İzmir Institute of Technology

The Graduate School of Engineering and Natural Sciences

RECONSTRUCTION OF X-RAY IMAGES

A Thesis in

Computer Engineering

by Hüseyin Cüneyt AKA

Submitted in Partial Fulfillment of the Requirements
for the Degree of

Master of Science

June 1997

We approve the thesis of **Hüseyin Cüneyt AKA**

Date of Signature

09/07/1997

Prof. Dr. Sıtkı Aytaç
Thesis Advisor
Chairman of Department of Computer Engineering

09/07/1997

Prof. Dr. Halis Püskülcü
Committee Member
Department of Computer Engineering

09/07/1997

Prof. Dr. Tanju Aktuğ
Committee Member
D.E.Ü. Faculty of Medicine,
Department of Pediatric Surgery

Acknowledgment

I wish to express my sincerely gratitude to

- my advisor, Prof.Dr. Sıtkı Aytaç, for his constant support and trust,
- Prof.Dr.Halis Püskülcü who shared his knowledge about data analysis,
- Prof.Dr. Tanju Aktuğ, for his assistance in obtaining VUR images and providing detailed information, resources about VUR disease,
- Associate Professor Mihran Tuceryan, who answered all my questions patiently about texture analysis,
- my family, for their patience and love,
- my friends in IIT for their helps and clues,
- Bahattin Tayanç who provided us necessary software and hardware tools and service,

In addition, I wish to acknowledge to the following organizations for their support.

- Computer Center of İzmir Institute of Technology, for providing necessary computing environments and tools.
- Dokuz Eylül University Faculty of Medicine, Department of Pediatric Surgery, for their valuable inputs utilized in the thesis.
- Biomed-3 Organization Committee, for motivation by awarding us for the presentation performed in Biomed-3 Conference.

Abstract

We have presented an integrated approach in retrieving, reconstructing, and storing images obtained from noisy X-rays in this study. The X-ray images are used to detect human body's invisible parts. The problem of blurring and uneven illumination is always faced. Although it is partially solved by the physicians via lighting the X-rays, this method is not working properly in some cases such as Vesico Ureteral Reflux disease. This may cause loss of some meaningful part of the information and failure in diagnosis process. In order to decrease such errors, some computational methods has been developed by means of image processing. Due to its very nature, reconstruction, retrieving and registration of x-ray images has been chosen as a subject of this study.

We have begun attacking the problem of reconstruction and extraction, then started to generate multi-layer hierarchical solutions. We have tried so many different approaches for each layer in our experiments. In each experiment, some methods produced accurate results, some methods did not. Thus, we have exerted every effort to optimize the solution for each layer. Although we have worked with limited number of sample images,(due to the problem of retrieving x-rays which is seen in this case) the results show us that, all the samples that we have processed, could have been reconstructed and stored as we have expected.

Storing of the huge amount of data is an another problem in our area of interest, because of image characteristics. Every kidney image consists of nearly 120.000 (around 300x400) pixels. However, in our case, the boundaries of kidney region are sufficient for diagnosis. In other words, storing the boundaries instead of complete image has the same precision. We detected and stored the kidney's boundary coordinates on both x and y axis. Although this was sufficient for our study, we have decided to develop a much more flexible file format by ordering x and y coordinate couples in counter clockwise direction with the same information for further studies such as computer aided diagnosis systems.

Öz

Bu çalışmada, gürültü düzeyleri yüksek Röntgen filmlerinden elde edilmiş görüntülerin sayısal ortama aktarımı, yeniden yapılandırılması ve saklanması üzerine çok katmanlı bir yöntem sunulmuştur. İnsan bedeninin bakıldığında gözle görülemeyen kısımlarının görüntülenmesi için kullanılan Röntgen filmlerinde daima düzgün olmayan ışık dağılımları ve bulanık alanlar bulunmaktadır. Bu sorunlar filmlerin arkadan aydınlatılması yöntemiyle kısmen çözülmüş olsa da her durumda bu çözüm yeterli olmamaktadır. Bu sorunların sonucu olarak film üzerindeki anlamlı bilgiler ya görülememekte veya kaybolmaktadır. Saydığımız bu nedenlerden dolayı Röntgen filmlerinin yeniden yapılandırılması bu çalışmanın konusunu olarak seçilmiştir. Yukarıda belirtilen sorunların tipik bir örneği olan Veziko Ureteral Reflü (VUR) hastalığına ait Röntgen filmleri yöntemin geliştirilmesi sırasında ele alınmış ve üzerinde çalışılmıştır.

Çalışmaya, problemin yeniden yapılandırma ve görüntülerin anlamlı bölümlerin çıkarılması açısından yaklaşılmış ve daha sonra çok katmanlı çözümler üretilmiştir. Her katman için bir çok değişik yöntem uygulanmış ve en uygun çözüm seçilmiştir. Karşılaşılan diğer bir sorun ise katmanlar arası uyuşmazlıklar olmuştur. Bu problem ise üretilen bütün çözümlerin tek bir sistem olarak ele alındıktan sonra katman bazındaki çözümlerin tekrar gözden geçirilmesi ve bazılarının değiştirilmesi yoluyla aşılmıştır. Sonuçta, ele alınan bütün görüntülerin işlenmesinden sonra elde edilen sonuçlar beklenildiği gibi yeniden yapılandırılabilmiştir.

Röntgen filmleriyle ilgili diğer bir sorun da çok fazla miktarda veri içeren bu görüntülerin sayısal ortamda saklanmasıdır. Çalışmalarımız sırasında gördük ki, VUR da hastalığın tanınmasında kullanılan anlamlı veriler ana nesnenin sınırlarında bulunmaktadır. Buradan yola çıkarak, sadece ana nesnenin sınırların saklamanın bütün görüntüyü saklamakla teşhis açısından aynı duyarlılığa sahip olduğunu gördük. Sonuç olarak, elimizdeki VUR görüntülerini yaklaşık 1/60 oranında sıkıştırmış olduk. Sıkıştırma sırasında daha ilerideki bilgisayar görselliği alanında yapılacak çalışmalara yardımcı olması açısından, ana şekle ait kenarları saat yönünün tersinde sıralayarak sakladık.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

1.1 Motivation

The problem of detection and reconstruction have conventionally been treated as separate issues in image processing. The medical image processing concept uses both issues to provide more clear and meaningful data to the physicians. However, there are so many different ways to retrieve data from human body's invisible parts, using the x-rays is the most common approach because of its ease of use and low of cost. Although, it is so common and used by physicians in many years, it is not sufficient for a strict diagnosis in some cases. It can be said that uneven illuminations and blurring, which always occurs, are the drawbacks of this system. This also effects the results of the diagnosis process. Vesico ureteral reflux (VUR) is a typical case, in which x-rays are always blurred and low illuminated, so that it is chosen as an object of this study. We have developed an multi-layered reconstruction system for VUR x-ray images and furthermore a data storing module has been presented.

1.2 Machine Vision

The goal of the machine vision system is to create a model of real world from images. *A machine vision system recovers useful information about a scene from its two-dimensional projections*. Since images are two-dimensional projections of the three-dimensional world, the information is not directly available and must be recovered. The recovery requires the inversion of a many-to-one mapping. To recover the information, knowledge about the objects in the scene and projection geometry is required.

To consider applications of a machine vision system and type of processing required, let us consider the three images shown in Figures 1-1 to 1-3. These figures show three different applications of a machine vision system. The information recovered

by the vision system in the three cases is different. In the first figure, results of designed system diagnosis of a disease using computed tomography images. Machine vision systems help a physician to recover information by enhancing the images. Quantitative measurements on regions of interest can also be made easily available. Such systems are being developed for all imaging modes useful in different aspects of health care. Similar applications are being developed for inspection of industrial, agricultural, and other products. Machine vision systems have been used for quality control of products ranging from pizza to turbine blades, from submicron structures on wafers to auto-body panels, and from apples to oranges.

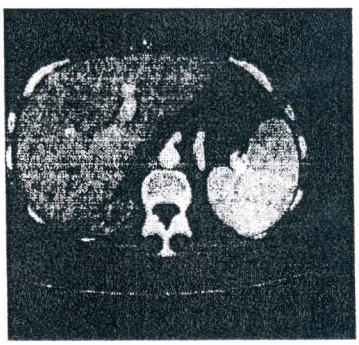


Figure 1-1 Medical images may be processed by a computer vision system to assist in diagnosis. This image is a contrast-enhanced CT image of a human liver displayed at the soft tissue window settings.

Figure 1-2 shows two pairs of images acquired by a mobile robot. Each pair represents a stereo pair at a particular time instant. These images are used to recover three-dimensional structure of the environment by the robot for autonomously navigating in its environment. The information obtained from stereo and motion is combined to get a robust map of the environment at a resolution that is sufficient for the task. Such techniques are useful in autonomous navigation of automobiles, airplanes, tanks and robots.

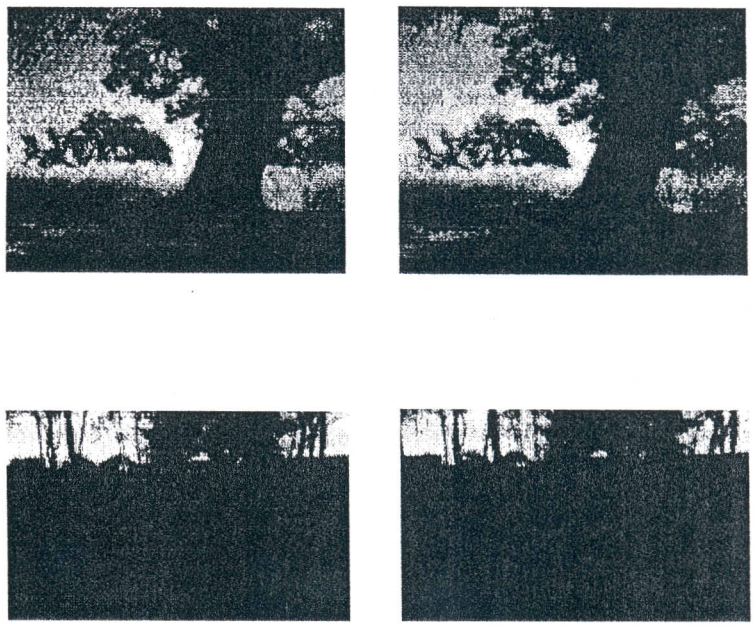


Figure 1-2 Two pairs of stereo images acquired by a mobile robot. These will be used in recovering the layout of the environment by the robot. (Images provided by Robert Bolles at SRI.)

Figure 1-3 An image of arctic region. This image should be analyzed to find age and size of ice floes and other objects. (arctic image provided by Jaison Daida at the University of Michigan.)

Figure 1-3 shows an image of an arctic region taken from satellite. Different regions in the image correspond the ice floes of different age. Such images are routinely taken by satellites. These images are extremely important in weather forecasting, global change analysis, agriculture and forestry, and the other applications. Machine vision system are playing important role in analysis and management of the exceedingly large volume of data collected by satellites.

1.3 Relationship to Other Fields

Many field are related to machine vision. Techniques developed are used in many areas for recovering information from images.

Image processing is a well developed field. Image processing techniques usually transform images into other images; the task of information recovery is left to a human user. This field includes topics such as image enhancement, image compression, and correcting blurred or out-of-focus images. On the other hand, machine vision algorithms take images as inputs but produce other types of outputs, such as representation for the object contours in an image. Thus, emphasis in machine vision is on recovering information automatically, with minimal interaction with a human. Image processing algorithms are useful in early stages of a machine vision system. They are usually used to enhance particular information and suppress noise.

Computer graphics generates images from geometric primitives such as lines, circles, and free-form surfaces. Computer graphics techniques play a significant role in visualization and virtual reality. Machine vision is the inverse problem: estimating the geometric primitives and other features from the image. Thus, computer graphics is the synthesis of images, machine vision is the analysis of the images. In the early days of these two fields, there was not much relationship between them, but in the last few years these two fields have been growing closer. Machine vision is using curve and surface representations and several other techniques from computer graphics, and computer graphics is using many techniques from machine vision to enter models into the computer for creating realistic images. Visualization and virtual reality are bringing these two fields closer.

Pattern recognition classifies numerical and symbolic data. Many statistical and syntactical techniques have been developed for classification of patterns for classification of patterns. Techniques from pattern recognition play an important role in machine vision for recognizing objects. In fact, many industrial applications rely heavily on pattern recognition.

Artificial intelligence is concerned with designing systems that are intelligent and with studying computational aspects of intelligence. Artificial intelligence is used to analyze scenes by computing a symbolic representation of the scene contents after the

images have been processed to obtain features. Artificial intelligence may be viewed as having three stages: perception, cognition, and action. Perception translates signals from the world into symbols, cognition manipulates symbols, and action translates symbols into signals that effect changes in the world. Many techniques from artificial intelligence play important roles in all aspects of computer vision. In fact, computer vision is often considered a subfield of artificial intelligence.

Design and analysis of *neural networks* has become a very active field in the last decade. Neural Networks are being increasingly applied to solve some machine vision problems. Since this field is in it infancy, there are no established techniques for machine vision yet.

In our study, we will consider one of the subfield of the machine vision, medical image processing. The study covers that the processing of X-ray images for reconstruction.

1.4 Medical Imaging

The study of medical imaging is concerned with the interaction of all forms of radiation with tissue and the development of appropriate technology to extract clinically useful information from observations of this interaction. Such information is usually displayed in an image format. Medical images can be as simple as a projection or shadow image -as first produced by Roentgen nearly 100 years ago and utilized today as a simple chest X-ray- or as complicated as a computer reconstructed image- as produced by computerized tomography (CT) using X-rays or by magnetic resonance imaging (MRI) using intense magnetic fields.

Although, strictly speaking, medical imaging began in 1895 with Roentgen's discoveries of X-rays and of the ability of X-rays to visualize bones and other structures within the living body (Choe et al., 1993) contemporary medical imaging began in the 1970s with the advent of computerized topography (Choe et al., 1993). Early, or what we call *classical*, medical imaging utilizes images that are a direct manifestation of the interaction of some form of radiation with tissue. Three examples will illustrate what is the mean of classical imaging. Fist is the conventional X-ray procedure in which a beam of X-rays is directed through the patient onto a film. The developed film provides a shadow the image of the patient which is the direct representation of the passage of X-

rays through the body. Although such images are not quantitative, they do provide some measure of the attenuation of X-rays in tissue. Thus a section of soft tissue will appear darker than an equally thick section of bone, which attenuates more of the X-rays. It should be noted that even with current technological developments conventional X-ray imaging still represents the major imaging procedure at most medical facilities.

As a second example of classical imaging, consider a conventional nuclear medicine procedure. Here a radioactive material is injected into the patient and its course followed by a detector which is moved over the patient in a specified manner. Although the image recorded by the detector generally has poor spatial resolution, its real advantage is that it provides a measure of physiological function from the time course of the radioisotope uptake. Clearly the conventional nuclear medicine image is a direct measure of the location and concentration of the radioactive isotope used.

As a final example of classical imaging, consider conventional medical ultrasound. Here, a pulse of ultrasonic energy is propagated into the patient and the backscattered echo signal is recorded by the same transducer. By angulating or moving the transducer(or by using a transducer array) positionally sequential echo signals are recorded and a cross-sectional image of the subject is displayed directly on a video monitor. Ultrasound images are really a mapping of echo intensities and are a direct result of the interaction of the ultrasound with tissue.

The coverage of medical imaging consist of two different processes:1. the collection of data concerning the interaction of some form of radiation with tissue, and 2. the transformation of these data into an image (or a set of images) using specific mathematical methods and computational tools. In this study we will examine that the second part of the medical imaging and more.

The first device capable of producing true reconstructed images was developed by G.N. Hounsnfield (Choe et al., 1993) in 1972 at EMI in England. Hounsnfield's X-ray computerized tomography device was based in part on mathematical methods developed by A.M. Cormack (Choe et al., 1993) a decade earlier. For their efforts Hounsnfield and Cormack were awarded the Nobel Prize in medicine in 1979. Put quit simply, CT imaging is based on the mathematical formalism that states that if object is viewed from a number of different angles, then a cross-sectional image of it can be computed (or reconstructed). Thus X-ray CT yields an image that is essentially a mapping of X-ray attenuation tissue density.

Table 1-1 Image reconstruction algorithms

2-D and 3-D Projection Reconstruction	2-D Projection Reconstruction	Parallel-Beam Mode
		Fan-Beam Mode
	3-D Projection Reconstruction	Parallel-Beam Mode
		Cone-Beam Mode
Iterative Method	Algebraic Reconstruction Technique (ART)	
	Maximum Likelihood Reconstruction (MLR) or Expectation Maximization (EM)	
Fourier Reconstruction	Direct Fourier Reconstruction (DFR)	
	Direct Fourier Imaging (DFI) in NMR	

The introduction of X-ray CT in 1972 represents the real beginning of modern imaging and has altered forever our concept of imaging are merely taking a picture. It has also led to the development of 3-D imaging and is making quantitative imaging a reality. The application of reconstructive tomography to conventional nuclear medicine imaging has led to the development of two new imaging modalities: single photon emission computed tomography (SPECT) and positron emission tomography (PET). Similar applications to the laboratory technique of nuclear magnetic resonance (NMR) has led to the magnetic resonance imaging (MRI). The CT concept is currently being extended to 3-D magnetoencephalography, electrical impedance tomography, and photon migration tomography, to name a few. Inherent to the development of these imaging modalities has been the development of new reconstruction techniques, which are detailed in Table 1-1.

1.5 Contributions

This study presents an integrated approach in detection and reconstruction of 2-D x-ray images. We begin by conducting a close examination on image enhancement , and medical image nature. This yields an advanced understanding of the various aspects of medical image reconstruction which lights our way in development of our study. We then proceed to formally develop the system for the solution reconstruction and storing of VUR x-ray images. The VUR x-rays supplied from Dokuz Eylül University School of Medicine Department of Pediatric Surgery, has been used overall study. At the end of

the study, regarding the system developed, a Motif based X-Windows application has been developed to help to the physicians in their work.

1.6 Organization

We organize this thesis as follows:

- Chapter 1 presents a brief introduction to the study
- Chapter 2 covers literature survey on image processing.
- Chapter 3 includes fundamentals of early image processing and modeling of gray scale images.
- Chapter 4 introduces image enhancement, reconstruction and converting techniques.
- Chapter 5 presents most common edge detection approaches in detailed.
- Chapter 6 presents our study in both experimental and application framework ways.
- Chapter 7 includes conclusion and future works
- Appendix A includes the application framework and images.

In the first chapter we have briefly introduced the problem and than explained why we have exerted all of our efforts on this subject. This chapter has also includes introducing to image processing and medical imaging. At the end of the chapter, the structure of overall study is placed.

Since, the power of computers are growing tremendously, image processing concepts are being used in many applications. It can also be implemented as an interdisciplinary framework. The reasons above have made it very popular in recent days. In this chapter, it is presented some of application and a literature survey on computer vision and image reconstruction.

The third chapter models and processing algorithms that rely significantly on image gray-value or intensity variation are considered. The algorithms developed in this chapter are seldom used alone to achieve a system goal (such as object inspection), but rather are often the initial preprocessing of a gray-level image.

In the fourth chapter, it is focused that the enhancement, reconstruction and image converting techniques. There were so many different approaches stated in the

literature. So best one must be selected which produces the best results. The most of techniques presented in this chapter has been used in the process of reconstruction.

The early stages of vision processing identify features in images that are relevant to estimating the structure and properties of objects in a scene. Edges are one such feature. Edges are significant local changes in the image and are important features for analyzing images. Edges typically occur on the boundary between two different regions in an image. Edge detection is frequently the first step in recovering information from images. In the fifth chapter, all the terms above are inspected in detailed fashion.

All the chapters above are considered for only one reason which involves reconstruction of medical images. The sixth chapter, includes all of our efforts from the starting point of the project to the completion. It has mainly two different parts. The first one considers retrieving and preprocessing of medical images. The another part includes reconstruction process and the storing information.

Chapter 2

IMAGE PROCESSING CONTEXT AND ITS INTERESTING AREAS

2.1 Introduction

As the speed, capability, and economic advantages of modern signal processing devices continue to increase, there is simultaneously an increase in efforts aimed at developing sophisticated, real-time automatic systems capable of emulating human abilities. The area of image processing and computer vision are growing rapidly, so that it is not possible to define or describe it in a simple phrase. The section below presents the detailed information which covers the main aspects, application areas, and literature survey on image processing.

2.2 Computer Vision Applications

The classification of image processing applications is more than a study. In this section, it has been shown that some of the selected applications can be recognized as the edge points of this area. They are very important from the point of imaging, because all the applications below has found a place for themselves in human life. In other words, if there exists a theory which can not be implemented so as to make life easy, has no meaning more than being a theory by itself.

Robotics covers that industrial inspection, surface measurement or mapping, manipulator guidance, and vehicle guidance. Inspection using machine vision, in some applications, has been shown to be more reliable than human visual inspection. Inspection of printed circuit boards can be given as an example of this case. Surface measurement or mapping interests in if the parts has been produced within the tolerances. It is fairly common for factory robotics applications to require that the manipulator device identify, move to, grasp, and than move an object in a cluttered

workspace using visual guidance. Part of this process involves the determination of the part orientation such that the part may be in an acceptable manner. The need for autonomous vehicle systems becomes apparent in the need to free human drivers from hazardous environments. The term 'fire and' forget describes weapons delivery systems that contain a built-in capability to adjust their trajectory on the basis of continuously acquired image information.

Image compression and enhancement is another area of interest of image processing. The image compression applications are dedicated to provide transferring and storing of more data with low cost. This also means that much faster communication medium. Image enhancement is another area where results are viewed subjectively. Given imagery degraded by the imaging environment (e.g., atmospheric distortion or insufficient scene illumination) or the transmission medium (e.g., significant electrical noise has now been added to the image data) , it is necessary to characterize this distortion and then attempt to process the imagery so that the visual effect of this distortion is minimized.

Medical applications which we are also interested in and chosen as a subject of our study , are widely used in medical environment. Energy sources that interact with biological mechanism generate image-like entities. This includes X-rays, radio frequency, magnetic, and ultrasonic energy used by physicians for medical diagnosis. The processing of these images in digital form may improve their subjective appearance and consequent utility.

