COPYRIGHT AND CITATION CONSIDERATIONS FOR THIS THESIS/ DISSERTATION

- Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

- NonCommercial — You may not use the material for commercial purposes.

- ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

How to cite this thesis

THE USE OF FORMAL LANGUAGE THEORY IN COMPUTER VISION

by

GRAEME NEILL VAN NIEKERK

dissertation

submitted for the fulfillment of the requirements for the degree of master in natural sciences

in

COMPUTER SCIENCE

FACULTY OF SCIENCE

at

RAND AFRIKAANS UNIVERSITY

SUPERVISOR : PROF. E.M. EHLERS

APRIL 1991
ABSTRACT:

In this dissertation, a study of the field of computer vision as well as various fields relating to computer vision is made.

An investigation of organic vision is made involving the study of the organic focusing device and visual cortex in humans. This is also done from a psychological point-of-view.

Various network models emulating the neuronic networks as well as component networks of the human visual cortex are investigated. Recent work done in the area of neural networks and computer vision is also mentioned.

The mathematical theory and techniques used in the area of image formation and image processing, is studied.

The study of the field of artificial intelligence and its relation towards the computer vision problem, is made as well as a discussion of numerous application systems that have been developed.

Existing industrial applications of computer vision are studied as well as the mentioning of systems that have been developed for this purpose.

The use of parallel architectures and multiresolution systems for computer vision application, are investigated.

Finally, a discussion of the formal language theory and automata is given in terms of its relevance to computer vision. The discussion centers around the the recognition of two and three-dimensional structures by various automata in the two dimensions.

From this study, a formal model for the recognition of three-dimensional digital structures, is proposed and informally defined. It will be the aim of further study to fully develop and implement this model.
OPSOMMING:

In hierdie verhandeling word 'n formele studie van die veld van rekenaarvisie gemaak asook verskeie ander velde wat verwant is aan rekenaarvisie.

'n Studie van biologiese visie word gedoen. Dit behels 'n studie van die fokusapparatuur en visuele korteks asook die psigologie van die organiese visiesysteem van die mens.

Verskeie netwerkmodelle wat poog om die neuroniese netwerke van die korteks na te boots, word ondersoek asook onlangse werk wat gedoen is ten opsigte van die gebruik van neurale netwerke in rekenaarvisie.

Die wiskundige teorie en tegnieke wat gebruik word in die veld van beeldverwerking, word bestudeer.

'n Studie van die invloed wat kunsmatige intelligensie op die rekenaarvisie-probleem kan uitoefen, word ondersoek asook verskeie reeds ontwikkelde stelsels genoem.

Bestaande industriële toepassingstelsels vir rekenaarvisie, word geondersoek.

Die gebruik van parallelle verwerkingsargitekture en multi-resolusie stelsels in rekenaarvisie word ondersoek.

Ten slotte word 'n studie gedoen in die area van die formele tale-teorie met betrekking tot rekenaarvisie. Die studie begin by die een-dimensionele grammatikas en tale wat oor die algemeen in die studie van formele tale gedoen word. Daarna beweeg die studie oor na die formele automate wat funksioneer op 'n twee-dimensionele invoerband wat simbole bevat. Hierdie automate is spesifiek gedefinieer as formele modelle vir die herkenning van twee-dimensionele figure. Die studie gaan vandaar oor na die bestudering van drie-dimensionele automaatteorie, wat veral daarop toegespits is om as teoretiese modelle te dien vir die herkenning van drie-dimensionele digitale strukture.

Uit hierdie studie, word 'n formele model vir die herkenning van drie-dimensionele digitale strukture voorgestel wat sal uitloop op die volledige ontwikkeling en implementering van die model.
I WOULD LIKE TO THANK THE FOLLOWING:

THE Creator, without whom nothing will exist,

MY PARENTS for all their support and encouragement, through
the years,

PROFESSOR EHLERS for all her patience, support and encou­ragement,

MY FRIENDS who supported me tremendously,

THE FOUNDATION for research development for their financial
support.
CHAPTER 1

INTRODUCTION

DISCUSSION ................................................................. 1-1

1.1 PURPOSE OF STUDY .................................................... 1-1

1.2 OVERVIEW .............................................................. 1-2

CHAPTER 2

BIOLOGICAL (ORGANIC) VISION

2.1 THE AIM OF THIS CHAPTER ........................................... 2-1

2.2 INTRODUCTION .......................................................... 2-1

2.3 THE CREATION OF ORGANIC VISION ................................... 2-1

2.4 THE ORGANIC FOCUSING DEVICE ..................................... 2-2

2.5 THE STRUCTURE OF THE HUMAN VISUAL CORTEX .................... 2-7

2.6 COLOUR VISION .......................................................... 2-9

2.7 AN ALGORITHMIC APPROACH TO BIOLOGICAL VISION .......... 2-11

2.8 SUMMARY ............................................................... 2-15

CHAPTER 3

NETWORK MODELS FOR COMPUTER VISION

INTRODUCTION ............................................................... 3-1

3.1 KABRISKY’S MODEL ..................................................... 3-1

3.2 BARON’S MODEL .......................................................... 3-3

3.3 THE USE OF NEURAL NETWORKS AS A METHOD OF SIMULATING THE NEURON NETWORKS OF THE BRAIN AND THE SUBSEQUENT RELEVANCE OF NEURAL NETS TO COMPUTER VISION ........................................... 3-7

3.3 TAXONOMY OF SOME IMPORTANT NEURAL NET MODELS THAT HAVE BEEN DEVELOPED TO DATE .................. 3-9

3.4 SUMMARY ............................................................... 3-18
7.4 A BRIEF LOOK AT THE MAINSTREAM OF VISION HARDWARE IMPLEMENTATIONS OVER THE PAST FEW YEARS ........................................ 7-4

DIMENSIONAL INSPECTION OF NATURAL AND FABRICATED ITEMS ........................................ 7-6

ELECTRONIC CIRCUIT AND LAYOUT INSPECTION ........................................ 7-9

TABLE OF APPLICATION FOR AUTOMATED ASSEMBLY AND MANUFACTURE-VISION SYSTEMS .......... 7-12

TABLE OF APPLICATIONS FOR SYSTEMS FOR ELECTRONIC ASSEMBLY ........................................ 7-14

TABLE OF APPLICATIONS FOR BIN PICKING ........................................ 7-16

7.5. SUMMARY ........................................ 7-18

CHAPTER 8

PROCESSING ARCHITECTURES USED FOR COMPUTER VISION

8.1 INTRODUCTION AND HISTORICAL REVIEW ........................................ 8-1

8.2 MULTI-RESOLUTION OF IMAGES ........................................ 8-2

8.3 CLASSIFICATION OF PARALLEL ARCHITECTURES ACCORDING TO THE FLYNN TAXONOMY ........................................ 8-7

8.4 CLASSIFICATION OF SIMD-COMPUTERS ACCORDING TO FOUNTAIN'S TAXONOMY OF AUTONOMY: ........................................ 8-8

8.5 SUMMARY ........................................ 8-11

CHAPTER 9

GRAMMARS, LANGUAGES AND AUTOMATA AND COMPUTER VISION

INTRODUCTION ........................................ 9-1

9.1 TWO-DIMENSIONAL GRAMMARS, LANGUAGES AND AUTOMATA A BRIEF OVERVIEW ........................................ 9-1

9.1.1 MATRIX GRAMMARS ........................................ 9-2

9.1.2 THE LANGUAGE GENERATED BY A MATRIX GRAMMAR ........................................ 9-2

9.1.3 THE ACCEPTING DEVICES OF THE LANGUAGES GENERATED BY M ........................................ 9-2

9.1.4 ARRAY GRAMMARS ........................................ 9-3

9.1.5 LANGUAGES GENERATED BY ARRAY GRAMMARS ........................................ 9-3

9.1.6 THE DIFFERENT ACCEPTING DEVICES OF THE LANGUAGES DESCRIBED IN 9.1.5 ........................................ 9-4

9.2 THREE-DIMENSIONAL GRAMMARS AND AUTOMATA ........................................ 9-8

9.3 SUMMARY ........................................ 9-12

CHAPTER 10

CONTENTS
A MODEL FOR THE RECOGNITION OF THREE-DIMENSIONAL IMAGES

DISCUSSION ........................................................................................................ 10-1

10.1. INTRODUCTION ....................................................................................... 10-1

10.2 THEORETICAL BACKGROUND .................................................................. 10-1

10.2.1 PYRAMID CELLULAR ACCEPTORS ....................................................... 10-1

10.2.2 RANDOM CONTEXT STRUCTURE AUTOMATA .................................... 10-3

10.2.3 THREE-DIMENSIONAL CONTEXT DIGITAL STRUCTURE AUTOMATA
(3DCDS) ........................................................................................................ 10-5

10.3 SUMMARY ............................................................................................... 10-7

CHAPTER 11

CONCLUSIONS AND FUTURE RESEARCH

DISCUSSION ........................................................................................................ 11-1

REFERENCES
INTRODUCTION

DISCUSSION:

Ever since the development of the computer, it has been a dream to make it an intelligent entity, capable of acting like man or even surpassing his capabilities. The capability of man's sense organs, was developed over millions of years of evolutionary change. Man's capability of intelligent behaviour has at its heart, the cerebral cortex. Linked to the cerebral cortex, is man's sensing systems, enabling him to act in an intelligent way in the external world. This is so because these systems present the cerebral cortex with vast amounts of information concerning the external world. The cerebral cortex then decides what the relevant action will be, based on the information it has received from the sensing systems. If the computer of today or of tomorrow is to become more 'intelligent', in other words, having some of our intelligent capabilities, it will be necessary to augment the 'senses' through which the computer communicates with the outside world. During the early years of computer development, the only communication channel was through a deck of punched 'cards', submitted in batch mode as input for the computer. On these cards, the computer read the instructions, and acted accordingly. If a compiler error was detected, the whole deck of cards containing instructions, had to be corrected as a whole, and then submitted again as input. The output of such a program was also presented on these punched 'cards'. In later years, it became possible to program the computer interactively. This was due to the development of time-sharing systems. In recent years, input/output capacity was further increased when interactive input/output devices such as the light pen, data tablet and mouse, were introduced. There are two communication channels in humans, capable of carrying large amounts of information concerning the external world to the cerebral cortex. There it is processed, and an intelligent act is performed based on the decision which was made after the processing of the information. The two communication channels under consideration here, are that of sight and hearing. Vast amounts of information about the world around us, are received through our visual sensing devices. After processing of this information is done in the visual cortex, we act accordingly in an intelligent manner. Therefore, it is not surprising to find that a lot of research revolves around an attempt to emulate the capability of sight for the computer. Giving sight to a computer involves the study of many academic fields of interest. Examples include image processing, pattern recognition, artificial intelligence, robotics, cybernetics, computer science and neuroscience to name just a few. In all these fields, significant progress has been made over the years. In this dissertation-study, this progress is addressed.

In section 1.1 and 1.2, the purpose of this study as well as a brief overview, is given.

1.1 PURPOSE OF STUDY:

The study of computer vision involves various academic fields of interest. In figure 1, these fields are shown in their relation to the central problem of computer vision as well as their relations towards each other.
Computer Vision is defined in [103] as being the extraction of visual information of a scene by computer analysis of one or more images of that scene. Visual information may contain positional information of an object or objects in two or three dimensions, or it may contain information concerning the object or its motion, geometry and their relation to one another. Computer vision has numerous fields of application, which include remote sensing, robotic navigation, reconnaissance and industrial automation. The purpose of this study was to look at the various areas shown in figure 1 and study the interactiveness of the various fields to the computer vision problem. It was the intention from the very start to establish as broad a base as possible from which later research could be directed. In the end, a theoretical framework was established, for the development of an abstract theoretical model for the understanding of three dimensional digital structures. From this framework, the intention is to fully develop this model theoretically and then to implement it in a practical stage of development.

1.2 OVERVIEW:

In this section, a broad overview of the literature study undertaken, is given.

CHAPTER 2 deals with the study of the biological vision system of humans. Fields of interest that were investigated, include the nature and evolution of organic vision as well as its psychology. The physical structure of the human visual cortex is also given. It is generally agreed that if an understanding of the human vision system is reached, it will be that much easier to construct computer vision systems which will match the capabilities of
the human system. This is done under the assumption that the hardware technology will be available for this.

In **CHAPTER 3** various network models that have been developed to emulate the neuronic network of the human visual cortex, are discussed. Various neural networks that have been developed as well as their use in computer vision, is discussed. Neural networks have the ability of being 'trained' in the recognition and classification of patterns. Because of this ability, neural networks form the topic of extensive research at various universities [110].

**CHAPTER 4** investigates the mathematical aspects of image formation. The techniques of forming an image of a scene of the external world, are discussed.

**CHAPTER 5** discusses the various techniques available through which the formed image can be mathematically analyzed. The measurement of various properties of an object in an image, is discussed. These properties contain information on the position, shape, surface texture, brightness and so on of the object. This property-measurement is done in two categories: (1) Object-detection in an image and (2) Object extraction from the image and reproduction produced on another image-grid. In each category, different properties of the object can be measured.

**CHAPTER 6** discusses the field of artificial intelligence and its relation to computer vision. The case for a knowledge-based approach being used for the intelligent processing of a vision task, is presented. Some artificially intelligent systems that have been developed over the past few years, dealing with the problem of object-recognition in three-dimensional space, are discussed. These systems make use of knowledge as an aid in the task of object-recognition.

**CHAPTER 7** investigates the application of computer vision and image processing in the factory environment. Applications range from automated inspection and assembly to robotic navigation through an obstacle course. Extensive examples are given in order to present a broad outlook of the application possibilities of computer vision in the industrial environment.

**CHAPTER 8** investigates the various processing architectures being used in computer vision systems today. Because visual information processing is computationally expensive, the use of parallelism greatly decreases computational cost and increases the quality of visual information processing. The idea of representing an image at reduced resolution, is investigated because visual information processing is reduced in this way. The processing architecture used extensively for this purpose, is the Pyramid architecture.

In **CHAPTER 9**, the use of formal language theory for computer vision, is investigated. Various two and three-dimensional grammars generating languages which can be recognized by two and three-dimensional recognition devices, are given. From the work done by various authors in this field of language theory, a theoretical model for the recognition of three-dimensional digital structures, is proposed in the next chapter.

In **CHAPTER 10** a framework is presented by which a certain recognition device will be able to recognize three-dimensional digital structures. In this chapter, the integration of
The pyramid cellular acceptor discussed in chapter 9 and the Random Context Structure Automaton, is proposed.

The **AIM** of this study was to obtain information on the various fields relating to computer vision. The theoretical model proposed in **CHAPTER 10** must be fully developed and implemented in future study. The knowledge of the previous chapters will have to be used for this purpose.
CHAPTER 2

BIOLOGICAL (ORGANIC) VISION
the electromagnetic spectrum, and extracts an image of the scene from a particular viewpoint in space, but does not partition the scene into meaningful elements. The geometrical relationships and properties are however, retained in a fairly accessible form.

The important question of how the acquired information (acquired by the eye) is transformed into an interpretation of the environment, can not be answered by attempting to define vision in terms of the structure of a particular type of receptor like the organic eye.

2.3.2 The main organ of vision

The main organ of vision should perform the task of interpreting what is seen. The organ that performs this task is the human brain while the eye does the sensing. The memories of past visual experience and wired-in processing machinery may have a great influence on how the scene is interpreted [118].

2.3.3 The task of vision: an empirical definition:

From the above discussion, we can define the task of vision in organic matter as the process of transforming sensory information (gathered by the eye) into knowledge of shape, identity or configuration of the objects in the environment (the scene). The device responsible for this process is the (human) brain, more specifically, the visual cortex.

2.4 THE ORGANIC FOCUSING DEVICE

The purpose of this section is to establish the universal mechanisms devised by nature through evolution, that offer a solution to the problem of the visual understanding of the world. Furthermore, we will attempt to understand the architecture of organic visual systems by looking at the evolution and physiology of such systems.

2.4.1 The process of "Seeing"

"Seeing" may be defined as the physical recording of the pattern of light energy received from the surrounding world. It consists of the following three operations:

- Selective recording (gathering) of light emanating from the external visual world.
- Focusing this light on a light-sensitive surface.
- Converting light energy into a pattern of chemical change (or electrical activity) that is related in some specific way to the visual scene which was the source of the light.

2.4.2 Lens properties and abilities of the Organic Eye

In all the varieties of life, only two basic types of eyes have evolved and are commonly found [117]. The two types are:
• the single lens, camera-like eye, found in molluscs and chordates,

• the multi-lens compound eye found in arthropods (like insects and spiders).

The single lens of the camera eye forms a sharp retinal image of objects located almost anywhere in its field of vision. The compound eye does not produce a single coherent image, but dissects the image into a mosaic of image parts. This mosaic is very similar to the one produced by an image on the retina-cells of the camera eye.

The ability to focus requires a sophisticated control mechanism [117]. This mechanism must be able to identify something of interest in the prefocussed image, and move the lens to sharpen the boundary between the object and the background. To create this sharp image, a very finely partitioned retina, is needed. A computing capacity capable of dealing with this huge volume of data and making almost instantaneous decisions, is also required.

These are precisely the features found in our own organic visual system. The retina has approximately 130 million light-sensitive cells, some with a diameter of one or two micrometers. The computing facility is the brain with a capacity of 100 billion neurons working in tandem with the retina to complete a vision (or processing) task [119, 118, 117, 90].

Another structural property of organic image forming eyes is the ability to use shape and colour to classify objects. In a later section, the mechanism by which eyes detect color differences, will be discussed. What is important however, is that a retinal cell that is sensitive to a particular colour must ignore the light energy associated with other colors. The ability to see colour means a loss of sensitivity in dim light. Thus, an object may not be identified at all, because it did not provide enough colour contrast to be detected.

In the human eye, there are two distinct visual systems based on two types of photoreceptors, rod cells and cone cells. The cone cells extract colour information, and are used for detailed vision [117]. They are small and are most densely located near the centre of each retina. They communicate with the visual cortex through the ganglion cells in the retina. The rod cells are more sensitive to light than the cone cells. Their relative density is at its greatest in the peripheral regions of the retina, and they are unable to detect colour. They function in groups and feed the brain through a small number of ganglion cells. Their main purpose it would seem, is to detect anomalies (like movement) in the visual field and then to allow the cone cells to do the detailed analysis via focusing. As mentioned earlier, the cone cells cannot function in dim light. Hence, the rod cells must take over the task of shape detection in dim light. Figure 1 and 2 [117] depict the two cells.
THE HUMAN EYE:

The human eye is a remarkable instrument with respect to both sensitivity and resolution. Some of the notable characteristics of this organ are as follows [117],[116: (1), (3), (7)]:

- there are 120 million rod cells in each eye.
- there are 6 million cone cells in each eye.
- there are 1 million nerve fibres in the optic nerve exiting each eye.
- the diameter of cone cells in the fovea: 1 to 3 micrometers.
- there are 250 million receptor cells in the eyes.
- the distance from center of lens to fovea: 17mm.
- rod cells are 500 times more sensitive to light than cone cells.

Figure 3 [117] shows the anatomy and nervous organization of the human eye.
Anatomy Of The Human Eye.

Neural Organization Of The Retina.
2.4.3 Transformation of light energy into a model of the external world

How is the pattern of light energy which is projected onto the light-sensitive cells of the eye, transformed into a model of the external world? Even in simple organisms, we know little of their neural machinery and even less about the way the machinery actually functions. As we ascend the evolutionary scale, more of the processing is shifted from the neural networks in the photoreceptive eyes (cells) to the brain. The problem we now face is to find out how the information from the eyes is coded into the language of the brain (neural terms) and how it is then interpreted.

From [118], it was established that by inserting fine glass electrodes into nerve cells, researchers were able to stimulate cells to send a message and then to eavesdrop on the conversation. The brain's neurons communicate with one another through pulses of electricity which are conducted along the fibres that stretch away from each cell. These signals are passed on from one nerve fibre to the next by neurotransmitters. Neurotransmitters are chemical message-bearers, that travel across the gaps, the size of a sliver, which separate the fibers. The gaps are called synapses, and it is here where the critical link from one nerve fiber to another is made. Thus, it is within the pattern of signals sent across a multitude of synapses that a memory resides, as well as a pattern of visual information.

When light strikes the retina, the decomposition of pigments in the rods and cones results in electrical activity. This activity is integrated in the bipolar cells and ganglion cells which comprise the sixth and eighth layer of the ten-layered visual system of the retina. The ganglion cells of the eye feed the brain with visual information coded into electrical pulses. The other attributes of illumination such as colour, are determined by which cells are 'firing' the electrical pulses.

The human retina is effectively divided, vertically in half, and the nerve fibres from the left half of each retina send information about the right half of the visual field to the striate cortex in the left occipital lobe of the brain. Similarly, the right half of each retina sends information about the left half of the visual field to the right striate cortex.

The nerve fibres from the eye reaching the striate cortex, preserve the topology and much of the geometry of the imaged scene information. The striate cortex contains a specific region called the visual projection area. This area is in approximately one-to-one spatial correspondence with the retina. When these nerve cells are stimulated, the subject 'sees' elementary visual events such as coloured spots or flashes of light, in the expected location of the visual field.

Immediately surrounding the visual projection area is an area called the visual association area. Electrical stimulation in this area leads to complex recognizable visual hallucinations. Lesions of this part of the occipital cortex neither reduce visual acuity nor lead to loss of any portion of the visual field. Any defect of the human cortex leads to imperfect perception and reproduction of pictorial objects. The destruction of the visual cortex leads to complete loss of pattern vision. There is very little that can be said about the relationship between brain architecture and the performance of high level functions, but a little is known about how the elementary neural processing is done [117, 118]. There are a few processing 'tricks' that nature has discovered and that we find in almost all eyes of all species. One of these
is lateral inhibition for detection when something different or unusual has occurred in the visual field. It renders part of the lens the ability to concentrate on an important visual information area while the rest of the lens concentrates on the "blurry" visual area surrounding the important area being concentrated on by the one part of the lens. Figure 4 [117, 116 : (1)] shows the nerve pathways from the retina to the visual cortex of the brain. Nerve fibres from the right side of each retina pass to the right side of the brain, while fibres from the left side of each retina pass to the left side of the brain.

OPTIC NERVE ----

Nasal Visual field

Temporal visual field

FIGURE 4

Visual Cortex Of Occipital Lobes.

2.5 THE STRUCTURE OF THE HUMAN VISUAL CORTEX.

The human visul system can convert the normal flat images (overlapping) projected onto the retinas of its eyes into a three-dimensional model of the surrounding environment [117]. In this section the human visual system’s machine employed to perform the stereo-function, will be discussed. It must be pointed out, however, that the stereo function is not yet fully understood.

2.5.1 The ganglion cells

The ganglion cell has a receptive field consisting of an excitatory center and an inhibitory surround. Thus, each cell responds best to a roughly circular spot of light of a particular size in a particular part of the visual field. The path from the receptor cells in the retina to the cells in the visual cortex is indicated schematically in figure 5 [116 : (2), 117].
Visual Pathway From The Rod-cells To The Visual Cortex In The Brain.

2.5.2 The first transformation of visual information

It is the visual cortex that performs the transformation. The first transformation is the integration of information from the retinal ganglion cells so that the cortical cells respond to specifically oriented line segments rather than spots of light. Depending on the particular cell, its maximum response will either be triggered by a moving bright line on a dark background or by a boundary between light and dark regions. The line-orientation, speed and direction of motion are also important. There appears to be a hierarchy of cell types, with simpler cortical cells feeding the more complex cells. Neurons in the visual cortex with orientation specificity vary in their complexity. 'Simple' cells appear to obtain their inputs from a line of retinal cells and the far more numerous 'complex' cells behave as though they receive their input from a number of simple cells, all with the same receptive field orientation, but differing slightly in the exact location of their fields.
2.5.3 The second transformation of visual information

The second transformation performed by the visual cortex, is to combine inputs from the two eyes. The cells in the visual cortex that receive direct input from the retinas, are simple "monocular" cells and are stimulated by only one of the eyes (not both) meaning that the one set of monocular cells are stimulated by one eye and the other set by the other eye. About half of the complex cells are monocular, the rest are binocular (they can be influenced independently by both eyes). In such cells, the left and right receptive field inputs are generally identical in all respects, except that the stimulation ability of one eye typically dominates the other.

The visual cortex is subdivided into roughly parallel columns of tissue, approximately perpendicular to the surface of the cortex. Figure 6 [116: (1), 117], gives a schematic description of the elementary unit of the visual cortex. Each column is divided into 50 micrometer-thick slabs and has ten degree shifts in its line orientation. Slabs are arranged in coherent blocks with each block containing a right eye dominant column, and a left eye dominant column. Blocks near the centre of the gaze have small receptive fields, blocks whose receptive fields have to adapt to higher fields of eccentricity, have larger receptive fields.

![Diagram of the visual cortex](image)

**FIGURE 6**

An Elementary Unit Of The Visual Cortex.

2.6 COLOUR VISION.

Given the appropriate environmental conditions, colour enhances an organism's ability to identify visible objects, to determine their physical properties and can play an important role in visual communication. How does the eye measure the spectral attributes of light energy impinging on it, how is this information represented internally and how is it transformed into the subjective impression we call colour?
2.6.1 The Young-Helmholtz theory

The Y-H theory asserts that there are three colour-sensitive types of receptors in the eye which correspond to red, green and blue [117] respectively. Furthermore, all color perception is the result of the relative strength of the signals received from these three receptor systems.

This theory is valid for simple patches of light. However, it is not sufficient to explain human colour perception in complex natural scenes. Land demonstrated that our final perception of colour at any point in a scene is dependent on colours perceived in other parts of the scene, and that we can perceive colors in complex scenes that cannot be exactly reproduced by a simple mixture of the three primary colors. Land showed that red and white light could induce the human visual system to perceive a coloured scene [116: (3)].

Another problem with the Y-H Theory is that the "blue" pigmented cones are not spatially distributed in the same manner as the "red" and "green" cones. They are not as common as the latter and are actually absent from the central fovea [117]. Thus, the above results show that color perception involves processes occurring at higher levels in the brain. We cannot explain colour vision strictly in terms of local stimulation of retinal nerve cells. Nevertheless, we may still profit by understanding the local sensory apparatus the eye employs to detect and encode the spectral attributes of incidental light.

The blue cones seem to be distinguished from the red and green cones in many other important respects. They do not contribute to the perception of boundaries between differently colored regions and make little, if any, contribution to the total luminance signal.

Only the cone cells are directly involved in colour vision. The retinal image consists of a pattern of light energy. This image is transformed into a pattern of nerve activity by the presence of photosensitive pigments in the rods and cones that absorb part of the incident light energy. All cone cells appear to be anatomically alike.

Absorption Of Light Energy By Three Populations Of Cones In The Eye [116 (3), 117].
2.7 AN ALGORITHMATIC APPROACH TO BIOLOGICAL VISION.

This section explores the nature of the algorithmic techniques employed by organic visual systems through an examination of their successes and failures in interpreting both natural and contrived images.

2.7.1 The perception of the visual (natural) world

Perception is not the result of a set of stimulus patterns, but an optimal interpretation of sensory data based on past experience of both the organism and its ancestors [117, 118]. The senses provide evidence for checking hypotheses about the nature of our environment. The best way to describe our visual system is to say that it appears to operate flawlessly, spontaneously and without surprises.

At the highest levels of performance, shape recognition involves the ability to ignore variations in size, brightness, position and orientation. This is also called Constancy [116]. Many species can recognize a shape by its outline, while others have great difficulty in recognizing the outline if they have been trained to recognize the solid shape. Humans have great difficulty recognizing faces presented upside down. The reason for this may be that while the visual system can generally deconstruct the visual field into meaningful sub-units, it appears that our visual system cannot extract and manipulate portions of an image (like a face) which is tilted at 180°. This tends one to conclude that we are able to recognize objects whether they are different in size or rotation. However, our visual system does not disregard rotational changes that was made to human faces. The visual system finds it difficult to recognize human faces tilted by 180°. It seems that our pattern recognition cannot distinguish the features of a human face when tilted by 180°.

Pattern vision is not a single integrated function in any organism. Biologically important visual tasks are handled by special mechanisms. For example, it appears that our brain has distinct procedures for processing visual information about faces—a task of great biological importance [117].

2.7.2 The organization of the perceptual process

We do not see objects in the same configuration under the same lighting conditions from the same perspective in space again. How can we then partition a scene without knowing what might be present? If one was merely searching for a few well-known objects, we could determine whether each of the objects was present at each possible location in the visual field. However, there are thousands of objects that can appear in an almost infinite variety of configurations that we can recognize, thus, exhaustive matching against stored models is not a reasonable explanation of human perception.

For this reason, then, it is largely agreed that there must be a set of generic criteria that can be applied independently of the scene content. They must be able to underly the procedures discovered by nature for partitioning the visual field.

Psychologists have attempted to discover the laws underlying the decisions made by the human visual system. Wertheumer in 1923 and Koffka in 1935 [116] proposed the famous gestalt laws which include:
1) The Law of Proximity:
Stimulus Elements close together tend to be perceived as a group.

2) The Law of Similarity:
Similar stimuli tend to be grouped.

3) The Law of Closure:
Stimuli tend to be grouped into complete figures.

4) The Law of Good Continuation:
Stimuli tend to be grouped as to minimize change or discontinuity.

5) The Law of Symmetry:
Regions bounded by symmetrical borders tend to be perceived as coherent figures.

6) The Law of Simplicity:
Ambiguous stimuli tend to be revalued in favour of the simplest alternative.

7) The Law of Common Fate:
If a group of dots were moving with uniform velocity through a field of similar stationary dots, the moving dots would be perceived as a coherent group.

The problem these laws of perception leave unanswered is that we do not know their purpose. They explain some kind of underlying psychology in vision which has not yet been determined. It is not even clear whether they are a complete set of laws governing this psychology; for example, completeness (explaining "all" data), stability (consistency of explanation of "all" data), and limited complexity (economy of explanation of "all" data, is low). These criteria are merely criteria which can be used for the acceptance of visual data and the Gestalt Laws seem to follow these "rules" or criteria for visual data acceptance.

2.7.3 The Illusion

How can we discover the nature of the algorithms employed by organic visual systems? In [116 (1), 116 (2)], the authors did an analysis of what happens when light strikes the retina and eventually reaches the visual cortex. It was learned that particular stimulations turn neurons on or off and groups of neurons perform particular transformations in the visual cortex. It is unknown how visual information is exploited in later stages.

Visual Illusions have provided much of the information on which theories of the functioning of visual systems are based [116: (4), 116: (5), 117]. The term illusion refers to those situations in which our perceptions differ markedly from what we know corresponds to the actual physical situation.
The most remarkable aspect of human vision is how accurate it is, in spite of its distorted inputs. Our eyes are far from being a perfect focusing instrument, for they distort the image that they project onto the retina, because its finite aperture, out-of-bound lens and cornea. Thus, it is clear that there are a number of independent problems with which the system must deal. It would therefore be logical to infer that illusions occur due to a failure of one or more of the problem-solving mechanisms (for dealing with distortion etc.), for a single mechanism cannot underly all illusions. The visual system is able to correct the distortions due to a limited imaging system.

Illusions are produced when we are presented with an impoverished visual environment that misinforms a single functional system.

**EXAMPLES:**

**The Ponzo Illusion:**

Apparent depth alters size perception [117, 116: (4)]

**The Necker Cube:** [117, 104, 116 : (8)].

This example gives an indication of a reasoning process carried out by the perceptual system: We perceive the cube either as face-in-front or view-from-above. The process "iterates" until one of the perception possibilities remain and is accepted.
Other illusion-examples [116: (4)].

**Poggendorf:**

Oblique lines are colinear.

**Müller-Lyer:**

The length of the horizontal lines is the same.

**Upside-Down T:**

The two lines have the same length.

**Others include:** The Judd, the Titchener circle, the Delbouf circle, the Lipps figure, Zöllner figure and so on.

It has been suggested in [116: (4)] that most illusions mentioned and drawn above, consist of two underlying phases namely:

- the phase that **distorts** the input and the phase that **induces** the distortion.
- a 'test' component receives the distorted input.
In the Müller-Lyer illusion, the arrow-heads are the inducing part and the horizontal lines are the test component.

2.7.4 Visual Memory

It is known that a list of objects can be memorised if they are set in the context of a visual scene. The more vivid the scene, the better the objects are remembered. The implication is then, that we employ distinct mechanisms for both symbolic and iconic information, and that our storage and recall ability for iconic information is significantly better than that available for symbolic information [117, 118].

2.7.5 Visual Thinking

There are those who believe that all thinking are perceptual in nature [118]. Some feel that visual thinking is a skill that can be learned and that it improves with practice.

