

BANKNOTE RECOGNITION USING INDUCTIVE LEARNING

by

Mehmet ŞEVKLİ

BS. in I.E., Sakarya University, 1997

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
The requirements for the degree of
Master of Science
in

Industrial Engineering

92741

Fatih University

July 2000

2000.07
FATİH ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHÜRÜ
MERKEZİ



Fatih Üniversitesi

Tarih: 17 / 07 / 2000

Fen Bilimleri Enstitüsü Müdürlüğü'ne

TUTANAK

Mehmet...SEKİ... 'a ait

"...Banknote...Recognition...Using...Inductive...Learning..."

adlı çalışma ...50... dk.'lık süre içinde savunulmuş ve jüri tarafından

Endüstri...Müh... Anabilim Dalında *YÜKSEK LİSANS TEZİ* olarak oy

birliğiyle / ~~oy çokluğuyla~~ kabul edilmiştir / ~~edilmemiştir~~.

~~..... yarınl ek süre verilmiştir.~~

Başkan Doc...Dr...M. Sabih...AKSOY

Üye Prof...Dr...Mazhar...ÜNSAL

Üye Yrd...Doc...Dr...Avcullah...ARSLAN

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis supervisor Assoc. Prof. Dr. M. Sabih AKSOY for his motivation and willpower which kept me doing all the work. His academic activities will be references in my future studies.

I would like to thank Prof. Dr. Mazhar UNSAL for participating in my thesis committee.

I would like to thank Assist. Prof. Dr. Nurullah ARSLAN for his kindness and his comments on the work done.

I would like to thank my best friends Özgür UYSAL and Ali TURKYILMAZ who always kept me going on

I would like to thank my research assistant friends, for their valuable helps.



ABSTRACT

In this thesis we propose an alternative technique to banknote recognition. The technique employs the RULES-3 inductive learning algorithm to extract the necessary set of rules and recognizes a banknote. 10 of 3x3 masks were used to represent a banknote. Each banknote (or example) consists of 10 frequencies of each mask. The system was trained for 5 different types of banknotes' back and front sides by using 10 samples for each type and then tested for 25 "new" examples. The efficiency of the system, that is the ability of correctly classifying new examples, is found to be 96%.



ÖZET

Bu tezde, para tanıma için alternatif bir teknik geliştirilmiştir. Bu teknikte gerekli kural setini çıkarma ve tanıma işlevi *Rules-3 Endüktif Öğrenme* yoluyla yapılmaktadır. Banknotu temsil etmek için 10 tane 3x3 mask kullanılmıştır. Herbir banknot(veya örnek) 10 adet maskın frekans sayısını içermektedir. Sistem 5 adet değişik paranın arka ve ön yüzü her bir banknottan 10'ar adet kullanılarak öğretilmiştir. Daha sonra sisteme öğretilmemiş 25 adet yeni banknot test edildi. Sistemin etkinliği doğru sınıflama açısından ölçüldüğünde 96% olarak bulunmuştur.



TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT.....	iv
ÖZET	v
TABLE OF CONTENTS	vi
LIST OF TABLES.....	ix
LIST OF FIGURES.....	x
1 INTRODUCTION	1
1.1 Problem Statement	1
1.2 Outline of the Thesis.....	2
2 BANKNOTE RECOGNITION SYSTEMS AND MACHINE VISION.....	3
2.1 Banknote Recognition Techniques.....	3
2.2 What is Machine Vision?	6
2.2.1 Why Has Machine Vision Emerged Now?.....	7
2.2.2 Why does industry need machine vision?.....	8
2.2.3 Application Areas	9
2.3 Why Use Vision?	9
2.3.1 Safety and Reliability.....	10
2.3.2 Product Quality	11
2.3.3 Flexible Automation	12
2.4 Advantages in Using Vision	13
3 IMAGE PROCESSING	16
3.1 Binary Images	16
3.2 Gray-Scale Images	17
3.3 Color Image.....	17
3.4 Correcting Imaging Defects	18
3.5 Noisy images	19

3.6	Spatial filters	20
3.7	Image Quantisation.....	21
3.8	Segmentation	22
3.9	Thresholding.....	23
3.10	Edge line detection	27
3.10.1	Roberts Operator.....	28
3.10.2	Sobel Operator	28
3.10.3	Prewitt Operator.....	29
3.10.4	Kirsch Compass Masks.....	30
4	PATTERN CLASSIFICATION.....	32
4.1	Statistical Image (Pattern) Recognition.....	33
4.1.1	Parametric Methods	33
4.1.1.1	Bayes Decision Theory.....	33
4.1.2	Non Parametric Methods	34
4.1.2.1	K Nearest Neighbor Classifier (KNN).....	35
4.1.3	Fuzzy KNN Classifier.....	36
4.2	Artificial Neural Networks.....	36
4.2.1	Adaline Algorithm	37
4.2.2	Learning Vector Quantization (LVQ).....	38
4.3	Inductive Learning	39
4.3.1	Categories of Inductive Learning Methods	40
4.3.1.1	Divide-and-Conquer Induction Algorithms.....	41
4.3.1.2	RULES-3 Inductive Learning Algorithm	47
5	IMPLEMENTATION	52
5.1	The Operation of the New Technique.....	52
5.1.1	Edge Detection.....	54
5.1.1.1	Snapping the Banknote	55
5.1.1.2	Increasing the Contrast Level of the Image.....	55
5.1.1.3	Performing Noise Reduction (image filtering)	56
5.1.1.4	Converting to a binary image.....	57

5.1.2	Learning Session.....	59
5.1.3	Recognition of Banknotes.....	64
6	CONCLUSION	66
7	APPENDIX.....	67
7.1	Files In the Disk.....	67
7.1.1	Main.m (Main Program).....	67
7.1.2	Fmask.m (Finding Mask Frequency).....	68
7.1.3	Cercsmall.m (Cutting Edges of Image)	70
7.2	Hardware Requirements.....	71
7.3	Software Requirements.....	72
	REFERENCES.....	73
	REFERENCES NOT CITED.....	80

LIST OF TABLES

TABLE 4-1 TRAINING SET FOR THE CLS-1 EXAMPLE PROBLEM.....	43
TABLE 4-2 TRAINING SET FOR CAR ACCELERATION PROBLEM ADAPTED FROM [65]	49
TABLE 4-3 SUMMARY OF RULE INDUCTION SEQUENCE FOR CAR ACCELERATION PROBLEM.....	51
TABLE 5-1 THE WHOLE BANKNOTES USED IN TURKEY.....	53
TABLE 5-2 THE SET OF 10, 3X3 MASKS USED IN THE THESIS.....	54
TABLE 5-3 THE SET OF 28, 3X3 NEIGHBORS	54
TABLE 5-4 COMPARING FILTER TYPE.....	57
TABLE 5-5 SET OF EXAMPLES PART A.....	60
TABLE 5-6 SET OF EXAMPLES PART B	61
TABLE 5-7 EXTRACTED SET OF RULES BY RULES-3 FOR THE BANKNOTES SHOWN IN TABLE 5-1 PART A	62
TABLE 5-8 EXTRACTED SET OF RULES BY RULES-3 FOR THE BANKNOTES SHOWN IN TABLE 5-1 PART B	63
TABLE 5-9 EXAMPLES TO BE RECOGNIZED	65

LIST OF FIGURES

FIGURE 2-1 THE CATALOGUE OF CURRENCIES FOR WINDOWS	4
FIGURE 2-2 SODECO BNA5 MACHINE	6
FIGURE 3-1 IMAGE TO BE THRESHOLDED	24
FIGURE 3-2 BRIGHTNESS HISTOGRAM OF THE IMAGE	25
FIGURE 3-3 THE TRIANGLE ALGORITHM IS BASED ON FINDING THE VALUE OF B THAT GIVES THE MAXIMUM DISTANCE D.	26
FIGURE 4-1 DISTRIBUTION OF TWO CLASS DATA, AND CLASSIFICATION EXAMPLE.....	32
FIGURE 4-2 SINGLE NEURON SYSTEM.....	37
FIGURE 4-3 AN EXAMPLE DECISION TREE	40
FIGURE 5-1 HOW THE SYSTEM WORKS	52
FIGURE 5-2 GRAY LEVEL OF IMAGE	55
FIGURE 5-3 CONTRAST OF GRAY LEVEL IMAGE	55
FIGURE 5-4 HISTOGRAM OF GRAY LEVEL OF IMAGE	56
FIGURE 5-5 LINEAR FILTER	56
FIGURE 5-6 MEDIAN FILTER	57
FIGURE 5-7 BINARY IMAGE	58
FIGURE 5-8 3X3 NEIGHBORHOOD	58
FIGURE 5-9 EDGE IMAGE.....	58

1 INTRODUCTION

Robotic applications are becoming more and more important in every life. New technology has brought some new needs together like high speed, low costs, and easy reach to this technology. All firms try to find some new methods to provide satisfaction to their customers.

1.1 Problem Statement

Today, Banknotes are the still the most widespread means of payment. Banknotes are the main means used in personal transactions. They are state-controlled and usable by everyone, including children. They are a safe means of payment, without risk, without upper or lower quantity limits. They are still the preferred means of payment for small or medium sums, easy to use, and universally accepted, and, as they give no personal information, they protect the user's anonymity.

Automatic cash payment is currently on the increase in several market sectors, particularly those where 24-hour availability of goods and services is important, for example in public transport, petrol stations, car parks, vending machines, etc. Automation allows the operator to make substantial savings on salaries and space. Customer service and satisfaction are of prime concern in these sectors.

Banks use ATMs (Automatic Teller Machine) to provide unlimited and fast services to their customers to let them get what they need using those machines themselves. Besides that, some firms sell their products using those machines.

When someone wants to deposit some money into his account using ATM, he should put it in an envelope and leave it to the machine. Afterwards a human being has to control for how much money is in the envelope and put it to the account. What we propose here is to have a unit in an ATM, which can recognize the banknote without human. If a scanner or a camera is installed somewhere in ATM, and when a banknote is fed to the machine it will snap the banknote and send it to the visual inspection system to have a decision what banknote is. The decision is given using Rules-3 inductive learning system.

1.2 Outline of the Thesis

In this thesis we propose an alternative technique for banknote recognition. The technique employs RULES-3 inductive learning algorithm to extract the necessary set of rules and to recognize a banknote.

Two banknote recognition systems are reviewed in chapter two. These are *Catalogue of Currencies for Windows* and *Sodeco Banknote Processing System*. In addition, in this chapter, the definition, application areas and advantages of machine vision are explained.

In chapter three, basic image-processing stages are examined. These include thresholding, noise reduction (e.g. linear, median filtering) and edge detection (e.g. Sobel, Roberts, Laplacian and Prewitt Operators) techniques.

In chapter four, several methods for Pattern Classification are briefly described. Statistical Pattern Recognition (Parametric, Non Parametric and Fuzzy KNN Classifier Techniques), Artificial Neural Networks (Adaline Algorithm, Learning Vector Quantization Methods) and Inductive Learning Techniques (Divide-and-Conquer, RULES-3 Algorithms) are among these.

Chapter Five focuses on the implementation stages of the study. All the steps of the banknote-recognition process are described in detail and the application of the induction algorithm to the recognition process is emphasized.

The thesis comes to an end with the Conclusion Chapter, where the application areas of the proposed system are mentioned and improvement possibility related to this study are unveiled.

2 BANKNOTE RECOGNITION SYSTEMS AND MACHINE VISION

In this chapter banknote recognition techniques and why use machine vision are explained.

2.1 Banknote Recognition Techniques

There are several techniques for banknote recognition in this chapter. Two of them will be explained. These are the catalogue of currencies for windows and Sodeco banknote processing systems.

The catalogue of currencies for windows gives a list of the countries, the country legal tender and preclusion banknotes, coins or/and the traveler's cheques; displays a full color size of the banknote, coin and cheque from its front or back side (coin also from its edge side); displays a color particular security feature detail with instruction of its recognition, detection and location on the banknote or cheque; displays the full color size of the banknote under UV (ultraviolet) and IR (infra-red) light with relevant description; gives a complete description of the entire banknote or coin or cheque and its security features; automatically tracks the currency legal tender ; displays a full color size of known often occurring counterfeits and their description; displays the main notable marks of the counterfeit versus the genuine banknote; gives a security features guide for recognition and detection of the banknotes, coins and/or cheques; tracks stolen banknotes and/or cheques; customizes internal rules; gives the catalogue settings (stand-alone or network installation, manual or automatic update, maintain, and so on); The Catalogue benefits can be explained as follows;

Protects the institution from financial harm; the tellers are able to catch even the excellent counterfeits;

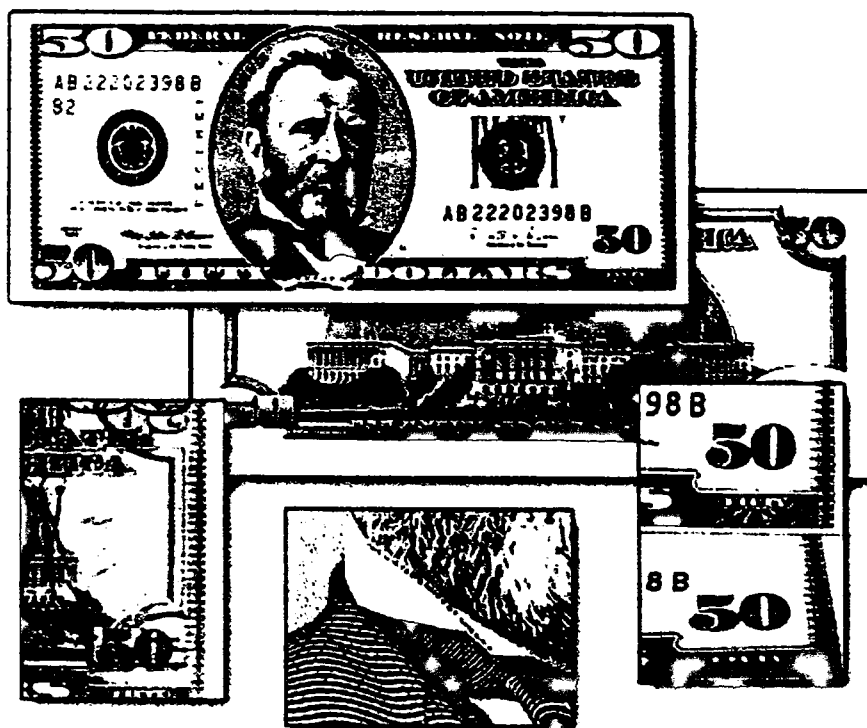


Figure 2-1 The Catalogue of Currencies for Windows

Better service to the clients; the checking with the Catalogue does not frustrate the clients;

Ability to customize institution currency; the chief cashiers can manage internal currency rules;

Protect the reputation of the institution; the tellers work with relevant information;

Readiness to manage parallel runs of the EURO currency and national currencies;

Ability for central installation; central management of the Catalogue of Currencies.

Characteristic features of the Catalogue of currencies are as follows:

The possibility to obtain an information about the tender or its forgery at the very moment when needed by the operator at the desk. Quick, operative access to the relevant information about the tender (in reality, information highway between eminent of the currency and the desk operator, where an up to date and complex information on currency can protect operators at the desk against purchase of currency not valid or forged. On the

other side, the high professional soundness of operators makes a good name to the banking house and improves its credit at his customers).

Catalogue of currencies enables to keep confidentiality against the customer (The customer, standing at the back of the monitor, cannot see the monitor screen during traveler's check exchange. He cannot guess that an operative is using computer to check validity of his banknotes. The checking does not frustrate the customer).

The operative at the desk need not leave the desk from the reason of checking validity of the banknote; this also increases the credit of the banking house against the customer.

The system is updated continuously (diskettes, Internet) about new information on currency or any important information related to problem of protection against money falsifiers, latest information, experiences, methods of protection (for instance, the method of protection of checks and cards after Frank Abagnale, a world known expert on protection of the checks and cards). [1]

The other system is Sodeco Banknote Processing System

Sodeco Banknote Processing System can process 2, 4, 6, or 13 notes inserted in any of four ways and will be available in more than 120 denominations/currencies for notes 60 to 86 mm wide by 95 to 172 mm long.

Acceptance rate: Exceptionally high (97-99% and constant over time)

Security: Effective protection against fraud theft and vandalism

Profitability: Durability + reliability + availability = Low life-cycle cost

Main function of the validator provides the following functions:

Acceptance of several different denominations inserted in any direction thanks to the ergonomically designed opening. A single slot is enough for all note widths from 60 to 86 mm. There are exceptions such as the USA variant in which the only accepted width is 66 mm.