2.3 Literature Survey On Computer Vision And Image Reconstruction

Wang and Rao (Wang and Rao, 1996) have directed their efforts on optical ramp edge detection. They have used expansion matching approach (EXM) which optimizes a novel matching criterion called Discriminative Signal-to-Noise Ratio (DSNR) and has been shown to robustly recognize templates under conditions of noise, severe occlusion, and superposition.

Law (Law, 1996) has used fuzzy reasoning for image filtering, edge detection and edge tracing. He has filtered the images by applying fuzzy reasoning based on local pixel characteristics to control the degree of Gaussian smoothing. Filtered images are then subjected to a simple edge detection algorithm which evaluates the edge fuzzy

membership value for each pixel, based on local image characteristics. Finally, pixels having high edge membership are traced and assembled into structures, again using fuzzy reasoning to guide the tracing process.

The detection of lines in satellite images has drawn a lot of attention. Merlet and Zerubia (Merlet and Zerubia, 1996) have involved the problems of resolution, noise, and image understanding. Then they have used algorithm of Fischler, which achieves robustness, rightness, and rapidity. They have presented a mathematical formalization of the Fishler algorithm, which allows them to extend the cost both to cliques of more than two points, and to neighborhoods of size larger than one (to take into account the curvature). Thus, all the needed information (contrast, gray-level, curvature) is synthesized in a unique cost function defined on the digital original image. This cost is used to detect roads and valleys in satellite images (SPOT).

Zhu and Chirlian (Zhu and Chirlian, 1995) have presented a nonlinear algorithm for critical point detection of 2D digital shapes. Their algorithm eliminates the problems arising from curvature approximation and Gaussian filtering in the existing algorithms. By quantifying the critical level to the modified area confined by three consecutive "pseudocritical points," a simple but very effective algorithm is developed. The comparison of their experimental results with those of many other CPD algorithms shows that the proposed algorithm is superior in that it provides a sequence of figures at every detail level, and each has a smaller integral error than the others with the same number of critical points.

Image segmentation based upon contour extraction usually involves three stages of image operations: feature extraction, edge detection and edge linking. Heijden (Heijden, 1995) has studied to the first stage: a method to design feature extractors used to detect edges from noisy and/or blurred images. The method relies on a model that describes the existence of image discontinuities (e.g. edges) in terms of covariance functions. The feature extractor transforms the input image into a "log-likelihood ratio" image. Applications on real world images are presented showing the capability of the covariance model to build edge and line feature extractors. Finally he has shown that the covariance model can be coupled to a MRF-model of edge configurations so as to arrive at a maximum a posteriori estimate of the edges or lines in the image.

Humans have a well developed ability to detect curvilinear structure in noisy images. Good algorithms for performing this process would be very useful in machine

vision for image segmentation and object recognition. Gigus and Malik (Gigus and Malik, 1991) have developed a simple feedforward and parallel approach to this problem based on the idea of developing filters tuned to local oriented circular arcs. This provides a natural second order generalization of the idea of directional operators popular for edge detection. Curve detection can then be done by methods very similar to those used for edge detection.

Nordstrom (Nordstrom, 1989) present and analyze a global edge detection algorithm based on variational regularization. The algorithm can also be viewed as an anisotropic diffusion method. This puts an isotropic diffusion, as a method in early vision, on more solid grounds; it is just as well-founded as the well-accepted standard regularization techniques. The unification also brings the anisotropic diffusion method an appealing sense of optimality, thereby intuitively explaining its extraordinary performance.

When computing descriptors of image data, the type of information that can be extracted may be strongly dependent on the scales at which the image operators are applied. Lindeberg (Lindeberg, 1996) have presented a systematic methodology for addressing this problem. A mechanism is presented for automatic selection of scale levels when detecting 1D image features, such as edges and ridges. A nonadaconcept CVAP/of a scale-space edge is introduced, defined as a connected set of points in scale-space at which: (i) the gradient magnitude assumes a local maximum in the gradient direction, and (ii) a normalized measure of the strength of the edge response is locally maximal over scales. An important consequence of this definition is that it allows the scale levels to vary along the edge. For a certain way of normalizing these differential descriptors, by expressing them in terms of so-called gamma-normalized derivatives, an immediate consequence of this definition is that the edge detector will adapt its scale levels to the local image structure. Since the scale-space edge is defined from the intersection of two zero-crossing surfaces in scale-space, the edges will by definition form closed curves.

Stone and Isard (Stone and Isard 1995) have proposed a general method for obtaining shape from texture. Their approach is based on adaptive scale filtering which is a general method for deriving shape from texture under perspective projection without recourse to prior segmentation of the image into geometric texture elements (texels), and without thresholding of filtered images. Due to their assumptions, if texels

on a given surface can be identified in an image then the orientation of that surface can be obtained.

Han et al (Han, 1995) have described and compared the techniques of a posteriori image restoration and iterative image feature extraction. Image feature extraction methods known as Graduated Nonconvexity (GNC), Variable Conductance Diffusion (VCD), Anisotropic Diffusion, and Biased Anisotropic Diffusion (BAD), which extract edges from noisy images, are compared with a restoration/feature extraction method known as Mean Field Annealing (MFA). This equivalence shows the relationship between energy minimization methods and spatial analysis methods and between their respective parameters of temperature and scale. As a result of the equivalence, VCD is demonstrated to minimize a cost function, and that cost is specified explicitly.

Liu and Ehrich (Liu and Ehrich, 1995) have concerned the problem of obtaining subpixel estimates of the locations of straight edges in binary digital images using dithering. By adding uniformly distributed independent random noise it is shown that estimation bias may be removed and that the estimation variance is inversely proportional to the length of the line segment.

In most systems the first step in tracking objects is to separate the foreground from the background. This means to detect the regions (apparent shape) of independently moving objects regardless of their speed, direction or texture. Ridder et al (Ridder, 1995) have directed their efforts to solve this problem and presented an algorithm to handle illumination changes by using a Kalman-filter system for each pixel of a frame. Each filter predicts the grey values in the following frame, compares the prediction with the present value and obtains an estimation of the background pixel value. The illumination changes are quickly adapted in the background sequence in contrast to a slow adaption in the regions that include the moving objects.

It is the another interesting area of image processing is data analysis. Chien and Mortensen (Chien and Mortensen, 1996) have proposed Multimission VICAR Planner (MVP) system. It was an AI planning system which uses knowledge about image processing steps and their requirements to construct executable image processing scripts to support high-level science requests made to the Jet Propulsion Laboratory (JPL) Multimission Image Processing Subsystem (MIPS). This article describes a general AI planning approach to automation and application of the approach to a specific area of

image processing for planetary science applications involving radiometric correction, color triplet reconstruction, and mosaicing in which the MVP system significantly reduces the amount of effort required by image processing experts to fill a typical request.

One of the most important part of terrestrial navigation systems is where the roads are located and how they are composed. Barzohar and Cooper (Barzohar and Cooper, 1996) have developed a system to find main roads using aerial images. They have used geometric-stochastic models for road image generation. They have used Gibbs Distributions. The map estimation is handled by partitioning an image into windows, realizing the estimation in each window through the use of dynamic programming, and then, starting with the windows containing high confidence estimates, using dynamic programming again to obtain optimal global estimates of the roads present.

Until the point of start, the conventional methods have found themselves a wide application areas in computer vision. Object recognition is one of them. Iverson and Zucker (Iverson and Zucker, 1995) have added another milestone on this way. They have proposed a language for designing image measurement operators suitable for early vision. They refer to them as logical/linear (L/L) operators, since the operators unify aspects of linear operator theory and Boolean logic. They have proposed that a family of these operators appropriate for measuring the low-order differential structure of image curves is developed. The resulting operators allow for coarse measurement of curvilinear differential structure (orientation and curvature) while successfully segregating edge-and line-like features. By thus reducing the incidence of false-positive responses, these operators are a substantial improvement over (thresholded) linear operators which attempt to resolve the same class of features.

The problem of segmenting an image into separate regions and tracking them over time is one of the most significant problems in vision. Geiger et al (Geiger, 1995) propose a method to further explore the information provided by the user's selected points and apply an optimal method to detect contours which allows a segmentation of the image. The method is based on dynamic programming (DP), and applies to a wide variety of shapes. It is exact and not iterative.

Greisen et al (Greisen, 1988) has designed the Astronomical Image Processing System, AIPS for short. It is the National Radio Astronomy Observatory's contribution to the class of major image processing systems. It was one of the earliest of these

systems and most widely used in astronomy. That success is due in large part to the scientific success of the telescope it was created to support, namely the Very Large Array (VLA) , built and operated by the NRAO.

Chapter 3

IMAGE GREY-LEVEL MODELING AND EARLY PROCESSING FUNDAMENTALS

3.1 Preface

In this chapter models and processing algorithms that rely significantly on image gray-value or intensity variation are considered. The algorithms developed in this chapter are seldom used alone to achieve a system goal (such as object inspection), but rather are often the initial preprocessing of a gray-level image.

3.1.1 Deterministic and Stochastic Transforms

We have started our study with examination of gray-level dependent image models and corresponding processing algorithms. Schalkoff (Schalkoff, 1989) has considered these models and algorithms into four subdivisions which are also examined below.

1. Stochastic (probabilistic) models that have no geometry dependence. An example is the use of image intensity histograms for enhancement purposes (histogram equalization).

2. Deterministic models that rely on local gray-value variations. An example edge enhancement algorithms.

3. Stochastic models that rely on local gray-value dependence. An example is the use of co-occurrence matrices.

4. Deterministic models that rely on global image properties, both geometric and intensity based. An example is the use of a new set of basis function to represent the entire images. This introduces image transforms and is our first objective.

Firstly, we have studied the concept of a *basis function matrix* for the representation of descritized image data. The use of a separable linear transform, implemented as pre-and post multiplication of the image function by transform matrices,

is used to facilitated a change of basis functions. Examples of this type of transform are the Discrete Fourier Transform and the Hadamard Transform (Ryser, 1963), (Pratt, 1978). Implicit in the use of such transformation is the assumption that the underlying image possesses some characteristics that may be related to the transform image basis functions. In other words, the image intensity array has some properties that are more clearly discernible or extractable in the transformed representation. If the image has a significant contribution due only to a few basis functions, the resulting transform matrix will be sparse, which leads to a strategy for compression or encoding of image data. Furthermore, the transform approach may be of use in the development of algorithms for image classification and feature extraction.

The 1-D Fourier Transform is reviewed first and the discrete version of this 1-D transform is represented in matrix vector notation. Properties all the complex matrix used to achieve this transform explored. This make sense, since the 2-D Discrete Fourier Transform (DFT) may be expressed using this matrix and calculated using the 1-D DFT.

The 2-D DFT then follows. Its shown that the structure and visualization of this transform is related to a larger class of linear separable transformations. Studying transforms as basis function changes allows inside in to the transform properties and utility. Following development of the 2-D DFT, a broader class of transforms is considered in several representative transforms and their basis functions and properties are studied.

3.1.2 Global Versus Local Image Modeling and Processing Approaches

The fundamental concept of global versus local model applicability enters into the selection of image models and subsequent design of processing algorithms. Therefore, it is important to understand the representational imitations of a specific model. For example, many global approaches employing global information representations (e.g., moments, Fourier transforms) must assume that there is a single object that is completely visible in the image.

Applying a moment-based modeling approach, for example, to an image consisting of a group of overlapping parts most likely produces a set of features (i.e., transform coefficients) that a relatively little in common with in individual part features. Clearly there is a regionally varying or local information about needs to be modeled and

extracted first, followed by a globally based model and corresponding algorithm to interpret the determine the significance of the local model information. A more "intelligent" algorithm would then be used to infer a more global assessment of the image(s) content; for example , a set of object production rules may be used to determine if non objects are parts thereof are present.

3.2 Basic Linear System, Discrete Transform Concepts and Image Basis Functions

Let us postpone temporarily our concern for the quantitative and qualitative effects of a sampling a continuous 2-D function $f(\underline{x})$, and assume that a matrix of image intensities, $f(x_1, x_2)$ is available. We are interested in developing a methodology that allows straightforward analysis and visualization of the plethora of avaiable2-D transforms. Our effort is concentrated on linear separable transforms, which are written in the form

$$[F] = U^T [f] V \tag{3-1}$$

where $[f]$ is an N x N image function matrix, and U and V are matrices that effect the appropriate transform (at this point we assume they are real matrices). An image function matrix may be written in terms of "basis function matrices" as

$$[f] = \begin{bmatrix} f_{11} & f_{12} & . & . & f_{1N} \\ f_{21} & & & & \\ . & & & & \\ . & & & & \\ f_{N1} & f_{N2} & . & . & f_{NN} \end{bmatrix}$$

$$= f_{11} \begin{bmatrix} 1 & 0 & . & . & 0 \\ 0 & & & & . \\ . & & & & . \\ . & & & & . \\ 0 & 0 & . & . & 0 \end{bmatrix} \tag{3-2}$$

$$+f_{12}\begin{bmatrix} 0 & 1 & . & . & 0 \\ 0 & & & & . \\ . & & & & . \\ . & & & & . \\ 0 & 0 & . & . & 0 \end{bmatrix}$$

+.....+

$$f_{NN}\begin{bmatrix} 0 & 0 & . & . & 0 \\ 0 & & & & . \\ . & & & & . \\ . & & & & . \\ 0 & 0 & . & . & 1 \end{bmatrix}$$

or, by defining the set of N^2 NxN matrices on the right side of the Eq. 3-2 as E_{mn},

$$[f] = \sum_{m=1}^{N} \sum_{n=1}^{N} f_{mn} E_{mn} \tag{3-3}$$

Hereafter, these matrices are referred to as basis *function matrices*.

Referring to Eq. 3-1, observe that the associatively property of matrix multiplication allows us to write the k,l^{th} element of F as

$$F_{kl} = \sum_{m=1}^{N} \sum_{n=1}^{N} u_{mk} f_{mn} v_{nl} \tag{3-4}$$

$$= \sum_{m=1}^{N} \sum_{n=1}^{N} f_{mn} (u_{mk} v_{nl})$$

Thus, Eq. 3-4 indicated that F_{kl} is formed by summing the projection of [f] onto a matrix formed as the outer product of the k^{th} column of U and the l^{th} column of V. Writing U and V as

$$U = \left[\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_N\right]$$

$$V = \left[\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_N\right] \tag{3-5}$$

we form the $(k,l)^{th}$ element of [F], F_{kl}, in Eq. 2-1 by writing Eq. 3-4 as

$$F_{kl} = \sum_{\substack{\text{all elements} \\ \text{in product}}} [f] \otimes \mathrm{u}_k \, v_l^T \tag{3-6}$$

where $\otimes$ operator denotes a product of two matrices formed by a point-by-point multiplication of corresponding elements. An alternative view of Eq. 3-6 is that F_{kl} is formed by correlating [*f*] with $\underline{u}_k, \underline{v}_l^T$. Denoting the outer product operation of $\underline{u}_k$ and $\underline{v}_k$ in Eq. 2-6 as $>\underline{u}_k . \underline{v}_l<$, we see that the result is a matrix whose rank is exactly1. More important, the entire separable linear transform in Eq. 3-1 is achieved by projecting the image function matrix, [*f*], onto the N^2 basis functions of the transform as indicated by Eq. 3-6. Thus, Eq. 3-2 or Eq. 3-3 may be viewed as a "transform" whose i,j^{th} element is formed by projecting [*f*] onto the basis function E_{ij}, where

$$E_{ij} => \underline{e}_1, \underline{e}_2 < \tag{3-7}$$

and $\underline{e}_i$ is a vector whose elements are zero except for the i^{th} element, which equals 1. The reader is encouraged to show that this results in "transform" matrices

$$U^T = I$$

and

$$V = I$$

where *I* is an NxN identity matrix.

The inverse of Eq. 3-1 is now considered. For practical reasons as well as ease of presentation, it is assumed that *U* and *V* are orthogonal matrices, (i.e., they are composed of orthonormal column vectors); therefore [*f*] may be recovered by computing the inverse transform as

$$[f] = [U][F][V]^T \tag{3-8}$$

Decomposed the transformed function [F], as in Eq. 3-2, yields

$$[F] = \begin{bmatrix} F_{11} & 0 & . & . & 0 \\ 0 & & & & . \\ . & & & & . \\ . & & & & . \\ 0 & 0 & . & . & 0 \end{bmatrix} + \begin{bmatrix} 0 & F_{12} & 0 & . & 0 \\ 0 & & & & . \\ . & & & & . \\ . & & & & . \\ 0 & . & . & . & 0 \end{bmatrix} \tag{3-9}$$

$$+\ldots\ldots+$$

$$\begin{bmatrix} 0 & 0 & . & . & 0 \\ 0 & & & & . \\ . & & & & . \\ . & & & & . \\ 0 & . & . & . & F_{NN} \end{bmatrix}$$

Equation 3-8 may be expanded as

$$[f] = [\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_N][F][\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_N]^T \qquad (3\text{-}10)$$

It is left for the reader to show that Eq. 3-10 and the decomposition of $[F]$ in Eq.3-9 yield

$$[f] = \sum_{k=1}^{N} \sum_{l=1}^{N} F_{kl}\, \underline{u}_k\, \underline{v}_l^T \qquad (3\text{-}11)$$

Thus whereas F_{kl} represents the projection of the image function matrix onto the transform basis function $> \underline{u}_k, \underline{v}_l <$, Eq. 3-11 indicates that the image function is recovered by summing the transform basis functions weighted by the corresponding projection. This is shown in Figure 3-1. This result, although intuitively appealing, also suggests applications, since:

1. In the process of identifying certain image features, it may be appropriate to choose these features as basis functions, compute the transform, and thereby determine the content of each of these features in the image.
2. Images (or ensembles of images) that may be represented or approximated by the weighted sum of a small number of the basis functions of a particular transform may be efficiently stored and/or transmitted in transformed form

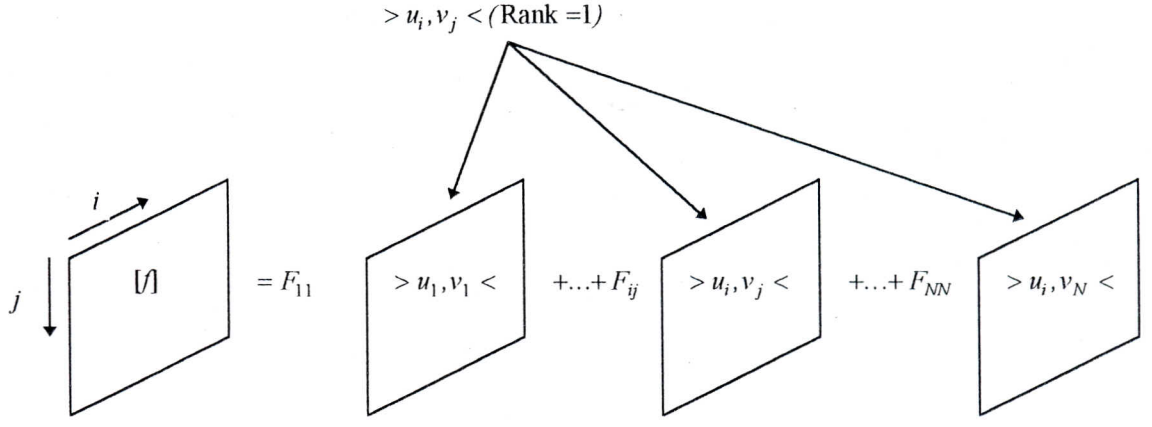


Figure 3-1 Expansion of [f] onto set of rank-1 basis function matrices.

3.3 Discrete Transforms

We define a transform of a 1-D function whose argument is an integer ranging from 0 to N-1. Such functions may be N samples of a continuous function, but it is not necessary. The Discrete Fourier Transform (DFT) of this 1-D function is defined as

$$F(u) = \sum_{k=n}^{N-1} f(k) \exp\left[-j \frac{2\pi}{N} uk \right] \qquad 0 \le u \le N-1 \qquad (3\text{-}12)$$

The universe of this transform, the Discrete Inverse Fourier Transform (DIFT) is given by

$$f(k) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) \exp\left[j \frac{2\pi}{N} uk \right] \qquad (3\text{-}13)$$

Matrix-vector representation facilitate analysis of this transform. Defining a complex variable z as

$$z = \exp\left[-j \frac{2\pi}{N} \right] \qquad (3\text{-}14)$$

and an NxN complex matrix Z as

$$Z = \begin{bmatrix} 1 & 1 & 1 & . & . & 1 \\ 1 & Z & Z^2 & & & \\ 1 & Z^2 & & & & \\ . & . & & & & \\ . & . & & & & Z^{(N-1)^2} \\ 1 & Z^{N-1} & & & & \end{bmatrix} \qquad (3\text{-}15)$$

(i.e., Z is a matrix whose u,k^{th} element $Z_{uk} = z^{uk}$), we index row and column elements in Z from 0 to N-1 to write the 1-D DFT as

$$\underline{F} = Z \underline{f} \qquad (3\text{-}16)$$

Where F is an Nx1 column vector of transform values i.e.,

$$\underline{F} = \begin{bmatrix} F(0) \\ F(1) \\ . \\ . \\ F(N-1) \end{bmatrix} \tag{3-17}$$

and $\underline{f}$ is a Nx1 column vector of function values i.e.,

$$\underline{f} = \begin{bmatrix} f(0) \\ f(1) \\ . \\ . \\ f(N-1) \end{bmatrix} \tag{3-18}$$

The relationship in Eq. 3-16 is important because it represents a *change* of *basis vectors* for our representation of $\underline{f}$, achieved in this case as a change of coordinates. The original basis function set was the columns of the NxN identity matrix. I, whereas in the transform the new basis vectors for the representation are the columns of Z. It is also important to note that N is not only even, but also a power of 2. This is useful in "fast" transform implementations mentioned subsequently.

A complex matrix, F, with complex conjugate matrix F^* may be said to be

1. Hermitian, if

$$\left(F^*\right)^T = F \tag{3-19}$$

and

2. Unitary, if

$$F^{-1} = \left(F^*\right)^T \tag{3-20}$$

In dealing with real matrices, Case 2 corresponds to the case of an orthogonal matrix. Finally, recall that if $\underline{x}$ and $\underline{y}$ are complex vectors, their inner product,

$$< \underline{x}, \underline{y} > = \left[\underline{x}^*\right]^T \underline{y} \tag{3-21}$$

and if $< \underline{x}, \underline{y} > = 0$, the vectors are said to be orthogonal.