2.8 SUMMARY

From this investigation into the biological vision system of man, it is clear that there is a substantial amount of knowledge about the visual system that is still to be discovered. The visual path from the visual cortex is well known and studied. This is also the case for the the operations of the visual cortex. Through experiments [116], knowledge was gained as to what cortical cells was responsible for operations such as early visual processing and association. The locations of these cells in the visual cortex were also established. What is still unknown is how these operations are performed by these cells. Research projects suggest [118] that the neuronic language spoken by the neurons in the visual cortex as well as the rest of the cortex, must be decoded. This language is electrically coded. If this venture succeeds, a type of neuronic language can be established. This will greatly enhance our understanding of what algorithmic process or processes are controlling the visual process. The Gestalt laws appear to be perception laws. However, they fail to explain the phenomenon of the illusion. The question arises whether or not they are part of a larger set of perception laws.
DIAGRAM OF THE VISUAL PATHWAY

Figure 8

Visual association area
geometrical and topology
Projection area
Simple cortical cells (monocular)
Receive input from ganglion cells (Responds to line segments)

(Binocular) Cortical cells (Complex)
Monocular cortical cells (Complex)

Optic chiasma

Retina

Optic chiasma

(coded electrical activity fed to brain)
ganglion cells
Bipolar cells
rods and cones
CHAPTER 3

NETWORK MODELS FOR COMPUTER VISION
INTRODUCTION

The aim of this chapter is to look briefly at various models for the processing of visual information. For this purpose, it was decided to examine the models of Kabrisky and Baron[90]. We also decided to look at the work of Minsky and Papert[89] in which they defined the idea of a linear device called a perceptron in 1969. The limitations of perceptrons being a single layer network will also be discussed. The models of Kabrisky and Baron, will be examined from a high level point of view, which will provide an opportunity to discuss these network problems in a general way. Because perceptrons are single layer networks and because the theory of perceptrons, will be examined, the discussion will then move to the Multi Layer Network. The possibilities of a MLN in terms of machine training, will then be examined. In this regard, the back propagation algorithm developed by various authors like Rumelhart, Hinton and Williams, are discussed.

Thus the aim of this chapter is to present high level network models (Baron and Kabrisky) for visual information processing, as well as a more mathematical approach of modelling through the SLN of perceptions and MLN (for example the back-propagation approach).

3.1 KABRISKY’S MODEL

Kabrisky's model was designed to simulate the brain's visual cortex. The visual cortex is also known as the primary cortex or striate cortex [116, 90]. The primary (visual) cortex is also called VISUAL I.

Kabrisky’s model is a neuronic circuit comprising millions of neurons in VISUAL I. The components of the model are column shaped cortical areas, 0.5 mm in diameter. This component unit consists of about 300 neurons. Kabrisky decided on a column shaped configuration because the actual neuronic architecture corresponds to this shape. The most basic component in Kabrisky's neuronic circuit network is the basic computational unit (BCU). The BCU's input and output are both numerical quantities. The model can then be viewed as a discrete time cellular automaton. Each BCU communicates with its four nearest neighbours.
In figure 2, the output result of a BCU \( Q \) is shown. The state-transitions are also shown in figure 2.

A BCU produces an output \( Q \) at a time \( t \). \( Q \) is determined by a function \( F \) of its input \( P \) and two memory parameters SS and SF. SS is a function \( G \) of recently stored values of SF. SF in itself is a function of past values of itself, SS and \( P \). Mathematically, we state the above as follows:

\[
Q(t) = F(P(t), SF(t), SS(t))
\]

where

\[
SS(t) = G(SF(t), SF(t-1), SF(t-2), ...)
\]

and where

\[
SF(t+1) = H(SF(t), SF(t-1), ..., SS(t), SS(t-1), ..., P(t), P(t-1), ...)
\]

Thus, the value of SF is partially determined by the values of \( P \), SS and SF from the four neighboring BCU’s.

The significance of this model is that it can readily classify groups of patterns along predefined resemblances. A pattern is considered recognized if the total arrangements of a BCU’s \( Q \) outputs after a combinatorial summation, stabilize within a certain time-limit.
The model could also verify the positions of objects in space. It could also determine rotations and changes in the size of objects. Distortion was partially recognizable in patterns.

Certain limitations concerning hardware restricted the simulation to $10 \times 10$ array of BCU's in one plane. The software was written in *FORTRAN II* and the program comprised 120 statements. Even with these restrictions, the implementation was able to perform the above mentioned functions.

The model has a certain merit in the Computer Vision problem because it is said to have shed some light on some of the operations of which the visual cortex is able to perform. This includes the ability to recognize objects in space in spite of changes in size and rotation. It is however, quite obvious that this network model is far from equalling the human vision system, but was an attempt to observe through simulation, some functions that the visual cortex performs daily.

### 3.2 BARON'S MODEL

*BARON* [90] designed a network model for visual information processing that could perform visual processes like search, select, correlation, recognition, store and recall. Control networks in the models regulate the flow of information. This section will also give a brief high-level discussion of Baron's contribution.

#### 3.2.1 The Mathematical Neuron

Baron defined a mathematical neuron as a composition of four three-dimensional units or components:

(a) a cell body

(b) a dendrite field

(c) an axon

(d) an axon-field

#### 3.2.2 The Memory Block

Baron defines a memory block as a mutually linked group of neurons (mathematical neurons in 3.2.1) embedded together onto a background medium. The block is divided into several identical memory cells. Each cell contains one type of neuron. The following types of neurons can be found in each memory cell:

(a) Input neurons. These neurons receive weighted inputs to the cell from the external world.

(b) Memory neurons. These neurons constitute the most recently stored pattern (as output).
(c) **Output neurons.** Input and output neurons both carry received and transmitted patterns of informations to and from the source of the pattern (signal). In groups, information is sent to, and received, from a group block.

(d) **Recall neurons.** These neurons are able to recall patterns which are already stored by memory-effector neurons.

(e) **Memory-effector neurons.** These neurons are responsible for pattern storage.

(f) **Recognition neurons.** These neurons match incoming input patterns with stored patterns.

In order to formulate Baron's model, two fundamental assumptions must be formulated

**Assumption 1:**

The activities and organization (configuration) of individual visual neurons are ignored because they are not relevant in explaining the total behavior of the system.

**Assumption 2:**

The way in which external networks affect the function and operations of the primary network, is ignored.

### 3.2.3 The Model

The model consists of various primary networks working interactively with one another in a recognition task. The primary networks with their various component networks are listed below.

(A) **Information Processing Networks.**

(1) The Retina

(2) Primary Visual Selection-Networks

(3) Permanent Visual Storage-Network (Memory)

(B) **Control Networks.**

(1) The Selection Control Network

(2) The Memory Reaction Network

(3) The Memory Co-ordinating Network

In figure 3, the **WAY** in which control and information flow occurs in the network model is shown.
As can be seen from figure 3, the organic base for the network model is to be found in the human retina, visual pathways to the visual cortex (VISUAL I) and additional associated cortical areas. Each component of the model will now be discussed in order to describe the model in a functional way:

(A1) **The Retina.**

The Retina receives the input pattern from the external world. The input signal is usually a time-varying light pattern. The retina also receives information indirectly from the selection control network (B1). Neural Networks in the ‘retina’ transform the input pattern. The output pattern of the retina is the result of a set of activated processors (output). (These processors can be interpreted as retinal processors).

(A2) **The Visual Selection Network.**

Processing of the retinal output is done here. The result of this processing is then sent to (A3).

(A3) **Permanent Visual Storage Network.**

This is a memory block with memory reaction neurons. These neurons are controlled by inputs from (B2). Patterns are permanently stored in (A3).

(B1) **Selection Control Network.**

Information about the already stored patterns is available for the visual control mechanisms. Some of the tasks that (B2) can perform are for example to choose certain
areas in the visual field for further analysis. The chosen field can be enlarged, rotated and reflected.

(B2) **Memory Reaction System.**

This component directs patterns that must be stored in their selective memory locations in (A3).

(B3) **The Memory Recognition System.**

This network component receives correlation-information from recognition neurons situated in the memory storage network (A3). This component functions with a pre-set threshold: if the threshold-value is smaller than the incoming correlation-information, then the recall-neurons are instructed to produce the relevant output-information. This output information contains both the control and pattern information sent to B1. From B1 it is sent to other areas.

3.2.4 **What does the model do?**

Four vision processing steps are performed by the network:

**Step (1):**

A general **undirected** scan of the pattern is made. Areas of the visual field containing motion, smaller detail, brightness and contrast are selected for further analysis. When an object is recognized in the area in view, the system goes to (2). (2) will terminate when a proper memory map has been constructed. This map will contain all objects in the relevant area in view. Any object in this area, will be recognizable.

**Step (2):**

The selection control networks will continue to transform the signal until the memory storage network sends out a strong recognition signal.

**Step (3):**

If an object is required to precisely match an object in memory, then all transformations must be tested for equality.

**Step (4):**

If a specific object’s position must be located, steps (1) - (3) must be used intensively, together with verbal processing networks. These networks are not discussed here.

3.2.5 **Results obtained by the network:**

(1) Specific areas chosen by the selection control networks, were dependent on the memory positions chosen by these networks.
(2) Areas that were scanned for fine detailed information, produced images of a very high quality as opposed to areas that were briefly scanned, producing images with partial information on that area or areas.

(3) In the search for object-matching, the system always made the correct identification. This also applied to distorted image (distorted because of noise).

3.2.6 A summary of Baron's model

Simulations of this model were done in FORTRAN and implemented on a regular sized computer. The input to the Retina was encoded as different brightness levels that were integrated on a 60 * 60 grid. Contrast enhancement was done using lateral inhibition as preprocessing. The input pattern was transformed into a pattern of contours. The intensity levels were thus reduced [117]. This transformed pattern as well as the original were made available to the selection networks where they were further processed in terms of the reduction in size and translation. This produced a set of 10 selected square regions of varying size.

3.3 THE USE OF NEURAL NETWORKS AS A METHOD OF SIMULATING THE NEURONS OF THE BRAIN AND THE SUBSEQUENT RELEVANCE OF NEURAL NETWORKS TO COMPUTER VISION

INTRODUCTION:

In recent years, much research has been done in the area of networks that can simulate the neuron (biological) networks in the human brain [110, 108, 107, 106, 120]. The human brain is made up of billions of neurons all of which are connected in some kind of network. This network of neurons is considered to be one massive parallel processing architecture. Operations like visual information processing and speech recognition seems to be done in parallel.

The aim of this section is to discuss briefly the various neuronic network models that have been developed to date to simulate the network of biological neurons in the brain. These models attempt to achieve good performance by densely connecting simple computational elements. The best systems are far from matching human performance in areas like speech recognition and image processing. However it is in these areas that neural networks have the greatest potential [107, 108]. Neural network models explore many competing hypotheses simultaneously, using massive parallel nets composed of many computational elements connected by links with variable weights. The computational elements are typically non-linear. The simplest node sums N weighted inputs and the result is passed through a non-linearity as in figure 4.
The simplest node is characterized by an offset or threshold $\Theta$ which is internally set. $Y$ is transmitted as output if $Y > \Theta$.

There are a number of different net models which will be discussed briefly in this section. Neural net models are classified according to:

- the network topology used
- the node characteristics
- the training/learning rules of the model.

These rules specify an initial input weight-set and then control the way in which weights are adapted during computational use to improve performance. Each node or computational element, allows neural net models to be far more robust and fault tolerant than traditional Von Neumann computers [110]. This is, because there are more processing nodes, each with its own local connections. Damage to a few nodes need not impair the overall performance of the system. Adaptation or learning is currently a major area of research in neural net models [110, 120, 106, 107, 108]. This ability to adapt and to learn is crucial for operational areas like image processing and speech recognition.
3.3 TAXONOMY OF SOME IMPORTANT NEURAL NET MODELS THAT HAVE BEEN DEVELOPED TO DATE

3.3.1 Nets accepting binary values as input

In figure 5 [110], a taxonomy of six important neural nets that can be used for classifiers, are presented.

**NEURAL NET CLASSIFIERS FOR FIXED PATTERNS:**

- **BINARY INPUT**
  - Supervised: Hopefield net
  - Unsupervised: Hamming net

- **CONTINUES-VALUED INPUT**
  - Supervised: Carpenter-Grossberg classifier, Gaussian classifier

**FIGURE 5**

Neural Net Classifiers For Fixed Patterns
3.3.1.1 The Hopfield Net

As can be seen from figure 5 [110], the Hopfield net is used with binary inputs only. These nets are less successful when used with input values that are continuous.

The net has N-nodes receiving binary input and output which can either be the value of -1 or +1. Each node's output is then fed to all other via weights denoted $w_{ij}$. The first step in the operational process of the net is the setting of the weights according to the given recipe from example patterns. The next step is to present an unknown pattern to the net at time $t = 0$. After activating the net, the net iterates in discrete time steps (intervals) according to the given weight-setting of step 1. When outputs no longer charge after successive iterations, the net is considered to have converged. This uncharged output after convergence is the net output.

Hopfield's net has two major limitations [110]: The first is that the number of patterns that can be stored and recalled is very small. If too many patterns are stored (when the net is used as a content addressable memory), the net may converge to an undefined pattern not present in the example patterns. If the net is used as a classifier, it will obviously produce a 'no match' as output.

The second limitation is that an exemplar (input) pattern is unstable if the pattern shares common bits with another pattern. A pattern is unstable if it is applied at time $t = 0$ and the net converges to some other pattern.

The primary applications of the Hopfield-net is the retrieval of complete data/images from fragments. From the operational point of view and considering the limitations mentioned above, it is clear that weights are set prior to the presentation of the input and that the output of the net is produced according to this weight-setting. Therefore, the net cannot learn.

3.3.1.2 The Hamming Net

In the previous example of a neuronic network model, an input exemplar pattern is presented to the net after the weight-settings have been done. Bit-values are then reshifted randomly and independently with a given probability [110]. Thus the signals that are sent through the memoryless binary symmetry channel are fixed. If the input pattern differs slightly from the predefined weight-setting pattern, the Hopfield net will not be able to classify the presented input pattern accurately.

It is against this background then that the concept of the Hamming distance was introduced. In the Hamming Network model weights and thresholds are first set in a lower net (sub) such that the matching scores produced by the nodes in the middle are equal to N - the Hamming Distances to the exemplar patterns. The Hamming distance is the number of bits in the input which differ from the corresponding exemplar-bits. The network which implement this algorithmic solution to the bit-matching problem, is called a Hamming Network.

In figure 6 the outline of the Hopfield Net and Hamming Net are presented.
The weights and thresholds are thus set according to the condition of the matching scores of the middle nodes.

After setting the weights and thresholds, the net is presented with a binary pattern with \(N\) elements at the bottom of the net. The middle nodes form a sub-net called MAXNET. Thresholds and weights in MAXNET are fixed and the the lower sub-net calculates the matching scores as outputs. The upper net (MAXNET) then selects the node with the maximum output. The node selected is that node whose calculated Hamming Distance is the smallest. The input of that one node is then removed until only one output score of the last node in the lower net is positive and presented to MAXNET. In this time, MAXNET has iterated, removing maximum output scores from the lower nodes.

The advantage of the Hamming net, is that it implements an optimum minimum error classifier when bit values are random and independent. It also requires fewer connections than the Hopfield net. For \(N\) inputs and \(M\) classes, the Hopfield net requires \(N^2\) and the Hamming net only \(N \times M\) connections \([106]\). Thus the number of connections in the Hamming net grows in a linear way \((N \times M)\).

An obvious limitation is that it also cannot learn because weights and thresholds are present before an input pattern is presented. The output is then dependent on this predefined settings of weights and thresholds.
3.3.1.3 The Carpenter/Grossberg Classifier

The next classifier that accepts binary input, ill now be considered. The Carpenter/Grossberg Classifier differs from the Hamming and Hopfield net models in that it can be trained without supervision. The net implements a clustering algorithm. The leader algorithm selects the first input as the exemplar for the first cluster. Then the next input is compared to the first cluster exemplar and it "follows the leader". It is clustered with the first exemplar if the distance from the first is less than a certain threshold. Otherwise the second cluster is considered to be the exemplar for the next (new) cluster. The process is repeated for all inputs. The amount of clusters grow in relation to time and depend on the metric distance used between clusters and the threshold. Major components of the Carpenter-Grossberg classifier: three input nodes and two output nodes. In figure 7 the graphical representation of this net is given.

![Diagram of the Carpenter/Grossberg Classifier](figure7)

**FIGURE 7**

The Carpenter/Grossberg Classifier

The structure is similar to the Hamming net, but differs however from the Hamming net in that feedback-connections are provided from the output nodes to the input nodes. The Hamming net does not have this feature.

The operational process of the net begins by initializing the setting of all examplers represented by connection weights to zero (0). The matching threshold or vigilance is preset. This threshold varies between 0.0 and 1. As already mentioned, this threshold determines the degree of similarity between a new input-pattern and a stored exemplar. A value near one constitutes a very close match while values near 0 constitute a poor match. As in the Hamming Net, the inputs are presented sequentially at the bottom of the net. Thereafter, the input is compared to all stored examplers in parallel to produce matching scores. Using lateral inhibition, the exampler with the highest score is then selected, after which it is compared to the input by calculating the ratio of the dot-product of the input...
and the best matching example divided by the number 1 bits in the input. If the ratio is greater than the vigilance threshold, the input is similar to the best matching example. This example is then updated to its newer version by a logical AND operation between its bits and those in the input. Otherwise, if the ratio is less than the threshold, the input is different to all examplars and it is added as a new example. The effect is that each new example requires a node; 2N connections are needed to compute all the matching scores. Results [110] have shown that the C/G classifier-algorithm performs flawlessly with perfect input patterns but that the slightest distortions due to noise can cause problems in the classifying of an input-pattern correctly. Modifications are needed to enhance the performance of this algorithm when noise occurs. The weights can be adapted more slowly and change the vigilance threshold dynamically during training.

In this section, neural net models that accept binary values as their input and produce as output the required pattern they 'think' they were, have been discussed. Two of the models had no training capabilities while the last one did. However, the weakness of the last model in this specific classification taxonomy of neural net models is that it does not function successfully in the presence of noise.

3.3.2 Neuron network models that accept continuous-valued inputs.

3.3.2.1 The Single Layer Perceptron

The single layer perceptron [106, 89, 120, 110] can accept both binary values and continuous values as inputs. A perceptron decides whether an input belongs to one of two classes (say A or B). The single node computes a weighted sum of the input elements \( x_0, \ldots, x_{n-1} \), subtracts a threshold value and the result is passed through a hardlimiting nonlinearity such that the output is either +1 or -1. In figure 8, a graphical representation [110] of the single layer perceptron is given.

\[
Y = f_h \left( \sum_{i=0}^{n-1} W_i x_i - \Theta \right)
\]

\[
Y = \begin{cases} 
+1 & \text{Class A} \\
-1 & \text{Class B} 
\end{cases}
\]

**FIGURE 8**

The Single - Layer Perceptron

Usually a decision rule is designed which demands a response to A if \( y = +1 \) or to B if \( y = -1 \). To analyze the behavior of the perceptron net is to plot a map of the decision
regions which was created by the input variables $x_0, ..., x_{n-1}$, in multi-dimensional space. The regions specify which input values belong to class A and which to class B. A hyperplane boundary separates the regions A and B. In figure 8 the hyperplane, in this case, a decision boundary line determined by n-connection weights and threshold. The operational process of the single layer perceptron is as follows:

- Initialize the connection weights and threshold to small values (non-zero).
- Present the input as N continuous valued inputs.
- Compute $y$ (the output).
- Adapt connection weights if an error occurs.
- The output decision is according to $y$'s value after output. The decision of where the output belongs is made according to the decision rule.

It was proved by Minsky and Papert in their book on Perceptrons [89], that if the inputs from the two classes are separable, the perceptron convergence procedure converges and constructs the hyperplane exactly between the two classes. The perceptron convergence procedure fails when classes cannot be separated by a hyperplane. The Single Layer Perceptron fails to solve the XOR - problem, parity or connectivity problems [106, 120, 89]. Figure 9 shows these problems which cannot be solved.

![Decision region for class A and Separation of meshed regions](image)

**FIGURE 9**

### 3.3.2.2 Multi-Layer Perceptron

These perceptrons are feed forward nets with one or more hidden layers of nodes. The hidden node layers are not directly connected to input or output nodes.

A single-layer perceptron forms halfplane decision regions. A multi-layer perceptron (for example a two-layer perceptron) can form any convex region in space. A three-layer perceptron can form complex decision regions and can separate mesh-classes [89, 106].
A three-layer net can produce arbitrarily complex decision regions. The number of nodes in the second layer must be greater than one when decision regions are meshed or disconnected and cannot be formed from one convex area.

Multi-layered perceptrons can also be constructed with multiple output nodes when sigmoidal nonlinearities are used and the decision rule is to select the class corresponding to the output node with the largest output. Decisions are bounded here by smooth curves and not straight segments. The back-propagation algorithm of Rumelhart [110] can train these nets.

**The Back Propagation Algorithm (BP)**

The Back Propagation Algorithm uses a gradient search technique to minimize a cost function equal to the mean square difference between the desired outputs and the actual outputs of the net. The desired output of all the nodes is low (0 or 0.1). Otherwise this node corresponds to the class to which the current input belongs in which case the desired output is high (1 or 0.9).

The net is trained by selecting small random weights and internal thresholds and then all training data is pre-entered repeatedly. The weights are adjusted after every trial using side information. This information specifies the correct class. This is done until weights converge and the cost function reduced to an accepted low value. The high level procedural steps for BP-algorithm is as follows:

- **Step 1:** Initialize Weights and Thresholds;
- **Step 2:** Present Inputs and the desired Outputs;
- **Step 3:** Calculate Actual Outputs;
- **Step 4:** Update weights accordingly;
- **Step 5:** If cost function is acceptably low, STOP, otherwise: Repeat : Goto Step 2.
The BP algorithm has been tested with a number of deterministic problems such as the exclusive-or problem that was mentioned earlier and various other problems related to speech synthesis and recognition, and also on problems related to visual pattern recognition [Rumelhart, 110].

Kolmogorov proved that multi-layer networks with hidden layers can implement any mapping. The theorem [110, 211] states: the continuous function of N-variables can be computed using linear summations and nonlinear (continuously increasing) functions of only one variable. It states that a three layer perceptron with \((2N^2 + N)\)-nodes using the abovementioned functions, can compute any continuous function of N variables.

3.3.2.3 Kohonen's Self Organizing Feature Maps

The algorithm creates a vector quantizer by adjusting weights from common input nodes to M output nodes arranged on a two-dimensional grid [110].

The output nodes are interconnected with many local connections. Continuous-valued input vectors are presented sequentially in time without specifying the desired output. The input space is sampled through weights which specify cluster/vector centres. The specification is done such that the point density function of the vector centres tends to approximate the probability density function of the input vectors. The weights are organized such that the nodes which lie close together are sensitive to inputs that are physically similar. Output nodes will thus be ordered in a natural manner. A neighbourhood is defined around each node as shown in figure 11.

![Diagram of a Neighbourhood Defined Around A Node](image)

**FIGURE 11**

A Neighbourhood Defined Around A Node
As time elapses, the neighbourhood slowly decreases in size. Weights between input and output nodes are initially set to small random values and then the input is presented. The distance from the input and all nodes is calculated as a summation of the input to a node at time $t$ and the weight from the input node to the output node at a time $t$.

Then a certain output node is selected whose fault-measurement between an input node and that output node is a minimum. If we normalize the weight vectors to have constant length, the node with the minimum Euclidean distance (the norm of the fault) can be found in the net. This node will then form the dot product of the input and the weights. Then we must find the node (output) with the maximum value. This can be done by lateral inhibition (as in the MAXNET in the Hamming net).

Once this node, has been found, the weights to it are modified as well as all weights of all nodes in its neighborhood, so that they are all more responsive to the incoming input. weights eventually converge.

The algorithm performs relatively well in noise because the number of classes is fixed, weights adapt more slowly and because the adaptation stops after training.

3.3.3 Other neural network models

Various other models have been developed over the years. Examples of network models include:


- **Bidirectional Associative Memory**: Developed by Bart Kosko, 1985. Useful for content-addressable associative memory. Limitations include low storage density. Very easy network to understand [110].

- **Centebellatron**: Developed by David Marr and James Albus, 1982. Controls motor action of robotic arms [110].

- **Neocognition**: K. Fukushima, 1984. Can perform handprinted-character recognition. One of the most complicated networks developed to date. Limitations include the fact that it requires an unusually large number of processing elements and consequently a large amount of connections [110].

- **Madaline**: Developed by Widrow, 1978-84 [110]. Applied in radar jamming (adaptive nulling); adaptive modems, adaptive equalizers in telephone lines. Limitation is that it requires a linear relation between input/output.
3.4 SUMMARY

The six nets that have been discussed are common components in many more complex systems under development [106].

The greatest potential for these models lie in the area of high-speed processing which is possible through massive parallel VLSI implementations.

As already described, the basis of vision computing lies in the area of parallel processing architectures which are being used more commonly for this purpose.

In this section some neural network models that have been designed over the past few years, have been discussed. The oldest neural network model to date has been the Perceptron which was designed by Rosenblatt in 1958. The perceptron has proved useful in the construction of mechanisms which can explain some brain mechanics. These devices were, however, unable to solve several interesting cases. This was demonstrated by Minsky and Papert in [89] and by other authors in [110]. Minsky and Papert proposed a fundamental change in the original design of the perceptron. They introduced one or more hidden node-layers. The main task of these hidden node-units was to store the knowledge accumulated by several instances of the analysis of the problems. This knowledge is in the form of a connectivity pattern from one layer of node-units to the next. Connection weights are set up between these unit layers.

This gave the neural net the ability to be trained. This analysis which showed that a single layered perceptron had certain limitations concerning the solution of certain visual problems, as well as subsequent multi-layered perceptrons and other multi-layered neural net models where feedback circuits and energy functions were introduced, stimulated new interest in neural networks.

Within the last few years a flurry of activity has occurred. In 1985 the first commercially available neural computer was made. It was designed by TRW AT Research. Other neural computers followed for example Mark III, Mark IV and Odyssey [110].

The relevance of Neural Networks for computer vision is apparent: a key characteristic of intelligent behaviour is the visual ability to sense and perceive the external environment. This ability is essential for navigation and tracking as well as for manipulation of the environment. Neural equivalents of this human ability would change technologies such as recognition, robotics, automation, control of automation and automated navigation for future space missions to the outer planets and beyond [110].

Some of the networks that have been described had learning capabilities. This ability to be trained is basically the fundamental cornerstone of Wechsler’s Distributed Associative Memory (DAM) [110].

In [110], Wechsler describes a system which can recognize objects, regardless of changes in rotation or scale. The database memorizes the object and the information is then recalled to classify an object. The changes in rotation and scale can then be estimated. Furthermore the system has a low resistance to noise and occlusion. One of the papers in [110], describes several experiments in which an object was presented to the system, the system 'memorized'
the object and the object was then recalled from memory. The work is presently being extended to three dimensional object recognition. Further reading can be found in [108]. Furthermore, Wechsler shows in [110], that DAM could be used for BIN- PICKING. For a discussion of BIN- PICKING the reader is referred to CHAPTER 7 of this dissertation. The system also demonstrates invariance to geometrical transformation and a robust reaction towards noise, occlusions and memory-loss.

From the above mentioned examples, it is clear that neural network models have credibility in computer vision applications. Thus, to summarize, neural networks are merely an attempt to simulate the neuronic networks found in the human brain. This chapter has discussed the neural net as a theoretical concept and its relevance to the vision problem. The main reason neural networks are used to simulate intelligent behaviour is that they are an endeavour drawing on fields such as neuroscience, psychology, signal processing, physics and cognitive studies.
4.1 INTRODUCTION

The very first step in the vision process is to form an image. In the related field of image processing, the input and output are both images, the only difference being that the output is an enhanced version of the input image. There are several techniques for enhancing the image (such as restoration and reconstruction), which will be discussed in chapter 5.

The aim of this chapter is to look briefly at the various factors that must be considered in order to form an enhanced output image.

An image is formed when a sensor registers radiation because of interaction with physical objects. This chapter, will examine the various methods by which this radiation-sensing from the sensor is transformed to an enhanced image. This involves an analysis of the theories of monocular and binocular vision as well as reflection theory, which is concerned with the amount of light radiation reflected from the object. The importance of the frequency domain is addressed when the role of the Fourier transform in image processing, is examined. Lastly the ways in which an image is digitized, will be discussed.

The image function is a mathematical expression of the image. The geometric function projects the three dimensions onto two dimensions. The representation plane is usually two-dimensional, on which a three-dimensional object must sometimes be projected. The ways in which an image is formed on such a plane, will be discussed.

When considering the radiometric model, aspects like image-forming geometry, variation in the light source, reflection from objects to the sensor, must be considered.

When the spatial model is regarded, tendency is to discuss the important role of transforming the image from its spatial domain to its frequency domain for analysis. The best known transformation for this purpose is the Fourier Transform, the importance of which transformation in the field of image processing, will be discussed.

The digitizing model is concerned with the process of gathering different, discrete image-samples. The idea of representing images in digitized manner, is discussed.

4.2 THE IMAGE FUNCTION

The image function is a mathematical representation of the image.

Most images are represented by two coordinate-variables x and y, which are also sometimes called the arguments of the function. The function produces a vector quantity. Typically, a specific brightness location at (x,y) would be described by \( f(x) = f(x,y) \), where \( f(x,y) \) is the brightness of the graylevel of the image at coordinates (x,y).

In the process of forming an image, the image consisting as continuous functions must be transformed to a series of discrete points [103, 104, 93, 94]. This strategy must be adopted because the number of image elements per square unit available to describe the image, is finite. The number of image units available for picture description is called resolution. The resolution determines the amount of detailed information that can be represented in
the image. Every picture element produces a small area of detail. It is therefore essential that the best point-representation is chosen for a particular visual application.

In a highly constrained vision environment where lighting is strictly controlled, for example, resolution is of the utmost importance. This is evident in the area of visual inspection, where parts are inspected for the presence or absence of specific features. It is obvious that the image formed by the vision inspection system must be as good as possible in order to make the correct inspection decisions, as any loss of information in the image would lead to the system making an incorrect decision.

Usually, the image representation is described as a discrete function. This means that the representation of the image as a continuous function is converted to a discrete function to represent the image. The delta-function is used to describe the image comprising samples of discrete points. The function as in [94], is defined as:

\[ \delta(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ \approx & \text{if } x = 0 \end{cases} \]

\( \delta(x) \) could be further interpreted as being the limit of a finite set of n-delta functions each concerned with only one spatial coordinate x namely:

\[ \delta(x) = \lim_{n} \delta_{n}(x). \]

where:

\[ \delta_{n}(x) = n \quad \text{if} \quad |x| < \frac{1}{2n}, \]

\[ = 0 \quad \text{otherwise}. \]

It must be pointed out that images are represented by functions of two spatial variables \( f(x) = f(x,y) \) where \( f(x,y) \) is the brightness of the grey level of the image at a spatial coordinate \( (x,y) \). In the definition of the delta-function above, only the case for the spatial coordinate \( x \) was defined. Coordinate \( x \) may have a grey level of 0 or above 0. The delta-function is a mathematical way of indicating how an image consisting of an infinite number of image functions describing the image, can be represented on a two dimensional grid which can only represent the image in a finite set of points on the grid. Thus an imaging function's aim, is to represent images on a grid of limited size. In the next section, an empirical method of forming a monocular image on an image plane, is briefly discussed.

4.2.1 Monocular Image Forming:

The fundamental model by which image points are projected through a camera opening on an image plane, is called point projection. The point of a projection coincides with the point at the back of the image plane. Figure 1 demonstrates this. In this figure, the coordinate of the A-image is non-existent. The image plane in figure 1 is in front of the point of projection. Therefore the image formed is still upright. If the image plane was behind the point of projection, in other words, \( z = 0 \), then the resultant image would have been turned upside down.
Monocular Image Formation When The Focal Length Is 0, and For The Focal Length Greater Than 0.

In figure 1(B) the focal length has a positive value because \( z > 0 \), whereas in figure 1(A) the focal length was zero. The equivalent results for figure 1(B) can be obtained, by turning A in figure 1(A) upside down. As can be seen in both figures, as the image B's projection nears the viewpoint, it gets bigger.

Assume that the height of the object (scene) is \( h \). To find the coinciding height of the image, figure 2 must be consulted: Assume the projected height of the object is \( h' \). The relation between \( h' \) and its real height \( h \), its \( z \)-coordinate and its focal length, is required.
Because the geometry of the triangles ABC and ADC is the same, it may be stated that the following ratio's are equal:

\[
\frac{h_y'}{f} = \frac{h_y}{f-h_z}, \text{ where } f \text{ is the focal length.}
\]

Thus:

\[
h_y' = \frac{f.h_y}{f-h_z}
\]

Similarly for:

\[
h_x' = \frac{f.h_x}{f-h_x}
\]

Thus, it can be seen that the position of the image on the image-plane depends on the position of the scene and the focal length being used.
4.2.2 Binocular Image Formation

When binocular image formation is considered, the problem of above is extended so that image of an object is related to a certain distance separating the image forming devices and also the focal length being used. To obtain this information, we must turn to figure 3:

In this figure 3, the two eyes do not converge, but are parallel and looking into infinity in the z-direction.

From figure 3, the following geometrical information is extracted:

\[
\begin{align*}
x_2 &= \frac{(x_1 - d)f}{f - z} \\
x_3 &= \frac{(x_1 + d)f}{f - z}
\end{align*}
\]

From the above:

1. \((f - z)x_2 = (x_1 - d)f \) and
2. \((f - z)x_3 = (x_1 + d)f \)

\((2 - 1):\)

\[(f - z)x_3 - (x_1 + d)f - (f-z)x_2 + (x_1-d)f = 0\]
Thus, through algebraic manipulation:

\[ z = f(x_3 - x_2) - 2 d f / (x_3 - x_2) \]

Thus, the position of the image can be determined by the length of the two image forming devices (the baseline) as well as the focal length (f). If this baseline's length is unknown, it must be determined. This means determining the position of points \( x_3 \) and \( x_2 \). This process of determining this depth-information is also known as triangulation [85].