Validator power supply, operation control by the automat with the possibility of remote control.

A banknote has its own peculiar size, texture and printing style. Size and printing style are the main criteria for automatic validation procedures.

Banknote size is checked by light barriers. The width is checked by a light barrier at the insertion slot; an appropriate signal causes the shutter to open, the motor to start and the light to turn red once the note is inside the validator. Other sensors check the length during conveyance. The most elaborate part of the validation technique is spectral analysis. This test uses several spectral measures processed by a sophisticated software program and provides an exceptionally high acceptance rate while ensuring excellent security to avoid fraud and forgeries. The banknotes are measured at various points on both sides.

In Spectral analysis, two opto-electronic measuring heads measure the printing ink response to light emissions. The results are compared with reference data. [2]



Figure 2-2 Sodeco BNA5 Machine

2.2 What is Machine Vision?

The definition of “machine vision” adopted by the Machine Vision Association of the Society of Manufacturing Engineers is: “the use of devices for optical non-contact sensing to automatically receive and interpret an image of a real scene, in order to obtain information and/or control machines or processes.” A description which often satisfies a non-technical enquirer, is simply ‘a TV camera linked to a computer.’ though this can hardly be considered a formal definition.

The MVA/SME definition has several key phrases, which help to narrow the definition and distinguish machine vision from subjects with which it is often confused. Notice that it talks of a real scene; this distinguishes it from the field of computer graphics, where the objective is to display a representation of a scene based on stored or computed data which is already in digitized form. Many of the problems of machine vision lie in

translating a real scene to a digital representation. The second point to note is the requirement for automatic interpretation. Image enhancement is a process that makes an image easier for a human to interpret-it may well be a step in a machine vision process, but the end result of a machine vision operation is not an image. It is an numerical or logical result which may be simply recorded, or used as the basis for further action.

It is easy to forget that the system is interpreting an image of a scene; it has no knowledge of the objects in the scene except those characteristics, which appear in the image. Except in some rather special cases, an image is a three-dimensional representation of what is probably a three-dimensional scene, and this means that some information about the scene is not available for interpretation, or must be inferred from additional images or other sensors. [3]

The use of the word “optical” could be an unnecessary restriction on the definition of machine vision. Because images can be formed by beams of electrons or acoustic waves, or by electromagnetic radiation either side of the visible spectrum. Although there are as yet few industrial application for such techniques.

2.2.1 Why Has Machine Vision Emerged Now?

The foundations for machine vision were laid in the; 1960's, in two main areas. Firstly, the development of very high altitude aerial photography and the availability of satellite images led to considerable research into the basic techniques of image processing - primarily enhancement for human interpretation. Secondly, simultaneously but quite separately, techniques of analyzing images to extract numerical data from television-type images of microscopic sections were developed for scientific applications. In this period, image processing was performed using large and expensive computers, image analysis did not use recognizable computers; instead dedicated circuitry was used to extract very limited information at TV rates without storing the image digitally.

The recent rapid development of practical industrial applications for image analysis, in the form of machine vision, can be almost wholly attributed to the rapid reductions in the cost of computing power. This in turn resulted from the emergence of microprocessors and more particularly from the considerable reduction in the cost of a unit of memory, as more and more memory became available on a single ‘chip.’ Images contain enormous amounts of information, so the amounts of memory required to hold and

manipulate an image could not be used economically in industrial situations until the beginning of the 1980's. [4]

2.2.2 Why does industry need machine vision?

In manufacturing industry, human operators have in the past always carried out tasks requiring the use of eyesight; this has restricted the spread of automation mainly to repetitive mechanical tasks, or to those in which variations are closely controlled and the necessary feedback of information from the process is fairly simple.

The development of machine vision to such a level that it is now a robust and competitively priced sensing technique has opened up a new world of automation possibilities. This is happening at a time when manufactures are looking for higher levels of automation as well as greater flexibility and when product quality can make all the difference between the success and failure of a company.

Machine vision is in some respects better and in other respects worse than human eyesight. In certain circumstances it can be faster. It will continue working round the clock and will not tire, lose concentration or have 'Monday morning' feelings. It can work in conditions, which would be very unpleasant or impossible for a human operator. It can take dimensional measurements more accurately than a person can estimate by eye, and it can give an objective measure of other variables, such as colour, which an inspector could only assess subjectively.

On the other hand, machine vision is not backed by the power of the human brain. It cannot use all the subtle inferences, which the human eye-brain combination can bring into play in order to analyze a complex scene. Early vision systems, and many of today's simpler systems, are fussy about lighting and may have difficulty in coping with reflections and with (to the human eye~ insignificant random variations in the objects they are looking at. Some of these~ limitations are being overcome as more powerful new computers become available and as artificial intelligence moves from the software laboratory into industry.

There are many different ways in which it is, or will be, possible to use vision systems in industry. The difficulty for the industrial manager and production engineer lies in assessing where it would be worthwhile to introduce vision now, where it is worth

planning for the introduction of vision in projects over the next two or three years, and where the use of vision would be quite inappropriate in the foreseeable future.

2.2.3 Application Areas

There are three broad areas of application for vision systems in industry, in terms of the kind of information they are asked to provide. These are:

Inspection - This has been, and is likely to remain, the most important area. There is a great diversity of applications in this area, including not only visual examination for defects, such as is normally done by eye, but also measurement of dimensions or other characteristics, counting, checking of orientation, and so on.

Recognition - Recognition of an object from its shape, or from 'normal' printed characters rather than from a deliberately encoded machine-readable marking such as a barcode, can simplify certain manufacturing operations. Recognition, however, is probably the least frequently encountered of the three application areas considered.

Guidance - Guidance is the most rapidly growing application area for vision, according to some reports. Applications include control of robots, vehicles, orientation devices, and so on, in parts handling, sorting and transport. The requirements for robot vision are different in many respects from the needs of inspection, and quite different types of system have been developed to deal with it.

Whatever the application machine vision always includes the following three elements:

Acquiring a picture image of the object or scene under examination

Carrying out various processes on the image in order to identify and make a quantitative assessment of the important features.

Making a decision or reporting on the basis of the information. [4]

2.3 Why Use Vision?

Some idea of the present opportunities for using machine vision can be obtained from the types of industries and applications where it is already in operation. Because it is still a very new technology and users are still learning how to get the best out of it, it is not

surprising that most applications so far have been in situations where there is no reasonable alternative way of reaching the desired goal. [4]

2.3.1 Safety and Reliability

Safety and reliability requirements are probably the most important pressures on many manufacturers driving them to find better methods of inspecting their products. In pharmaceuticals, automobile engineering, aerospace, defense and some other industries where human lives can be at stake there are very stringent requirements for the reliability of products, and the trend is always towards higher and more difficult standards.

In such circumstances one might think that nothing less than 100% inspection by skilled personnel could be tolerated. Experience has shown, however, that even repeated inspection, three or four times, on large batches of work can fail to remove every defective product. In fact the better the general level of quality, the more difficult it is to spot the occasional defect.

In the pharmaceuticals industry vision sensing is playing a very important part in checking the correctness of labeling, and particularly the legibility and correctness of date stamps on drugs, which deteriorate rapidly over time.

In the engineering industries the inspection of safety critical components and assemblies is increasingly calling vision sensing in aid. Crack detection has already been mentioned. Critical assemblies like brakes are being inspected by machine vision to ensure that no parts are missing, that the correct parts have been used and that they have been put together correctly.

Razor blade manufacture combines the need for complete assurance of product safety with the impossibility of visual inspection by a human operator because of very high production rates and the use of automatic assembly machines. A very high level of quality, can be maintained by a combination of good product design and the extensive use of optical and other sensing devices in the assembly machine; but in pursuit of its zero defects policy Wilkinson Sword has turned to machine vision for an inspection operation on a double blade assembly which would be very difficult by any other method.

Glass bottle manufacture (another high-volume operation) can produce a variety of defects, which result in almost-invisible pieces of glass being left inside the bottle.

Vision sensing has been invoked successfully to identify these potentially dangerous hazards [4]

2.3.2 Product Quality

Below the absolute demands of safety there is the large and increasingly important area of product quality. In a highly competitive world, quality is often as important a consideration as price, if not more so. Quality standards are rising all the time, and if one's competitor can offer products with consistently better quality he is likely to get the business.

This pressure has led to some surprising applications and potential applications for machine vision. For instance in consumer toiletries and other goods sold in the supermarket, a bottle with a missing label, or even with a label which is slightly defective - badly printed or not centered correctly on the bottle is a discouragement to the prospective buyer. At best, that bottle will be pushed to one side and another taken, but it could be left in place, effectively sealing off all the bottles behind it on the shelf. Instead of going to another bottle of the same type the customer might take a similar product of a rival manufacturer from an adjacent shelf, and at worst might thereafter, get into the habit of buying 'Product B'.

Labeling and the appearance of packaging generally are a matter of serious concern to consumer goods manufacturers, and machine vision can add automatic inspection to what is often a large-volume automated bottling or packaging process. Blemishes on the fresh foods intended for canning or freezing are not safety-critical but could well deter a customer from buying the same product again. There are already some machine vision applications of this type and others are projected.

Continuous inspection of large areas of sheet metal, photographic film and other materials immediately after production to identify surface defects, which must be cut out of the finished product, lends itself to vision scanning techniques. Particularly in the case of photographic emulsion the operation could not be done by human eye, and in all such applications the speed of production, the infrequency of defects and the need for satisfactory marking make direct inspection virtually impossible.

Physical measurement in high-speed repetitive engineering operations can usually be carried out satisfactorily by dedicated gauging equipment, but where the inspection is

rather to check that all the items are present and in their correct positions in a package or an assembly, or to look for damaged or marked items, machine vision will probably be the only alternative to visual inspection and may well be more reliable. [4]

2.3.3 Flexible Automation

Flexible automation is an important and growing area where machine vision will be an essential element. Flexible automation in engineering is the industry's response to market pressures for greater variety in the product range, more prompt deliveries, and faster and more frequent introduction of updated products. Since the industry is unable to meet the challenge by frequent changes in employment and is under pressure to minimize the working capital tied up in stocks and work-in-progress, methods of automated manufacture are being developed which allow different products to be made in random sequence according to the order book, or which allow a quick and inexpensive change from one product line to another.

Wherever a process or an operation has the possibility of drifting out of adjustment or producing random errors, some form of inspection and corrective system is essential, and in an automated process that inspection, and preferably the feedback, need also to be automatic.

This is the area where inspection methods, which would be suitable for large-volume repetitive work, may no longer be acceptable because they usually involve expensive fixturing and location of gauges and other devices in fixed positions for repetitive measurement. While a vision system will run faster if parts are always presented in exactly the same position it does not require the precision of mechanical fixturing. It can also recognize and inspect a number of different work pieces in any order, and is quickly re-programmed for different work.

It is easy to overlook the fact that in manual, operations a 'free' inspection is included as a matter of course. If an operator sees that a part is defective in some way - quite apart from any formal measurements he may take - he will throw it out. An automated system will not detect even the most obvious faults unless they are specifically defined and looked for. A machine vision station can often take over these lost informal inspection tasks, provided they have been identified and taught to it.

Flexible automation is also an area where machine vision shows great potential for robot guidance in handling and assembly operations. A vision system monitoring a conveyor can identify work pieces as they arrive in its field of view, establish their orientation and pass instructions to an industrial robot enabling its gripper to pick up each part correctly. There are also applications in such tasks as robot loading to numerically controlled machine tools of a variety of similar parts, which must be distinguished from each other.

A more difficult challenge is guiding a robot gripper to pick up work pieces, which have been thrown at random into a bin. Every engineer knows that it is bad practice to lose orientation of work-in-progress, but there are situations where it cannot be avoided except at prohibitive cost. The bin-picking robot, which could take parts at random from a box or stillage, would be a valuable automation tool. At present a general-purpose bin-picking robot is not feasible at a commercially worthwhile speed and cost, but it is possible to go part-way towards it. Bin-picking has been demonstrated where there is some restriction on the randomness with which parts are stored and where the parts all belong to a restricted group. This is perfectly practical in most instances. [4]

2.4 Advantages in Using Vision

They can be summarized as follows:

No alternative - There are some situations where vision has provided a solution to a problem where no alternative has been found feasible, so that it has been the means of making a new production process possible. In such a case the cost of the vision system is not the main consideration.

Safety - Where human life is at stake, whether it is in handling dangerous drugs or potentially lethal vehicles, every item must be inspected at least once, sometimes more often, and even then there may be sample testing of finished products. The cost is high, and the penalties for failure may be very high indeed, so here again the expense of machine vision is less important than its ability to take on, reliably, all or part of the task of visual inspection.

Quality - Below the level of safety criticality, the cost of maintaining quality is less easily balanced against the cost of failures in quality. One of the things that may be possible with machine vision is to change from sample inspection to 100% inspection.

When samples of outgoing goods are inspected and the customer uses sample inspection of goods inward, a single defect found by the customer may lead to the return of an entire batch, which must then be sorted completely before it is returned. The level of inspection of finished products is generally chosen to take account of this possibility. Machine vision holds out the possibility of fast enough operation to allow complete inspection of every product, reducing the cost of sample inspection for both vendor and customer and making the vendor a more attractive supplier.

Companies that sell products with a warranty work out a similar equation. They are able over a period of time to calculate a relationship between the amounts they spend on inspection and test of their products and the amount they have to spend in rectifying or replacing products returned under warranty. Machine vision again can alter the terms of the relationship, making it possible to offer more attractive warranty terms or take some other step to improve one's place in the market.

Automation aid - As we have already seen, if machine vision is instrumental in the introduction of flexible automation, then it can claim an important share in many tangible and intangible benefits, like reduced work in progress, shorter door-to-door time, possibly a change from manufacturing for stock to manufacturing against orders, shorter lead time from design to production, and capital savings through the purchase of adaptable plant and tooling with a longer life than dedicated equipment. [4]

Costs-The total cost of installing a machine vision system can vary very widely indeed, depending on the amount of customization and special development work involved. It must also be remembered that costs are constantly falling, and at the same time capabilities are increasing. In all but the most straightforward applications the cost of hardware is a relatively small part of the total installation, but within a few years, as software is streamlined possibly through the use of Expert systems for industrial use, even customization costs may be expected to fall.

For costing purposes, applications can be divided into three broad groups, as detailed below. The figures given are very approximate 1984 costs for complete installations, delivered by a contractor, commissioned and running, and including the training of user's personnel.

Reasonably straightforward applications, of a type already installed elsewhere and requiring minimal customization. Here the cost of hardware could account for about half

the total, and standard software for the greater part of the remainder. Total cost could be in the region of £20,000-30,000.

One-off applications involving some special studies, special software and a fair degree of customization, but within an essentially self-contained system. The project cost here might range from £30,000 to £60,000, but a repeat order for similar equipment could be fulfilled for about one third less.

Vision cells integrated with mechanical and electronic equipment in a complex system, which may include industrial robots. In these last two types of applications the hardware cost of the vision system is much smaller in relation to the other costs maybe 25% or less - and the cost of a vision cell can vary very widely, depending on its complexity and the amount of innovation involved, from £45,000 upwards. Some systems have cost in the region of £250,000, but these may include the cost of other equipment such as robots. [4]



3 IMAGE PROCESSING

We have seen that the human visual system receives an input image as a collection of spatially distributed light energy; this form is called an optical image. Optical images are the types we deal with everyday cameras capture them, monitors display them, and we see them. We know that these optical images are represented as video information in this form of analog electrical signals and have seen how these are sampled to generate the digital image $I(r, c)$.

The digital image $I(r, c)$ is represented as a two-dimensional array of data, where each pixel value corresponds to the brightness of the image at the point (r, c) . In linear algebra terms, a two-dimensional array like our image model $I(r, c)$ is referred to as a matrix, and one row (or column) is called a vector. [5..9] This image model is for monochrome (one-color, this is what we normally refer to as black and white) image data, but other types of image data that require extensions or modifications to this model, Typically, these are multiband images (color, multispectral), and they can be modeled by a different $I(r, c)$ function corresponding to each separate band of brightness information. The image types we will consider are:

- Binary
- Gray-scale
- Multispectral

3.1 Binary Images

Binary images are the simplest type of images and can take on two values, typically black and white, or '0' and '1' A binary image is referred to as a 1 bit/pixel image because it takes only 1 binary digit to represent each pixel. These types of images are most frequently used in computer vision applications where the only information required for the task is general shape, or outline, information. For example, to position a robotic gripper to grasp an object, to check a manufactured objects for deformations, for facsimile (FAX) images, or in optical character recognition (OCR).