Referring to the definition of the complex variable Z, we note

$$
\begin{aligned}
Z^N &= 1 \\
Z^{N/2} &= -1 \\
Z^{N/4} &= -j \\
Z^{3N/4} &= j
\end{aligned}
\tag{3-22}
$$

With this, notice the Z matrix has the following properties:

1. $Z = Z^T$
2. the columns (or rows) of Z are orthogonal
3. the inner product of a column (or row) with itself = N.

This yields the important result

$$
\left[Z^*\right]^T Z = NI \tag{3-23}
$$

or equivalently,

$$
Z^{-1} = \frac{1}{N}\left[Z^*\right]^T \tag{3-24}
$$

Thus, Z is an "almost unitary" matrix.

On the basis of the above, we may form the DIFT in vector matrix notation as

$$
\begin{aligned}
\underline{f} &= Z^{-1}\,\underline{F} \\
&= \frac{1}{N}\left[Z^*\right]^T \underline{F}
\end{aligned}
\tag{3-25}
$$

which simplified to

$$
\underline{f} = \frac{1}{N} Z^* \underline{F} \tag{3-26}
$$

The significance of this result is that the original functions may be recovered from the transformed version (*without the need for matrix inversion*). This is a property which is especially useful in computation of 2-D transforms, where the transform matrices may be quite large and therefore the computation of general inverses is impractical.

Elementary 2-D functions. The utility of mathematically characterized as a 2-D delta function is identical to its 1-D counterpart, namely, it yields a technique to characterize 2-D systems via impulse response or point-spread functions. It also has significant utility in our modeling of the reconstruction process.

Defining a 2-D function rect(x,y), as follows:

$$\text{rect}(x,y) = \begin{cases} 1 & |x| \le \frac{1}{2} \quad \text{and} \quad |y| \le \frac{1}{2} \\ 0 & \text{elsewhere} \end{cases} \tag{3-27}$$

allows us to define the 2-D delta function through a limiting process. Letting

$$\delta_n(x,y) = n^2 \,\text{rect}(nx,ny) \qquad n>0 \tag{3-28}$$

we observe that:

1. δ_n is non-zero only inside a $1/n$x$1/n$ square in the image plane, and
2. $\int\int_{-\infty}^{\infty} \delta_n(x,y)\,dxdy = 1.0 \qquad \forall\, n$ (3-29)

Taking the limit as an $n \to \infty$ yields the (2-D) Dirac delta function:

$$\delta(x,y) = \begin{cases} \infty & at \quad (0,0) \\ 0 & elsewhere \end{cases} \tag{3-30}$$

Note that $\delta(x,y)$ also retains the 2nd property cited above, namely

$$\int_{-\infty}^{\infty} \delta(x,y)\,dxdy = 1.0 \tag{3-31}$$

Several additional properties of $\delta(x,y)$ are note worthy:

1. The *sifting* property, which may be derived from:

$$\lim_{n\to\infty}\left\{\int\int_{-\infty}^{\infty} g(x,y)\delta_n(x,y)\,dxdy\right\} = g(0,0) \tag{3-32}$$

Note the integrated quantity above represents the average value of $g(x,y)$ over a $1/n$x$1/n$ window centered at (0,0).

The above expression may be generalized to:

$$\int\int_{-\infty}^{\infty} g(x,y)\delta_n(x-\alpha, y-\beta)\,dxdy = g(\alpha,\beta) \tag{3-33}$$

2. An extremely useful property (which actually comprise a transform pair) is

$$\int\int_{-\infty}^{\infty} exp\left[j2\pi(ux+vy)\,dxdy\right]dudv = \delta(x,y) \tag{3-34a}$$

and

$$\int\int_{-\infty}^{\infty} \delta(x,y)\,exp\left[j2\pi(ux+vy)\,dxdy\right]dudv = 1.0 \tag{3-34b}$$

This relationship may be shown by either invoking the sifting property, or using the rect function, computing the transformation indicated by Eq. 3-34b, and applying the limiting process.

One of the most frequently encountered and useful transform pairs is based upon the rect function. If

$$f(x,y) = \text{rect}(x,y) \tag{3-35a}$$

then it is straightforward to show that

$$\begin{aligned} F_{\text{rect}}(u,v) &= \int_{-\infty}^{\infty}\int \text{rect}(x,y)\exp\left[-j2\pi(ux+uy)\right]dxdy \\ &= \frac{\text{Sin}\pi u}{\pi u}\cdot\frac{\text{Sin}\pi v}{\pi v} \\ &\overset{\Delta}{=} sin\,c(u,v) \end{aligned} \tag{3-35b}$$

Linear Operations and Convolution. The linearity concept for 2-D functions and their operations may be defined by considering two image functions $f_1(\underline{x})$ and $f_2(\underline{x})$. And operator, σ, is said to be linear if

$$\sigma\left\{\alpha f_1(\underline{x}) + \beta f_2(\underline{x})\right\} = \alpha\sigma\left\{f_1(\underline{x})\right\} + \beta\sigma\left\{f_2(\underline{x})\right\} \tag{3-36}$$

where α and β arbitrary scalars.

Two additional remark are:

1. The concept of linearity should be distinguished from other independent concepts such as time or space-varying; and
2. Alternately, a linear operator is one which superposition (indicated above) holds.

By representing an image function using the sifting property of the δ function, we arrive it:

$$f_1(x,y) = \int_{-\infty}^{\infty}\int f_1(\xi,\eta)\delta(x-\xi,y-\eta)\,d\xi\,d\eta \tag{3-37}$$

Suppose an output image function, $f_2(\underline{x})$ is obtained by a linear operator on $f_1(\underline{x})$, i.e.,

$$f_2(x,y) = \sigma\left\{f_1(x,y)\right\} \tag{3-38}$$

The concepts of a linear operator and the above decomposition of f_1 allows us to expand (3-38) as:

$$f_1(x,y) = \int_{-\infty}^{\infty}\int f_1(\xi,\eta)\sigma(x-\xi,y-\eta)\,d\xi\,d\eta \qquad (3\text{-}39)$$

where the second term in the above integral is defined as the *impulse response function* or *point-spread function* of the linear system, i.e.,

$$\sigma\{\delta(x-\xi,y-\eta)\} \triangleq h(x,y,\xi,\eta) \qquad (3\text{-}40)$$

is the response of the system, at (x,y), to a 2-D point source applied at (ξ,η). If the system is *space (or shift) invariant*, then

$$h(x,y,\xi,\eta) = h(x-\xi,y-\eta) \qquad (3\text{-}41)$$

that is, the response is only a function of the (vector) *difference* between the point of application and response, not the absolute locations.

Applying the result of Eq. 3-41 to Eq. 3-39 yields

$$f_2(x,y) = \int_{-\infty}^{\infty}\int f_1(\xi,\eta)h(x-\xi,y-\eta)\,d\xi\,d\eta \qquad (3\text{-}42)$$

which is referred to as the *convolution integral* for a linear, space invariant 2-D system.

3.4 Convolution, Correlation and Related Operations

The formulation of Eq. 3-42 spawns two associated and similar operations - namely, *correlation and matched filtering*.

2-D and higher dimensional convolution is merely an extension of Eq. 3-42 is formulated with the input image function, f_1, and the system impulse response function, h, expressed in terms of the (ε,η) coordinate system. Without loss of generality, assume that coordinate system origin is chosen such that both functions are centered about (0,0). Note also that typically (or partially) the system impulse response, h, is only nonzero over a finite region in the (ε,η) plane, denoted R, and therefore the infinite limits in Eq. 3-42 are only of theoretical significance. As a practical matter, the amount of computation required to form the output image, f_2, is related to the nonzero extent of h, which is typically small relative to the extent of the input image, f_1. A convenient

interpretation of the convolution operation of Eq. 3-42 leads to a graphical interpretations as follows:

1. Rotate $h(\varepsilon,\eta)$ about the origin by 180° in the (ε,η) plane. This is a reflection of h about both the ε and η axes. This forms $h(-\varepsilon,-\eta)$.
2. Translated the rotated or reflected h by an amount (x, y) with respect to $f_1(\varepsilon,\eta)$ in the (ε,η) plane. This forms $h(x-\varepsilon,y-\eta)$. For example, the value of $h(0,0)$ is now at location (x,y). This makes intuitive sense, since one contribution to f_2 at (x,y) should be due to the response to an impulse at this point in the input image. This is determined by $h(0,0)$.
3. Integrate $f_1(\varepsilon,\eta)\,h(x-\varepsilon,y-\eta)$ over the region in which both functions are nonzero.

This is shown graphically in Figure 3-2a.

Through a change of variables, the convolution expression of Eq. 3-42 may be reformulated as

$$f_2(x,y)=\int_{-x}^{x}\int f_1(x-\varepsilon,y-\eta)h(\varepsilon,\eta)d\varepsilon d\eta \qquad (3\text{-}42b)$$

$$f_2(x,y)=\int_{-x}^{x}\int f_1(\varepsilon,\eta)h(\varepsilon+x,\eta+y)d\varepsilon d\eta \qquad (3\text{-}42c)$$

Again through a change of variables, the correlation of Eq. 3-42c may be written as

$$f_2(x,y)=\int_{-x}^{x}\int f_1(\varepsilon-x,\eta-y)h(\varepsilon,\eta)d\varepsilon d\eta \qquad (3\text{-}42d)$$

Note that *the operations of convolution and correlation differ only in the* 180^{o} *rotation (or reflection) of* $h(\varepsilon,\eta)$. If $h(\varepsilon,\eta)$ is symmetric with respect to the ε and η axes, these operations yield the same result. Due to symmetry $h(\varepsilon,\eta)$, many formulations used for enhancement and restoration often refer to these operations as either correlation and convolution. A graphical view of correlation is shown in figure 3-2b.

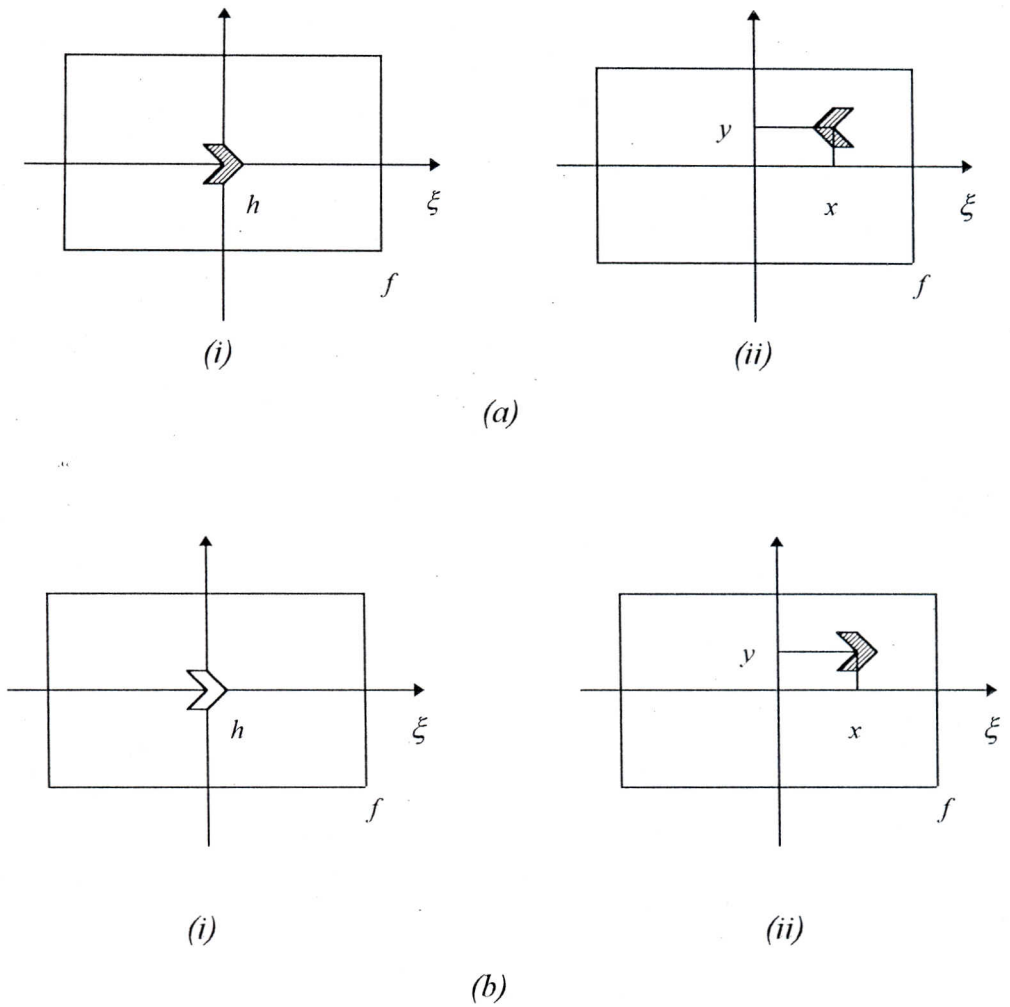


Figure 3-2

(a) Graphical view of convolution from (3-42)

(i) Function f and h (nonzero regions of support shown)

(ii) The convolution result (integral of product in shaded region)

(b) Graphical view of correlation from (3-42c)

(i) Image of f and template h

(ii) Correlation of f and h at location x

Finally, we remark that the operations of convolution in Eq. 3-42 and correlation in Eq. 3-42c may be viewed alternately as the equivalent processes of *matched filtering* and *template matching*, respectively. Both operations are used to find regions in the image function. $f_1(\varepsilon,\eta)$, which closely match a function termed the *template*. In the case of matched filtering, $h(\varepsilon,\eta)$ is chosen to be 180^o -rotated version of the template, whereas in correlation we use the value of the template for $h(\varepsilon,\eta)$.

3.5 Discrete Versions of Convolution and Correlation

The discrete version of convolution and correlation follow from the analog formulations. The discrete convolution of functions f_1 and h, to produce f_2, may be written as

$$f_2(m,n) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} f_1(i,j)\,h(m-i,n-j) \qquad \text{(3-42e)}$$

where the limits on the summation are due to the assumed rectangular region of support (or an MxN "window") of h. In contrast with Eq. 3-42e, "periodic" convolution of the functions f and h is defined as

$$f_2(m,n) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} f_1(i,j)\,h([m-i],[n-j]) \qquad \text{(3-42f)}$$

where h is assumed periodic with

$$[m-i] = (m-i) \quad mod \quad M$$

and

$$[n-j] = (n-j) \quad mod \quad N$$

Note, without proof, that the second definition of discrete convolution is necessary to yield the multiplicative property of the corresponding discrete signal spectra in the frequency domain (i.e., the DFT of f_1 and h). Recall that due to sampling, f_1 is also assumed periodic. This assumption poses a small dilemma, since neither function, practically speaking, is periodic. To overcome this dilemma, and therefore be able to use the frequency domain analysis tools, we assume that f_1 and h are defined over larger sampling lattices whose corresponding extends are much greater than the regions of support of f_1 and h. *This is accomplished in practice by forming the larger arrays by padding* f *and* h *in two dimensions with zeros to achieve the larger extent.*

Similarly, the discrete correlation of f_1 and h may be written by analogy with Eq.3-42c as

$$f_2(m,n) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} f_1(i,j)\,h(m+i,n+j) \qquad \text{(3-42g)}$$

The graphical interpretation for these operations using discrete functions are analogous to their continuous counterparts.

Chapter 4

IMAGE ENHANCEMENT, RESTORATION, AND CONVERSION

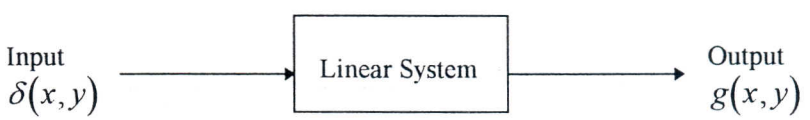
4.1 Introduction

A number of low level algorithmic approaches for the enhancement or restoration of images are developed, the majority of which are related to image "edge" information. *It is difficult to underestimate the utility of reliable edge extraction*, particularly as a precursor to higher-order image processing operations. In addition, several additional model-based approaches are used for image conversion. This includes halftones renditions of images and the reconstruction of image data from projections.

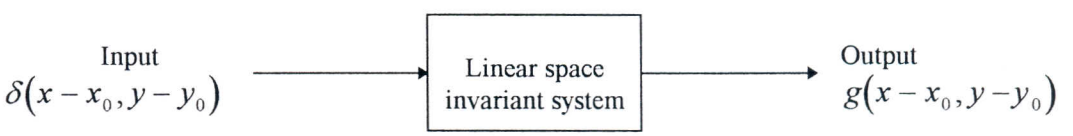
4.2 Operators and Models for Enhancement and Restoration

4.2.1 Linear Systems

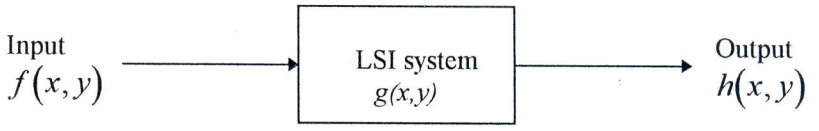
Many image processing operations can be modeled as a *linear system*:



For a linear system, when the input to the system is an impulse $\delta(x,y)$ centered at the origin, the output $g(x,y)$ is the system's *impulse response*. Furthermore a system whose response remains the same irrespective of the position of the input pulse is called a *space invariant* system:



A linear space invariant (LSI) system can be completely described by its impulse response $g(x,y)$ as follows:



where *f(x,y)* and *h(x,y)* are the input and output images, respectively. The above system must satisfy the following relationship:

$$a.f_1(x,y)+b.f_2(x,y) \Rightarrow a.h_1(x,y)+b.h_2(x,y) \qquad (4\text{-}1)$$

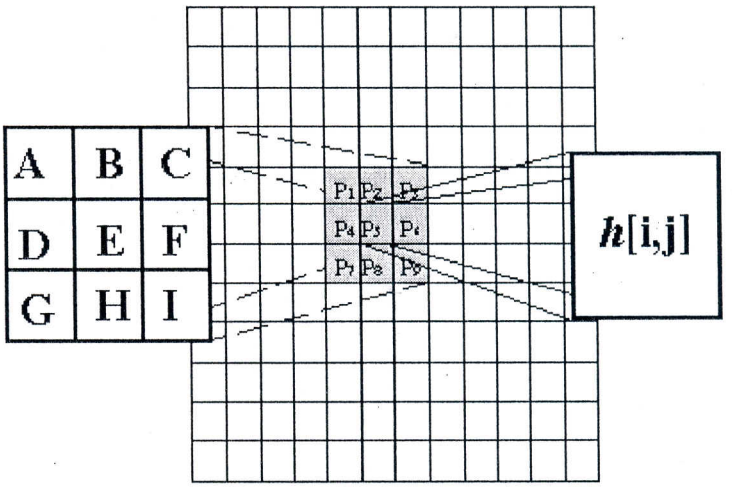
where $f_1(x,y)$ and $f_2(x,y)$ are the input images, and a and b are constant scaling factors.

For such a system, the output *h(x,y)* is the *convolution* of *f(x,y)* with the impulse response *g(x,y)* and is defined as:

$$\begin{aligned} h(x,y) &= f(x,y)*g(x,y) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x',y')g(x-x',y-y')dx'dy' \end{aligned} \qquad (4\text{-}2)$$

For discrete functions, this becomes:

$$\begin{aligned} h[i,j] &= f[i,j]*g[i,j] \\ &= \sum_{k=1}^{n}\sum_{l=1}^{m} f[k,l]g[i-k,j-l] \end{aligned} \qquad (4\text{-}3)$$



$$h[i,j] = Ap_1 + Bp_2 + Cp_3 + Dp_4 + Ep_5 + Fp_6 + Gp_7 + Hp_8 + Ip_9$$

Figure 4-1 An example of a 3 x 3 convolution mask. The origin of the convolution mask corresponds to location E and the weights A,B,...,I are the values of g[-k,-l], k, l=-1, 0, +1.

If f and h are images, convolution becomes the computation of weight d sums of the image pixels. The impulse response, $g[i,j]$, is referred to as a *convolution mask*. For each pixel $[i,j]$ in the image, the $h[i,j]$ is calculated by translating the convolution mask to pixel $[i,j]$ in the image, and then taking the weighted sum of the pixels in the neighborhood about $[i,j]$ where the individual weights are the corresponding values in the convolution mask. This mask illustrated in Figure 3-1 using a 3 x 3 mask.

Convolution is a linear operation, since

$$g[i,j]*\{a_1h_1[i,j]+a_2h_2[i,j]\} = a_1\{g[i,j]*h_1[i,j]\}+a_2\{g[i,j]*h_2[i,j]\} \qquad (4\text{-}4)$$

for any constants a_1 and a_2. In other words, the convolution of a sum is the sum of the convolutions, and the convolution of a scaled image is the scaled convolution. Convolution is a spatial invariant operation, since the same filter weights are used throughout the image. However, a spatially varying filter requires different filter weights in different parts of the image.

4.2.2 Discrete Linear Operators and Spatial Smoothing

The analysis must be started with the formulation and analysis of several simple operators. Local linear operators are considered first. These operators form the output pixel intensity at *(x,y)* from a weighted summation of input pixel intensities in the *neighborhood* of *(x,y)*. This is accomplished via a window kernel which is convoluted with the image intensities in the local region. Figure 4-1 illustrates this operation. The corresponding frequency domain ramification of this type of gray-level operator are significant.

