackle the problem of similar classes within the framework of randomized decision theory.

The basic assumption made is that the class-conditional probability densities are multivariate normal densities with unknown parameters.

Quoting from (Ref. 39 , p.22)

"Of the various density functions that have been investigated, none has received more attention than the multivariate normal density.

It must be confessed that this attention is due largely to its analytical tractability. However, the multivariate normal density is also an appropriate model for an important situation, viz., the case where the feature vectors x for a given class w_i are continuous valued, mildly corrupted versions of a single typical or prototype vector μ_i . This is what one would expect if the feature extractor were intentionally designed to extract features that were different for patterns in different classes but as similar as possible for patterns in the same class".

One could also add to these properties of the Multivariate Normal Density (MND) the fact that its parameter vectors, namely the mean vector and the covariance matrix, constitute the minimum representative data description set for any sample set, irrespective of its underlying distribution.

The sample mean $\bar{x}$ can be thought as the single point that best represents all of the data in the sense of minimizing the sum of squared distances from $\bar{x}$ to the samples.

The sample covariance matrix tells us how well the sample mean describes the data in terms of the amount of scatter that exists in various directions.

In that sense the same parameters form a natural representation of a sample set in its most condensed form, and can be used as normalizing factors in distance estimations for an ultimately condensed nearest neighbour classifier.

The general Multivariate Normal Density is written as:

$$p(x) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp \left[-\frac{1}{2} (x-\mu)^t \Sigma^{-1} (x-\mu) \right] \quad (13)$$

where:

x is a d -component column vector,
 μ is the d -component mean vector,
 Σ is the d -by- d covariance matrix,
 $(x-\mu)^t$ is the transpose of $x-\mu$,
 Σ^{-1} is the inverse of Σ ,
and $|\Sigma|$ is the determinant of Σ .

If the measured features x_i are assumed statistically independent, then the covariance terms σ_{ij} of Σ are zero. A convenient set of discriminant functions $g_i(x)$ can be defined for the normal density in the following form:

$$g_i(x) = \log p(x/w_i) + \log P(w_i) \quad (14)$$

where $P(w_i)$ denotes the a priori probability that an observed object belongs to class w_i . A priori probabilities represent our prior knowledge of how likely it is to observe an object that belongs to class w_i . We can assume that all the a priori probabilities $P(w_i)/i=1,2,\dots$ are equal, in which case the discriminant functions can be written only in terms of the class-conditional densities $p(x/w_i)$:

$$g_i(x) = \log p(x/w_i) \quad (15)$$

Then, from Eq. (13)

$$g_i(x) = -\frac{1}{2}(x-\mu_i)^t \Sigma_i^{-1} (x-\mu_i) - \frac{1}{2} \log |\Sigma_i| - \frac{d}{2} \log 2\pi \quad (16)$$

The classification rule is simply to assign a feature vector x to class w_i if :

$$g_i(x) > g_j(x) \quad \text{for all } j \neq i \quad (17)$$

The choice of discriminant functions is not unique. We can multiply them by a positive constant or bias them by an additive constant. We can therefore eliminate the last term of Eq. (16) since it is a constant term.

$$\text{The term } \log |\Sigma_i| = \log \prod_{i=1}^d \sigma_{ii}^2$$

measures the hyperellipsoidal scattering volume of the cluster representing class w_i and it can be assumed to obtain similar values for all the classes.

The most significant terms of the discriminant functions described by Eq. (16) are the class dependent terms involving the main parameter vectors μ_i and Σ_i , and the observed feature vector x . We can therefore neglect the scattering volume terms and define $g_i(x)$ as:

$$g_i(x) = - (x - \mu_i)^t \Sigma_i^{-1} (x - \mu_i),$$

$$g_i(x) = - \sum_{j=1}^d \frac{(x_j - \mu_{ij})^2}{\sigma_{ij}}$$

(17)

where: x_j is the observed value of feature j ,
 μ_{ij} is the mean value of feature j in class i ,
 σ_{ij} is the mean variance of feature j in class i ,
and d is the number of features.

This form of the discriminant functions provides a further insight of the classification operation, and it can be interpreted in two different ways:

a) The classifier can be described as a minimum distance classifier based on the MND assumption and the assumption of statistical independence of the measured features. According to this interpretation, the classification of a feature vector x is based on the measurement of the normalized Euclidean distances $r_j^2 = \sum_{j=1}^d \frac{(x_j - \mu_{ij})^2}{\sigma_{ij}}$ from x to each of the

mean vectors, and the assignment of x to the class of the nearest mean. These normalized distances are sometimes called the squared Mahalanobis distances.

b) The classifier can be described as an ultimately condensed nearest neighbour classifier, where the whole sample set is reduced to a minimum number of representative samples (the mean vectors), and the feature distances from these vectors are normalized by the standard deviations of the corresponding features, so that the more reliable features (features with less spread) count more in the classification process.

For simplicity we can abbreviate the metric of Eq. 17 by writing

$$g_i(x) = -d(x, \mu_i) \quad (18)$$

The objective of all classifiers that are designed according to the Bayes decision theory is to minimize an expected risk function R by making those decisions that incur the minimum loss. We can define a conditional risk R associated with a decision a_i as:

$$R(a_i|x) = \sum_{j=1}^C \lambda(a_i|w_j) P(w_j|x) \quad (19)$$

where $\lambda(a_i|w_j)$ is the loss incurred for deciding a_i when the right decision is w_j , and $P(w_j|x)$ is the a posteriori probability that the observed feature vector x belongs to class w_j . In the case of an industrial vision system, as in many other P.R. applications it is beneficial to introduce a reject option, in other words to introduce an additional class of unknown or unrecognizable patterns.

In most industrial applications the cost of a misclassification is usually considerably higher than the cost of a reject or no decision, since rejects can always be recycled.

A simple version of the decision rule involving a reject option can now be formulated as:

Accept the pattern x and identify it as of the k th pattern class whenever

a) $d(x, \mu_k) \leq d(x, \mu_j)$ for all $j = 1, 2, \dots, c$

where all classes are assumed to have an equal a priori probability, and the number of classes is c .

and

(20)

b) $d(x, \mu_k) \leq r_t$

where r_t is a threshold distance which we choose to call the reject threshold.

The statistical decision theory approach to P.R. has been criticised for focusing entirely on statistical relationships among scalar features and not considering other structural properties that characterise patterns.

These criticisms led to various proposals for the so called linguistic models for pattern classification whereby patterns are viewed as sentences in a 'language' defined by a set of grammar rules.

Despite the apparent difference between the syntactic model and the classical statistical model, there are many similarities, in that statistical classification procedures are used by both models.

5.2. Data Organisation for Classification and Learning

The successful implementation of a statistical classifier is highly dependent on the organization of the model data and the observed features data.

The following basic requirements must be fulfilled:

- a) The organization of the model data and the observed features must result in a modular and expandable system both from the point of view of adding new models and new procedures for evaluating new features.
- b) The model information must be completely separate from the actual image analysis program modules (see Ch. 6).

Three types of data are involved in the classification process:

- a) The initial measurements or primitive features which reside in the primitive features buffer. They are calculated by the image analysis modules.
- b) The compound features which are combinations of the primitive features calculated by the classifier subject to availability of the primitive data, and according to the particular model requirements.
- c) Models and repertoires of models, containing the model data in terms of means and variances calculated during the learning process.

In relation to the paradigm proposed by M. Levine and S. Shaheen (Ref. 45) , the two first types of data constitute the Short Term Memory of the system whereas the model data is part of the Long Term Memory.

The primitive features buffer needs no further description, being a simple structured list. We will therefore focus our attention to the compound features and the structure of the models.

5.2.1. The Compound Features Set

The description of shape is a task of great importance, both in Artificial Intelligence terms, and in more practical terms.

A good listing of shape descriptive features can be found in Ref.39,ch 9. What is more important in our case is how to use the evaluated initial measurements (primitive features) in order to get an economical and efficient description.

By combining the primitive features in various ways we can get a set of so called compound features with the following properties:

- a) They provide a compact and hopefully informative description of the shape and the size of the objects.
- b) They are invariant under scaling, translation and rotation of the shapes.
- c) They aid in a better understanding and interpretation of the classification process in terms of human conception of shape and size.

An additional requirement introduced by the system modularity concept, is that the compound features set should also be modular, in the sense that the compound features should be grouped in subsets corresponding to the primitive features evaluated by the existing modules.

A set of features satisfying the above requirements to a certain extent is listed in Fig. 5.2. This set of features is obviously neither unique nor comprehensive. One of the major advantages of a modular design is the flexibility it offers in redefining the feature set.

MODULE	Primitive Features	Compound Features	Attributes Described
CONTOUR	Contour Area (A) Contour Perim. (P)	$A = 4\pi \frac{A}{p^2}$ $B = m \left(\frac{A}{A_0} \right)^*$	Shape ('compactness') Size
DATA	Second moments (IX, IY, IXY) Radii RMIN, RMAX, RAVR	$C = \frac{IX+IY}{N \cdot (RAVR)^2}$ $D = \frac{RMAX-RAVR}{RAVR}$ $E = \frac{RAVR-RMIN}{RAVR}$ $F = m \cdot \left(\frac{RAVR}{R_0} \right)^{2*}$	Shape (Normalized radius of gyration) Shape (Elongatedness) Shape (Elongatedness) Size
HOLES	Total holes area (AH) Total holes perm.(PH) Second moments IXH, IYH, IXYH Centre of internal pattern CXH, CYH Number of points (NH)	$G = 4\pi \left(\frac{AH}{PH^2} \right)$ $H = \frac{IXH+IYH}{NH \cdot A}$ $I = \frac{NH}{A}$ $J = \frac{(CXH)^2 + (CYH)^2}{A}$	"Shape" of int. pattern "Shape" of int. pattern Size (Relative) Eccentricity
* m is the area magnification factor Ao is the total image area (Ao = 128* 96) Ro is a distance normalizing constant (Ro = (128² +96²)^{1/2})			

Fig. 5.2 - The Set of Compound Features.

5.2.2. Models and Model Repertoires

Within the context of statistical classification, a model of a shape as formulated in 5.1, is a set of prototype feature vectors evaluated by a statistical analysis of the sample set obtained in the learning process.

Each prototype feature is represented by a sample mean value and the corresponding standard deviation.

Models are therefore created as structured data files (data arrays). The structure of models is illustrated in Fig. 5.3. A model directory resides at the top of each model, followed by a list of prototype feature identifiers. Each feature is represented by a label, a floating-point mean value, and a floating-point standard deviation (a total of 9 bytes).

The use of label identifiers serves two purposes:

- a) New features can be easily introduced and integrated within the existing data structure. A new label is assigned to each new feature in alphabetical order by the model creating routine (up to 26 compound features can be labelled). A feature evaluating routine bearing the same label has then to be included in the relevant system model that calculates the primitive features needed in order to evaluate the new compound feature.
- b) They can be used as node identifiers for a future classifier design based on binary tree decision techniques.

Each model occupies $(9d+12)$ bytes of memory, where d is the dimensionality of the feature space (number of features).

Models can be joined together to create repertoires. The structure of a repertoire is illustrated in Fig. 5.4. The total memory space occupied by a repertoire consisting of N models is given by

$$R = N.(9d+12).$$

For a 12-dimensional feature space, the 3K-byte repertoire buffer can accommodate one repertoire comprising up to 25 models.

MODEL DIRECTORY

NAME	Number of Features	Face Number
------	-----------------------	----------------

Characters 2 bytes 2 bytes

Feature Label (L)	Sample Mean (m)	Standard Deviation (σ)
A	4-bytes	4-bytes
B		
C		
D		
E		
F		
G		
H		
I		
J		

Fig. 5.3 - Structure of a Model.

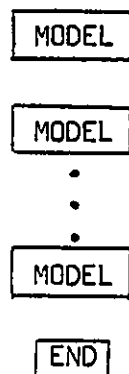


Fig. 5.4 - Structure of a Model Repertoire.

5.3. The Classifier

The design of the classifier is based on the statistical formulation of section 5.1., and more specifically on the decision rules (20).

The flowchart of Fig. 5.5 illustrates the implementation of these decision rules, using the data structures described in 5.2.

The rejection threshold r_t is user defined, but an optimum value can be obtained for a given application and can be kept fixed thereafter.

The classifier first reads the directory of the top model of the repertoire. It then reads the label associated with the first compound feature prescribed by the model (L_1), and searches in a menu of routines that evaluate compound features, for a routine bearing the same label.

If the system module that evaluates the primitive features that have to be combined to give the required compound feature is not present, the classifier is directed to a 'dummy' routine instead of the prescribed routine, since no measurements exist for its evaluation. If the required module is present, it activates the relevant routine by assigning it a label that exists in the menu.

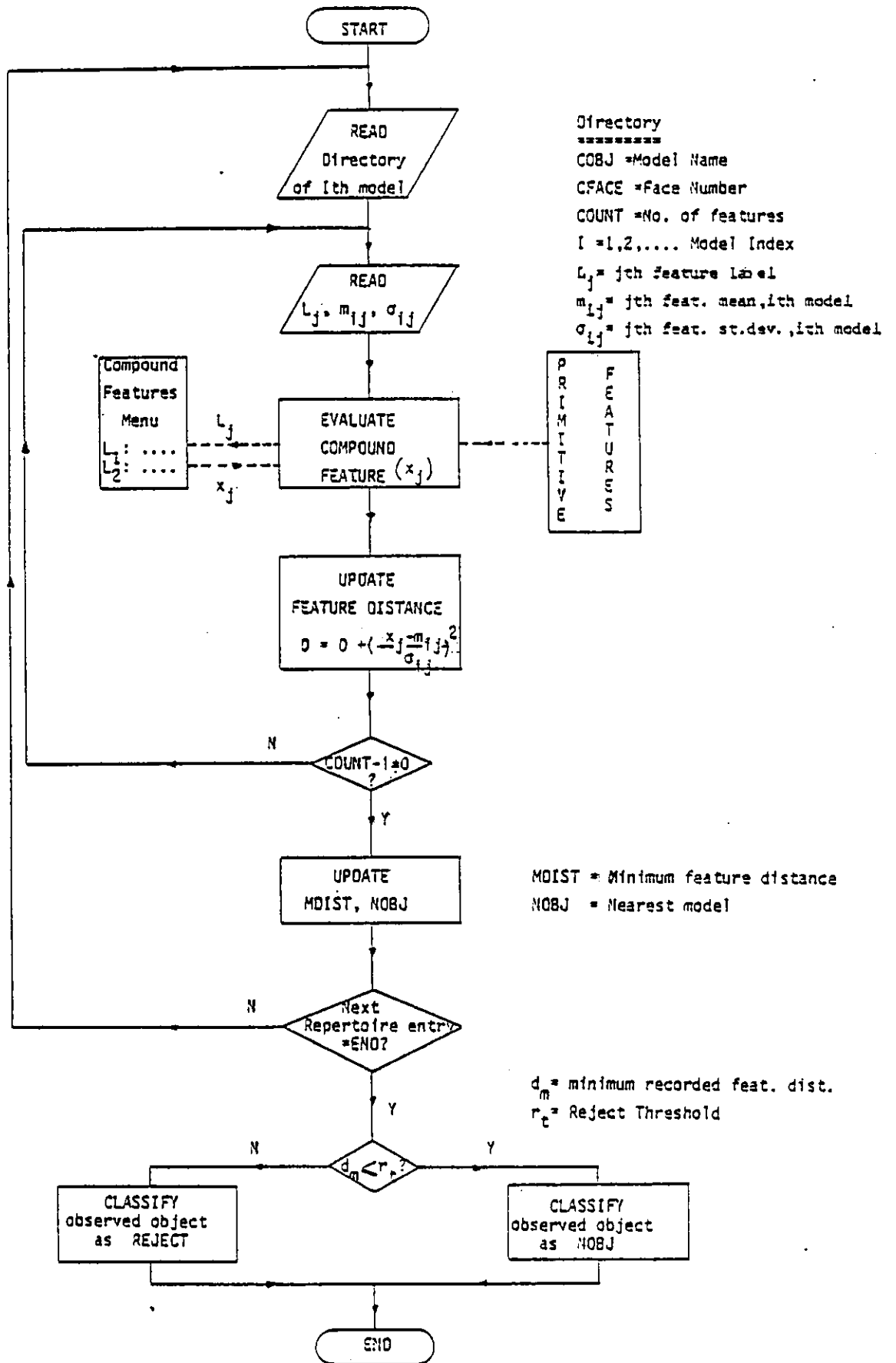
The evaluated compound feature is next compared with the mean μ_{11} of the first model, and a normalised distance in the direction of the first feature is calculated.

The same process is repeated for all the features prescribed by the model and an overall distance D is calculated. The observed feature set is then compared with the next model, and the same process is repeated for all the models of the repertoire.

Finally, the minimum recorded distance is compared with the rejection threshold r_t , in order to check for the fulfilment of the acceptance condition, and the observed object is either identified as being in the same class as its nearest sample mean, or it is rejected as unrecognizable.

Since a given part can appear in a number of stable states, and since each stable state typically represents a different image, and

Fig. 5.5 : CLASSIFICATION FLOWCHART



requires a different handling by the end effector, we treat each stable state as if it was a different part to be recognized.

5.4. The Learning Process

The learning process can be described in simple terms as the process of creating models from labelled samples. A set of sample images of the same object is presented to the system by the supervisor (operator).

Each sample is accompanied by a label (a name). The system first evaluates the prescribed features for each sample, and then it updates the following files:

- a) A sample file bearing the specified label.
- b) A model file bearing the same label.

The model file as described in section 5.2.2. contains the means and the standard deviations of the sample features. These statistical parameters can be evaluated either from the whole training set (the sample file), or in an iterative way, by using the following formulae :

$$\text{MEAN: } m_n = (n-1/n) \cdot m_{n-1} + (1/n) \cdot x_n \quad (21)$$

$$\text{ST. DEV.: } \sigma_n^2 = (n-1/n) \cdot (\sigma_{n-1}^2 + m_{n-1}^2) + (1/n) \cdot x_n^2 - m_n^2$$

$$n = 1, 2, \dots,$$

where: m_n is the new mean value,
 σ_n is the new st.dev. value,
both obtained by taking into account
the nth sample x_n , and their previous
values m_{n-1} and σ_{n-1} .

At any stage the operator can join model files to create new repertoires, and select one of them as the current repertoire.

Observed objects that resemble models existing in the current repertoire will be classified accordingly.

If the relevant model is not included in the repertoire, they will be rejected as unrecognizable.

At least two samples are needed for the creation of a model.

CHAPTER 6 : DESCRIPTION OF SYSTEM HARDWARE AND STRUCTURE OF SYSTEM SOFTWARE

6.1. System Hardware

The experimental vision workstation, that was used for the development of the image analysis algorithms, is illustrated in Fig. 6.1. A block diagram of the vision workstation is shown in Fig. 6.2. The main hardware components of the system are:

1. A LINK 109B self-contained monochrome television camera, with the following features:
 - . Horizontal resolution of 800 points for a vidicon tube.
 - . Fully stabilised and regulated circuitry to give stable pictures over a wide range of ambient temperature and AC power supply variation.
 - . External mixed sync and mixed blanking pulses.
(These pulses are provided by the digitiser, and they originate from the same crystal oscillator, so that the digitiser and the camera are kept at full synchronization).
 - . Good geometrical and linearity characteristics (linearity better than 1 % within a circle whose diameter is equal to picture height, better than 2.5 % over the remainder).
2. A home-built column digitiser, based on an original design by A. Eagles. This was modified to make it compatible with the I/O interface specifications of the SWTP microcomputer.

The digitiser was originally intended for off-line applications (Ref. 51 and 52), therefore it employs a low-speed 4-bit A/D converter, and it takes 40 secs to digitise a TV frame.