Binary images are often created from gray-scale images via a threshold operation, where every pixel above the threshold value is turned white ('1'), and those below it are turned black ('0'). [10]

3.2 Gray-Scale Images

Gray-scale images are referred to as monochrome, or one-color images. They contain brightness information only, no color information. The number of bits used for each pixel determines the number of different brightness levels available. The typical image contains 8 bits/pixel data, which allows us to have 256 (0-255) different brightness (gray) levels. This representation provides more than adequate brightness resolution, in terms of the human visual system's requirements, and provides a "noise margin" by allowing for approximately twice as many gray levels as required. This noise margin is useful in real-world applications because of the many different types of noise (false information in the signal) inherent in real systems. Additionally, the 8-bit representation is typical due to the fact that the byte, which corresponds to 8-bits of data, is the standard small unit in the world of digital computers. In certain applications, such as medical imaging or astronomy, 12 or 16 bits/pixel representations are used. These extra brightness levels become useful only when the image is "blown-up," that is, when a small section of the image is made much larger. In this case we may be able to discern details that would be missing without this additional brightness resolution. Of course, to be useful, this also requires a higher level of spatial resolution (number of pixels). If we go beyond these levels of brightness resolution, we typically divide the light energy into different bands, where each band refers to a specific subsection of the visible image spectrum. [10]

3.3 Color Image

Color images can be modeled as three-band monochrome image data, where each band of data corresponds to a different color. The actual information stored in the digital image data is the brightness information in each spectral band. When the image is displayed, the corresponding brightness information is displayed on the screen by picture elements that emit light energy corresponding to that particular color. Typical color images are represented as red, green, and blue, or RGB images. Using the 8-bit monochrome standard as a model, the corresponding color image would have 24 bits/pixel—8-bits for each of the three-color bands (red, green, and blue).

For many applications, RGB color information is transformed into a mathematical space that decouples the brightness information from the color information. After this is done, the image information consists of a one-dimensional brightness, or luminance, space and a two-dimensional color space. Now the two-dimensional color space does not contain any brightness information, but it typically contains information regarding the relative amounts of the different colors. An additional benefit of modeling the color information in this manner is that it creates a more people-oriented way of describing the colors.

3.4 Correcting Imaging Defects

The first class of image processing operations considered here are those procedures applied to correct some of the defects in as-acquired images that may be present due to imperfect detectors, inadequate or nonuniform illumination or an undesirable viewpoint. It is important to emphasize that these are corrections that are applied after the image has been digitized and stored and therefore will be unable to deliver the highest quality result that could have been achieved by optimizing the acquisition process in the first place.

Of course, storing an optimum-quality image is sometimes impractical. If the camera can collect only a small number of photons in a practical time or before the scene changes, then the noise present in the image cannot be averaged out by acquiring and adding more photons or video frames, and other noise reduction means are needed. If the source of illumination cannot be controlled to be perfectly centered and normal to the viewed surface (for instance the sun), or if the surface is curved instead of planar, then the image will have nonuniform illumination that must be corrected afterwards. If the viewpoint cannot realistically be adjusted for instance the path of a space probe or satellite), or if the surface is irregular (as in the case of a metal fracture), then some parts of the scene will be foreshortened this must be taken into account in comparing sizes or measuring distances.

Even in typical laboratory setups such as the light microscope, keeping the instrument in ideal alignment may be very time-consuming and achieving adequate stability to collect dim fluorescence images for a long time very difficult, so that it becomes more practical to trade off some of the ultimately achievable image quality for convenience and speed, and to utilize processing methods to perform these corrections.

When the first space probe pictures were collected and the need for this type of correction was first appreciated, it required lengthy computations on moderate sized computers to apply them. It is now possible to implement such corrections on desktop machines, in measured in seconds or minutes, so that they can be practically applied to routine imaging needs. [11]

3.5 Noisy images

Noisy images may also occur due to instability in the source or detector during the time required to scan or size an image. The pattern of this noise may be quite different from the essentially Gaussian noise due to counting statistics but it still shows up as a variation in brightness in uniform regions of the scene. One common example is the noise field emission SEM images due to the variation in field emission current at the tip. With a typical time constant of second the electron emission may shift from one atom to another producing a change of several percent in the beam cur. The usual approach to minimize the effects of this fluctuation in the viewed image is to use scan times that are either shorter or longer than the fluctuation time.

Assuming that an image represents the best quality that can practically be obtained, this section will deal with-ways to suppress noise to improve the ability to visualize and demarcate for measurement features, which are present. The underlying assumptions in all of these methods are that the pixels in the image are much smaller than any of the important details, and that for most of the pixels present, most of their neighbors represent the same structure. Various averaging and comparison methods can be applied liased on these assumptions.

These are very much the same assumptions as that inherent in classical image averaging, in which the assumption is that pixel readings at different times represent the same structure in the viewed scene. This directly justifies averaging (or integrating) pixel readings over time to reduce random noise. When the signal varies in other ways described above, other methods such as median filtering can be used.

All cameras have some readout noise. In the typical CCD cam the electrons from each transistor must be transferred across each line in order to be read out as a voltage to the computer. In interline transfer camera, a separate row of transistors adjacent each detector is used instead, somewhat reducing the read noise. Of course, additional sources

of noise from the other associated electronics-clock signals, wiring from the camera to digitizer, pickup of electrical signals from the computer itself, and so forth-may degrade the signal even more. But even if careful design and proper wiring minimize those other sources of noise, there is an irreducible amount of noise superimposed on the signal each time the camera image is read out and digitized. [11]

3.6 Spatial filters

Spatial filtering is typically done for noise removal or to perform some type of image enhancement. These operators are called spatial filters to distinguish them from frequency domain filters. Three types of filters discussed here include

- Mean filters
- Median filters
- Enhancement filters

The first two are used primarily to conceal or remove noise; although they may also be used for special applications. For instance, a mean filter adds a softer. The enhancement filters highlight edges and details within the image.

Many spatial filters are implemented with convolution masks. Because a convolution mask operation provides a result that is a weighted sum of the values of a pixel neighbors, it is called a linear filter. One interesting aspect of convolution mask is that the overall effect can be predicted based on their general pattern. For example if the coefficients of the mask sum to one, the average brightness of the image will be retained. If the coefficients sum to zero, the average brightness will be lost and will return a dark image. Furthermore, if the coefficients are alternating positive and negative, the mask is a filter that returns edge information only; if the coefficients are all positive, it is a filter that will blur the image. The mean filters are essentially averaging filters. They operate on local groups of pixels called neighborhoods and replace the center pixel with an average of the pixels in this neighborhood. This replacement is done with a convolution mask such as the following 3 x 3 mask:

$$\begin{vmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{vmatrix}$$

Note that the coefficients of this mask sum to one, so the image brightness will be retained, and the coefficients are all positive, so it will tend to blur the image. There are other more complex mean filters that are designed to deal with specific types of noise.

The median filter is a nonlinear filter. A nonlinear filter has a result that cannot be found by a weighted sum of the neighborhood pixels, such as is done with a convolution mask. However, the median filter does operate on a local neighborhood. After the size of the local neighborhood is defined, the center pixel is replaced with the median, or center, value present among its neighbors, rather than by their average.

A median filter can use a neighborhood of any size, but 3×3 , 5×5 , and 7×7 are typical. Note that the output image must be written to a separate image (a buffer), so that the results are not corrupted as this process is performed.

The enhancement filters considered here include laplacian-type and difference filters. These types of filters will tend to bring out, or enhance, details in the image.

Two 3×3 convolution masks for the laplacian-type filters are

$$\begin{vmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{vmatrix} \quad \begin{vmatrix} 0 & -2 & 0 \\ -2 & 5 & -2 \\ 0 & -2 & 0 \end{vmatrix}$$

The laplacian-type filter will enhance details in all directions equally. The difference filters will enhance details in the direction specific to the mask selected. There are four difference filter convolution masks, corresponding to lines in the vertical, horizontal, and two diagonal directions:

$$\begin{vmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{vmatrix} \quad \begin{vmatrix} 0 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 0 \end{vmatrix} \quad \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{vmatrix} \quad \begin{vmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{vmatrix}$$

3.7 Image Quantisation

Image Quantisation is the process of reducing the image data by removing some of the detail information by mapping groups of data points to a single point. This can be

done either to the pixel values themselves $I(r, c)$ or to the spatial coordinates (r, c) . Operation on the pixel values is referred to as gray-level reduction, while operating on the spatial coordinates is called spatial reduction.

The simplest method of gray-level reduction is thresholding. We select a threshold gray level and set everything above that value equal to '1' (255 for 8-bit data) and everything at or below the threshold equal to '0'. This effectively turns a gray-level image into a binary (two-level) image and is often used as a preprocessing step in the extraction of object features such as shape, area, or perimeter.

A more versatile method of gray-level reduction is the process of taking the data and reducing the number of bits per pixel. This can be done very efficiently by masking the lower bits via an AND operation. With this method, the number of bits that are masked determines the number of gray levels available.

For example, to reduce the image by a factor of two, we simply take every other row and column and delete them. To improve the image quality when applying the decimation technique, we may want to preprocess the image with an averaging, or mean, spatial filter this type of filtering is called anti-aliasing filtering.

To perform spatial quantisation we specify the desired size, in pixels, of the resulting image. For example, to reduce a 512 x 512 image to one quarter its size, we specify that we want the output image to be 256 x 256 pixels. We now take every 2 x 2 pixel block in the original image and apply one of the three methods listed previously

In order to create a reduced image. It should be noted that this method of spatial reduction allows for simple forms of geometric distortion, specifically, stretching or shrinking along the horizontal or vertical axis.

3.8 Segmentation

In the analysis of the objects in images it is essential that we can distinguish between the objects of interest and "the rest." This latter group is also referred to as the background. The techniques that are used to find the objects of interest are usually referred to as *segmentation techniques* - segmenting the foreground from background. In this section the most common techniques *thresholding* and *edge finding* and techniques for

improving the quality of the segmentation result will be presented. It is important to understand that:

There is no universally applicable segmentation technique that will work for all images, and,

No segmentation technique is perfect. [12]

3.9 Thresholding

This technique is based upon a simple concept. A parameter θ called the brightness threshold is chosen and applied to the image $a[m,n]$ as follows:

$$\begin{array}{ll} \text{If } a[m,n] \geq \theta & a[m,n] = \text{object} = 1 \\ \text{Else} & a[m,n] = \text{background} = 0 \end{array} \quad [13]$$

This version of the algorithm assumes that we are interested in light objects on a dark background. For dark objects on a light background we would use:

$$\begin{array}{ll} \text{If } a[m,n] < \theta & a[m,n] = \text{object} = 1 \\ \text{Else} & a[m,n] = \text{background} = 0 \end{array} \quad [13]$$

The output is the label "object" or "background" which, due to its dichotomous nature, can be represented as a Boolean variable "1" or "0". In principle, the test condition could be based upon some other property than simple brightness (for example, *If (Redness {a[m,n]} >= θ_{red})*, but the concept is clear.

The central question in thresholding then becomes: how do we choose the threshold θ . While there is no universal procedure for threshold selection that is guaranteed to work on all images, there are a variety of alternatives.

* *Fixed threshold* - One alternative is to use a threshold that is chosen independently of the image data. If it is known that one is dealing with very high-contrast images where the objects are very dark and the background is homogeneous and very light, then a constant threshold of 128 on a scale of 0 to 255 might be sufficiently accurate. By accuracy we mean that the number of falsely-classified pixels should be kept to a minimum.

* *istogram-derived thresholds* - In most cases the threshold is chosen from the brightness histogram of the region or image that we wish to segment. An image and its associated brightness histogram are shown in Figure 3.2.

A variety of techniques have been devised to automatically choose a threshold starting from the gray-value histogram, $\{h[b] \mid b = 0, 1 \dots 2^B - 1\}$. Some of the most common ones are presented below. Many of these algorithms can benefit from a smoothing of the raw histogram data to remove small fluctuations but the smoothing algorithm must not shift the peak positions. This translates into a zero-phase smoothing algorithm given below where typical values for W are 3 or 5:

$$h_{\text{smooth}}[b] = \frac{1}{W} \sum_{w=-(W-1)/2}^{(W-1)/2} h_{\text{raw}}[b-w] \quad W \text{ odd} \quad [13]$$

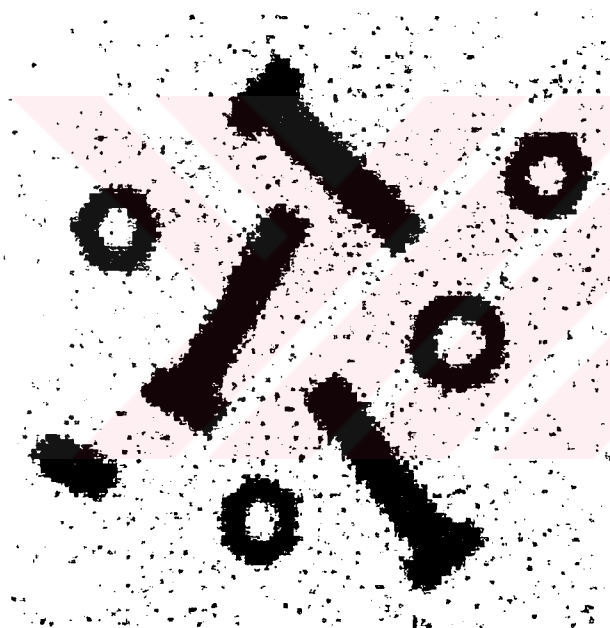


Figure 3-1 Image to be Thresholded

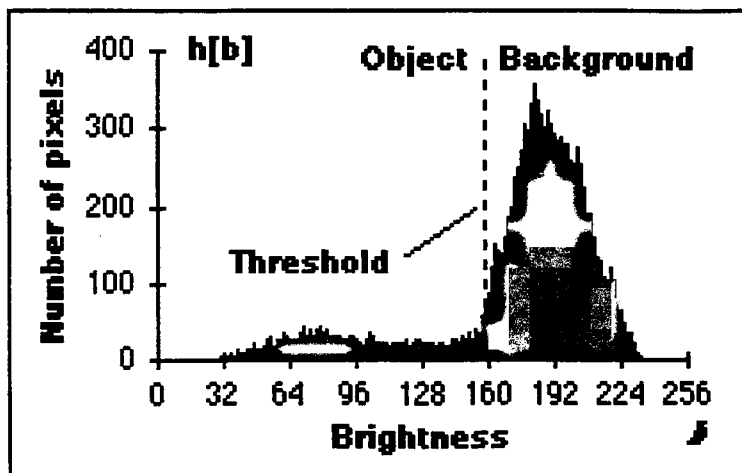


Figure 3-2 Brightness Histogram of the Image

Pixels below the threshold ($a[m,n] < \theta$) will be labeled as object pixels; those above the threshold will be labeled as background pixels.

* *Isodata algorithm* - This iterative technique for choosing a threshold was developed by Ridler and Calvard. The histogram is initially segmented into two parts using a starting threshold value such as $\theta_0 = 2^{B-1}$, half the maximum dynamic range. The sample mean ($m_{f,0}$) of the gray values associated with the foreground pixels and the sample mean ($m_{b,0}$) of the gray values associated with the background pixels are computed. A new threshold value θ_1 is now computed as the average of these two sample means. The process is repeated, based upon the new threshold, until the threshold value does not change any more. In formula:

$$\theta_k = (m_{f,k-1} + m_{b,k-1}) / 2 \text{ until } \theta_k = \theta_{k-1} \quad [14]$$

* *Background-symmetry algorithm* - This technique assumes a distinct and dominant peak for the background that is symmetric about its maximum. The technique can benefit from smoothing as described in equation. The maximum peak ($maxp$) is found by searching for the maximum value in the histogram. The algorithm then searches on the *non-object pixel side* of that maximum to find a $p\%$ point as in equation.