Spatial smoothing filters are typically used for noise removal and the reduction of effects due to undersampling. The simplest example of a smoothing filter is one that employs spatial neighborhood averaging and may be formulated as a linear operation on the input image of the form $g(x,y) = 0[f(x,y)]$, which may be quantified mathematically as

$$g(x,y) = \frac{1}{M}\sum_{S} f(m,n) \tag{4-7}$$

where S is an M-pixel neighborhood of points surrounding (and perhaps including) the point (x,y). Often, S is a rectangular neighborhood of (x,y) -for example, an n x n square. The smoothing operator of Eq. 4-7 for the case of n x n "window" is formulated

$$g(x,y) = \sum_{i=-n/2}^{n/2}\sum_{j=-n/2}^{n/2} h_{sm}(i,j) f(x+i, y+j) \tag{4-8}$$

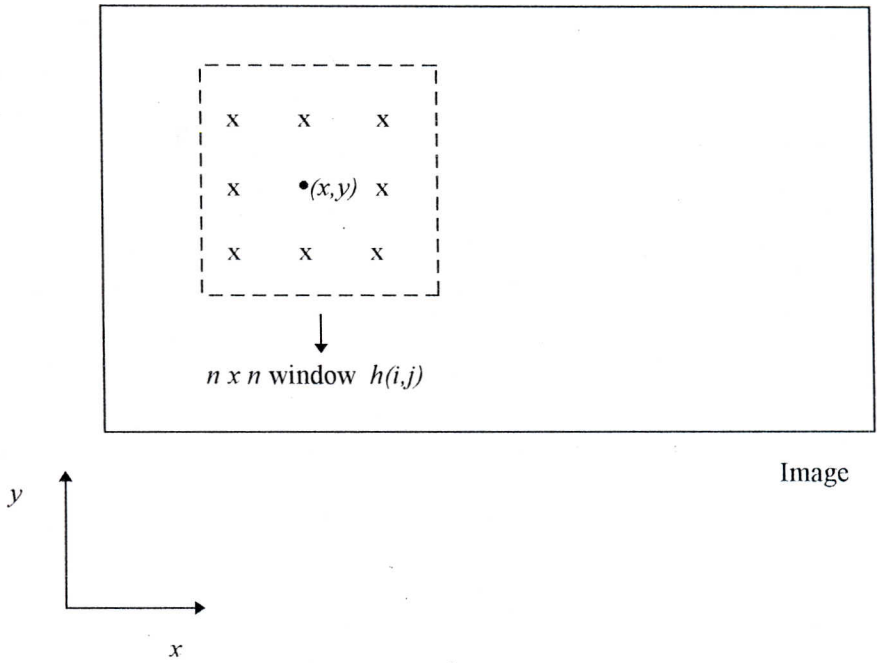


Figure 4-2 Filtering as a "window" operator

where $h_{sm}(i,j)$ is the smoothing function, whose nonzero values in region S are given by

$$h_{sm}(i,j) = \frac{1}{n^2} \tag{4-9}$$

Often we may use a change of variables to rewrite Eq. 4-8 in a more useful form. Note that Eq. 4-8 is in convolution (or correlation) form.

In the case of $n = 3$, the output image function is then the correlation of the 3 x 3 window function

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

with the image function $f(x,y)$ for all (x,y) locations (neglecting "edge" effects in the image plane).

Note that n is often chosen to be odd in order for the window center to be on the sampling grid. It is often convenient to shift the origin of the window using

$$\begin{aligned} k &= i + n/2 \\ l &= i + n/2 \end{aligned} \tag{4-10}$$

Fourier transform of this window function is

$$H_{sm}(u,v) = \frac{1}{N}\sum_{k=0}^{n-1}\sum_{l=0}^{n-1} h_{sm}(k,l)\exp\left[-j2\pi(uk+vl)/N\right] \tag{4-11}$$

where the scaling term $\left(1/N^2\right)$ has been distributed between the forward and inverse transforms. Using Eq. 4-9, Eq. 4-10 reduces to

$$H_{sm}(u,v) = \frac{1}{N}\sum_{k=0}^{n-1}\sum_{l=0}^{n-1}\left(1/n^2\right)h_{sm}(k,l)\exp\left[-j2\pi uk/N\right]\exp\left[-j2\pi vl/N\right] \tag{4-12}$$

Equation 4-12 represents a separable transform. Hereafter, we ignore the (1/N) scaling term and illustrate the properties of this transform by computing the first term; that is,

$$H_1(u) = 1/n\sum_{k=0}^{n-1}\exp\left[-j2\pi uk/N\right] \tag{4-13}$$

Defining the quantity

$$z = \exp\left[j2\pi u/N\right] \tag{4-14}$$

Eq. 4-13 may be rewritten (scaled by 1/n) as

$$H_1(u) = 1/n\sum_{k=0}^{n-1} z^{-k} \tag{4-15}$$

This type of summation occurs frequently in the derivation of Z-transforms, and may be rewritten as

$$H_1(u) = \sum_{k=0}^{n-1} z^{-k} - \sum_{k=n}^{\infty} z^{-k} \tag{4-16}$$

which reduces to

$$H_1(u) = 1/\left(1-z^{-1}\right) - z^{-n}/\left(1-z^{-1}\right) \tag{4-17}$$

Equation 4-11 may be viewed as the difference of the z-transforms of two unit step functions, one occurring at $k = 0$ and the other (a negative step) at $k = n$.

Recalling the definition of z in Eq. 3-14, Eq. 3-17 becomes

$$H_1(u) = \frac{1-z^{-n}}{1-z^{-1}} = z^{-(n-1)^2}\,\frac{z^{n/2}-z^{-n/2}}{z^{1/2}-z^{-1/2}} \tag{4-18}$$

or

$$H_1(u) = \exp\left[-j\pi u(n-1)/N\right]\sin\left(\pi un/N\right)/\sin\left(\pi u/N\right) \tag{4-19}$$

for $n \geq 1$. Note the effect of the shift due to Eq. 4-10 is a phase term in 4-19.

A similar derivation may be made for $H_2(v)$. Notice from Eq. 4-19 that this separable frequency response function has low-pass characteristics, as shown by the magnitude plot in Figure 4-3. This figure shows the effect of varying the window size from 3 x 3 to 9 x 9. Note the corresponding narrowing of the filter frequency response, as expected. This plot is analogous to the (phase shifted) *sinc* function that would have been obtained in the continuous case.

An example of an image processed with this smoothing filter for the case $n = 3$ is shown Figure 4-4. Note the somewhat blurred appearance of the output image. This blurring is an undesirable side effect of the filter.

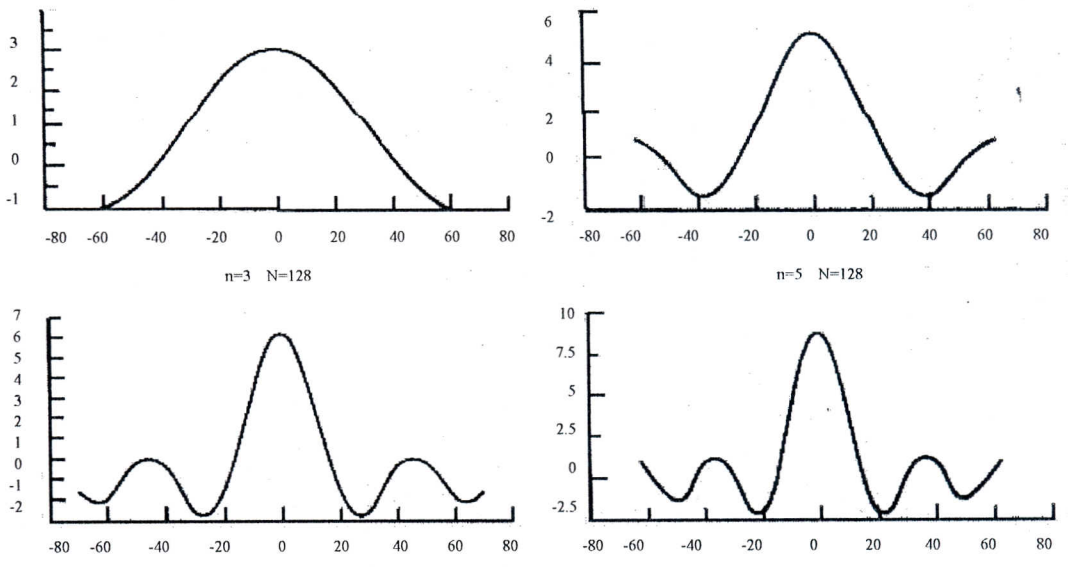
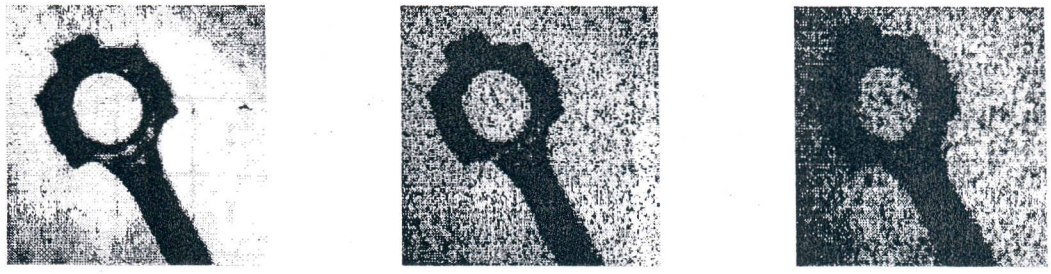


Figure 4-3 Frequency domain effects of smoothing plot of $\sin[\pi un/N]/\sin[\pi u/N]$



(a) (b) (c)

Figure 4-4 Smoothed images

(a) Original

(b) Corrupted image

(c) Smoothed using 3 x 3 window (note loss of edge sharpness)

4.2.3 Nonlinear Operators

Properties of 2-D linear class of filters are often inferior to a wider class of filters which, although defying detailed theoretical analysis, often yield superior results for a given image degradation model or application. For example, the smoothing operator reduces image noise but blurs edges. This is an undesirable side effect and spawns a search for alternative smoothing approaches. Similarly, the sharpening operator sharpens edge information but also introduces sensitivity to (high frequency) noise and "ringing."

4.2.3.1 Rank and Median Filters

The main problem with local averaging operations is that they tend to blur sharp discontinuities in intensity values in an image. An alternative approach is to replace each pixel value with the median of the gray values in the local neighborhood. Filters using this technique are called *median filter.*

. Median filters are a subset of the class of *rank* filters where the output image intensity at spatial location $\underline{x} = (x, y)^T$ is chosen on the basis of the relative rank or intensity of pixels in the neighborhood of $\underline{x}$. Given a set of N pixel intensities obtained over a local image region. S, denoted simply as f_i, $i = 1,2,\ldots,N$, an ordering of these values in increasing values, i.e.,

$$R(\underline{x}) = \{f_1, f_2, \ldots, f_N\}$$

where

$$f_i \leq f_{i+1} \tag{4-20}$$

may be used to derive the rank filter operation. The output image intensity, $g(\underline{x})$ is

$$g(\underline{x}) = Rank_i \qquad R(\underline{x}) \tag{4-21}$$

where the $Rank_i$ is the intensity of the output intensity at position ,or rank, j in $R(\underline{x})$. For example, choosing $j = 1$ yields the *min* filter, where the first element of $R(\underline{x})$ is chosen; that is,

$$\begin{aligned} g(\underline{x}) &= \min \quad Rank_j \\ &= \min\{f(\underline{x})\} | \underline{x} \in S \ \} \end{aligned} \tag{4-22}$$

where S is the chosen neighborhood of $\underline{x}$. Alternately, the *max* filter is derived from choosing $j = N$ and yields

$$\begin{aligned} g(\underline{x}) &= \min \quad Rank_j \\ &= \min\{f(\underline{x})\} | \underline{x} \in S \ \} \end{aligned} \tag{4-23}$$

The most popular rank filter is derived for the case of N odd, and is based on choosing the filter output to be f_m where m is the median intensity; that is, for N intensity samples the position

$$m = (N+1)/2 \tag{4-24}$$

is the median. Thus, f_m is the pixel intensity in the ordered sample set that is greater than $(N-1)/2$ of the samples. We denote the median operator on a sequence of samples as

$$\text{med } (R(\underline{x})) = Rank_{\frac{N-1}{2}}(R(\underline{x})) \tag{4-25}$$

For example, suppose an image with assumed smooth gray-value variations were corrupted by an impulse or "spike"-like noise. Low-pass filtering (smoothing) would tend to distribute this noise intensity over the pixels surrounding the noise spike. In contrast, the median filter usually removes this type of image noise without other degradation. The following example illustrates the approach.

Suppose we obtain $N = 5$ image intensity samples in a neighborhood of pixel location (x,y) that yield $R(\underline{x}) = \{100,110,120,130,240\}$. Clearly the pixel intensity value of 240 is either a noise spike or an image feature. In this case, the output of the rank filter yields $g(\underline{x}) = 120$.

The neighborhood, or window shape, chosen for the median filter greatly affects its filtering effects. The median filter may take an assortment of shapes. Examples are shown in Figure 4-5. The shape chosen for the window may be based on a priori knowledge of the image noise characteristics, such as horizontal or vertical orientations.

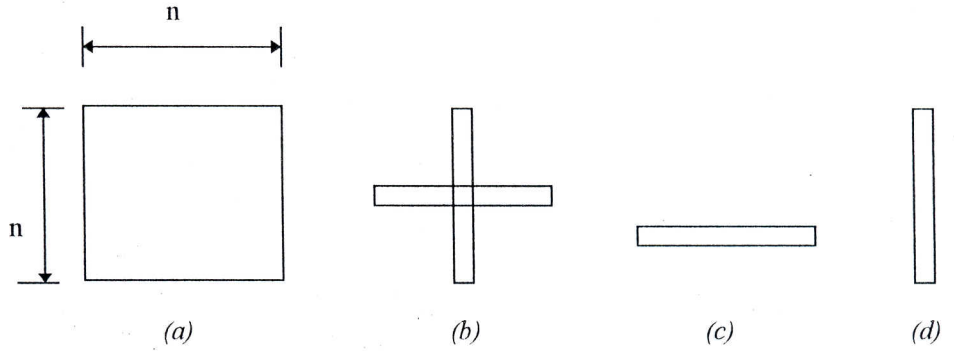


Figure 4-5 *Median filter window shapes*

(a) *n x n window*

(b) *n x n cross*

(c),(d) *Horizontal and vertical strips*

To avoid the computational expense of sorting large arrays of numbers, median filters often are implemented with small size window. The image may be repeatedly filtered with multiple passes of this window.

The median operator is obviously a nonlinear filter, because, given two image functions yielding sample sequences $R_1(\underline{x})$ and $R_2(\underline{x})$.

$$med\left(R_1(\underline{x}) + R_2(\underline{x})\right) \neq med\left(R_1(\underline{x})\right) + med\left(R_2(\underline{x})\right) \qquad (4\text{-}26)$$

Surprisingly, however, a number of properties of the median filter allow a limited quantitative analysis. For example, given a constant K and sequence $R(\underline{x})$,

$$med\left(KR(\underline{x})\right) = K \quad medR(\underline{x}) \qquad (4\text{-}27)$$

$$med\left(K + R(\underline{x})\right) = K + \quad medR(\underline{x}) \qquad (4\text{-}28)$$

Schalkoff (Schalkoff, 1989) has implied that the median filter has a number of other interesting properties.

1. *The median filter reduces the variance of the intensities in the image.* This shown pictorially in the input and output "noise" images and corresponding histograms in Figure 4-6. Thus, the median filter has a capability to significantly alter image texture.

2. *Intensity oscillations with a period less than the window width are smoothed.* This property is significant when considering multipass implementations of a fixed size median filter. In general, regions unchanged by the filter in a given pass are left unchanged in future passes.

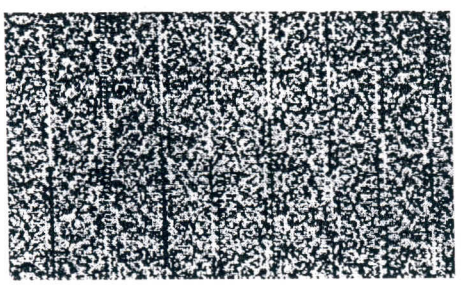
3. *Median filters will change the image intensity mean value* if the spatial noise distribution in the image is not symmetrical within the window.

4. *Median filters preserve certain edge shapes.* This is shown by the series of the examples in Figure 4-7. This is important in multipass median filtering implementations, because the fixed points of a median filter are primarily edges and regions of monotonic slope.

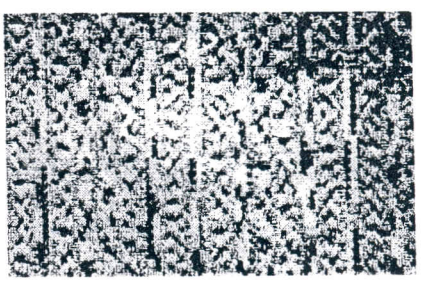
5. Given a symmetrical window shape, *the median filters preserves the location of edges.* This is also verified in Figure 4-7.

6. In the application of a median filter, *no new gray-values generated.* Binary images remain binary, and the dynamic range of a median filtered image cannot exceed that of the input image.

7. *The shape chosen for a median filter may affect the processing results.* This is shown in Figure 4-8.



(a)



(c)

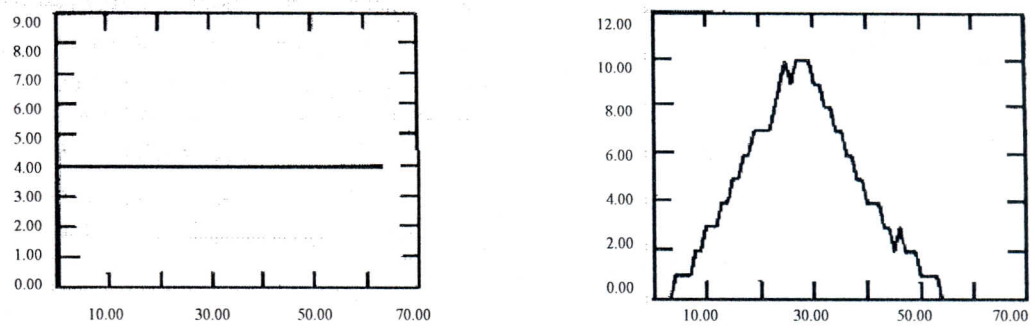


Figure 4-6 *Median filter statistical effects*

(a) Input "noise" image

(b) Histogram (normalized) of "noise"

(c) Median filtered "noise" image

(d) Median filtered "noise" histogram

Median Filter Practical Implementation and "Fast" Algorithms. Unlike linear filters, which involve multiplication and summation computations, *the median filter computation requires the sorting of a list of numbers.* For large window sizes, this may become a significant computational burden. The computation may be minimized several ways. One efficient approach for neighboring window locations (e.g., (x,y) and (x+1,y)) is based on *adjustment* and modification of the sorted list at the previous location, (x,y). As we move the *n x m* window we discard *n* points, add *n* new points, and leave the remaining *mxn* -2*n* points unchanged. This approach is only efficient if the amount of common image intensity data is significantly large from one position to the next.

A clever alternate approach, based on list sorting, is to calculate the median of $R(\underline{x})$ iteratively, by bit position , starting with the MSB. After the median bit (0 or 1) for a bit position is determined, only intensity values with this bit value are retained for subsequent calculations. Notice that *the procedure terminates, for* n-*bit data, after* n *iterations, regardless of the size of the window.*

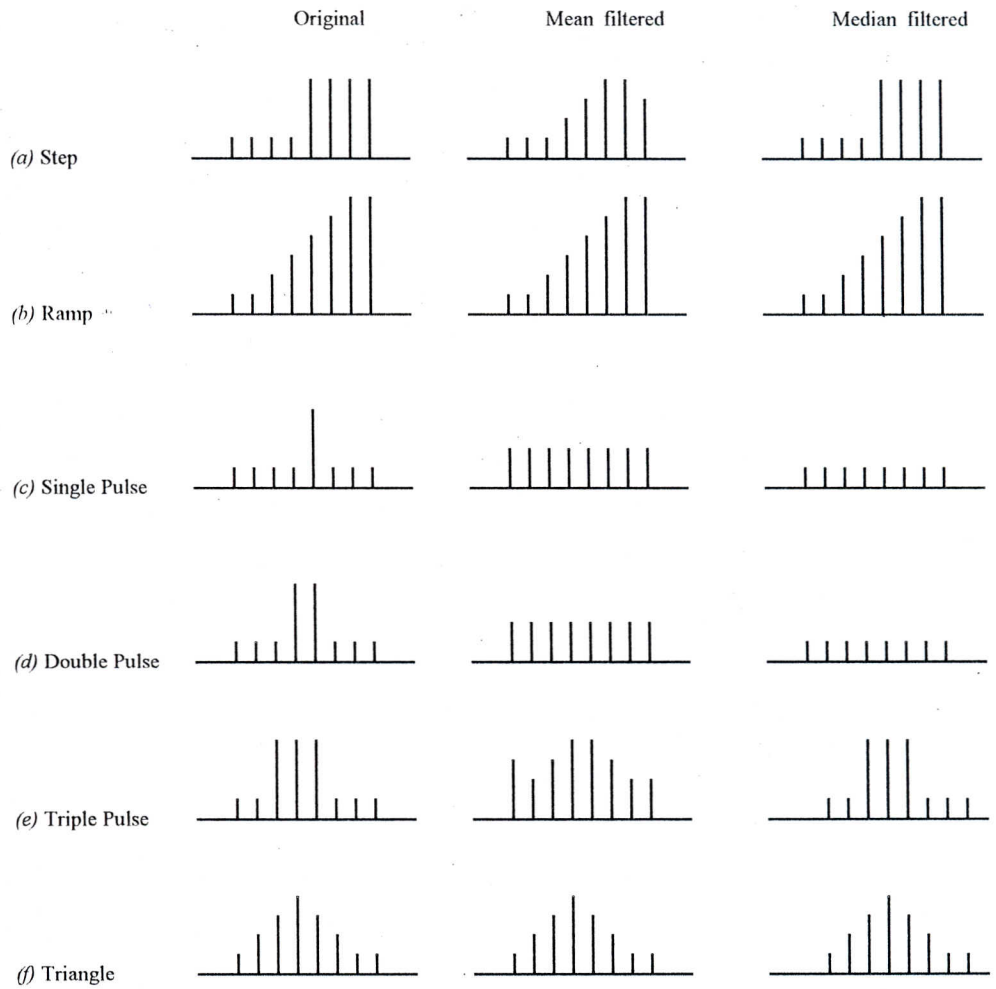


Figure 4-7 Median filter effects on intensity profiles

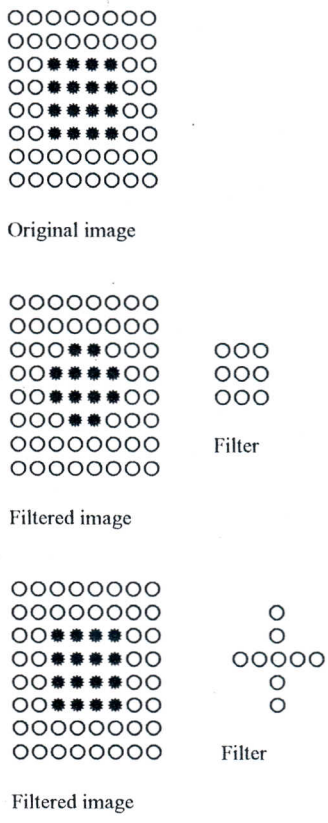


Figure 4-8 Median filter shape and edge location effects

Median filters are very effective in removing salt and pepper and impulse noise while retaining image details because they do not depend on values which are significantly different from typical values in the neighborhood. Median Filters work successive image windows in a fashion similar to linear filters. However, the process is no longer a weighted sum. For example, take a 3 x 3 window and compute the median of the pixels in each window centered around $[i,j]$:

1. Sort the pixels into ascending order by gray level.
2. Select the value of the middle pixel as the new value for pixel $[i,j]$.