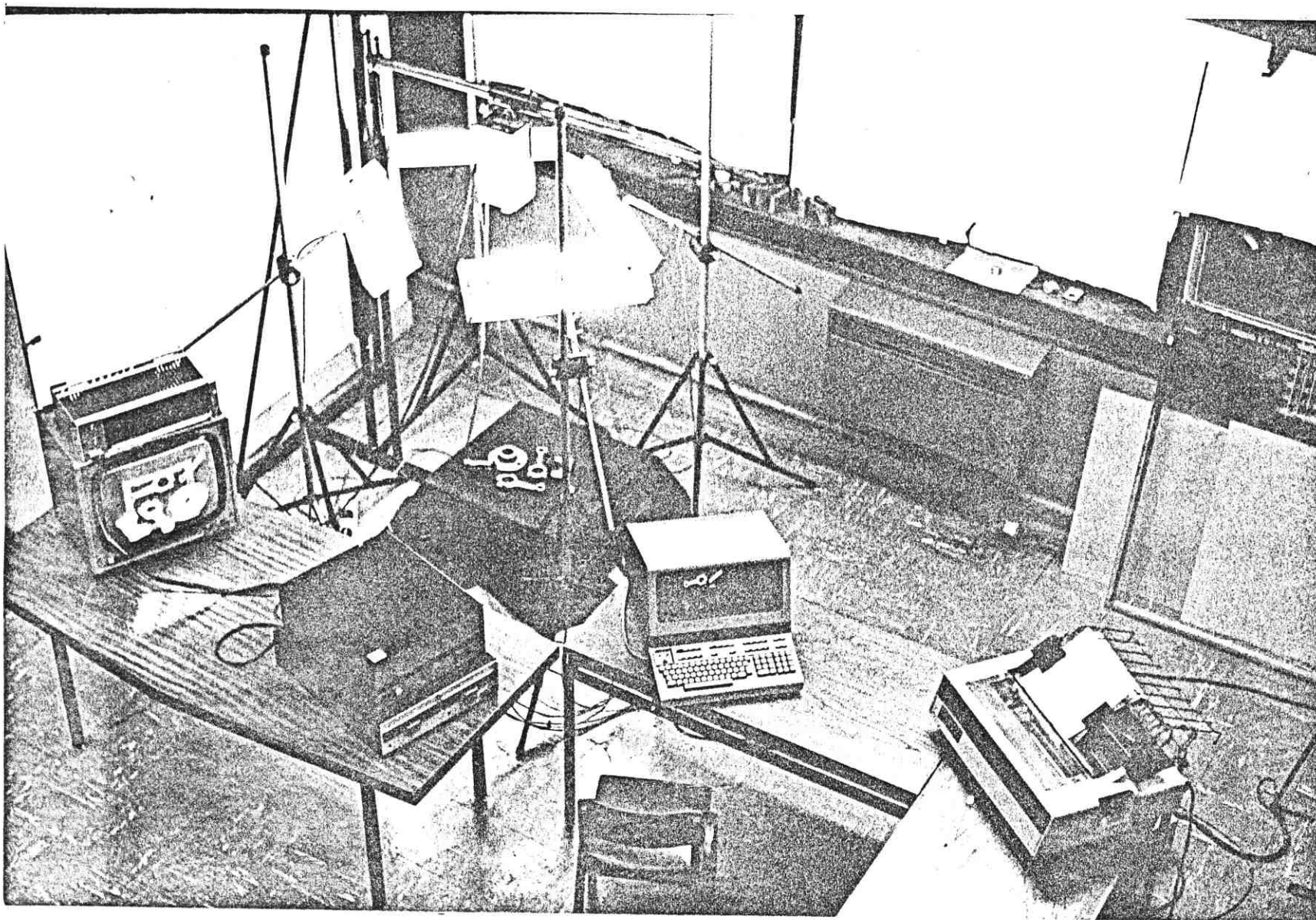


Fig. 6.1 : Picture of the Vision workstation.

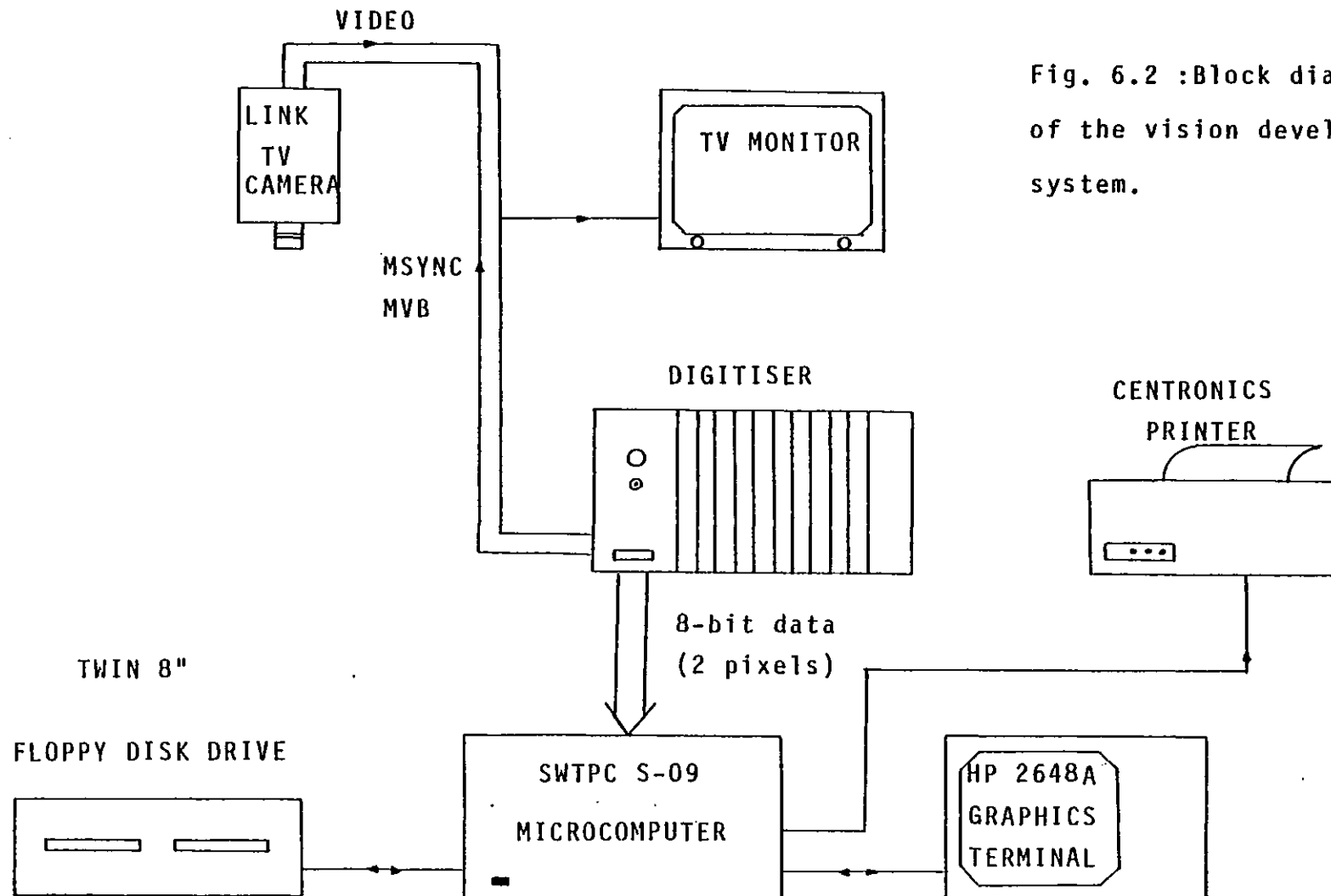


Fig. 6.2 :Block diagram of the vision development system.

The method of processing the video signal is as follows:

- 1) take the first point on the first horizontal line
- 2) process the data for this point whilst the TV camera is scanning the remainder of the line (Scanning one line takes 64 μ secs)
- 3) take the first point on the next line and process it
- 4) repeat for the remainder of the first column
- 5) pass data for whole column to computer
- 6) move on to next column and repeat.

The resultant digital information is in the form of 1024x576 points, with at each point the light intensity value coded as a 4-bit integer (16 grey levels). The effective horizontal resolution is 800 points (that of the TV camera).

The digitiser is interfaced to the microcomputer via an 8-bit bidirectional interface board comprising an MC6820 Peripheral Interface Adapter (PIA). An image acquisition interface routine (DIG. CMD) transforms the high-resolution input array to the required low resolution (128 by 96) image array, by averaging the incoming intensity levels over an 8 by 6 window. Details of the digitiser hardware and the interface routine can be found in Appendices A and B.

2. A Southwest Technical Products' (SWTP) 8 bit microcomputer system, incorporating a Motorola M6809 8 2MHz microprocessor as the CPU, 64K of dynamic RAM of which 56K is available to the user (6K is utilised by the FLEX9 operating system) and five I/O communication links. An extra parallel interface board was added to the system for communication to the digitiser.

A dual drive, double density, double sided 8 inch floppy disk unit was used for permanent storage of programs, data files and image files.

The capabilities of the microcomputer were enhanced by the addition of an extra board comprising an Am 9511 Advanced Micro Devices' floating-point processor, two MC6840 Programmable Timer Modules

(PTM's), and a TMS9914 IEEE 488 bus controller.

The board was designed and built in the College, its purpose being to aid in the development of the vision system, and its future integration with the control system of the robot.

At the present stage of development, only the floating point processor was used mainly to perform the floating point calculations needed at the classification stage.

4. An HP2648A alphanumeric/graphics terminal was used for the development of the system. The graphics terminal is not necessary for the operation of the system, but it aids considerably in the visualisation and evaluation of its performance. Obviously there is no better development tool for image processing algorithms than a means of displaying the image array, either as an array of integers, or as a binary image.

For the same purpose an ordinary text printer (Centronics 703) was used for the generation of hardcopies of program listings and image data. A Paper Tiger IDS-560 graphics printer was also used at certain stages to provide hardcopies of images, in the form of dotted arrays.

5. An ordinary monochrome TV monitor.

6.2. System operation - ICAROS Operating System

The experience obtained during early development stages showed that an interactive operating environment is necessary for the evaluation of vision software.

In fact, two of the most important features influencing the development of vision algorithms are software modularity and informative displays of images and data.

A simple interactive operating system was designed on these grounds.

ICAROS, which is an acronym for Imperial College Automatic Recognition Operating System, is an operating system within an operating system, in the sense that it is embedded within the FLEX9 Operating system. It is created as a FLEX9 Utility Command, but in the

same time it uses FLEX9 as a vehicle for the execution of all the standard FLEX commands, plus a set of user written utility commands.

The basic steps of a typical operating cycle are illustrated by the flow-diagram of Fig. 6.3. The principal operating modes are SEE, MONIT and LEARN.

On start of execution ICAROS will go through the following fixed sequence:

- . It will format the terminal screen dividing it into 3 sections, one for graphics displays of images and graphics text (softkey assignment), one for headings and display of current parameters, and the rest for interaction with the operator.
Fig. 6.4 is a picture of the formatted screen.
- . It will then load the current program code from disk (Command CURRENT CMD), and the current Repertoire of models (Data file CUR. REP).
- . Finally, it will rest at an IDLE state waiting for the operator to press one of 7 software defined keys (softkeys), to specify the operating mode.
- . If the softkey bearing the label SEE is pressed, the system enters its normal RECOGNITION operating mode. Program control is temporarily granted to CURRENT.CMD, which processes the image array currently residing in the image memory. The processing modules that constitute CURRENT.CMD are executed in sequence (image segmentation, feature extraction, classification), and at the end of processing a successfully closed outline, the name of the classified object is displayed, along with the position of its centre (XC, YC) and the orientation of its least moment of inertia(LMI) axis (THETA). If the HOLES module is present (see p. 159) the orientation angle of the LMI axis of the internal pattern is also

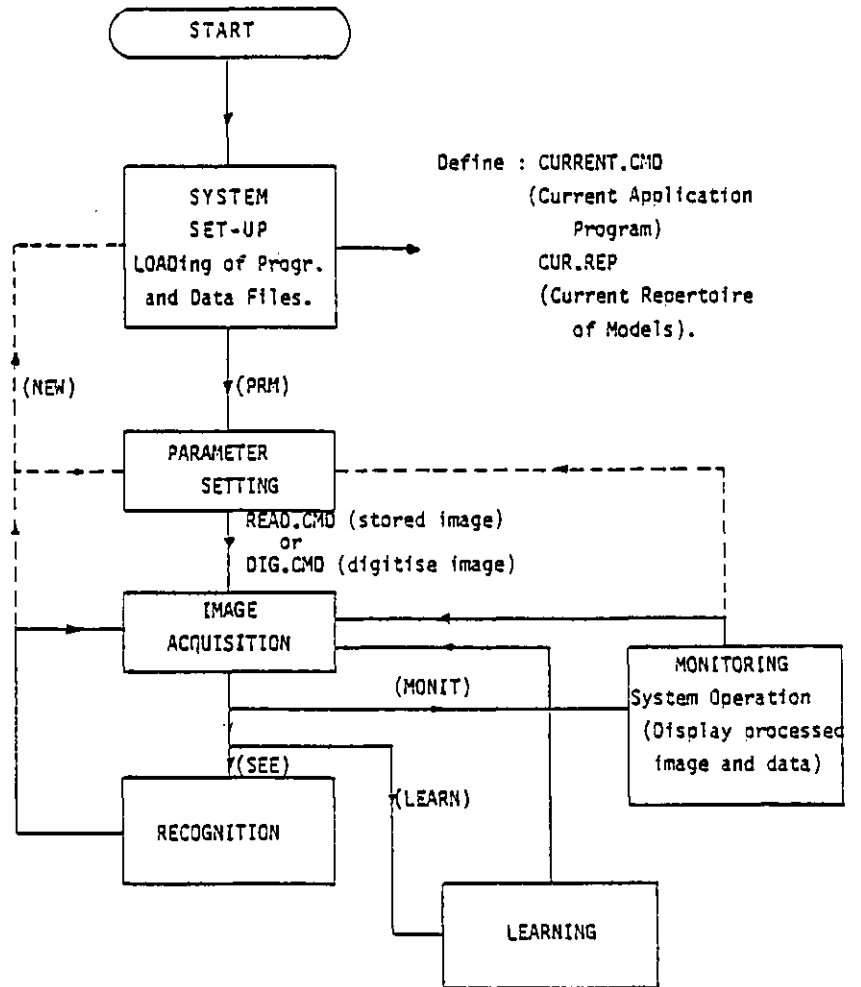


Fig. 6.3 : Typical Operating Cycles of ICAROS.

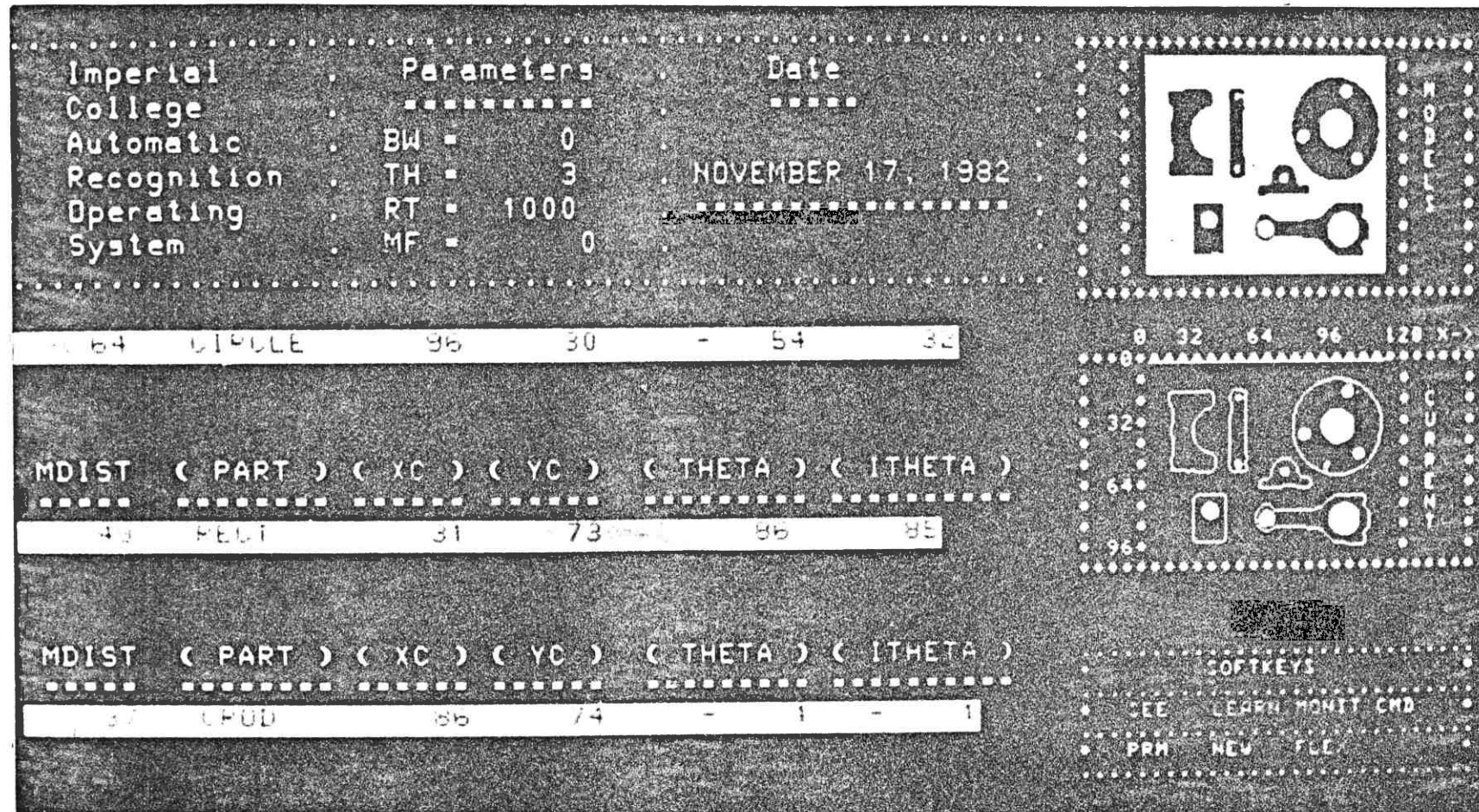


Fig. 6.4 : ICAROS - Picture of the formatted screen.

displayed (ITHETA).

The same process is repeated for all the objects in the scene and when the whole image array has been scanned, program control is returned to ICAROS.

SEE simulates the normal operation of the vision system, the only difference from a real operation being that, instead of displaying the outcome of classification on the screen, the same information will be routed to the robot controller.

- . If the softkey bearing the label LEARN is pressed, the system enters a training loop.

Control is passed to CURRENT.CMD again, but with a 'prescription' (defined by the status of an operating mode FLAG, labelled OPMODE) to bypass the classification stage, and call instead, a BASIC routine called SAMPLE.BAS, which:

- a) Reads the primitive features buffer (see p.82,133), and records the values of the measured primitive features for each successfully closed outline in a data file on disk (a sample file with extension.SMP).

The last processed outline is displayed on the screen, and the operator is asked to give a name (a label) to the corresponding object. The data file bearing the same name as the one specified is updated.

- b) Evaluates a set of compound features (A, B, C, D, ..., as in Fig. 5.3), and updates their mean values and standard deviations using the iterative formulae of section 5.4.

It reads their previous values from, and stores the updated values in a model file having the same name as the one entered by the operator and extension .MOD. For example, if the operator enters the name PART1 for the viewed object, the updated files will be PART1.SMP and PART1.MOD

At least two different samples must be obtained for each new object, so that the standard deviations of the model features are not zero. If however they are zero, they will be replaced by a low value of, say $m_j/1000$, where m_j is the mean value of the related feature j .

When all the objects found in the current image are labelled, control is returned to ICAROS.

After each individual object is labelled and before moving to the next object, routine CURRENT.CMD is automatically reloaded from disk by ICAROS, because the BASIC interpreter and routine SAMPLE.BAS are loaded in the same memory area (see memory map in Fig. 6.6).

- . If the softkey bearing the label MONIT is pressed the system will perform exactly the same operation as in SEE mode, but it will go through the various modules step by step, displaying at each step the processed image of the current object, plus the measured features at that step.

- . If the softkey bearing the label CMD is pressed, the system will wait for the operator to enter either a standard FLEX9 command (the full menu of FLEX commands can be found in Ref.48), or a user written utility command.

Utility commands perform basic image handling, such as digitisation, storage and retrieval of image files, and image array displays, either by using standard ASCII characters, or by plotting dots on the graphics terminal, or on a graphics printer etc.

A brief description of the vision utility commands is given in Appendix B.

All utility commands were written in 6809 assembly language; most of them employ FLEX9 system routines, and make use of FLEX9's File Management System (FMS).

- . If the sofkey bearing the label PRM is pressed, the system will ask the operator to redefine any of the 4 basic vision

parameters, namely the black or white background flag (BW), the minimum gradient threshold (TH), the classification reject threshold (RT), or the magnification factor (MF).

The first two parameters relate to the image segmentation stage.