When one is seeking three-dimensional sensing, the biological existence of stereo vision as motivation to perform this artificially, cannot be denied and must be investigated. Two cameras can give relative depth or absolute three-dimensional location, depending on the elaboration of the processing measurements. The process is described conceptually: [11, 85, 105, 125].

- Take two images separated by a baseline.
- Identify points between the two images.
- Use the triangulation method to derive the two lines on which the world point lies [94].
- Intersect the lines.

### 4.2.3 Reflection

Reflection of an object determines how brightly it will appear in an image. The brightness will be determined by the object's surface characteristics and geometrical approach of the image forming process [103, 242, 85].

A fluctuation in light and energy \( E \) is measured in a watt-unit. Brightness is measured in terms of area and solid angle. The radiation-intensity \( I \) of the source is the excited fluctuation per solid-angle unit. The following formula will give \( I \) [85].

\[ I = d \Phi / d\omega \text{ watt per angle unit, where } d\omega \text{ is an incrementing solid angle.} \]
The solid angle of a small area (say the area is \( dA \))**perpendicular** on a radius \( r \) is

\[
d\omega = \frac{dA}{r^2} \text{ watt per unit length}
\]

The brightness of an image at a point is proportional to the radiance of the scene from which the image is produced. The grey level is a numerical measurement of the brightness in an image. The brightness of the image is usually called image **irradiance** and is the amount of light flooding (or flux) a surface element \( dA \). This flux is given by:

\[
E = \frac{d\Phi}{dA}
\]

The fluctuation-excitation from the surface of \( dA \) is described as the radiance \( L \). \( L \) is the amount of light emitted by a surface area in a unit solid angle of the area:

\[
L = \frac{d^2 \Phi}{dA \cos \theta \, dw}, \text{ with } \theta \text{ being the unit angle from the surface normal.}
\]

The brightness of the image at a point \( f \) (not the focal length), is proportional to the scene’s (from which the image originates) brightness. The grey level is a quantifiable measurement of \( f \). \( f \) is also dependent on the reflection-properties of the surface formed in the image. The reflectance of a surface is given by its two-directional reflexive distribution function (TRD) [85]. The TRD is calculated by the ratio of the reflected light radiation in the direction of the viewer with respect to \( f \) in a direction of a small area of the source.

### 4.2.4 How does geometrical properties affect the image forming process?

Geometry [88,93,86,242,110,103] is one of the tools that can be used in techniques for computer vision. When using geometry as a **tool** for computer vision, the intention is to show the relation between grey levels and the brightness of objects that become part of the image [85,104]. In [85], a formal mathematical analysis for various geometrical concepts of an object, is given. These concepts are:
(1) \( f \), the image radiation intensity, 

(2) \( L \), the scene radiation intensity. 

From the mathematical analysis in [85], it is found that \( f \) is directly proportional to \( L \); that is, \( f \propto L \). (\( f \) relates directly to \( L \)).

Furthermore, it is found that the quantity of proportionality involves a fourth-power cosine of the incoming angle on the lens-axis as depicted in figure 4 [85].

\[
E = \frac{\pi}{4} \cos^4 \alpha (D/f_p) L
\]

**FIGURE 4**

From [85], the quantity of proportionality (\( E \)) is given by:

\[
E = \frac{\pi}{4} \cos^4 \alpha (D/f_p) L
\]
In [104],[236] various properties of geometry are defined. These properties include:

- Borders
- Thickness
- Elongatedness
- Convexity
- Connectedness
- Area
- Perimeter
- Compactness
- Components
- Holes

All of these can be used as guidelines for the classification of grey level membership of a certain brightness set. Assume the grey levels range from 0 to 1 (usually levels range from 0 to 256). Grey levels can then be classified in terms of a certain brightness set, say 0, meaning the grey level at that point is completely dark or black and 1, meaning it is completely bright at that point. Between 0 and 1 there may be a fuzzy amount of darkness and brightness values.

As pointed out in [104], most of these properties are 2D and are of very little value if the orientation of the scene is unknown beforehand.

### 4.3 THE SPATIAL DISTRIBUTION OF POINTS IN AN IMAGE

#### 4.3.1 The Fourier Transform

An image is a spatial varying function [19,103,110,85]. It is a spatial representation of an two dimensional or three dimensional object, or it can be the representation of another image. In computer vision, this image is usually referred to as a digital image or video image. A video image is an image in electronic signal format, capable of being displayed on a cathode ray tube screen or monitor. The image can be created by a charged coupled device camera as is the case in many astronomical studies, a tactile sensor, a range sensor or a frame buffer which can control an analog to digital converter. The grey level is a number assigned to position on the image. Thus, in one domain the image is composed of of intensity values. The image is usually called the intensity image. The Fourier transform is used to transform the intensity image into the domain of spatial frequency.
4.3.2 What is the Fourier Series?

The Fourier series is a set of basic functions which can be linearly combined to construct with relative accuracy any arbitrary periodic function. They provide a unique way to express a periodic function in terms of components at discrete frequencies, resulting in an explicit description of the frequency composition of the function. In computer vision applications, the discrete Fourier series is mainly used because all operations involve digital signal analysis.

The Fourier Transform is used to analyse the differences in grey levels over small distances (spatial variations of grey levels) [19]. It transforms an intensity image into an image in the spatial frequency domain. The definition given here is only valid for the one dimensional case and can be easily extended to the two dimensional case. The transformed spatial frequency coordinates in this domain is denoted by (u,v). In the one dimensional case, it is denoted merely as (u). The one dimensional Fourier Transform is given as:

\[
F[f(x)] = F(u),
\]

where:

\[
F(u) = \int_{-\infty}^{\infty} f(x) \ast \exp(-j2\pi ux) \, dx, \text{ where } j = -1^{1/2}
\]

F(u) is simply another representation of the image function f(x). The Fourier Transform expresses a function as a sum of sine waves of different frequency and phase. It transforms a function to a linear combination of sinusoidal bar patterns, having different orientations and frequencies. The coefficients of the frequencies can be arranged into a two dimensional array with the distance from the origin of the array corresponding to the spatial frequency and the direction to orientation. Peaks in the sinusoidal waves indicate that the grey level values in the image, are periodic. The rate of drop in the coefficients away from the origin, indicates the rate of fluctuation in the grey level values. An image is coarse, if the drop rate is fast, while an image is busy if the drop rate of coefficient-values is slow. If the grey levels varies in terms of direction, it means that the grey levels are more proportionately distributed.

One can easily transform the spatial frequency image back to the original intensity image by applying the inverse of F(u), namely f(x), on F(u):
the meaning of \( F(u) \) can be better understood when the equation above is applied for a specific value for \( x \), say \( x_0 \):

\[
f(x_0) = \int F(u) \exp \left(j 2\pi u x_0\right) du
\]

Thus, \( f(x_0) \) can be interpreted as follows: that any point in an image can be represented by a weighted sum of the number of complex exponential functions, at different spatial frequencies \( u \). \( F(u) \) is a weighted function of these different functions. If the frequencies are not widely distributed, it means that the grey levels (grey level intensities) vary slowly in the image such as will be the case for a continuous surface. If the components of \( F(u) \) have a very high frequency, it is usually concluded that the grey levels are changing very quickly in relation with one another as is the case for edges of objects in an image. Thus, the Fourier Transform gives an indication of image brightness and thus how measurable the geometrical properties of the image are. It should be pointed out, however, that the Fourier Transform \( F(u) \) was defined for the one-dimensional case. For the two-dimensional case, the definition is similarly formed \([19,85]\). The above equations with which the Fourier transform was defined, is in a continuous form. The Fast Fourier Transform is almost universally used in image processing and computer vision because it is versatile to manipulate the transform in the frequency domain compared to the digital domain.

4.3.3 Convolution of spatial functions and the Fourier transform

In machine vision, the spatial joining of two functions is widely applied. Applications include template matching, edge detection, border analysis (border decompositions) image smoothing, image interpolation \([19,85,105,110,104,77,103]\). Sometimes, convolved functions can be large and complicated. Computation of the spatial convolution can be very slow and the results inaccurate. How spatial convolution and Fourier analysis relates to each other can be of great assistance in these cases. The joining of two functions \( f \) and \( g \) is a new function \( h \) with argument (displacement) \( y \) and is given by:
The transform of the convolution-equation given above is:

\[ h(y) = f(t) * g(t) = \int_{-\infty}^{\infty} f(y) g(t-y) \, dx. \]

If we substitute \( x = t - y \) we get the following:

\[ H(t) = f(t) * h(t) e^{-jwt} \, dt = \int_{-\infty}^{\infty} f(y) g(t-y) e^{-jwt} \, dy \
= \int_{-\infty}^{\infty} f(y) e^{-jwt} \, dy \int_{-\infty}^{\infty} g(x) e^{-jwx} \, dx \]

This results in:

\[ H(t) = F(jw) G(jw) \]

The spatial convolution is the multiplication of the Fourier transforms of the two functions. The above can be visualized as one function being "spread" or "wiped" over another. When an image is formed by an optical system, the image can be described by a convolution. The system then has a so-called point-distribution function [85]. It is the image of a single point and is the ideal point distribution function.

In [85], further properties of the Fourier transform are given with respect to the spatial domain and frequency domain. These include the Fourier transform for scaling, shifting, conjugation, symmetry and differentiation. Some of these properties are quite useful for image formation [19, 110, 104, 85]. For example, the convolution of the Fourier Transform can be used for:

- **Large template matchings:**

It can be time consuming to calculate spatial convolutions of large templates. The size of the transform is dependant on the size of the search area. For the same size search area,
small template convolution may be faster if spatial convolution is used and then as the template's size increases, frequency convolution may be used because it will be faster.

- **Repetitive convolution (joining) of known functions:**

In edge detection, image smoothing, image interpolation and template matching, the same functions appear repeatedly in the convolution integral. Thus, if the transform can be implemented so that the same functions are not repeatedly transformed, one less transform can be performed for each frequency convolution, making it faster.

In many cases, edge detection in machine vision involves the approximation of the first (Sobel, Roberts gradient) or the second (Laplacian, Marr-Hildreth operator) derivatives. The Fourier transform can again be useful especially if the spatial domain is large. In [19] the Fourier transform of a functional derivative is given. In this example, only one transformed is performed: forward and inverse. The Marr-Hildreth algorithm operates in this way. Edges are detected by finding the zero-crossings in the Laplacian of a Gaussian smoothed image. For a definition of the Laplacian operator, the reader is referred to chapter #5.

Because of the continuous development of VLSI-technology and other hardware technology, higher processing capabilities will be the result. It is therefore quite obvious that the Fast Fourier Transform will play a more important role in future image-formation processes.

### 4.4 DIGITAL IMAGES

In computer vision and image processing applications, images are represented digitally [85,110,107,104,103,105,93,79]. The digital images is represented by an m-vector (discrete valued image functions f(x) implemented in 1, 2, 3 or 4 dimensions.

The image is a continuous and is represented by samples that are taken at constant intervals. For f(x) above, the value of m is usually 1 and the domain and range of f(x) are discrete. The domain of f is usually finite (and usually a rectangle), and the range of f is positive and also bounded namely $0 \leq f(x) \leq M$, where M is some integer. While the image is being sampled, its intensities are quantified in different amounts of grey levels.

When an image is sampled, two things must be considered:

1. The length of the sampling interval: the length of the interval will determine the amount of detail that will be present for subsequent analysis

2. The spatial representation pattern: connectedness and distance are influenced by this choice.
4.4.1 The Spatial Representation Pattern

The spatial samples of f(x) can be represented by small cells of grey levels that will comprise the image [105,85]. These cells are commonly called pixels (picture elements). The pattern onto which the plane is divided is called the tessellation. In [85] three commonly used tessellation examples, are shown. One is the rectangle tessellation which is most commonly employed. Other examples are the triangle and hexagonal tessellations. As stated, the rectangular tessellation is universally used. It does, however, present the problem of connectedness, related to the definition of neighbour pixels for a pixel (say p). There are two suggestions on how to solve this problem: 4-pixel connectedness or 8-pixel connectedness. Figure 5 illustrates this concept graphically.

4-pixel Connectedness

8-pixel Connectedness

FIGURE 5

A fundamental concept used in sampling algorithms [104,105] is the distance between two pixels. The distance between two pixels is a distance d, which means that:

1. d(x,y) = 0 if x = y where x and y are pixels. In this case, x and y are the same pixel.

2. d(x,y) = d(y,x) which means the distance between a pixel x and pixel y is the same as d(y,x) the distance between pixel y and x.

3. d(x,y) + d(y,z) = d(x,z). On the rectangular representation plane, where the distances between pixels are all in the same unit-lengths, any of the standard distance formulas can be used. These include the Euclidian technique, city block or Chess Board.
4.4.2 The length of the sampling interval

The following must be asked: in the process of sampling the image points, where intervals between each point, are plotted; how far apart must these sampling-intervals be in order to construct an accurate image of the scene?

We want the points sampled to form a continuous function image, as well as a quantifiable distance from which an image can be constructed as image points. In [85], the mathematical analysis is made using the "comb-function" of the Fourier Transform. From this analysis, it is found that, in order to sample correct (accurate) versions of the scene, it must be ensured that spatial frequencies are not greater than half of the image sampling-function. For example, if the frequency in the spatial domain is $x_0$, then by analysis, it is found that the frequency-interval (in the frequency domain) must not be greater than $1/2 \cdot x_0$.

4.5 SUMMARY

The main aim of this chapter was to examine the way in which an image is formed. This was done for monocular image formation as well as binocular image formation. The importance of the Fourier transform in the vision problem whose function is to transform the image intensities to a domain of spatial frequencies, was then addressed. The idea of the convolution of spatial function were also given and discussed in relation with the Fourier transform. In the last section the types of representation pattern used to digitize an image, were discussed. In the last section the idea of sampling and what must be considered when we sample and image, was investigated.
5.1 INTRODUCTION

The aim of this chapter is to look at the various mathematical techniques that are used in order to analyse detected objects and extracted objects in images. First, we look at the techniques that are used to detect objects in an image, and secondly, we look at the various techniques employed when we physically extract the object from the image. An image is described by variations of brightness from point to point in an image-plane.\[127,54,73,76,79,85,93,94,97,103,104,110\]. Thus, this chapter discusses the approach that must be followed after the image was obtained.

In order for a digital computer to process the image, the image must be transformed to a digital image. A digital image is a discrete ordering of numbers which represent intensity-values on a squared grid. The grid consist of a number of discrete points each having a certain brightness-value. A single element of such an ordering of brightness-values is called a picture element or pixel. The brightness-value assigned to pixels, is called the grey level.

In a previous chapter we looked at the mathematical theory of image formation or sampling. In this chapter, we'll be discussing the various techniques that are used to find images or parts of images and then to analyse the image; finding certain properties of the image. Thus, we'll be asking the question: "After we have sampled the image, what can we find out about it and how?" We'll be answering this question in an informal way.

5.2 THE IMAGE ANALYSIS PARADIGM

Rosenfeld have suggested a procedural approach to image analysis in [103, 94]. This approach was done for two and three-dimensional case.

5.2.1 2-D Image analysis paradigm

The paradigm of two-dimensional image analysis is discussed in [94]. The scenes that are considered in this paradigm are two-dimensional, for example views of the surface of the earth from geosynchronous orbit (36000 km).

The properties of the object as well as its relations towards other objects, must satisfy certain pre-defined constraints. Object recognition is the process of searching for a collection of image parts (areas in the image) that correspond to the object parts. Then the defined constraints (defined beforehand) must be satisfied.
THE TWO DIMENSIONAL IMAGE ANALYSIS PARADIGM

2D SCENE ~ Image

- (numerical intensity values are assigned to pixel positions)

SEGMENTATION:

Feature Detection

- Local patterns in the image are identified. These patterns can be step edges, curves, lines, corners and so on.

- Patterns are usually detected through template-matching. A symbolic image is established.

- Subpopulations of images are identified (Pixels are now given labelled-values).

RESEGMENTATION

Property measurement of label - valued pixels

- Pixels with label-values are now partitioned into connected regions. Various parts can now be determined in a region.

- The relational structure of object-parts is established.

- The process of labelled pixels being grouped into new sets of pixelgroups, is called resegmentation.

- The grouping is done based on a set of criteria being predefined for pixels that are being grouped.

- When a collection of object-parts have been established, properties can be measured and relations among object-parts.

- The process of matching subgraphs of the established labelled graph consisting of nodes representing object-parts and arcs representing the relations between the parts, is completed. The sub-graph must be matched against a predefined subgraph. In this way, a relational structure is established. Thus, after the relational structure has been established, a labelled graph exists where each node represents an object part and the arcs of the graph have labelled values representing relational information between the parts. During model matching, we match a subgraph of this labelled graph against a predefined (already existing) subgraph. Recognition of the object will then be established based on the results of this graph-analysis.
5.2.2 The 3-D Image Analysis Paradigm

3-D image analysis is much more difficult to accomplish than 2-D image analysis for various reasons. Variations in the surface's orientation and illumination make it difficult to identify local patterns. Furthermore, it is difficult to identify subpopulations of images because of these varying degrees of orientation in the surface of the object. Thus, because of this, segmentation is a more complex operation than in the 2-D case. Resegmentation becomes more difficult because it was difficult to establish pixels with the labelled values as described in the 2D case. Therefore it will be that more difficult to group the labelled pixels into pixel groups. Property measurement is also then more complex because of varying orientations of objects. David Marr has suggested in [94,103] in his book called vision that the 3-D scene analysis paradigm also comprises of certain phases of analysis before the final recognition of the object(s) is made. This phased recognition process is depicted in figure 1 [94],

![Diagram of 3-D scene analysis paradigm]

The Analysis-paradigm of a 3D Scene.

The outline of figure 1 as described in [94,103]. In the 3-D case we follow the strategy of defining objects as certain spatial arrangements of parts. The properties of these parts as well as their relations must satisfy certain predefined constraints and criteria. Recognition of an object then becomes the task of looking for a collection of image parts that correspond to the same parts seen from a known viewpoint. Conventional segmentation techniques are not efficient for finding parts in the image that correspond to the parts of the objects seen from a certain viewpoint. That is because the visible surfaces will in general be curved and the brightness of the surfaces that are illuminated by a light source will vary. However, if we know at every point of the image what the orientation of the surface normal at the corresponding scene point is, we can detect edges of orientation. These edges of orientation will enable us to find sudden changes in the surface normals and thus will enable us to determine the orientation of the surface from these changes. The techniques that have been used to find such surface information from the image are called 'Shape from X' or shape detection with the help of certain clues extracted from the image. Three classes of 'shape from X'-techniques exist namely [86,85,79,60,94,34]:

- **Shape from X**: Techniques for finding the shape of an object from a single image or a sequence of images. These techniques rely on the assumption that the surface of the object is smooth and the lighting is constant.
- **Shape from Stereo**: Techniques that use two or more images of the same scene taken from different viewpoints to infer the 3-D structure of the scene.
- **Shape from Texture**: Techniques that use the texture of the surface to infer the shape. This approach is based on the assumption that the texture is distributed uniformly on the surface.
1. **Shape from Shading.**

The brightness of a grey level at point P depends on the following:

- The illumination intensity incident on P.
- The reflective properties of surface S on which P lies.
- The spatial orientation of S at P.

If the brightness of point P in a scene is given, the technique of establishing the shape of an object from its shading, aims to do the following:

- Recovery of the intensity of the illumination at P.
- Recovery of the reflectivity of surface S at this point P.
- Recovery of the spatial orientation of surface S at point P.

If sufficient boundary information is given, a plausible orientation function for a smooth surface can be determined. If several images of the same scene is used, more information about the scene can be obtained. If these images each receive the same degrees of illumination from different degrees of angles, the resultant shading at each point can give partial information of the surface orientation at that specific point. The different images which are recorded in this way, can all be recorded and the constraint information of all the images can the be intersected to obtain the global shape of the object.

2. **Shape From Texture or Patterns**

Assume a surface S which is uniformly textured, covered with markings whose shape, size and spacing are stationary. Assume also that S is isotropic. This means that any cross-section through P situated on S, has the same probability densities in size and spacing. If S is slanted towards the view direction at P, the surface texture at P will also be slanted. The surface texture at this point P, will then be non-isotropic. The aim of establishing the shape from the texture, is now to calculate the amount of slant from the amount of variation in the mean size of the markings as a function of direction. The mean size is smallest in this specific direction. The drawback of this approach of shape-establishment, is the fact that the sizes of the markings cannot be measured without first segmenting them from the image. This process can have faults in segmenting the sizes of the markings. For this reason, an alternative approach exists [103].

If patterned illumination is used, one can use a pattern of light with dark bars. The slant component in the direction across the bars will reduce the bar spacing in the image. By calculating the amount of reduction in the bar spacing, the amount of slant can be obtained. In this way, information regarding the shape of the object is obtained.
3. Shape From Shape.

The basic aim of this technique, is to derive the object's orientation from the shape of its image. If this shape is unknown, plausible assumptions are needed about the shape of the object. Different classes of assumptions may be defined. The following are two classes of such assumptions which can be used to determine the shape of the object:

**CLASS A:** Assume that the object is viewed from general viewing positions. Under this assumption small changes in the movement in the sensor of the vision system, results in small changes in the image. Certain conclusions can be made regarding the quantitative shape changes of the object's image. Another assumption may be that if the image has continuous curves, then the scene from where the image originates, has corresponding continuous curves. This is also true for parallel lines in the image resulting in parallel lines in the scene.

**CLASS B:** One can also make the assumption that the 3D shape in a scene is the simplest of shapes that will form shape of the image.

The basic working principle of the above techniques is that they are used in the first place to extract edges from the image. They operate on a map of features rather than grey levels. This map of features (image) are derived from the image. This map is called the "primal sketch" by Marr. Between the primal sketch and the establishing of a 2D surface map in figure 1, the process of recovery takes place. Recovery yields the 2D surface map. Recovery is defined by Marr as the process of finding the surface orientation from an image. The 2D surface map contains only the information of surface that are illuminated and visible. Once the surface orientation has been established at each point of the image, it is then feasible to segment the image into parts that will correspond to surfaces that are smooth in the corresponding scene from which the image originates. Like before, the feature detection process yields a symbolic image. In this transformed image, the pixel values denote types of features. Again, part-properties and relations between the different parts can be established and represented by a surface graph. This graph will only contain information of visible surfaces.

It must be pointed out, however, that even if the above is accomplished up to the point of defining a surface graph, the recognition of the image can still be very difficult. Again it is because of object-orientation: the image may depict only one side of the object and usually we don't know which side of the object is shown in the image.

If the object's viewpoint is unknown, the recognition problem becomes much more difficult compared to the case where the viewpoint is known beforehand. In the case of an unknown viewpoint we will not be able to make a prediction of how the object will appear in the image. Predictions can be made of how the image may appear from a great many viewpoints. We can then look for the resulting image configurations of parts.

This approach is called the brute-force approach but is undesirable because it is expensive in terms of computation. An alternative is suggested in [94] that works with a certain set of different features of an object. The parts of the image are then tested against Sis set of features. Certain conditions apply to a set of features in order to be used as testing features. They are:
Some of the features will always be visible regardless of object orientations.

- They must be easy to detect in the image
- They must be distinctive features
- They constrain object position and orientation.

This approach generally solves the recognition problem from an unknown viewpoint. Recognition is simple if the object itself is unknown but not its position or orientation (pre-defined).

5.3 PROPERTIES OF AREAS OR REGIONS

In most cases, we can view pictures as a set of roughly uniform textured areas [85, 104]. Very often, the object is on a background and its texture is different from the texture of the background. A property of such an area can be determined by a local operation on every pixel [103, 104]. A certain value is yielded by such a local operation at every pixel p in the image. This value is determined by the grey scale of the pixel and a set of neighbouring pixels. In sections 5.4, 5.5, 5.6 we discuss three local operations that are employed in scene analysis to detect properties of areas in the image.

5.3.1 Determining the properties of an area by analysis of the power spectrum

This technique involves the analysis of the local ordering of gray scales in an area. In this regard we must analyse the magnitude of the area's Fourier Transform defined in two dimensions. If the grey level at position (x,y) in the area is f(x,y), then the two dimensional Fourier transform is defined by:

$$F(u,v) = F[f(x,y)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{2\pi i (ux + vy)} f(x,y) dx dy$$

The power spectrum is given as

$$|F(u,v)|^2 = F(u,v) F^*(u,v)$$

$F^*(u,v)$ is the complex conjugate of $F(u,v)$. If the ordering of local patterns are periodic with the period lengths $(u_0, v_0)$; then a high value will be yielded at $(s/u_0, s/v_0)$ where $s$ is the diameter of the area. We say an area is "busy" if the grain of the pattern is very fine.
and the patterns are closely packed together. This results in higher values in the power spectrum. These higher values are far from the origin of the area. An area with a rougher texture will yield higher values near the origin.

### 5.3.2 Analysis of the statistics of local properties

This is an alternative technique for texture analysis in an area. It involves that we analyse the frequency distribution of various local properties's values. A typical local property in this case is a directionally-based difference in averages measured over non-overlapping neighbourhoods of the picture.

Assume we have a neighbourhood with radius $r$ with its midpoint at $(x,y)$. Let $A(r)(x,y)$ be the average of the grey levels $f(x,y)$ in the neighbourhood of radius $r$. The difference of averages between non-overlapping $A(r)(x,y)$'s measured in a certain direction $\Theta$ away from $A(r)(x,y)$, is defined by:

$$D(r, \Theta)(x,y) = A(r)(x + r \cos \Theta, y + r \sin \Theta) - A(r)(x - r \cos \Theta, y - r \sin \Theta).$$

If the texture of an area is "busy", $D(r, \Theta)(x,y)$ should have large values for small values of radius $r$. If the texture is coarse, small values for $r$ will yield small values for $D(r, \Theta)(x,y)$. In this way, we can get an idea of how the greyscale is distributed in an area in a specific region.

### 5.3.3 Analysis of joint grey level statistics

This technique involves the analysis of texture based on the joint frequency distribution of pairs of grey levels at different separating distances $(\Delta x, \Delta y)$ over the whole area, where $(\Delta x, \Delta y)$ is the distance between grey level pairs $(x1, y1)$ and $(x2, y2)$ and where $\Delta x = x2 - x1$ and $\Delta y = y2 - y1$. The process begins by dividing the greyscale into $n$ intervals. The frequency-distribution can be represented by a $(n \times n)$-matrix $M(\Delta x, \Delta y)$. The $(h,k)$-th element $m_{hk}$ is the number of times a point with a certain grey level will appear in the $k$-th interval in position $(+\Delta x, +\Delta y)$ or $(-\Delta x, -\Delta y)$. This position is relative to a point with a grey level in the $h$-th interval. If the area has a coarse texture, the entries of $M(\Delta x, \Delta y)$ will be situated around the main diagonal for small values of $((\Delta x)^2,(\Delta y)^2)^{1/2}$. The reason for this is that pairs of points which can be separated by $(\Delta x, \Delta y)$, have similar grey levels. For a "busy" texture, the entries of $M(\Delta x, \Delta y)$ will be more widespread in $M(\Delta x, \Delta y)$.

### 5.3.4 The normalisation of greyscale

The techniques described in 5.3.1, 5.3.2 and 5.3.3 is sensitive to changes in the grey scale of the picture [92,103]. This means that any changes made in the greyscale, will yield different results in terms of the specific technique that is used. The problem forces us to normalise the grey scale before any analysis is done. Various agreements exist on how to do this.
normalisation. One such agreement is to force the picture to accept a specific standard frequency distribution of grey levels. In other words, all grey levels occur equally often. For example, a uniform distribution of grey levels where all grey levels will appear an equal number of time. The positive result of normalization is that the problem of greyscale change in the picture does not affect the results of the analysis of the picture. The texture remains unchanged while the grey scales are more fierce or "harsher".

This technique is also known as histogram equalization. The normalization of greyscale in an intensity-scaled image has aims to distribute the intensity values forming the image as evenly as possible over the entire grey range of \( \{0 \leq f \leq Q-1\} \). The value of Q is usually finite, representing discrete intensity values with which digitized images are formed. The intensity mapping function \( T(i) \) for \( i = 0, 1, \ldots, Q-1 \) is formed and stored as a Q-dimensional vector. Scaling is done by indexing this vector using the intensity values of the input image pixels as indices. The effect of intensity mappings is demonstrated with the aid of intensity histograms. Given that the intensity values of image pixels \( f \) are represented by \( q \) bits, \( Q = 2^q \) different intensity values can be distinguished for \( \{0 \leq f \leq Q-1\} \). The Q-dimensional vector \( h(i) \) whose elements can be calculated by

\[
h(i) = \sum_{m=0}^{M} \sum_{n=0}^{N} \delta\{f(m,n)-i\}
\]

where \( \delta(a) = 1 \) for \( a = 0 \) and \( \delta(a) = 0 \) elsewhere.

is referred to as the intensity histogram. The \( i \)th element of this vector represents the number of pixels in the image with intensity value \( i \). In the ideal case the histogram of an image transformed in this way would have constant values \( h(i) = MN/Q \), where \( M, N \) are arbitrary integers [263]. It is possible to achieve an intensity scaling which changes the intensity values of a given image so that the histogram of the resulting image comprises an approximately constant number of picture elements at constant intervals. Intensity intervals in which the histogram of the initial image has high average values must be scaled using a mapping function with a high derivative in the given interval in order to reduce the mean frequency of occurrences in these intervals, and vice versa. It can be shown that the cumulative distribution function

\[
H(l) = \frac{1}{MN} \sum_{\alpha=0}^{j} h(\alpha)
\]

exhibits this property. By using a suitable normalization, a scaling function can be produced from the image histogram which transforms the intensity values of the image pixel by pixel so that the output image has an approximately constant number of image pixels in constant intensity intervals. In figure 2(b), the effect of greyscale (or histogram equalization)nor-
malization is shown. The brightness values of pixels at low intensities are shifted to higher intensities. The average brightness as well as the contrast of the original image is increased. In figure 2(d), the histogram of the intensity-mapped image is shown. The cumulative distribution functions 2(a) and 2(b) are shown in 2(e) and 2(f) respectively [263].

**FIGURE 2** Example of histogram equalization: (a) original image; (b) intensity-scaled image; (c), (d) histograms of (a) and (b); (e), (f) cumulative intensity distribution functions corresponding to (a) and (b)
5.4 THE DETECTION OF OBJECTS IN A PICTURE

When doing image analysis, the task of object-detection comes into focus. We’ll need the ability to successfully detect the presence of objects in a scene for subsequent analysis. Furthermore, we need techniques that will enable us to detect edges of different textured areas. In this section, we will be discussing three techniques that are commonly used in the detection of objects namely [10,22,54,73,76,85,93,94,103,104]:

- Template matching
- Edge-detection
- Moments

5.4.1 Template Matching.

Template matching is the easiest way of detecting a certain feature in an object. The feature of the object is compared to a prototype or template of the feature. This template is kept in storage memory and is used as a matching reference for the feature of the object. An equivalence-test is carried out to determine the degree of equivalence between the feature of the object and the template. This comparison-test must be done in every position and for every orientation of the object. In [93], different mathematical approaches are presented to measure the degree of equivalence. In all these approaches, a degree of difference between template and object-feature is measured. If the result is 0 (degree of difference), it means a perfect comparison was measured and if the results are very high, the degree of equivalence between the object and the template is very low.

This comparison is computationally expensive. A suggested solution [93] is to eliminate certain positions where the template and picture is not likely to correspond. Another strategy is to use a failing-tempo measurement of correspondence. A standard equality-test used, is to measure the Euclidian Distance between the image function \( f(x) \) and template \( t(x) \). If the image point \( y \) has perfect correspondence to a point in the template, then the Euclidian distance will be 0. Otherwise it will be greater than 0. The point’s position is then approximately where the point in the model is.

The drawback of template matching approaches is that it is often difficult to select a suitable template from every pattern class and establishing a sufficient comparison criterium. The problem worsens if there is a fierce distortion in patterns belonging to a certain class. However, template-matching is the easiest approach that is available for pattern recognition. The input pattern whose group-classification is unknown, is compared with the template of each class. Comparison is done on a classification criterium that is defined. It is often done beforehand. It is often difficult to establish a suitable criterium.

5.4.2 Edge-Detection.

The logical rational behind edge-detection is that there is a difference between the object and its background. In other words there is a sudden change in grey level, colour or texture at the border between the object and its background on which it appears. In edge-detection, we try to detect these sudden changes in grey level, texture or colour. In this way we
are able to detect the presence of an object in the picture. There are two strategies available to detect these changes.