This process is illustrated in Figure 4-9. In general, an odd-size neighborhood is used for calculating the median. However, if the number of pixels is even, the median is taken as the average of the middle two pixels after sorting.

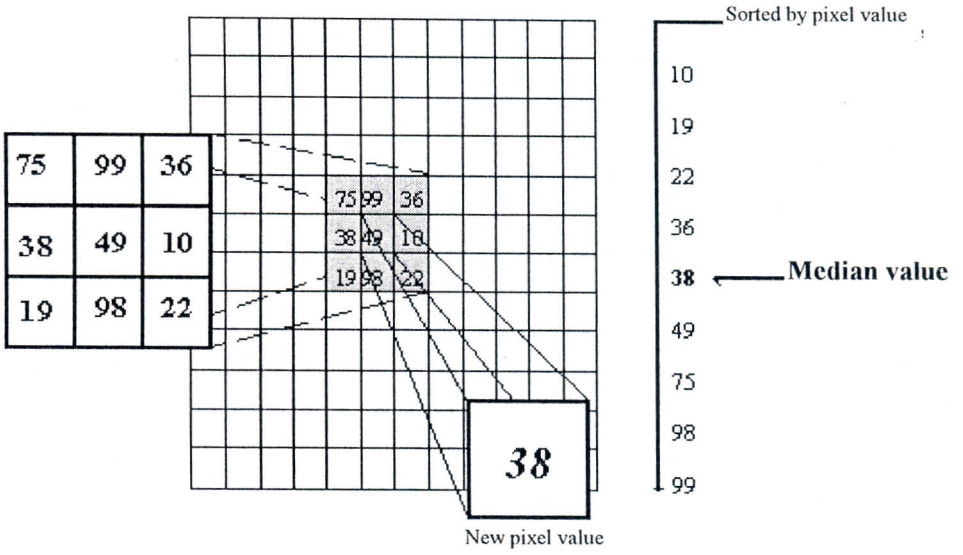


Figure 4-9 An example illustrating the median filter a 3 x 3 neighborhood.

4.2.3.2 Gaussian Smoothing

The Gaussian smoothing filter is a very good filter for removing noise drawn from a normal distribution. The zero-mean Gaussion function in one dimension is

$$g(x) = e^{-\frac{x^2}{2\sigma^2}} \tag{4-29}$$

where the Gaussian spread parameter σ determines the width of the Gaussian. For image processing, the two-dimensional zero-mean discrete Gaussian function,

$$g[i,j] = e^{-\frac{(i^2+j^2)}{2\sigma^2}} \tag{4-30}$$

is used as a smoothing filter. A plot of this function is shown in Figure 4-10.

Gaussian functions have five properties that make them particularly useful in early vision processing. These properties indicate that the Gaussian smoothing filters are effective low-pass filters from the perspective of both the spatial and frequency domains, are efficient to implement, and can be used effectively by engineers in practical vision applications. The five properties are summarized below.

1. In two dimensions, Gaussian functions are rotationally symmetric. This means that the amount of smoothing performed by the filter will be the same in all directions. In general, the edges in an image will not be oriented in some particular direction that is known in advance; consequently, there is no reason a priori to smooth in one direction

than in another. The property of rotational symmetry implies that a Gaussian smoothing filter will not bias subsequent edge detection in any particular direction.

2. The Gaussian function has a single lobe. This means that a Gaussian filter smoothes by replacing each image pixel with a weighted average of neighboring pixels such that the weight given to a neighbor decreases monotonically with distance from the central pixel. This property is important since an edge is local feature in an image, and a smoothing operation that gives more significance to pixels farther away will distort the features.

3. The Fourier transform of a Gaussian has a single lobe in the frequency spectrum. This property is straightforward collary of the fact that the Fourier transform of a Gaussian is itself a Gaussian, as will be shown below. Images are often corrupted by undesirable high-frequency signals (noise and fine texture). The desirable image features, such as edges, will have components at *both* low and high frequencies. The single lobe in the Fourier transform of a Gaussian means that the smoothed image will not be corrupted by contributions from unwanted high-frequency signals, while most of the desirable signals will be retained.

4. The width, and hence the degree of smoothing, of a Gaussian filter is parameterized by σ , and the relationship between σ and the degree of smoothing is very simple. A larger σ implies a wider Gaussian filter and greater smoothing. Engineers can adjust the degree of smoothing to achieve a compromise between excessive blur of the desired image features (too much smoothing) and excessive undesired variation in the smoothed image due to noise and fine texture (too little smoothing).

5. Large Gaussian filters can be implemented very efficiently because Gaussian functions are separable. Two-dimensional Gaussian convolution can be performed by convolving the image with one-dimensional Gaussian and then convolving the result with the same one-dimensional filter oriented orthogonal to the Gaussian used in the first stage. Thus, the amount of computation required 2-D Gaussian filter grows linearly in the width of the filter mask instead of growing quaratically.

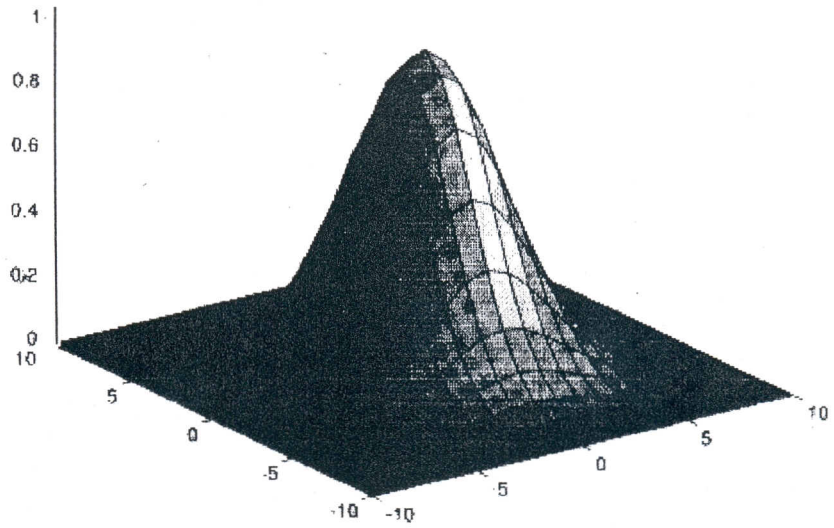


Figure 4-10 The two-dimensional Gaussian function with zero-mean

The rotational symmetry of the Gaussian can be shown by converting the function from rectangular to polar coordinates. Since the radius in polar coordinates is given by $r^2 = i^2 + j^2$, it is easy to see that the Gaussian function in polar coordinates,

$$g[r,\theta] = e^{-\frac{r^2}{2\sigma^2}} \qquad (4\text{-}31)$$

does no depend on the angle θ and consequently is rotationally symmetric. It is also possible to construct rotationally nonsymmetric. Gaussian functions if they are required for an application where it is known in advance that more smoothing must be done in some specified direction. Formulas for rotationally nonsymmetric. Gaussian functions are provided by Wonzencraft and Jacobs, where they are used in the probabilistic analysis of communication channels.

The Gaussian function has the interesting property that its Fourier transform is also a Gaussian function. Since the Fourier transform of a Gaussian is real function, the Fourier transform is its own magnitude. The Fourier transform of a Gaussian is computed by

$$F\{g(x)\} = \int_{-\infty}^{\infty} g(x)e^{-j\omega x}dx$$

$$= \int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} e^{-j\omega x}dx$$

$$= \int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} (\cos\omega x + j\sin\omega x)dx \qquad (4\text{-}32)$$

$$= \int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} \cos\omega x dx + j\int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} \sin\omega x dx$$

The gaussian is a symmetric function and the *sine* function is a antisymmetric so the integrated in the second integral is antisymmetric. Therefore, the integral must be zero, and the Fourier transforms simplifies to:

$$F\{g(x)\} = \int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}} \cos\omega x dx \qquad (4\text{-}33)$$

$$= \sqrt{2\pi}\sigma e^{-\frac{\omega^2}{2\nu^2}}, \qquad \nu^2 = \frac{1}{\sigma^2} \qquad (4\text{-}34)$$

The spatial frequency parameter is ω, and the spread of the Gaussian in the frequency domain is controlled by ν, which is the reciprocal of the spread parameter σ in the spatial domain. has a wider spectrum, and a wider Gaussian function in the spatial domain has a narrower spectrum. This property relates to the noise suppression ability of a Gaussian filter. A narrow-spatial-domain Gaussian does less smoothing, and in the frequency domain its spectrum has more bandwidth and passes more of the high-frequency noise and texture. This simple relationship between spatial-domain Gaussian width and frequency-domain spectral width enhances the ease of use the Gaussian filter in practical design situations. The Fourier transform duality of Gaussian functions also explains why the single-lobe property in the spatial domain carries over into the frequency domain.

Another useful property of the Gaussian functions is their seperability. The seperability of Gaussian filters is easy to demonstrate:

$$g[i,j] * f[i,j] = \sum_{k=1}^{m}\sum_{l=1}^{n} g[k,l]f[i-k,j-l] \qquad (4\text{-}35)$$

$$= \sum_{k=1}^{m}\sum_{l=1}^{n} e^{-\frac{(k^2+l^2)}{2\sigma^2}} f[i-k,j-l] \qquad (4\text{-}36)$$

$$= \sum_{k=1}^{m} e^{-\frac{k^2}{2\sigma^2}} \left\{ \sum_{l=1}^{n} e^{-\frac{l^2}{2\sigma^2}} f[i-k, j-l] \right\} \quad (4\text{-}37)$$

The summation in brackets is the convolution of the input image $f[i,j]$ with a vertical one-dimensional Gaussian function. The result of this summation is a two-dimensional image, blurred in the vertical dimension, that is then used as the input to a second convolution with a horizontal one-dimensional Gaussian that blurs the image in the horizontal dimension (see Figure 4-11). Since convolution is associative and commutative, the order of the convolutions can be reversed so that the horizontal convolution is performed first and the vertical convolution is performed on the result of the horizontal convolution.

This method can be implemented using the composition of two horizontal convolutions and a single horizontal convolution mask. The input $f[i,j]$ is first convolved with a horizontal Gaussian, and the result is placed in a temporary array in its transposed position. The temporary array is then used as input to the same convolution code so that the vertical convolution is performed by horizontal convolution. The output data from second convolution is again transposed as the convolution is performed so that the data is restored to its proper (original) orientation.

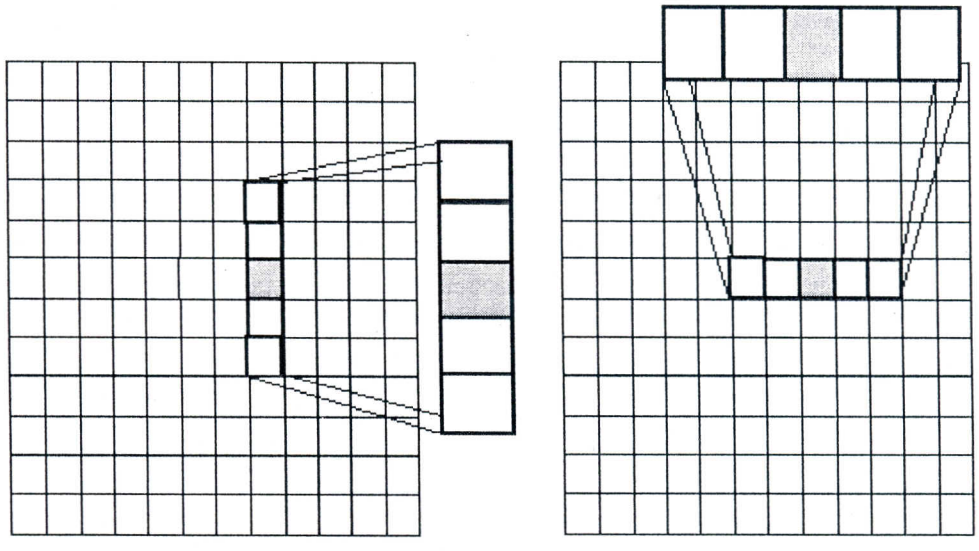


Figure 4-11 An example of the seperability of Gaussian convolution. Left: Convolution with the vertical mask. Right: Convolution with the horizontal mask. Note that the origin of each mask is shaded.

The samples of a Gaussian filter, or the coefficients obtained from the binominal expansion, form a Gaussian filter. When discrete Gaussian filters are convolved, the result is a larger discrete Gaussian filter. If an image is smoothed with an $n \times n$ discrete Gaussian filter, and this intermediate result is smoothed by $m \times m$ discrete Gaussian filter, then the result is exactly the same if the original image had been smoothed by an $(n+m-1) \times (n+m-1)$ discrete Gaussian filter. In other words, convolving row n in Pascal's triangle with row m yields row $n+m-1$ in Pascal's triangle.

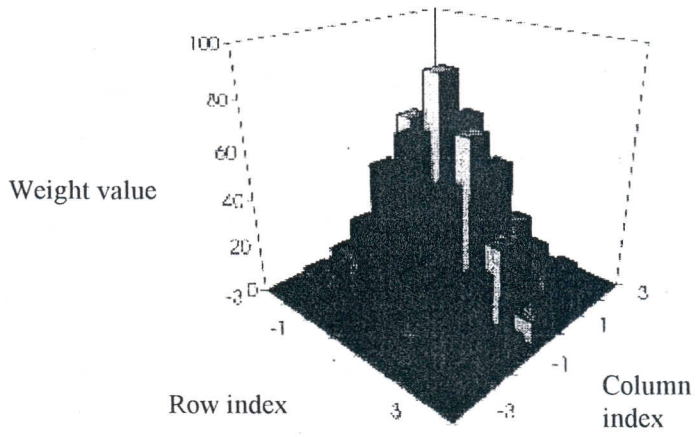


Figure 4-12 A 3-D plot of the 7 x 7 Gaussian mask.

4.2.4 Conversion of Gray-Level to Binary Images

An image contains a continuum of intensity values before it is quantized to obtain a digital image. The information in an image is in these gray values. To interpret an image, the variations in the intensity values must be analyzed. The most commonly used number of quantization levels for representing image intensities is 256 different gray levels. It is not uncommon, however, to see digital images quantized to 32, 64, 128, or 512 intensity levels for certain applications, and even up to 4096 (12 bits) are used in medicine. Clearly, more intensity levels allow better representation of the scene at the cost of more storage.

In the early days of machine vision, the memory and computing power available was very limited and expensive. These limitations encouraged designers of vision applications to focus their efforts on binary vision systems. A binary image contains only

two gray level. The difference this makes in the representation of a scene is shown in Figure 4-13.

In addition, designers noted that people have no difficulty in understanding line drawings, silhouettes, and other images formed using only two gray levels. Encouraged by this human capability, they used binary images in many applications.

Even though computers have become much more powerful, binary vision systems are still useful. First of all, the algorithms for computing properties of binary images are well understood. They also tend to be less expensive and faster than vision systems that operate on gray level or color images. This is due to the significantly smaller memory and processing requirements of binary vision. The memory requirements of a gray level system working with 256 gray levels will be eight times that of a system working with a binary image of the same size. The storage size may be reduced by using techniques such as run-length encoding. The processing time requirements are lower because many operations on binary images may be performed as logical operations instead of integer mathematical operations.

Smaller memory requirements and faster execution times are not the only reasons for studying binary vision systems. Many techniques developed for these systems are also applicable to vision systems which use gray scale images. A convenient way to represent an object in a gray level or color image is to use its masks. The mask of an object is a binary picture in which the object points are 1 and other points are 0. After an object has been separated from the background, its geometric and topological properties may be required in decision making. These properties can be computed from its binary image.

In general, binary vision systems are useful in cases where a silhouette contains enough information to allow the recognition of an object and where the environment can be adequately controlled. To obtain a good silhouette, the objects must be easily separated from background. This can be achieved by using special illumination techniques and by having only a few objects in the scene. There are many industrial situations that fulfill these requirements. For example, binary vision systems have found application in optical character recognition, chromosome analysis, and recognition of industrial parts. In these cases, the binary vision system usually uses a threshold to separate objects from the background. The proper value of this threshold depends on illumination and on reflectance characteristics of objects. The resulting binary picture

allows computation of geometric and topological properties (*features)* of objects for the given mask. In many applications, these characteristics are enough for recognition of objects.

It should be mentioned here, however, that with the increase in the complexity of applications, more and more vision systems are using gray scale images. This is due to the fact that in many material handling and assembly tasks, the illumination cannot be controlled to obtain good contrast between objects and background. Care has to be exercised to make the system insensitive to small changes in illumination and reflectance characteristics of other objects in a scene. In many applications, this becomes a formidable task. Similarly, in inspection tasks, it may not be possible to recover subtle information using only two intensity levels. Internal details of an object may be lost in thresholding and may make the task of detecting surface defects very difficult.

Objects pixels will have the value 1 and background pixels will have 0. In displaying pictures, 0 is white and 1 is black; thus, in binary images, the background is white and objects are black. We will also assume that pictures are of size $n \times m$ pixels and are represented in a computer as a two-dimensional array. This representation allows us to visualize images with the spatial relationships between points maintained in the form familiar to people.

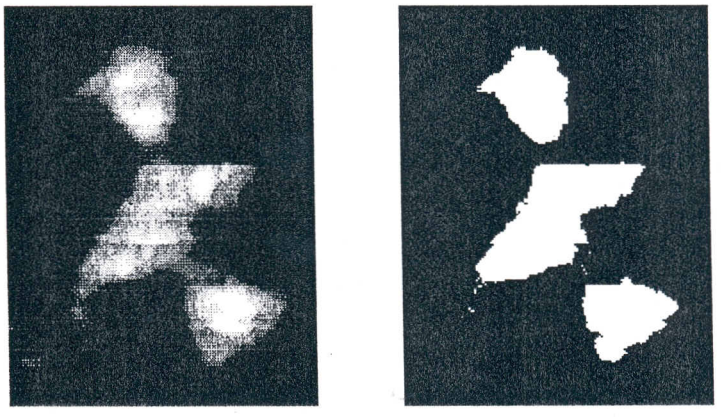


Figure 4-13 A gray level image and its corresponding binary image

Conversion of gray-level images to binary representation is important for a number of reasons:

1. To identify the extend of objects, represented as a region in the image. For example, we may wish to separate a visual target from its background.

2. To concentrate on shape (or morphological) analysis, in which case the intensities of pixels are less significant than the shape of a region.

3. To display an image on an output device which has only one bit intensity resolution, that is, a binary or two-level display device, such as printer. The objective is to convert the gray-level image so that the subjective appearance is that of a multi-intensity image. The advent of desktop publishing makes this an area of growing significance.

4. To convert an edge-enhanced image to a line drawing of a the imaged scene. It is necessary to distinguish strong edges that correspond to object outlines (and features) from weak edges due to illumination change, shadows, etc.

4.2.4.1 Thresholding

One of the most important problems in a vision system is to identify the subimages that represents objects. This operation, which is so natural and so easy for people, is surprisingly difficult for computers. The partitioning of an image into regions is called segmentation. Ideally, a partition represents an object or part of an object. Formally, segmentation can be defined as a method to partition an image, $F[i,j]$, into subimages, called regions, $P_1,\ldots,P_k$, such that each subimage is an object.

- $\cup_{i=1}^{k} P_i$ = Entire image ($\{P_i\}$ is an *exhaustive* partitioning.)
- $P_i \cap P_j = \varnothing, \quad i \neq j$ ($\{P_i\}$ is an *exclusive* partitioning)
- Each region P_i satisfies a predicate; that is, all points of the partition have *some* common property.
- Pixels belonging to adjacent regions, when taken jointly, do not satisfy the predicate.

As shown above, a partition satisfies a predicate. This predicate may be as simple as *has uniform intensity* but is more complex in most applications. Segmentation is a very important step in understanding images.

A binary image is obtained using an appropriate segmentation of a gray scale image. If the intensity values of an object are in an interval and the intensity values of the background pixels are outside this interval, a binary image can be obtained using a thresholding operation that sets the points in that interval to 1 and points outside that

range to 0. Thus, for binary vision, segmentation and thresholding are synonymous. Many cameras have been designed to perform this thresholding operation in hardware. The output of such a camera is a binary image. In most applications, however, cameras give a gray scale image and the binary image is obtained using thresholding.

Thresholding is a method to convert a gray scale image into a binary image so that objects of interest are separated from the background. For thresholding to be effective in *object-background separation*, it is necessary that the objects and background have sufficient contrast and that we know the intensity levels of either the objects or the background. In a fixed thresholding scheme, these intensity characteristics determine the value of the threshold.

Let us assume that a binary image $B[i,j]$ is the same as a thresholded gray image $F_T[i,j]$ which is obtained using a threshold T for the original gray image $F[i,j]$. Thus,

$$B[i,j] = F_T[i,j] \tag{4-38}$$

where for a darker object on a lighter background

$$F_T[i,j] = \begin{cases} 1 & if \quad F[i,j] \le T \\ 0 & otherwise. \end{cases} \tag{4-39}$$

If it is known that the object intensity values are in a range $[T_1, T_2]$, then we may use

$$F_T[i,j] = \begin{cases} 1 & if \quad T_1 \le F[i,j] \le T_2 \\ 0 & otherwise. \end{cases} \tag{4-40}$$

A general thresholding scheme in which the intensity levels for an object may come from several disjoint intervals may be represented as

$$F_T[i,j] = \begin{cases} 1 & if \quad F[i,j] \in Z \\ 0 & otherwise \end{cases} \tag{4-41}$$

where Z is a set of intensity values for object components.