- . BW is a parameter that defines the 'colour' of the background. It is a binary valued flag; if it is assigned a value of one, the system knows that the objects have higher reflectivity than the background, and thus they will appear in the image as 'white' blobs. A zero value denotes 'black' objects on a 'white' background.
- . TH defines a minimum gradient threshold to be used for the location of edges by the gradient operator (see p.70) Its real use is only when a noisy image is to be processed, or when the background presents small variations in light reflectivity. For 'clean' images with reasonably high contrast between the objects and the background, the default value three will be adequate.

The magnification factor MF is normally evaluated automatically from information obtained during the training stage, but the operator can override the estimated value by entering a new value.

The reject threshold RT can be set at a fixed value once enough information has been obtained after the training of the system for new objects.

The remaining two softkeys (NEW and FLEX) are used for vectoring the program counter to either an intermediate entry point of ICAROS, or to the Warm Start Address of the FLEX operating system.

So effectively, if NEW is pressed, ICAROS will return to the beginning of its operating cycle (see Fig. 6.3), and reload CURRENT.COMD and CUR. REP.

This key is normally used after a new definition of the operating routine `CURRENT.CMD`, and/or a new definition of the current repertoire of models.

A set of operating routines can be easily built by appending different sets of source modules (see p.156), and assigning different names to their assembled versions.

While in ICAROS, the operator can redefine the current operating routine simply by using the standard FLEX command `RENAME`. The same applies to the repertoires of models.

Finally, FLEX is the exit key for ICAROS.

If this key is typed, control returns to the FLEX operating system.

The present form of the vision operating system is limited, and it only caters for the basic needs of image handling and visualisation. However, the system is very easily expandable, either by the introduction of new utility commands or by definition of new operating modes. The use of FLEX9 as an infrastructure for the vision operating system may be considered as a drawback, in the sense that it renders it system dependent.

On the other hand, it offers considerable flexibility and expandability. Above all, the entire development system including the vision software was a very low-cost alternative to the special purpose vision computers or digital image processors available in the market at present.

Most of these systems offer very high-speed image processing at high resolutions, but they are more orientated towards other research fields, such as biomedical applications, electron microscopy, etc, where the prime objective is image enhancement rather than pattern recognition (Ref.46).

6.3. System Software

One of the main features of the system, as we have already mentioned, is that almost 80 % of the software was written in 6809 assembly language.

All the utility commands, as well as all the vision modules and the operating system are assembly routines. Only the routines used for the creation of models and model repertoires were written in XBASIC (Extended Basic), mainly because speed of operation is not a critical factor during the training stage.

The inherent disadvantages of this approach, namely the excessive programming effort required at the development stages, and the difficulty of transferring the software to other systems have been carefully considered.

However, we believe that the final decision was at least partly justified, for the following reasons:

- a) One of the major objectives of the project was to investigate the feasibility of implementing image analysis techniques on a low-cost microcomputer system.
At the beginning of the project 16-bit microcomputer development systems with reasonable software backup were not available.
- b) Software transfer is not a straightforward process anyway. Considerable effort may be required even in the case of high-level routines, especially when the application software is transferred from a minicomputer or a mainframe to a microcomputer. Most of the microcomputer based compilers tend to have certain peculiarities and limitations, imposed by the limited capabilities of the microcomputer systems themselves.
On the other hand, upgrading the software written in assembly code within the same microprocessor family (for instance from 6809 code to 68000 code), can be a relatively easy task for an experienced assembly programmer, and it offers the full benefit

from new technology.

- c) The contribution of this work was intended to be in the direction of proposing simplified, hence cost effective, image processing algorithms covering the whole spectrum of at least the basic steps in classical image analysis, rather than concentrating on a particular aspect of it.

The ultimate criterion for judging the results of this effort is obviously acceptable speed of operation, at low cost.

We believe that this objective is best served by coding in a low-level language, where simple solutions are imposed by the lack of availability of more powerful 'tools', as those offered by high-level languages, at the cost of execution time and increased memory requirements.

6.3.1. Software Modules

In an attempt of offsetting some of the forementioned disadvantages of assembly programming, and enhancing the expandability of the software, all the system software was written in a modular form. At least two distinct advantages of a modular software design can be listed:

1. Flexibility in system configuration

The development of a fixed, general purpose, low-cost, and efficient system is an almost impossible task, at least at the present state of the art.

The only viable alternative to the impossible task of satisfying all these contradicting requirements, is the design of an easily reconfigurable system. Modularity of software is the best way of achieving this aim.

2. Software maintenance and expandability

A modular software design aids considerably in testing, debugging and modifying the existing package, as well as expanding it.

An illustrative diagram of the structure of the vision software is given in Fig. 6.5.

The basic software building blocks are vision modules and vision commands.

Vision commands are used mainly for image file handling and display purposes (see Appendix 8).

Vision modules are the basic building blocks of vision application routines. Vision modules are in the form of source code that cannot be assembled and run on its own. The normal procedure of configuring a new application program is to append the source modules and then assemble the resulting source program, to obtain the required object code.

The existing vision modules at present are the following:

1. CORE: This is a common module shared by both the operating system and the application program (CURRENT.CMD). It contains declarations of all the system parameters, and common variables, a memory configuration (Figure 6.6), and all the low-level routines (common subroutines).
2. CONTOUR: This module is the main program supervisor, and it must precede all the other modules.
It performs a raster search of the image area, and as soon as it meets a new object it tries to follow its outline. At the same time it evaluates the basic primitive features of the outline, namely the centre coordinates, the enclosed area, and the perimeter length. It then calls all the other modules in sequence. When all the required information is extracted, and the object is classified, control is returned to CONTOUR, which 'seals' the outline of the current object, and continues the raster search, until the whole image array is searched.
3. DATA: For applications where the primitive features evaluated by CONTOUR are not sufficient for a discriminative description of the objects of interest, the addition of this module may provide the

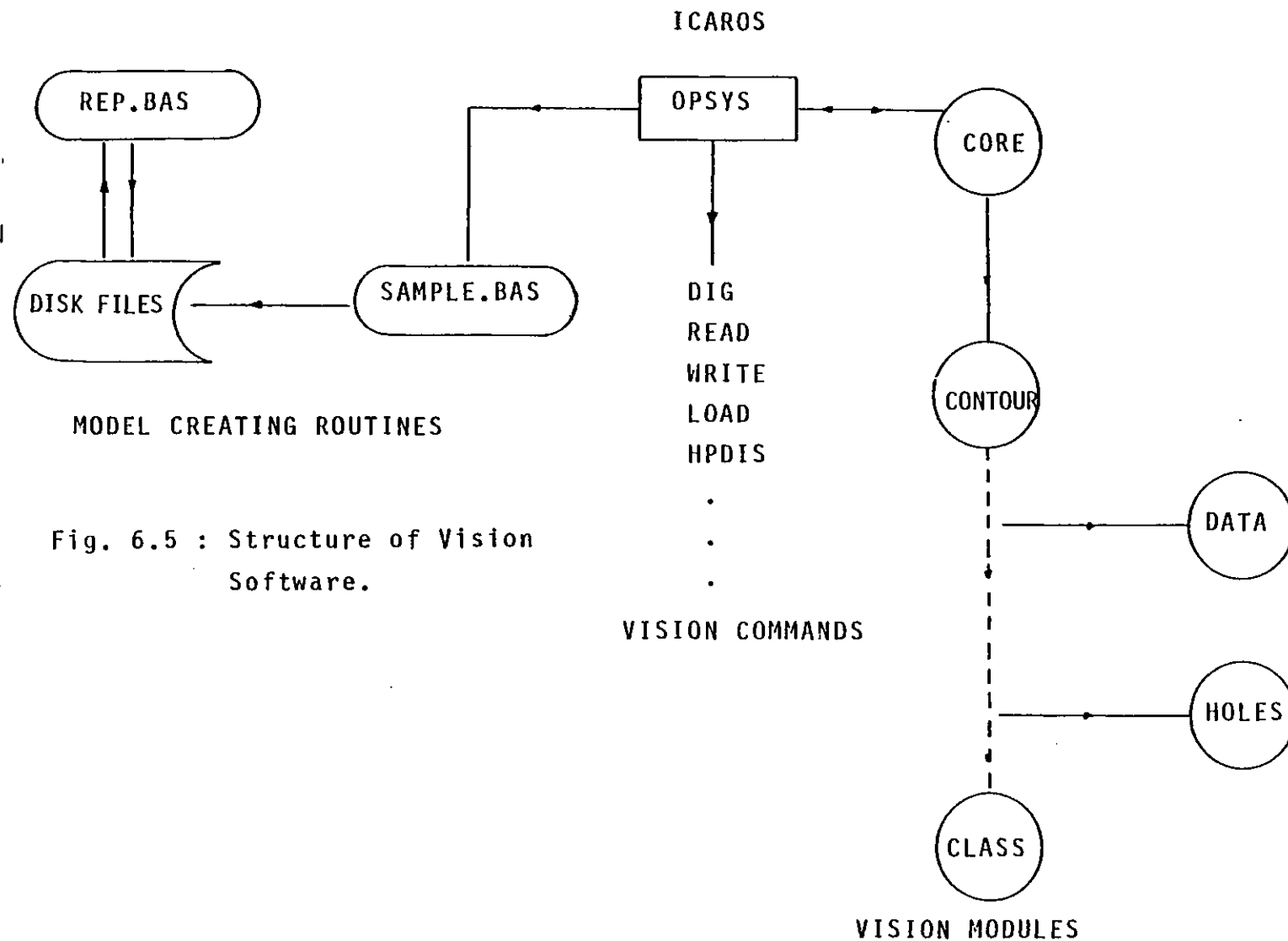


Fig. 6.5 : Structure of Vision Software.

Fig. 6.6: SYSTEM MEMORY MAP

ADDRESSES		DESCRIPTION		
\$ 0000				----- BASIC TRAINING ROUTINES (Overlay Area) ----- ↓
\$ 1000	USTOP	18K for Application routines (Modules). Program CURRENT.CMD		
\$ 5800	USEREN			
\$ 5810	TSTACK	2K STACK AREA (Outline Chain-code)		
\$ 6010	OUTLINE			
\$ 6020	BUFTOP	4K DATA BUFFER	\$6F50	
\$ 7020	BUFEND			
\$ 7030	COREST	5K SYSTEM CORE		
\$ 8430	COREND			
\$ 8440	OPTOP	2K ICAROS Operating System		
\$ 8C40	OPEND			
\$ 8DF6	IMST	12K IMAGE AREA		
\$ 8F00	IMEND			
\$ C000		8K FLEX. Op. System.	C100	1.5K
\$ DFFF			C6FF	Utility Command Area

required discriminative features. It is a feature evaluating routine, measuring characteristic radii and moments of inertia from outline data (see p.82).

The presence of this module is also necessary if the orientation of the objects is required.

4. HOLES: This module performs the description of the internal pattern (holes, protrusions, and other surface irregularities), as described in section 4.4.

5. CLASS: This module is responsible for the classification of the current object, in terms of the features evaluated by previous modules.

It reads the contents of the current repertoire of models (see p.135) and calculates a feature 'distance' d_i ($i = 1, 2, \dots, c$) from the point in the feature space that corresponds to the observed object, to each of the c points representing the models of the repertoire. It then assigns the observed object to the nearest class represented by the model points.

CLASS closes the sequence of modules and should always be included in the application program.

The minimum configuration of an application program therefore is CORE + CONTOUR + CLASS.

The maximum configuration using the existing modules is CORE + CONTOUR + DATA + HOLES + CLASS.

Module linkage is performed by the assembler. Each module that is present in the appended source file, 'stores' its start address at a fixed location in a RECIPE table, that resides at the bottom of the memory area occupied by the modules (Fig. 6.6).

If the module is not present its entry in the RECIPE table is pointing to a default start address of a 'dummy' routine, that simply signals the absence of the particular model. During execution, the supervisory routine (CONTOUR) calls all the routines that are specified in the RECIPE table in sequence.

6.3.2. Database Structure - Image Files

The main types of data used by the system are:

- a) Image Arrays.
- b) Model Arrays.

Image arrays consist of picture elements (pixels.). Each pixel is represented by an 8-bit word. A full description of the digital representation of pixels was given in section 4.1.2. Model arrays can be concatenated to form repertoires of models. Model arrays and model repertoires were described in section 5.2.2.

An image array preceded by an image directory forms an image file. A typical low-resolution image file (128 x 96 pixel) contains a total of 12556 8-bit information of which 12416 8-bit words constitute the image array, the rest being the contents of the image directory.

The image directory consists of two parts (Fig. 6.7). The first part contains informations about the layout of the image array; the second part is a list of the addresses of the first pixel at the top of each column.

Two pointers are used to access any of the 128 x 96 pixels; the CXPNT (Current-X pointer) which points to the entry of the column start address list that corresponds to the required column, and the CYOFS (Current-Y offset), which provides the Y-offest from the top of the column (the line number).

A column terminator (hexadecimal FF) marks the end of each column.

The described structure permits easy future expansion of image files to accomodate higher resolution image arrays. The only routine that will need to be modified is the digitiser interface routine (DIG), which structures the incoming information in the above mentioned form.

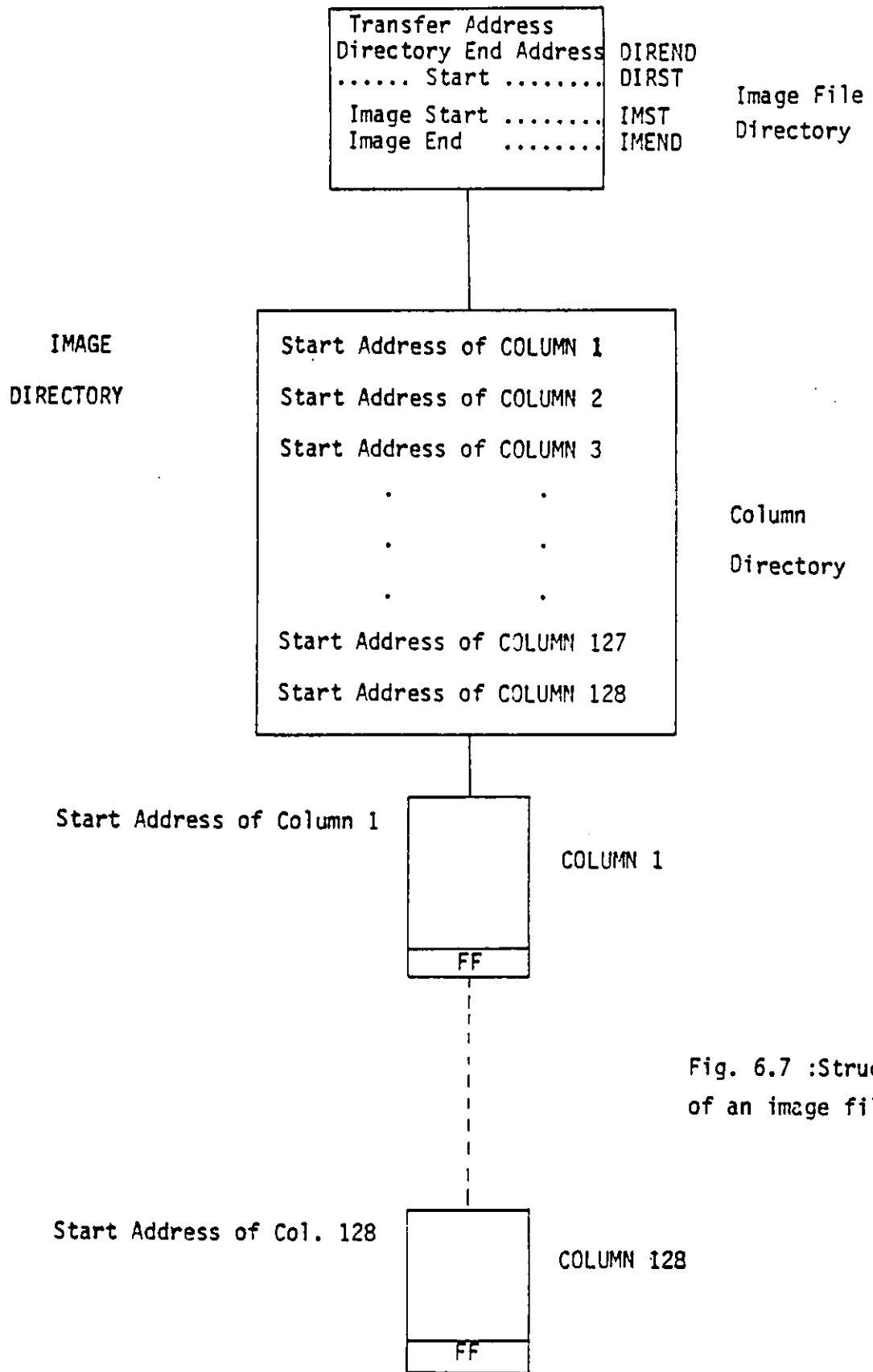


Fig. 6.7 :Structure of an image file

CHAPTER 7 : PERFORMANCE - EVALUATION

The purpose of the evaluation study that follows is to establish a set of criteria and to outline the performance of the system according to these criteria.

The performance criteria considered in this study relate to the required information, namely the identity, the position and the orientation of the viewed objects.

Hence the related performance parameters are the classification accuracy, the positional accuracy, and the directional accuracy, respectively.

The limitations of the proposed system are also discussed, and the evaluation study ends with a comparison of the system with a few representative existing systems.

7.1. Positional Accuracy, Directional Accuracy, Timing

7.1.1. Positional Accuracy

The position of an object on a plane is determined by the centre of the silhouette of its image (see p.82). Since the acquired analog image is converted to a 128 by 96 image array, the prime factor determining the positional accuracy in terms of real world coordinates is the 'effective' resolution of the image, which can be expressed in mm/pixel (a pixel here denotes the unit length of the image array, rather than the picture element itself).

For a fixed objective lens, the effective resolution is directly proportional to the distance of the lens from the centre of the viewed scene.

The main reasons causing errors in the evaluation of the centre are the following:

a. The spatial quantization of the image function:

A grid quantization error of ± 0.5 pixel may be expected in both X_c and Y_c .

b. The quality and the sharpness of the image:

Poor top lighting, poor contrast and poor focusing of the image may cause shadows and blurring. The resulting error in the evaluation of the centre cannot be estimated, but it can be quite large.

c. The relationship between the height and the width of the object:

A large height - to - width ratio may cause substantial parallax errors, depending on the distance of the camera lens from the centre of the scene, and the distance of the object from the centre of the scene.

Fig. 7.1. illustrates some of the above mentioned error sources. The geometrical distortion introduced by the camera is generally negligible compared to the other error sources.

Experimental Evaluation of the positional accuracy

A set of four cylindrical objects with varying diameters and heights were used for the experimental evaluation of the expected overall positional error. These objects were placed at various positions along the diagonals of a rectangle, and they were viewed from three different positions along the optical axis of the camera (see Fig. 7.2).

For each position of the camera the effective positional resolution was evaluated from the recorded measurements of the average radii of the viewed circles, and the corresponding known radii of the cylindrical objects.

The observed positional errors are given in Fig. 7.3 expressed in mm as well as in pixels.

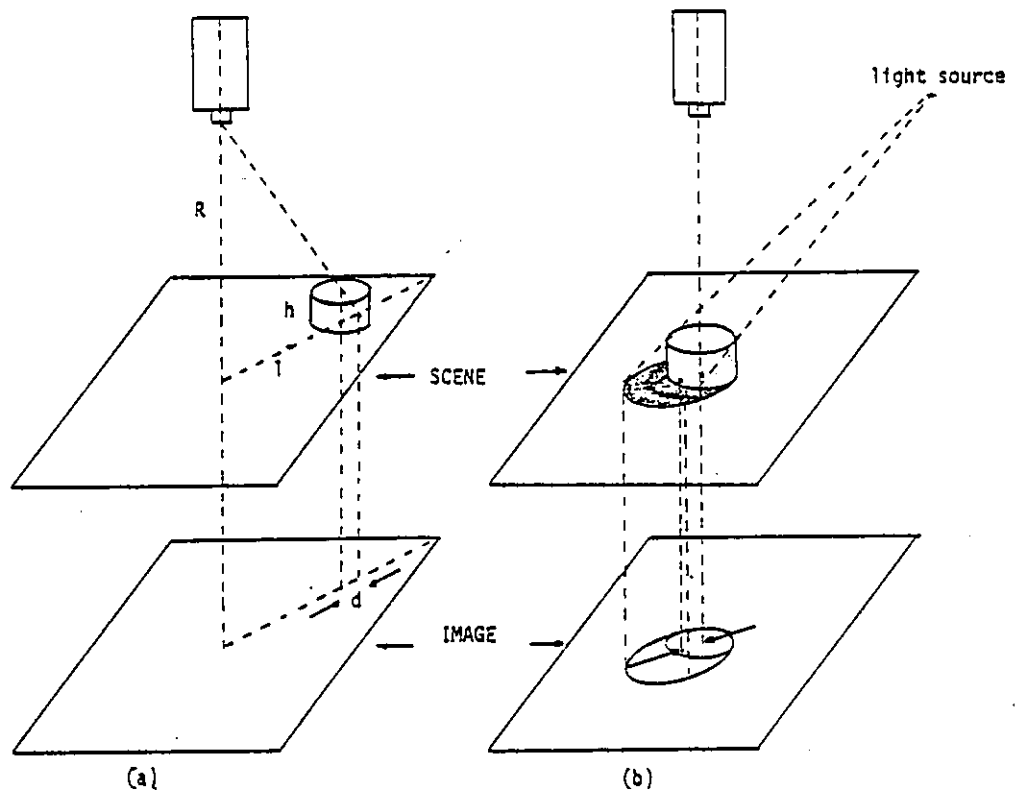


Fig. 7.1 : a) Displacement of the centre due to parallax.

$$d = f(R, l, h)$$

b) Displacement of the centre due to shadows.