In Figure 3.2, where the object pixels are located to the *left* of the background peak at brightness 183, this means searching to the right of that peak to locate, as an example, the 95% value. At this brightness value, 5% of the pixels lie to the *right* (are above) that value. This occurs at brightness 216 in Figure 3.2. Because of the assumed

symmetry, we use as a threshold a displacement to the *left* of the maximum that is equal to the displacement to the right where the $p\%$ is found. For Figure 3.2 this means a threshold value given by $183 - (216 - 183) = 150$. In formula:

$$\theta = maxp - (p\%o - maxp) \quad [13]$$

This technique can be adapted easily to the case where we have light objects on a dark, dominant background. Further, it can be used if the object peak dominates and we have reason to assume that the brightness distribution around the object peak is symmetric. An additional variation on this symmetry theme is to use an estimate of the sample standard deviation based on one side of the dominant peak and then use a threshold based on $\theta = maxp \pm 1.96s$ (at the 5% level) or $\theta = maxp \pm 2.57s$ (at the 1% level). The choice of "+" or "-" depends on which direction from $maxp$ is being defined as the object/background threshold. Should the distributions be approximately gaussian around $maxp$, and then the values 1.96 and 2.57 will, in fact, correspond to the 5% and 1 % level.

* *Triangle algorithm* - This technique is illustrated in. A line is constructed between the maximum of the histogram at brightness b_{max} and the lowest value $b_{min} = (p=0)\%$ in the image. The distance d between the line and the histogram $h[b]$ is computed for all values of b from $b = b_{min}$ to $b = b_{max}$. The brightness value b_o where the distance between $h[b_o]$ and the line is maximal is the threshold value, that is, $\theta = b_o$. This technique is particularly effective when the object pixels produce a weak peak in the histogram.

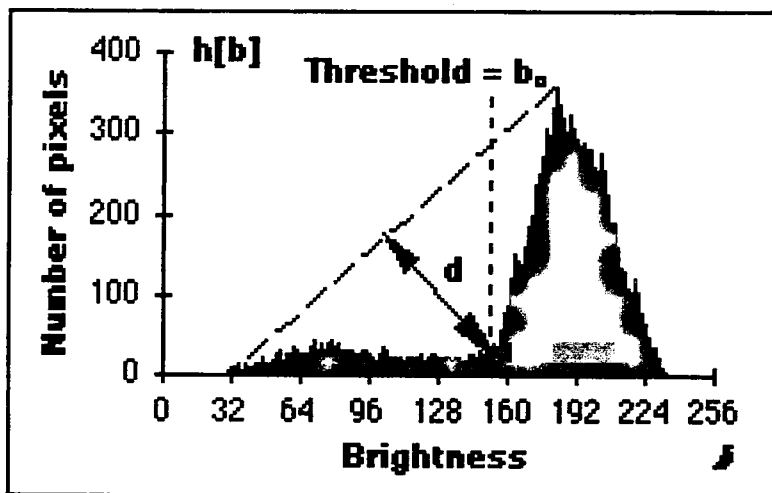


Figure 3-3 The triangle algorithm is based on finding the value of b that gives the maximum distance d .

The three procedures described above give the values $\theta = 139$ for the Isodata algorithm, $\theta = 150$ for the background symmetry algorithm at the 5% level, and $\theta = 152$ for the triangle algorithm for the image in Figure 3.1.

Thresholding does not have to be applied to entire images but can be used on a region-by-region basis. Chow and Kaneko developed a variation in which the $M \times N$ image is divided into non-overlapping regions. In each region a threshold is calculated and the resulting threshold values are put together (interpolated) to form a thresholding surface for the entire image. The regions should be of "reasonable" size so that there are a sufficient number of pixels in each region to make an estimate of the histogram and the threshold. The utility of this procedure--like so many others--depends on the application at hand.

3.10 Edge line detection

The edge and line detection operators presented here represent the various types of operators in use today. Many are implemented with convolution masks, and most are based on discrete approximations to differential operators. Differential operations measure the rate of change in function, in this case, the image brightness function. A large change in image brightness over a short spatial distance indicates the presence of an edge. Some edge detection operators return orientation information (information about the direction of edge) whereas others only return information about the existence of an edge at each point. Also included in this section is a special transform, the Hough Transform, which is specifically defined to find lines. Edge detection methods are used as a first step in the line detection process. Edge detection is also used to find complex object boundaries by marking potential edge points corresponding to places in an image where rapid changes in brightness occur. After these edge points have been marked, they can be merged to form lines and object outlines.

With many of these operators, noise in the image can create problems. That is why it is best to preprocess the image to eliminate, or at least minimize, noise effects. To deal with noise effects, we must make tradeoffs between the sensitivity and the accuracy of an edge detector. For example, if the parameters are set so that the edge detector is very sensitive, it will tend to find many potential edge points that are attributable to noise. If we make it less sensitive, it may miss valid edges. The parameters that we can set include the

size of the edge detection mask and the value of the gray-level threshold. A larger mask is less sensitive to noise; a lower gray-level threshold will tend to reduce noise effects.

Edge detection operators are based on the idea that edge information in an image is found by looking at the relationship a pixel has with its neighbors. If a pixel's gray-level value is similar to those around it, there is probably not an edge at that point. However, if a pixel has neighbors with widely varying gray levels, it may represent an edge point. In other words, an edge is defined by a discontinuity in gray-level values. Ideally, an edge separates two distinct objects. In practice, apparent edges are caused by changes in color or texture or by the specific lighting conditions present during the image acquisition process.

3.10.1 Roberts Operator

The Roberts operator marks edge points only; it does not return any information about the edge orientation. It is the simplest of the edge detection operators and will work best with binary images (gray-level images can be made binary by a threshold operation). There are two forms of the Roberts operator. The first consists of the square root of the sum of the differences of the diagonal neighbors squared, as follows:

$$\sqrt{[I(r,c) - I(r-1,c-1)]^2 + [I(r,c-1) - I(r-1,c)]^2} \quad [10]$$

The second form of the Roberts operator is the sum of the magnitude of the differences of the diagonal neighbors, as follows:

$$[I(r,c) - I(r-1,c-1)] + [I(r,c-1) - I(r-1,c)] \quad [10]$$

The second form of the equation is often used in practice due to its computational efficiency-it is typically faster for a computer to find an absolute value than to find square roots.

3.10.2 Sobel Operator

The Sobel edge detection masks look for edges in both the horizontal and vertical directions; and then combine this information into a single metric. The masks are as follows:

RETIROU
DIAPOGRAMA
CERKEZI

$$\text{Row Mask} \begin{vmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{vmatrix}$$

$$\text{Column Mask} \begin{vmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{vmatrix}$$

Each of these masks is convolved with the image. At each pixel location we now have two numbers: s_1 , corresponding to the result from the row mask, and s_2 , from the column mask. We use these numbers to compute two metrics, the edge magnitude and the edge direction, which are defined as follows:

EDGE MAGNITUDE

$$\sqrt{s_1^2 + s_2^2} \quad [11]$$

EDGE DIRECTION

$$\tan^{-1} \left[\frac{s_1}{s_2} \right]$$

The edge direction is perpendicular to the edge itself because the direction specified is the direction of the gradient, along which the gray levels are changing.

3.10.3 Prewitt Operator

The Prewitt is similar to the Sobel, but with different mask coefficients: The masks are defined as follows:

$$\text{Row Mask} \begin{vmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{vmatrix}$$

$$\text{Column Mask} \begin{vmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{vmatrix}$$

These masks are each convolved with the image. At each pixel location we find two numbers p_1 , corresponding to the result from the row mask, and p_2 , from the column

mask. We use these results to determine two metrics, the edge magnitude and the edge direction, which are defined as follows:

EDGE MAGNITUDE

$$\sqrt{p_1^2 + p_2^2} \quad [12]$$

EDGE DIRECTION

$$\tan^{-1} \left[\frac{p_1}{p_2} \right]$$

As with the Sobel edge detector, the direction lies 90° from the apparent direction of the edge.

3.10.4 Kirsch Compass Masks

The Kirsch edge detection masks are called compass masks because they are defined by taking a single mask and rotating it to the eight major compass orientations; North, Northwest, West, Southwest, South, Southeast, East, and Northeast. The masks are defined as follows:

k_0	K_1	k_2	k_3
$\begin{vmatrix} -3 & -3 & -5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{vmatrix}$	$\begin{vmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{vmatrix}$	$\begin{vmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{vmatrix}$	$\begin{vmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{vmatrix}$
k_4	K_5	k_6	k_7
$\begin{vmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & -3 \\ -3 & 0 & -3 \\ 5 & 5 & 5 \end{vmatrix}$	$\begin{vmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{vmatrix}$

The edge magnitude is defined as the maximum value found by the convolution of each of the masks with the image. The mask that produces the maximum magnitude defines the edge direction; for instance, k_0 corresponds to a vertical edge, whereas k_5

corresponds to a diagonal edge in the Northwest/Southeast direction. We also see that the last four masks are actually the same as the first four, but flipped about a central axis. [15]



4 PATTERN CLASSIFICATION

The aim of image classification is to map feature space to the output class space such that:

Each input vector is mapped to one and only one class. This is to say each sample vector in the input space will be assigned to one class only.

Many vectors from the input space map to one class. The classification algorithm categorizes the input space this way.

The class space is constructed by the information extracted from the training sample set. The main difference between many classification algorithms is the construction of the class space. Each classifier has its own characteristic rules to construct the class space.

All the vectors from the input space, which have not been used in training, are also mapped to one class in the output class space. This is the generalization principle of the classification algorithm. The performances of the classifiers are measured by their correct classification rates of test sample set.

To give an example for the classification: let us have two classes in the feature space. Class 1 is represented by rectangles; class 2 represented by circles. In the figure 4.1 the layout is shown.

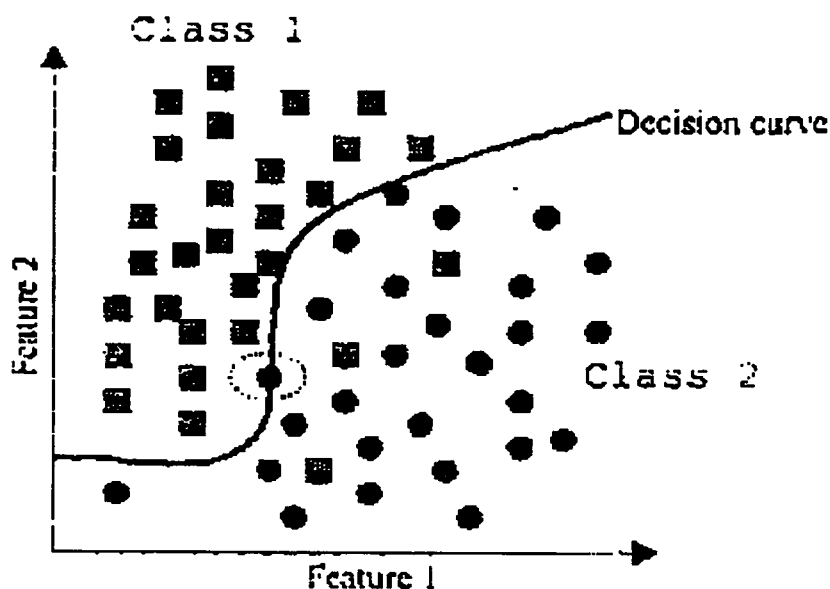


Figure 4-1 Distribution of two class data, and classification example

The classes are represented by the decision curve in the feature space, which is formed by the discriminant of function the classifier. By finding the decision curve, the feature space is divided into two regions, one for Class 1 and another for Class 2. The classifier can classify an unknown test sample this way.

A test sample may also be located on the decision curve. One examples of this situation is shown in the Figure 4.1. The sample is circled with dashed lines. The classifier can not decide on which class the sample belongs and reject the sample if the reject option is not desired, then the decision curve is included into the region of one class.

Classification methods discussed in this chapter can be divided into three main categories: Statistical Image Recognition, Artificial Neural Network and Inductive Learning. [15..19]

4.1 Statistical Image (Pattern) Recognition

Statistical Pattern Recognition has a strong background and has led many applications over the years.[20] In these methods one tries to model the class distributions and find a discriminant function to minimize the classification error. Two approaches for modeling the class distribution are:

Assuming that the class distribution comes from a known family of distributions

Let the data form the class distributions

These two cases are explained in the following sections.

4.1.1 Parametric Methods

In this approach, class distributions or the prior probability distribution function of each class is assumed to belong to a known family of distributions. The parameter of these distributions is found from the training data. Hence the aim is to estimate the parameter of a predefined probability distribution function according to the training data set given.

4.1.1.1 Bayes Decision Theory

Bayes decision Theory is the fundamental statistical approach to the problem of pattern classifications. [20] Given prior probability of each class and assuming a prior class

distribution, Bayes Theory tries to reach the posterior class distribution function. The prior class distribution function is taken as Gaussian probability distribution function (PDF).

The Bayes classifier chooses the class label corresponding to the greatest discriminant value from the given test data.

The discriminant function for Bayes Classifier is:

$$g_i(x) = \frac{1}{2} \log \left| \sum_i \right| - \frac{1}{2} (x - \mu_i)^T \sum_i^{-1} (x - \mu_i) + \log P(w_i) \quad [21]$$

$\sum_i$ is the covariance matrix.

μ_i is the mean vector for class i.

$P(w_i)$ is the prior class probability for class w_i

In practical applications the discriminant function can be simplified as:

Case 1: ($\sum_i = \sigma^2 I$) The covariance matrices of all classes are taken as identity matrices. In this case, the Euclidean distances are calculated between the test data and class center of each class. The class center of a class is the mean vector of its distribution. The closest class center is chosen as the correct class. This classifier is also called as Minimum Distance Classifier.

Case 2: ($\sum_i = \sum$) The covariance matrices of all classes are taken as identical. In this case, the Mahalanobis distance is calculated between each class center and the test data. As the variances are taken into consideration, the distances are normalized. The closest class center is chosen as the correct class.

Case 3: ($\sum_i$ Arbitrary) No simplification is done; the discriminant function is taken as is.

4.1.2 Non Parametric Methods

In constructing the classifier by parametric methods, a distribution for the probability distribution function (PDF) is assumed and the classifier is formed. If the assumed PDF does not model the actual data sufficiently, the classifier will not operate properly. [20] To overcome this difficulty, non-parametric methods are introduced.

In non-parametric methods, the likelihood PDF is directly constructed from the training data given. The initial estimate for the likelihood PDF is the histogram of data, which simply approximates the distribution roughly. The likelihood PDF will be as:

$$f(x) = \frac{\text{number of sample data in the same bin as } x}{nv} \quad [22]$$

v is the volume of the bin

n is the number of samples in the bin

4.1.2.1 K Nearest Neighbor Classifier (KNN)

In Parzen Windows Approach bin size is taken constant, sample size is assumed variable to improve approximation of likelihood PDF. [23] The approximation for the likelihood PDF can be improved by taking constant sample size and variable bin size also. This is called K Nearest Neighbor estimator Variable bin size can be given as $dk(x)$.

The estimated PDF becomes:

$$f(x) = \frac{1}{n * dk(x)} \sum_{i=1}^n K\left(\frac{x - X_i}{dk(x)}\right) \quad [24]$$

The Kernel function is taken as:

$$K(u) = \begin{cases} 1/2 & \text{if } |u| < 1 \\ 0 & \text{otherwise} \end{cases} \quad [25]$$

In KNN classifier, the volume of the bin is taken variable, while the number of samples in the bin is taken as constant. The discriminant function of KNN classifier becomes from the above estimated probability distribution function: [20]

$$g_i = \frac{k_i}{k} \quad [26]$$

where

k is the neighborhood size taken

k_i is the number of samples belonging to 'class i' in the k neighborhood

There is a special case of KNN such that the neighborhood size is taken as one. This classifier is called Nearest Neighbor classifier (NN). It simply assigns the test sample to the class having a training sample closest to it.[27]

4.1.3 Fuzzy KNN Classifier

This classifier is a variation of the actual KNN; the distance measure is replaced by membership function of the fuzzy logic.[28]

The discriminant function becomes:

$$g_i(x) = \frac{\sum_j F(x - X_j) m_j(X_j)}{\sum_j (x - X_j)} \quad [29]$$

where

$$F(u) = \frac{1}{\|u\|^{2/r-1}} \quad [30]$$

and

$$m_j(X_j) = \begin{cases} 1 & X_j \in \text{class } w_i \\ 0 & \text{otherwise} \end{cases} \quad [31]$$

r is the parameter for Fuzzy KNN classifier.

4.2 Artificial Neural Networks

Fascinating power of the brain in signal processing, decision-making, storage of information and controlling body actions attracted many researchers from different disciplines.

In contrast to this fascinating power, very simple cells called neurons form the brain. Neuron is a specialized cell that processes the incoming information and conducts it to the next neuron.

In the fifties, the idea of copying brain functions and neurons arose. The pioneering works of McCulloch [32], Pitts, Hebb [33] and Rosenblatt [34] were the beginning of a new discipline. They imitated biological neurons to form artificial counterparts of them.