Note how knowledge about the application domain is incorporated into the thresholding algorithm. It has, in fact, been tailored for the domain; therefore, the same threshold values may not work in a new domain. The threshold is usually selected on the basis of experience with the application domain. In some cases, the first few runs of the system may be used for interactively analyzing a scene and determining an appropriate value for the threshold.

Automatic thresholding of images is often the first step in the analysis of images in machine vision systems. Many techniques have been developed for utilizing the intensity distribution in an image and the knowledge about the objects of interest for selecting a proper threshold value automatically.

4.2.4.2 Automatic Thresholding

To make segmentation more robust, the threshold should be automatically selected by the system. Knowledge about the objects in the scene, the application, and the environment should be used in the segmentation in a form more general than a fixed threshold value. Such knowledge may include

- Intensity characteristics of objects
- Sizes of the objects
- Fractions of an image occupied by the objects
- Number of different types of objects appearing in an image

A thresholding scheme that uses such knowledge and selects a proper threshold value for each image without human intervention is called an automatic thresholding scheme. Automatic thresholding analyzes the gray value distribution in an image, usually by using a histogram of the gray values, and uses the knowledge about the application to select the most appropriate threshold. Since the knowledge employed in these schemes is more general, the domain of applicability of the algorithm is increased.

suppose that an image contains n objects $O_1, O_2, \ldots, O_n$, including the background, and gray values from different populations $\pi_1, \ldots, \pi_n$ with probability distributions $p_1(z), \ldots, p_n(z)$. In many application, the probabilities $P_1, \ldots, P_n$ of the objects appearing in an image may also be known. Using this knowledge, it is possible to rigorously formulate the threshold selection problem. Since the illumination geometry of a scene controls the probability distribution of intensity values $p_i(z)$ in an image, one cannot usually precompute the threshold values. Most methods have been developed for automatic threshold selection use the size and probability of occurrence and estimate intensity distributions by computing histograms of the image intensities.

Many automatic thresholding schemes have been used in different applications. To simplify the presentation, we will follow the convention that objects are dark against

a light background. In discussing thresholds, this allows us to say that gray values below a certain threshold belong to the object and gray values above the threshold are from the background, without resorting to more cumbersome language. Some algorithms can be generalized to handle object gray values from an arbitrary set of pixel values.

4.2.4.2.1 P-Tile Method

The p-tile method uses knowledge about the area or size of the desired object to threshold an image. Suppose that in a given application objects occupy about p percent of the image area. By using this knowledge to partition the gray value histogram of the input image, one or more thresholds can be chosen that assign p percent of the pixels to the object. Figure 4-14 gives an example of a binary image formed using this technique.

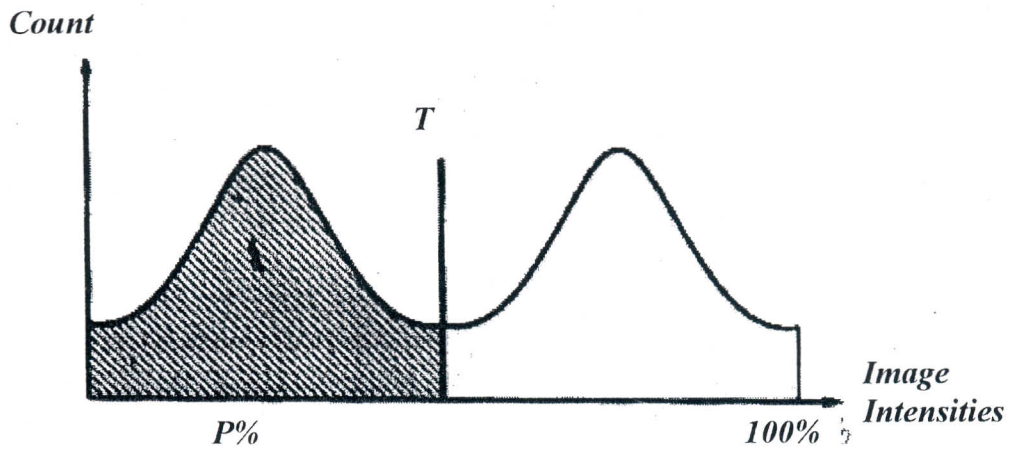


Figure 4-14 The shaded areas in the histogram represent p percent of the image area. The threshold is selected so that p percent of the histogram is assigned to the object.

Clearly, this method is of very limited use. Only a few application, such as page readers, allow such an estimate of the area in a general case.

4.2.4.2.2 Mode Method

If the objects in an image have the save gray value, the background has a different gray value, and image pixels are affected by zero-mean Gaussian noise, then we may assume that the gray values are drawn from two normal distributions with parameters (μ_1, σ_1) and (μ_2, σ_2). The histogram for an image will the show two

separate peaks, as shown in Figure 4-14. In the ideal case of constant intensity values $\sigma_1 = \sigma_2 = 0$, there will be two spikes in the histogram and the threshold can be placed anywhere between the spikes. In practice, the two peaks are not so well separated. In this case, we may detect peaks and valleys in the histogram, and the threshold may be set to the pixel value corresponding to the valley. It can be shown that the probability of misclassification is minimized by this choice of the threshold when the size of the object is equal to that of the background. In most applications, since the histogram is sparsely populated near the valley, the segmentation is not sensitive to the threshold value.

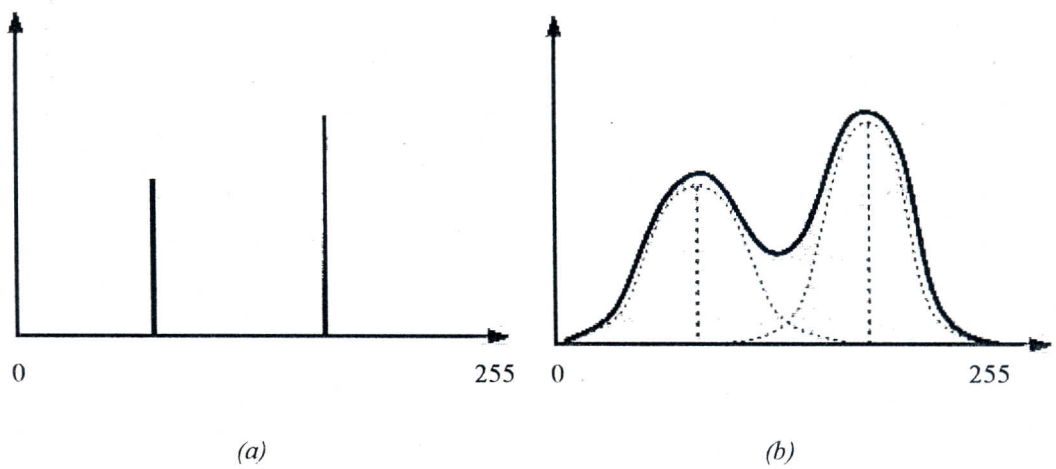


Figure 4-15 Ideally, the intensities of the background and objects will be widely separated. In the ideal case, the threshold T can be anywhere between the two peaks, as shown in (a). In most images, the intensities will overlap, resulting in histograms as shown in (b).

The determination of peaks and valleys is a nontrivial problem, and many methods have been proposed to solve it. For an automatic thresholding scheme, we should have a measure of the *peakness* and *valleyness* of a point in a histogram. A computationally efficient method is given in Algorithm 4-1. This method ignores local peaks by considering peaks that are at some minimum distance apart. The peakness is based on the height of the peaks and the depth of the valleys; the distance between the peeks and valleys is ignored.

This approach can be generalized to images containing many objects with different mean gray values. Suppose there are *n* objects with normally distributed gray values with parameters $(\mu_1,\sigma_1),(\mu_2,\sigma_2),\ldots,(\mu_n,\sigma_n)$, and the background is also normally distributed wit parameters (μ_0,σ_0). If the means are significantly different, the variances are small, and none of the objects is very small in size, then the histogram for

the image will contain n+1 peaks. The valley locations $T_1, T_2, \ldots, T_n$ can be determined, and pixels with gray values in each interval $(T_i, T_{i+1}]$ can be assigned to the corresponding object (Figure 4- 16).

4.2.4.2.3 Peakness Detection Algorithm

1. *Find the two highest local maxima in the histogram that are at some minimum distance apart.*
2. *Find the lowest point* g_k *in the histogram H between* g_i *and* g_j.
3. *Find the peakness, defined as* $\min\left(H(g_i), H(g_j)\right) / H(g_k)$.
4. *Use the combination* (g_i, g_j, g_k) *with highest peakness to threshold the image. The value* g_k *is a good threshold to separate objects corresponding to* g_i *and* g_j.

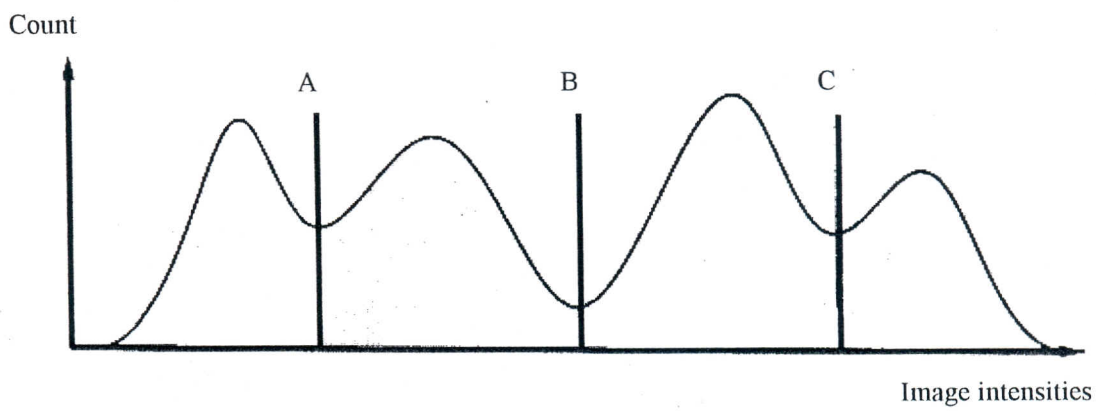


Figure 4-16 Histogram for an image containing several objects with different intensity values

4.3 Image Morphology

4.3.1 Introduction

Morphology is the science of form and structure. In computer vision it is about regions or shapes-how they can be changed and counted, and how their areas can be evaluated.

Although, morphological filters can be used to perform so many different tasks, Low (Low ,1991) has classified its application areas under four main titles as below:

1. Smoothing the edges of a region. This is useful if, say, it is required to create a line drawing of a face from image. Using standard segmentation techniques, the edges will be noisy because the lighting or capture equipment makes the outline inaccurate. Morphology can not add information to this, but since it is known that the edges of a face are normally curved, then the unlikely hills and valleys can be smoothed away.
2. Forcing shapes onto region edges. Instead of edges being curved it may be that prior knowledge is available that regions are square-e.g., in the image of chessboard. Morphology can be used to force a shape onto a region so that where it was noisy and approximately square it becomes more square and less noisy.
3. The same morphological operations can be used to count regions (or granules in morphological terms). For example, how many dark cells are there in the image? Using morphological operations is just way of region of granule counting.
4. Morphological operations can be used to estimate sizes of regions (or granules). This is clearly essential as a tool for the image processor.

Morphological operations are most easily seen on binary images (that is 1s and 0s only). Indeed, the early mathematics of morphology was concerned purely with position rather than intensity.

4.3.2 Basic Morphological Operations

Consider the following image:

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

This could effectively be represented as a set (in mathematical terms) of all those pixels in 5 x 3 image that have value '1', namely

$$\{(0,0),(0,2),(0,4),(1,1),(1,3),(1,5),(2,0),(2,2),(2,4)\}$$

And now it is possible to do set operations on images

$$A = \begin{matrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{matrix} \qquad B = \begin{matrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{matrix} \tag{4-42}$$

Giving

$$A \cup B \quad (A \ \textit{union} \ B) = \begin{matrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{matrix} \tag{4-43}$$

$$A \cap B \quad (A \ \text{intersection} \ B) = \begin{matrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{matrix} \tag{4-44}$$

Note here that the matrix really represents a set of known pixels (value'1') in among a set of unknown pixels (represented by '0'). The set of unknown pixels theoretically extends infinitely up, down, left, and right so that image A above could also be written as follows:

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

where the circle around the top left 1 indicates the position of the origin.

Two set operations (using an image and template) that are fundamental are called dilation and erosion. The first has the effect of filling in the valleys between spiky regions edges, while the second has the effect of deleting spiky edges. These operations derive from Minkowski addition and Minkowski subtraction (Low ,1991) respectively, and are defined as follows.

4.3.2.1 Dilation (Minkowski addition)

A template (made from 1s and 0s) is created with a known origin, denoted by a circle around it.

The origin of this template is stepped over every element in the whole of the image. Where the origin of the template corresponds to a 1 in the image, the template is 'unioned' with that part of image. The resulting template-sized results *using their original image positions* and giving a resulting image that is larger (unless the template is 1 x 1) than the original.

Image

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

Template

1	0
(1)	0

The 'first' 1 in the image on the top line

0 0 0 0 0 0 1 0 0 1 0

When the template is applied to this 1, the following results:

0 0 0 0 0 0 1 0 0 0 0

0 0 0 0 0 0 1 1 0 1 0 ← new top line

And then application of the template to the second gives

0 0 0 0 0 0 1 0 0 1 0

0 0 0 0 0 0 1 1 0 1 1

Application across the whole image gives

0	0	0	0	0	0	1	0	0	1	0	0	← new row
0	0	0	0	0	0	1	1	0	1	1	0	
0	0	0	0	0	1	1	1	1	1	1	1	
0	0	0	0	1	1	1	1	1	1	1	1	
0	0	0	0	1	1	1	1	1	1	1	1	
0	0	0	0	1	1	1	1	1	1	1	1	
0	0	0	0	1	1	1	1	1	1	1	1	
0	0	0	0	1	1	1	1	1	1	1	1	
									↑ new column			

Continually applying this to the image fills out all the holes and makes the image grow, one row and one column at a time.

4.3.2.2 Erosion (Minkowski subtraction)

This operation removes spikes from the edges of the region. This performs with the template which is created for each dilation. The template is stepped over the image but this time only in positions where the whole of the template lies on the top of the image, i.e. it is now allowed to go off the edge. This means that the resulting matrix will be smaller than the image (unless the template is 1 x 1).

For every stepped position in the image the template is compared with the corresponding window to the image. If the template is exactly the same as the image window, then the element corresponding to the origin in the resulting matrix is set to a 1.

This is particularly useful when the template is all 1s. It is also effectively equivalent to correlation. This is clear with a template containing a 0 as follows:

Image

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

Template

1	0
(1)	0

The first 'match' is at (6,3), there being only two other matches, so the result is:

```
0 0 0 0 0 0 0 0 0 0 0   ← Unused row
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0 0 1 0
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 0 0

          ↑ Unused column
```

This shows that the template matches the image in only three places. However, with a 'all 1s' template the spike removal becomes apparent. For example, with the same image and the template

```
1 1
1 1
```

the result is

```
0 0 0 0 0 0 0 0 0 0 0   ← Unused row
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 1 0 0
0 0 0 0 1 1 1 1 1 0 0
0 0 0 0 1 1 1 1 0 0 0
0 0 0 0 1 1 1 1 1 1 0

          ↑ Unused column
```

This clearly reduces the image, cutting out the spiky edges.

The techniques above have been applied to normal gray-level images. One way is to associate all the elements that have a constant gray-level value with binary 1 and all other pixels with 0 . A better approach uses the actual values in gray-level

morphological operations defined slightly differently to the binary/unary operations defined above.

4.3.2.3 Gray-level Erosion and Dilation

It is used for flattening and filling valleys on region edges in gray-level images. Let *I(x,y)* be an image of gray levels and *R(x,y)* be the resulting image after *I(x,y)* has been dilated/eroded with *m x n* template *T(x,y)*.

$$0 \le i \le m-1, \qquad 0 \le j \le n-1 \tag{4-45}$$

Gray level dilation is defined as

$$R(x,y) = \underset{\substack{0\le i\le m-1\\ 0\le j\le n-1}}{\text{maximum}}\{I(x-i,y-j)+T(i,j)\} \tag{4-46}$$

and gray level erosion is

$$R(x,y) = \underset{\substack{0\le i\le m-1\\ 0\le j\le n-1}}{\text{minimum}}\{I(x+i,y+j)-T(i,j)\} \tag{4-47}$$

Both of the above definitions use locations outside of an *M x N* image, so where *I(x,y)* is not defined, for the purposes of implementing the above calculations *I(x,y)*=0.

The dilation example uses *(x-i,y-j)* coordinates, which are not ideal in that the template is effectively rotated about both diagonals and then operates on the elements to the left and above the 'home' position. This can be avoided by defining the coordinates on the template to be $0 \le i \le m-1$, and $0 \le j \le n-1$.

Note that dilation is precisely the dual of the erosion operation. In some circumstances this mathematical relationship is important.

Consider the following initial image:

0	0	0	0	0	0	0	0
0	0	2	4	4	2	0	0
0	0	4	8	8	4	0	0
0	0	2	4	4	2	0	0
0	0	0	0	0	0	0	0

Dilating by the template

$$\begin{matrix} 1 & 1 \\ 1 & 1 \end{matrix}$$

gives

$$\begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 3 & 5 & 5 & 5 & 3 & 1 \\ 1 & 1 & 5 & 9 & 9 & 9 & 5 & 1 \\ 1 & 1 & 5 & 9 & 9 & 9 & 5 & 1 \\ 1 & 1 & 3 & 5 & 5 & 5 & 3 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{matrix}$$

or subtracting 1 throughout

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 4 & 4 & 4 & 2 & 0 \\ 0 & 0 & 4 & 8 & 8 & 8 & 4 & 0 \\ 0 & 0 & 4 & 8 & 8 & 8 & 4 & 0 \\ 0 & 0 & 2 & 4 & 4 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Showing that dilation with a constant (all 1s) template is equivalent to a stepping spatial operation that takes the maximum of a pixel window.

Conversely, eroding by the same template gives

$$\begin{matrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 1 & 3 & 1 & -1 & -1 & -1 \\ -1 & -1 & 1 & 3 & 1 & -1 & -1 & -1 \\ -1 & -1 & 1 & 3 & 1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \end{matrix}$$

or if 1 is subtracted throughout

0	0	0	0	0	0	0	0
0	0	2	4	2	0	0	0
0	0	2	4	2	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Showing that erosion with a constant template is equivalent to taking a spatial minimum. The values in the template (or structuring element) effectively define the set of pixels which are searched for the maximum (dilation) or the minimum (dilation).

Note that dilation gives a larger image, while erosion gives a smaller image; in both cases, however, the original structure of the image is maintained.

If the template is not constant, say

1	0
0	1

the results are as follows:

Original								Dilated - 1								Eroded + 1							
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	2	4	4	2	0	0	0	0	2	4	4	3	1	0	0	0	2	4	3	0	0	0
0	0	4	8	8	4	0	0	0	0	4	8	8	7	3	0	0	0	3	4	2	0	0	0
0	0	2	4	4	2	0	0	0	0	3	7	8	8	4	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	3	4	4	2	0	0	0	0	0	0	0	0	0

Showing how dilation skews the image in favor of the higher values in the template. If the range of values in the template is small the shape of the template has much less effect on the final result than if the range is large, e.g. 0 to 100. Generally, if T is the template used for dilation and erosion, $D(I)$ is the dilated image of I, and $E(T)$ is the eroded image of I, then

$$D(E(D(I)))=D(I) \tag{4-48}$$

i.e., the result of eroding and then dilating a previously dilated image is a return to that original dilated image.

Similarly

$$E(D(E(I)))=E(I) \tag{4-49}$$

Clearly, changing the templates for erosion and dilation gives different results.

Chapter 5

EDGE DETECTION AND TRACKING

5.1 Introduction

The early stages of vision processing identify features in images that are relevant to estimating the structure and properties of objects in a scene. Edges are one such feature. Edges are significant local changes in the image and are important features for analyzing images. Edges typically occur on the boundary between two different regions in an image. Edge detection is frequently the first step in recovering information from images.

An edge in an image is a significant local change in the image intensity, usually associated with a discontinuity in either the image intensity or the first derivative of the image intensity. Discontinuities in the image intensity can be either (1) *step* discontinuities, where the image intensity abruptly changes from one value on one side of the discontinuity to a different value on the opposite side, or (2) *line* discontinuities, where the image intensity abruptly changes value but then returns to the starting value within some short distance. However, step and line edges are rare in real images. Because of low-frequency components or the smoothing introduced by most sensing devices, sharp discontinuities rarely exist in real signals. Step edges become *ramp* edges and line edges become *roof* edges, where intensity changes are not instantaneous but occur over a finite distance. Illustration of these edge profiles are shown in Figure 5-1.

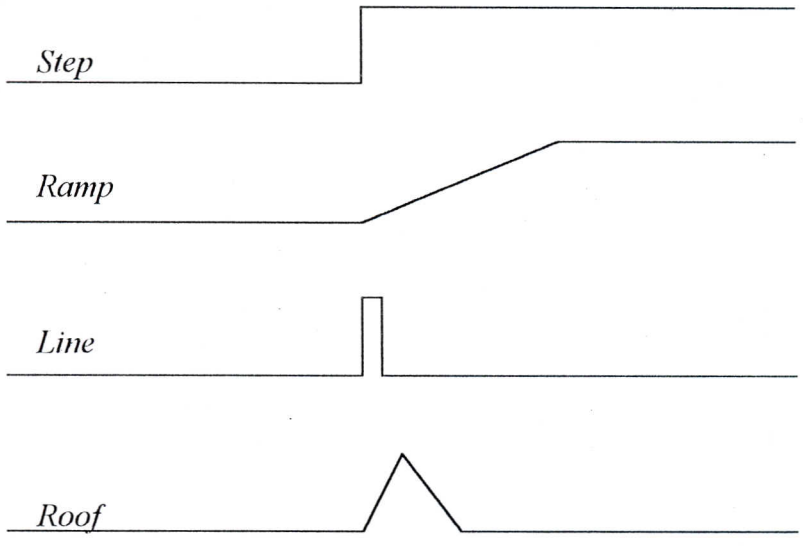


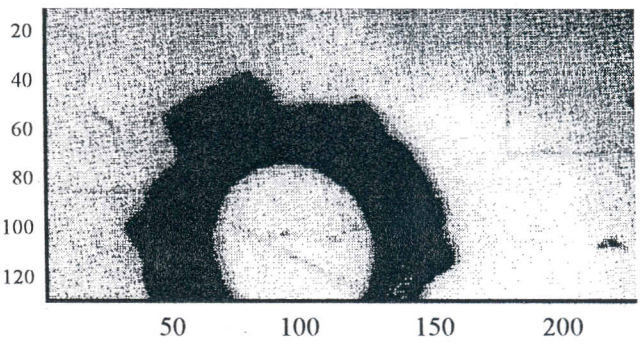
Figure 5-1 One-dimensional edge profiles.