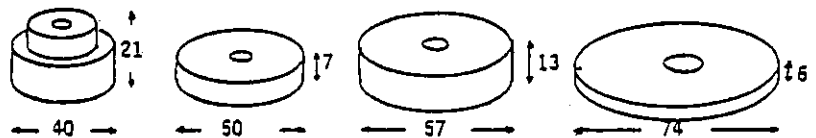


Fig. 7.2. : Test objects used for the evaluation of the positional accuracy.
All dimensions are in mm.

EVALUATION OF POSITIONAL ACCURACY

CAMERA POSITION (cm)	POSITIONAL RESOLUTION (mm/pixel)	MEASURED DISPLACEMENTS				Max. Positional Error + $\sqrt{(\Delta X)^2 + (\Delta Y)^2}$	
		ΔX (mm)	ΔX (pixels)	ΔY (mm)	ΔY (pixels)	mm	pixels
127	4	2.0	0.5	-1.0	-0.25	7.3	1.8
		2.0	0.5	7.0	1.75		
		2.0	0.5	1.0	0.25		
		6.0	1.5	1.0	0.25		
		4.0	1.0	0.0	0.00		
		0.0	0.0	0.0	0.00		
80	2.5	-4.0	-1.6	1.0	0.4	6.9	2.7
		6.0	2.4	-1.5	0.6		
		1.0	0.4	3.5	1.4		
		6.0	2.4	3.5	1.4		
		5.0	2.0	2.5	1.0		
		0.0	0.0	2.5	1.0		
50	1.4	-1.5	-1.0	0.0	0.0	2.9	2.1
		0.0	0.0	-2.0	-1.4		
		-2.5	-1.8	0.0	0		
		+2.5	1.8	1.5	1.0		
		-2.0	-1.4	0.0	0.0		
		-1.0	0.7	0.0	0.0		

Fig. 7.3: Result of positional accuracy tests

Considering an inherent error of ± 1 mm in positioning the cylinders on the rectangle, the measurements obtained show a maximum positional error of ± 2 pixels.

This estimate can serve only as an indication of the range of the expected error, since in the general case the positional accuracy will depend on unpredictably varying parameters, such as the shape of the viewed objects, the lighting conditions, e.t.c.

7.1.2. Directional Accuracy

The same reasons affecting the accuracy of the evaluation of the position of an object, also influence the accuracy of the evaluation of its orientation.

Therefore it is equally difficult to estimate analytically the expected directional error.

The experimental evaluation of the range of the directional error was performed along the same lines as before.

A set of objects presenting long rectangular projections on the image plane was chosen for this purpose (see Fig. 7.4).

These objects were placed at various positions, and at various orientations specified by the angles between their axes of least moment of inertia and the X-axis, and they were viewed from different camera positions.

The differences between the actual orientations and the orientations evaluated by the system are given in Fig. 7.5).

The results obtained from this test show that the maximum directional error for the given set of objects is $\pm 2^\circ$.

7.1.3. Timing Performance

The processing times depend on two separate factors: the complexity of the image and the amount of information that has been selected for classification purposes. (The evaluation of second-order features of the outline, and the description of the internal pattern is optional- see also fig. 6.5.

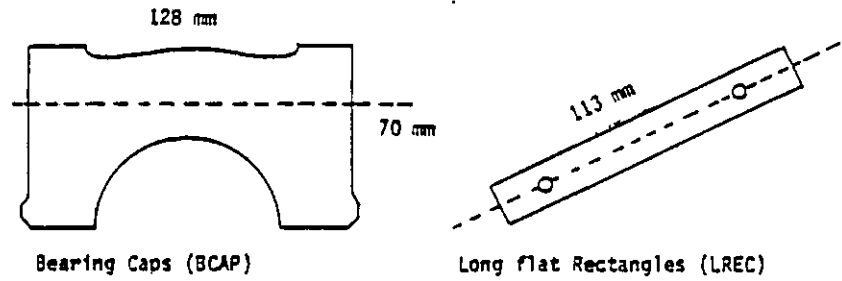


Fig. 7.4: Objects used for the evaluation of directional accuracy.

The dashed lines denote the directions of the LMI axes.

EVALUATION OF DIRECTIONAL ACCURACY				
Camera Distance (mm)	Viewed Object	θ_{real} (deg)	$\theta_{\text{meas.}}$ (deg)	$\Delta\theta$ (deg)
127	LREC	-33	-33	0
	LREC	33	31	2
	LREC	0	0	0
127	LREC	90	89	1
	LREC	-27	-26	-1
	LREC	-33	-34	1
127	LREC	0	0	0
	LREC	40	40	0
	LREC	90	-89 *	-1
127	BCAP	-10	-9	-1
	BCAP	0	0	0
	BCAP	50	50	0
	BCAP	-60	-58	-2
	BCAP	60	60	0
88	BCAP	33	33	0
	BCAP	33	31	2
	BCAP	-33	-34	1
	BCAP	-33	-32	-1

Fig. 7.5.: Table of results of directional accuracy tests

* The inherent 180 degree ambiguity is manifested here.

To obtain an estimate of the processing time, we measured the time required by the system to analyze a scene containing four objects with six holes, all processing options having been selected.

The processing time for that scene was found to be approximately 4 seconds (1 second per object).

In general, processing times in the range of 0.5 seconds to 2 seconds may be required, depending on the perimeter of the processed shape, its area, and the number of its internal features.

7.2. Detection and Identification Accuracy

The main task of the system is to detect the presence of all the objects that appear in the viewed scene and to identify them correctly.

We have already mentioned in ch.5 that the design of the classifier was based on statistical considerations, hence there is an inherent probability of classification error. The error rate constitutes the basis for the evaluation of the performance of any pattern recognition system (Ref. 39, p.16).

In the case of a complete vision system, however, we must also consider detection errors and classification errors relating to the environment.

We can therefore distinguish the following three categories of errors:

a) Undetected objects: Objects that appear in the scene may not be detected at all, due to the following reasons:

- .Low contrast between the background and the workpiece itself.

- .The object is touching another object already detected.

- .Three-dimensional objects can generate sharp reflections and shadows. The contour follower can in such cases be misdirected or halted altogether (formation of contour loops).

Obviously a meaningful detection rate can only be obtained for a system working in a real industrial environment with a known set of workpieces, under optimized lighting conditions.

Needless to repeat that the best solution, wherever applicable, is the use of transmitted light.

b) Rejected objects: Detected objects can be rejected as unrecognizable if their description is not adequately "similar" to any of the prototypes of the repertoire. The option of rejection was introduced in order to replace some of the more costly classification errors (see p. 130). The ratio between the reject rate and the error rate can be optimized by experimentation.

c) Misclassified objects: Classification errors may arise from:
. Objects with similar silhouettes.

. Shape distortions caused by the environment
(shadows, sharp reflections, perspective transformations)

The performance of the system in terms of the above mentioned errors (mainly rejections and misclassifications) was checked against a set of ten different shapes in the test case that will be described next.

7.2.1. Test case

The implementation of vision systems in industry is a fairly new concept, hence there is still a lack of adequate information concerning the requirements related to the working environment. The wide range of potential applications makes it even more difficult to select a representative set of parts and a representative environment, in order to test the performance of the system under realistic conditions.

The test case considered in this section may serve as an example and in the same time as an indication of the capabilities and the limitations of our system, but it is not meant to be a typical application.

The system was trained to recognize ten different shapes appearing as two-dimensional silhouettes of nine objects, which can be grouped in two sets according to their dimensions: (see also APPENDIX C)

The first set of "small" objects consists of:

- . a small spanner (SPAN)
- . a key (KEY)
- . a coin (COIN)
- . an ordinary mains plug. (PLUG.)

The second set of larger objects includes:

- . a bearing cap. (BCAP)
- . a connecting rod. (CROD)
- . an object with the shape of a large ring with three symmetrical holes.

- . a simple rectangular object with a hole. (RECT)
- . an object with the shape of the capital Greek letter omega (OMEGA).

Two distinct shapes, corresponding to its two stable positions on a horizontal plane , were considered in the case of the bearing cap (BCAP1 and BCAP2).

The bearing caps and the connecting rods were supplied by the FORD Motor Company.

The system was tested with a set of 148 test samples. In order to obtain a realistic estimate of the error rate the set of test samples must not coincide with the set of samples that was used for training (Ref. 39 , p.74). The above mentioned objects were randomly positioned on a black horizontal base, and they were viewed under incident illumination obtained from four fluorescent lamps covered with ordinary diffusers.

The angle of illumination (relative to the optical axis of the camera) was set to a large value to minimize the amount of directly reflected light.

Finally, the lens was slightly defocussed in order to mask out irrelevant surface irregularities.

The results obtained for each of the 148 samples are listed in fig. 7.6. The number following the name of each identified object is the recorded feature distance from the prototype vector of the identified class.

Classification errors are marked with an asterisk.

Discussion of the results

The results of the classification test are summarised in fig 7.6 , in two separate groups.

GROUP 1 (small flat objects) -32 samples

NAME	F. DIST	NAME	F.DIST	NAME	F.DIST	NAME	F.DIST	NAME	F.DIST
KEY	15	COIN	16	SPAN	4	SPAN	7	KEY	14
PLUG	15	SPAN	7	COIN	5	PLUG	7	PLUG	5
SPAN	9	PLUG	4	SPAN	11	COIN	3	SPAN	4
COIN	5	KEY	18	KEY	19	PLUG	15	COIN	7
PLUG	5	COIN	7	PLUG	5	SPAN	2		
SPAN	11	PLUG	7	COIN	6	KEY	27		
KEY	12	KEY	15	KEY	7	COIN	19		

GROUP 2 (large 3-D objects) -116 samples

NAME	F. DIST	NAME	F.DIST	NAME	F.DIST	NAME	F.DIST	NAME	F.DIST
BCAP1	8	OMEGA	27	CROD	46	CROD	21	RECT	10
OMEGA	12	BCAP1	5	CROD	44	BCAP2	13	BCAP2	33
RING	42	RING	113	RING	80	CROD	13	OMEGA	46
RECT	5	OMEGA	44	BCAP1	17	BCAP2	3	<u>BCAP2*</u>	225
(BCAP2)	N.D.	RECT	20	BCAP2	7	RECT	12	BCAP2	9
RING	49	CROD	103	RECT	4	BCAP2	37	CROD	40
BCAP2	28	BCAP2	4	RING	50	BCAP2	15	BCAP1	7
BCAP2	23	<u>PLUG*</u>	178	BCAP1	8	OMEGA	30	BCAP1	6
OMEGA	10	CROD	38	CROD	54	BCAP2	26	BCAP1	6
RECT	12	BCAP2	14	BCAP2	7	CROD	92	BCAP1	7
RING	131	BCAP1	9	OMEGA	12	RING	47	CROD	92
OMEGA	31	RECT	14	RING	58	CROD	21	CROD	34
BCAP2	30	OMEGA	28	CROD	15	BCAP2	22	(RECT)	N.D.
BCAP2	17	CROD	48	BCAP2	10	BCAP2	17	RING	65
RECT	18	RECT	12	RECT	8	BCAP1	10	BCAP1	13
BCAP1	15	RING	11	BCAP2	21	BCAP1	7	CROD	26
<u>PLUG*</u>	162	BCAP1	6	OMEGA	59	OMEGA	51	RECT	13
BCAP1	14	BCAP2	19	RING	62	CROD	50	RECT	4
OMEGA	29	OMEGA	37	CROD	54	BCAP2	11	OMEGA	50
RECT	10	BCAP2	29	BCAP1	13	CROD	53	BCAP2	29
BCAP1	10	RECT	6	BCAP1	6	RECT	20	(RECT)	N.D.
BCAP2	9	BCAP2	8	OMEGA	25	BCAP2	10		
RING	83	RING	33	(RECT)	N.D.	BCAP2	17		
RECT	15	OMEGA	26	RING	63	BCAP1	7		

Fig. 7.6 : LIST OF CLASSIFICATION TESTS

- N.D. means Non-Detected
- The asterisk signifies misclassification

The first group includes the results obtained with samples of small, essentially flat objects, whereas the second group refers to the results obtained with samples of larger objects with prominent three-dimensional structure. We must clarify at this point that, although the tests were conducted in two groups, the repertoire of the system included all the objects considered.

The results obtained with samples from the first group show that the performance of the system with flat objects is optimum, especially when the shapes of these objects are essentially dissimilar.

As far as the second group is concerned we note that all three types of errors now appear.

Four samples were not detected, and three other were misclassified.

From the four cases of non-detection, three were due to touching outlines. The objects were randomly positioned in the scene, many at a time, and care was taken so that they did not touch each other.

However, due to the low spatial resolution of the system and the existence of shadows and perspective distortions, the outlines of objects that are close to each other although not touching, appear in the image array as actually touching each other. This means that an adequate separation of the objects is necessary, when they are viewed at low resolution.

The fourth reason of non-detection was low contrast.

The reject threshold in both groups was initially set to a high value ($RT=1000$), thus effectively eliminating the probability of rejection.

However, we can see from the list of results of fig.7.6 that if the reject threshold was set to a much lower value (say $RT=100$), then all the three misclassification errors would have been converted to rejects, leading to a zero "error rate".

The penalty, however, for this would have been that three

correct classifications would have also been converted to rejects, leading to a reject rate of approx .5 percent.

These two extreme cases indicate the extend of compromise that can be achieved, by varying the reject threshold.

Finally, by using the method of estimating the error rate mentioned in Ref. 39 p. 73 , we conclude that for the given set of objects, with probability 0.95 the true error rate of the classifier will be in the range of 1 to 5 percent.

If we consider only the second group of objects, then the estimated region of the error rate, with probability 0.95, now becomes 1 to 9 percent.

7.3. Comparison with existing systems

In section 2 we described the basic features of three popular vision systems, the SRI Vision Module, the CONSIGHT system by General Motors, and the BBC (Brown - Boveri Corporation) Optoelectronic Sensor. These systems were selected as representative of the state of the art at the beginning of the project.

They are all mainly intended for the same application area as our system, namely that of sensor controlled manipulation and assembly. They process two-dimensional images of simple mechanical parts having one-coloured surfaces , and they all employ general-purpose programmable processors.

In the comparison study that follows we intend to present the differences and the similarities of our system with each one of these three systems, both in terms of methodology and in terms of performance characteristics.

For reasons of completeness we also included in our study a highly sophisticated model-based vision system proposed by W.A. Perkins

which can work successfully in visually noisy scenes and, to a certain extend, with overlapping parts.

The **basic characteristics** of these systems, namely their spatial resolution, their brightness resolution (number of intensity levels) and the type of processor used, are listed in table 7.1.

The W.A. Perkins system and our system use gray - level techniques, a significant feature that offers considerable flexibility in terms of the amount of visual noise that they can handle. The extend to which the gray-level information is exploited by the model-based system of W.A. Perkins is undoubtedly larger than in our case , a fact that becomes evident from the level of sophistication of the image analysis techniques used by this system (see Table 7.2., image segmentation).

However this is achieved by the use of a powerful mainframe computer, and still the typical processing speed obtained is prohibitively low.

On the other hand, the three first systems use binary representation of the intensity levels , therefore they can employ fast and efficient techniques for image segmentation, based on "run-length coding". Considering their processing resources (the first uses a 16-bit microcomputer , the second uses a minicomputer , and the third a combination of a 16-bit microcomputer and a very fast bit-slice processor), and the flexibility they offer , they achieve realistic processing times at a reasonable cost.

The exact evaluation of capital cost for a whole system from the information available in the literature is not easy; however a rough estimate of the region of the cost (low-medium-high) can be made by judging their hardware configurations. The main limitation of these systems is the need for a noise-free image with a high contrast. To achieve this it is necessary, at least in some cases, to use transmitted light or other contrast enhancement techniques.

The clever lighting arrangement used by the CONSIGHT system enables it to get around this difficulty without the need of restructuring

COMPARISON TABLES

Table 7.1. BASIC CHARACTERISTICS

	SRI Vision Module	CONSIGHT (by General Motors)	BBC Optoelectronic Sensor	W.A. PERKINS Model-Based System	Proposed System
Spatial Resolution	128 x 128	128 x 1 (linear array)	256x256 (maximum 512x512)	256 x 256	128 x 96
Brightness Resolution	Binary System (computer-controlled dual thresholds)	Binary System	Binary System	32 intensity levels (gray-level system)	16 intensity levels
Hardware	Solid-state TV camera, Preprocessor interface, LSI-II 16-bit microcomputer, 28K-word memory	Solid-state line camera, PDP 11/34 minicomputer or (LSI-II microcomputer)	LSI-II microcomputer and bit-slice microprocessor Two image memories	IBM 370/168 main-frame computer	6809B based 8-bit microcomputer AM9511 floating point processor
Prominent Feature	Modular industrial system, one of the first practical designs	Clever lighting arrange- ment (projected light).	Fast processing achieved by microprogrammable processor (bit-slice)	Sophisticated system Extensive image analysis enables it to work in visually noisy scenes	Gray-level proces- sing achieved at reasonable speeds and low cost
Applications	Sensor-controlled manipulation In- process inspection Visual servoing (limited)	Sensor-controlled manipulation and assembly	Sensor-controlled manipulation Inspection Visual servoing	Sensor-controlled manipulation	Sensor controlled manipulation
Timing	1.4secs for a typi- cal scene including one object	5secs robot cycle time	32-200 msec for a typical scene	20 secs per object	1-2 secs per object
Cost	MEDIUM	MEDIUM	MEDIUM TO HIGH	VERY HIGH (experimental system)	LOW
REFERENCES	(.24.)	(.15.)	(.25.)	(.17.)	(.27.)

COMPARISON TABLES

Table 7.2 IMAGE DESCRIPTION METHODOLOGY

	SRI Vision Module	CONSIGHT (by General Motors)	DDC Optoelectronic Sensor	W.A. PERKINS' Model-Based System	Proposed System
IMAGE SEGMENTATION	Run-length coding followed by connectivity analysis	Run-length coding, connectivity analysis based on 6-connected regions	Connectivity Analysis	Edge points-chains- -Concaves-Regions	Combined binary and gradient follower
DESCRIPTION	Global shape des- criptors (area, perim, no of holes area of holes,max and min radii etc)	Global shape descriptors	Area, radius of smallest circle, distribution of area as a function of the distance from centre, no of inter- sections of a set of circles with boundary.	Model-Based (Number and relative position of concaves)	Global Shape Descriptors (Area, perimeter, second moments, characteristic radii, global features of internal pattern)
CLASSIFICATION TECHNIQUE	Nearest Neighbour Classifier (NNC) or automatically generated Binary Decision Tree.	Binary Decision Trees	Binary Decision Trees	Template-matching technique based on the concaves model	NNC
TRAINING	By showing	By showing	By showing	By showing	By showing
ORIENTATION DESCRIPTION	Direction of Least Moment of Inertia axis (LMI axis)	Direction of LMI axis, or user definable axis	Obtained by the circles model	Obtained by the best fit of the model of concaves	Direction of LMI axis

the environment. However this system still suffers from occlusions of internal features due to the shadowing effect.

Accuracy specifications were only obtainable for the BBC system and the W.A. Perkins system (see table 7.3).