From that time on, Artificial Neural Network has been widely used in many areas such as: signal processing, system identification and adaptive control, function approximation, pattern recognition, image recognition. The combination of important papers in ANN can be found in the excellent book of Anderson [35]

As artificial neural networks algorithms, adaline, backpropagation algorithm and LVQ is used.

4.2.1 Adaline Algorithm

Widrow [36] and Hoff suggested Adaline as a supervised neural network algorithm. It is trained by a gradient descent rule, called Least Mean Squares (LMS), which is very similar to the perceptron-learning algorithm. In this thesis adaline will be used as a classifier although perceptron is used generally. The reason is:

To show the main difference between adaline and perceptron algorithms, it is better to explain their weight update rules on the same neuron system given in Figure 4.2.

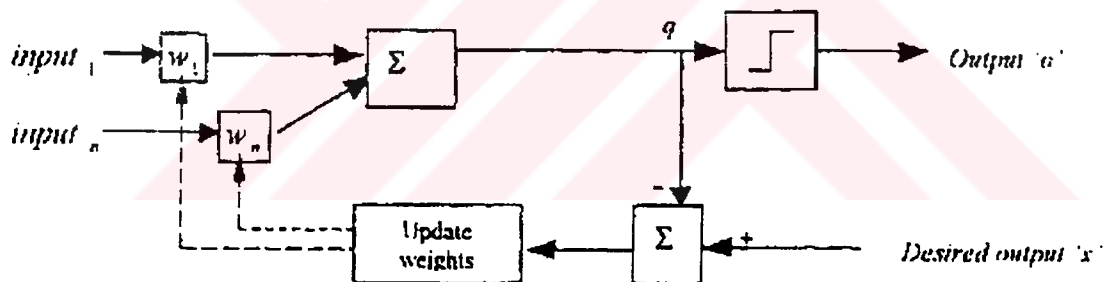


Figure 4-2 Single Neuron System

The weight update rule for adaline is:

$$w_i(t+1) = w_i(t) + \alpha(t)[x(t) - q(t)] \quad [37]$$

The weight rule perceptron is:

$$w_i(t+1) = w_i(t) + \alpha(t)[x(t) - o(t)] \quad [38]$$

$\alpha(t)$ is adaptation gain

w_i is weight of neuron i

The difference between these two algorithms is that, the error fed back takes continuous values in adaline. Since the error fed back takes continuous values, training

continues until asymptotic convergence, whereas in perceptron the error fed back take binary values. The convergence time is improved but convergence to the desired location is not guaranteed.

In many real life applications, classes are not well separated. Applying single perceptron as a classifier gives poor results. The reason is that the error fed back takes binary values. In this thesis adaline algorithm is used for that reason.

4.2.2 Learning Vector Quantization (LVQ)

Kohonen proposed LVQ as a supervised ANN algorithm. The primary goal of LVQ is to represent many points by fewer prototypes in the feature space [39,40]

LVQ is the supervised counterpart of SOM. It is explained in the previous chapter that SOM is a clustering algorithm. In LVQ, the supervisor directs the system for clustering in supervised mode.

The learning rule for LVQ is:

$$w_{ij}(t+1) = \begin{cases} w_{ij}(t) + \alpha(t)[x(t) + w_{ij}(t)] & \text{if } C_s(t) = C_r \\ w_{ij}(t) + \alpha(t)[x(t) - w_{ij}(t)] & \text{if } C_s(t) \neq C_r \\ w_{ij}(t) & \text{otherwise} \end{cases} \quad [41]$$

where

C_r is the correct class of feature r

$C_s(t)$ is the decision made at time t

$\alpha(t)$ is the adaptation gain.

w_{ij} is the weight between neurons i and j .

The weights are adjusted with a learning rule similar to LMS. The difference is that, a penalty rule is also added to the learning phase. If the decision made is correct at that time, the weight of the neuron is increased. But if the decision made is not correct, then the weight of the neuron is decreased. If the weight of the neuron does not affect the decision, it is left unchanged.

4.3 Inductive Learning

In recent years, there has been a growing amount of research on inductive learning. [42] In its broadest sense, induction (or inductive inference) *is a method of moving from the particular to the general - from specific examples to general rules* [43.46] explains inductive learning as

"A process of acquiring knowledge by drawing inductive inferences from teacher or environment - provided facts. Such a process involves operations of generalizing, specializing, transforming, correcting and refining knowledge representations".

Induction can be considered the process of generalizing a procedural description from presented or observed examples [47..49].

The purpose of inductive learning is to perform a synthesis of new knowledge, and this is independent of the form given to the input information [50].

Inductive learning includes learning from examples and learning from observation and discovery [51.52]. In order to form a knowledge base using inductive learning, the first task is to collect a set of representative examples of expert decisions. Each example belongs to a known class (for example, + or -) and is described in terms of a number of attributes, (for example "hair" or "eyes"). These examples may be specified by an expert as a good tutorial set, or may come from some neutral source such as an archive. The induction process will attempt to find a method of classifying an example, again expressed as a function of the attributes, that explains the training examples and that may also be used to classify previously unseen cases [45]. The outcome of an induction algorithm is either a decision tree or a set of rules. Production rules can easily be extracted from decision trees [53]. Each path of a decision tree can be regarded as an IF-THEN production rule. For example, the following production rules can be obtained from the decision tree shown in Figure 4-3

+

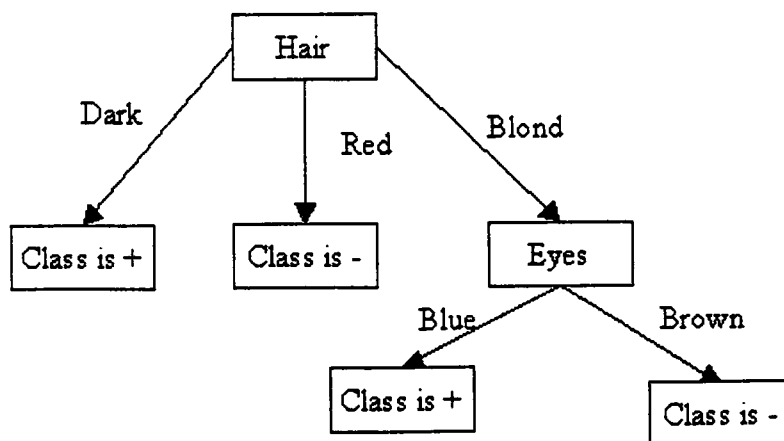


Figure 4-3 An example Decision tree

IF hair IS dark THEN Class IS +

IF hair IS red THEN Class IS -

IF hair IS blond AND eyes ARE blue THEN Class IS +

IF hair IS blond AND eyes ARE brown THEN Class IS -

4.3.1 Categories of Inductive Learning Methods

Quinlan [54] states that there are two categories of induction methods:

Divide-and-conquer methods

Covering methods.

This classification depends on the strategy that the algorithms employ during the search for generalized descriptions. Divide-and-conquer methods have received considerable attention among researchers in the area of applied AI [55]. With a divide-and-conquer method, the outcome is a decision tree. CLS [56] and ID3 [57] are two divide-and-conquer algorithms. In general, a divide-and-conquer algorithm constructs decision trees from a training set of objects according to the following simplified procedure:

If all training objects belong to a single class, the tree is a leaf labeled with that class.

Otherwise,

Apply a test based on one attribute

Divide the training set into subsets, each corresponding to one of the possible (mutually exclusive) outcomes of the test, and

Repeat the procedure for each subset.

The ability of the decision tree to classify previously unseen objects depends on the choice of the test to divide the training set. A good choice should facilitate the development of pure subsets, which do not require further division.

Algorithms of the covering type, in particular the AQ family of algorithms [46], represent classification knowledge as a disjunctive logical expression defining each class. These algorithms search for a set of possible generalizations in an attempt to find "good" hypotheses that satisfy certain requirements. The search proceeds by choosing as the initial working hypotheses some elements from the partially ordered set of all possible descriptions. If the working hypotheses satisfy certain criteria then the search ends. Otherwise, slightly generalizing or specializing them modifies the current hypotheses. These new hypotheses are then checked to see if they satisfy the termination criteria [58]. The process of modifying and checking continues until the assumed criteria are met. The core of the covering method can be summarized as follows [55]:

Find a conjunction of conditions that is satisfied by some objects in the target class, but not by objects from other classes.

Append this conjunction as one disjunction of the logical expression being developed

Remove all objects that satisfy this conjunction and, if there are still some remaining objects of the target class, repeat the procedure.

4.3.1.1 Divide-and-Conquer Induction Algorithms

4.3.1.1.1 CLS: Concept Learning System

One of the pioneers of concept learning using computer programs was Hunt who developed a series of learning algorithms called Concept Learning Systems, CLS-1 to CLS-9 [57]. All CLS versions except CLS-9 can deal only with binary classification problems. These algorithms are heuristic in the sense that they contain "rules of thumb"

which usually produce a concept in the form of a structurally simple tree with a minimum of computation. The CLS-1 algorithm, which is the basis of all later versions, is described in [57] as follows:

A sample of objects to be classified is randomly chosen from the given training set. It is assumed that these objects can belong to one of two classes, the Positive and Negative classes.

A search is made for some characteristics (values of attributes) or a set of characteristics, which appear in objects in the Positive class and not in objects in the Negative class. If such characteristics are found, they are made the test associated with the root of the concept tree and the problem is solved. (The information is stored in matrix form; all that CLS-1 needs to do is compare successive rows representing positive or negative objects, column by column).

If step (2) cannot be carried out, it is reversed. A search is made for a set of characteristics, which appear in objects in the Negative class and not in any object in the Positive class. If such a set can be found, then the problem is solved as before.

If steps (2) and (3) fail, the relative frequencies of characteristics that appear in the objects in the Positive class are counted. The characteristic with the highest frequency of occurrence is chosen as the test at the current node. The training set is split into two subsets, one containing all objects having the characteristic just chosen, and the other containing all objects that do not.

The same procedure is reapplied to the subsets until no further divisions are required (which means all objects in the training set are correctly classified).

This procedure assumes that there are both positive and negative distances in the original set of examples. If, at any point in solving either the original problem or a sub-problem, CLS-1 encounters a sample containing only positive (negative) instances, the current node is immediately made a positive (negative) endpoint and the next sub-problem is attacked. In steps 2 and 3, if more than one characteristic qualify for selection as a test, a random choice is made between them.

To illustrate the operation of CLS-1, consider the example problem for which the training set is given in Table 4.1. The training set is a collection of strings of four letters.

Each string is described in terms of four attributes, each having a consonant as its value. There are two classes in the problem, the Positive class and the Negative class.

Positive Class	Negative Class
JRTK	JVBK
QPQZ	QPWY
BRYL	NTYD
XVQM	MRQM
PTQW	JCTK
FRKV	BRQV
NSQK	LRQW

Table 4-1 Training set for the CLS-1 example problem

In step 1, the first object, JRTK, in the Positive class, is compared with the second object, QPQZ, in the same class. Since they have no common letters (values) in corresponding positions, step 1 fails.

In step 2, the first object, JVBK, in the Negative class and the second object, QPWY, in the same class again have no common letters, thus step 2 also fails.

In step 3, no letter occurs more than once in the first position in any object in the set of objects in the Positive class. However, K occurs twice in the fourth position, R occurs three times in the second position and Q occurs four times in the third position. Therefore Q is selected as the best characteristic for the first node.

In step 4, the set of examples can be split into two subsets

In step 5, the entire procedure is reapplied to the first and the second subsets. For the first subset, QP-Z and XV-M, wht.ch are the first and the second objects in the Positive class, contain no common letters, so step 1 fails again. In step 2, objects MR-M, BR-V and LR-Q in the Negative class contain R in the second position and no object in the Positive class contains R in the same position. Therefore, the first subset is classified. The same

procedure is repeated for the second subset. Since all objects in the Positive class in the second subset contain R in the second position but objects in the Negative class do not, no more subsets remain, and the problem is solved.

As mentioned above, Hunt et al. have developed nine versions of CLS. CLS-2 and CLS-3 are similar to CLS-1 except that an additional control parameter is inserted to indicate the maximum memory size of the computer (CLS-2 CLS-3 differ from each other in the way that the set of examples is stored). CLS-4 and CLS-5 differ from CLS-1 only in the manner of choosing a sample. They were developed to improve the performance of CLS-1 by introducing some control over the order in which objects are added to the working set, following the occurrence of classification errors. CLS-6 is akin to CLS-1 except that sub samples are selected using relative frequency. CLS-7 differs from CLS-1 only by redefining the Positive set. In this algorithm, the set of positive instances is defined at each node as the larger of the two sets. CLS-8 is similar to CLS-7 except that the smaller, rather than the larger set, is considered as the Positive set. CLS-9 has the advantages of many of the earlier algorithms and allows multi-branch trees to be generated. More details of these algorithms can be found in [56].

4.3.1.1.2 ID3 Inductive Learning Algorithm

ID3, a descendant of Hunt's CLS, is a simple algorithm for discovering a set of classification rules from a collection of objects belonging to different classes [45]. ID3 uses a multi-branch rather than a binary decision tree. It differs from CLS-9 in the way it chooses the test node (attribute) to divide the original set of examples or subsets. The attribute selection part of ID3 is based on the assumption that the complexity of the decision tree is strongly related to the amount of information contained in the tree. ID3 uses an information-theoretic approach aimed at minimizing the expected number of tests to classify an object. The method can be summarized as follows:

Select at random a subset (or window) of the given instances.

Form a rule to explain the current window.

Find the exceptions to this rule in the remaining instances.

Form a new window from the current window and the exceptions to the rule generated from it.

If there are no exceptions to the rule then stop. Otherwise, repeat steps 2 to 5 until no exceptions remain.

The core procedure of the algorithm is the recursive process to form a decision tree from the current window. Let S be a set of objects. If all objects in S belong to the same class, then the decision tree is a leaf bearing that class name. Otherwise, S contains representatives of more than one class. An attribute is selected to partition S into subsets $S_1, S_2, \dots, S_n$ where S_i contains those members of S that have the i^{th} value of the selected attribute. An object is classified by starting at the root of the decision tree. To find the root of the tree, the information gain of each attribute is computed. Let A be a set of m attributes $(A_1, A_2, \dots, A_m)$ and C a set of p classes $(C_1, C_2, \dots, C_p)$. The set of possible values for an attribute A_j is referred to as $\text{Range}(A_j)$. Each example in S is an $m+1$ tuple of the form: $(V_1, V_2, \dots, V_m, C_k)$ where $V_j \in \text{Range}(A_j)$, $j=1, \dots, m$ and $C_k \in C$ is the class of the example.

Define P_{S_i, C_k} , the probability of occurrence of examples of class C_k in the a set S_i , to be the proportion of examples in S_i that are in class C_k . As a measure of the randomness of example distribution in S_i over the possible; classes in C , the information measure is defined thus [59..61]:

$$I(S_i) = - \sum_{k=1}^p P_{S_i, C_k} \log_2 P_{S_i, C_k} \quad [59]$$

ID3 aims to partition S to produce subsets S_i in which the examples are distributed less randomly over the possible classes. To choose the attribute that would best achieve this, for each attribute A_j having more than one value in the examples in S , ID3 partitions S into subsets S_i consisting of ail examples in S having value V_i for attribute A_j . If the number of examples containing value V_i in S is $n(S)$ and the number of examples containing value V_i in subset S_i is $n(S_i)$, the information entropy of the resulting partition is given by:

$$E(A_j, S) = \sum_{V_i \in \text{Range}(A_j)} I(S_i) \frac{n(S_i)}{n(S)} \quad [59]$$

ID3 chooses as a node the attribute A_j that maximizes the following quantity:

$$\text{Gain}(A_j, S) = I(S) - E(A_j, S) \quad [59]$$

If $\text{Gain}(A_j, S)$ is the same for more than one attribute A_j , then one of them is randomly chosen.

ID3 is a non-incremental algorithm and is a good choice for building a classification rule set for problems in which a database of instances is available and is not likely to change. However, for problems in which new instances are expected to become available on a regular basis, it is preferable to accept instances incrementally, without needing to build a new decision tree from the beginning each time. Since ID3 was developed, it has been improved several times by Quinlan and other researchers. Schlimmer and Fisher have proposed ID4 [62], which incrementally builds a decision tree similar to that which ID3 would build. Instead of forming a decision tree from a window of instances, ID4 updates a tree based on each individually observed instance. Utgoff [63] has described ID5, which builds on ID4. But differs from ID4 in its method of replacing the test attribute. When ID4 test attribute, it discards the sub trees below the old test attribute. ID5 reshapes the tree by pulling the best attribute up from below. An experimental comparison of ID3, ID4 and ID5 is given in [63].