**Method 1: The first order derivative: The gradient**

We can detect sudden changes in the grey level by applying a differential operator on the given picture. The result of such an operation yields high values where boundaries or edges appear and lower values elsewhere. A single operation that is used for this purpose is the gradient [93]. Sudden changes in the grey level can be detected by applying a derivative operation to the given picture. The result should yield high values where edges are present, and low values elsewhere. If the direction of the edges are unknown, this operation should be isotropic (direction-independent). A simple isotropic derivative operation is the gradient. Assume we have a picture with grey level $f(x,y)$ at point $(x,y)$. The gradient is then given as a vector-valued function with its magnitude given by:

$$|G[f(x,y)]| = \left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2$$

and its direction given by:

$$D[f(x,y)] = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right)$$

In mainly 2D applications, pictures consists of surfaces whose orientation varies slowly [93,85]. The image of such pictures consists of areas with a slowly changing grey level. The tempo of grey level change is high at the borders between the areas. The use of the above gradient-formula is then used to detect such changes in grey level. A drawback in the use of the gradient as detector is that discontinuities may be present in the illumination of the picture. This means that we will be presented with a border that is discontinuous. If the gradient method is used in such a case, it will not yield good results. The process of edge-detection starts with a point that is assumed to be on the border. The border can then be extended in the contour-direction that is determined by directional indicator mentioned above. Martelli (1976) [85] suggested a representation model for edge detection which will briefly be discussed here:

1. Assume that we have represented an edge up to a point $x_i$. Move to a point $x_j$ in a direction perpendicular on $x_i$'s gradient. Apply a gradient-operator on $x_j$. If the magnitude of $x_j$ is greater than a certain threshold, we can add $x_j$ to the border.
OTHERWISE:

(2) Calculate the average of the grey level of the 3*3 neighbourhood-ordering of which \( x_i \) is the centre. Compare this average with a well-chosen threshold, and then determine if \( x_j \) must be inside or outside the object.

(3) Make another attempt at a point \( x_k \) alongside \( x_i \) in a direction perpendicular on \( x_i \)'s gradient plus or minus depending on the result of (2).

Thus, an edge can be detected through the first order derivative method by calculating the maximum difference between the grey levels of pixel-pairs that are neighbours of a certain pixel \( p \). From the above discussion it is clear that the first-order derivative method is a local operation because pixels are tested individually for inclusion in the object-boundary. Further problems occur with this method when the object is blurred. This will result in discontinuities in the border. Any first-order derivative strategy will not know whether a certain pixel belongs to an object, the object-boundary or outside the object (belonging to the background of the image). Blurring occurs when the picture is transformed to an arrangement of pixel intensities. One reason for the blurring of an object may be that there are some precision faults in the hardware that is used. Defects and dirt may also influence the end result of the image being formed. When the image is blurred, the pixels belonging to the noise and dirt mix with the pixels of the real image. The suggested solution is to measure the difference in gradient values between each pixel-pair that is separated by two pixel-unit lengths. In this way, we can measure more accurately, which pixels belong to the picture and which belong to the dirt and noise. We adopt this approach because of the problems we encounter with single pixels intensities in terms of noise. The problem with noise is that single-pixel intensity differences is very sensitive towards it and edges. The solution in this regard, is to calculate intensity averages in the grid and then to measure sharp differences in the calculated averages.

**Method 2: The Second Order Derivative**

The second order derivative is used as a detector of changes in the grey levels of regions. This technique involves taking the sum of the differences in grey levels in the x-direction and also the y-direction. Assume again that we have a picture \( f(x,y) \) where \((x,y)\) are coordinates in the picture. Then the LAPLACIAN operator is given by:

\[
L[f(x,y)] = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}
\]

Laplacian-operators do not react as strongly with edges as gradient-operators. The sum of differences of grey levels in the x and y-directions are computed in the neighbourhood of a given pixel. The operator yields positive values on one side of the edge and negative on the other side. The zero-crossing (between the one side of the edge and the other) define the edge's location. An advantage of this method is that closed curves can be defined in the image through the zero-crossings. This makes this technique a more global method,
because edge detection decisions do not need to be made at every pixel but can be made according to the average contrast around a curve [103, 104]. A drawback of this method [93] is that we still need the first-order derivatives to determine the contrast of the border. Another advantage [103] that this method presents is that the effect of noise can be reduced by using a Laplacian-operator that is based on differences of averages. The effects of noise can also be reduced by using a reduced-resolution version of the image.

5.4.3 Other approaches to edge detection: Variations in the gradient magnitude approach:

(a) Local maximum selection:

The edge is located to the nearest pixel by finding local maxima of the gradient magnitude. This approach is usually chosen when the image is blurred because of some digitization. Pixels located on the border of two different regions, have grey levels having values somewhere between the grey levels of the pixels in the interior of the region and those pixels belonging to the exterior of the region.

(b) Differences of Averages:

Noise in the image yields high gradient magnitude values [103]. To reduce the effects of noise, it is suggested that rather than computing first derivatives of single-pixel grey levels to obtain the gradient, we compute the differences of local average grey levels. The only drawback is that the image is usually blurred. A solution can be to compute averages in the direction that is perpendicular to the edge that we are looking for.

(c) Polynomial Fitting:

This method comprises the fitting of a polynomial surface to the grey levels in a certain neighbourhood of the given pixel. We use the gradient of this surface as an estimation of the gradient of the grey level.

5.4.4 Mathematical tools that can be used to analyse properties of objects:

When an object has been successfully detected by using any of the methods mentioned in the previous section, we can derive useful information concerning the detected object [103], [93, 104]. We must know how the object differs from its background. In this case we can extract certain properties of the object and measure them.

(a) Moments:

Moments give us information about the global spatial distribution of grey levels in the image [103,104,93,85]. The formal definition of the moment is given in [103,104]. Definitions of the first-order moment and second-order moment is given in [93,103]. Suppose that an object has generally higher grey levels than the background. The position of the object in the region or area R, and of its extent in various directions, are found by computing moments of the picture’s grey levels f(x,y) over the region or area.
The (i,j) moment of \( f(x,y) \) over the region or area \( R \) is defined as:

\[
m_{ij} = \int_{R} x^i y^j f(x,y) \, dx \, dy
\]

The coordinates \((\bar{x}, \bar{y})\) of the centroid of \( R \) (its "centre of gravity" if grey levels are thought of as mass) are \( \bar{x} = m_{10}/m_{00} \) and \( \bar{y} = m_{01}/m_{00} \).

If a coordinate system is used with \((\bar{x}, \bar{y})\) as the origin, the the moments \( m_{ij} \) are called central moments in this system and they indicate how the grey levels in \( R \) are distributed relative to the centroid. The moments of inertia of \( R \) around vertical and horizontal lines through the centroid are given by \( m_{20} \) and \( m_{02} \), respectively. If \( m_{20} \) is greater than \( m_{02} \), the object is likely to be horizontally elongated because the grey levels are more spread out horizontally than vertically. The asymmetry of the object about the vertical and horizontal lines can be measured by the magnitudes of \( m_{30} \) and \( m_{03} \) which are zero if there is perfect symmetry.

The central moments are given as

\[
\mu_{pq} = \sum_{m} \sum_{n} (m-\bar{x})^p (n-\bar{y})^q f(m,n) \text{ where } p,q = 0,1,2,\ldots
\]

Unlike \( m_{ij} \), \( \mu_{pq} \) are invariant to shifts of the image signal in the \((m,n)\) coordinate system. If the central moments are normalized such that

\[
\eta_{pq} = \mu_{pq} / \mu_{00}^{\delta+1} \text{ where } \delta = (p+q)/2 + 1 \text{ for } p + q = 2,3,\ldots
\]

a set of seven moments can be defined which are invariant to shift, rotation and scale changes.

Moments can be used to extract the properties of a blob, spot or smudge. The spot, however, must be fully contained in the frame of reference otherwise properties at the side of the frame of reference cannot be measured and cannot be extracted for analysis. In general, moments indicate how the grey levels are distributed relative to the centroid which forms the centre of mass of the area under analysis.

(b) The Main-Axis method:

The main axis and moments of a set of points, determine the direction and amount of points distribution of the object in space. If a set of possible weighted points is translated so that its centre of mass is located at the origin, a symmetry matrix \( M \) can be calculated. The eigenvalues of \( M \) indicate the amount of distribution while the eigenvectors indicate the distribution of points in orthogonal directions.
(c) Fourier Coefficients:

Another method that can be used to extract information about the distribution of grey levels in an area is to determine the Fourier coefficients of a point $(x,y)$ in the area namely:

\[
\iint f(x,y) \cdot \sin (ux + vy) \, dx \, dy
\]

also:

\[
\iint f(x,y) \cdot \cos (ux + vy) \, dx \, dy
\]

If the specific coefficient is high, the grey levels will be periodically distributed with a specific period phase and orientation [93,104]. The coefficients can be arranged in an array with the distance from the origin corresponding to the spatial frequency and a direction which correspond to the orientation of the area. Extreme values in this array, indicate the periodicity of the grey levels in the area. In other words there are values which are extreme followed by lower values followed by another extreme value. This would mean that the grey levels in the area are more periodically distributed in the area. A drop in sizes of the coefficients in a direction away from the origin, indicates a slowness in the fluctuations of the grey level. If the image is harsh, the rate of drop in the sizes of the coefficients is fast (away from the origin) while for a "busy" image, the tempo of drop is slow because the grey level distribution is more evenly throughout the image. If the rate of drop varies in relation with direction, it indicates that the grey level is more directionally distributed. In other words, if the tempo varies in relation to the direction, it indicates the directionality of the fluctuations in coefficients.

(d) Second order grey level statistics:

This is another method for obtaining information on the distribution of grey levels in an image. The basic rationale is as follows: For a certain specific distance $\delta$, we count how many times each pair of grey levels $(i,j)$ appears in a relative position in this distance $\delta$. If this is done repeatedly, it results in the forming of a co-appearance matrix say $A$. Each entry of $A$ is a score-indication of the number of times a specific grey level pair appears in the area under analysis.

If it is found that the concentration of scores is near, the main diagonal of $A$ (these scores are relative to $\delta$), it will indicate that the image is harsh while scores away from the main diagonal indicates a "busy" image.
(e) Geometrical Normalization

Discussion:

The moments and product of inertia of an area $R$ are given as

$$\int I_x = y^2 \, dR, \quad I_y = \int x^2 \, dR \quad \text{and} \quad I_{xy} = \int xy \, dR$$

Let $x, y$ be a pair of orthogonal axes about which the moments of inertia $I_x$ and $I_y$ and the product of inertia $I_{xy}$ are known for the area shown in figure 3. Consider a counterclockwise rotation of the axes through an angle $\Theta$ and denote the new axes as $x'$ and $y'$. The coordinates $(x, y)$ of an infinitesimal area $dR$ of the planar section can then be transformed to $(x', y')$ and written as

$$x' = x \cos \Theta + y \sin \Theta \quad \text{and} \quad y' = y \cos \Theta - x \sin \Theta$$

Then beginning with an expression analogous to the equations of $I_x, I_y, I_{xy}$ above:

$$I_{x'} = \int (y \cos \Theta - x \sin \Theta)^2 \, dR$$

which results in (after manipulation):

$$I_{x'} = I_x + I_y/2 + [I_x - I_y/2] \cos 2\Theta - (-I_{xy}) \sin 2\Theta$$

Similarly:

$$I_{y'} = I_x + I_y/2 - [I_x - I_y/2] \cos 2\Theta - (-I_{xy}) \sin 2\Theta$$

and

$$I_{x'y'} = I_x - I_y/2 \sin 2\Theta + I_{xy} \cos 2\Theta$$

These equations relate the moments and product of inertia of areas in the new $x'y'$-coordinate system to the initial ones in $xy$-coordinates for rotation of the axis through an angle of $\Theta$. 

FIGURE 3. Rotation of axes
The previous methods of obtaining grey level information from the image for example moments and projections, are all dependent on the object's position and orientation in the area. That means that if the position and orientation changed, we would get new moment-values when we computed it for this changed position and orientation. In this section we'll discuss a method that is invariant to changes in the object's position or orientation.

The author states in [103] that this can be achieved by using suitable combinations of moments. The central moment mij is invariant to changes in the position of the object (translation changes), while the moment of inertia around the centroid of an area R is invariant to rotational changes.

Further suggestions are made in [93] that states that if a coordinate system is implemented which has its origin at the centre of mass (centroid) of the image (the system is standardized), this will yield invariance to translational-changes. To obtain invariance to rotation we must find the line that is specified by y = x tan Θ, going through the origin. The moment of inertia, m20 + m20, will be around the line specified by y = x tan Θ. We then solve Θ so that:

$$\sum (x \sin \Theta - y \cos \Theta)^2 \cdot f(x, y) \, dx \, dy = m_{20} \sin 2\Theta - 2m_{11} \sin \Theta \cos \Theta + m_{02} \cos 2\Theta$$

The moment of inertia will be a minimum around the line y = x tan Θ. The line formed by y = x tan Θ is also called the main-axis of the area R. (By definition [262], the principal axes or main axes are those about which the rectangular moment of inertia is a maximum or a minimum. Such axes are always mutually perpendicular. The product of inertia, defined by Ixy vanishes for the main axes.) When we have computed Θ so that the equation above yields a minimum result, we can choose a rotating coordinate system such that this line would be vertical (say). The object will then be elongated in the y-direction. Other forms of normalization that can be used are the autocorrelation and power spectrum of f. Both these methods are invariant to translation of the image f. In figure 4, the area has a moment of inertia I0 = \int y^2 \, dA around the horizontal axis passing through its own centroid. For I0, y is measured from the centroidal axis. The moment of inertia of the same area around another horizontal axis z-z is given as

$$I_{zz} = \int (d + y)^2 \, dA$$

where y is measured from the axis through the centroid. Squaring the quantities in parentheses and placing the constants outside the integral signs gives

$$I_{zz} = Ad^2 + 2d \int y \, dA + I_0$$

Since the axis from which y is measured passes through the centroid of the area \( \int y \, dA = 0 \) and \( I_{zz} = I_0 + Ad^2 \).

This is generally known as the main-axis theorem: The moment of inertia of an area around any axis is equal to the moment of inertia of the same area around a parallel axis passing through the area's centroid, plus the product of the same area and the square of the distance between the two axes.
In figure 5, an example of the effect geometrical normalization has on an image is depicted. The input picture in (a) has a grey level range of 0 to 63. In (b) the grey level range has been set from 0 to 32 for the purpose of noise removal. The grey level of the background points have been set to zero. In (c), (b) was normalized so that its principal (or main) axis was vertical.

(f) Projections and cross-sections:

Projections of \( f(x,y) \) can be used in various directions to get useful information about the grey level distribution in an area \( R \). For example, the projections of \( f \) in the \( x \) and \( y \) directions are respectively given by:
\[ \int_{R} f(x,y) \, dx \]

and

\[ \int_{R} f(x,y) \, dy \]

If the projections are done in an effective amount of directions, it will be enough to reconstruct a picture. Numerical properties like their Fourier coefficients and moments can be used as object descriptors [93].

Cross-sections of \( f(x,y) \) present more detailed information concerning the arrangement of grey levels in \( R \). An object with a high grey level will have peaks in cross-sections that will correspond with certain object-parts. For example, the cross-sections of \( f(x,y) \) in the direction of \( x \) are just the functions \( f(x,y) \) for particular values of \( y \) while the value of \( x \) remains constant. Peaks in the cross-sections correspond to object-parts.

5.5 OBJECT EXTRACTION

In the previous section we looked at techniques which could detect the presence of an object on a background. The object was not explicitly removed from the background. In this section, our aim is to look at techniques available for physically extracting the image from the background and reconstructing it somewhere else. We are immediately faced with a decision-problem:

1. What points are part of the image?
2. What points are part of the background?

The result of this decision can be represented in the form of a picture. This picture may have two possible values: The picture has the value 1 for points that belong to the object and the value of 0 for points belonging to the background. Various techniques exist for extracting an image from its background. We will now discuss these techniques briefly:

5.5.1 Thresholding:

We use this technique where the grey level range of the objects is distinctively higher than that of the background [103]. The assumption that is made in this regard is that the image will consist of areas that have roughly uniform grey levels. The thresholding-technique
involves that for a certain value \( \Theta \) we can construct another image \( \delta(x,y) \) from the original image \( f(x,y) \) and it is performed under the following conditions:

\[
\delta(x,y) = 1 \text{ if } f(x,y) \geq \Theta \text{ and } 0 \text{ otherwise}
\]

\( f(x,y) \) is the grey level of the original image at coordinates \( (x,y) \). The above assumption means that a picture will consist of surfaces that will have orientation and reflection and is also constantly illuminated. Problems do occur with this assumption in that some surfaces are curved and thus will not be constantly illuminated and grey levels will not be constant in the image. Under this assumption, partial thresholding can be implemented. The information of the object's grey levels can be retained while still being able to separate them from the background. Partial thresholding can be mathematically described as follows:

We create a new picture say \( s(x,y) \) where:

\[
s(x,y) = \begin{cases} 
 f(x,y) & \text{if } f(x,y) > \Theta \\
 0 & \text{otherwise}
\end{cases}
\]

This technique is usually applied in the construction of histogram [93,103,104,110]. The histogram is a graphical representation of the different grey levels that is present in the image. On the one axis of the graph the different grey levels is presented in the image and on the other axis, the amount of pixels belonging to a certain grey level. The graph will then be plotted accordingly.

Usually the grey levels of areas will produce peaks separated by valleys of grey levels. These valleys of grey levels represent grey levels that was not strongly present in the image. To obtain a suitable value for the threshold, we must examine the histogram. In a histogram we can determine the minimum value that separates two peaks in the histogram. The threshold will then be assigned that minimum value. When a histogram has two clearly separated peaks, it means that two distinct separate areas are present in the image. Such a histogram is usually referred to as being bi-modal [103,104].

If an object does not distinctly differ from its background, we won't be able to reconstruct the object successfully from its original version via thresholding. A suggested solution for this problem is to transform the original image and to have a distinct set of grey levels so that the background and object have different grey levels. One example of such a transformation is to blur the image. We will get an image that is darker than its background.

Other problems that occur with the thresholding-method are as follows:

1. If the image contains noise, the reconstructed image will also contain noise.

Solutions:
- Smoothing the image before or after thresholding.
Smoothing the image before or after thresholding.

Adapt the value of the threshold in order to minimize the 'business' of the resultant image.

(2) Peaks in the histogram may be too small to be detected.

Solutions:

- Apply local thresholding.
- Partition the image into blocks so that each block will have roughly the same grey level and thus obey the constancy assumption.

(3) Peaks are not easily detected.

Solution:

- Suppress pixels that have a high gradient from the histogram. Peaks that are detectable will now be produced.

(4) Illumination that varies, produces pixel intensities that are more distributed. This results in varying graylevels in the image:

Solutions:

- Interpolate the thresholds so as to obtain changing values of variable thresholds. We can then use this varying values, to classify pixels in the rest of the image.
- Otherwise, we can also partition the image into block areas where each area obeys the constancy assumption.

5.5.2 Region-growing

5.5.2.1 Discussion

In the previous section we described the technique of thresholding in order to physically extract the image from its background. The weakness of this method was that it was assumed that the grey levels of the different areas in the image, was roughly the same. Various problems are encountered if we try to obey this assumption.

Thus we are in need for a more flexible technique that does not lean so heavily on the fixed assumption of constant grey levels. The technique of region-growing is such a technique. The basic approach of this technique is that we add successive points of the area or add successive sub-areas of an image to already reconstructed sub-areas of the image. This approach enables us to alter the acceptance-criteria as the "growing" process develops.
A simple method whereby areas are grown is the technique where we use a high threshold-value to form the kernels of objects. After the kernel is formed, we use a lower threshold-value to add points to this already formed kernel. The grey levels of these points must be greater than some specified threshold-value in order for them to be added to the object that is being grown.

The criteria that we use in this case may depend on the object's shape or texture. If we want to extract an object with a simple shape and texture and which has roughly constant grey levels, we can start by forming object kernels. The kernels are coupled components with constant grey levels. We can add two components lying alongside each other, if it will result in a more compact shape. The grey levels of these areas must be roughly the same. We keep on repeating this operation if the results yield a more compact shape.

Another approach to the segmentation of an object is to make some standard partition of the picture (for example a square-reference). We divide an image into quadrants if the grey levels of the image vary considerably. We add neighbouring quadrants if their grey levels are roughly the same. This process is usually referred to as splitting-and-merging [103,104,85,93]. The results of this operation is unions of squares which have roughly the same grey levels.

5.5.2.2 Region-growing algorithms

- **Blob-colouring.** Local Technique. This technique is a simple region-growing approach [103]. Assume a given binary image has four connected blobs of 1's on the background and 0's on the inside. A label is assigned to each blob. Scan the image from left to right and top to bottom with a special L-shaped template. After one full scan of the image, the colour equivalences can ensure that each blob has only one colour.

- **Thresholding.** Global Technique. This technique has already been discussed. This technique is very useful in pyramid refinement of grey levels. Each level in the pyramid represents the image better than the previous level.

- **Splitting and Merging.** Global technique. Given a set of areas $R_k$, where $k = 1,2,\ldots, m$, [85,103, 104]. The partitioned area $R_1, \ldots, R_k$ whose grey levels are roughly the same, is united.

- **Tracking.** Local Technique. This is a special region growing technique that starts at a point situated on a curve or edge. Successive points are then added to the curve or edge until the curve or edge has been successively formed. Acceptance criteria can again be used for forming the edge or curve. For example, we can look for neighbouring points that have a high contrast under the constraint that the resultant curve's bending will be small. If, however, we were to find during this process of edge or curve formation, that the bending of the curve has been more than was allowed under the bending-constraint, we can do backtracking to previous points that are already part of the edge or curve, and recompute the next point that must be part of the curve. Thus, we can backtrack to the point just before the point where the extreme bending of the curve starts and re-compute the next point that will form part of the curve.

**Comment on splitting-and-merging**
To make the algorithm of splitting-and-merging more effective, the image's pixels must be organised in the form of a pyramid structure [103]. This structure consists of areas, and the areas are ordered into groups of four each. Any area can also be divided up into four subareas (except for an area consisting of only one pixel). Each group of pixels can be merged into a single larger area. The algorithm for implementing this structure is given as a high-level disruption [85]:

(1) Choose any rasterstructure and homogeneous property say H. If it is valid that for any region R in this structure, \( H(R) = \text{FALSE} \), we must divide this area into four sub-areas. If for any four suitable areas \( R_{k1}, ..., R_{k4} \) it is found that \( H(R_{k1} \cup R_{k2} \cup R_{k3} \cup R_{k4}) = \text{TRUE} \), we merge them into a single area. If there are no more areas that can be merged, we stop the process.

(2) If there are any neighbouring areas \( R_i \) and \( R_j \) so that \( H(R_i \cup R_j) = \text{TRUE} \), merge \( R_i \) and \( R_j \).

The above mentioned technique is also used in cluster analysis [80,93,103,115]. Cluster analysis revolves around the idea that a set of sample points having some degree of complexity is usually formed by discernible subsets and then by locating and describing these clusters of points provides the description of the datastructure which can be used in the solving of a pattern recognition problem. This can be interpreted as follows:

(1) The samples correspond to samples of one class and the clusters correspond to subclasses of that class.

(2) The samples have no assigned class membership. The intention of cluster analysis is then to partition all the samples in a pattern into smaller groups so that a simpler pattern classification problem can be solved on each of the smaller samples.

Cluster analysis can form part of a pattern classification algorithm. The analysis may revolve around the idea that the pattern space is divided into subregions so that a classification algorithm may be applied seperately on these regions. The algorithm may be applied to all subareas irrespective of class-membership.

Other techniques are less mathematical in nature.

- **The State - Space Approach**

  A two dimensional starting image, is viewed as a finite state where each point can be seen as an area, distinct from all other points. The state changes as we add boundaries or remove them. The aim here is to find suitable changes in states that will yield the best possible partion:

### 5.6 Properties of Objects that have been Extracted

Once an object has been extracted from the picture, several geometrical properties of the object can be analysed. In this section, we briefly discuss some of these measurable properties. These properties are geometrical in nature and are used for the mathematical
analysys of the image's scene. For a more detailed discussion, the reader is referred to [103].

5.6.1 Connectedness

One of the properties that can be analysed is whether two points in the image are connected or not. If we use this property fully, we can establish if the whole image is connected. Mathematically speaking, we say that two points with coordinates respectively \((x_0, y_0)\) and \((x_n, y_n)\) of \(D\) (the object) is connected if there is a succession of points of \(D\) namely \((x_0, y_0), (x_i, y_i), \ldots, (x_n, y_n)\) so that the point given by the coordinate \((x_k, y_k)\), is a neighbour point of the point \((x_{k-1}, y_{k-1})\) where \(1 = k = n\). Such a succession of points in \(D\) is called a path in \(D\) [93,103,104]. It states briefly, that if \(D\) is elongated, we can decimate (thinning \(D\) out) by removing simple borderpoints of \(D\). Simple points of \(D\) are points that each have more than one neighbourpoint in \(D\). The connectedness of \(D\) is not eliminated and arcs that are already thin, will not be further thinned out. For a discussion on thinning and expansion, see Rosenfeld [93]. A maximal set of mutual connected points in \(D\), is called a connected component of \(D\). If \(D\) has only one connected component, then \(D\) is connected. A border of \(D\) consists of points in \(D\) that are neighbours of points in the compliment of \(D\) namely \(D\). In other words, \(D\) consists of all points outside \(D\). The points that are not part of the border of \(D\) or \(D\) are called the internal points of \(D\).

5.6.2 Size, convexity and compactness

The amount of border points of \(D\), defines the circumference of \(D\). Assume \(A\) is the area of \(D\) where the area is the amount of borderpoints of \(D\). Assume further that \(P\) is the circumference of \(D\). The compactness of \(D\) is usually given by:

\[
D = \frac{A}{P^2}
\]

The span of \(D\) in a certain direction, is the length of it's projection perpendicular to this direction [103].

The diameter of \(D\) is the maximum span in any direction of the largest distance between any two points in \(D\) [103,104,93].

We can now define the convexity of \(D\): \(D\) is convex if its cross-section along any given line consists of one line segment at the most. We call the smallest convex set in \(D\), the convex-shell of \(D\).

5.6.3 Arcs and curves

We say \(D\) is a closed curve in the digital case, if \(D\) is connected to neighbours in \(D\). \(D\) is called a curve, if \(D\) is connected and all points accept two points in \(D\), have two neighbourpoints in \(D\). \(D\) is elongated if its largest span is far greater than its smallest span. This
definition, however, is insufficient and a better definition of elongatedness is suggested by Rosenfeld in [93,104].

5.7 SUMMARY

In this chapter, the various techniques that are used today in image processing, for the formation of images of pictures, were discussed. We started off with the 2-D and 3-D image analysis paradigms that are commonly used today for image analysis. We then discussed techniques based on pixel-intensities, to analyse the properties of regions. In both cases, we discussed various properties that can be measured after successful object detection or extraction, have been accomplished.
CHAPTER 6

ARTIFICIAL INTELLIGENCE AND COMPUTER VISION
6.1 INTRODUCTION

The aim of this chapter is to look very briefly at the impact the field of artificial intelligence may have on the field of computer vision. We depart from the idea that image understanding involves the task of knowledge engineering because interpretation in itself requires knowledge of what is being interpreted. Without this knowledge (which must be available in some way for the interpretation process), interpretation of a scene is impossible. Therefore, it is important that a computer vision system designed for image interpretation, has knowledge of the scene’s content prior to the interpretation process starting. When one wants to discuss knowledge engineering for image interpretation by a computer vision system, one invariably arrives at the idea of a knowledge base being used by the system in order to succeed in an interpretation task. This knowledge base contains knowledge for the successful interpretation of the scene. In this way, the understanding of visual scenes can be done by computer analysis. This is primarily the problem of computer vision [71, 103, 104, 85, 224, 228].

In [93, 103], a graphical representation is given for 2-D and 3-D image analysis. It is also discussed in chapter #5. This paradigm of image analysis can be divided into two parts. The first part of the analysis, is usually referred to as low-level vision and the second part high-level vision. During the low-level vision process, features that describe sensory data are extracted and described in a more abstract way. This description of sensory data usually consists of a segmented image where different regions are labeled in the image according to the pixel intensities that were measured in the image. Segmentation and feature detection are regarded as assigning labels to the image pixels, indicating the class to which the pixels belong. The output of the segmentation process is a symbolic image where pixel-values are labels and not intensities. The labeling of pixels are done according to the satisfaction of a given predicate-rule which dictates the conditions of pixel-membership to a region.

In the high-level vision process, a consistent interpretation to the labels obtained in the low-level vision process, is sought. The interpretation is done by using a priori information about the scene’s domain. Interpretation is usually referred to as model matching: The segmented image obtained during the low-level process is matched against a priori knowledge (the model) of the scene. Thus, interpretation is considered as a mapping between sensory data and the model of the scene. This model is usually represented as concepts relevant to the image’s domain [228]. The process of interpretation is not easy because interpretation is not absolute: uncertainties arise when formulating hypotheses based on the data or the model. Therefore computer vision interpretation overlaps significantly with the subfield of AI called knowledge-based systems. When knowledge-engineering is applied to computer vision for the purpose of image interpretation, major computational issues must be addressed:

1) As discussed in chapter #2, neuroscience and cognitive science can only give clues to early organic vision processing [116]. There are no experts available that can perform introspection to explain their own vision knowledge. It is not possible to understand the mechanisms of organic vision through introspection because of the nature and number of neuronic components used in the process.
2) If a small knowledge base is used, the computer vision system can afford to index object-knowledge item on the bases of the extracted knowledge. There are only a few indexes of object-knowledge items in a small knowledge base. The larger the amount of object-knowledge indexes becomes in the knowledge base, the more difficult it becomes to effectively index into specific object-knowledge item.

3) There are uncertainties present in the visual data being used in the task of visual processing. For example, visual data is locally ambiguous. This feature makes it often difficult to recognise an image when the surrounding context is not available for consideration. Noise is also present in the sensors and other hardware items being used. Ambiguity can also result in the transformation of the three dimensional scene into a two-dimensional image. Occlusion, lighting variation, points-of-view and so on can all effect the two-dimensional appearance of an object.

4) The amount of low-level data-volumes involved in vision processing, is quite substantial. Processing of colour images (RGB [85]) at a moderate resolution of 512 x 512 and at a rate of 30 frames per second, would require over 20 million bytes of memory space in order to process the visual data. When a single image operator such as a convolution operator in a 3x3 pixel neighbourhood [85] is used, over 500 million arithmetic instructions are performed per second by the single operator.

When considering knowledge-based vision processing systems, three fundamental issues must be addressed [228]:

- **What knowledge must be used?** [228]
- **How is this knowledge represented?**
- **How is this knowledge used?**

In this chapter we intend briefly to discuss the various techniques that are used to represent visual knowledge of scenes. We also intend to look at the various ways that exist to control the knowledge that is used in the interpretation task. In the last section we briefly treat various systems that have been developed over the years for the task of image interpretation. We end this chapter by a conclusion section in which we try to identify the future of AI in the computer vision task.

### 6.2 WHAT KNOWLEDGE MUST BE USED?

#### 6.2.1 Physical knowledge

Important knowledge in this regard would be the physical laws governing imaging processes in the real-world environment. Geometry of illumination sources, objects, the camera and spectral properties of the light source are also knowledge sources. Also, information concerning shading, texture, motion, stereo vision can be used in combination with this knowledge to obtain (recover) the 3-D shape from the 2-D image. [228].


6.2.2 Visual-Perception Knowledge

The Gestalt laws which were discussed in chapter #2, can be used for the grouping of primitive pictorial entities into more global entities. For example, this knowledge plays an important part in the grouping of 3-D features into global characteristics of an object.

6.2.3 Semantic Knowledge

For recognition of objects, knowledge about properties and their relations play a very important role in achieving this goal. Semantic knowledge is usually referred to as being domain-specific in that specific knowledge about specific entities is represented and that a generalization of this knowledge to all objects is not possible.

6.3 KNOWLEDGE REPRESENTATION

When we consider knowledge-based systems for the task of image interpretation, we find ourselves in the area of AI which is concerned with the representation of and usage of knowledge. A.R. Rao [71] uses Brachman and Smith as a guide to define the terms knowledge, symbol, representation and knowledge representation as being the following:

- Knowledge is a collection of descriptions, procedures and methods (for problem-solving) to perform the tasks of organizing and summarizing observations and to support a degree of problem solving ability, in other words to be able to automate the problem solving process of a certain degree.

- A symbol represent ("stands for") something else. This "something else" is found by reason of association.

- A representation is a specific kind of symbol. The symbol’s structure is perceived as to correspond in some way with the appearance (of structure) of another entity which is represented ("stands for") by the symbol.

- A representation of knowledge is defined as being the combination of datastructures and interpretation procedures.

When considering knowledge-representation for image interpretation, we tend to ask what kinds of knowledge must be encoded for successful image-understanding. Rao [71] states the following different approaches to knowledge representation for image understanding:

- Objects must be represented. This includes different instances, different objectclasses, relations between different objects and also constraints.

- Knowledge of how to perform the tasks must also be encoded. In this case, the concept to feedback is addressed because performance measurement is involved when systems must modify their performance during processing.

- Metaknowledge must also be encoded. This knowledge are defined in AI as being knowledge about knowledge; that is, how "much" do we know; what is the extent and
reliability of our knowledge? Often in AI, this knowledge also guides the use of knowledge which is needed to perform an interpretation task.