The terms "flexibility" and "versatility" that appear in table 7.3 as task performance characteristics , are used here with the following meanings:

By flexibility we mean "the ability to perform a given manipulation task under changeable local conditions. A degree of adaptation is implied: the task goals are fixed but the action may alter or adjust to suit the conditions in the work environment" (see Ref 47)

The main contributing factor in the direction of achieving a flexible system is the modularity of the desing.

The term **versatility**, on the other hand, is concerned with the range of potential applications rather than the performance within a given task specification.

Versatility is often used to mean teachability or reprogramming abilities.

Having established a basis for comparison, and having outlined the merits and the limitations of the four representative systems, we shall now attempt to evaluate the performance of our system with respect to these systems. The proposed system has similar performance characteristics with the first three systems in terms of timing , accuracy and flexibility. It lacks the versatility of the first two systems (limited range of applications at the present stage). However it is fair to mention that this is not an inherent disadvantage, since the modularity of its desing makes it potentially expandable to other application areas (mainly visual inspection).

On the other hand, it is capable of achieving these performance characteristics with less processing power (an 8-bit microcomputer),

COMPARISON TABLES

Table 7.3 PERFORMANCE CHARACTERISTICS

	SRI Vision Module	CONSIGHT (by General Motors)	BBC Optoelectronic Sensor	W.A. PERKINS ⁻ Model-based System	Proposed System
Positional - Directional Accuracy	-	-	10,5% of image height ≈ 1,5 pixel $\Delta\theta \leq \pm 1^\circ$	Max. pos. error: 6 pixels ($\pm 2\%$) Max. dir. error: $\pm 5.0^\circ$	Max. pos. error: ± 3 pixels ($\pm 2\%$) Max. dir. error $\pm 1^\circ$
ERROR RATES	-	-	-	0% (limited data, 24 samples)	Estimated range of error rate: 1% to 5% at 0% reject rate (148 samples)
FLEXIBILITY	HIGH (Vision Library routines combined with high-level goal directed main program)	FAIR	HIGH	FAIR	FAIR
VERSATILITY	FAIR	Limited	FAIR	FAIR	Limited
LIMITATIONS	Stringent require- ments on the scene to be imaged (trans- mitted light, fluorescent, conveyors, etc.).	The structured lighting arrangement essentially eliminates the need of transmitted light. However some internal features (holes) are still subject to occlu- sion due to the shadowing effect	High contrast necessary	Can handle visual noise. High contrast unnecessary. Can handle overlapping parts, but only some occlusion.	Can handle variations in light intensity and some visual noise. Cannot handle overlapping parts, or parts with a prominent three-dimensional structure.

hence at a lower cost.

Furthermore, it process a larger amount of original information (gray-levels instead of binary information), at a higher degree of sophistication (combination of gradient and binary techniques). This renders our system less prone to malfunctioning due to changes of local conditions (better "signal-to-noise" characteristics).

Although the use of gray-level techniques enhanced the ability of the system to work with visually-noisy scenes, it did not have the desired effect of complete insensitivity to working conditions (lighting, texture and colour of workpieces).

These limitations and the inability of the system to handle overlapping parts are its "weak" points in comparison with the system proposed by W.A. Perkins.

Yet we must not forget at what cost, both in terms of capital and of processing time these features of the admittedly highly sophisticated "Perkins" system were achieved.

CHAPTER 8 : CONCLUSIONS - SUGGESTIONS FOR FUTURE WORK

8.1. Conclusions

The remarkable interest that has developed the last few years for the application of image processing and pattern recognition techniques in industry has stimulated a worldwide research and development effort in two major directions:

a) Development of new sophisticated techniques and adaptation of existing theoretical models with the aim of solving the 'difficult' problems of three-dimensional computer vision, recognition of overlapping parts, general purpose automatic inspection, etc. In more general terms basic research in the field was, and still is, mainly concerned with the development of effective techniques to overcome the need of a structured environment. Although the problem of three-dimensional scene analysis has not yet been solved, significant advances have been made in the other two directions (image analysis of visually 'noisy' scenes, recognition of overlapping parts).

However, the proposed techniques were prohibitively expensive both in terms of processing requirements (mainframe computers), and in terms of operating speeds (from ten seconds to several minutes). The use of special purpose array processors and bit-slice microprocessors can provide realistic operating speeds, though still at a high cost.

b) Development of commercial vision systems based on a careful control of the environment (structured lighting, restricted range of suitable workpieces). These systems were implemented either on special-purpose hardwired logic, or on 16-bit microcomputers (mainly the early versions of the LSI-11).

The major feature of almost all existing industrial vision systems is that they process binary images obtained after thresholding of the incoming video signal. This functional restriction was mainly imposed by the cost of solid-state memories, and the need for a large data reduction in order to achieve real-time operation.

The evident gap between the highly-sophisticated experimental systems, and the low-level commercial vision systems can now be bridged in two ways; either by the exploitation of new technology (VLSI special purpose processors, the new generation of 16/32 bit microprocessors, analog microprocessors, etc.), and/or the development of new medium-level techniques of image analysis.

The aim of this project was to investigate the feasibility of developing a low-cost medium-level vision system using an 8-bit microprocessor with 16-bit capabilities.

The proposed system can be classified as a medium-level vision system, in the sense that it differs from existing commercial systems in that it employs a combination of gray-level and binary techniques for image segmentation. On the other hand the level of complexity of its image processing and pattern recognition techniques is significantly lower than that of the high performance experimental systems.

Furthermore, judging from its processing and memory requirements, it can be definitely termed as a low-cost alternative, achieving real-time operation with less stringent restrictions on the working environment than those imposed by purely binary systems. It cannot handle overlapping parts, or parts with pronounced three-dimensional structure.

It can tolerate a certain amount of visual noise and variations in the lighting conditions. It can successfully locate objects with varying surface reflectivities, as long as they have the same 'colour' relationship with the background ('white' objects on black background and vice-versa).

In the course of developing suitable techniques to achieve an acceptable compromise among cost, performance, and operating speed, we introduced the concept of a 'mirror' image, and we proposed a new contour follower and a new method of description for internal features (holes and protrusions).

The mirror image consists of numerically coded 'traces', which can be used for inter-process communication (information obtained during a processing stage is implanted in the original image for use

by the next stage). The 'mirror' image can be seen as an 'intermediate' memory structured in the same form as the original image array. The use of coded traces by the proposed system is limited (sealing of processed outlines, storage of the chain-code along the points of the outlines), but it can be extended to more complex operations.

The proposed contour follower employs combined binary and gradient (magnitude and direction) local information to select the most probable of three candidate steps leading to a new point of the outline of an object. At each new point we evaluate a local intermediate intensity level which is averaged with the intermediate intensity level obtained at the previous point, and the result is taken as a reference level for the topological classification of pixels around the new point.

This iterative definition of the reference level provides a means of adaptation to slight contrast variations along the border, resulting in a smoother outline.

Finally the description of internal features (holes and protrusions) is based on global attributes of the whole set of internal points considered as an entity (internal pattern). This unified description is less sensitive to occlusions of individual holes, and geometrical distortions of their shape, or their relative positions. A fast algorithm establishing the degree of connectivity for each internal point is used for noise-cleaning. The same algorithm evaluates the contribution of each valid internal point to the total perimeter and the total area of the internal pattern.

The software is written in 6809 assembly code and is structured in a modular form to enhance the flexibility and the expandability of the system. The functional modules along with a set of utility commands are integrated in an interactive operating system, which provides the user with basic facilities for development and evaluation purposes. The performance of the present experimental system was evaluated in terms of speed, accuracy (positional and directional) and reliability (detection and identification errors), and it was compared with the performance of existing systems.

The results of the comparison study showed that, even in its present form, the proposed system offers a competent alternative at a lower cost, and with a high potential for expansion.

Proposals for further development, optimization, and hardware realization are given in the next section.

The development of vision systems for the industry is a rapidly growing process with new achievements being reported at an accelerating rate, especially in the last two years.

The latest development can be grouped in three broad classes:

a) Systems based on template matching routines realized by hardwired logic: New hardwired vision systems are still being proposed and marketed, though at a diminishing rate (Ref. 49,50) They offer very high speed of operation, but they lack in flexibility and versatility which are essential requirements for an industrial vision system with wide scope.

b) General purpose development systems: These systems provide a flexible and comprehensive set of software and hardware tools to aid a vision expert in the design of special purpose vision systems (Ref. 53,54)

However, the final choice of a suitable combination of software and hardware components rests with the designer. The cost of these systems is generally too high, so that to use them as they stand for a particular application is definitely an 'overkill'.

c) General purpose vision systems: The prevailing trend in the field seems to be the exploitation of the latest developments in microprocessor technology, especially the MOTOROLA MC68000 16/32 bit single-chip microprocessor. A combination of very fast pre-processors (microcoded bit-slice processors, convolution processors, or special-purpose hardwired logic), a 16/32 bit microprocessor, and a fast frame-buffer memory, is now a reasonable configuration in terms of cost, and it offers processing capabilities enabling the implementation of sophisticated image analysis algorithms.

Unpublished information concerning two such systems (the AUTOVISION SYSTEM by Victor Scheinman of Automatix Inc, USA, and a system developed by W.K. Giloi for INTERNATIONAL ROBOMATION/INTELLIGENCE), reveals the following important points:

- 1) The dramatic progress in VLSI technology make gray-scale image analysis systems feasible that combine high performance with low cost.
- 2) The trend in the design of these new high performance vision systems is to apply existing sophisticated image analysis and pattern recognition techniques, which were originally implemented on large computers.

The author believes that this work proves , to a certain extent that the implementation of gray-scale image analysis can be carried out at an even lower level of cost and complexity, provided that the exploitation of new technology is accompanied with parallel development of what we call medium-level techniques, either from scratch, or as result of a reasonable simplification of existing highly sophisticated techniques.

This concept combined with the concept of software and hardware modularity may help in the desing of future vision systems that will be easily reconfigurable and cost optimized for a wider range of applications.

8.2. Suggestions for future work

Vision systems have a long way to go to reach the status of a mature technology and to gain universal acceptance in the world of industry. On the other hand the present state of the art is more than adequate for a wide range of important applications, a fact that has already been proved in practice.

Consequently, numerous suggestions can be made for future work directed either to the development of new techniques, or the application

of existing techniques.

We shall make an attempt here to give some guidelines concerning improvements of the proposed system, expansion of its capabilities, and requirements for the hardware realization of a stand-alone derivative.

8.2.1: Immediate improvements: The system at its present form cannot handle two-tone objects; as for example objects consisting of a black part and a gray part.

The same applies to objects with a prominent three-dimensional structure, resulting in shadows or sharp reflections extending across the viewed surface.

This limitation stems partly from the limited intensity resolution (16 gray levels), but mainly from the serial nature of the contour follower. If two branches originate from a contour point, one being the continuation of the outline, and the other being an internal contour, the contour follower may be misdirected, since it relies mainly on local information.

To overcome this limitation, the contour follower must be equipped with some means of identifying such branches, probably by the incorporation of a look-ahead provision.

Several other immediate improvements may be suggested, such as:

- a) Introduction of new features, especially features relating to the convexity of the outline, the relative location of points of high curvature (corners) etc.
- b) Incorporation of a multiple local template-matching technique. A simplified version of this technique was tried at the early stages of this work. The outline of an object was modelled by a network of points defined by the operator during training.

Two different schemes of matching were tried. The first scheme involved a superposition of the network over the tested outline, the other involved a series of predefined paths connecting the points of

the network. Both schemes produced good results, however they were abandoned at that time, because they required a 360 degrees rotation of the network to establish the angle of best fit.

By making use of the available orientation information in terms of the directions of the principal axes, an improved version of these techniques may yield a powerful global shape descriptor.

c) Development of an alternative classification technique based on the automatic generation of binary trees.

8.2.2. Expansion of the system: The system at its present form cannot be used for inspection applications. Such applications have been investigated in a parallel project, conducted by Mr F. Maali. The techniques that have been developed in these two projects can now be integrated under a common operating system. Alternatively the existing structure can be expanded to cover inspection applications.

Any future expansion to such application areas will necessarily involve:

- a) Higher spatial resolution, variable resolution, automatic selection of windows of high resolution.
- b) New interactive learning procedures.

Another direction for expansion would be the further development of the operating system (ICAROS). To facilitate this development the existing software should either be translated to a suitable high-level language (Pascal, or the more suitable for such applications "C" language), or it should be structured in the form of a special-purpose command language, which will supervise the sequencing of the existing algorithms, as well as allow simple 'if' conditions, loops and subroutines of commands to be formed.

A new set of interface commands should be developed for the control of the flow of information between the vision system and the end-effectors

of a flexible automation workstation (robots, conveyor belt controllers, etc).

8.2.3: Hardware realization: A block diagram for the proposed hardware configuration is given in fig. 8.1. The main components of a stand-alone system could be the following:

1) A fast digitiser incorporating a 6 or 7 bits video A/D converter (ultra-fast A/D converters with conversion rates of 20-30 MHz are readily available at comparatively low prices), 64K bytes of frame store (8 bits per byte), control and interface circuitry. For a maximum spatial resolution of 256 by 256 pixels the access cycle of the frame store should be less than 200 nsec. Dynamic memory chips available at present have a maximum capacity of 64 K bits at 120-150ns access cycle. Slower chips can be used, provided that the frame store is organized so that for instance 2 or 4 bytes are written or read simultaneously.

Desirable additional features of the digitizer would be:

- .programmable multiple camera interface to accomodate various types of video inputs.
- .variable spatial and intensity resolution ranging from 64 by 64 7-bit pixels, to 512 by 512 4-bit pixels.
- .a monitor interface to enable the display of the contents of the image array on an ordinary TV monitor.

A simplified block diagram of the digitiser is given in fig. 8.2. The data transfer from the digitiser to the processor must be performed via a Direct Memory Access link.

- 2) A 6809B 2-MHz Microprocessor
- 3) An AM9511 floating-point processor
- 4) 48K bytes of workspace RAM and 12K of system ROM.
- 5) An IEEE-488 programmable interface controller for the communication of the system with the end effectors (robots, conveyor controllers etc).
- 6) A floppy -disk controller.

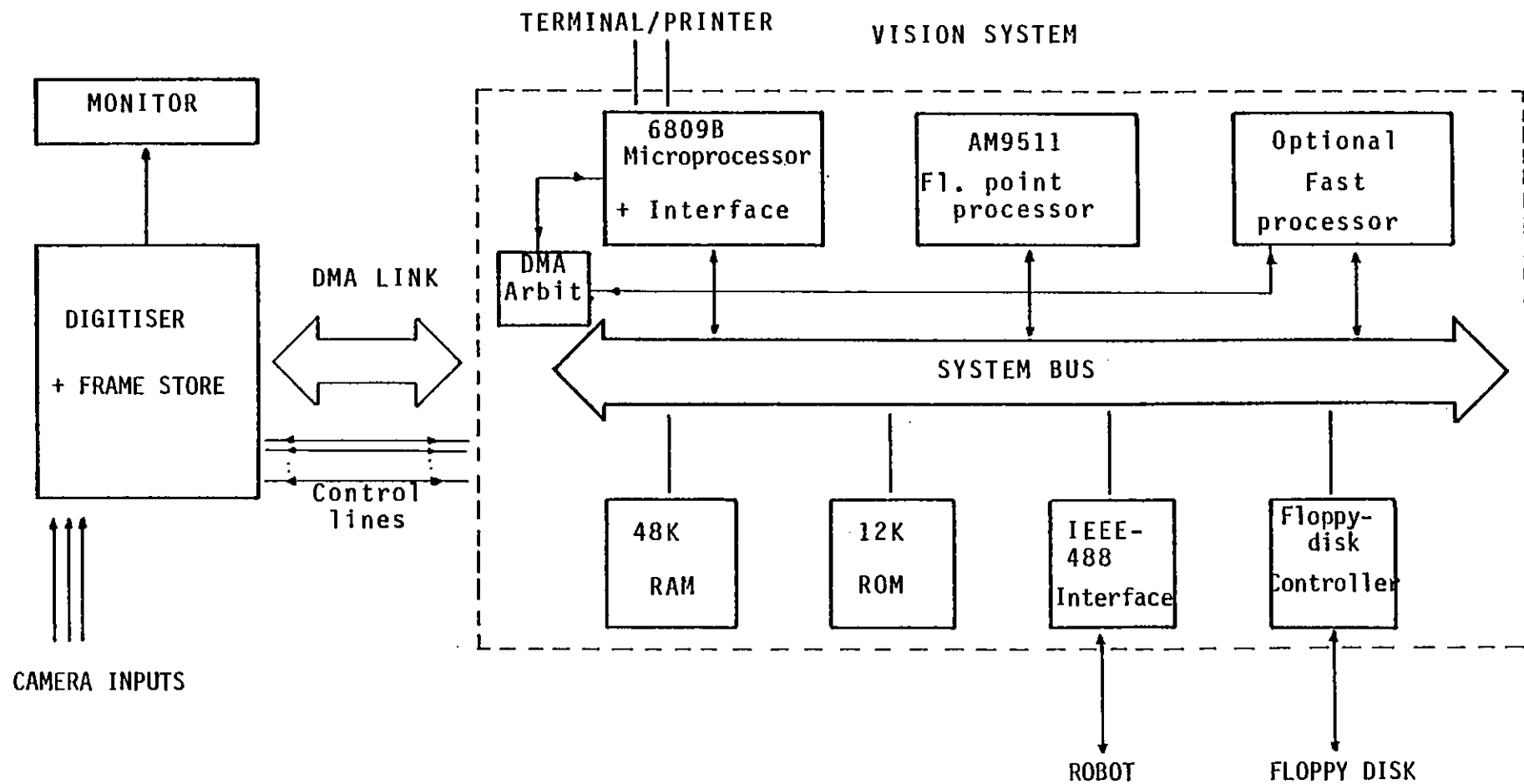


Fig. 8.1 : Block diagram of proposed hardware for a prototype Vision System and Digitiser.

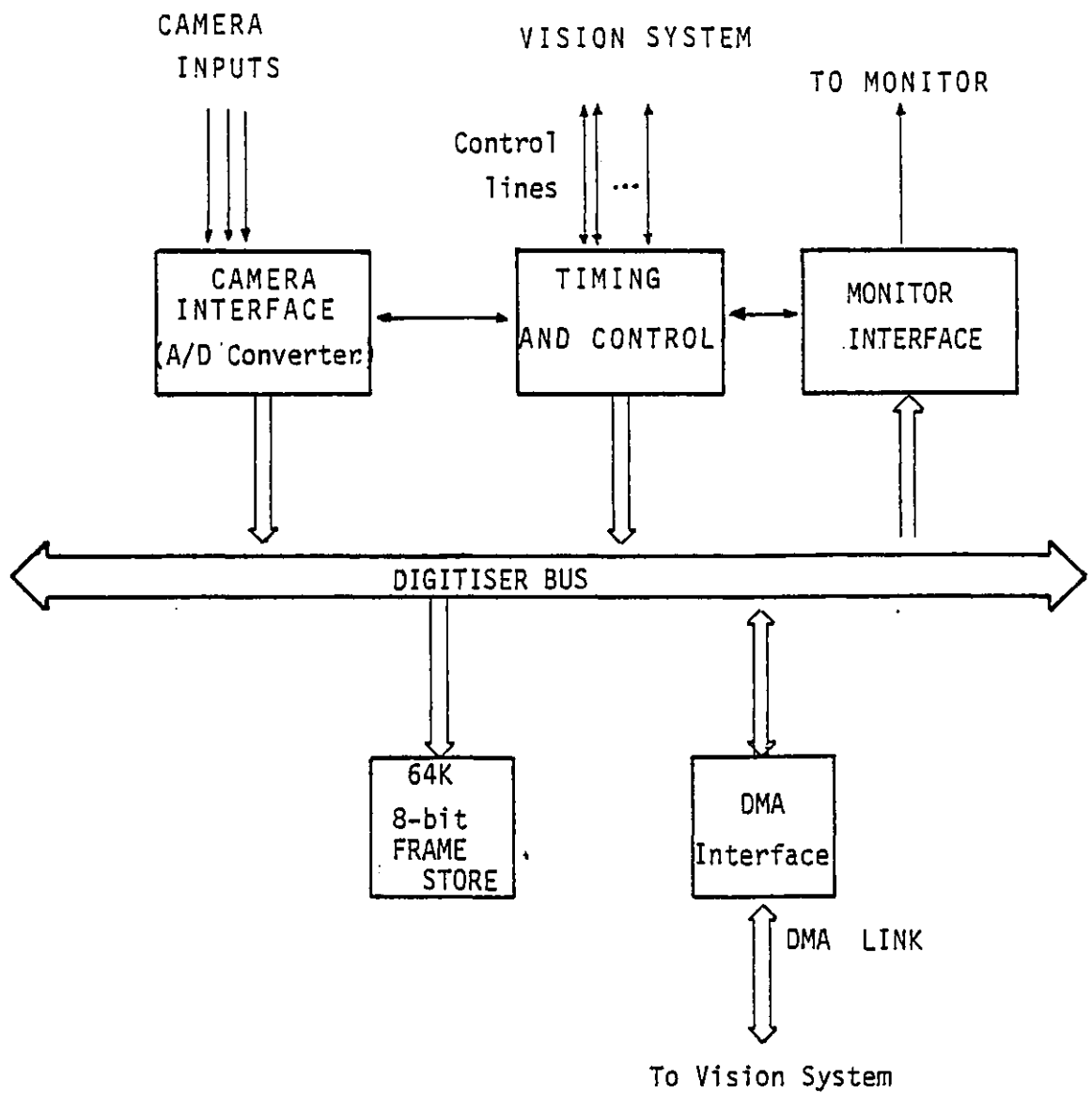


Fig. 8.2; Block diagram of a general purpose video digitiser.