A problem with ID3 is that the decision tree produced might not be as general as it could be. This is caused by the fact that decision rules can sometimes involve unnecessary or irrelevant conditions. When the algorithm chooses an attribute for branching out of a node, it produces a branch for each value of the attribute. However, some of those values may be irrelevant to the classification. Those irrelevant values will result in overspecialized classification rules. Cheng et al. [60] have developed GID3 (General version of ID3) to overcome this problem. Essentially, the algorithm is similar to ID3 except that not every value of the attribute is chosen to produce a branch. In order to avoid branching on irrelevant values of the attribute, only values that appear to be relevant; according to their information measure (a quantity similar to the information measure used in ID3), may potentially be branched on. A user-defined tolerance level is adopted to specify the degree of tolerance for the deviation of the entropy measure of an attribute-value pair from the minimal entropy measure over all pairs. Cheng et al. have not described how to set the tolerance level systematically.

4.3.1.2 RULES-3 Inductive Learning Algorithm

In recent years, there has been a growing amount of research on inductive learning. In its broadest sense; induction (or inductive inference) is a method of moving from the particular to the general - from specific examples to general rules. Induction can be considered as the process of generalizing a procedural description from presented or observed examples. [43..45] These examples may be specified by an expert as a good tutorial set, or may come from some neutral source such as an archive. The induction process will attempt to find a method of classifying an example expressed as a function of the attributes that explains the training examples and that may also be used to classify previously unseen cases.

RULES-3 [64] is a simple algorithm for extracting a set of classification rules from a collection of examples for objects belonging to one of a number of known classes. An object must be described in terms of a fixed set of attributes, each with its own range of possible values, which could be nominal or numerical. For example, attribute "length" might have nominal values {short, medium, long} or numerical values in the range {-10, 10}.

An attribute-value pair constitutes a condition in a rule. If the number of attributes is N_a , a rule may contain between one and N_a conditions. Only conjunction of conditions is permitted in a rule and therefore the attributes must be all different if the rule comprises more than one condition.

RULES-3 extracts rules by considering one example at a time. It forms an array consisting of all attribute-value pairs associated with the object in that example, the total number of elements in the array being equal to the number of attributes of the object. The rule forming procedure may require at most N_a iterations per example. In the first iteration, rules may be produced with one element from the array. Each element is examined in turn to see if, for the complete example collection, it appears only in objects belonging to one class. If so, a candidate rule is obtained with that element as the condition. In either case, the next element is taken and the examination repeated until all elements in the array have been considered. At this stage, if no rules have been formed, the second iteration begins with two elements of the array being examined at a time. Rules formed in the second iteration therefore have two conditions. The procedure continues until an iteration when one or more candidate rules can be extracted or the maximum number of iterations for the

example is reached. In the latter case, the example itself is adopted as the rule. If more than one candidate rule is formed for an example, the rule that classifies the highest number of examples is selected and used to classify objects in the collection of examples. Examples of which objects are classified by the selected rule are removed from the collection. The next example remaining in the collection is then taken and rule extraction is carried out for that example. This procedure continues until there are no examples left in the collection and all objects have been classified. This algorithm can be summarized as follows:

Step 1. Define ranges for the attributes, which have numerical values and assign labels to those ranges

Step 2. Set the minimum number of conditions (N_{cmin}) for each rule

Step 3. Take an unclassified example

Step 4. $N_c = N_{cmin} - 1$

Step 5. If $N_c < N_a$ then $N_c = N_c + 1$

Step 6. Take all values or labels contained in the example

Step 7. Form objects which are combinations of N_c values or labels taken from the values or labels obtained in Step 6

Step 8. If at least one of the objects belongs to a unique class then form rules with those objects; ELSE go to Step 5

Step 9. Select the rule, which classifies the highest number of examples

Step 10. Remove examples classified by the selected rule

Step 11. If there are no more unclassified examples then STOP; ELSE

go to Step 3

Here N_c is the number of condition(s) for each rule and N_a is the number of attributes for each example.

4.3.1.2.1 A simple Problem Illustrating the Operation of RULES-3

The following problem will be used to illustrate the operation of RULES-3 step by step. The problem, adapted from [65], involves three nominal attributes: Car-Size, Fuel and Max-Speed. Car-Size has three values, large, compact and medium. Fuel has three values, propane, petrol and diesel. Max-Speed has three values; high, average and low. There are three classes of acceleration, good, excellent and poor. The example set for the problem is given in Table 4.2.

No	Car-size	Fuel	Max-Speed	Acceleration
1	Large	Diesel	High	Good
2	Large	Propane	High	Good
3	Compact	Petrol	High	Excellent
4	Large	Petrol	High	Excellent
5	Medium	Diesel	Low	Good
6	Compact	Petrol	Low	Good
7	Medium	Petrol	Average	Excellent
8	Medium	Diesel	Average	Poor

Table 4-2 Training set for car acceleration problem adapted from [65]

In Step 1, none of the attributes have numerical values in the problem, so ranges are not defined.

In Step 2, the minimum number of conditions is set to 1.

In Steps 3-8, two rules are extracted from example 1. These are:

IF Car-Size IS large AND Fuel IS diesel THEN Acceleration IS good.

IF Fuel IS diesel AND Max-Speed IS high THEN Acceleration IS good.

In Step 9, because each of these rules can classify only one example (example 1), one of them can randomly be chosen. If the first rule is chosen, in Step 10, example 1 is classified using this rule. In step 11, the example is removed from the list of unclassified examples. The test in Step 12 for examples remaining to be classified is positive. Therefore, the procedure returns to Step 3 for a new iteration.

In the second iteration, there are seven unclassified examples. One rule can be extracted from example 2, namely,

IF Fuel IS propane THEN Acceleration IS good. Only example 2 can be classified using this rule and so it is removed from the list of unclassified examples.

In the third iteration, two rules are extracted from example 3. They are:

IF Car-Size IS compact AND Max-Speed IS high THEN Acceleration IS excellent.

IF Fuel IS petrol AND Max-Speed IS high THEN Acceleration IS excellent.

The first rule classifies only one example (example 3). The second rule can classify two examples (examples 3 and 4). Therefore, the second rule is selected. Using this rule, examples 3 and 4 can be classified and so they are removed from the list of unclassified examples.

In the fourth iteration, one rule can be extracted from example 5, namely,

IF Max-Speed IS low THEN Acceleration IS good. Using this rule, examples 5 and 6 can be classified and they are removed from the list of unclassified examples.

In the fifth iteration, two rules are extracted from example 7. These are:

IF Car-size IS Medium AND Fuel IS Petrol THEN Acceleration IS Excellent.

If Fuel IS Petrol AND Max-Speed IS Average THEN Acceleration IS Excellent.

As each of these rules can classify only one example (example 7), either of them can be chosen. If the first rule is chosen, example 7 can be classified. It is deleted from the list of unclassified examples.

In the sixth iteration, only one rule can be extracted from example 8, namely,

IF Fuel IS Diesel AND Max-Speed IS Average THEN Acceleration IS Poor

Since there are no more unclassified examples, the procedure ends. The above rule induction sequence is summarized in Table 4.3

Iteration	Example Considered	Number of Extracted Rules	Classified Examples	Unclassified Examples
1	1	2	1	2,3,4,5,6,7,8
2	2	1	2	3,4,5,6,7,8
3	3	2	3,4	5,6,7,8
4	5	1	5,6	7,8
5	7	2	7	8
6	8	1	8	-

Table 4-3 Summary of rule induction sequence for car acceleration problem

5 IMPLEMENTATION

In this thesis we propose alternative technique for banknote recognition. The technique employs RULES-3 inductive learning algorithm to extract the necessary set of rules and recognize banknote. 10, 3x3 masks were used to represent a banknote. Each banknote (or example) consists of 10 frequencies of each mask. The system was trained for 5 different types of banknotes' back and front sides by using 10 samples for each type and then tested for 25 "new" examples. The efficiency of the system, that is the ability of correctly classifying new examples, is found to be 96 %.

5.1 The Operation of the New Technique

The operation of banknote recognition is depicted in the following chart:

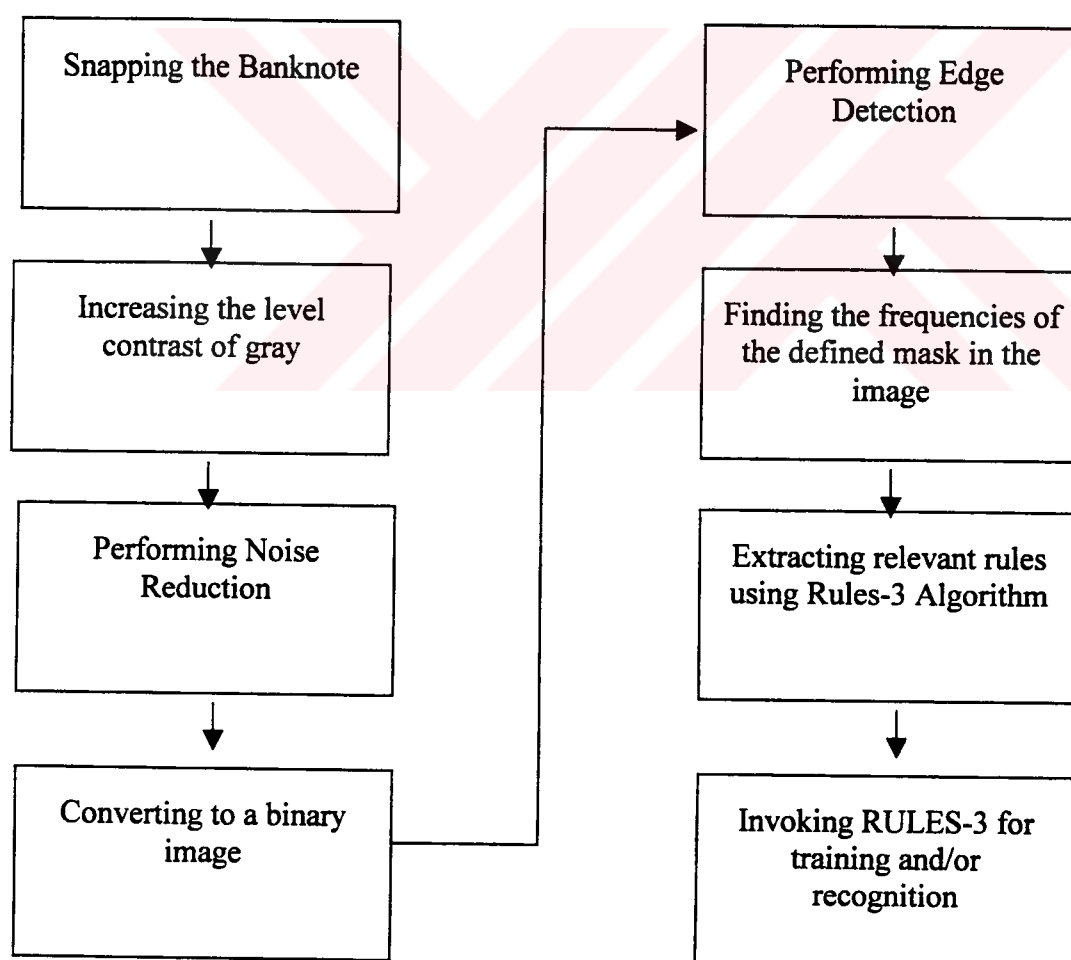


Figure 5-1 How The System Works

First of all, Banknotes must be learnt. This is performed as explained below. The whole banknotes in Turkey are given in Table-5.1



Table 5-1 The Whole Banknotes Used in Turkey

We consider each of these banknote as a candidate to be an example for RULES-3. In order to make an example we use the set of 10 3x3 masks, which are given in Table.5-2 [66], we can define 28 masks like these masks. These are given in Table 5.3 To get 10 masks among the 28 masks, 5 different types of banknotes' back and front sides by

using 10 samples for each type tested (Find the frequency) and then the average of frequencies is taken each type banknotes then descending by the average frequency. Thus, First 10 masks are chosen as attributes. These attributes are given in table 5.2

Mask1	Mask2	Mask3	Mask4	Mask5
				
Mask6	Mask7	Mask8	Mask9	Mask10
				

Table 5-2 The Set of 10, 3x3 Masks Used in the Thesis

Mask1	Mask2	Mask3	Mask4	Mask5	Mask6	Mask7
						
Mask8	Mask9	Mask10	Mask11	Mask12	Mask13	Mask14
						
Mask15	Mask16	Mask17	Mask18	Mask19	Mask20	Mask21
						
Mask22	Mask23	Mask24	Mask25	Mask26	Mask27	Mask28
						

Table 5-3 The set of 28, 3x3 Neighbors

5.1.1 Edge Detection

Before applying the masks, the operations of banknote recognition are depicted in the following order:

5.1.1.1 Snapping the Banknote

First of all, the banknote is snapped by using a video camera or a scanner and thus, the original image is obtained (Figure 5.2). This is a gray level image. For information about gray scale image (see chapter 3). In MATLAB software (main.m) this process is performed using `imread ()` function.



Figure 5-2 Gray level of image

A column graph of this image is given in figure 5.4.

5.1.1.2 Increasing the Contrast Level of the Image

Passing from a gray level image to a binary image with a thresholding value causes an information loss. In order to correct image defects and obtain binary image very efficiently. We decided to increase the contrast of image. This process is done (main.m) by `imadjust` command command in Matlab program.

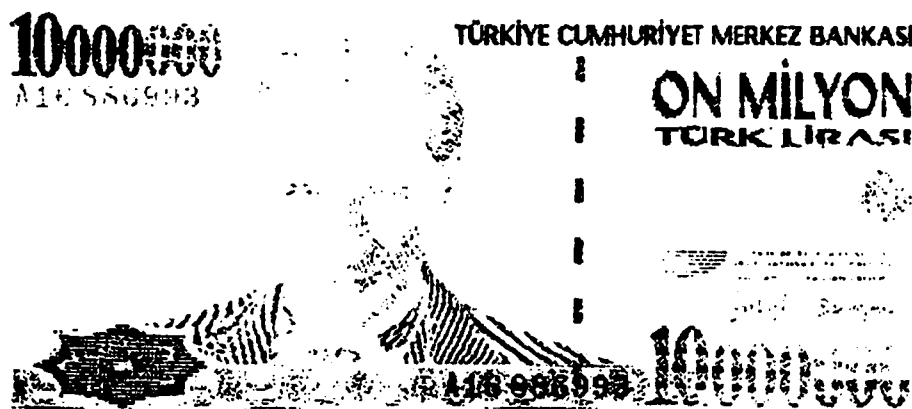


Figure 5-3 Contrast of Gray Level Image

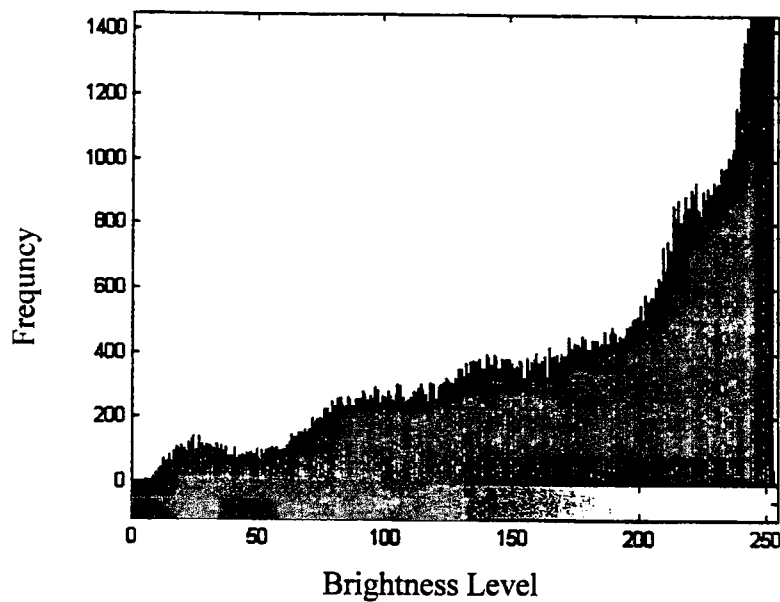


Figure 5-4 Histogram of Gray Level of Image

5.1.1.3 Performing Noise Reduction (image filtering)

In this process, to improve image quality, some noise reduction methods such as linear (mean) filtering and median filtering are applied. In figure 5.6, linear filters are applied on the image using filter2 command. In figure 5.7 Median filters are applied on the image using ordfilt2 command. These filters were tested for efficiency; that is, the ability of correctly classifying the given banknote. As a result, linear filtering is chosen, because the efficiency of linear filtering is higher than median filtering (table 5.4).