6.3.1 Knowledge Representation Schemes

6.3.1.1 Semantic Networks [71, 85]

A semantic net represents information as a set of nodes interconnected by labeled arcs that represent the different relationships among the nodes. These networks have the following features:

1) They all contain a datastructure of nodes which represents concepts. A hierarchy of nodes is constructed by an IS-A property link. In [85], other property links are mentioned. These include the positional relations of LEFT-OF and RIGHT-OF, SUPPORT etc. The basic notion behind the IS-A relation is that of property inheritance [85]. For example IF Fred IS-A camel and a camel IS-A mammal, then Fred is considered as being a mammal. In other words, the basic notion is that of X IS-A Y meaning that X is an element of a set Y. This and other property relations establishes a hierarchy of nodes connected by these relational links.

2) They contain procedures of inference which operate on the specific data structure of nodes. Information is inherited from the top hierarchy levels to the lower levels of the hierarchy along relational property links (for example the IS-A links).

In [85], Ballard and Brown mentions the work they have done in implementing semantic networks to achieve goal-directed image interpretation. They set out to find ships in a dockscene as well as ribs in chest radiograms. Image analysis is done by a mapping between a model and its image. The model contained generic knowledge which included different kinds of knowledge about the domain of the specific image.

6.3.1.2 First-Order (Predicate) Logic

In [71] the use of first-order logic is addressed from the viewpoint of the integrated circuit domain. The authors define simple two-place predicates to define certain visual information. Representing knowledge in first-order logic form has the following advantages and disadvantages [71]:

...
<table>
<thead>
<tr>
<th>ADVANTAGES</th>
<th>DISADVANTAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>It is always precise in terms of interpretation.</td>
<td>Rules cannot be contained or embedded in a hierarchical framework.</td>
</tr>
<tr>
<td>There is no ambiguity in decision making.</td>
<td>This means that we cannot index into relevant knowledge because all rules are the same.</td>
</tr>
<tr>
<td>Rules are structured in a simple if-then manner.</td>
<td>This approach cannot handle uncertainties because it is exhaustive and precise in nature.</td>
</tr>
<tr>
<td>Knowledge can be contained in a collection of such rules.</td>
<td>Imprecise information cannot be handled.</td>
</tr>
<tr>
<td>New facts can be deduced from old facts and added to the database.</td>
<td></td>
</tr>
<tr>
<td>Through precision of existing facts, these new facts are always precise and consequently the database is always consistent.</td>
<td>This means, that incomplete information of a scene will always impair the interpretation process if this knowledge representation scheme is employed.</td>
</tr>
</tbody>
</table>

### 6.3.1.3 Frames

The previous two approaches to knowledge representation seemed limited for knowledge-based vision systems because of certain limitation already outlined above. These limitations come in where information is imprecise. In both semantic networks and first-order logic approaches, imprecise information impair the deductive inference process which is the key feature of these two methods. Representation of visual knowledge should cater for more common knowledge about objects and capture more structured information of objects.

A **frame** is roughly defined as a highly modular chunk [85,288]. A frame is a datastructure which can represent classes of objects. The frame has knowledge attached to it. This knowledge states for example how to use the frame and what to expect after it is subsequently used.

A frame is usually considered as a **slot-and-filler** structure formed by a network of nodes and relations between them. The frame's top levels are fixed and contain facts that are considered to be true. The lower levels have many **slots** that is usually filled with data describing aspects and features of objects. A **filler** must adhere to a certain set of conditions that are associated with each slot. If we perform vision analysis of a scene, different frames can represent different knowledge about the scene. For example frames may describe how an object may look from different viewpoints or how it may look when occluded by other objects or for example how the object looks if it is deformed. Several frames will be needed if we want to describe the appearance of a deformed object.

When a frame is used to represent a situation, a matching process tries to assign values to each frame's slots. The process is usually controlled by current goals and certain information attached to each frame. The process of **matching** involves:
The frame which was selected for use, does a test to confirm its relevance: is it appropriate to be used?

If a node in the frame cannot confirm its default value received, the frame requests certain relevant information to assign values to such a slot.

After the frame is informed of a change in appropriateness control is transferred to the next relevant frame.

In Artificial Intelligence [224], Lenat cites the drawbacks of using frames as knowledge representation scheme as given by Brachman. These drawbacks revolve around:

- prototype case representations with default values which can be changed, does not cater for one type of representation: "Composite descriptions whose meanings are functions of the structure and interrelation of their parts.", meaning that descriptions of parts of objects may depend on how these descriptions are functions of the object's structures and the interrelational nature of the object's parts.

- the IS-A relation is confusing because an entity X may belong to set Y (C IS-A Y IS-A Z) and Y belongs to a greater set Z or a set B can be a member also of the greater set Z (B IS-A Z)The question then arises if X (an entity like a person's name) is also B when B may refer to a set of entity names and the X-entity may perhaps not be a member of the B-set of entities.

A special kind of frame is called a schema [71,85,224]. This refers to frames used in visual context. Related facts can be arranged in a way so as to eliminate redundancy. They can also receive commands on the usage of information contained in it. Interferences based on the interconnections of different nodes, can be made. Most schema-based approaches have been applied to natural language understanding [71,92,243] environments and fewer to image understanding because of the knowledge derived from images which is sometimes error-prone and because it takes considerable procedural flexibility to understand real images.

In [243] a general purpose vision system is proposed consisting of many single object vision systems or schemas. When the process of image interpretation is performed, active schemas cooperate and compete with each other in order to arrive at a certain consistent interpretation of the image. Each schema is an expert at recognizing an object. Together they recognize the scene. There is one schema for each class of objects. A schema instance activates an instance of the object-class. Schema instances communicate to arrive at a consistent interpretation of the scene. In this approach, schema instances communicate through a global structure called a blackboard. On this blackboard, a schema can publish its own contributing knowledge to the partial interpretation process and also gain access to any other already published knowledge of other schemas. Each schema instance posts information on the blackboard about what he considers are possible members of its object-class. This is done in the form of an object-hypothesis. Other instances read this hypothesis and can compute relations between objects. Other features which are reported in [243] concerning the schema-blackboard approach are:
Each schema instance has *internal* hypotheses about possible instances of its object-class in the scene, which it develops and maintains. These hypotheses are visible only to a single schema instance. For this reason then, each schema has its own local blackboard. This blackboard is only accessible to the schema instance’s own strategies. *Internal* hypotheses consist of *tokens* representing images and endorsements representing object-evidence on the presence or absence of an object. *Tokens* can represent any type of image event in the problem space or domain of the schema. One can get *low-level* tokens which are formed directly from the pixel information, while more *symbolic* tokens or *abstract* tokens are formed when low-level tokens are grouped into more complex entities. *Endorsements* are used by a schema as a reflection of determining what rules have been used and how successful they had been. In totality, the set of all possible *endorsements* represents the problem space or domain of the schema. Thus, the schema reasons through these object-specific endorsements contained in its own local scope of environment. It is also possible for a schema to handle uncertainty. Through *endorsements*, a schema can backtrack or trace the way it has reasoned because endorsements record the sources of information used by the schema.

*Strategies* are simple control programs that run concurrently within each schema. They represent the encoding of the control knowledge which includes the order of knowledge activation and the addition of endorsements based on the results obtained.

A schema instance contains in itself a number of internal hypotheses and each of them and each of them must be available to its currently active strategies. These internal hypotheses must also not be visible to other schema instances. Thus, each schema instance contains its own *local* blackboard. The blackboard is accessible to all strategies making up a single schema instance.

The goal of this approach, is to provide the basis of a general purpose vision system through this experiment in knowledge-directed computer vision. The system was designed to run on a parallel processor. It was important to distribute the vision computation task as much as possible to avoid bottlenecks in communication. The drawback of the system is that the effectiveness of this proposed vision system cannot as yet be quantified. This is because the system’s parallel environment cannot as yet (1988) be specified. Furthermore, the system’s knowledge base contains only 15 template objects. The authors stress the importance of the into the expansion and maintenance of larger knowledge bases containing up to 100 template objects.

### 6.2.1.4 Production Systems

Production systems have perhaps become the most popular knowledge representation method for *expert systems*. A rule-bases system, contains antecedant consequent pairs. These pairs are referred to as *production rules*. A database is usually constructed which is made up of these type of rules. The database contains all known (true) facts. Usually, on interpreter matches the antecedant with the database, selecting the specific rule that is appropriate for usage. ("to fire").

---

**CHAPTER 6**

**PAGE 6-7**
The advantages of the use of production-systems is that:

- **Knowledge** can be represented in a modular way, allowing greater flexibility in the representation of knowledge.

- Adding or deleting certain rules will not effect the other existing rules. Thus, maintenance of the database is possible.

- **Uncertainty** can be allowed by the usage of some degree of statistical reliability associated with each rule. Conclusions can then be reached based on the strength of this statistical reliability. Usually, probabilities are assigned to each rule to implement this feature.

- "Rules of thumb" or **heuristic knowledge** can be represented by a collection of these rules.

The disadvantages of using production systems is [71, 223]:

- Rule-modules may be in conflict with each other. This happens when antecedant-parts of certain rules are satisfiable concurrently and the interpreter is left with the decision as to what rule is appropriate to fire. The use of meta-knowledge (knowledge about knowledge) usually solves this problem but then again it can be asked what knowledge controls meta-knowledge?

6.4 CONTROL IN THE USE OF THE KNOWLEDGE

6.4.1 Parallel vs sequential control [71]:

Control strategy must be essentially sequential because understanding is sequential and cyclic involving a hypothesize-test approach at different stages [71].

6.4.2 Top-down vs bottom up control

The consensus is that low-level processing (segmentation-processing) must follow a bottom-up approach and that feedback from higher levels of the processing hierarchy follow a top-down manner of control to the lower levels of processing.

6.4.3 Local and Global control

Local processors operate in parallel but are guided globally. Global control will follow a sequential approach based on the result of the hypothesis that was tested.

6.4.4 Distributed and central control

There are two interacting phases in this approach namely segmentation done partially and interpretation. Partial segmentation is data-driven and forms the first analysis stage. Algorithms usually employed for this analysis are for example edge detection, region growing, splitting and merging and so on. The system uses these partially segmented image-results for the task of image-interpretation. The result of the interpretation will
cause an improvement in segmentation and the process follows in a cyclic (two-directional) manner. In [71], an interpretation system is discussed called Acronym. It is a domain independent model-driven interpretation system where model based vision is divided into four parts: modeling, prediction, description and interpretation. It features general geometric reasoning. This feature is also evident in many of the system applications developed to date. Some of these systems will be discussed in the next section.

6.4.5 Goal Achievement and Belief Maintenance

These two processes are high-level processes that feature different control styles. Belief maintenance is concerned with the upholding of a current state or fact, while goal achievement seeks to uphold a set of future (already "planned") states [243,85,71,223].

6.4.5.1 Goal Achievement

This control strategy consists of two phases namely planning and execution [85]. In the planning phase, the world is designed in a simulation to generate in this simulation, a plan. This plan is a sequence of actions that will or should achieve some goal when carried out. Planning is a useful computational mechanism that doesn’t need cognitive characteristic behaviour. The question which arises in the planning-phase is the question of choosing a specific plan to achieve a goal: Given a set of plans, one must find a way of choosing between them. The usual solution is to have a method of scoring plans based on a measurement of their effectiveness. Goal achievement has received important attention in the field of robotic navigation. As discussed in chapter 7, a robotic transport vehicle sets out on a route, travelling from a certain point to another. The task of route-planning and eventual goal achievement is done beforehand by a planner. The vehicle then sets out on this planned route. During the journey, the plan may have to be altered due to new information that was received by the robotic vision system. For example, an obstacle may block the path that was initially planned. Thus, replanning the route that will achieve the end-goal is done [68,69,66,67].

6.4.5.2 Belief Maintenance

Briefly stated, belief maintenance is concerned with the process of accepting certain results supported by a priori knowledge of the results. In computer vision, constraint satisfaction mechanisms are computationally suited for the upholding of belief structures. These mechanisms can operate in parallel, seeking the minimization of inconsistency in received data. Furthermore, they can tolerate noise in the received data-inputs. Constraint satisfaction is usually applied during the low-level vision process where local parameters containing noise are combined into globally consistent images.
6.5 SYSTEM-APPLICATIONS

INTRODUCTION:

In this section, a very general overview is presented in tabled form of some of the systems existing today which perform an intelligent operation or operations to solve a specific vision interpretation problem.

<table>
<thead>
<tr>
<th>AUTHOR(s)</th>
<th>REFERENCE</th>
<th>DESCRIPTION OF TASK/SYSTEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) R J Woodham &quot;Analyzing Images of Curved Surfaces&quot;</td>
<td>Artificial Intelligence 17(1981), pp. 117-140, [244]</td>
<td>An algorithm is described which is used to derive information that is contained in image intensities about an object-scene. A matrix is introduced that contains properties of surface curvature. Image analysis is simplified because surface curvature properties are also properties of this matrix.</td>
</tr>
<tr>
<td>(b) M Barry et al. &quot;A Multi-level Geometric Reasoning system for vision&quot;</td>
<td>Artificial Intelligence 37, 1988, pp. 291-332, [245]</td>
<td>An approach is made towards model-based vision in which geometric and algebraic reasoning is used explicitly. A system based on this approach is being developed. (1988). The system has 3 components namely a hierarchical organization of geometrical knowledge, labelling algorithms and algebraic reasoning algorithms. The system will be used for the establishment of a 3-D model from 2-D images; as well as 2-D Image-to-Image matching.</td>
</tr>
<tr>
<td>(c) J S B Mitchell &quot;An Algorithmic Approach to some problems in terrain navigation&quot;</td>
<td>Artificial Intelligence 37, 1988, pp. 171-201, [246]</td>
<td>The path planning problem is examined. A &quot;map&quot; of the region terrain is given; an optimal path from one point to another must be found. The formulation of some path planning problem given and suggested algorithms to solve certain cases are given.</td>
</tr>
</tbody>
</table>
(d) E L Walker and M Herman
"Geometric reasoning for constructing 3-D scene descriptions from images"

Artificial Intelligence, 37, 1988, pp. 275-290, [247]

A system called MOSAIC is described. This system uses domain specific knowledge to drive the geometric reasoning needed to produce 3-D models for complex real-world scenes. The system can be improved in terms of robustness, extensibility and flexibility by representing the domain knowledge explicitly using the knowledge of prediction and verification. A frame-based system is being developed to achieve this goal.

(e) Chia-Hoang Lee
"Interpreting image curve from multi-frames"

Artificial Intelligence 35, 1988, pp. 145-163, [248]

A method of reconstructing the structure from multi-frames is given. A necessary and sufficient condition concerning the number of frames for determining the motion with only two feature points observable, are established.

(f) A Witken et al.
"Constraints on deformable models: recovering 3-D shape and non-rigid motion."

Artificial Intelligence 36, 1988, pp. 91-123, [249]

The problem of inferring the 3-D structures of non-rigidly moving objects from images is addressed. The authors' solution is the use of dynamic, elastically deformable object models that offer geometric flexibility to satisfy a diverse amount of real-world constraints. These models are specialized to inducing a preference for axi-symmetry. The recovery of 3-D shape and non-rigid motion from natural images, is described.
<table>
<thead>
<tr>
<th>References</th>
<th>Journal</th>
<th>Year</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>(g) K Ikeuch and B K P Hom</td>
<td>Artificial Intelligence 17, 1981, pp. 141-184, [250]</td>
<td>The authors propose an iterative method for computing shape from shading by using occluding boundary information. Instead of using the gradient space to express the orientation of surface patches, they use the stereographical plane because this plane makes it possible to incorporate occluding boundary information. From numerical experiments it is shown that this method is effective and robust. Scanning electron microscope pictures using their method, is analyzed.</td>
<td></td>
</tr>
<tr>
<td>(h) S Draper</td>
<td>Artificial Intelligence 17, (1981), pp. 461-508, [251]</td>
<td>This is a revision paper concerning the application of the gradient space and dual space in programs that interpret line-drawings and examines whether it can be used as a basis for a fully adequate program. Various other authors's work is analyzed.</td>
<td></td>
</tr>
<tr>
<td>(i) B P K Horn and B G Schunk</td>
<td>Artificial Intelligence 17, (1981), pp.185-203, [252]</td>
<td>A method for finding the optical flow pattern in an image is given, which assumes that the velocity of the brightness pattern varies smoothly everywhere in the image. An iterative approach is given which can successfully compute the optical flow for a number of synthetic image sequences. Examples are given where the smoothness assumption is violated as singular points or along lines in the image.</td>
<td></td>
</tr>
<tr>
<td>(j) T O Binford</td>
<td>Artificial Intelligence 17, (1981), pp.205-244, [253]</td>
<td>The paper discusses the generation of effective scene descriptions from images, that is, generating surface descriptions from image boundaries.</td>
<td></td>
</tr>
<tr>
<td>Reference</td>
<td>Source</td>
<td>Text</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>------</td>
<td></td>
</tr>
</tbody>
</table>
| (k) L S Davis and A Rosenfeld  
"Co-operating processes for low-level vision: a survey." | Artificial Intelligence 17, (1981), pp. 245-263, [254] | Various cooperating local parallel processes are discussed that are used for the assignment of numerical or symbolic labels to image or scene parts. Different methods for controlling such processes are also addressed. |
| (l) B E Flinchbaugh and B Chandrasekaran  
"A Theory of spatio-temporal aggregation for vision." | Artificial Intelligence 17, (1981), pp. 387-407, [255] | A theory of spatio-temporal aggregation is proposed to explain the process of grouping together elements in an image sequence, whose positions and motion have consistent interpretation as the retinal projections of coherent clusters of particles in the physical world. Certain assumptions of confluence and adjacency are made in order to constrain the infinity of possible interpretations to a computationally more manageable domain of interpretations. |
| (m) R A Brooks  
"Symbolic reasoning among 3-D models and 2-D images." | Artificial Intelligence 17, (1981), pp. 285-348, [256] | Model-based vision systems are described in terms of four components namely models, prediction of the image features, description of the image features and lastly the interpretation. New approach to prediction and interpretation is given, based on the propagation of symbolic constraints Prediction, description and interpretation proceed concurrently from coarse object sub-parts and class interpretations of images, to fine distinctions among object subclasses and more precise 3-D quantification of objects. |
| (n) H K Nishihara  
"Intensity, visible surface, and volumetric representations". | Artificial Intelligence 17, (1981), pp 265-284, [257] | The paper treats human vision as a computational process: it produces descriptions of the external world from retinal input-images. Some empirical conclusions are made on the nature of these processes based on the problems they solve. |
| (o) T Kanade “Recovery of the Three-dimensional shape of an object from a single view.” | Artificial Intelligence 17, (1981), pp. 409-460, [258] | This paper aims to identify some of the assumptions needed in order to obtain definite ideas about the 3-D shapes of objects when these objects are represented on a 2-D picture plane. These assumptions are geometric assumptions. The paper demonstrates how the theory of and techniques which use such assumptions can provide a shape-recovery method. The method is two-phases
1. The application of the Orgami theory which models the world as a collection of plane surfaces and recovers the possible shapes.
2. The mapping of image regularities into shape constraints for recovering the probable shapes in a quantifiable way. Shape recovery from a single viewpoint is demonstrated for object-scenes such as a box or a chair. The method can recover 3-D shapes of an object contained in a single image. |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(p) Y Shirai</td>
<td>Artificial Intelligence, 4, (1973), [259]</td>
<td>A system is described for the recognition of context-sensitive polyhedra. Object-recognition is done in a step-wise manner where each recognition result dependent on the previous results of recognition. The most simple evaluation is made at each step and tested. This system is effective for scenes containing a few blocks or wedges.</td>
</tr>
<tr>
<td>(q) K Stevens &quot;The visual Interpretation of structure contours.&quot;</td>
<td>Artificial Intelligence 17, (1981), [250]</td>
<td>A surface contour is an image of a curve across a physical surface. Interpretation is done by assuming that the physical curves are restricted to their geometric relationship of the underlying surface.</td>
</tr>
</tbody>
</table>
6.6 CONCLUSIONS

In this chapter we briefly looked at the effect that knowledge engineering may have on the field of computer vision. Knowledge engineering approaches seem to attempt to make image processing "intelligent". This means that the image is not only processed in the usual ways, but that the image is also understood in terms of knowledge corresponding to the segmented image. We have discussed how this knowledge can make image understanding possible by describing the object's features, parts, orientation etc. We have discussed the relational approach through which these description can be connected to each other. It was also evident that the more knowledge a knowledge-base contains, the better it will be prepared for the handling of uncertainties. However, the issue of what knowledge to use during an interpretation-task becomes a decision problem. Usually, meta-knowledge is used to solve this problem, but subsequently, more knowledge will be needed to control the use of meta-knowledge. Thus, a knowledge-based approach to the task of image understanding both have an advantage in that so much knowledge as needed for interpretation can be encoded into the knowledge-base but that in this we also must consider the decision-problem which arises by the encoding of these relative large amounts of knowledge. This is especially true for production-systems where large amounts of rules (usually if-then) are encoded into the knowledge-base.

However, the knowledge-based approach to image understanding still has merit because of the relevance it has concerning biological vision systems. We are able to understand scenes because we possess already stored knowledge of those scenes. The heuristic knowledge which we seem to possess, aids the understanding-task in that we are able to recognize or understand different objects belonging to the same object class. For example, we know that a small hammer and a large hammer remains a hammer. In this regard, heuristic knowledge plays a very important role in knowledge-based systems, in that they are able to give a knowledge-based system some degree of "intelligence."

All the systems which were briefly discussed, uses some kind of knowledge-based approach in the understanding of images. This knowledge may contain facts about geometry-
knowledge like shape-knowledge, \((o, j, g, f, a)\), curve-knowledge \((d, e, m, q)\). Through the use of this knowledge, the systems are able to recognize objects in 2-D space or 3-D space, do construction-planning with 3-D objects, reason about the structure of 3-D objects from given 2-D viewpoints of the object and so on.

It is clear from only the mentioned applications that image understanding is very closely related to the sub-field of knowledge engineering in Artificial Intelligence. It also borders the other field of AI like search, automated reasoning, problem solving and AI-architecture. It seems, that the development of intelligent visual systems will in future be drawn into these AI-fields.

The question which can be asked is why there are no general purpose vision systems available in general. We do know that there are specific systems which can perform certain tasks like inspection and navigation. They will be addressed in chapter #7. These systems are able to define, structure and apply knowledge better relevant to their own task domain. Systems in restricted domains use specific knowledge bearing on the task to obtain results of a certain recognition task. General systems need general knowledge. Image understanding involves the search of large search spaces of possible interpretations. This makes it a computationally expensive task because the data volume is large and because uncertainty exists in the subsets of the data. As already stated, knowledge in computer vision applications includes:

- *domain independent* knowledge about occlusion, perspective, physical support.
- *domain dependent* knowledge: It is knowledge that the system is expected to encounter. For example, the object's 3D appearance and how it will appear on a 2D image as well as the object's spatial relation to other objects.
- *control knowledge* addresses the issue of how to extract, organize and match knowledge from an image and compare it to stored models.

In this chapter, we have seen that the fields of artificial intelligence, image understanding, pattern recognition aspects are unified in computer vision research, largely motivated by a great multitude of applications of which only a few are mentioned in this chapter. The development of a general computer vision system that can approach the capabilities of the human eye-brain system, seems quite remote at present. The objects to be interpreted require a great deal of knowledge about themselves and how they relate to each other. This amount of knowledge is not yet representable in any knowledge-based system.
CHAPTER 7

COMPUTER VISION AND THE INDUSTRIAL ENVIRONMENT
7.1 INTRODUCTION

Machine Vision has developed rapidly over the last ten years. Nine years ago, only six companies were doing meaningful work in image processing. In 1978, the first hardware was presented by OCTEK [1]. Today, there are over 100 companies actively involved in research in the field. [4, 52, 53, 55, 60, 63, 64, 65, 66, 67, 69, 120].

It is, therefore, the aim of this chapter, to look very briefly at available and experimental hardware, VLSI-technology, Robotic uses and Processing architectures for machine vision.

Section 1 addresses the reason for implementing sight in machines (in this case processors).

Section 2 addresses the various requirements to which any vision system must adhere to, in order to be implemented in a particular problem area.

In Section 3, the various processing architectures under consideration for computer vision are discussed. We also address the issue of sequentiality versus parallelism in processing architectures.

In Section 4, recent hardware implementations which have put machine vision in the spotlight are examined. We will also look at feature needs and developments, especially in the area of robotics (automation).

7.2 THE REASON FOR MACHINE VISION

It is estimated that 75% of the information received by humans, is visual [119]. The human vision system has been fine-tuned for processing and interpreting vast amounts of data rapidly and accurately. In chapter 2, the human visual system's ability to do this, was addressed. If this system can somehow be duplicated, so that the computer will be able to see and understand what it sees, it would mean that the problem of the knowledge acquisition bottleneck would be solved. Strong incentives for research into viable computer vision systems, can be found in the area of robotics - to provide robots with proper feedback capabilities to carry out tasks. Success in this problem area would also lead to major implications for AI itself. Once computers can recognize and understand complex scenes, the task of knowledge acquisition should be greatly simplified, and thus the performance of the machine should improve very rapidly as the knowledge base grows. The main driving force for computer vision systems to be developed, is that they can replace expensive human labour resources and faulty human decision making. The system would be cheaper, [6, 68, 69], and more consistent in its decision making. Computer Vision systems, that can do numerous factory-related jobs have been developed, Examples are:

- Analysis of steam-dispersion in high-pressure valves [6];
- Inspection of date-and department codes on packaging [6];
- Assembly of mechanical products [120, 12, 15, 127, 128];
- Control of Robotic Systems in the handling of ceramic and other parts. [64, 69].
Systems have also been developed experimentally that are able to: [67, 68, 69], [120, 121, 122, 123, 124, 125].

- Navigate a path through obstacles;
- Explore and recognize objects that are encountered;
- Correct its own mistakes through feedback, at a process control centre.

It is clear that Computer Vision Systems that can perform numerous tasks often done by humans, are currently being developed. Currently, there is mixing of hardware architectures and processors. These hybrid-architectures will continue to appear in commercial systems. In the future, however, more VLSI-implementations of specific algorithms will appear. This is made possible by a clearer understanding and definition of the problem which has to be solved by the system.

7.3 REQUIREMENTS FOR VISION SYSTEMS TO BE IMPLEMENTED

According to King [6], the four requirements of a vision system to be implemented is the following. (These requirements are basic indications of what is needed in a vision system, before any implementation can be done.)

(1) The system must have a high resolution capability;
(2) The system's processing speed must be fast;
(3) The system must be very flexible and extendable;
(4) The system must be affordable.

These requirements will now be discussed in greater detail.

7.3.1 The system must have high resolution capabilities

In the past, resolution has been limited by cameras and sensors. Today, solid-state cameras have resolution of 700 * 490 pixels. In the past, it was 520 * 490 at the most. The resolution determines how precisely an object can be measured and analyzed. Resolution can be a limiting factor in the processing of visual scenes in the following ways:

- determination of position: accuracy of position is severely influenced when resolution is not good.
- Identifying the number of parts in an image is not possible with a low resolution.

There are, of course, hardware limitations in representing images with a large size. [85]. For reasons of economy, images of considerably less spatial resolution than that required to preserve fidelity to the human viewer, are used. The choice of spatial and grey-level resolution for any computer vision task, is an important one which depends on many factors. It is typical in computer vision to have to balance the desire for increased resolution (both
COMPUTER VISION AND THE INDUSTRIAL ENVIRONMENT

grayscale and spatial) against its cost. The resolutions most frequently used are 128x128, 256 * 256 or 512 * 512 pixels per image grid (plane).

7.3.2 The system should have fast processing speed

This factor may be in direct conflict with the previous section. With a higher resolution, there are more pixels to process (in other words, the pixels may have more gray scale colors). The processing time will then be longer. Therefore, a certain balanced trade-off is needed to unite these two factors in a system. The advantages of fast processing power in a computer vision system are the following:

- In a factory-related task like inspecting parts or determining whether a part is present or not, the faster processing system will be able to inspect more production lines and will thus contribute significantly to overall production-time and also to the quality of end-products.

- If a fast processing system is used, fewer systems are needed to be multiplexed over factory lines.

- More complex processing can be done in a shorter time period.

7.3.3 The system's software and hardware should be flexible and extendable

In the course of the development of a particular vision system, it is sometimes necessary to add new processing functions to the system at minimum cost. That implies that a minor change in functionality and overall performance occurs in the system if a new processing element or function is added. Usually a processing element is a camera or video signal processor. Any algorithm being developed for visual inspection should be flexible in its response to new components. [120]. It should, therefore, be fast enough to meet the demands of the production environment. It thus requires a very carefully balanced approach to develop efficient software and the specific (specialized) hardware implementations of low-level algorithms. Low-level means that the algorithms or modules rely on context-independent knowledge of the properties of boundaries, surfaces or regions in a 2D or even a 3D-image. Boundaries can be detected in a 2D-image by applying a differential operator (for example, the Laplacian operator) to determine the changes in the greyscale intensities in the image. The principal benefit of applied machine vision is its flexible response to changes in the production line and manufacturing procedures.

7.3.4 The system should be affordable

The cost of an industrial vision system is determined by two factors:

- the hardware being used.

- the development time.

Vision Systems are still relatively expensive, although the payback-time is short. This hampers the idea of widespread acceptance of this new technology. Furthermore it is still
difficult to buy an 'off-the-shelf' system. Most applications require a high degree of application-specific software development and engineering of the environment.

The abovementioned facts lead us to the conclusion that vision systems are still expensive and their development-time is still lengthy. These two factors make vision application systems unaffordable to most. However, recent developments in this field may soon lead to more widespread applications to industrial systems. These developments can roughly be divided into two broad categories:

- visual inspection;
- visually controlled robotic guidance.

Today, context-independent preprocessing (e.g. two dimensional edge-detection [103, 104 93] and feature extraction), is realizable in hardware, which reduces processing time considerably.

7.4 A BRIEF LOOK AT THE MAIN STREAM OF VISION HARDWARE IMPLEMENTATIONS OVER THE PAST FEW YEARS

As stated earlier, the application of computer vision can be subdivided (very roughly) into two broad categories:

- Visual Inspection;
- Visually Controlled Robotic Guidance.

Over the past few years, the use of computer vision systems for industrial purposes has become increasingly widespread. In this section, the industrial implementations where computer vision was successfully applied, will be discussed. We know that industrial robots and flexible assembly and manufacturing systems are now part of the industrial scene. Throughout the 1980's robots and robotic systems have been deployed in the manufacturing environment at an ever-increasing rate. The following hierarchical description of the application of Computer Vision in the industrial environment, illustrates to what extent Computer Vision has been established in the industrial environment.

INDUSTRIAL APPLICATION OF COMPUTER VISION:

Automated Visual Inspection and Visually Controlled Robotic Guidance

7.4.1 Automated Visual Inspection

Visual Inspection involves the comparison of an image (or image sub-section) against a pre-defined standard image. Visual Inspection has been used to inspect products in the automobile, lumber, textile, glass and metal-processing industries. The task involves the inspection of known parts to ensure that they meet specified standards. The working environment for this type of implementation of vision applications, is more constrained. This means that certain issues must be taken into account when a wide spectrum of applied visual inspection tasks is studied:
• Primary concerns are optics and illumination;

• Structured light systems [10, 11, 12, 15, 120] using lasers or other optics and backlit work-stations to produce binary images, selective focus etc., are used to obtain illumination;

• High resolution is often needed over a large volume;

• This leads to the need for a 2-dimensional or 3-dimensional scanning system;

• High resolution images give rise to a massive amount of information which has to be stored. Some kind of data - reduction or compression is needed to avoid this.

The basic inspection task has two functional phases:

(1) A 'teach' phase.

In this phase, specific features of the item (object) to be inspected, are recorded and quantified, i.e. its quantity is established. In the context of the inspection task, this 'a priori' knowledge given to the system will then be used by the system in the second phase of the task.

(2) A 'run' phase.

This phase involves the inspection of the items. The inspected items or objects are then classified, which involves either rejection or acceptance of the item. The decision to reject or accept is based on the 'a priori' (pretaught) knowledge.

Some benefits of automated inspection:

• Automatic inspection can be carried out in dangerous or unfavourable environments (production environments). Often, such an environment is dangerously hot, or contains lethal gas which is unsuitable for human inspection (manual inspection).

• If the number of (defective components of products) is reduced in the sequential manufacturing process, the overall speed of production will increase, and low rejection ratio will result.

• If the number of faulty products is reduced at the point of sale, customer confidence will increase which will lead to higher sales-figures.

Automated Visual Inspection:

Automated Visual Inspection can be divided into the following categories [123, 124]:

Automated Visual Inspection:

• Dimensional surface inspection.
• Electronic inspection of fabricated items.
• Electronic layout inspection of circuit boards

7.4.1.1 Dimensional Inspection of Natural and Fabricated Items

Items in this category have a three-dimensional structure. Manipulation and control of items in three dimensions are often needed to make the inspection complete. [125] Some of the applications introduced over the last few years are listed in TABLE 1.