The design of the hardware should be as flexible and as general as possible to allow future upgrade to a 16-bit based system, enhanced by a special -purpose processor (convolution processor, bit-slice processor or even array-processor). The cost of the basic system just described should be less than £5000 including the price of a single floppy-disk drive for the storage of models and vision library routines.

Apart from the development of the stand-alone vision system proposed in this thesis , future research in the field should be directed to the investigation of the use of other types of processors, such as CCD processors, analog signal processors or special-purpose vision processors that may become available soon . Although forecasts cannot be confidently made, we believe that the future, as far as the vision hardware is concerned, belongs more to the general-purpose processors, less to the special-purpose devices, and even less to the hard wired processors.

REFERENCES

1. "Robotics in the UK", The Industrial Robot, March 1981, p 32.
2. C.M. Lau, MSc Thesis, Dept of Management Science, Imperial College, University of London, September 1980.
3. J.L. Nevins, D.E. Whitney, "Robot Assembly Research", Computer Vision and Sensor Based Robots, edited by G.G. Dodd and L. Rossol, General Motors Research Laboratories, Plenum Press, 1979, p 285.
4. M. Salmon, A. d'Auria, "Programmable Assembly System", Computer Vision and Sensor Based Robots (G.G. Dodd and L. Rossol, G.M. Res. Labs), Plenum Press, 1979, p 160.
5. J.G. Bollinger, J. Bascum, "The Use of Tactile Sensing for the Guidance of a Robotic Device for Welding", pp 193-203. Proceedings of the 1st Int. Conf. on Robot Vision and Sensory Controls, April 1-3, 1981, Stratford-upon-Avon, UK.
6. J.A. Purbrick, "A Force Transducer Employing Conductive Silicon Rubber", pp 73-80. Proc. of the 1st Int. Conf. on Robot Vision and Sensory Controls, April 1-3, 1981, Stratford-upon-Avon, UK.
7. M.H. Larcombe, "Carbon Fibre Tactile Sensor", pp 273-276 Proc. of the 1st Int. Conf. on Robot Vision and Sensory Controls, April 1-3, 1981, Stratford-upon-Avon, UK.
8. C.J. Page, H. Hassan, "Non-Contact Inspection of Complex Components Using a Rangefinder Vision System", pp 245-259 Proc. of the 1st Int. Conf. on Robot Vision and Sensory Controls, April 1-3, 1981, Stratford-upon-Avon, UK.

9. Video Digitizer 270A, Colorado Video., Boulder Colorado 80306 USA.
10. E. Dummermuth, "Bit-Slice Microprocessors", pp 196-201 Machine Design, April 6, 1978.
11. R. Lumia, E.A. Parrish, "An Example of the Use of CCD Processor in Automation", IEEE Trans. on Instrumentation and Measurement, Vol. IM-28. No. 2, June 1979.
12. M.E. Hoff, M. Townsend, "Single-chip n-MOS Microcomputer Processes Signals in Real Time", Electronics/March 1, 1979.
13. I. Aleksander, T.J. Stonham, "Guide to Pattern Recognition Using Random Access Memories", Computers and Digital Techniques, February 1979, Vol. 2, No. 1, pp 29-39.
14. C.A. Rosen, "Machine Vision and Robotics: Industrial Requirements", Computer Vision and Sensor Based Robots, G.G. Dodd and L. Rossol, G.M. Research Labs, Plenum Press 1979, pp 3-18.
15. M.R. Ward, L. Rossol, S.W. Holland and R. Dewar, "CONSIGHT, An Adaptive Robot with Vision", Robotics Today, Summer 1979, pp 26-32.
16. P. Saraga, J.A. Weaver, "An Experiment in Flexible Automation", Phillips Technical Review 38, 1978/79, No. 11/12, pp 329-337.
17. W.A. Perkins, "A Model-Based Vision System for Industrial Parts", IEEE Trans. on Computers, Vol. C-27, No. 2, February 1978.
18. R. Kelley, J. Birk, D. Duncan, H. Martins and F. Tella, "Workpieces from Bins into Machines", Proc. 9th Intern. Symposium of Industrial Robots, March 13-15, 1979, Washington D.C., USA.

19. E. Johnston, "A CCTV Camera Controlled Painter"
IEE Colloquium on Industrial Applications for Robot
Vision 24 November 1981, IEE London.
20. R.G. Abraham, "APAS: Adaptable Programmable Assembly
System", Computer Vision and Sensor Based Robots,
Plenum Press 1979, pp 117-136.
21. D. Graham, "Vision in Automated Surface Finishing",
IEE Col. on Ind. Applications for Robot Vision,
24 November 1981, IEE London.
22. M.L. Baird, "SIGHT-1: A Computer Vision System for
Automated IC Chip Manufacture".
IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-8
No. 2, February 1978.
23. S. Kashioka, M. Ejiri, and Y. Sakamoto, "A Transistor
Wire-Bonding System Utilizing Multiple Local Pattern
Matching Techniques", IEEE Trans. Syst. Man, Cybern.,
Vol. SMC-6, pp 562-570, August 1976.
24. G.J. Gleason, G.J. Agin, "A Modular Vision System for
Sensor-Controlled Manipulation and Inspection",
9th Int. Symp. on Industrial Robots, March 13-15, 1979,
pp 57-69.
25. R. Karg, O.E. Lanz, "Experimental Results with a
Versatile Optoelectronic Sensor in Industrial Applications",
9th Int. Symp. on Industrial Robots, March 13-15, 1979,
pp 247-264.
26. A. Rosenfeld, M. Thurston, "Edge and Curve Detection for
Visual Scene Analysis", IEEE Trans. Comput., Vol. C-20,
pp 562-569, May 1971.

27. L.S. Davis, "A Survey of Edge Detection Techniques", Computer Graphics Image Processing, Vol. 4, pp 248-270, 1975.
28. M.H. Hueckel, "An Operator which Locates Edges in Digitized Pictures", J. Ass. Comput. Mach., Vol. 18, pp 113-125, January 1971.
29. R.S. Ledley, "High-Speed Automatic Analysis of Biomedical Pictures", Science Vol. 146, pp 216-223, October 1964.
30. E. Greanias, R. Meagher, R. Norman, P. Essinger, "The Recognition of Handwritten Numerals by Contour Analysis", IBM J. Res. and Dev., Vol. 7, pp 14-21, January 1963.
31. Y. Nagakawa, A. Rosenfeld, "Edge/Border Coincidence as an Aid in Edge Extraction", IEEE Trans. Sys., Man, Cyb, Vol. SMC-8, pp 899-901, December 1978.
32. B.J. Schachter, A.L. Steven, W. Zucker, A. Rosenfeld, "An Application of Relaxation Methods to Edge Reinforcement", IEEE Trans. Sys., Man, Cyb., Vol. SMC-7, pp 813-816, November 1977.
33. H. Freeman, "Computer Processing of Line-Drawing Images", Computing Surveys, Vol. 6, No. 1, March 1974.
34. M. Levine, "Feature Extraction: A Survey", Proc. IEEE, Vol. 57, pp 1391-1405, August 1969.
35. M.K. Hu, "Visual Pattern Recognition by Moment Invariants" IRE Trans. Inf. Theory, pp 179-187, February 1962.
36. G.J. Agin, "Computer Vision Systems for Industrial Inspection and Assembly", IEEE Computer, pp 11-20, May 1980.

37. E.A. Edmonds, A. Schappo, S.A.R. Scrivener, "Graphics Without Data Structures", Proc. 4th Int. Conf. on Computers in Design Engineering, CAD80, March 1980, Brighton UK, p 139.
38. P.W. Kitchin, PhD Thesis, University of Nottingham, October 1977.
39. R.O. Duda, P.E. Hart, "Pattern Classification and Scene Analysis", Wiley-Interscience.
40. P. Martini and G. Nehr, "Recognition of angular orientation of objects with the help of optical sensors", The Industrial Robot Vol. 6, pp 62-69, June 1979.
41. W.K. Pratt, "Digital Image Processing", Wiley-Interscience New York, 1978.
42. L. Kanal, "Patterns in Pattern Recognition: 1968-1974", IEEE Trans. on Inf. Th., Vol. IT-20, November 1974, pp 697-717.
43. P.E. Hart, "The Condensed Nearest Neighbor Rule", IEEE Trans. on Inf. Th., May 1968, pp 515-516.
44. A. Rosenfeld, "Connectivity in Digital Pictures", Journal of the Association for Computing Machinery, Vol. 17, No. 1, January 1970, pp 147-160.
45. M.D. Levine, "A Modular Computer Vision System for Picture Segmentation and Interpretation", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-3, September 1981, pp 540-555.
46. VICOM Digital Image Processor, VICOM Systems Inc, San Jose, California.
47. M.H. Lee, "Research into Flexible Task Performance in Manipulator Systems", 8th Int. Symp. on Industrial Robots, Vol. 2, pp 714-720.

48. FLEX Programmer's Manual, Technical Systems Consultants, Inc., 1979.
49. W.K. Taylor, G. Ero, "Real Time Teaching and Recognition System for Robot Vision", The Industrial Robot, June 1980, pp 99-106.
50. T. Costlow, "Plug-in Boards Add Vision Functions to Range of Robots", Electronics, June 30, 1982, pp 47-48.
51. R. Gray, C.B. Besant, "Digitising Line Drawings Using a Standard T.V. Camera", Proc. 4th Int. Conf. on Comp. in Design Engineering. CAD 80, Brighton UK, March 1980, pp 186-191.
52. C.S. Ow, R.I. Crane, "A Simple Off-Line Automatic Image Analysis System with Application to Drop Sizing in Two-Phase Flows", Int. J. Heat and Fluid Flow, Vol. 2, No. 1, 1980, pp 47-53.
53. J.W. Lockton, "Industrial machine vision: software modelling and hardware implementation holds the key" IEE Colloquium on "Applications of Machine Vision to Industrial Inspection", November 1982, London UK.
54. D.H. Mott, "The practical application of vision systems", IEE Col. on "Applications of Machine Vision to Industrial Inspection", November 1982, London, UK.
55. HP 2648A Reference Manual, Hewlett-Packard Company, Cupertino, California, 1979, Part No. 02648-90002.

APPENDIX A: COLUMN DIGITISER HARDWARE

DESCRIPTION OF THE SYSTEM

Basic details of the 625 line interlaced television scanning format are shown in fig. A1. Each frame consists of two fields of $312\frac{1}{2}$ lines. Commencing with the EVEN field the scan starts at the top left hand corner nominally at the start of line 1. Each line is scanned from left to right under the action of the line timebase, the return to the start of the following line being a rapid fly-back. The first $22\frac{1}{2}$ lines are blank, video data commencing at the centre of line 23. Scanning continues in numerical order and video data ends at the end of line 310. After $312\frac{1}{2}$ lines, frame fly-back occurs returning the scanning point to the upper left hand corner of the picture, when it commences to scan the ODD field, lines $312\frac{1}{2}$ to 625. In this field, video data starts at the start of line 336 and ends at the centre of line 623, at the end of line 625 frame fly-back again occurs, the scanning point returns to scan the next even field. In addition to the 25 blank lines inserted between video data in successive fields, the first $10.5\mu s$ and last $1.5\mu s$ of each line are also blanked, these blank periods are included to allow for the fly-back and settling time of the horizontal and vertical time base amplifiers.

To convert the signal obtained from the television camera into data suitable for computer processing using inexpensive components it is necessary to use relatively low data transfer rates. So the operation of the system is to sample the amplitude of the video signal once per scan line and convert the value obtained to a 4 bit binary number. The process starts on the left hand edge of the picture and in the course of two fields each useful video line is sampled, the sampling point is then stepped one increment horizontally to the right and the sampling operation repeated and so on till the right hand edge of the picture is reached.

It was decided to aim for a system taking about 1000 samples across each horizontal line, this giving a resolution somewhat in excess of the resolution obtainable from a good quality camera

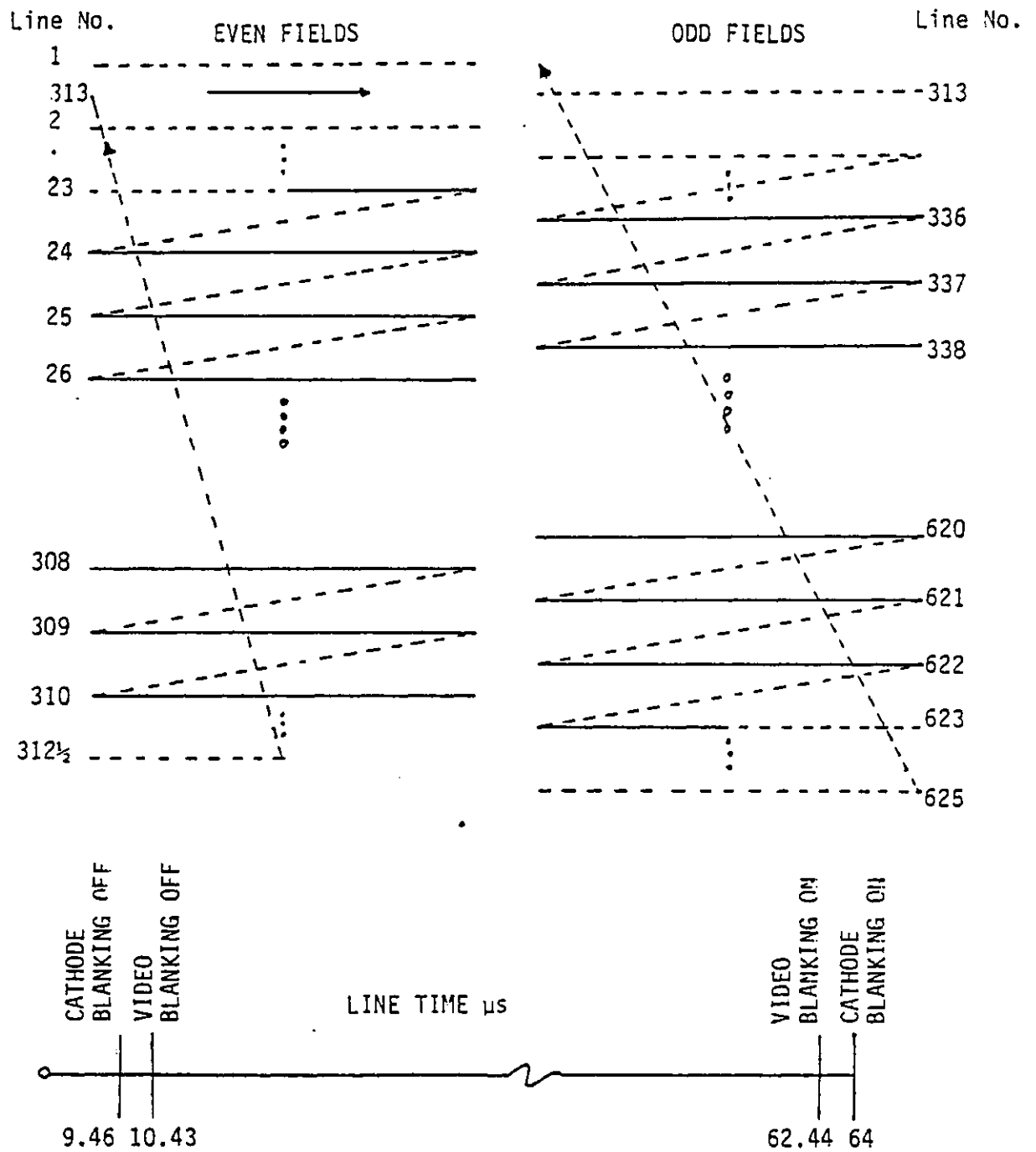


Fig. A1 : 625 line , 50 fields/sec interlaced TV scan.

(about 800 lines). As each scan line takes $64 \mu\text{s}$ of which $52 \mu\text{s}$ contain video information a basic clock frequency of 20 MHz is required. An integrated circuit is available from Ferranti which provides all the synchronization and blanking pulses required for a closed circuit television system. This circuit requires a drive signal of 2.625 MHz for 625 line 50 field/second operation, this figure when multiplied by 8 gives 20.50 MHz which was therefore chosen as the basic clock frequency. A block diagram of the equipment is shown in fig. A2 . The operation is as follows:

When it is required to digitize a picture the computer sends a CLEAR signal which sets the even field pulse counter and clock pulse counter to correspond to the upper left hand corner of the picture. The computer then sends a CONTINUE signal and after the next even field pulse, the clock pulse counter is loaded with the content of the even field pulse counter, it is then decremented by clock pulses until it underflows , this generates a SAMPLE pulse to the analogue to digital converter which converts the magnitude of the video signal to a 4 bit binary value and then produces a CONVERSION COMPLETE pulse, which transfers the data to the shift register store. At the end of the video line the clock pulse counter is reloaded with the same value and the next lower video is sampled, thus after the first field the store contains 288 sets of data corresponding to the 288 useful video lines of the first field. On the second field the data is passed direct to the output circuit, where it is interleaved with the data being moved out of the shift register store and is presented to the computer as a series of 144 , 16 bit words , a READ signal being sent to the computer each time a fresh data word is presented. After two fields an EVEN FIELD pulse increments the even field pulse counter and the procedure is repeated one increment to the right. When $128 \times$ values have been processed the cycle stops to allow the computer to transfer the information to disk. The computer then generates another CONTINUE pulse and a further $128 \times$ values will be sampled , eight such blocks of information constitute a complete picture , after which the system will halt. The time taken to acquire a full picture is about 45 seconds.