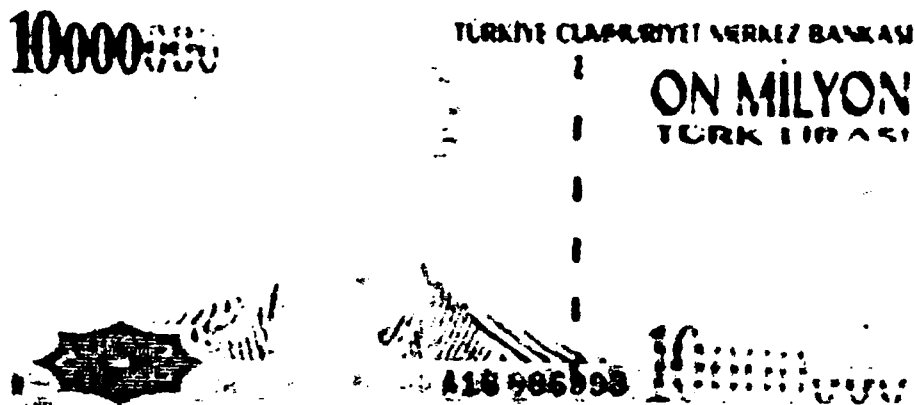


Figure 5-5 Linear Filter

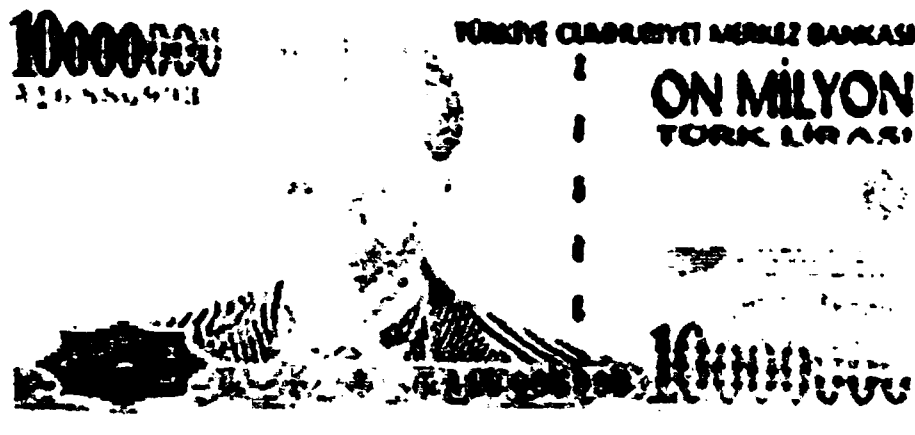


Figure 5-6 Median Filter

Filter Type	Efficiency (correctly classifying)
Median	92 %
Linear	96 %

Table 5-4 Comparing Filter Type

5.1.1.4 Converting to a binary image

In this process, we must decide for the threshold value. Some thresholding methods that were explained in chapter 3 are applied. In this thesis threshold value was decided to be 230. This means that if pixel value is greater then 230 this pixel turns white color otherwise it turns black color. Converting to binary image is performed using *im2bw* function in MATLAB.



Figure 5-7 Binary Image

Then the edge detection is applied to binary image. In this thesis the Sobel edge detection operator was chosen because of its good performance among the Roberts, Laplacian, Prewitt, Kirsch compass, Robinson compass, Canny and Frei-Chen operators. These operators deal with a 3x3 neighborhood that is given in figure 5.9. The discrete form of the Sobel operator is explained in chapter 3. This process is performed using the *edge* function in MATLAB.

$(i-1, j-1)$	$(i-1, j)$	$(i-1, j+1)$
$(i, j-1)$	(i, j)	$(i, j+1)$
$(i+1, j-1)$	$(i+1, j)$	$(i+1, j+1)$

Figure 5-8 3X3 Neighborhood

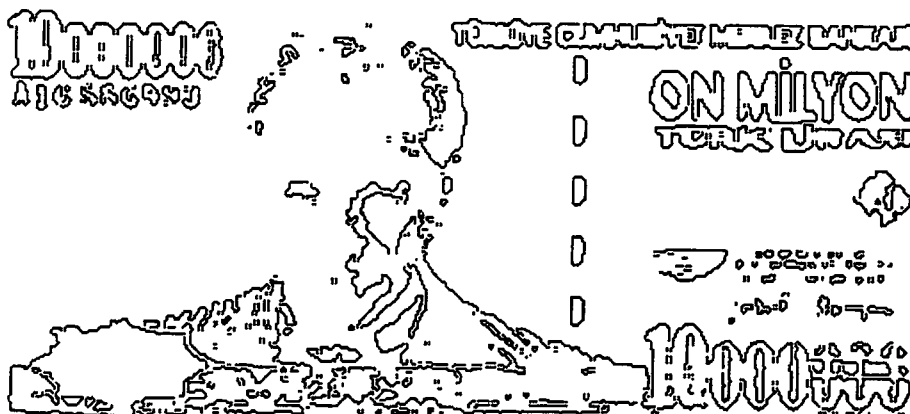


Figure 5-9 Edge Image

5.1.2 Learning Session

In order to extract the set of rules for all banknotes, the necessary set of examples must be formed. Each example has 10 attributes, each representing a 3 x 3 mask.

The value of each attribute is the frequency of the mask on the pattern being processed. Here the banknotes namely 250.000 TL, 500.000 TL, 1.000.000 TL, 5.000.000 TL and 10.000.000 TL are represented by frequency of Mask1, Mask2, Mask3...Mask10 respectively. In this thesis, the system was trained for 5 different types of banknotes' back and front sides by using 5 samples for each type. So the number of examples is 50. The set of examples for all banknotes shown in Table 5-1 is given in Tables 5-5 and 5-6.



Mask1	Mask2	Mask3	Mask4	Mask5	Mask6	Mask7	Mask8	Mask9	Mask10	Classes
161	143	268	1388	122	180	262	105	369	98	Back250TL
128	124	238	1265	104	144	227	94	338	95	Back250TL
134	93	206	1218	85	151	221	76	293	94	Back250TL
128	114	232	1302	106	140	243	95	364	106	Back250TL
132	147	305	1253	130	185	257	117	379	136	Back250TL
209	259	379	1765	231	272	375	231	834	252	Front250TL
182	236	363	1710	217	254	332	204	733	227	Front250TL
238	243	346	1778	226	244	363	220	723	238	Front250TL
212	252	348	1794	221	277	360	201	809	242	Front250TL
218	259	372	1830	251	278	358	225	812	246	Front250TL
90	100	245	1378	88	118	204	87	426	91	Back500TL
117	114	290	1571	133	168	246	100	460	136	Back500TL
87	92	261	1422	80	128	218	71	439	86	Back500TL
110	109	282	1461	99	150	216	87	489	98	Back500TL
84	93	267	1411	85	101	202	77	411	90	Back500TL
206	195	319	1744	211	215	250	178	783	210	Front500TL
187	200	290	1607	192	214	267	176	797	204	Front500TL
175	212	319	1740	187	221	288	177	798	192	Front500TL
183	208	283	1642	185	210	263	183	779	198	Front500TL
202	225	288	1656	187	209	254	198	743	200	Front500TL
246	190	344	1198	168	250	279	152	612	179	Back1000TL
322	278	384	1457	191	245	308	237	787	179	Back1000TL
260	213	302	1402	148	209	228	189	513	135	Back1000TL
292	220	303	1359	156	207	231	179	559	142	Back1000TL
292	220	303	1359	156	207	231	179	559	142	Back1000TL

Table 5-5 Set of examples part A

Mask1	Mask2	Mask3	Mask4	Mask5	Mask6	Mask7	Mask8	Mask9	Mask10	Classes
327	318	362	1167	269	378	374	271	1102	275	Front1000TL
297	304	390	1207	250	361	376	250	1052	260	Front1000TL
344	302	394	1341	299	397	397	254	1172	305	Front1000TL
296	319	385	1251	311	392	367	278	1122	308	Front1000TL
310	283	416	1201	271	361	379	251	1040	263	Front1000TL
248	225	357	1312	178	204	303	197	737	178	Back5000TL
280	241	354	1401	208	197	318	205	736	205	Back5000TL
271	252	402	1170	209	254	303	207	708	220	Back5000TL
303	297	444	1365	248	271	340	239	824	261	Back5000TL
274	232	418	1225	220	273	308	228	697	227	Back5000TL
215	310	285	1207	261	275	299	265	1076	268	Front5000TL
270	261	299	1291	243	269	310	239	1018	253	Front5000TL
217	299	300	1084	251	273	310	229	1052	280	Front5000TL
228	296	323	1170	258	291	318	241	948	269	Front5000TL
233	297	311	1157	245	271	301	246	1017	269	Front5000TL
262	263	318	1113	200	272	276	229	857	217	Front10000TL
245	260	319	1058	216	216	287	245	885	227	Front10000TL
251	238	327	1210	217	223	274	213	863	229	Front10000TL
227	249	308	1128	202	203	271	212	848	235	Front10000TL
236	258	300	1135	185	226	298	220	851	195	Front10000TL
197	230	254	928	196	241	204	203	846	217	Back10000TL
171	192	236	734	159	162	201	174	766	176	Back10000TL
157	190	225	715	156	166	197	164	690	162	Back10000TL
148	178	215	633	128	171	185	155	687	142	Back10000TL
168	208	215	699	168	181	184	186	772	175	Back10000TL

Table 5-6 Set of examples part B

After the necessary set of examples is formed, RULES-3 is invoked to extract a set of rules. If six quantisation levels are defined for each attribute and minimum number of attributes are 10. The set of rules shown in Table.5-7 can be extracted by RULES-3

IF 128=<m1<172 AND 130=<m2<168 AND 246=<m3<286 AND
 1233=<m4<1433 AND 119=<m5<158 AND 151=<m6<201 AND
 256=<m7<292 AND 71=<m8<106 AND 293=<m9<440 AND
 86=<m10<123 THEN CLASS IS Back250TL

IF 172=<m3<216 AND 206=<m4<244 AND 326=<m6<366 AND
 1633=<m13<1833 AND 197=<m16<236 AND 251=<m17<301 AND
 328=<m21<364 AND 176=<m22<211 AND 587=<m23<734 AND
 197=<m24<234 THEN CLASS IS Front250TL

IF 84=<m1<128 AND 92=<m2<130 AND 206=<m3<246 AND
 1233=<m4<1433 AND 80=<m5<119 AND 101=<m6<151 AND
 184=<m7<220 AND 71=<m8<106 AND 293=<m9<440 AND
 86=<m10<123 THEN CLASS IS Back500TL

IF 172=<m1<216 AND 168=<m2<206 AND 286=<m3<326 AND
 1433=<m4<1633 AND 158=<m5<197 AND 201=<m6<251 AND
 256=<m7<292 AND 176=<m8<211 AND 734=<m9<881 AND
 197=<m10<234 THEN CLASS IS Front500TL

IF 304=<m1<348 AND 244=<m2<282 AND 366=<m3<406 AND
 1433=<m4<1633 AND 158=<m5<197 AND 201=<m6<251 AND
 292=<m7<328 AND 211=<m8<246 AND 734=<m9<881 AND
 160=<m10<197 THEN CLASS IS Back1000TL

IF 304=<m1<348 AND 282=<m2<320 AND 366=<m3<406 AND
 1233=<m4<1433 AND 275=<m5<314 AND 351=<m6<401 AND
 364=<m7<400 AND 246=<m8<281 AND 1028=<m9<1175 AND
 271=<m10<308 THEN CLASS IS Front1000TL

Table 5-7 Extracted set of rules by RULES-3 for the banknotes shown in table 5-1 part A

IF 216=<m1<260 AND 206=<m2<244 AND 326=<m3<366 AND
 1233=<m4<1433 AND 158=<m5<197 AND 201=<m6<251 AND
 292=<m7<328 AND 176=<m8<211 AND 734=<m9<881 AND
 160=<m10<197 THEN CLASS IS Back5000TL

 IF 260=<m1<304 AND 244=<m2<282 AND 286=<m3<326 AND
 1233=<m4<1433 AND 236=<m5<275 AND 251=<m6<301 AND
 292=<m7<328 AND 211=<m8<246 AND 881=<m9<1028 AND
 234=<m10<271 THEN CLASS IS Front5000TL

 IF 260=<m1<304 AND 244=<m2<282 AND 286=<m3<326 AND
 1033=<m4<1233 AND 197=<m5<236 AND 251=<m6<301 AND
 256=<m7<292 AND 211=<m8<246 AND 734=<m9<881 AND
 197=<m10<234 THEN CLASS IS Front10000TL

 IF 128=<m1<172 AND 168=<m2<206 AND 206=<m3<246 AND
 633=<m4<833 AND 119=<m5<158 AND 151=<m6<201 AND
 184=<m7<220 AND 141=<m8<176 AND 587=<m9<734 AND
 160=<m10<197 THEN CLASS IS back10000TL

Table 5-8 Extracted set of rules by RULES-3 for the banknotes shown in table 5-1 part B

5.1.3 Recognition of Banknotes

In order to recognize a banknote, the following steps are required

- Find edges of banknotes
- Find mask frequencies
- Recognize a banknote using extracted rules.

For example, for the front view of one million (Front1000TL), first, the edge of banknote can be obtained using the same procedure explained above. The banknote is then scanned pixel by pixel from bottom to top and from left to right using ten masks. The frequencies were calculated as follows.

Mask1 Mask2 Mask3 Mask4 Mask5 Mask6 Mask7 Mask8 Mask9 Mask10

299 306 402 1245 295 373 380 261 1056 293 Back1000TL

The system respond us the relevant rules

IF $260 \leq m1 < 304$ AND $282 \leq m2 < 320$ AND $366 \leq m3 < 406$ AND
 $1233 \leq m4 < 1433$ AND $275 \leq m5 < 314$ AND $351 \leq m6 < 401$ AND
 $364 \leq m7 < 400$ AND $246 \leq m8 < 281$ AND $1028 \leq m9 < 1175$ AND
 $271 \leq m10 < 308$ THEN CLASS IS Back1000TL

In this rule set all attributes (10) are satisfied, if 10 attributes are not satisfied, system looks for which rules set satisfy nine, eight or seven attributes. If system doesn't find less than seven attributes the banknote wont be recognized.

In this thesis 25 "new" examples shown in table 5-9 were tested. The efficiency of the system, that is the ability of correctly classifying new examples, is found to be 96 % that means 24 banknotes are correctly classified. One banknote is not correctly classified. We think that if set of examples is large in size, the efficiency of the system that is the ability to correctly classify will be close to %100.

Mask1	Mask2	Mask3	Mask4	Mask5	Mask6	Mask7	Mask8	Mask9	Mask10	
299	306	408	1245	273	373	380	261	1025	293	Front1000TL
320	283	404	1188	255	361	347	244	1047	241	Front1000TL
268	208	311	1325	161	199	219	186	543	150	Back1000TL
272	270	371	1467	194	250	318	211	710	195	Back1000TL
259	234	335	944	188	257	306	219	891	219	Back5000TL
105	74	278	1439	82	127	205	58	414	87	Back500TL
200	237	308	1666	211	220	267	183	818	224	Front500TL
247	182	286	1265	145	215	251	163	572	142	Back1000TL
271	231	341	1293	164	249	290	188	616	165	Back5000TL
241	173	314	1301	138	230	259	158	527	128	Back1000TL
302	309	387	1106	269	400	382	266	1167	291	Front1000TL
328	290	412	1167	275	398	388	261	1032	277	Front1000TL
298	289	367	1094	262	342	377	268	999	267	Front1000TL
274	234	395	1203	210	259	319	207	714	210	Back5000TL
285	274	410	1365	214	228	311	227	687	212	Back5000TL
242	220	395	1322	191	228	301	198	691	204	Back5000TL
241	298	286	1054	255	267	267	246	1016	266	Front10000TL
234	285	336	1652	257	257	316	239	1033	269	Front5000TL
227	270	312	1837	262	251	287	222	1324	272	Front5000TL
159	193	224	703	152	172	206	158	753	164	Back10000TL
146	190	228	689	159	186	178	169	771	168	Back10000TL
228	202	383	1532	189	240	320	165	612	191	Back250TL
216	245	367	1997	214	281	337	209	755	233	Front250TL

Table 5-9 Examples to be Recognized

6 CONCLUSION

In this thesis, an alternative technique is proposed for banknote recognition. The advantages of this technique can be summarized as follows:

- Each side of a banknote is represented by a rule. So the pattern is not stored in the memory in graphics form. This saves memory space.
- Because the number of conditions in each rule and the total number of rules are not big in number, the decision can be made in a short time.
- It is easy and cheap to develop software and hardware for this technique since it is not complicated

Application areas of this system can be summarized as follows:

- Petrol station automats
- Parking automats
- Currency exchange machines
- Vending machines: food, drinks, magazines; batteries, videocassettes, compact discs etc.
- Video games machines
- Cash depositing points

In this thesis, the proposed banknote recognition system is unable to recognize counterfeit banknotes. Future work can be carried out on this special field.