It is also possible for an edge to have both step and line characteristics. For example, a surface that changes orientation from one flat surface to another will produce a step edge; but if the surface has a specular component of reflectance and if the surface corner is rounded, there can be highlight due to the specular component as the surface orientation of the rounded corner passes the precise angle for specular reflection. The edge profile generated by such a situation looks like a step edge with a superimposed line edge. There also edges associated with changes in the first derivative of the image intensity. For example, mutual reflection from the sides of a concave corner generate roof edges. Edges are important image features since they may correspond to significant features of objects in the scene. For example, the boundary of an object usually produces step edges because the image intensity of the object is different from the image intensity of the background.

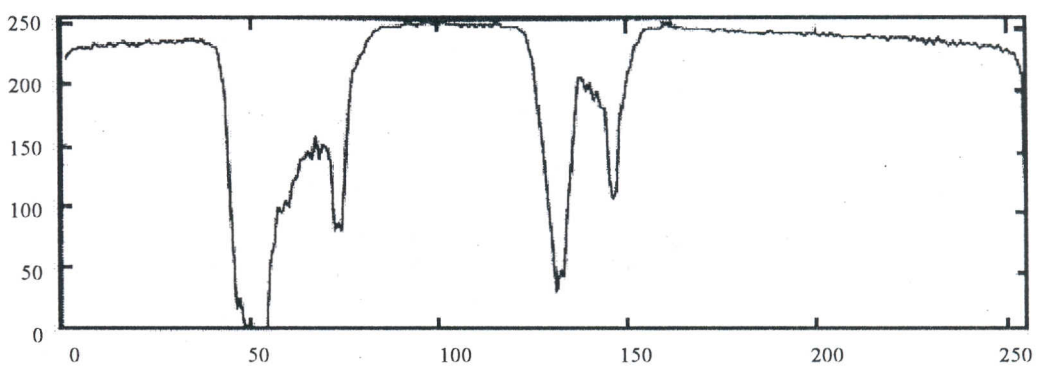
The profile of an ideal step edges are displayed in Figure 5-2. By definition, an edge is a significant local change in the image intensity. The plot shows step changes in image intensity that are edges because the changes are significant and local. The plot also shows changes that are not edges, because they violate part of the definition. The changes due to noise are not edges even though the changes are local, because the changes are not significant. The changes due to shading, such as the ramp on the right side of the plot, are not edges even though the changes are significant, because the

changes are not local. Real images are very noisy. It is difficult to develop an edge detection operator that reliably finds step edges and immune to noise.

The coordinates of an edge point may be the integer row and column indices of the pixel where the edges was detected, or the coordinates of the edge location at subpixel resolution. The edge coordinates may be in the coordinate system of the original image, but more likely are in the coordinate system of the image produced by the edge detection filter since filtering may translate or scale imaging. An edge fragment may be conceptualized as a small line segment about the size of a pixel, or as a point with an orientation attribute. The term *edge* is commonly used for either edge points or edge fragments.



(a)



(b)

Figure 5-2 (a) The top half of the connecting rod image. (b) The profile of an ideal step change in image intensity is plotted to show that the edges are not perfectly sharp and the image is corrupted by noise. The plot is horizontal slice through the circular portion of the connecting rod corresponding to the bottom edge of the partial rod image shown above.

The edge set produced by an edge detector has been partitioned into two subset by Jain et al. (Jain et al., 1995) : correct edges, which correspond to edges in the scene, and false edges, which not correspond to edges in the scene. A third set of edges can be defined as those edges in the scene that should have been detected. This is the set of

missing edges. The false edges are called false positives, and the missing edges are called false negatives.

The difference between edge linking and edge following is that edge linking takes as input and unordered set of edges produced by an edge detector and forms an ordered list of edges. Edge following takes as input an image and produces an ordered list of edges. Edge detection uses local information to decide if a pixel is an edge, while edge following can use global information.

5.2 Gradient

Edge detection is essentially the operation of detecting significant local changes in an image. In one dimension, a step edge is associated with a local peak in the first derivative. The gradient is a measure of change in a function, and image can be considered to be an array of samples of some continuous function of image intensity. By analogy, significant changes in the gray values in an image can be detected by using a discrete approximation to the gradient. The gradient is the two-dimensional equivalent of the derivative and is defined as the *vector*

$$G[f(x,y)] = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \tag{5-1}$$

There are two important properties associated with the gradient: (1) the vector $G[f(x,y)]$ points in the direction of the maximum rate of increase of the function $f(x,y)$, and (2) the magnitude of the gradient, given by

$$G[f(x,y)] = \sqrt{G_x^2 + G_y^2} \tag{5-2}$$

equals the maximum rate of increase of $f(x,y)$ per unit distance in the direction G. It is common practice, however, to approximate the gradient magnitude by absolute values:

$$G[f(x,y)] \approx |G_x| + |G_y| \tag{5-3}$$

or

$$G[f(x,y)] \approx \max(|G_x|, |G_y|). \tag{5-4}$$

From vector analysis, the *direction* of the gradient is defined as:

$$\alpha(x,y) = \tan^{-1}\left(\frac{G_x}{G_y}\right) \tag{5-5}$$

where the angle α is measured with respect to the x axis.

Note that the magnitude of the gradient is actually independent of the direction of the edge. Such operators are called *isotropic operators.*

5.2.1 Numerical Approximation

For digital images, the derivatives in Equation 5-1 are approximated by differences. The simplest gradient approximation is

$$G = f[i, j+1] - f[i, j] \tag{5-6}$$

$$G \cong f[i, j] - f[i+1, j]. \tag{5-7}$$

Remember that j corresponds to the x direction and i to the negative y direction. These can be implemented with simple convolution masks as shown below:

$$G_x = \begin{bmatrix} -1 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{5-8}$$

When computing an approximation to the gradient, it is critical that the x and y partial derivatives be computed at exactly the same position in space. However, using the above approximations, G is actually the approximation to the gradient at the interpolated point $[i, j + -]$ and G_y at $[i + y, j]$. For this reason, 2 x 2 first differences, rather than 2 x 1 and 1 x 2 masks, are often used for the x and y partial derivatives:

$$G_x = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix} \tag{5-9}$$

Now, the positions about which the gradients in the x and y directions are calculated are the same. This point lies between all four pixels in the 2 x 2 neighborhood at the interpolated point $[i+1/2, j+1/2]$. This fact may lead to some confusion. Therefore, an alternative approach is to use a 3 x 3 neighborhood and calculate the gradient about the center pixel.

5.2.2 Steps in Edge Detection

Algorithms for edge detection contain three steps:

Filtering: Since gradient computation based on intensity values of only two points are susceptible to noise and other vagaries in discrete computations, filtering is commonly used to improve the performance of an edge detector with respect to noise. However, there is a trade-off between edge strength and noise reduction. More filtering to reduce noise results in a loss of edge strength.

Enhancement: In order to facilitate the detection of edges, it is essential to determine changes in intensity in the neighborhood of a point. Enhancement emphasizes pixels where there is a significant change in local intensity values and is usually performed by computing the gradient magnitude.

Detection: We only want points with strong edge content. However, many points in an image have a nonzero value for the gradient. and not all of these points are edges for a particular application. Therefore, some method should be used to determine which points are edge points. Frequently, thresholding provides the criterion used for detection.

Many edge detection algorithms include a fourth step:

Localization: The location of the edge can be estimated with subpixel resolution if required for the application. The edge orientation can also be estimated.

It is important to note that detection merely indicates that, an edge is present near a pixel in an image, but does not necessarily provide an accurate estimate of edge location or orientation. The errors in edge detection are errors of misclassification: false edges and missing edges. The errors in edge estimation are modeled by probability distributions for the location and orientation estimates. We distinguish between edge detection and estimation because these steps are performed by different calculations and have different error models.

Many edge detectors have been developed in the last two decades. Here we will discuss some commonly used edge detectors. As will be clear, edge detectors differ in use of the computational approach in one or more of the above three steps.

5.2.3 Roberts Operator

The Roberts cross operator provides a simple approximation to the gradient magnitude:

$$G[f[i,j]] = |f|i,j| - f|i+1,j+1|| + |f|i+1,j| - f|i,j+1||. \quad (5\text{-}10)$$

Using convolution masks, this becomes

$$G[f[i,j]] = |G_x| + |G_y| \quad (5\text{-}11)$$

where G_x and G_y are calculated using the following masks:

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad G_y = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad (5\text{-}12)$$

As with the previous 2 x 2 gradient, operator, the differences are, computed at, the interpolated point $[i+1/2, j+1/2]$. The Roberts operator is an approximation to the continuous gradient at that point and not at, the point [*i, j*] as might the, expected.

5.2.4 Sobel Operator

As mentioned previously, a way to avoid having the gradient calculated about an interpolated point between pixels is to use a 3 x 3 neighborhood for the gradient calculations. Consider the arrangement of pixels about the pixel [i, j] shown in Figure 5-3. The Sobel operator is the magnitude of the gradient computed by

$$M = \sqrt{s_x^2 + s_y^2}, \quad (5\text{-}13)$$

where the partial derivatives are computed by

$$s_x = (a_2 + ca_3 + a_4) - (a_0 + ca_7 + a_6) \quad (5\text{-}14)$$

$$s_y = (a_0 + ca_1 + a_2) - (a_6 + ca_5 + a_4) \quad (5\text{-}15)$$

with the constant c=2.

Like the other gradient operators, s_x and s_y can be implemented using convolution masks:

$$s_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad s_Y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad (5\text{-}16)$$

Note that this operator places an emphasis on pixels that are closer to the center of the mask. The Sobel operator is one of the most commonly used edge detector. Hence, it can be found so many issues in literature about this operator. Picton (Picton, 1984) has studied on this subject, to be optimized the operator, can be given as an example to emphasize its popularity.

$$\begin{bmatrix} a_0 & a_1 & a_2 \\ a_7 & [i,j] & a_3 \\ a_6 & a_5 & a_4 \end{bmatrix}$$

Figure 5-3: The labeling of neighborhood pixels used to explain the Sobel and Prewitt operators .

5.2.5 Prewitt Operator

The Prewitt operator uses the same equations as the Sobel operator, except that the constant c = 1. Therefore:

$$s_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad s_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \qquad (5\text{-}17)$$

Note that, unlike the Sobel operator, this operator does not place any emphasis on pixels that are closer to the center of the masks.

5.3 Second Derivative Operators

The edge detectors discussed earlier computed the first derivative and, if it was above a threshold, the presence of an edge point was assumed. This results in detection of too many edge points. (Notice the thick lines after thresholding in Figures 5-4 to 5-7.) A better approach would be to find only the points that have local maxima in gradient values and consider them edge points, as shown in Figure 5-8. This means that at edge points, there will be a peak in the first derivative and, equivalently, there will be a zero

crossing in the second derivative. Thus, edge points may be detected by finding the zero crossings of the second derivative of the image intensity.

There are two operators in two dimensions that correspond to the second derivative: the Laplacian and second directional derivative.

5.3.1 Laplacian Operator

The second derivative of a smoothed step edge is a function that crosses zero at the location of the edge (see Figure 5-8). The Laplacian is the two-dimensional equivalent of the second derivative. The formula for the Laplacian of a function *f (x, y)* is with the constant c = 2.

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{5-18}$$

The second derivatives along the x and *y* directions are approximated using difference equations:

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial G_x}{\partial x} \tag{5-19}$$

$$= \frac{\partial\left(f[i,j+1] - f[i,j]\right)}{\partial x} \tag{5-20}$$

$$= \frac{\partial f[i,j+1]}{\partial x} - \frac{\partial f[i,j]}{\partial x} \tag{5-21}$$

$$= \left(f[i,j+2] - f[i,j+1]\right) - \left(f[i,j+1] - f[i,j]\right) \tag{5-22}$$

$$= f[i,j+2] - 2f[i,j] + f[i,j] \tag{5-23}$$

However, this approximation is centered about the pixel [i, *j* + 1]. Therefore, by replacing *j* with *j* – 1, we obtain

$$\frac{\partial^2 f}{\partial x^2} = f[i,j+1] - 2f[i,j] + f[i,j-1] \tag{5-24}$$

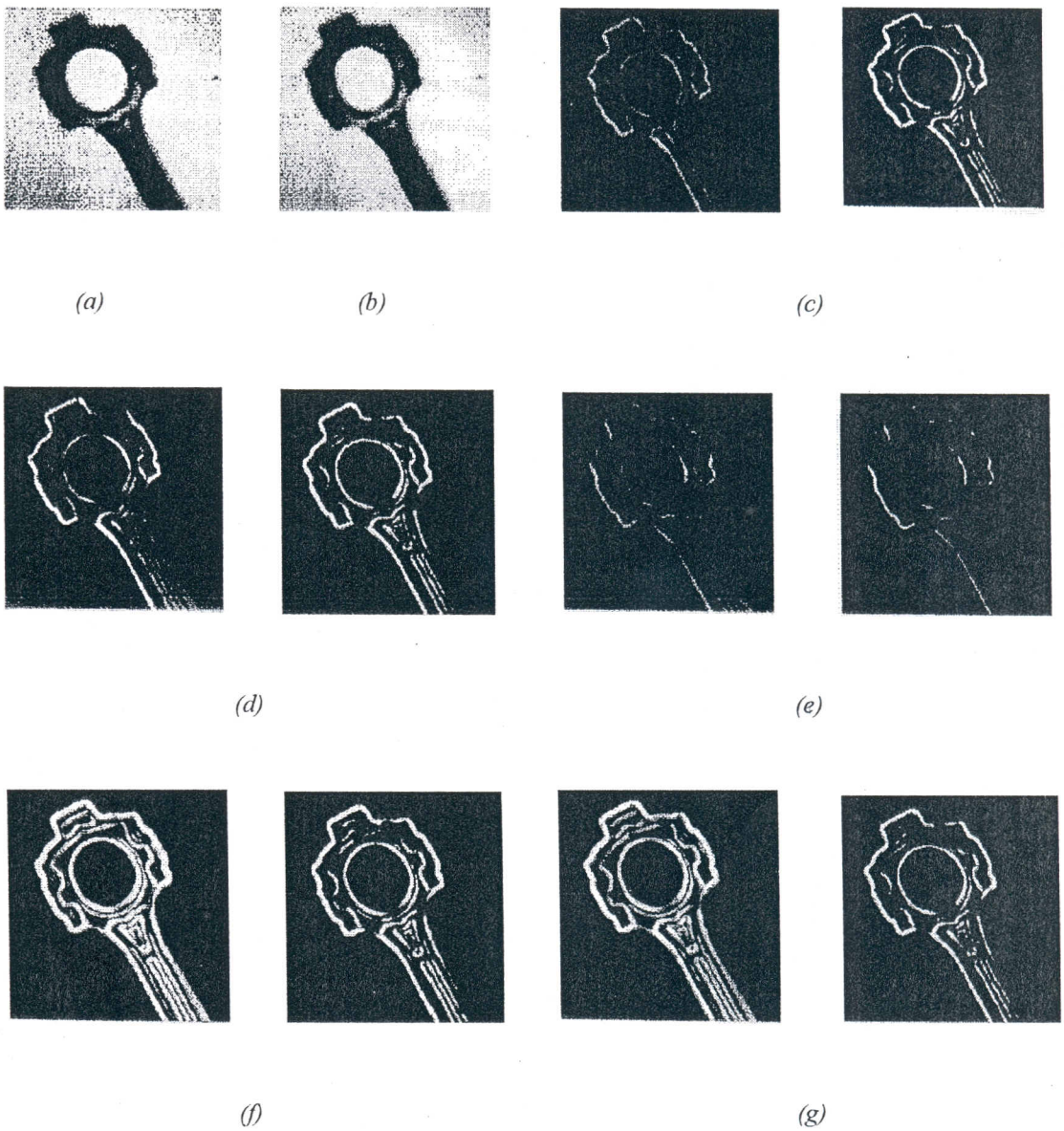
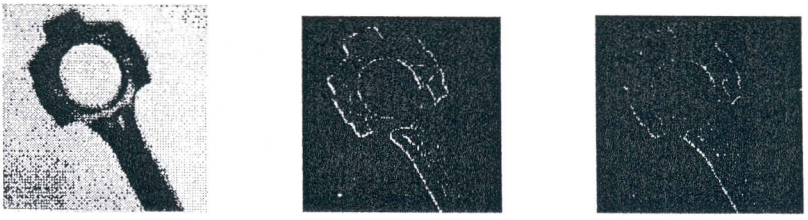
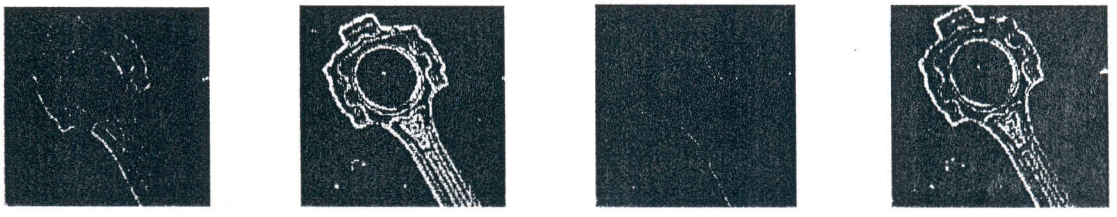


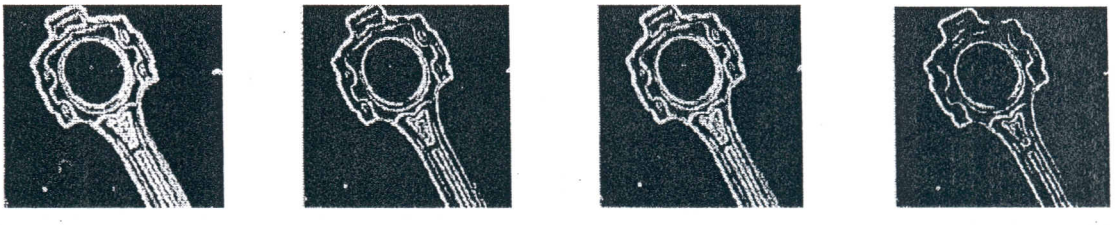
Figure 5-4 A comparison of various edge detectors. (a) Original image. (b) Filtered image (c) Simple gradient using 1 x 2 and 2 x 1 masks, T=32. (d) Gradient using 2 x 2 masks, T=64. (e) Roberts cross operator, T=64. (f) Sobel operator, T=225. (g) Prewitt operator, T=225.



(a) (b)

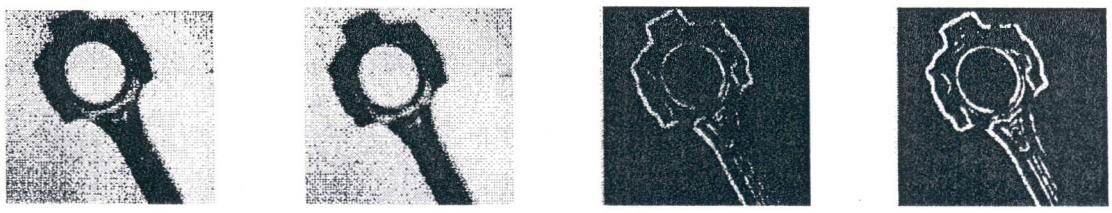


(c) (d)

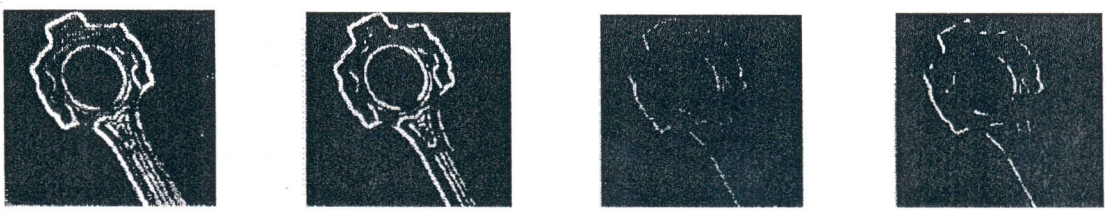


(e) (f)

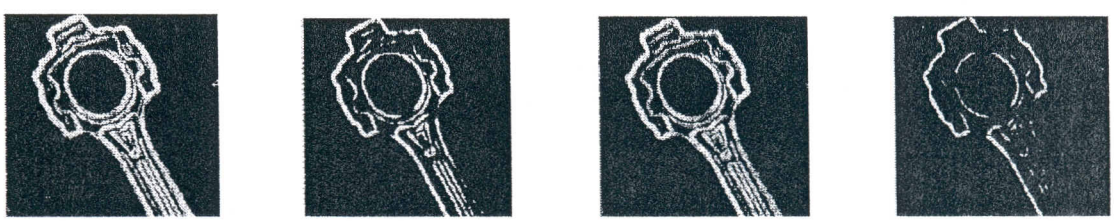
Figure 5-5 A comparison of various edge detectors without filtering. (a) Original image (b) Filtered image (c) Simple gradient using 1 x 2 and 2 x 1 masks, T=32. (d) Gradient using 2 x 2 masks, T=64. (e) Roberts cross operator, T=64. (f) Sobel operator, T=225. (g) Prewitt operator, T=225.