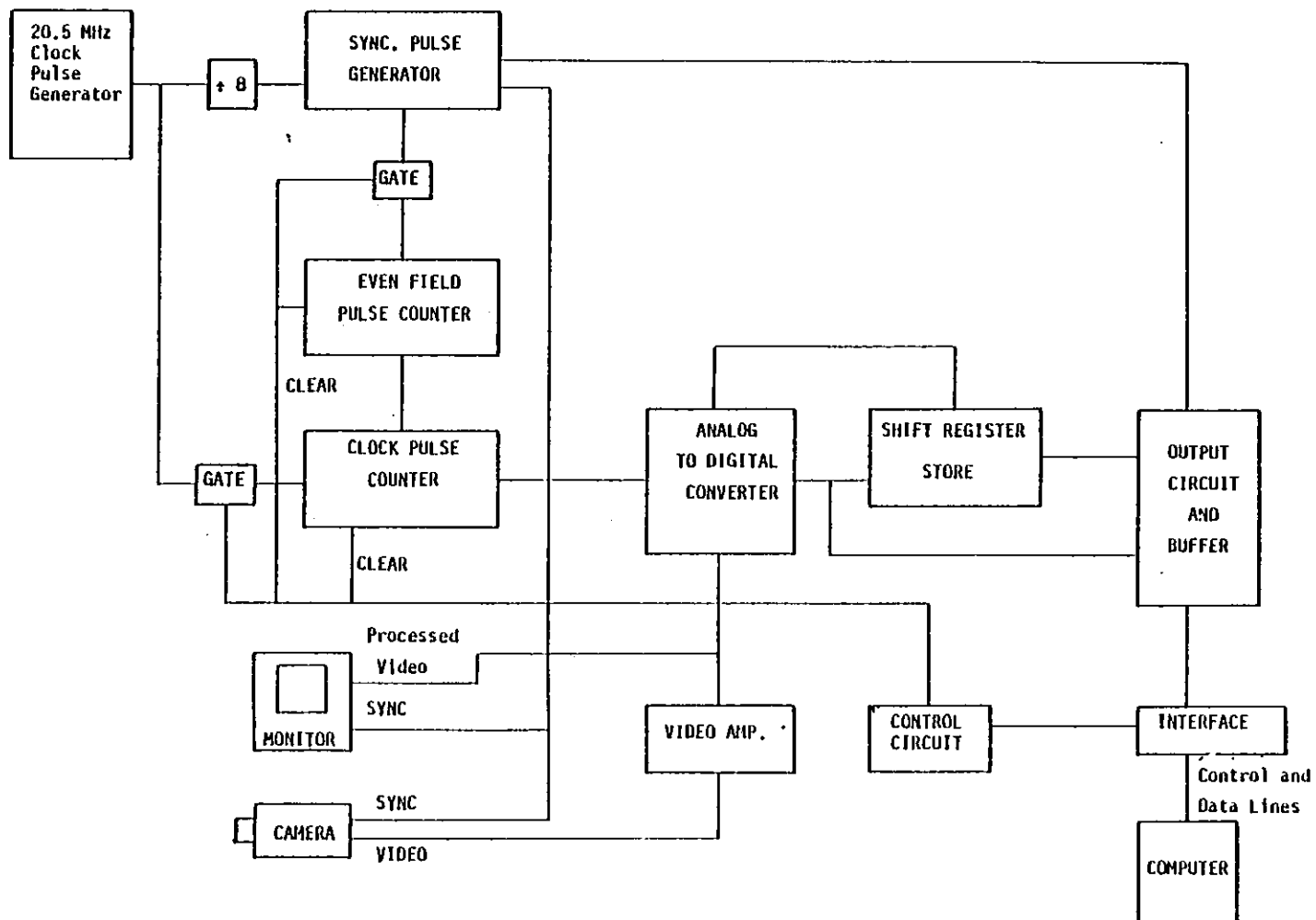


Fig. A2 : Block Diagram of Column Digitiser.

The system has been implemented using integrated circuits the majority being Low Power Schottky TTL, Cmos shift registers and special circuits for the sync pulse generator , and analogue to digital conversion have also been used.

APPENDIX B: VISION UTILITY COMMANDS

DIG: Digitiser interface routine controlling the flow of data from the digitiser to the computer via the MC 6820 Peripheral Interface Adapter (PIA).

A total of four control lines are used for the synchronization of the data transfer. The computer waits in a polling loop until it gets a NDA (Next Data Available) signal from the digitiser , indicating the presence of new data on the 8 parallel data lines.

Since each pixel is represented by 4 bits, an 8-bit word contains two successive pixels of the digitised column. The duration of a horizontal scan line is 60 μ secs including the blanking period. Therefore two new pixels are available approximately every 120 μ secs, which gives the computer enough time to read, rearrange and store the incoming data in the image memory.

At the end of the 128th column the digitiser sends an EOP (End Of Pass) signal signifying the end of a digitisation pass.

The computer responds with a CONT (CONTINUE) signal which initialises a new pass.

The digitiser has a horizontal resolution of 1024 pixels, obtained in 8 passes, but, since the required horizontal resolution by the vision system is only 128 points, the incoming data is combined selectively to yield the required resolution (128 by 96).

This shrinking mechanism is accompanied by a smoothing operation performed by the 8 by 6 smoothing and selection mask illustrated in fig. B1.

The zero coefficients simply denote the skipping of the corresponding column or line.

The interface routine also creates the column directory and the image directory described in section...

WRITE: This command is used to transfer the contents of the image memory on a disk file in exactly the same format (image directory - column directory - image array).

The operator (or the program) defines the name of the image file along with the disk drive and the extension (e.g. WRITE 1: TEST1 . PIC).

READ: The contents of a binary image file can be transferred from the disk to the image memory using the command READ (e.g. READ 1: TEST1. PIC).

PRA : An image array stored in the image memory can be displayed on an ordinary terminal using user-defined ASCII characters to represent the 16 gray levels , or a hardcopy of the image array can be obtained from an ordinary text printer.

The operator can select a portion of the image array to be displayed, by defining the coordinates of the top-left pixel and the dimensions of the rectangular portion (see fig. B2).

The format of this command is the following

```
(P,) PRA X1, Y1, DX, DY, -----% ,
                                intensity field
,-----
    trace field
```

The letter P precedes the command if the selected output is the text printer. Up to 16 different ASCII characters can be assigned to corresponding intensity levels , and another 16 to the relevant trace codes.

Example : PRA 10,10, 60,60, **** %.. X *. A 60 by 60 portion of the image array will be displayed starting from pixel (10,10). The first four intensity levels (0,1,2,3) will be marked with a "black" symbol (*) thus creating the impression of a binary thresholded image (Th=4). All points of the image array that have been painted during processing with traces having code numbers 2 and 3 will be marked by an X and an

$$\begin{bmatrix} 1 & 0 & 3 & 0 & 3 & 0 & 1 & 0 \\ 3 & 0 & 4 & 0 & 4 & 0 & 3 & 0 \\ 1 & 0 & 3 & 0 & 3 & 0 & 1 & 0 \end{bmatrix}$$

or

$$\begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 4 & 4 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix}$$

Fig. B1 : Smoothing and data selection mask
used for the reduction of the incoming
image data (from 1024 by 288 pixels to
128 by 96 pixels). The resulting value
for each pixel is normalised by 32.

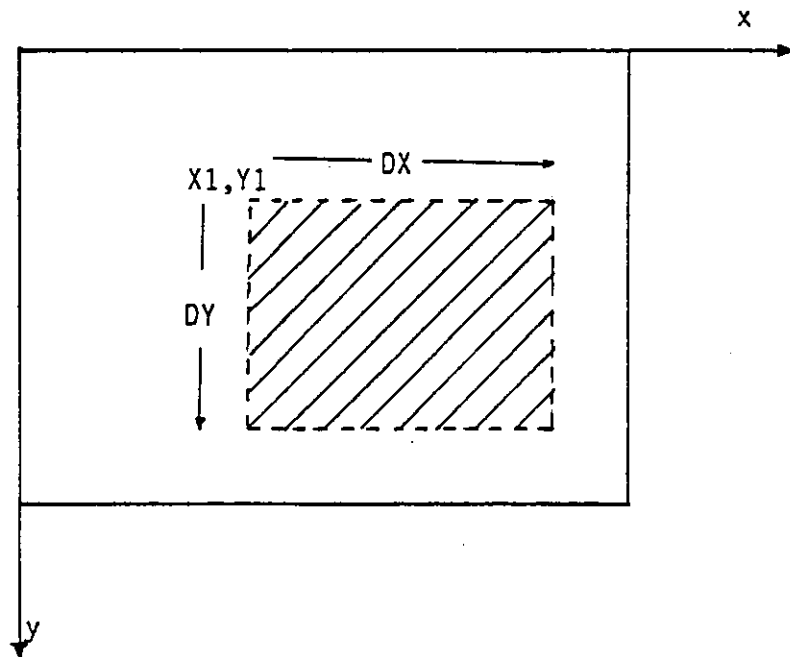


Fig. B2 : Selection of window of interest
for display purposes (Command PRA).

asterisk (*) respectively.

All the other intensity levels and all the other traces will not be displayed.

HPDIS: A dot-matrix display of a binary thresholded image can be obtained on an HP 2648A Graphics Terminal using the programmable area fill mode (Ref....)

A 128 by 96 matrix is segmented in 192 (16 by 12) sub-arrays each consisting of 8 by 8 dots.

The intensity levels (0= black, 1=white) of the 64 pixels within each of the sub-arrays are displayed by a dot-pattern defined by 8 8-bit words.

Each word corresponds to the dot-pattern of a line of the sub-array.

Hence the 8 numbers defining an all-white sub-array will be: (256 (11111111) , 256, 256, 256 256, 256, 256,256.

The operator (or the program) defines the position of the dot image array on the HP screen (one out of 6 fixed positions), the binary threshold and a trace code that he may desire to be displayed.

If a trace code is defined,HPDIS will only display all the specified traces and not the intensity information.

Example: HPDIS 1, 40, 0 will display a binary image where all intensity levels greater than 4 will appear as white dots.

HPDIS 1, 40, 80 will display all trace codes with the MS bit set, i.e. all trace codes from 8 to 15

APPENDIX C: FEATURE LISTS OF TEST OBJECTS

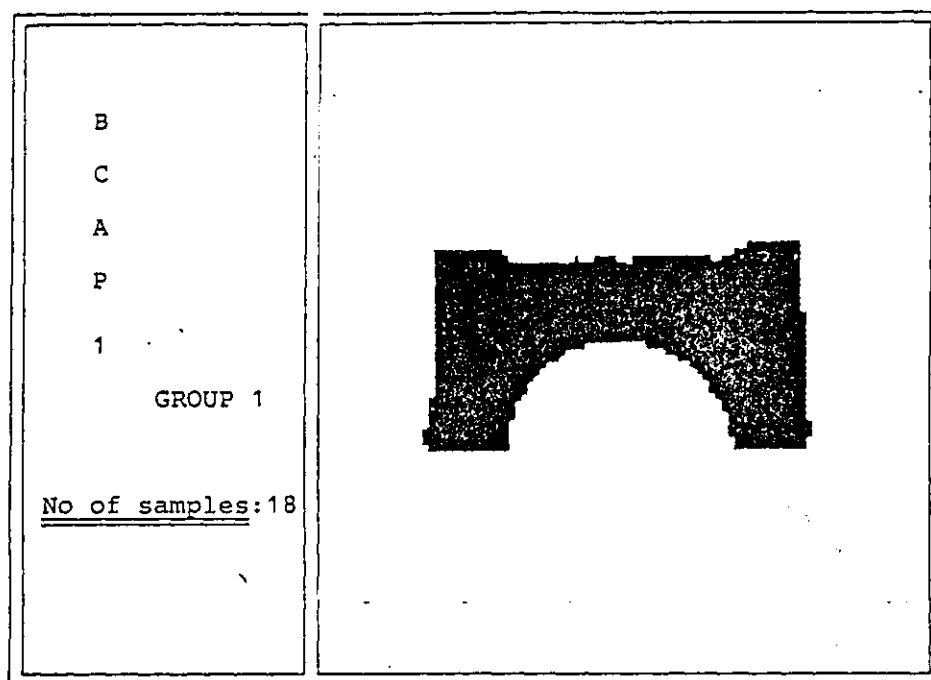
Nine different objects were used as a test set for the evaluation of the detection and identification accuracy of the system (section 7.2.1). One of them, the bearing cap has two stable positions on a plane, i.e. two different "faces" (BCAP2).

The following feature lists summarize the orientation independent primitive features of the model shapes (i.e. all the measured primitive features, except the second moments) and the derived compound features A,B,C,... (see section 5.2.1. for a definition of the compound features).

The pictures accompanying the feature lists are binary thresholded images of the objects, obtained with transmitted light.

The ten model shapes are grouped in two sets according to their physical dimensions. The first group (GROUP1) includes the shapes of four "small" objects (SPAN , KEY , PLUG, COIN), the second group consists of six model shapes pertaining to five larger objects (BCAP1 , BCAP2 , CROD , CIRCLE , RECT , OMEGA).

Two typical scenes, one with transmitted light and one with top lighting are shown in fig's C1 and C2.



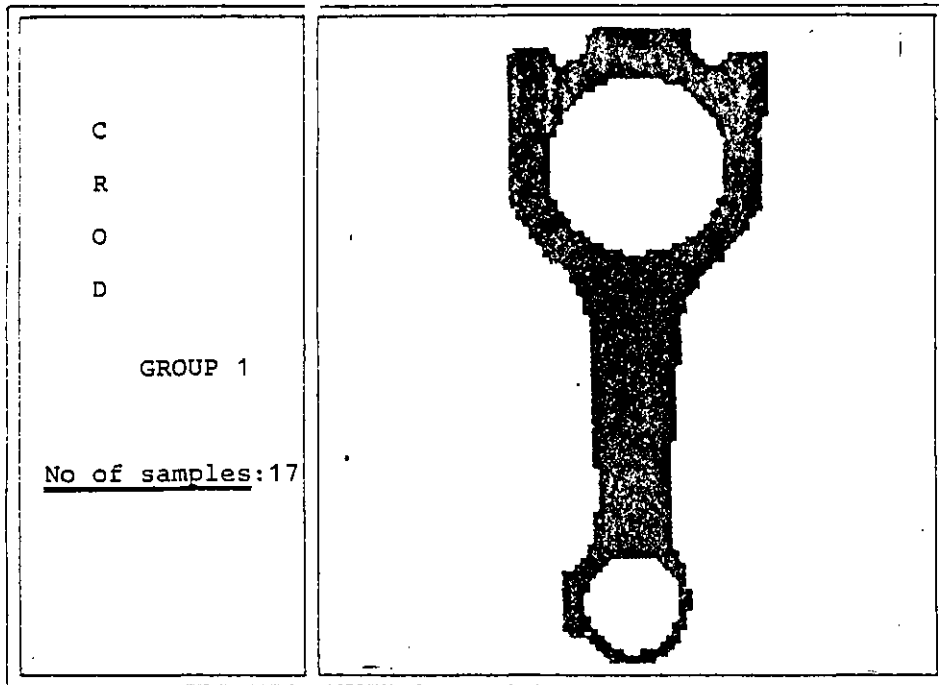
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	113	2.7	2
AREA	463	20	4
MAXDIS	19	0.6	3
MINDIS	.8	0.5	60
AVRDIS	13	0.3	2
PERH	0	0	0
AREAH	0	0	0

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.4531	.0124	3
B	.0377	.0016	4
C	.4807	.0281	6
D	.9351	.0394	4
E	1.1494	.0084	1
F	6.51×10^{-3}	2.35×10^{-4}	4
G	0	0	0
H	0	0	0
I	0	0	0
J	0	0	0

B C A P 2 GROUP 1 <u>No of samples:18</u>	
---	--

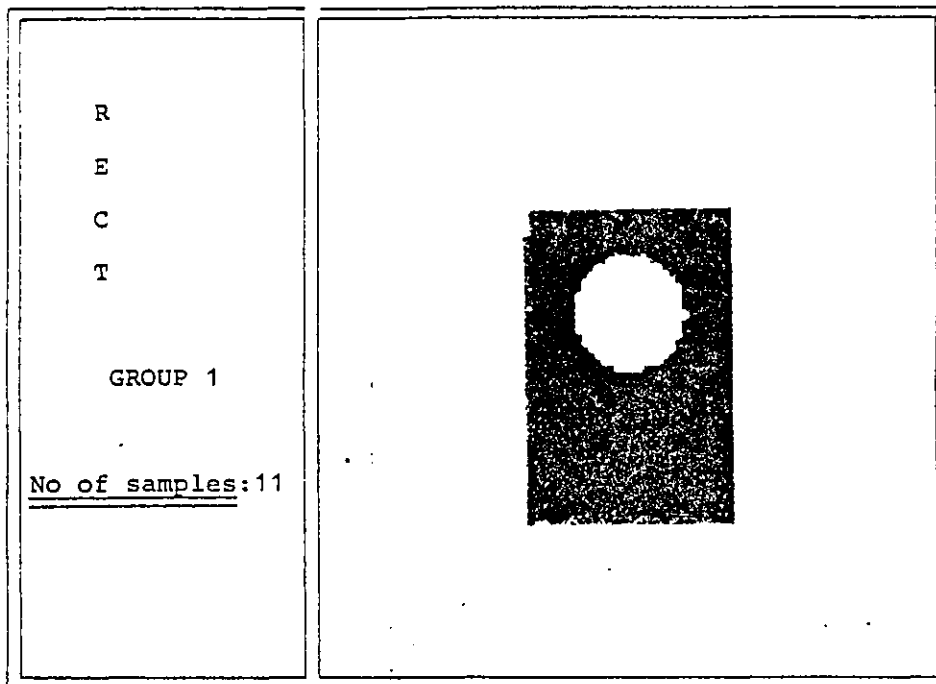
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST.DEVIATION	% DEVIATION
PER	82	2	3
AREA	263	23	9
MAXDIS	18	0.5	3
MINDIS	3.4	0.5	15
AVRDIS	10.7	0.2	2
PERH	9.3	3.2	34
AREAH	4	1.7	42

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.485	.036	7
B	.021	1.8×10^{-3}	8
C	.676	.039	6
D	.683	.043	6
E	1.206	.016	1
F	4.4×10^{-3}	1.9×10^{-4}	4
G	.703	.506	70
H	.749	.063	8
I	.04	.01	35
J	.22	.20	92



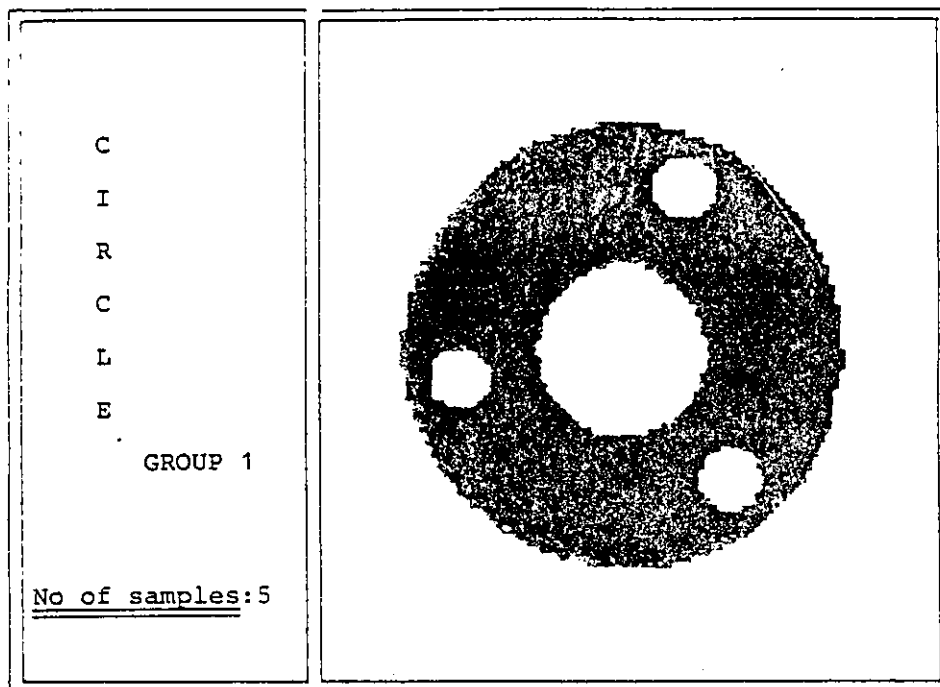
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	131	22	1
AREA	587	9	1
MAXDIS	26	1	3
MINDIS	3.8	0.4	10
AVRDIS	16	0.1	1
PERH	61	3	5
AREAH	129	3	2

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.428	.014	3
B	.048	7.7×10^{-4}	2
C	.673	.043	6
D	.759	.024	3
E	1.192	6.4×10^{-3}	0.5
F	9.8×10^{-3}	1.7×10^{-4}	2
G	.437	.046	10.5
H	.367	.015	4
I	.271	.008	3
J	.222	.027	12



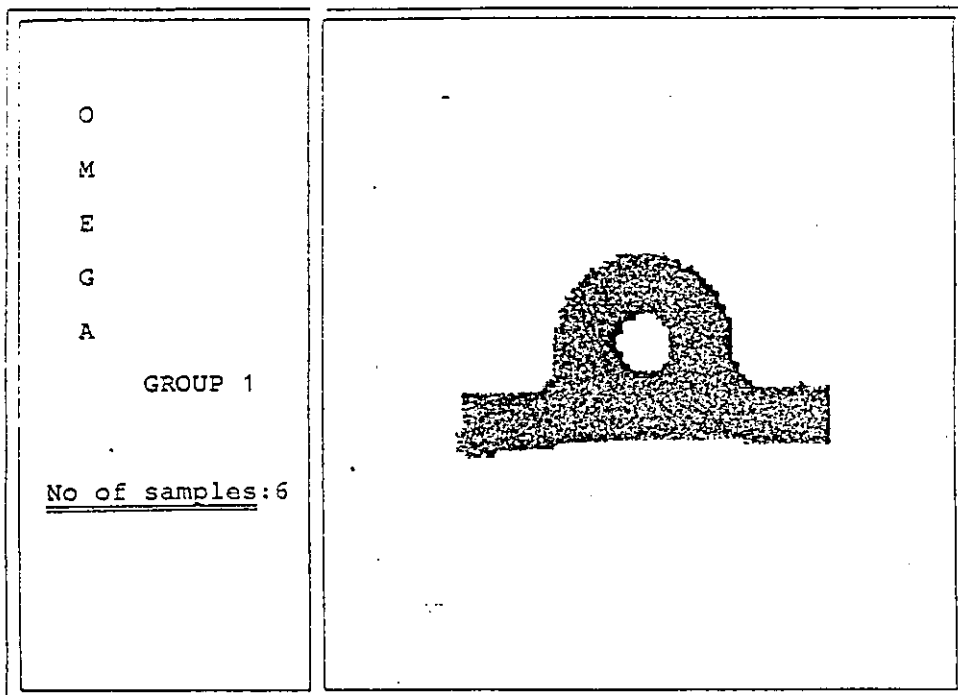
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	69	1	2
AREA	306	14	5
MAXDIS	12.8	0.4	3
MINDIS	6.7	0.4	7
AVRDIS	10	0.2	3
PERH	16	1	7
AREAH	19	3	16