7 APPENDIX

In this thesis, image-processing methods (e.g. converting gray level, increasing contrast of image, converting binary image and edge detection) are performed using MATLAB Software. In order to implement some image processing command, image-processing toolbar must be installed with MATLAB Software.

7.1 Files In the Disk

cercsmall.m: used for cutting pictures edges.

fmask.m: used for finding frequencies of defined ten masks.

main.m: includes complete program. This program contains as follows;

Run the rgb2gray.m program

Run the imadjust.m program

Run the filter2.m program

Run the im2bw.m program

Run the edge.m program

Run the cercsmall.m program

Run the fmaskall.m

Write the frequencies of the mask on the tez.inp file to be recognized

Thesis.exm: This file includes set of examples to extracted rules

Thesis.atr: This file includes attributes values.

Thesis.cls: This file includes class values.

7.1.1 Main.m (Main Program)

```
name=input('Banknote Name.....','s');
```

```
uz='.inp'; // extension of input file
```

```
location='D:\mythesis\rules3\';
```

```
isim=sprintf('%s%s',name,uz);
```

```

isim=sprintf('%s%s',location,isim);

fid = fopen(isim,'w');      // open input file

a=imread(name);            // read the image name that exists in the computer.

b=rgb2gray(a);             // convert it to gray level

c=imadjust(b, [0.3 0.5]);  // performing contrast.

h=fspecial('average',3);   // linear filter mask

d=uint8(round(filter2(h,c))); // applying linear filter

f=im2bw(d,9);              // converting binary image

g=edge(f,'sobel');         // applying sobel operator

ceresmall;                 // cutting edges of image

fmask;                     // finding frequency defined 10 masks.

mask=[cm01 cm02 cm03 cm04 cm05 cm06 cm07 cm08 cm09 cm10];

fprintf(fid,'%d,%d,%d,%d,%d,%d,%d,%d,%d,%d',mask); // writing mask values
on the input file

fprintf(fid,'%s\n',ad); // writing banknote name on the input file

fclose(fid); // closing input file.

```

7.1.2 Fmask.m (Finding Mask Frequency)

```

m01=[1 0 0;0 1 0;0 0 1]; // defining masks

m02=[0 1 0;0 1 0;0 0 1];

m03=[0 0 0;1 1 0;0 0 1];

m04=[0 0 0;1 1 1;0 0 0];

m05=[0 1 0;0 1 0;1 0 0];

m06=[0 0 1;0 1 0;1 0 0];

m07=[0 0 0;0 1 1;1 0 0];

```

```

m08=[1 0 0;0 1 0;0 1 0];
m09=[0 1 0;0 1 0;0 1 0];
m10=[0 0 1;0 1 0;0 1 0];
cm01=0;cm02=0;cm03=0;cm04=0;cm05=0;
cm06=0;;cm07=0;cm08=0;cm09=0;cm10=0;    // making equal zero
[q v]=size(g);          // size of edges images
[p r]=find(g);          // finding black pixels
w=[p r];
[u t]=size(w);
for m=1:u-1
z=w(m,:);
s=(g(z(1)-1:z(1)+1,z(2)-1:z(2)+1)); // finding 3x3 neighbours
    if s==m01          // comparing defined masks
        cm01=cm01+1;
    elseif s==m02
        cm02=cm02+1;
    elseif s==m03
        cm03=cm03+1;
    elseif s==m04
        cm04=cm04+1;
    elseif s==m05
        cm05=cm05+1;
    elseif s==m06
        cm06=cm06+1;
    elseif s==m07
        cm07=cm07+1;

```

```

elseif s==m08
    cm08=cm08+1;
elseif s==m09
    cm09=cm09+1;
elseif s==m10
    cm10=cm10+1;
end
end.

```

7.1.3 Cercsmall.m (Cutting Edges of Image)

```

[x y]=size(g);
for i=1:x-2
    for j=1:5
        k=g(i+1,j+1);
        if k==1
            g(i+1,j+1)=0;
        end
    end
end
end

```

```

for j=1:y-2
    for i=x-7:x-2
        k=g(i+1,j+1);
        if k==1
            g(i+1,j+1)=0;
        end
    end
end

```

```

        end
    end
    for i=1:x-2
        for j=y-7:y-2
            k=g(i+1,j+1);
            if k==1
                g(i+1,j+1)=0;
            end
        end
    end
end
for j=1:y-2
    for i=1:5
        k=g(i+1,j+1);
        if k==1
            g(i+1,j+1)=0;
        end
    end
end
end
end

```

7.2 Hardware Requirements

Minimum Hardware requirement,

486 Processor,

2 MB Graphics Card;

150 MB disk capacity for installing MATLAB 5.0 Software

32 MB RAM capacities;

7.3 Software Requirements

The operating systems

Win95, Win 98, Win NT, Win 2000

Software

MATLAB 5.0 or MATLAB 5.2



REFERENCES

- 1 The Catalogue of Currencies, 2000, "<http://www.apis.sk/english/catalwf.htm>"
 - 2 Sodeco Cash Management Systems BNA5 General Information, Mars Electronic, Switzerland, 1995
 - 3 Jain R, Kasturi R and Schunck B, "Machine Vision" 1995
 - 4 Braggins D and Hollingum J., "The Machine Vision Source Book", New York, Tokyo, 1986
 - 5 Bruce G., Batchelor and Paul F. Whelan "Intelligent Vision System for Industry", pp19-21, Great Britain, 1997
 - 6 Gerhard Ritter, Joseph N. Wilson, "Computer Vision Algorithms in Image Processing", 1997
 - 7 Anil K. Jain," Fundamentals of Digital Image Processing", 1989
 - 8 Jahne Bernd, "Digital Image Processing", San Diego, 1993
 - 9 Bracewell Ronald N., "Two- Dimensional Imaging", 1995
 - 10 Scott E. Umbaugh, "Computer Vision and Image Processing", pp24-35, Upper Saddle River, NJ, 1998
 - 11 Russ John C. "The Image Processing Handbook 2. Edition", Raleigh, North Carolina, 1998
-

12 Image Processing Fundamentals- Contents, 2000,

“<http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip-Contents.html>”

13 Image Processing Fundamentals- Segmentation Part, 2000,

<http://www.ph.tn.tudelft.nl/Courses/FIP/noframes/fip-Segmenta.html>

14 ISODATA Unsupervised Classification, 2000,

“<http://www.lter.alaska.edu/~dverbyla/nrm641/unsup1.html>”

15 Lippmann R. P.,” Pattern Classification Using Neural Network” IEEE Comm. Magazine November 1989, pp. 47-64, November 1989

16 Bishop C.M. “Neural Network for Pattern Recognition”, New York: Oxford University Press, 1990

17 Sarle W.S. “Neural Network and Statistical Models” proceeding of the Nineteenth Annual SAS Users Groups International Conference, April 1994

18 Kohonen T., Barna G. and Chrisley R.; “ Statistical Pattern Recognition with neural network: benchmark studies ” proceeding IEEE Int. Conf on Neural Networks, San Diego, 1988,pp I-61-I-68, 1988

19 Chen C.H. “On the Relationship Between Statistical Pattern Recognition and Artificial Neural Networks” Neural Network in pattern recognition and their application, Singapore, 1991

20 Duda R. O and Hart P. E., Pattern Classification And Scene Analysis , New York: John Wiley& Sons,1996

21 Mallat, S.G. “Multifrequency Channel Decomposition of Images and wavelet models” IEEE Trans on. ASSP Vol. Assp-37, no 12 pp 2091-2110, December 1989

-
- 22 Laine, A. and Fan, J. "Texture Classification by wavelet Packet Signatures" IEEE trans on. PAMI, Vol Pami -15 no 11 pp 1186-1191, November 1993
- 23 Jain A. K. and Ramaswami M. D. " Classifier Design with Parzen Windows" Patter Recognition and artificial Intelligence E. S Gelsema and I. N. Kanal North-Holland, 1988
- 24 Burt, P. J., Adelson E.H. "The Laplacian Pyramid as a Compact image code" IEEE trans on COMM, Vol Comm-31 No 4 pp 532-540, April 1983
- 25 Kandel E. Schwartz J., Jessell T. Principle of Neural Science, New Jersey: Prentice Hall Inc. 1991
- 26 Nichols J. G., Martin A. R Wallace B. G. From Neuron to Brain, Sunderland: Sianuer Ass. Inc. 1992
- 27 Cover T. M. and Hart P. E., "Nearest Neighbor Pattern Recognition" IEEE Trans. On IT Vol It-13. pp 21-27. January 1967
- 28 Keller J. M., Gray, M. R., Givens J. A. " A fuzzy K-nearest Neighbor Algorithm" IEEE Trans On SMC Vol Smc-15 no: 4 pp 580-585, October 1985
- 29 Gagolowicz A., "Some application to the study of human vision" IEEE Trans on PAMI Vol Pami-3 No.5 pp 520-533 September 1983,
- 30 Hubel, D. H. and Wiesel T. N. "Ferrier Lecture: Functional Architecture of macaque Monkey Visual Cortex" no.198 pp1-59. 1997
- 31 Hubel D. H " Exploration of the Primary Visual Cortex, 1955-78, "Nature Vol-299, pp 515-524, October 1982

-
- 32 Mc Culloch W.S. and Pitts W. "A logical Calculus of the ideas Immanent in nervous activity" Bulletin of mathematical Biophysics. Vol. 5 pp115-133
- 33 Hebb. D. O the Organization Behaviour, Newyork : Willey Inc.
- 34 Rosen Blatt F. " The Perceptron A Probabilistic model for information storage and organization in the brain "Psychological Review vol. 65 pp.386-408
- 35 Anderson J and A. RosenFeld E. "Neuro computing I,II Cambridge", Mass: MIT, 1990
- 36 Widrow B, Winter G., Baxter R. A. " Layered Neural Nets for Pattern Recognition" IEEE Trans on ASSp vol. Assp, vol. Assp-36 no.7 pp.1109-1118, July 1998
- 37 Campell, F. W. " The human Eye as an Optical Filter Pro. of the IEEE, Vol.56 pp 1009-1014
- 38 Mcllwain J. T."Large Receptive Field And Spatial Transformation in the visual System" International Review of physiology, Neurophysiology II, 1976
- 39 Bedzek, J. C and Pal N. R "Two soft Relatives of learning vector Quantization" Neural Networks vol.8 no .5 pp 729-743, 1995
- 40 Fransconi P. Gori M. Soda G. " Links Between LVQ and backpropogation" pattern recognition letters vol.18 pp.303-310, January 1997
- 41 Movshon, J. A., Thompson, I. D., Tolhurst, D. J. "Spatial and Temporal Contrast Sensitivity of neurons in areas 17 and 18 of the cat's visual Cortex " J. Physiol., no 283, pp 101-120,1978
- 42 Nakakuki Y., Koseki Y., Tanaka M., "Inductive Learning in Probabilistic Domain", in proc. Eight National Conf. Boston, 1990

-
- 43 Forsyth R., "Machine Learning Principles and techniques", London, 1989
- 44 Hancox P. J., Mills W. J and Reid B. J., "Artificial Intelligence / Expert System", Ergosyst Associates, Lawrence, KS, 1990
- 45 Quinlan, J. R., "Induction, Knowledge and Expert System", in Artificial Intelligence Developments and Applications, North Holland, 1988.
- 46 Michalski R. S., "A theory and Methodology of Inductive Learning", in readings in machine learning, California, 1990
- 47 Charniak E., McDermott D., "Introduction the Artificial Intelligence", Colifornia, 1985
- 48 Tanimoto S. L., "The Elements Of Artificial Intelligent", Maryland, USA, 1987
- 49 Rubin S. H., "Expert Systems for Knowledge Acquisition", in proc. First World Congress on Expert Systems, Vol. 3, December 16-19, pp.1793-1799, Orlando, Florida, 1991
- 50 Kodratoff Y., "Introduction the Machine Learning", London, 1988
- 51 Michalski R.S. and Stepp R. E., "Learning from observation: Conceptual Clustering", in machine Learning and Artificial Intelligence Approach, Vol 1, pp.331-363, 1983
- 52 Kocabas S., "A review of learning", Knowledge Engineering Review, pp.195-222, 1991
- 53 Al-Attar A., "Rule Induction From Mythology to Methodology", London, 1991
- 54 Quinlan J. R., "Learning Logical Definitions from Relations", in Machine Learning, 5, Kluwer Publisher, pp.239-266, Boston, 1990

-
- 55 Shapiro S. C. et. al., "Encyclopedia of artificial intelligence", Second Edition, 1991
- 56 Hunt E. B., Marin J., Stone P. J., "Experiments in Induction", Newyork, 1966
- 57 Quinlan J. R., "Learning Efficient Classification Producers and their Application to Chess End Games", in Machine Learning, An Artificial Intelligence Approach, pp.463-482, CA, 1983
- 58 Dietterich T. G. and Michalski R. S., "A Comparative review of Selected Methods for Learning from Examples", Machine Learning, an Artificial Intelligence, pp.41-81, 1983
- 59 Evangelos I. P. et al., "A minimum entropy Approach to Rule Learning from Examples", IEEE Trans. Systems Man and Cybernetics, pp.621-635, 1992
- 60 Cheng J et al., "Improved decision Tress: a Generalized Version of ID3", in proc. Fifth Int. Conf. pp.100-106, on Machine Learning, The University of Michigan, Ann Arbor, MI, 1988
- 61 Mace R., "Management Information and the Computer", 1974
- 62 Schlimmer J. C., Fisher D., "A Case Study of incremental Concept Induction", in proc. Fifth National Conf. on AI, pp. 496-501, CA, 1986
- 63 Utgoff P. E., "ID5: an Incremental ID3", in proc. Fifth Int. Conference on Machine Learning, pp. 107-120, The University of Michigan Ann Arbor, MI, 1988
- 64 Pham D.T and Aksoy, M.S, "A New Algorithm for inductive learning", Journal of System Engineering, pp. 115-122, 1995

65 Yasdi, R. "Learning classification rules from database in the context of knowledge acquisition and representation, IEEE Trans. On knowledge and Data Engineering 3, pp 293-306.1991

66 Pham D.T, Bayro E.J, "Neural Computing for Noise Filtering, Edge Detection and Signature Extraction", Intelligent System Research Laboratory, Cardiff, UK



REFERENCES NOT CITED

Aksoy, M.S, Cagil G. and Turker, A.K, “Number plate recognition using inductive learning”, Robotics and autonomous systems 870, Elsevier Sciences, 2000

Nazif A.M.,Levine M. D., “Low Level image Segmentation: An expert System”, IEEE Trans.on Pattern Analysis and Machine Vision, Vol.6, No.5,pp.555-571, September 1984

Pham D.T, Bayro E.J, “Neural Computing for Noise Filtering, Edge Detection and Signature Extraction”, Intelligent System Research Laboratory, Cardiff, UK

Dixon J. K., “Pattern Recognition with partly Missing Data”, IEEE Trans. On SMC, Vol. SMC-9, no.10, pp.617-621, October 1989

Zahid H., “Digital Image Processing Practical Applications of Parallel Processing Techniques”, England, 1991

Levine Martin D., “A knowledge Based Computer Vision Systems”, Montreal Canada, 1978

Gilbert B., Joseph L., “Tutorial on Robotics”, Second edition IEEE Computer Society, 1986

Joseph L., “Industrial Machine Vision It is Practical?”Cardiff, 1988

Batchelor B. G., Hodgson D. C., “Automated Visual Inspection”, North Holland, 1985

Patterson D. W., “Introduction to Artificial Intelligence and Expert Systems”, London, 1990