**TABLE 1**

**DIMENSIONAL INSPECTION OF NATURAL AND FABRICATED ITEMS:**

<table>
<thead>
<tr>
<th>AUTHORS</th>
<th>TASK DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>INSPECTOR [126] Perkins, W.A., 1984</td>
<td>INSPECTOR is a vision system for the inspection of automobile parts. Parts were presented randomly in 2-D orientations but with an already established 3D-base of fixed scale.</td>
</tr>
<tr>
<td>Inspection of automobile brake assemblies [127] Taulor &amp; Gregory, 1984</td>
<td>Description of a knowledge-based approach to inspect automobile brake assemblies. It was based on the comparison of grey scale one-dimensional profiles at key points in the image, against a known reference.</td>
</tr>
<tr>
<td>Identification of deformed panels in an automobile assembly plant [128] Schmidberger &amp; Ahlers</td>
<td>They used a single robot-mounted camera to find deformed panels in an automobile assembly plant. After the camera was moved to the point of interest, the image-enhancement was done and the image was converted to a binary form. The delineated form was then inspected against an already established reference. Inspection involves the finding of cracks, folds, missing holes and any other defects.</td>
</tr>
<tr>
<td>Inspection System for aluminium cashting [129] Decker &amp; Strecker, 1983</td>
<td>Developed an automatic X-ray inspection system for aluminium casting for the automobile industry. The system detected cavities within the casting, which were manifested as black spots on the projection of the X-ray. X-ray systems have also been used to inspect drill-holes in metallic components, to detect glass fragments in food jars [130, 131]</td>
</tr>
<tr>
<td>Dimensional measurement of metallic components [132] Mills &amp; Lim</td>
<td>Systems were developed for the dimensional measurement of metallic components. They provided precise dimensional tolerances of cylindrical components.</td>
</tr>
</tbody>
</table>
Detection of tool breakages
[134] Clements 1984

Designed a system to detect tool breakages on a CNC-machine by visual means. The tool tips on the contour of the threshold binary image were detected, a statistical measurement (of their shape) was taken, and these findings were compared before and after a tool was used on a machine or machines. If the differences were too extreme, the tool would not be allowed to be used further.

Water gear inspection
[135] Sternberg

Designed a system for water gear inspection. It was designed to locate missing or damaged teeth in water gears.

Inspection in food manufacturing industry
[136-141] Gagliardi

Visual inspection systems have been designed for the food industry. In [137] Kiwi fruit was scanned by an automated visual system. Shape and surface defects were located in apple bruises were described by a statistical classifier. This classifier was applied to the surface reflectance (infrared) characteristics. Circular biscuits were inspected [139] by means of edge detection, thinning and Hough Transformation.

[142] G. Baird

GAGESIGHT is a system that could locate pointers on analogue gauges to determine whether they were correctly positioned relative to predetermined markings on the gauges.

7.4.2 Surface Inspection

- What is Surface Inspection?

Surface inspection of natural and manufactured items has been implemented in a wide range of products, including surface inspection of metals ranging from continuous steel strip to metal castings and finished metal items. Inspection of surfaces involves the detection of defects on the surface of an item. For example, defects in metal castings may be raised areas. Various authors have given various names to such areas. They include 'fins' or indentations defined as 'notches', 'pits' or 'scratches'.

The difficulties that exist when complex curvatures are inspected, relate to the elimination of specular reflections, shadow detail and intensity gradients are discussed by [143], [144]. The solution suggested here was to design a local ring-illuminating system. In [145], [125] it is pointed out that machine parts are essentially 3D in nature and that multiple views of machine parts are needed. This feature requires the manipulation of the part, the imaging system or perhaps of both. In [145], a binary thresholding algorithm was applied in order to achieve a clear separation in intensity between the exterior boundary of the machine part in the viewed orientation and the background. Then the external contour was traced to detect any irregularities in the shape of the part. Statistical classification methods [146] can also be applied to measure the roughness of metal surfaces. It was noted that the intensity of the metal surface image is related to the variations of the surface normal. In this application, it was assumed that the lighting was very carefully controlled. In [147]
surface inspection was carried out together with dimensional-inspection. Cracks are one of the most common defects to occur in (both metallic products and non-metallic) Normally, the material to be inspected is coated with a fluorescent substance, after which the object is subjected to ultraviolet radiation for inspection. Cracks appear in parts as light or dark streaks against a background. This background’s intensity may vary. Therefore it is necessary to design algorithms that will trace the streaks of light in the object, but ignore them otherwise. The AUTOVIEW System [148, 149, 150] was designed to function in a sequence of 10 processing stages. The first stage commenced with the detection of streaks and spots. Then followed expanding, shrinking and skeletonisation stages which are common to binary image systems. Not all applications of surface inspection are done on metallic surfaces, but include wood, textiles [154, 155], core samples [156], and tyres [151].

7.4.1.3 Electronic Circuit and Layout Inspection

Electronic circuit board inspection is a very significant application area for visual techniques. It is able to use a priori knowledge to establish inspection standards. The inspection of electronic circuit boards (or printed circuit boards) is generally two-dimensional. It involves high-precision optics and very precise scanning systems. Usually a line-scan camera is mounted on an x-y table, which has a resolution of 10 * 10 pixels [157, 158]. Typical faults in electronic circuit boards include defects in drilling, insertions that were incorrectly made, components that were incorrectly replaced and solder defects. The above mentioned problems are most commonly solved through referencing systems [159, 160]. Here, direct comparisons of the board are made with a digital referencing representation. The simplest method that can be used is direct pixel-by-pixel comparison. It is, however, rarely used because of its many drawbacks:

- the need for very precise alignment;
- the board itself may have expanded or shrunk;
- very precise scanning calibration is needed;
- a large area of data storage is needed.

Feasible extraction has been employed as an alternative (and much more feasible) approach. Binary patterns are preprocessed to extract more compressed representations. By using local template matching of extracted features of the prototype, dimensional verification can be achieved. Non-referencing systems involve the comparison process on a generic level: generic features that should not be present are detected (for example, break in tracks). The reader is referred to TABLE 2 for a brief discussion of vision applications in the field of electronic circuit and layout inspection.
<table>
<thead>
<tr>
<th>AUTHORS</th>
<th>TASK DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>[162, 163] Silven, O et al., 1984</td>
<td>A system was developed where the binary image of the sample had to be compared to a prestored sample-version in the database. Borders between tracks were approximated by means of a series of piece-wise linear segments. When the comparison stage was reached, the linear segments were compared to the (template) database. The database was paged for a rapid loop-up comparison-operation. The only limitation of the system was that a very precise and accurate alignment of the sample-model was needed.</td>
</tr>
<tr>
<td>[164] West, M.A. et al, 1983</td>
<td>The abovementioned system was implemented at IBM. An electronic circuit board was scanned. An x-y table was used to produce a high resolution image of the board, and was subsequently compared to a designed database.</td>
</tr>
<tr>
<td>[165] Mandeville, J.R., 1985</td>
<td>Techniques of skeletonisation were used to find errors in already scanned width and positional information which was checked. The main drawback was that these features were compared to a small list of predefined feature types, leaving a low tolerance for errors.</td>
</tr>
<tr>
<td>[166] Skaggs, F., 1983</td>
<td>A system is described for interactive dimensional inspection of printed circuit boards.</td>
</tr>
<tr>
<td>[167], [168] Perkins, W.A. and Kaufman, P., 1984</td>
<td>Describe vision system for the inspection of components on flat (two dimensional) surface boards. In [168] linear segments were formed, piece by piece, through edge-detection, thinning and linking. These segments represented images of assembled digital watch printed circuit boards. The model scene was then aligned and a comparison (match), based on the information presented in the segments, was made.</td>
</tr>
<tr>
<td>[169], [170] Chou, J. and Kosner, J., 1985</td>
<td>A system that can verify whether specified components have been inserted (or not), is described. The underside of the circuit board was inspected to ensure that a lead tip was present.</td>
</tr>
</tbody>
</table>
Conclusions

The few examples studied show us that this application of vision is primarily concerned with matching an image (of the circuit board) against a predefined image in a database. Their limitation are obvious: precise alignment of the sample-model is needed in order to make a successful match-operation possible. One can surmise that a high resolution capability is absolutely essential, because a lower resolution would mean a loss of information in the image and a subsequent unsuccessful match-operation would be the result.

7.4.2 Visually Controlled Robotic Manipulation

In the previous section, the task of inspecting natural and manufactured items was discussed. The inspection environment is extremely constrained in terms of precise alignment and position. Any deviation in this constrained environment, will result in the inspection process yielding meaningless results. Systems applied to the inspection task would not conduct a proper inspection of an item if the constrained environment were to change in the course of the inspection.

This section deals with more flexible systems. Many such systems use robots in assembly-line operations such as the drilling of holes and the inspection of bolts. Other tasks include bolt insertion, spray painting and welding. These systems do not, however, receive active or passive feedback from the environment. They are constrained to function with very accurately toleranced part-positioning systems, and cannot respond to any error (or omission) in the feed components.

The application of vision to this area could lead to much more flexible and cost-effective small-batch manufacturing systems. The need for high resolution imaging devices, is paramount in inspection tasks. This requirement is much less constrained in the area of robotic manipulation. Other requirements which are much less constrained in this area include the fact that parts are not absolutely (precisely) located in 3D space. The sensor’s line of sight may itself be partially obscured by a robotic manipulator.

Visual Control Systems, especially those of the look-and-move variety, rely on two or three-dimensional sensing. In the two-dimensional case, however, the three-dimensional nature (geometry) will be retrieved from an established database. The parts represented in the database must, however, have a number of finite stable states, through which referencing of a viewed part in the scene, can be made in the database. The system must determine the nature and positional orientation of the part by comparing it against a known database. If the viewed part is grossly deformed, it will not be matched against any of the known orientations of the part contained in the database, unless the specific deformed part is also encoded into the database with specific reference to shape and orientation.
Visually Controlled Robotic Manipulation can be divided into the following application areas:

**VCRM:**
- Automated Assembly and Manufacture,
- Electronic assembly,
- Bin-picking,
- Navigation.

### 7.4.2.1 Automated Assembly and Manufacture

The principal benefit of applied machine vision in assembly and manufacture, is its flexible response to changes in the production line and manufacturing processes. There are two types of visual computational processing employed in automated assembly tasks:

- **binary image processing;**
- **grey level image processing.**

In a binary image, every pixel has either the value one or zero \([105,110]\). In a greyscale image, an image value is assigned to a position on an image. In terms of computational time (and complexity), binary image processing is much less complex than grey level processing.

Hence, binary image processing has been applied far more than grey level. The major drawback of binary image processing is that it requires a much more demanding investment in constructing highly-structured lighting environments. Greyscale processing provides much more detailed information on the internal and external structure of a component or part. Greyscale processing is often needed in a combined assembly and inspection process and is currently being implemented on a very limited scale. It makes specific assumptions about the geometry and environment of a component (part). The application areas of these technology-approaches are mainly large scale handling of manufactured items and also of electronic circuit board construction.

Table 3 is a brief discussion of vision application in automated assembly and manufacturing systems.
**TABLE 3**

**TABLE OF APPLICATION FOR AUTOMATED ASSEMBLY AND MANUFACTURE-VISION SYSTEMS**

<table>
<thead>
<tr>
<th>AUTHORS</th>
<th>TASK DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>[173] Kitchen, P.W. et al, and [174], Page, C.J. et al, 1985</td>
<td>In these systems, binary images and are processed by procedures which are described. These images are used by a robot with parallel pincher and gripping devices.</td>
</tr>
<tr>
<td>[175] Nehr, G. et al, 1985</td>
<td>The algorithms for the positional calculation of grasp-coordinates are given. The principle of 'look-then-move' is used. This caused the robot to become blind after the image was eventually sampled and a subsequent analysis was done.</td>
</tr>
<tr>
<td>[176] Linder, H. et al, 1985</td>
<td>A system which can grip and transport engine blocks, is described. An active light system scanned the top surface of each component to detect the possible presence or absence of a particular feature. This was done before the part was handled by the system.</td>
</tr>
<tr>
<td>[177] Schmidberger, et al, 1983</td>
<td>In this system, grey level images are first sampled. Then a robot is directed to attach a water-pump to a car's engine block. The grey level (Intensity)-image is then mapped on to a binary image by adapting the contrast locally or globally. Then the result of adaptation is changed by thresholding it to a thresholded image. The robot is then able to grasp the water-pump with the help of this image.</td>
</tr>
<tr>
<td>[178] , Bauman, R.D. et al, [179] Vander Plas, T.W. [1982, 1985]</td>
<td>At General Motors, CONSIGHT was developed as a system for binary imaging. Its task is to sort car components, up to 1400 per hour. The binary image is produced by an active laser triangulation-system, which results in a 3-D image. This system is connected to the action of the conveyor. Identifying and orienting parts are achieved by binary processing techniques. This information is then transmitted to a robot mounted lower down in the vision system. General Motors have also opted for the use of vision control to locate and tighten nuts on the undersides of cars.</td>
</tr>
<tr>
<td>[180] Groen, F.C.A. et al, 1985</td>
<td>A robot which can be visually directed to assemble hydraulic components for gas heaters, is described.</td>
</tr>
</tbody>
</table>
COMPUTER VISION AND THE INDUSTRIAL ENVIRONMENT

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[184], [185], [186], [187], Cronshow, A.J. Brun-Buisson, A. et al, Hilaman, C.R., Bonnet, P. et al, 1982, 1984, 1985</td>
<td>In these papers various other applications for binary vision control are described: chocolate decoration, glass bottle-sorting, stacking of paper cups [186], handling of plantlets [187].</td>
</tr>
<tr>
<td>[188] Amat, J. et al, 1984</td>
<td>A technique for visually guiding controllers (robotic) which compare successive images in certain time sequences to distinguish the object-boundary from clutter and noise is described. This is a grey level processing system. These systems make fewer demands on alignment and can solve ambiguities of orientation. They are, however, more complex than the binary processing systems.</td>
</tr>
<tr>
<td>[189] Franke, E., et al, 1985</td>
<td>Here, a system which directs a robot, is described. The robot drills a hole in a circular rivet. The Hough Transform is used to locate the centre of the rivet. The visual data obtained is greyscale visual data.</td>
</tr>
<tr>
<td>[190] Geschke, C.G., 1983</td>
<td>Here, visual sensing, together with the use of tactile control, are discussed. Dynamic visual tracking is used to guide a bolt to the location directly above the hole. The tactile sensor then monitors the external source along the bolt-axis, during insertion.</td>
</tr>
<tr>
<td>[191-192] Mikami, K. et al, Agrawal, A. et al, 1986, 1983</td>
<td>Here, the 'eye-in-vision' system is described, in preference to the fixed view camera. The drawback of this type of system is that the 'eye in the hand' is not always feasible. The 'eye in hand' system tends to have a very limited problem-domain in which it can be effectively used.</td>
</tr>
</tbody>
</table>

Conclusions:

Most systems developed for automated assembly and manufacture use binary imaging to obtain images. Robots are then directed to perform certain tasks where they require the information obtained in these binary images. One major drawback of binary imaging is that precise alignment of components is needed. Ambiguities in orientation and alignment position can cause faulty results. The system's processing is much less demanding, because a pixel on the image-grid can have only two possible values ('0' or '1'). There are also systems with grey level imaging capabilities, which make far fewer demands on alignment and environment, and can solve ambiguities of orientation. They are, however, much more complex because of the ambiguity-tolerance.
7.4.2.2 Electronic Assembly

Just like visual inspection, circuit board assembly and the assembly of consumer items are very active areas of vision applications. **TABLE 4** contains a discussion of vision applications for electronic circuit board assembly systems.

**TABLE 4**

<table>
<thead>
<tr>
<th>AUTHOR(S)</th>
<th>TASK DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>[193], Sanderson, A.C. et al., 1983</td>
<td>A flexible assembly station in which electronic components are inserted into printed circuit boards, are described. Up to 100 discrete packages were inserted into the boards. Positional fault-tolerances of up to 0.0127 - 0.0254 cm were allowed. One draw-back of automated insertions is that discrete component leads are very flexible, which makes the insertion task with visual control very difficult in this case. The visual control is, however, crucial because lead-tip positions are not known beforehand and must therefore be located by the vision system. Here, binary and grey-scale image processing are used to find the positional location on the lead tip. Force-sensing devices mounted on the gripper are used to monitor final insertion.</td>
</tr>
<tr>
<td>[194], Yajima, T. et al., 1985</td>
<td>The automation of hole-princing inspection is discussed. Visual defects in circular marks of different degrees of reflection on the board base are described.</td>
</tr>
<tr>
<td>[195], Buffa, M.G., 1985</td>
<td>A system whereby the deposition of solder on pads is controlled by the system is described.</td>
</tr>
<tr>
<td>[196, 197], Demazeau, Y. and Liscano, R., 1984, 1985</td>
<td>Systems for the location of flexible wires are described. The author of [196] used a stereo vision system to locate the boundaries of the wires. Depth data information was obtained in [197] by correcting the wire-image with a projected shadow where a known point of light source was used.</td>
</tr>
<tr>
<td>[198], Cowan, G.F. et al., 1985</td>
<td>The ACRONYM system is described. This system can reason about 2D Images on the basis of 3D-models. The system tried to interpret images by means of 2D projections of objects consisting of cones. These projections defined the cones's features. The system is currently being used for the assembly of a push-button switch in a unstructured environment.</td>
</tr>
</tbody>
</table>
Conclusions

In this brief study of this application field of Vision, it may be conclude that:

• visual information must be used for the subsequent assembly task;

• for the task of automated insertion, vision is both crucial to the task, as well as being a drawback. [193].

7.4.2.3 Bin-Picking

The bin-picking problem involves a robotic controller which picks a part from a jumbled pile of parts and then puts the part into an isolated environment for the next stage of the manufacturing process.

The problem has had many solutions. Most of the solutions constrained the environment in order to fit in with the application. The part may either be picked up from a deep container, or it may be picked up from a pile on a flat surface, thus constraining the third dimension. Binary or grey-level image processing may be employed. This will determine the lighting conditions to be used. There may be a single type of component or several different kinds of components on the pile. The latter makes identification of paramount importance in the process. The preferred gripping locations are wired in (pre-programmed). The visual task will then involve locating the most accessible component, and then the corresponding features. In the final stage of the process, the 2-D to 3-D image to robotic transformation is computed. TABLE 5 contains a discussion of vision applications for bin-picking.
TABLE 5

TABLE OF APPLICATIONS FOR BIN PICKING

<table>
<thead>
<tr>
<th>AUTHORS</th>
<th>TASK DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>[199, 200, 201], Kelly, R.B. et al. 1983, [107-108], Wechsler, H. et al., 1989</td>
<td>The bin-picking problem has been a major area of research of the Robotics Research Centre, University of Rhode Island [199, 200, 201] and at The Centre For Automation Research, University of Maryland [107, 108]. Kelly et al's approach was a 2D-imaging system. The third dimension i.e. the distance between the object and bin-site, was ignored. The gripper travelled along the line of the site toward a point of contact. A proximity sensor indicated that contact with this point was made. A shrinking-algorithm was applied to already thresholded binary images. Smooth regions were identified in this way, by means of which the gripper (vacuum) could locate the contact-point.</td>
</tr>
<tr>
<td>[203], Bach, J. et al, 1985</td>
<td>For the bin-picking task, a light striping principle was used to obtain depth from triangulation.</td>
</tr>
<tr>
<td>[204, 205], Svetkoff, D.J. et al, Coleman, E.N.</td>
<td>In these schemes, a feature based approach was used. 21/2D Sketch data was compared to a 3D-model, (geometric model) of the component. The 2D Sketch was obtained by means of range data, which was obtained by phase-sensitive detection of coherent laser radiation from the surface of the object.</td>
</tr>
<tr>
<td>[107, 108], Wechsler, et al.</td>
<td>As described earlier, object recognition was successfully performed by using Wechsler's Distributed Associative Memory [110]. The recognition did not vary according to size changes or rotation. Invariance to geometrical distortions and robust response in the presence of noise and occlusions, were also reported.</td>
</tr>
</tbody>
</table>

7.4.2.4 Robotic Navigation

Various robotic manipulation projects have been started at various institutions [64-69]. These include Purdue University, The Oak Ridge National Lab, Carnegie Mellon University, Institute National Polytechnique de Grenoble and the FMC Corporation. Autonomous mobile robots deal with the empirical world, which is never fully predictable. Therefore, the robot must monitor its environment to perceive and diagnose unexpected events [64, 65, 68]. The robot must adjust its plan to changing conditions. Large amounts of computer processing must be done on board the robot, because of the computational intensity of visual perception and replanning in real time. Intelligent mobile robots can go
where humans cannot, or do not want to go [66]. Mechanization of repetitive labour-intensive tasks can be achieved by mobile robots.

Applications of mobile robots include agricultural harvesting, combat and combat-support systems, planetary exploration vehicles (for example the planned Mars Rover), the handling of items from a production environment and their subsequent transfer to another location for (automated) assembly. Such robotic systems must be able to accept mission commands, constraints, must be able to plan actions, execute their plans and receive feed-back of the results of their actions. Great engineering challenges underlie the development of technologies for self-navigation. These include planning, perception, navigation and control [65, 66, 67].

At the Oak Ridge National Laboratory various robotic systems have been developed over the years [68], the latest member being the Hermies II B system. Hermies II B has focused on visual perception, goal recognition and navigation in unknown (dynamic) environments. The programmes used by the vision system show the successful integrated mixing of image processing and analysis algorithms on the concurrent multi-processors of the eight-node Ncube hypercube on board the robot. The paper states that by placing all the computer power needed for sensor analysis on board, planning and control can be achieved at least in the context of this project. Hermies II B can navigate among moving obstacles using only its sonar sensors. It can also navigate using its vision system, but only in obstacle-free regions.

Carnegie Mellon's Automatic Landing Vehicle Group [67] designed an autonomous mobile robot system capable of operating in outdoor environments. The system is able to drive a robot vehicle on a network of side-walks which includes a curved road through an area densely populated with trees.

The CMU-system can currently:

- execute a pre-specified mission over a mapped network of citadels and drive up a sloped path;
- recognize intersections and landmarks;
- drive on unmapped or very poorly defined roads;
- detect obstacles and stops until they are moved out of the way;
- avoid obstacles.

Furthermore, the system is highly modular in its architecture. This includes components for both global and local navigation. A route planner carries out the task of global navigation. It searches a map database to find the optimum path that will satisfy the mission requirements. It then oversees the execution. Several modules carry out local navigation by using a colour camera and laser range finder to recognize roads and landmarks. Perception, planning and control components are all integrated by the Codger software system. Codger also provides module-synchronization to maximize parallelism at global and local levels.
The objective of the FMC Corporation [67], was to develop an autonomous vehicle capable of real-time operation both on- and off-road. This was subsequently achieved. To date, the system has been capable of mission planning, route planning and the execution of this plan. Obstacles could be detected and avoided at 15 km/h and a road could be followed at 24 km/h. Recent research efforts focus on multiple vehicle control and advanced autonomous capabilities, such as dynamic replanning, reaching the road, local slope navigation, road-network navigation, as well as navigation based on landmarks.

In [65] a program which co-ordinates action and perception in a mobile surveillance robot, is described. The paper reports on a production system which controls two different levels of architecture: one for navigation and one for perception. Task level control is performed by a supervisor implemented as a production system. It chooses the navigating and perceiving procedures according to the mission plan and local environment. The planning of the mission involves deconstructing the mission into sub goals which can then be translated into procedure activation calls. These procedure-names encompass the task level procedures. Then a verification is done to determine whether the mission is feasible or not.

From the work being done in this area of robotic navigation and control, it is clear that most of it is concerned with planning and executing a route (mission). The work done to date, has followed an evolutionary process of development [65, 68] where the newer system is a slight improvement on its predecessor. Future research in this area is of paramount importance because of the economic and work-related issues at stake; i.e. the mechanization of many labour-intensive processes.

7.5. SUMMARY

As has been shown, the potential application areas for vision-driven automated systems are fairly numerous. Applications can be classified according to processing requirements. The discussion has focused on the following categories of vision system applications. The industrial application of computer vision has been divided into two broad categories:

(1) Visual Inspection;

(2) Visually Controlled Robotic Guidance.

In the area of visual inspection, various implementations by numerous researchers has been discussed. It has been shown that visual inspection can be used as a very powerful tool in the automation of quality control procedures. Specific measurements of important parameters in a processing or assembly environment can be done. Simple criteria, like a pass/fail decision, can be set up or more complex criteria where a diagnostic procedure can be used to recover or control errors, can be introduced. It can be implemented with various degrees of accuracy. For example, an inspection task may involve finding quantitative measurements and ensuring that the set of components is manufactured within certain allowed tolerance levels.

The vision systems which were studied required them to be able to identify objects or items. For example, identification markings (like drill holes) may be imaged, processed and
recognized as part of a sorting process, or they may indicate the places where holes must be drilled.

In Robotic Guidance through vision, various systems implemented in the areas of Automated Assembly and Manufacture Electronic Assembly, probing and placing of objects and navigation were studied.

Vision Control may be used to guide the actions of a mechanical manipulator to effect physical movement of items. The precise situation and environment can vary considerably. For example, objects may be picked up from a moving conveyor belt or, as is also often the case in practice, an object or item may be picked up from a storage bin (bin-picking). It was clear that picking-up tasks are generally more difficult to automate than object identification tasks, as there is no organization of objects in the picking-up tasks.

Visual Capability can be used to guide the movement of some manipulator or vehicle in executing a task or in travelling from one point to another. In this case, a general goal-oriented approach, which includes route-planning, perception and object avoidance, is needed. Systems that can fulfill precisely those requirements which visual guidance presents to a manipulator or vehicle, have been studied.
CHAPTER 8

PROCESSING ARCHITECTURES USED FOR COMPUTER VISION
8.1 INTRODUCTION AND HISTORICAL REVIEW

Computer Vision is very expensive in terms of computation. Conventional sequential processors do not have the processing speed capacity to perform complex visual tasks like three-dimensional object recognition in real time. There are various forms of parallel processing used to speed up this complex, higher-level computation, many of which involve combinatorial search-strategies. If we want to achieve vision in real time for complex scene-environments, we must be able to design computer algorithms and architectures that can make effective use of parallelism in the computational analysis of the scene.

The intention of this chapter is to discuss various parallel-architectures that are currently used in vision computation. Multi-resolution and its relevance to parallel architecture are also discussed. The problem here is that an image may consist of many picture elements each one with its own colour intensity, and which must be processed in order to do a successful analysis of the image. The problem is usually solved by subdividing the image into several sub-images; specifically, if an image is divided into quadrants and this operation is performed repeatedly, until a single picture element is reached, a hierarchy of version-levels will have been created. A level has a hierarchy of twice the number of processing elements in the previous level. These levels correspond to versions of the image at different resolution-stages. A quadtree-structure is created when the process terminates if it reaches a homogeneous block. The various requirements of processing architectures used for vision computation, are discussed. Lastly, the various autonomies in terms of processors and connectionism, are discussed briefly.

In the 1960's the first parallel processor architectures for image processing was developed through the ILLIAC [207] and SOLOMON [208] projects. Extensive parallelism was developed theoretically, but could not be fully implemented because of the technologically limited hardware available at the time. The important realization that emerged from these projects, was the idea of one single programme controller, which directed and co-ordinated the concurrent and identical operation of many processing units. Later, this idea became part of the Flynn taxonomy (classification) of parallel architectures. This specific approach was later called the Single Instruction, Multiple Datastream Approach.

In the 1970's, a new direction of development began to emerge: the implementation of multi-processor systems. The first large-scale multi-processor which was specifically developed for image processing and pattern recognition applications was called EMMA [209]. It consisted of hundreds of specially designed processors, as powerful as a minicomputer today, and interconnected by a level-hierarchy of buses.

The very fast development of VLSI in the late 1970's eventually led to a breakthrough in parallel processor development. These developments include high performance microprocessors, dense memory chips, and custom integrated circuits. All these developments led to a new generation of parallel computers which featured extensive parallelism at low cost. This opened a number of new applications fields to image processing and later to computer vision. Pipeline computers are specifically oriented to basic 3 x 3 neighbourhood operations, which proves useful in both linear and morphological image analysis. Mesh computers are able to perform a wider range of image processing and pattern recognition operations. With the advent of the mesh-connected computers, the era of fine
grain massively parallel computers began where processing power and speed is far greater because a very large number (many thousands) of small processors is used, instead of a limited number of powerful computers.

8.2 MULTI-RESOLUTION OF IMAGES

As described earlier in the introduction, an image can be divided up into several sub-images. As stated, when dividing the picture up into quadrants is considered, this operation is performed repeatedly in a recursive manner, a hierarchy of levels which will then correspond to different versions of the image at different resolutions, can be obtained. In this section various architectures that are employed in the field of computer vision to solve the vision problem will be discussed: the extraction of information about a scene by computer analysis of one or more images of that scene.

8.2.1 Multi-resolution Architecture

This architecture has been implemented in the Pipelined Pyramid Machine by Burt in [101],[210]. This system has been applied in the surveillance of a building and its outdoor environment in an outdoor scene [210]. A functional description of a pipelined architecture is given in [79]. This description serves as a reference for the functional performance of the pipelined pyramid machine. The architecture of the pipelined multi-resolution constructor [79] consists of a main pipelined loop which includes the image memory. The image memory contains all versions or viewpoints of the image under consideration. A filter unit receives information via the sub-array element values, which was obtained by suitable delay lines and a decimator unit which decides which subset of the elements will form the next version of the image. Operations of which the filter unit is capable, are convolutions implementing both Gaussian and a specific Laplacian transform of the original image. The task of the processor is to decompose the available computational primitives into a sequence of transformations: the available computational primitives [79].

8.2.2 Pyramid Architecture

Pyramids are perhaps the simplest hierarchical data structures for image information structured computer vision[104],[211][213],[85],[120],[228]. The simplest application of pyramids is in using them to provide reduced-resolution versions of an image[85],[104]. The pyramid provides a representative selection of the possible versions of an image at different resolution-degrees. There are quite a number of definitions of pyramids that have been proposed. Four parameters are needed to describe a pyramid fully [120],[213]:

- How many hierarchy levels are there (d)?
- The tesselation topology (t), defines the connectivity among the nodes of the same hierarchical level. Each node can only be connected to the four immediate neighbours in the cardinal direction (on the same level). This is called the 4-connectivity of the nodes. Each node can also be connected to the eight-near neighbours in the same level, thus 4 new directions are added (diagonal directions) which result in 8-connectivity of the nodes. Each node can also be connected in a hexagonal tesselation or 6-connectivity of the nodes.
The width of the support to build each level (w). This defines the interconnections between the nodes which belong to adjacent hierarchical levels: each node thus has w-sons in the successive hierarchy-level.

The degree of reduction which will occur between successive levels of the pyramid. Assume that this degree is denoted by p. The previous factor w was the width of the support used to build each new level. If there are no overlaps in the pyramid, then p = w. This means that the sets of sons of the nodes of a level n, which belong to the (n+1) level, are disjointed: no one node belongs to two fathers and then d = 1 + t + p. In the case of overlapping, w > p and d = 2w + t - p [211].

Other considerations that must be dealt with when discussing pyramid architectures include:

(1). A particular node is called the apex of the pyramid; if n = 1, and it represents the whole pyramid.

(2). If the apex and its links are eliminated when n = 1, we are left with the remaining nodes that are divided into p disjointed sets, where each of these sets represents a pyramid of n-1 hierarchical levels. It is usually called called a sub-pyramid of the apex

Two different approaches exist in the pyramidal structure of the image-processing system. These approaches are commonly known as the bin-pyramid and quad-pyramid approach [97-99].

For the bin-pyramid, p = 2. In this approach, each father-node has two sons and for alternating planes of the pyramid, the vertical connection is performed with rows and columns respectively. In this way, the vertical (inter-level) connections-structure constitutes a binary tree [97,98]. The smaller reduction ratio of 1:2 (p = 2) means that we can adopt the VLSI design because fewer pins are needed in the chip itself to support the architectural design. Figure 1 shows diagram of the bin-pyramid (binary-pyramid).

The quad-pyramid is an isotropic structure meaning every neighbour node in the pyramid receives information from all its neighbours. This is done in parallel. In this way local operations which are all performed recessively, can be supported. These local operations deal only with one plane. The distance from the apex to the base is n + 1 whereas it was 2n + 1 in the bin-case. The linear size of the base is of magnitude 2. In order to exploit the pyramidal architectures, several pyramid-primitive,can be introduced. The most general and significant are projection and injection,which allow the concurrent, bidirectional as well as vertical flow of the data through the planes. In most cases data is injected (presented) at the base of the pyramid to the apex [104]. Data-reduction takes place in certain tasks like moment, min/max computations, averaging and so on. An example of a projection primitive is a case where information flows from the vertex to the base according to projection such as if one assigns a value to a sub-architectural level. Some data-reduction work is more effectively implemented in bin-pyramids than quad-pyramids, while the quad-pyramid is a more effective architecture if fast propagation operations are desired,because the vertical neighbours' paths reduce the time from a factor O(n) to O(log n) and O(log n) < O(n). In figure 2 below, a diagram of a quad-pyramid, is given.In this case, p = 4, the distance from apex to base is n + 1 and the number of processing elements
are \((4/3) \cdot 2^n\). It has been suggested that the definition of a three-dimensional context can enable an instruction to hold only in that context [120]. A context may be made up by a subset of pyramid-planes. The idea of context is further discussed in chapter 9 and 10 where a theoretical model for the recognition of three-dimensional digital structures is intuitively defined. The model is based on the theoretical work done by Rosenfeld in [84] on pyramid automata and the random context grammars and automata defined by Ehlers and von Solms in [221].