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.800	.029	4
B	.025	1.1×10^{-3}	4
C	.278	.036	13
D	.329	.043	13
E	1.033	8.3×10^{-3}	1
F	3.9×10^{-3}	1.9×10^{-4}	5
G	.921	.062	7
H	.056	.012	22
I	.092	.015	16
J	.202	.025	12



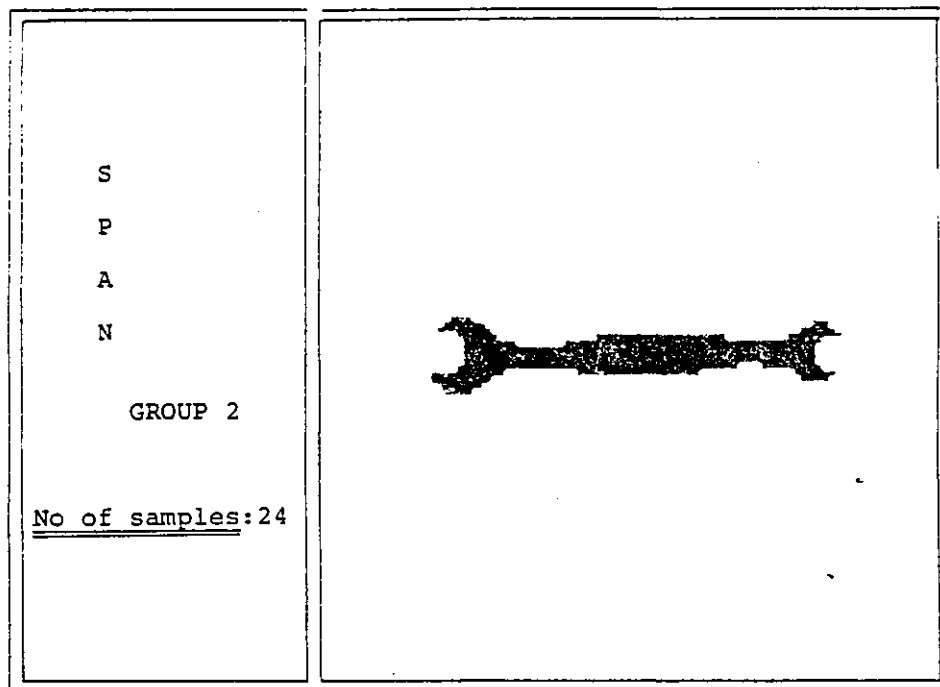
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	129	1	1
AREA	1222	15	1
MAXDIS	21	0	0
MINDIS	18.8	0.4	2
AVRDIS	19.7	0.1	1
PERH	83	2	3
AREAH	165	4	2

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.920	.012	1
B	.099	.001	1
C	.066	.006	9
D	.046	.017	38
E	1.003	.001	0
F	.015	1×10^{-4}	1
G	.298	.008	2
H	.069	.003	4
I	.169	.004	2
J	.028	.013	46



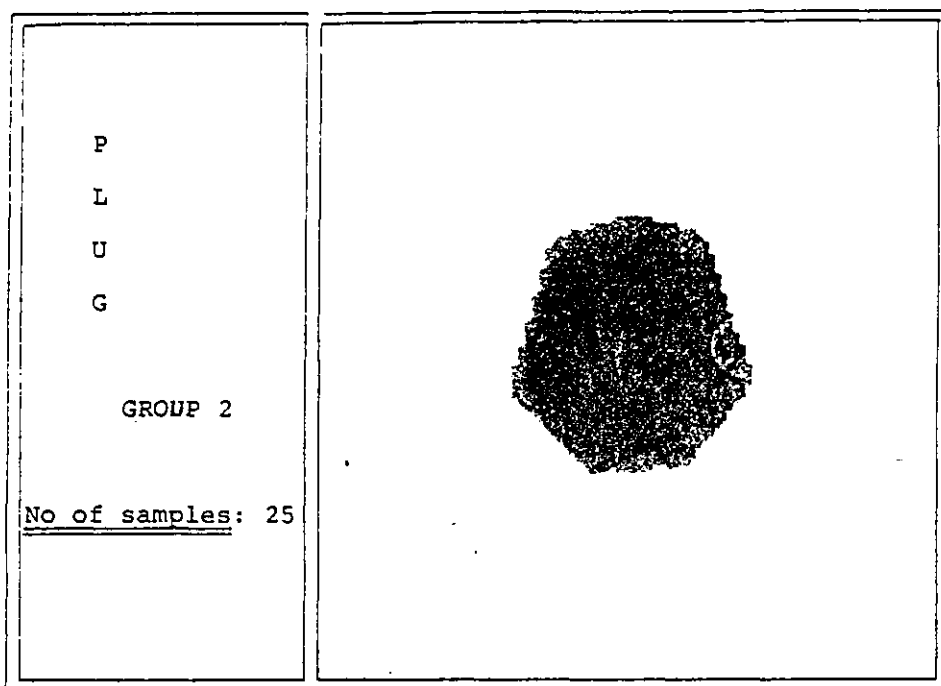
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	64	2	3
AREA	193	7	4
MAXDIS	12.6	0.5	4
MINDIS	4.3	0.5	11
AVRDIS	8.2	0.1	2
PERH	6	1.3	20
AREAH	2.8	1.1	40

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.587	.028	5
B	.016	5.1×10^{-4}	3
C	.534	.047	9
D	.474	.063	13
E	1.084	5.6×10^{-3}	0.5
F	2.6×10^{-3}	9×10^{-5}	3
G	.898	.145	16
H	.037	.011	29
I	.036	9×10^{-3}	25
J	.176	.028	16



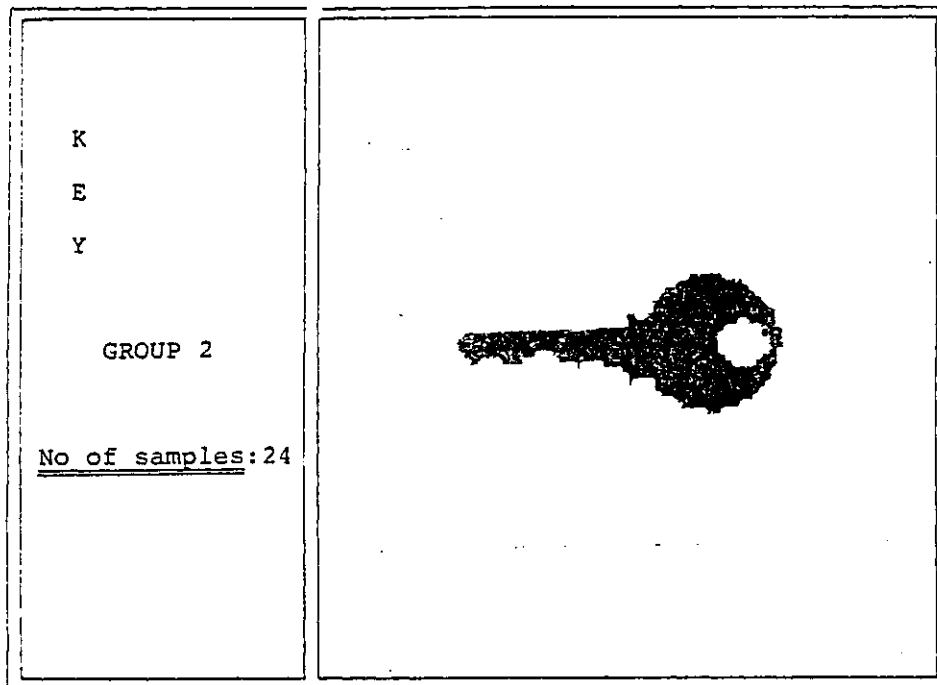
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	107	4	4
AREA	202	11	6
MAXDIS	22	2	7
MINDIS	2.1	.1	5
AVRDIS	12.3	.7	6
PERH	0	0	0
AREAH	0	0	0

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.221	.020	9
B	.016	9×10^{-4}	5
C	.77	.06	8
D	.83	.04	4
E	1.26	.02	1
F	.006	6×10^{-4}	10
G	0	0	0
H	0	0	0
I	0	0	0
J	0	0	0



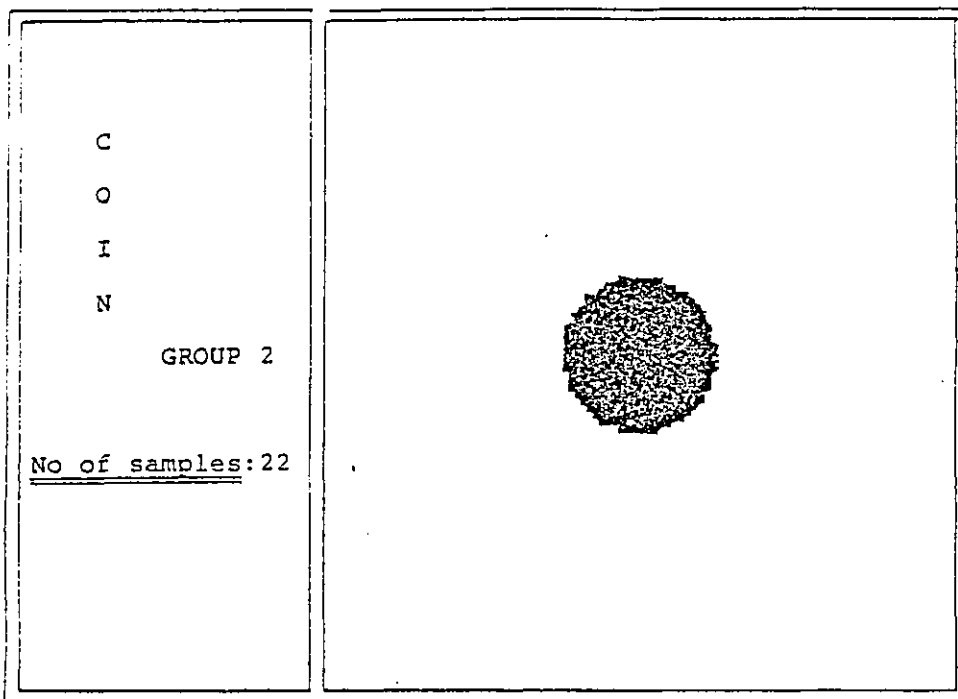
P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	118	6	5
AREA	926	53	6
MAXDIS	19.7	0.6	3
MINDIS	14	1	6
AVRDIS	17	1	9
PERH	44	10	23
AREAH	26	7	26

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.797	.046	6
B	.075	.004	6
C	.148	.085	6
D	.184	.064	35
E	.995	.077	8
F	.012	.002	21
G	.18	.05	28
H	.101	.015	15
I	.054	.013	25
J	.078	.035	46



P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	85	2	3
AREA	254	11	4
MAXDIS	18	1	4
MINDIS	2.7	0.4	14
AVRDIS	10.2	0.5	5
PERH	10	2	22
AREAH	8	3	37

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.439	.023	5
B	.021	8×10^{-4}	4
C	.744	.047	6
D	.739	.038	5
E	1.4	.8	5
F	4.2×10^{-3}	2.7×10^{-4}	6
G	.964	.185	19
H	.511	.092	18
I	.054	.012	22
J	.701	.07	10



P R I M I T I V E F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
PER	48	1	2
AREA	177	4	2
MAXDIS	8.7	0.4	5
MINDIS	6.3	0.5	9
AVRDIS	7.4	0.1	1
PERH	0	0	0
AREAH	0	0	0

C O M P O U N D F E A T U R E S			
	MEAN VALUE	ST. DEVIATION	% DEVIATION
A	.939	.031	3
B	.014	.003	2
C	.162	.040	24
D	.147	.064	43
E	1.026	.013	1
F	2.18×10^{-3}	4×10^{-5}	2
G	0	0	0
H	0	0	0
I	0	0	0
J	0	0	0

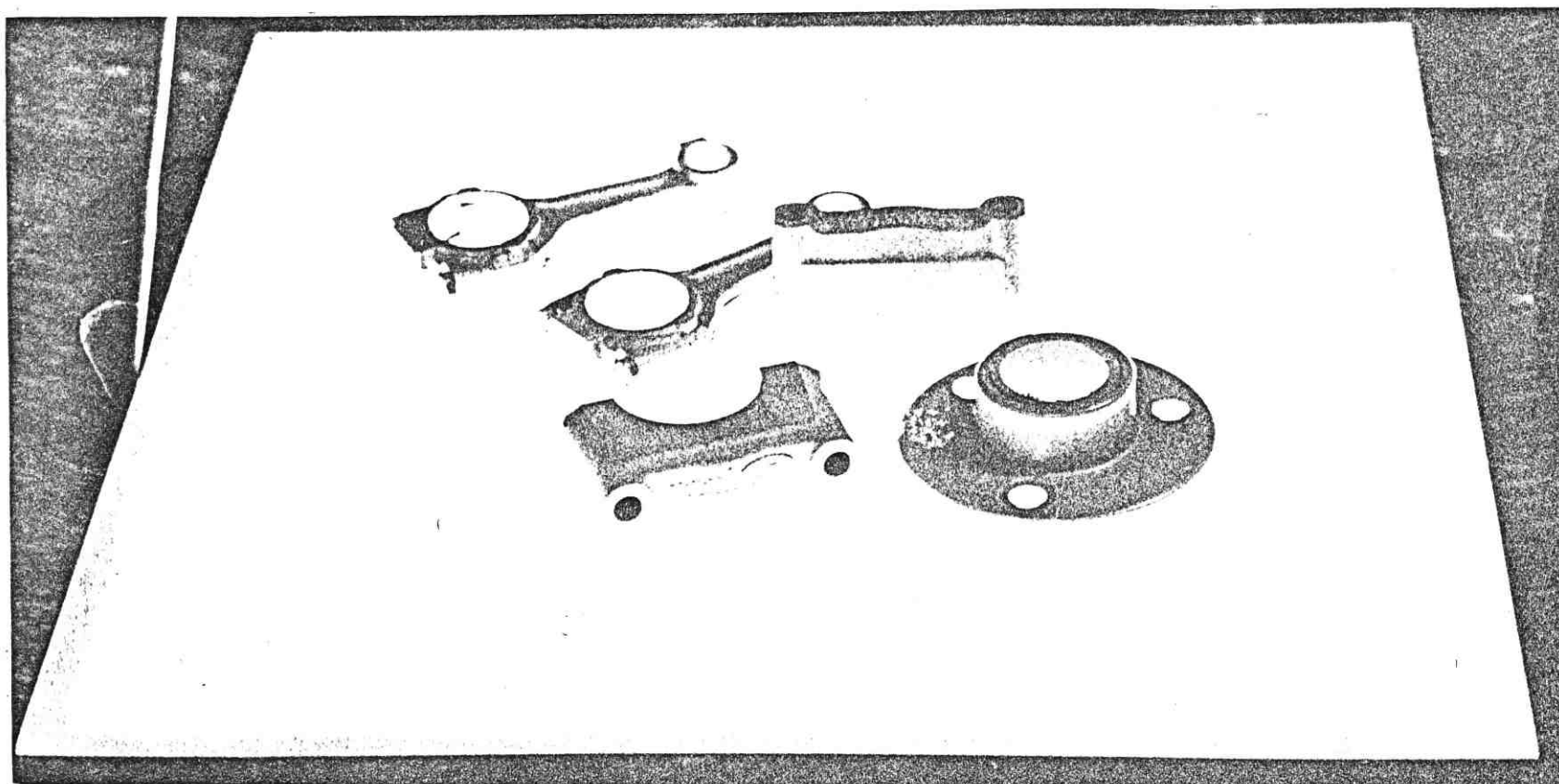


Fig. C1 : Typical scene - Transmitted light.

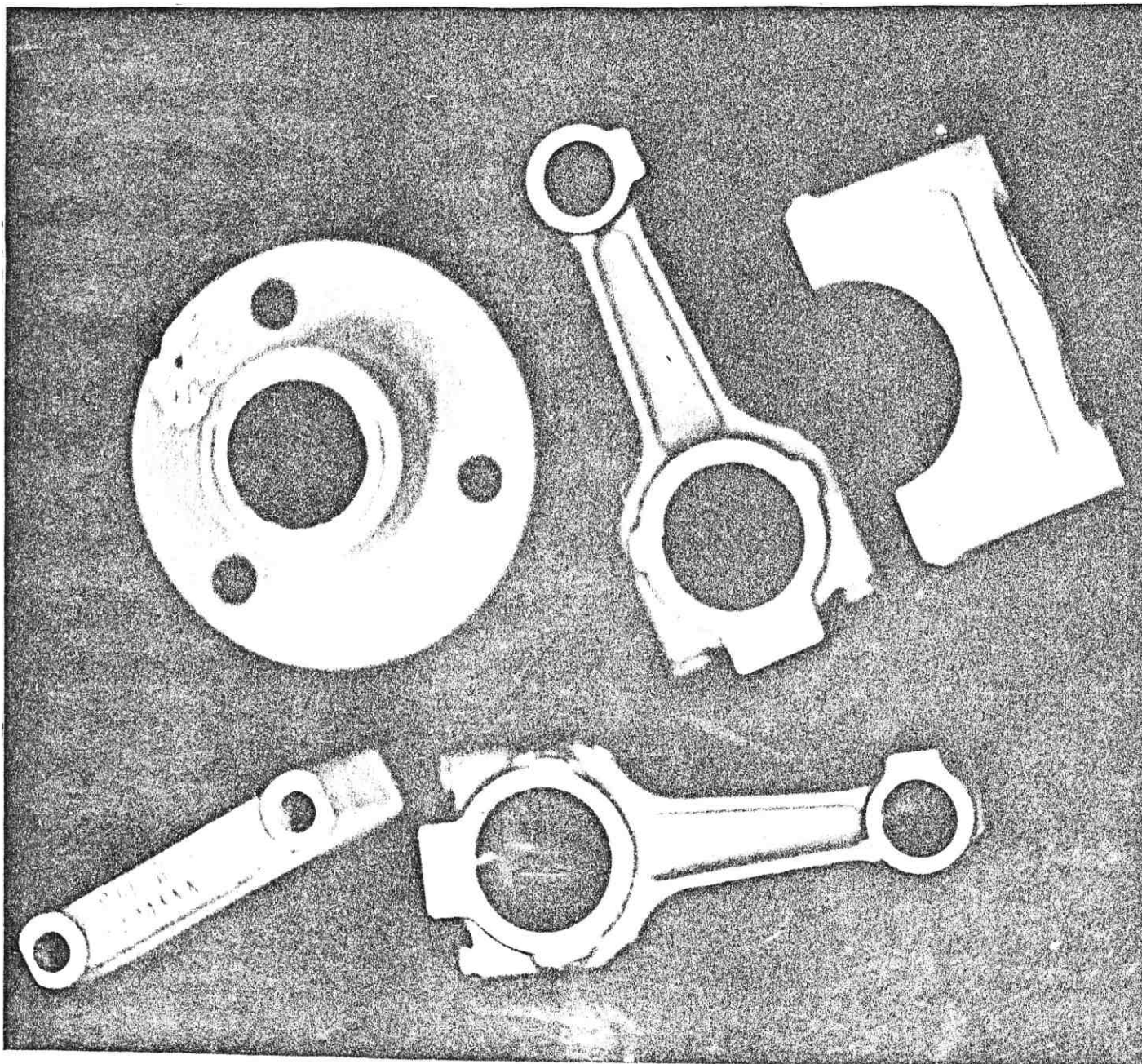


Fig. C2 : Typical scene - Top lighting.