Several other operations using pyramids as a tool to represent any image at different levels of detail [104], have been implemented. These include grey level Consolidation, where the digitized image is partitioned into non-overlapping neighbourhoods of equal size and shape after which these neighbourhoods are replaced by the average of the pixel-intensities in that neighbourhood. Pavlidis [120] demonstrated that pyramids can be used for edge-detection. They used two pyramids, one of which stores the image while the other stores the edges of the image. Other operations include the use of pyramid structures in correlation. In this application, binary search correlation uses the pyramids of the input image and reference patterns. The algorithm is as effective as binary search. In this strategy, the feature to be located is at some unknown location in the input image. The reference feature originates in another image, the reference image. The feature in the reference image is contained in a window of \((nxn)\) pixels. The correlator must then find an \((nxn)\) window in the input image that best matches the reference image window which contains the image. The centre of the search area varies in the input image as the improved resolution refines the point of best match.

![Figure 1: Bin-pyramid](image1.png)

![Figure 2: Quad-pyramid](image2.png)

### 8.2.3 Two-Dimension (Flat) Array Processors

It is possible to simulate a multi-resolution system using a flat array. The sub-images, which are obtained by recursive division, are embedded in each other. To gain access to lower levels of resolution, we must guarantee near neighbour access. This architecture provides a two-dimensional method or strategy in which the pyramid-architecture can be implemented [97], [99], [103]. It involves a method for directly implementing parent-children relationships.
The main features of this layout are that the top leftmost cell represents all the layers within the pyramid. The successive layers of the pyramid are split recursively, with the effect that the link between the father and sons is the same irrespective of the considered couple of layers. The outline of this layout can be found in [97].

The leftmost top cell of the 'flat' pyramid must be very tightly specified because this cell must work for every layer. In other words, it must represent the whole pyramid in one single cell. In the previous section, the structures of pyramid-architectures were described. It was seen that these structures are highly modular in that they consist of layers of cells ranging from the base of the pyramid to the single apex-cell. The cells in each layer of the pyramid are connected and the layer itself is interconnected to the layer above (father-layer) and the layer below (son-layer). This connection of cells constitutes a highly modular architecture. Because of this, the two dimensional (flat) array approach has a major drawback in that this modularity is lost. Furthermore, if the topmost left cell is faulty, it will effect the performance of the whole pyramid, since it is equivalent to the apex cell of the modular-structured pyramid described in the previous section.

The logical flat pyramid attempts to answer the questions of how processing elements may be distributed spatially as well as functionally in order to attain maximum efficiency. The degree of hardware economy and, simplicity as well as modular simplicity are also addressed. The physical structure of this type of pyramid is a mesh [97, 98, 213]. In the application it is shown that the position of viewing interest can vary in a region. The pyramid-mesh can hence be placed in a position of interest in a region. It further features an external control system which can support more than one logical pyramid. Hence, more than one pyramid can be supported concurrently, while they all work in different positions of interest in different regions, in parallel. The pyramid may have its base and apex wherever it is required in the image. The hierarchical dependance of the children processors (inside the regions of interest) is defined by the location of the apex-cell. In [97], it is stated that the relationships between cells may differ according to the location of the pyramid's apex.

In conclusion, a few general features of the mesh-architecture as well as the mesh-of-trees architecture are given. In general, the mesh-computer is a two dimensional grid of processors, each connected to its four nearest neighbours [213]. The edge-processors, however, has fewer connections. Recently, the mesh-of-trees has been considered as a vision architecture by [99]. It starts with a base, which is a mesh of n-processors. A tree is then added over each row and each column. The trees are all disjoint except for their leaves and form the base-processors. The total number of processors in these architectures is $3n^2 - 2n$. In both the pyramid approach and flat array architecture, the aim is to achieve a varying resolution capability: the next layer is a finer grained resolution than the previous layer.

Recently, real time systems have been developed by means of a reduction in data to be analyzed. The strategy for image understanding has followed the following approach:

- Consider only regions of interest
- Choose the required spatial and greyscale resolution
• Parallelize this operation in the most efficient way. In this regard the external control system that will be needed as well as scheduling approaches, must be planned [213].

These three factors are addressed in the multi-resolution approach and specifically in the pyramidal architecture [99, 236]. A pyramid computer is complicated to implement because of the requirements of sophisticated technology for the covering of fault tolerances and chips. Different architectural solutions have been proposed. The architecture used in general today, is the flat array in which multi-resolution is produced.

The other one is in the use of high level vision through hypercubes [99, 213, 216, 214, 211]. For multi-resolution to be used effectively, an adequate high-level language as well as software environment, is needed. Data types must be matched to the image data, and operators must be used for exclusive image processing computation. Furthermore, we need an effective scheduling scheme is needed to partition tasks interactively in order to make full use of processing elements.

8.2.4 Mesh Network

In a mesh network, the nodes are arranged into a q-dimensional lattice [97-99, 213]. Communication is allowed only between the neighboring nodes: the interior nodes communicate with 2q other processors. The disadvantage of these networks is that data-routing requirements often prevent the development of $O(\log n)$ parallel algorithms.

8.2.5 Butterfly Network

This network has $(k + 1) \times 2^k$ nodes which are divided into $(k + 1)$ rows (ranks). Each two rows have $n = 2^k$ nodes. The rows are indexed from 0 to $k$. Let node $(i, j)$ refer to the jth node on row i with $0 \leq i \leq k$ and $0 \leq j \leq n$. The node $(i, j)$ on row i is connected to two nodes on row $(i - 1)$ and node $(i - 1, m)$; $m$ can be found through the $i$-th most significant bit in the binary representation of $j$. The entire network is made up of such "butterfly" patterns, hence the name of the network [213].

8.2.6 Hypercube Network

This network [99, 211, 213, 214, 215, 216] is a butterfly network in which all the columns collapse into single nodes. This network consists of $n = 2^k$ nodes which form a k-dimensional hypercube. Nodes are indexed from 0 to $2^k - 1$. Two nodes (or processors) are connected if and only if their indexes differ exactly in one bit position. The NCUBE is an example of a vision application of this network topology [214]. Other examples include Multi-processor (hypercube) implementations [215], Graph and Image Processing Algorithms [216] and the application of a hypercube on a connection computer [217].
8.3 CLASSIFICATION OF PARALLEL ARCHITECTURES ACCORDING TO THE FLYNN TAXONOMY

The best known taxonomy for parallel computers was proposed by Flynn [97,98]. It is concerned with the multiplicity of the data stream and instruction stream, and identifies four classes of computers:

8.3.1 Single Instruction Single Datastream (SISD)

SISD computers correspond to regular Von Neumann computers. They execute only one instruction at a time and process one data-entry (datum) at a time. They are not very commonly used in computer vision tasks because of their obvious computational limitations when used for vision processing. The traditional digital computer is a structure that operates on a stream of data taking one piece at a time and executing instructions in sequence on a series of data segments.

8.3.2 Single Instruction stream Multiple Datastream (SIMD)

These computers are based on a programme controller which drives the flow of the program. The controller broadcasts a single instruction to a set of processing elements that receive and execute this identical instruction. They, however, create their own computational output. The processing elements each have an arithmetic logical unit (like all processors), and a certain amount of storage capability (local storage) like registers and memory. In the next section, a further subdivision of the SIMD-computers, will be discussed. This subdivision is in fact a new category devised by Fountain [94] for classifying parallel computers. It is based on the type of autonomy of the processing elements. The elements can be fully autonomous or non-autonomous. For the case of full autonomy, we make three distinctions:

- addressing autonomy,
- connection autonomy and
- operation autonomy

All the network-topologies have been implemented in all four categories of Flynn's Taxonomy. Notable examples are given in [98], [57], [78], [94], [99], [101], [97], [104], [215], [216], [217], [223]. Some of the mentioned systems are PASM, NCUBE, DATACUBE, WARP, Cytocomputer, PIPE, CLIP4, DAP, YUPPIE and so on. These systems have been developed on the network-topologies described earlier. Other well-known topologies are the linear, 3D cube, ring-topologies.

8.3.3 Multiple Instruction Single Data Stream Computers; (MISD)

These computers are often related to the pipeline computational approach which was discussed in section 8.2.1. In this approach, a sequence of processors elaborates a stream of data (processing meaning elaborating) flowing through it. Each processor does some processing on the data item. Its output then serves as input for the next processor in the sequence. Processing of images by pipelined computers in this way are done in raster-scan
mode. The pixels are extracted from the image one at a time, and is then fed through the pipeline [98, 200]. Each processor performs an operation on the pixel until the pixel is fed through the sequence of processors. Examples of MISD-systems are the Cytocomputer by Sternberg [218] and the Pipe-system [219, 223], by Kent, Sheiner and Lumia. Both systems have been implemented as a linear network topology.

### 8.3.4 Multiple Instruction Multiple Data Stream (MIMD)

These computers consist of several regular Von Neumann computers. Each Von Neumann-computer operates its own programme and processes its own private data. Usually, there is a common memory for communication purposes. Various network topologies can be employed using the MIMD-approach. These include ring, bus, mesh, hypercube, linear and multi-stage networks. Various systems have also been developed using these network topologies. These include PASM, NCUBE, IUA, DATACUBE, WARP, ZMOB, Victor and so on. It must however be stressed that most of the so-called SIMD-architectures do not adhere to the precise taxonomy of Flynn. There seems to be an evolutionary development from strict SIMD to strict MIMD.

It is also noted that most of the network topologies used are ring, mesh, linear or n-cube. Networks which have interconnections can be either multi-stage or single-stage. The differences between these two approaches relate to the linking of the processors of interest. In the former, data passes through several stages of switches to reach the intended destination processors. In the latter's approach, the related processor are directly linked.

Another issue that must be considered in vision computation is granularity. Fine grain parallel computers are based on a large number (in the tens of thousands) of simple processing elements. They are usually organized in SIMD-mode and their data width is bit-serial. These computers are called massive parallel computers. Coarse grain parallel computers, consist of a smaller number of processing elements. They may have hundreds of powerful processors, organized in MIMD mode, working with 32 bit data streams and having floating point capabilities.

### 8.4 CLASSIFICATION OF SIMD-COMPUTERS ACCORDING TO FOUNTAIN’S TAXONOMY OF AUTONOMY:

Over the years, there has been a great interest toward massive parallelism due to the advances that have been made in integrated circuitry and packaging technology which both gave rise to the opportunity of developing parallel systems which have thousands of simple processing elements organized into a SIMD-approach. This has led to a number of architectures which do not precisely fit into the SIMD-approach. Fountain [94, 223] therefore refined Flynn's taxonomy by introducing another criterion for the SIMD-approach, called Processor Autonomy. In Section 8.4.2.3 another type of autonomy namely connection autonomy, will be discussed.

#### 8.4.1 Processor Autonomy

By enhancing the basic SIMD structure, each processing element may be provided with a certain level of autonomy. Processor autonomy is divided into three levels of autonomy:
8.4.1.1 **Operation Autonomy**

Previously, all processing elements performed the same operation. With operation autonomy, different operations can be executed concurrently, thus increasing the level of parallelism in the computer. Thus more than one instruction can be executed by all processor elements.

The most common kind of operation autonomy found in most fine-grained SIMD parallel computers identifies two clusters of processing elements: the processing elements that must execute an instruction and those that cannot. The selection of whether or not a processing element must perform an operation or not, is done by a special one-bit register in each element, called the 'Enable', 'Masking', or 'Activity' register. This single register enables or disables all the memory and registers any "load" operations that are done.

A conditional situation can be envisioned by this approach. An if-then rule-situation for local operations, can be designed. The rule can be of the normal if-then kind i.e. if (situation is satisfied) - then (execute a certain action), otherwise, do nothing. In this way, some elements will be executing a certain action and others not. If it extended to a general multi-cluster element-case, this conditional statement can be made a CASE-statement and enable massive parallelism through this.

An example of operation autonomy is to be found in the CLIP7-system [213,223,94]. This system is devoted to issues related to autonomy. CLIP7 has the regular 'activity bit' to decide whether an operation should be executed or not. CLIP7 also has a multi-bit register which is seen as an extension of the ALU-operation code. The broadcast is then interpreted differently in each processor under programme control.

A third type of operation autonomy can be found in Multiple SIMD-computers. There are several programme controllers, each coupled to a processing element-cluster. For example, in pyramid processors, different pyramid levels can be controlled by different controllers. Processors on the same level co-operate in SIMD-mode whereas processors at different levels execute different programmes.

Various cost-factors must be considered when any one of the above mentioned operation autonomies is introduced. The simplest case where the 'activity' bit is introduced, the cost per processing element is one register and the manipulation of this register. For the multi-cluster operation autonomy, the cost is a multi-bit register and supporting logic. This is significantly higher compared to a bit-serial processor, but is, however, small for a multi-bit ALU. The implementation cost of the multiple SIMD operation autonomy is higher than the previous schemes because controllers that are programmable on all levels, are needed.

8.4.2.2 **Addressing Autonomy**

This type of autonomy allows each processing element in a SIMD-computer to generate an address locally or to change the broadcast address independently. It is then able to select an operand from its memory at a different location from all other elements. Many examples of SIMD - systems support addressing autonomy, for example CLIP7, GF11, WARP [213,94] and so on. Addressing autonomy has been very useful in vision computation.
Algorithms that take advantage of autonomous memory addressing, exist. The cost, however, much too high for this type of autonomy because of the large number of processing elements that must each have their own autonomy. However, limited forms of addressing autonomy can be implemented practically.

8.4.2.3 Connection Autonomy

The aim of this form of autonomy is to derive effectively a task-graph algorithm to be used by the network during execution, and to derive the underlying network architecture (tree, cube, pyramid etc.) Examples of multi-processor algorithms are the Minimum Spanning Tree Algorithm, Single-Source, Shortest Path and so on [213].

The graph-mapping is done via a network configuration that can be dynamically changed subject to the local conditions of a processor in a massively parallel environment. In the domain of massive parallel computers, connection autonomy may be quite different than in reconfigurable computers. This is especially true in the case of graph embedding due to the network topology, the controller and granularity.

Connection autonomy can be divided into two different fields:

- packet switched connection autonomy.
- circuit switched connection autonomy.

Networks in massively parallel architectures are usually seen as directly linked. In graph notation, each processor is a node or vertex and the direct links between nodes are called edges. The network is usually static which means that the graph representation remains the same during operation. To obtain efficient graph embedding a dynamic response of the network-representation during computer operation, is needed.

- Packet Switched Connection Autonomy

Each processing element in the system is allowed to send messages to any other processing element. The sender - processor simply states the destination of the message and a 'router' must then take responsibility for handling this message. The 'router' is to be found at every processor.

The source-processor prepares a message containing the address of the of the message's destination. The message is then sent into the local router. The router acts like the nodes in a LAN-environment: it sends and receives messages. When a router receives the message, it checks the destination address. If a match occurs, the message is accepted, otherwise it is forwarded to another router until the final destination is reached.

The store-and-forward packet switching process can be seen as building up the embedded graph by composing data-dependent sub-graphs of the graphs that correspond to the physical links among the routers.
Circuit-Switched Connection Autonomy

Packet-switched connection autonomy for a high degree network like a hypercube, has a high wiring cost. Added to this disadvantage, the greater flexibility of dynamically reconfiguring a network may not be needed for much of vision computation. Regular vision computation is characterized by fixed task-graphs. Because of this we must look at another type of connection autonomy.

In circuit-switched connection autonomy, interconnection paths between non-neighbour processors are dynamically created by the crossing of intermediate located processors. A target graph can then be constructed in terms of this definition of a neighbourhood-relationship in the network. This type of connection autonomy can be applied to any network.

The cost of the router in this type of connection autonomy is much lower than in packet switching autonomy. The advantage of circuit switched autonomy (connection) is its ability to embed regular graphs on a given interconnection network. This does not involve the expense of sending the address of the message's destination together with the message. Delays in transmission can, however, occur when a long chain of routers have to be crossed. Various solutions regarding engineering and packaging of the circuit board-layout can be adopted to keep this transmission delay to a minimum.

In the future, it may be possible to combine the benefit of both the packet-switched and circuit-switched connection autonomy in the same network.

8.5 SUMMARY

The aim of this chapter was to look briefly at the various parallel architectures being used today for the task of computer vision and image processing. It is in this field that computers that are several orders faster than today's fastest computers are required. Applications of parallelism in the 1990's demand high-level, high performance computers. Some applications, however, do not need to run on parallel computers while others do. Besides the formal research being done in image processing and vision on (massive) parallel computers, other applications include areas such as weather prediction, computational aerodynamics, artificial intelligence, remote sensing, military uses, nuclear reactor environments, and robotic space exploration.

In all of the abovementioned areas, computer vision and image processing play very important parts. For example in the area of remote sensing, earth-resource data received from satellites must be analyzed. These results are important for areas such as agriculture, geology, forestry and so on. Thus, the aim of this chapter was to obtain some idea of the type of architectures needed for the computationally expensive task of processing images effectively and quickly.

The aim was to study the older classification for parallel architectures and the newer sub-classification for SIMD-architectures. The main concepts of resolution on a multi-level and how they are implemented, have been examined. The various parallel algorithms that exist and have been developed for parallel processing, have been omitted, as it was felt that they fell outside the ambit of study of computer vision and image processing. These
algorithms include Numerical Algorithms like Gaussian Elimination on multiprocessors, Graph Algorithms (graph-searching strategies, shortest path, minimum spanning tree and so on.) and Combinatorial Search techniques (Divide and Conquer, Branch and Bound, Alpha-Beta Search and so on) [213].

It is clear that parallelism is here to stay. There are some arguments against its use but most of them are being refuted. It is also clear that parallel computers will in future play major roles in many applications falling under the main applications mentioned earlier.
INTRODUCTION

One dimensional grammars and two dimensional grammars and automata have been successfully used in various pattern recognition models [92,85,226,225,115]. For example, in [92], various one dimensional grammars are defined for the generation of languages that can be interpreted as languages describing two dimensional figures like squares and rectangles. In [92], examples of higher dimensional grammars are also given. Languages that are generated include languages describing different figures like house roofs and walls. Other special grammars that are presented are grammars defining binary concatenation relations between primitives. Picture description languages are defined for the description of letters and other two dimensional picture primitives.

In this chapter, we have mainly concentrated on the work done in higher dimensional automata theory. The standard work of one dimensional (string) grammars and automata has been omitted because it is fairly general and well known at this stage [232,231]. Instead, work on higher dimensional recognition automata is not very well known. For this reason, the automata models defined by Rosenfeld [84,237,240] and other authors were studied [92,225,227,229,234,85,236,221]. From this study, a framework model is represented in chapter #10 which also appears in [242].

9.1 TWO DIMENSIONAL GRAMMARS, LANGUAGES AND AUTOMATA - A BRIEF OVERVIEW

In this section, a brief overview of the two-dimensional analogs of the one-dimensional grammars, automata and languages discussed in section 9.1, is given. In this case, the input tape is a two-dimensional array of symbols. A finite number of these symbols are the blank symbol #. The symbols that are not blank, make up a connected sub-array.

Rosenfeld defines the two-dimensional accepting device in [84] as being a "bug" that is at any given time in a set of states and located at a certain given position on an array of symbols (the input-tape). The "bug" operates in a series of discrete time steps. At each time step, the "bug" does the following:

- Read the symbol in its current position, erases it and then replaces it by another symbol.
- It may then change to a new state.
- It then moves to a neighbouring position, or it does not move at all but remains in its current position.

An array automaton is defined in the same way as the one-dimensional automaton in [84], [229]. An array automaton is again a triple M = (Q,V,δ) where Q,V,δ are defined as in the one-dimensional case. The only difference being the possible move direction for M, that increases to Δ = {L,R,U,D,N} where L constitutes a "left" move on the input-array, while the R represents a "right" move on the input-array by M. N still constitutes the "no-move" as in the one-dimensional case while the U and D represents the move "upwards" an "downwards" on the input-tape respectively.
It is further pointed out by Rosenfeld, that this automaton is a Turing Machine if a finite number of symbols making up the set of input symbols V, is not blank symbols (non-#'s). V however does contain blank symbols (#) also. If M does not create or destroy #’s, M is called a # -preserving Turing Machine M is called a tape-bounded Turing Machine if M "bounces off the #’s". In other words, the tape has boundaries marked by blank symbols (#) and M moves between these boundaries on the two-dimensional tape. If M never rewrites any symbols, then M is called finite state automaton. Their exists a deterministic tape-bounded finite-state automaton A that accepts a connected array consisting of non-#'s if and only if is rectangular. For a proof of this theorem, the reader is referred to Rosenfeld [84].

The grammars of array acceptors:

- **Matrix Grammars**

These grammars generate sets of rectangular arrays. For a formal definition of a matrix grammar, the reader is referred to Rosenfeld [84]. If a grammar G is a finite-state array grammar, the language generated by G, is a finite state array language. Further results established by Rosenfeld in [84] are the class of languages for which G is a finite-state grammar, is a subclass of the class of finite-state array languages. The class of matrix languages generated by a matrix grammar G, is a proper subclass of the deterministic tape-bounded finite-state array languages.

If grammar G is a context-free grammar, the language-generated by G is not a finite-state language. The class of languages for which G is context-sensitive, is a proper subclass of the class of the tape-bounded array languages. The class of matrix languages is properly contained in the class of languages constituting the Turing rectangular array languages.

## 9.1.1 MATRIX GRAMMARS

A matrix grammar M, is a pair (G, G) where G is a grammar and $\overline{G} = \{G_1, \ldots, G_n\}$ is a set of finite-state grammars such that the terminal vocabulary of G is the set $\{S_1, \ldots, S_n\}$ of initial symbols of the Gi’s for $i = 1, \ldots, n$.

## 9.1.2 THE LANGUAGE GENERATED BY A MATRIX GRAMMAR

These grammars generate rectangular arrays. M generates a string consisting of S1’s using the rules of G. It then forms a rectangular array from top row (string) Θ, by applying the rules of the different Gi’s of G in parallel. M terminates concurrently.

## 9.1.3 THE ACCEPTING DEVICES OF THE LANGUAGES GENERATED BY M

- **Rectangular Array Acceptors**: Rectangular array languages can be accepted by a deterministic # - preserving acceptor. The acceptors move in 5 directions on the input-tape namely, a left move, right move, up move, down or no move.

- **The 3-way acceptor**: This acceptor moves only in three directions on the input tape: left, right, down but not up. This acceptor cannot verify whether is rectangular or not. The acceptor starts at the upper left corner of the input rectangular Σ.
The class of languages for which the grammar $G$ is finite-state matrix grammar is a proper subclass of the deterministic tape-bounded finite state array languages. The class of matrix languages for which $G$ is context-sensitive is a proper subclass of the tape-bounded array languages. The class of matrix languages is proper subclass of the class of Turing rectangular array languages.

9.1.4 ARRAY GRAMMARS

- An array grammar is a quintuple $G = (V, V_T, \delta, S, \#)$ where $V$ is a finite non-empty set of symbols, $V_T$ is the terminal set of symbols, $S \in V - V_T$, $S$ is the 'start' symbol and $\#$ is a set of pairs of connected arrays $(\Lambda, E)$ for all of which: a) $\Lambda$ and $E$ are geometrically identical. b) $\Lambda$ does not consist entirely of blank symbols ('#'), and c) $E$ satisfies the following conditions: 1) If the non-#'s do not touch the border of $\Lambda$, then the non-#'s of $E$ must be connected. 2) Otherwise, every connected component of non-#'s in $E$ must contain the intersection of some component of non-#'s in $\Lambda$ with the border of $\Lambda$.

- An array grammar is monotonic if #'s are never created by any rule. An array grammar $G$ is context-free if for rules $\Lambda \rightarrow E$ of $G$, $\Lambda$ consists of a single non-terminal symbol and #'s (possibly present). An array Grammar $G$ is linear if $G$ is context-free and any $E$ contains at most one non-terminal symbol. An array grammar $G$ is linear if $G$ is context-free and any $E$ contains at most one symbol.

9.1.5 LANGUAGES GENERATED BY ARRAY GRAMMARS

- The language generated by an array grammar $G$ namely $L(G)$, is the set of non-null connected arrays on $V$ such that an accepting array can be derived in $G$ from the initial array which consists of a single symbol namely $S$. If an array grammar $G$ is monotonic, a monotonic array language will be generated. If $G$ is context-free, the language generated by $G$ is monotonic. If $G$ is a linear array grammar, the language generated by $G$ will be linear.

- Languages created by an array grammar are the languages accepted by Turing acceptors. For example: there exists a deterministic Turing Acceptor that scan an input array $\Sigma$ starting from any point $p$ on $\Sigma$. It then returns to entirely of $p$ with all the markings made on in the course of the erased scan of $\Sigma$. A deterministic tape bounded acceptor will also scan an array $\Sigma$ starting from position $p$ on $\Sigma$. In the course of its scanning process symbols are erased and replaced by others.

- The class of languages accepted by the deterministic bounded cellular acceptors is written as $L(D)(B)C$. The class of pseudo-real time OIA languages is closed under reversal if and only if linear-time OIA's are equivalent to real-time OIA's.

- If the class of real time Cellular Array languages is closed under reversal, then it is also closed under concatenation. The language $L = \{0^n \ 1^m / m,n > = 1 \text{ and } m \text{ divides } n\}$ is an example of a real-time CA language.
There exists a hierarchy of pebble-languages that contains the finite-state languages. The finite-state languages are then contained in the tape-bounded languages. [84].

The finite union of bottom-up Pyramid Cellular Acceptor - languages, or intersection of bottom-up PCA languages, is a is a bottom-up PCA language. That also applies to the compliment, concatenation, reversal of a bottom-up PCA-language. A language L that must be recognised by a bottom-up Pyramid Cellular Acceptor, must be accepted by all sub-pyramids of that bottom-up P.C.A. Recognition is done separately on problems of \(2^n\) \(x\) \(2^n\) blocks of input and merged into a single solution of dimensions \(2^n\) \(x\) \(2^n\).

The languages of a parallel/sequential acceptor A is the set of arrays accepted by A. Languages accepted by parallel/sequential acceptors is the same as the languages accepted bounded cellular acceptors.

If the class of real-time OIA-languages is closed under reversal, it will also be closed under concatenation [229].

9.1.6 THE DIFFERENT ACCEPTING DEVICES OF THE LANGUAGES DESCRIBED IN 9.1.5

- Tape bounded Acceptors accepts monotonic array languages.
- The set of input arrays \(\Sigma\) accepted by a cellular array acceptor C, is called the language of C and denoted as \(L(C)\).
- A cellular array acceptor is a 2D automaton of an array of 'cells' which can individually be in some state at any given time. It operates in a series of time steps. At each step, every cell reads states of its 4 horizontal and 4 vertical neighbours and may then possibly change to a new state. For a formal definition, the reader is referred to [84]. Cellular array acceptors can simulate Turing Acceptors.
- Other types of cellular array acceptors include the one-way cellular array (OCA) and one-way iterative array (OIA) [229]. In an OCA, communication between cells is one-directional from left to right. The next state of a cell depends on its current state and the state of its neighbour (left) cell. The OCA goes into an accepting state if the right cell of the OCA enters an accepting state. In an OIA, the input is serially fed to the left most cell [229]. OIA's are equivalent to OCA's. Pseudo-real time OIA's are equivalent to linear-time OCA's and they are equivalent to real-time CA's.
- (Deterministic) (tape-bounded) acceptors can simulate deterministic (bounded) cellular acceptors and also vice versa: (deterministic) bounded acceptors can simulate deterministic (tape bounded) acceptors. In a deterministic bounded cellular acceptor: 1) the top left cell can identify itself because it is uniquely surrounded by blank symbols. The class of real-time CA language is closed under reversal if and only if linear-time CA's are equivalent to real-time time CA's [229]. 2) This cell consult other cells to verify whether or not they are in accepting states.
- Pyramid cellular acceptors are special types cellular acceptors. They can accept many languages in less than \(O(\text{diameter of the base})\) time. This acceptor consists of a stack of
cellular automata; the baselayer has a diameter of $2^{2n}$ cells. The next stacked layer has a diameter of $2^{2(n-1)}$ cells. The acceptor ends in a single layer consisting of a single cell usually called the apex cell [227,84]. There are thus $(n + 1)$ layers and the time needed for acceptance of an input array presented to the baselayer is $n = \log_2 (2^D)$ $O(\log(\text{diameter})$ of the input). Each cell in a PCA has nine neighbour cells namely four brother cells constituting cells on the same layer (say $k$) of a certain 4 son cells on the layer directly below the layer $k$ where the cell under discussion is, and the father cell found on the layer above the cell being discussed. The initial state of the base layer is determined by the input array presented to the layer. Other types of PCA's are the bottom-up acceptors [229]. Here, the direction of information flow is one directional: from the bottom to the top. Now, only the son cells of a cell in represents layer $k$, represents his neighbour cells. The next state of the cell depends only on its current state and the states of its four sons. Each cell of bottom-up PCA is defined as being an identical finite-state automaton [227], $M = (Q, Q_T, \delta, A)$ where $Q$ is a finite non-empty set of states, $Q_T \subseteq Q$ and is a finite set of states. $A \subseteq Q$ is the set of accepting states and $\delta: Q^3 \to Q$ is the state transition function, mapping the current set of states of $M$ and its four neighbour cells in a $2^{n-1} \times 2^{n-1}$ block on the layer below into $M$'s next state. Non-deterministic bottom-up PCA's can simulate non-deterministic two-dimensional on-line tessellating acceptors [227], [225]. $O(\text{diameter} \log(\text{diameter}))$ time is needed for deterministic bottom-up PCA's to simulate deterministic finite two-dimensional automata, and $O(\text{diameter})$ time is needed for deterministic bottom-up PCA's to simulate non-deterministic two-dimensional finite automata [227]. Non-deterministic PCA's are equivalent to cellular acceptors.

- Another type of PCA is the UP-DOWN PCA. In this PCA information flows up and down. However, information does not move side-way in the layers of this acceptor. An UP-DOWN PCA is equivalent to a PCA.

- Bottom-up triangular cellular acceptors can simulate finite-state acceptors in $\log(\text{real})$ time. Bottom-up PCA's can recognize are arrays that symmetrical around their central column.

- Other types of acceptors: Pebble Acceptors do not rewrite their input but put marks on it. At each point in time, a finite number of these marks exist on the input. Different degrees of pebble acceptors exist. A K-degree pebble acceptor A is a tape-bounded acceptor where its set of states $Q$ and input symbols (vocabulary) are of special form: $Q = Q' \times \{0,1\}^K$, $V = (V' \times \{0,1\}^K \cup \{\#\}$. A two-degree pebble acceptor can simulate a finite-state acceptor; a 3-degree pebble acceptor A can find the upper left corner of its input array $\Sigma$ and also the outside border of $\Sigma$.

- A parallel/sequential acceptor is a 9-tuple $A = (Q, q_0, Q_A, \delta, \#, V, \#, t, b, \mu)$ where $Q$ is a finite non-empty set of states, $q_0 \in Q$ is the initial state, $Q_A$ is a finite set of non-empty accepting states, $\# \in Q$ and is the blank state. $V$ is a finite non-empty set of symbols and is the vocabulary of the tape, $\#, \#b$ are blank symbols in $V$, $\delta: Q \times Q \times Q \times V \to 2^{Q \times V}$ and is the state-transition function, $\mu: Q \times V \to 2^{\{-1,0,1\}}$ is the movement function.

DISCUSSION
A **plex grammar** is defined in [92] and [234]. These grammars are of special interest to pattern recognition modelling, because they can generate languages where the terminals have some arbitrary amount of attaching points for connecting to other primitives or sub-patterns. The plex-grammar's primitives are usually referred to as **Non Attaching Point Entities**. Each production of the grammar is in context-free form (see section 9-2). These productions describe the connections between the primitives or sub-patterns. Lists of labelled concatenation points is used for this purpose. The sentences generated by the grammars are not labelled graphs. They can however be transformed by assigning labelled nodes to the primitives and concatenation points.

Rosenfeld [237] have extended the idea of the string-grammar to grammars for labelled graphs called **web-grammars**. Each production of a web-grammar describes the method of rewriting of a graph $\alpha$ into another graph $\beta$ and it also contains an "embedding" rule $E$. This rules describes the connection of to its surrounding graph $\beta$ or host-web, when the graph is rewritten.

A Web-grammar $G$ is a 4-tuple $G = (V_N, V_T, P, S)$ where

- $V_N$ is a finite non-empty set of non-terminals,
- $V_T$ is a finite empty set of terminals,
- $S$ is a set of "initial" webs,
- $P$ is a set of web productions defined as follows: $\alpha \rightarrow \beta, E$, where $\alpha$ and $\beta$ are webs and $E$ is an embedding of $\beta$. If the sub-web $\alpha$ of the web $\omega$ must be replaced by another sub-web $\beta$, we must specify how must be embedded in in the place of sub-web $\alpha$.

**Example of a web-grammar:**

Consider a web-grammar $G = (V_N, V_T, P, S)$ where

- $V_N = \{A\}$, $V_T = \{a,b,c\}$, $S = \{A\}$ and $P$: 1) $A \rightarrow a ; E = \{(p,a)/(p,A)\}$ is an edge in the host web; 2) $A \rightarrow a ; A, E$ is defined as in 1).

The resultant language generated by $G$, is the set of all webs $c$ of the form:

![Diagram of a web-grammar example](image)

A special case of a web grammar is the grammar in which $V_T$ consists of only a single symbol. Every point of every web in the language has the same label. This type of web-grammar is called a **graph-grammar**, with its language referred to as a graph language [85].

Other examples of grammars defined in section 9.2 and 9.3 are shape-grammars, tree grammars, and array grammars [85]. Shape-grammars generate texture. The grammar...
starts with an initial shape and repeatedly applies defined shape-rules. Different textures can be generated by shape-grammars. For example, a hexagonal texture can be generated by a shape grammar where a single rule of the grammar is repeatedly applied [85]. Three-grammars can also be used to generate texture [85]. Array grammars are used for hierarchical levels of resolution. Tree grammars are also used for this reason [85]. It involves the representation of hierarchical levels of resolution in texture. One level for example can describe the placement of repeating patterns in texture windows and another levels can describe texture elements in terms of pixels. In array grammars, the use of a blank symbol is important to ensure that rules are applied in appropriate contexts. An example of a simple grammar generating a checkerboard pattern is given in [85]. Thus, a powerful way of describing the rules that control textural structure, is to describe it by a grammar. The grammar defines certain rewrite-rules (productions) that are applied to symbols. Complex textures can be generated in this way.

In [226], shapes are decomposed so as to recognize their structure. The aim to establish a method for the construction of an effective system which would be able to learn by itself from examples. The model uses sample shapes to infer grammars, where each grammar describes a class of patterns. The semantics of these grammars are formulated automatically during the structural decomposition of the figure-shape. A context-free grammar which can generate a sample figure as well as a new figure are given. Furthermore, if a sample of shapes is given, grammars can be constructed which can generate this sample. It is pointed out that the representation of shapes is not rotation invariant. An algorithm is still valid after a small angle rotation was applied. If the given primitive's character description has changed due to a small angle-rotation, it is said that the primitive is sensitive to a small rotation.

In [225], a description of a deterministic two-dimensional on-line tessellating acceptor is given. It is stated that this acceptor is a real-time model of rectangular array bounded cellular automata. The (2-OTA) is defined as 7-tuple $M = (K, E^2, \delta, \#, q_e, q_o, F)$ where

1) $K$ is a finite set of states,
2) $E^2$ is the set of all 2-tuples of integers ,
3) $\Sigma$ is a finite set of input symbols ,
4) $\delta$ is the transition function of a cell state ,
5) $q_e \in K$ and us the motive state ,
6) $q_o \in K$ and is the 'start' state ,
7) $F \subset K - \{q_e, q_o\}$ is a set of final states.

The set of all two-dimensional tapes accepted by a 2-OTA $M = (K, E^2, \#, \delta, q_e, q_o, F)$ is defined as $T(M) = \{ x \in \Sigma^2 | \exists Z \in K - \{q_e, q_o\}, \text{where} x \in \Sigma^2 \}$. A run of $M$ on $x$ is a two-dimensional tape $Z$ over $K-\{q_e, q_o\}$, where $x \in \Sigma^2$. 

CHAPTER 9
In this section, the various two-dimensional analogs of the one-dimensional grammars and acceptors, was given. We also discussed some examples employed by various authors. In the next section, a discussion of three-dimensional automata will be given and how these automata are used as theoretical models for the recognition of three-dimensional structures.

9.2 THREE-DIMENSIONAL GRAMMARS AND AUTOMATA

In this section, the representation of digital structures in three dimensions is described. These structures are represented by three-dimensional grammars. This section is a brief overview of current methods that exist in terms of these grammars, to represent three-dimensional digital structures [236]. This section will also be used as a reference to chapter 10, where a theoretical model is proposed for the recognition of three-dimensional digital structures.

9.2.1 Digital structures

Because the objects that will have to be recognized by three-dimensional automata, are digital structures, it is necessary to define the idea of a three-dimensional digital structure. Rosenfeld [240] defines a three-dimensional digital structure as follows:

Definition of a three-dimensional digital structure:

A three-dimensional digital structure \( \Sigma \) has a fixed Cartesian coordinate system \((x,y,z)\), where every point \((x,y,z)\) in \( \Sigma \) is defined by these Cartesian coordinate system. Thus, a three-dimensional structure \( \Sigma \), is an arrangement of elements described by the three-dimensional Cartesian coordinate system \((x,y,z)\).

9.2.2 Random Context Structure Grammars and Automata

A Random Context Structure Grammar and Automaton is an extension of the two-dimensional grammars generating matrixes as their languages [221]. An RCSG can recognize a three-dimensional digital structure through the application of 7 directional structure contexts, which determine which production-rule defined by the RCSG, is applied.

From [221], a RCSG is defined as follows:

A Random Context Structure Grammar is a 4-tuple \( G = (V_N, V_T, P, S) \) where \( V_N \) is a finite non-empty set of non-terminal symbols, \( V_T \) a finite non-empty set of terminals, \( P \) is a finite set of production rules and \( S \in V_N \) is referred to as the 'start' symbol. The productions of \( P \) is described as being six different types:

1) \( A \rightarrow \alpha (U_1; T_1/U_2; T_2/U_3; T_3 / U_4; T_4/U_6; T_5/U_7; T_6/U_7; T_7) \)

2) \( A \leftarrow \alpha (U_1; T_1/U_2; T_2/U_3; T_3 / U_4; T_4/U_5; T_5/U_6; T_6/U_7; T_7) \)

3) \( A /\alpha (U_1; T_1/U_2; T_2/U_3; T_3 /U_4; T_4/U_5; T_5/U_7; T_6/U_7; T_7) \)

4) \( A \leftarrow \alpha (U_1; T_1/U_2; T_2/U_3; T_3 /U_4; T_4/U_5; T_5/U_6; T_6/U_7; T_7) \)
5) $A \rightarrow \alpha (U_1; T_1/U_2; T_2/U_3; T_3/U_4; T_4/U_5; T_5/U_6; T_6/U_7; T_7)$

6) $A \rightarrow \alpha (U_1; T_1/U_2; T_2/U_3; T_3/U_4; T_4/U_5; T_5/U_6; T_6/U_7; T_7)$

where $A \in \text{VN}$, $\alpha \in (\text{VN} \cup \text{VT})$, $U_i$, $T_i \in \text{VN}$ and where $1 \leq i \leq 7$, $U_i \cap T_i = \emptyset$ for $1 \leq i \leq 3$, $1 \leq |\alpha| \leq 2$. If $|\alpha| = 2$, the specific production is referred to as a growth-production. The structural contexts of a three-dimensional digital structure, are determined by the lines and planes going through a point in the structure. In figure 1, the different contexts that can be associated with a symbol in 3D digital space are shown.

![Figure 1](image)

Different Contexts That Can Be Associated With A Symbol In 3d Digital Space

These contexts are given in [221] as follows:

1) HL refers to the horizontal line context-set. The context-set is the set of symbols found on the line, through the specific symbol under consideration. This line is parallel to the X-axis in figure 1 and $U_1, T_1 \subseteq \text{HL}$.

2) VL refers to the vertical line context-set. This context-set is the set of symbols that can be found on the line going through the specific symbol in the 3D digital space. The line is parallel to the Y-axis and $U_2, T_2 \subseteq \text{VL}$.

3) DL refers to the depth line context-set consisting of all the symbols found on the line through the symbol in 3D digital space. The line is parallel to the Z-axis and $U_3, T_3 \subseteq \text{DL}$.

4) HP refers to horizontal plane-context. This context-set consists of all the symbols on the plane going through the symbol in 3D digital space, where this plane is parallel to the XY-plane, formed by the X-axis and Y-axis and $U_4, T_4 \subseteq \text{HP}$.

5) VP refers to the vertical plane context-set. This context-set consists of the set of symbols lying on the plane going through the specified symbol and is parallel to the XZ-plane and $U_5, T_5 \subseteq \text{VP}$.
6) DP refers to the dept-plane context-set. This set consists of all the symbols found on the plane going through the specified symbols and this plane is parallel to the YZ-plane and \( U_6, T_6 \subseteq DP \).

7) This context consists of all the symbols contained in the three-dimensional figure. The context is also being referred to as the global context.

Productions from \( P \) allow the replacement of symbols in six context-directions, which is given in [221] as:

- \( A \rightarrow \alpha \) east
- \( A \leftarrow \alpha \) west
- \( A \rightarrow \alpha \) north
- \( A \leftarrow \alpha \) south
- \( A \uparrow \alpha \) up
- \( A \downarrow \alpha \) down.

The rule specifications described earlier can be described in terms of the three-dimensional digital symbol \( A \) and the symbol \( \alpha \): If the symbol \( A \) appears in the three-dimensional digital structure, \( A \) can be replaced by the symbol \( \alpha \) if:

- the symbols contained in \( U_i \) and no symbols in \( T_i \), appears in the specific context determined by \( i \), where \( 1 \leq i \leq 7 \).

- \( |\alpha| = 2 \), the background symbol appears directly to the east of \( A \).

- \( |\alpha| = 1 \), \( A \) is replaced with \( \alpha \), determined by the condition set in (1).

In [221], it is shown that these grammars can be used to model chemical structures like the methane-molecule. This grammar can also generate three-dimensional digital structures which can be recognized by the automaton defined in [221].

This grammar can be used to construct an automaton recognizing three-dimensional digital structures through specific directional contexts that is considered before the automaton replaces one symbol by another [221].

9.2.3 Three-dimensional Plex Grammars

This grammar is an extension of the two-dimensional web-grammars defined by Rosenfeld in [237]. 3D-plex grammars [234] are used to describe surfaces of objects in three-dimensional space. This means that an object consists of a number of surfaces with edges, which describe the object. From [234], a non-attaching curve entity (NACE) is a surface in three-dimensional space that is surrounded by n-edges that attach it to another surface. The edges attaching this surface to another, is usually referred to as attaching curves [234].
A three-dimensional plex-grammar is formally defined in [234] as being a G-tuple \( G_p = \{ N, T, R, S, I, i_0 \} \) where

- \( N \) is the set of non-terminal n-ACE's,
- \( T \) is the set of terminal n-ACE's,
- \( R \) is the set of production rules, and \( S \) is the initial n-ACE or 'start' symbol where symbol refers to a surface in space.
- \( I \) is a finite set of identifiers, \( \cup (N \cup \Sigma) = \emptyset \), \( i_0 \) is called the null-identifier.

From [234], the set of production rules \( R \), is described as follows:

The rules of the production-set \( R \) are of the form:

\[
\Psi \rightarrow \Delta \omega
\]

where:

- \( \Psi \) refers to the component list of n-ACE-symbols on the lefthand-side,
- \( \omega \) refers to the component list of n-ACE-symbols on the righthand-side,
- \( \Gamma \) indicates the attachment between the two component lists and
- \( \Delta \) and \( \Delta \omega \) indicate how the edges on the left hand-side of the rules are mapped to the right hand-side of the rules.

The symbols of \( I \) identifies the attaching curves of NACE's. The null-identifier is a place identifier and is not associated with any attaching curve.

The rule-form described earlier, is that of an unrestricted 3D-plex grammar. The component lists are strings in the form \( \Psi = a_1 a_2 \ldots a_m \) and \( \omega = b_1 b_2 \ldots b_n \) where \( a_i \) and \( b_j \) are single NACE's called components.

3D-plex grammars can describe surfaces of objects in three-dimensional space. The surfaces can be related to each other through relational descriptions of the surfaces. Thus, these surfaces and its relational descriptions can then be used to model 3D objects. Through surface-relations being described by the plex-grammars, 3D objects can be recognized.
9.2.4 Graph Grammars

In section 9.4.2, Random Context Structure Grammars were investigated. RCS Grammars are formal methods for the recognition of three-dimensional digital structures. Symbols appeared in certain fixed positions on the digital structure.

When using graphs to represent the objects in three-dimensional space, certain geometrical aspects can be inferred on the graph. The graph consists of nodes and arcs. The arcs attach nodes to each other. The arcs of the graph, can represent certain geometrical information, for example the length of an arc or the angle with which the arc appears in space.

A graph grammar can be defined in terms of a set of arcs and a separate set of nodes. Elements of the set of arcs can be tuples which consists of two nodes from the set of nodes and the angle made by the arc attaching these two nodes [238,239]. These graphs can be used to describe the edges of surfaces of objects in three-dimensional space. This is accomplished, by specifying the names of the nodes as well as the angle made by the arc attaching the two nodes to each other. The tuples consists of three parameters: the name of the two nodes as well as the angle made by the arc attaching the nodes to each other.

A variation of a graph grammar, is a tree grammar which generates graphs with no cycle in the paths specifying the graphs.

In this section, the potential use of three-dimensional grammars and automata for the recognition of three-dimensional objects, were discussed briefly. A theoretical discussion of two three-dimensional grammars were given, as well as a functional description of graph grammars.

These grammars can be used to construct theoretical automata-models for three-dimensional object recognition. In chapter 10, such a model is intuitively defined.

9.3 SUMMARY

In this chapter, the one-dimensional grammars, languages and automata were discussed. These languages present mathematical models of languages and the systems used in language generation and processing.

By the introduction of more versatile interconnection specifications for the symbols in the productions of grammars, special picture grammars such as the two-dimensional grammars defined by Rosenfeld in [84] can be defined. These grammar's accepting devices accept two dimensional picture-languages. Several examples of these accepting devices were discussed in a tabled overview. These devices were formally defined and a functional description of their operational features were given. Examples include array acceptors, matrix acceptors, tessellating acceptors, cellular acceptors and so on. Array grammars, matrix grammars,web grammars, shape grammars and texture grammars were all theoretically discussed.

Three-dimensional grammars and automata were then investigated. It was stated that these grammars allow the construction of three-dimensional automata-models for the recognition of three-dimensional objects, whether these objects can be recognized in terms of fixed
structural context (Random Contexed Structure Grammars), surface descriptions and relations (3D-plex grammars) or the definition of object-edges with reference to geometrical orientation (graph-grammars).

From this work, a theoretical model is proposed [242] for the recognition of three-dimensional digital structures. The model is based on the Random Context Structure Automaton defined by Ehlers in [221] and the pyramid cellular acceptor defined by Rosenfeld in [84].
CHAPTER 10

A MODEL FOR THE VISUAL UNDERSTANDING OF THREE-DIMENSIONAL IMAGES
DISCUSSION

In this chapter we propose a formal model for the understanding of three-dimensional digital structures. We try to integrate the concepts of pyramid cellular acceptors and random context structure automata. We introduce three-dimensional context digital structure automata [242].

10.1. INTRODUCTION

Today, the mainstream of research in machine vision is in the fields of image processing, and pattern recognition. The main foci being object detection and object extraction. Less work has been done in the field of image understanding and specifically in the understanding of three-dimensional images. The aim of this chapter is to:

- Briefly comment on the theoretical foundations on which the model is based.
- Present a model for the visual understanding of three-dimensional images.

Research in computer visual understanding comprises the study of scene analysis. Scenes consist of objects with various relations between the objects. An object in itself can also be subdivided into sub-objects. Usually these sub-objects are called primitives. Image understanding involves the recognition of an object by identifying certain pre-defined primitives to be part of the object. Different methods to solve this problem have been investigated. Knowledge-based methods [71,224,223,228], grammatical methods [92,221], and Rosenfeld’s formal models for picture recognition [84]. In this paper, our aim will be to integrate the concepts of pyramid cellular acceptors and random context structure automata [84].

10.2 THEORETICAL BACKGROUND

In this section pyramid cellular acceptors and random context structure automata are briefly discussed.

10.2.1 Pyramid Cellular Acceptors

Informally, we can define a pyramid cellular acceptor as a pyramidal stack of arrays. The bottom layer consists of $2^n \times 2^n$ cells. This layer is usually also referred to as the baselayer. The next layer after the baselayer consists of $2^{n-1} \times 2^{n-1}$ cells. The $(n + 1)$st layer consists of one single cell, which is usually called the apex cell. Each cell in the pyramid has nine neighbours namely 4 brother cells on the same layer as itself, 4 sons on the layer beneath its layer and a father cell in the layer immediately above its own. If we restrict the pyramid so that information is passed from the baselayer up to the apex, in other words information-passing is strictly one-directional in the pyramid, we generalize by stating that a cell in layer $k$ receives inputs from its block of children cells at layer $k - 1$. Information is passed on from son-cells to a father cell in this manner and from there to its own father-cell. The information is finally passed on to the apex-cell which accepts the pattern or rejects it. Rosenfeld [84] suggests that if the bottom-up approach is used, the cell in layer $k$ will only
have 5 neighbours instead of 9. It is further pointed out that the bottom-up acceptor becomes periodic in polynomial time. This means that if the son of a father-cell in layer k (say) M changes its state at every r-th time interval, the father-cell will then change its state accordingly. The whole acceptor state-changing time will then became polynomial. A formal prove is given in Rosenfeld [84]. Rosenfeld also discusses the basic functions that can be carried by this specific type of receptor. These functions are identified as:

- detection of the presence/absence of a specific local pattern given as input to the baselayer of the pyramid.
- detecting arbitrary local patterns
- counting of local properties
- counting of arbitrary local properties in an image.

In later work that will only be referred to here, Rosenfeld [103], suggests various implementation for pyramid acceptors. The theoretical base of pyramid cellular acceptors find physical reality in applications which include:

- Statistical computation of properties
- Detection of large features and describing them.
- Encoding of curves, blobs and ribbons.
- Region Detection (Gradient/Chain Pyramid).

The pyramid-application is of course implemented on suitable pyramidal parallel hardware which include:

- Overlapping pyramids: Each node-cell has a 4 x 4 block of children nodes; the blocks overlap by 50%.
- Non-overlapping pyramids: Each node has a 2 x 2 block of children nodes below its own layer.

Thus, the theoretical base which is created for example in [68,84], can give rise to numerous applied projects of visual recognition. In our theoretical model, the pyramid is also represented with an input image at its baselayer. Each layer establishes a more abstract view of the image. The apex cell goes over into an accepting state for subsequent object-recognition. In the practical implementation, the establishment of a mere abstract view of the input pattern is usually referred to as establishing a reduced resolution version of the image. At the apex-node the resolution is reduced to a minimum and the apex-node is able to 'observe' the entire pattern which was presented to the baselayer ($2^n \times 2^n$ base nodes.) Each node in this layer sees only a small portion of the input pattern. The reduced resolution is usually done by averaging of the received grey level information from four sons in the layer below the father. The average grey level is then sent to its father cell.
together with three other graylevel averages from the neighbour nodes. (three neighbours existing in the same block of nodes and which has the same father-node.)

In the end, the apex cell is presented with an image that was averaged by all the nodes in the layers below it except the base nodes.

10.2.2 RANDOM CONTEXT STRUCTURE AUTOMATA

Earlier in this paper, we stated that we will be aiming to integrate the functional operations of the pyramidal cellular accepters and random context structure automata (RCSA's). The two-dimensional PCA must control the recognition process executed by the RCSA. This section briefly discusses the theoretical background an possible use of RCSA's for three-dimensional object recognition.

The significance of a RCSA is that it is able to recognize a three-dimensional (digital) structure by way of symbol substitution and boundary establishment. Recognition is done according to 7 restrictions or contexts that must be satisfied in order for subsequent successful three-dimensional structure recognition.

The automaton goes into three states before the final acceptance state of the automaton is reached [221]. The four states are as follows:

(1) initial boundary state
(2) structure bounding state
(3) read/write state
(4) accepting state

The automaton consists of an arbitrary rectangle of read/write heads. The automaton accepts only connected structures. The operational process is discussed briefly.

Boundaries of the three dimensional input structure are determined and a box-frame is drawn which holds the structure in the input tape. A rectangle (arbitrary) of read/write heads are active and moves toward the eastern boundary of the structure where it stops. The rectangle stops at the eastern boundary because background-symbols have been found after the rectangle moved over the eastern boundary of the structure. The rectangle moves between these two boundaries.

The rectangle now moves back step by step to the western boundary. The read/write heads scan the symbols. The automaton remembers the symbols that was scanned by the rectangle. The contexts are established. Upon reaching the western boundary contextual knowledge is retained by the rectangle which starts to move to the eastern boundary again. It will substitute what was read by the active read/write heads by another symbol. This substitution however depends on the contexts applicable for that symbol. Upon reaching the eastern boundary, the context is forgotten, it goes back into the read/write state. It will only go into the acceptance state if the final symbol appears together with the background symbol on the tape.
Summary of operation of the RCSA:

(1) The boundaries of the structure is established. Read/write rectangle of heads moves to the eastern-boundary where it stops.

(2) The input rectangle moves back to western boundary establishing contextual knowledge of symbol-appearances on various lines of the structure. Later, these contexts will determine the way in which one symbol can be substituted by another.

(3) The input rectangle moves eastward, substituting symbols read by the read/write heads, (read and remembered by them in (2)). Substitution is done strictly according to an established context in (2). Only one read/write head is active in the rectangle, during this state. Therefore, only one substitution-operation will be carried out according to the one specific context established by the read/write head in (2).

(4) If the final symbol appears (written on the tape) on the tape together with the background symbol, the RCSA goes into the acceptance state. In a copy state a current context is forgotten and the location of the specific read/write head in the rectangle also. In the read/write state, another head with its established contextual knowledge becomes active and carries out its substitution according to that established context.

In this way, we are able to recognize 3D-objects according to certain (directional) contexts which are applied. In total, the RCSA makes use of 7 different contexts for symbol substitution. They are

(1) Horizontal Line Context (HL)
(2) Vertical Line Context (VL)
(3) Depth Line Context (DL)
(4) Horizontal Plane Context (HP)
(5) Vertical Plane Context (VP)
(6) Depth Plane Context (DP)
(7) Global Plane Context

Any of these contexts can be established by the current active read/write head in (3). The context are established according to symbol appearances on the geometrical planes and lines of the structure. In figure 1, chapter 9, we show the concept of the RCSA together with its structural contexts.

Possible uses of RCSA's:

- Generation of 3D representations of chemical molecules, for example the Methane molecule CH₄.
• Modelling of connected digital structures e.g. pyramid, boxes, cubes.

10.2.3 THREE-DIMENSIONAL CONTEXT DIGITAL STRUCTURE AUTOMATA (3DCDS)

The main concepts of a 3DCDS are illustrated in figure 1.

A 3DCDS consists of:

- An Extended RCSA,
- A three-dimensional input tape (potentially infinite),
- A PCA connected to the input-rectangle of the RCSA.

The input-rectangle probes the three-dimensional input tape. The read/write heads established the different image parts of the scene and collects information on the context of the parts. The pyramid then becomes active. Each level establishes a more abstract view of the scene until finally the automaton "understands" the scene.

The following discussion of 3DCDS's will be informal and an intuitive idea of the automata will be given. 3DCDS's accepts only scenes consisting of connected three-dimensional digital structures. (For a discussion of digital structures the reader is referred to Rosenfeld [235]. A 3DCDS can be in any one of 4 main categories of states:...
1) Initial bounding state

2) Structure bounding state

3) Identification state

4) Recognition state

1). Initial bounding state

Directly from the initial state the automaton goes into a bounding state and an arbitrary rectangle of read/write heads become active. While in his category of states the RCSA establishes boundaries for the 3-dimensional scene. It draws a box-frame which encompasses the input structure on the input tape and the rectangle eventually stops on the eastern boundary. The rectangle of active read/write heads has been determined and will be fixed for the rest of the operation of the automaton. The input-rectangle will now move (or probe) only between the eastern and western boundaries.

2). Structure bounding state

The 3DCDS now goes into a structure bounding state. During this state the 3-dimensional input tape is divided into a number of boxes. Each box potentially contains a digital structure which can be recognized by the automaton. This is accomplished by the input rectangle moving west step by step. Each active read/write head remembers the position of the first non-background symbol it encounters and the position of the next background symbol in its line of movement. The automaton goes into a copy state and returns to the eastern boundary. The information is fed into the pyramid and the input tape subsequently divided into boxes which contains the structures in the "foreground". This is done by dividing the input-rectangle into a number of sub-rectangles each of which corresponds with a specific box on the input tape.

3) Identification state

During this state the pyramid sends a message to one of the identified sub-rectangles to identify the structure in its given cube as one of the structures known to the 3DCDS. This is done on the same principle as an RCSA works. The RCSA denoted by the sub-rectangle either accepts the digital structure of it or it does not accept the structure. After a certain time period has elapsed it is again prompted by the pyramid to see if it is not a different structure known to the automaton. As soon as the RCSA accepts a structure the information is fed into the pyramid and that box is wiped clean. The recognition is done in the same way as the RCSA usually accepts the digital structure.

This is done for each of the boxes identified in the previous structure bounding state. The 3DCDS now has knowledge of the structures in the foreground. The automaton returns to a structure bounding state. The whole process is repeated until the input tape is empty. The 3DCD then goes into a recognition state.
4). Recognition state

When the input tape does not contain any more structures it will mean that all the boxes have been cleaned. It will also mean that the automaton identified all the structures in all the sub-rectangles which was suggested by the pyramid. The automaton will only accept a structure if the symbol appears together with a background symbol in a certain context. This context is defined in terms of the appearance of a symbol in the three-dimensional structure of the input-image.

10.3 SUMMARY

In this chapter, a theoretical model for the recognition of three-dimensional digital structures was proposed. The model is based on the three-dimensional automaton defined by Ehlers and von Solms [221] and the Pyramid Acceptor defined by Rosenfeld. As stated before, this model’s framework is presented here. It is now, intended to fully develop this model in future research in the following way:

- Establish the model fully in a theoretical way.
- Implement the model in a practical environment.

The knowledge contained in the earlier chapters, must then be used as material for the practical implementation phase.
CONCLUSIONS AND FUTURE RESEARCH
DISCUSSION:

In this chapter, a brief summary of the studied work are given, as well as conclusions drawn from the study.

In CHAPTER 2, the mechanisms of organic vision were investigated. The field of organic vision has special significance for research in computer vision: most of the work being done in computer vision is concerned with the design of systems which will emulate the human visual system. It is generally known what happens to a light beam striking the retina and ending up somewhere in the striate cortex. What is not known, is what "algorithms" govern this transformation of light into coded electrical impulses. These electrical impulses contain all the information that will later be reconstructed into a three-dimensional image of the scene that was observed in the external world. The "program" controlling this reconstruction is unknown. Recent work done on the function of the human memory aims at understanding the language 'spoken' between the billions of neurons in the human brain. It is suggested that this will shed new light on the algorithmic processes at work not only in the visual cortex but in the whole of the human brain. In chapter 2, the visual pathway of light in the human vision system was investigated. The biological structure of the human focussing device as well as some of its components were presented as a backdrop to the discussion of visual computation in humans.

In CHAPTER 3, various network models which were designed to emulate the neuronic network of the visual cortex, were investigated. These models were able to perform certain visual recognition tasks. However, it is clear that they are an attempt to establish some understanding of the visual recognition process in humans. They are not the final solution in this regard. Later in the chapter, other types of network models were investigated. These networks are an attempt to model the neuronic network of the brain. Various types of these networks were described based on the type of input they can receive. Some work done in computer vision using these networks were presented.

The following two chapters discussed mathematical aspects involved in image-processing and the various techniques being used to form an image were also addressed. Then, the various geometrical properties of this image are presented. These properties can be analysed in a mathematical way.

In CHAPTER 6, the role Artificial Intelligence play in computer vision is investigated. This is done from a theoretical point-of-view and subsequently also from an application point of view. Different knowledge representation schemes are investigated, and also analysed in terms of their representational effectiveness of knowledge-based facts concerning domains of interest. Various developed systems making use of some kind of AI-paradigm for computer vision tasks, are discussed briefly. All these systems employ some kind of knowledge representation scheme in order to perform a visual recognition of an object. A general feature of the knowledge-based approach is that it is only suitable for a small domain of interest. Once a task requires knowledge falling outside the domain that is covered by the knowledge, the task is performed poorly. The reason for this is that the knowledge-base contains knowledge that is very specific in its task-description. The knowledge-base is geared towards the solution of a specific task. Our visual system draws on vast amounts of knowledge about the external world. This enables us to understand the visual data received by us in an effortless way. As yet there exists no system capable
of containing this vast amount of general knowledge in a representable form. However, the idea of using knowledge as the means for image understanding has substantial merit because of the example given by the human visual system. Current and future research in these fields (Artificial Intelligence and Computer Vision) will be geared towards the establishment of techniques for the representation of these large amounts of knowledge used for the understanding of visual information. Currently the content of the knowledge being used for image understanding is highly application orientated. Future research will have to concentrate on making the knowledge domain more general.

CHAPTER 7 investigated the various manufacturing applications that are currently using computer vision as a method to automate the manufacturing process. These applications include the inspection of manufactured items and object surfaces, automated and electronic assembly, bin-picking, and robotic navigation. Some of these applications also use a knowledge-based approach to solve a specific manufacturing task. The chapter also discusses various requirements that a vision system must have in order to be effectively used in an industrial environment.

CHAPTER 8 investigates the need for using parallelism in computer vision. Various parallel architectures are discussed. The use of parallel processing architectures in computer vision tasks is relevant because of two reasons:

(1) The computational operations being carried out in the (biological) visual cortex, is done in a parallel way. The motivation of using parallelism in computer vision tasks has a biological merit.

(2) Because vision computation is very expensive in terms of processing, the traditional sequential way of processing data makes it unsuitable for use in the computer vision environment. The sequential processor is not suitable for the effective processing of the substantial amount of information involved in a computer vision task. The only alternative is the use of parallelism, where hundreds or perhaps thousands of processors operate in parallel to complete a vision task.

CHAPTERS 9 and 10 investigated the role formal language and automata-theory plays in the area of computer vision. From the traditionally known string grammars and their automata, the investigation proceeds to the two-dimensional acceptors defined by Rosenfeld. These acceptors were especially defined as formal theoretical models for picture recognition. With the advent of improved hardware technology, many of these abstract models were realised in practical implementations. Examples include the pyramid cellular acceptor and various other cellular acceptors. Three-dimensional automata are then investigated. These are formal theoretical models for the recognition of three-dimensional digital images. A special type of three-dimensional grammar, which can represent structures in three dimensions, is investigated. The automaton of this grammar replaces symbols along six edges of a three-dimensional figure in order to recognise in three-dimensional contexts. This grammar, referred to as a Random Context Structure Grammar, can be used to model the structure of chemical molecules. Other three-dimensional grammars investigated include shape grammars which can decompose shapes of objects and in this way recognise the object. The recognition is possible because certain geometrical features of an object determine the shape of the object. This grammar is used to determine the shape using geometrical features such as curves, holes, and so on. Another three-dimen-
sional grammar that was investigated is the three dimensional plex grammar. This grammar uses surfaces of objects to describe the objects in three dimensional space. The object is described in terms of different surface relations of the object.

In **CHAPTER 10** an automaton model for the recognition of three-dimensional digital structures is proposed. This model draws on the work done in defining Random Context Structured Automata and the special two-dimensional Pyramid Cellular Acceptors defined by Rosenfeld. The automaton will recognise a three-dimensional digital structure in terms of certain directional structure contexts defined by the Random Context Structured Grammar. The context, however, will be stored inside the Pyramid Cellular Acceptor. It is further planned to include the recognition of objects in three dimensions by way of surface-context recognition. It is the intention of further research, to fully develop this recognition model and to implement this model in a practical environment. This will entail a far wider knowledge base of computer vision than merely formal languages and automata-theory and must include much of the work studied in the earlier chapters.
REFERENCES

[1] - [63]


REFERENCES


[34] Dr. James Clark Solinsky, a Generalised Feature Extraction Approach, Vision '87, Conference Proceedings.


[38] Roland Lans, Robot Guidance for Handling of Engine Blocks from Pallet to Machining Station, Vision '87, Conference Proceedings.


[40] Frederick P. Andresen, Scott Roth, Bruce Shimans, A Simple and Versatile Inspection Tool, Vision '87, Conference Proceedings.


[58] Robert Lozar, Procedure To Generate a Knowledge Base for Robotic Pattern Recognition, Vision '87, Conference Proceedings.


[73] Yuji Sato, K. Takahashi, Data Compression of Gray-Scale Images by Level plane Coding.


[86] M. Herman, Representation of Incremental Construction of a three-dimensional Scene Model.


[88] D.T. Kuan, R.J. Drazovich, Model Based Interpretation of 3D Range Data.


[92] K.S. Fu, Syntactic Pattern Recognition. Appeared in Digital Pattern Recognition, Edited by K.S. Fu.

[93] A Rozenfeld and J.S. Weszka, Picture Recognition, Appeared in Digital Pattern Recognition, Appeared in [92].


REFERENCES


[104] Azriel Rosenfeld, SEMINAR: Topics in Computer Vision, University of the Witwatersrand, 7-9 May 1990.


[111] Ks. Fu, Recent Progress in Pattern Recognition, Appeared in Progress in Pattern Recognition, Edited by A. Rosenfeld, North-Holland, 1981 (Volume 1).


[113] K. Preston Jr., Image Manipulation Languages; a preliminary Survey.


[115] K.S. Fu, Syntactic Models in pattern Recognition and applications.

[116] Perceptual World, Readings from Scientific American, Edited by Irvin Rock.


REFERENCES


[117] Intelligence: the eye, the brain, the computer. Martin A. Fischler, Oscar Firschein, Addison-Wesley, 1987


REFERENCES


REFERENCES


REFERENCES