(a) (b) (c)

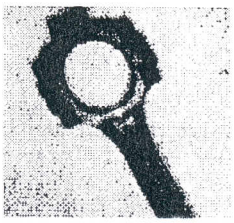


(d) (e)

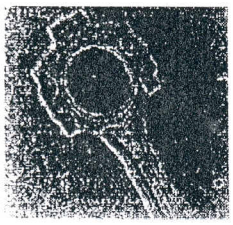


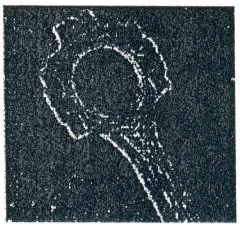
(f) (g)

Figure 5-6 A comparison of various edge detectors on a noisy image . (a) Original image (b) Filtered image (c) Simple gradient using 1 x 2 and 2 x 1 masks, T=32. (d) Gradient using 2 x 2 masks, T=64. (e) Roberts cross operator, T=64. (f) Sobel operator, T=225. (g) Prewitt operator, T=225.

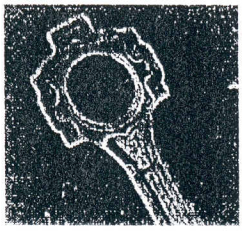


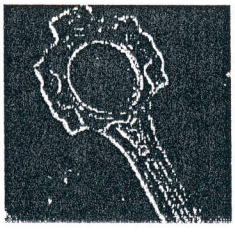
(a)

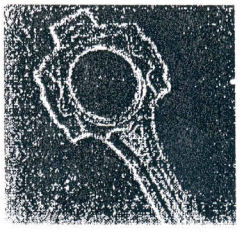


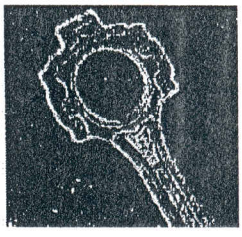


(b)



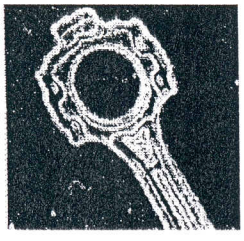


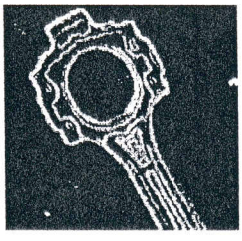


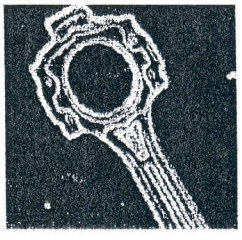


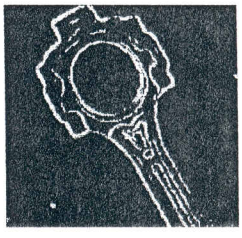
(c)

(d)









(e)

(f)

Figure 5-7 A comparison of various edge detectors on a noisy image without filtering . (a) Original image (b) Filtered image (c) Simple gradient using 1 x 2 and 2 x 1 masks, T=32. (d) Gradient using 2 x 2 masks, T=64. (e) Roberts cross operator, T=64. (f) Sobel operator, T=225. (g) Prewitt operator, T=225.

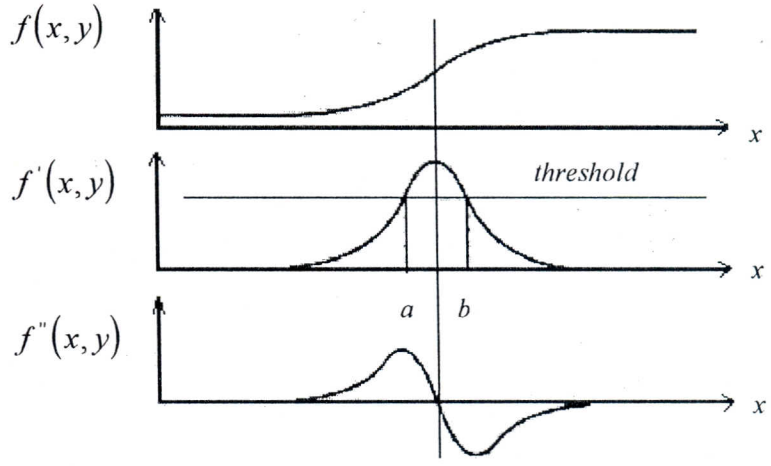


Figure 5-8: If a threshold is used for detection of edges, all points between a and b will be marked as edge pixels. However, by removing points that are not a local maximum in the first derivative, edges can be detected more accurately. This local maximum in the first derivative corresponds to a zero crossing in the second derivative.

which is the desired approximation to the second partial derivative centered about [*i*, *j*]. Similarly,

$$\frac{\partial^2 f}{\partial y^2} = f[i+1,j] - 2f[i,j] + f[i-1,j] \qquad (5\text{-}25)$$

By combining these two equations into a single operator, the following mask can be used to approximate the Laplacian:

$$\nabla^2 \approx \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad (5\text{-}26)$$

Sometimes it is desired to give more weight to the center pixels in the neighborhood. An approximation to the Laplacian which does this is

$$\nabla^2 \approx \begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix} \qquad (5\text{-}27)$$

The Laplacian operator signals the presence of an edge when the output of the operator makes a transition through zero. Trivial zeros (uniform zero regions) are ignored. In principle, the zero crossing location can be estimated to subpixel resolution using linear interpolation. but the result may be inaccurate due to noise.

Consider the example shown in Figure 5-9. This figure shows the result of the Laplacian on an image with a. simple step edge. A single row of the resulting image is:

$$\begin{bmatrix} 0 & 0 & 0 & 6 & -6 & 0 & 0 & 0 \end{bmatrix}$$

In this example, the zero crossing, corresponding to the edge in the original image, lies halfway between the two center pixels. The edge should be marked at either the pixel to the left, or the pixel to the right of the edge, as long as it is marked consistently throughout the image. In most, cases, however, the zero crossing rarely lies exactly between two pixels, and the actual edge

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 8 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 8 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 8 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 8 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 8 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 8 & 8 & 8 & 8 & 8 \end{bmatrix}$$

A sample image containing a vertical step edge

$$\begin{bmatrix} 0 & 0 & 0 & 6 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & -6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 6 & -6 & 0 & 0 & 0 \end{bmatrix}$$

Figure 5-9 The response of the Laplacian to a vertical step edge

location must be determined by interpolating the pixel values on either side of the zero crossing.

Now consider the example in Figure 5-10. This figure shows the response of the Laplacian to a ramp edge. A single row of the output of the Laplacian is

$$\begin{bmatrix} 0 & 0 & 0 & 3 & 0 & -3 & 0 & 0 \end{bmatrix}$$

The zero crossing directly corresponds to a pixel in the image. Again, this is an ideal situation, and the actual edge location should be determined by interpolation.

5.3.2 Second Directional Derivative

The second directional derivative is the second derivative computed in the direction of the gradient. The operator is implemented using the formula

$$\frac{\partial^2}{\partial n^2} = \frac{f_x^2 f_{xx} + 2 f_x f_y f_{xy} + f_y^2 f_{yy}}{f_x^2 + f_y^2} \qquad (5\text{-}28)$$

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \\ 2 & 2 & 2 & 2 & 2 & 5 & 8 & 8 & 8 & 8 \end{bmatrix}$$

A sample image containing a vertical ramp edge.

$$\begin{bmatrix} 0 & 0 & 0 & 3 & 0 & -3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & -3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & -3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & -3 & 0 & 0 \end{bmatrix}$$

Figure 5-10: The response of the Laplacian to a vertical ramp edge.

The Laplacian and second directional derivative operators are not used frequently in machine vision since any operator involving two derivatives is affected by noise more than an operator involving a single derivative. Even very small local peaks in the first derivative will result in zero crossings in the second derivative. To avoid the effect of noise, powerful filtering methods must be used. In the following section, we discuss an approach which combines Gaussian filtering with the second derivative for edge detection.

5.4 Laplacian of Gaussian

As mentioned above, edge points detected by finding the zero crossings of the second derivative of the image intensity are very sensitive to noise. Therefore, it is desirable to filter out the noise before edge enhancement. To do this, the *Laplacian of Gaussian* (LoG) combines Gaussian filtering with the Laplacian for edge detection.

The fundamental characteristics of the Laplacian of Gaussian edge detector are

1. The smoothing filter is a Gaussian.
2. The enhancement step is the second derivative (Laplacian in two dimensions).
3. The detection criterion is the presence of a zero crossing in the second derivative with a corresponding large peak in the first derivative.
4. The edge location can be estimated with subpixel resolution using linear interpolation.

In this approach, an image should first be convoluted with a Gaussian filter. This step smoothes an image and reduces noise. Isolated noise points and small structures will be filtered out. Since the smoothing will result in *spreading* of edges, the edge detector considers as edges only those pixels that have locally maximum gradient. This is achieved by using zero crossings of the second derivative. The Laplacian is used as the approximation of the second derivative in 2-D because it is an isotropic operator. To avoid detection of insignificant edges, only the zero crossings whose corresponding first derivative is above some threshold are selected as edge points.

The output of the LoG operator, h(x *y),* is obtained by the convolution operation

$$h(x,y) = \nabla^2\left[\left(g(x,y) * f(x,y)\right)\right]. \tag{5-29}$$

Using the derivative rule for convolution,

$$h(x,y) = \left[\nabla^2 g(x,y)\right] * f(x,y), \tag{5-30}$$

where

$$\nabla^2 g(x,y) = \left(\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right) e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{5-31}$$

is commonly called the *Mexican hat* operator (shown in Figure 5-11). Thus, the following two methods are mathematically equivalent:

1. Convolute the image with a Gaussian smoothing filter and compute the Laplacian of the result.
2. Convolute the image with the linear filter that is the Laplacian of the Gaussian filter.

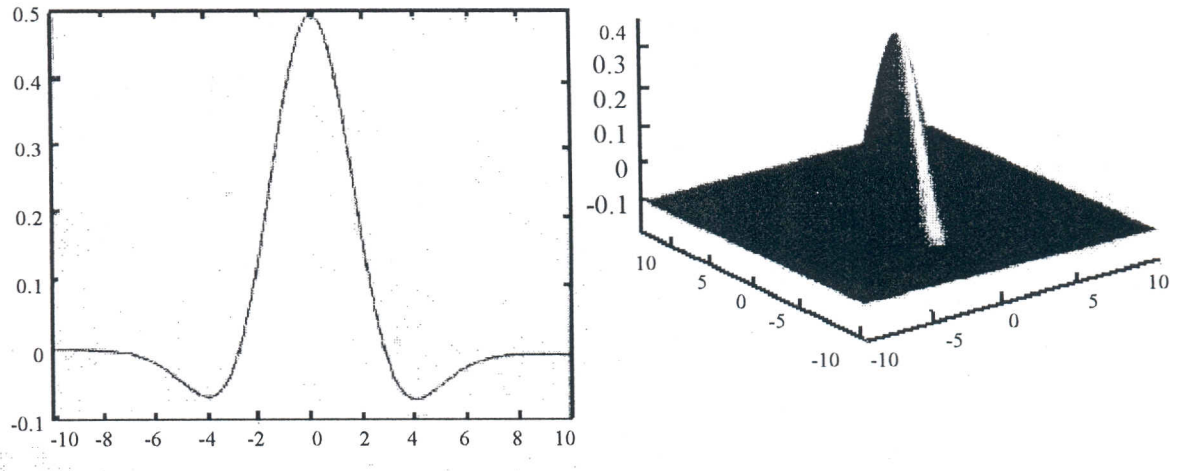


Figure 5-11 The inverted Laplacian of Gaussian function, σ=21 in one and two dimensions.

Typical masks to directly implement the LoG are given in Figure 5-12. Figure 5-13 shows the result of applying the Laplacian of Gaussian operator and detection of zero crossings. For a discussion on efficient methods to implement the Laplacian of Gaussian.

Filtering (usually smoothing), enhancement, and detection were the three steps in edge detection. This is still true for edge detection using the Laplacian of Gaussian. Smoothing is performed with a Gaussian filter, enhancement is done by transforming edges into zero crossings, and detection is done by detecting the zero crossings.

It can be shown that the slope of the zero crossing depends on the contrast of the change in image intensity across the edge. The problem of combining edges obtained by applying different-size operators to images remains. In the above approach, edges at a particular resolution are obtained. To obtain real edges in an image, it may be necessary to combine information from operators at several filter sizes.

5 x 5 Laplacian of Gaussian mask

$$\begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

17 x 17 Laplacian of Gaussian mask

0	0	0	0	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	0
0	0	0	0	−1	−1	−1	−1	−1	−1	−1	−1	−1	0	0	0	0
0	0	−1	−1	−1	−2	−3	−3	−3	−3	−3	−2	−1	−1	−1	0	0
0	0	−1	−1	−2	−3	−3	−3	−3	−3	−3	−3	−2	−1	−1	0	0
0	−1	−1	−2	−3	−3	−3	−2	−3	−2	−3	−3	−3	−2	−1	−1	0
0	−1	−2	−3	−3	−3	0	2	4	2	0	−3	−3	−3	−2	−1	0
−1	−1	−3	−3	−3	0	4	10	12	10	4	0	−3	−3	−3	−1	−1
−1	−1	−3	−3	−2	2	10	18	21	18	10	2	−2	−3	−3	−1	−1
−1	−1	−3	−3	−3	4	12	21	24	21	12	4	−3	−3	−3	−1	−1
−1	−1	−3	−3	−2	2	10	18	21	18	10	2	−2	−3	−3	−1	−1
−1	−1	−3	−3	−3	0	4	10	12	10	4	0	−3	−3	−3	−1	−1
0	−1	−1	−3	−3	−3	0	2	4	2	0	−3	−3	−3	−2	−1	0
0	−1	−1	−2	−3	−3	−3	−2	−3	−2	−3	−3	−3	−2	−1	−1	0
0	0	−1	−1	−2	−3	−3	−3	−3	−3	−3	−3	−2	−1	−1	0	0
0	0	0	0	−1	−1	−1	−1	−1	−1	−1	−1	−1	0	0	0	0
0	0	0	0	0	0	−1	−1	−1	−1	−1	0	0	0	0	0	0

Figure 5-12: Some useful Laplacian of Gaussinan masks

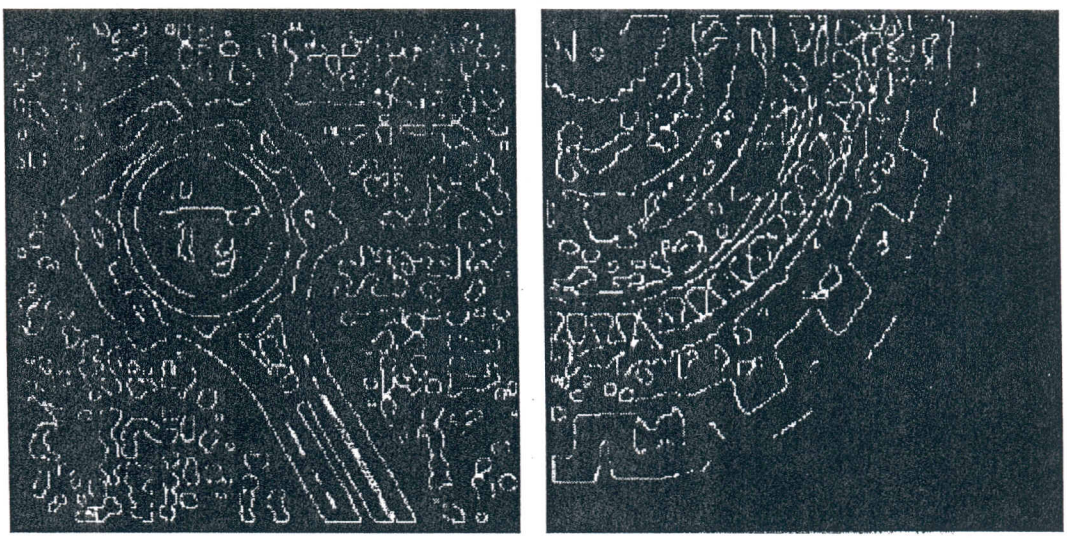


Figure 5-13: Results of Laplacian of Gaussian edge detector.

5.5 Gaussian Edge Detection

The essential idea in detecting step edges is to find points in the sampled image that have locally large gradient magnitudes. Much of the research work in step edge detection is devoted to finding numerical approximations to the gradient that are suitable for use with real images. The step edges in real images are not perfectly sharp since the edges are smoothed by the low-pass filtering inherent in the optics of the camera lens and the bandwidth limitations in the camera electronics. The images are also severely corrupted by noise from the camera and unwanted detail in the scene. An approximation to the image gradient must be able to satisfy two conflicting requirements: (1) the approximation must suppress the effects of noise, and (2) the approximation must locate the edge as accurately as possible. There is a trade-off between noise suppression and localization. An edge detection operator can reduce noise by smoothing the image, but this will add uncertainty to the location of the edge; or the operator can have greater sensitivity to the presence of edges, but this will increase the sensitivity of the operator to noise. The type of linear operator that provides the best compromise between noise immunity and localization, while retaining the advantages of Gaussian filtering, is the first derivative of a Gaussian. This operator corresponds to smoothing an image with a Gaussian function and then computing the gradient. The gradient can be numerically approximated by using the standard finite-difference approximation for the first partial derivatives in the z and y directions. The operator that is the combination of a Gaussian smoothing filter and a gradient approximation is not rotationally symmetric. The operator is symmetric along the edge and antisymmetric perpendicular to the edge (along the line of the gradient). This means that the operator is sensitive to the edge in the direction of steepest change, but is insensitive to the edge and acts as a smoothing operator in the direction along the edge.

5.6 Canny Edge Detector

The Canny edge detector is the first derivative of a Gaussian and closely approximates the operator that optimizes the product of signal-to-noise ratio and

localization. The Canny edge detection algorithm is summarized by the following notation. Let $I[i, j]$ denote the image. The result from convoluting the image with a Gaussian smoothing filter using separable filtering is an array of smoothed data,

$$S[i,j] = G[i,j;\sigma] * I[i,j] \tag{5-32}$$

where σ is the spread of the Gaussian and controls the degree of smoothing.

The gradient of the smoothed array $S[i, j]$ can be computed using the 2 x 2 first-difference approximations to produce two arrays $P[i, j]$ and $Q[i, j]$ for the x and y partial derivatives:

$$P[i,j] \approx \left(S[i,j+1] - S[i,j] + S[i+1,j+1] - S[i+1,j]\right)/2 \tag{5-33}$$

$$Q[i,j] \approx \left(S[i,j] - S[i+1,j] + S[i,j+1] - S[i+1,j+1]\right)/2 \tag{5-34}$$

The finite differences are averaged over the 2 x 2 square so that the x and y partial derivatives are computed at the same point in the image. The magnitude and orientation of the gradient can be computed from the standard formulas for rectangular-to-polar conversion:

$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2} \tag{5-35}$$

$$\theta[i,j] = \arctan\left(Q[i,j], P[i,j]\right), \tag{5-36}$$

where the arctan function takes two arguments and generates an angle over the entire circle of possible directions listed in algorithm 5.1. These functions must be computed efficiently, preferably without using floating-point arithmetic. It is possible to compute the gradient magnitude and orientation from the partial derivatives by table lookup. The arctangent can be computed using mostly fixed-point arithmetic with a few essential floating-point calculations performed in software using integer and fixed-point arithmetic.

5.7 Nonmaxima Suppression

The magnitude image array $M[i, j]$ will have large values where the image gradient is large, but this is not sufficient to identify the edges, since the problem of finding locations in the image array where there is rapid change has merely been transformed into the problem of finding locations in the magnitude array $M[i, j]$ that are local maxima. To identify edges, the broad ridges in the magnitude array must be thinned

so that only the magnitudes at the points of greatest local change remain. This process is called nonmaxima suppression, which in this case results in thinned edges.

Nonmaxima suppression thins the ridges of gradient magnitude in $M[i, j]$ by suppressing all values along the line of the gradient that are not peak values of a ridge. The algorithm begins by reducing the angle of the gradient $\theta[i, j]$ to one of the four sectors shown in Figure 5-14,

$$\xi[\mathrm{i}, \mathrm{j}] = \mathrm{Sector}(\theta[\mathrm{i}, \mathrm{j}]). \tag{5-37}$$

The algorithm passes a 3 x 3 neighborhood across the magnitude array $M[i, j]$, At each point, the center element $M[i, j]$ of the neighborhood is compared with its two neighbors along the line of the gradient given by the sector value $\xi[\mathrm{i}, \mathrm{j}]$ at the center of the neighborhood. If the magnitude array value $M[i, j]$ at the center is not greater than both of the neighbor magnitudes along the gradient line, then $M[i, j]$ is set to zero. This process thins the broad ridges of gradient magnitude in $M[i, j]$ into ridges that are only one pixel wide. The values for the height of the ridge are retained in the nonmaxima-suppressed magnitude. Let

$$N[i, j] = \mathrm{nms}(\mathrm{M}[\mathrm{i}, \mathrm{j}], ([\mathrm{i}, \mathrm{j}]) \tag{5-38}$$

denotes that the process of nonmaxima suppression. The nonzero values in $N[i, j]$ correspond to the amount of contrast at a step change in the image intensity. In spite of the smoothing performed as the first step in edge detection, the nonmaxima-suppressed magnitude image $N[i, j]$ will contain many false edge fragments caused by noise and fine texture. The contrast of the false edge fragments is small.

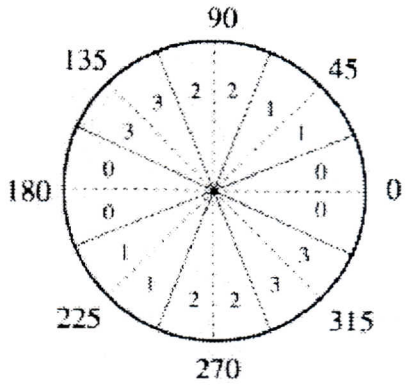


Figure 5-14: The partition of the possible gradient orientations into sectors for nonmaxima suppresion is shown. There are four sectors, numbered 0 to 3, corresponding to the four possible combinations of elements in a 3 x 3 neighborhood that a line must pass through as it passes through the center of the neighborhood. The divisions of the circle of possible gradient line orientations are labeled in degrees.

5.8 Thresholding

The typical procedure used to reduce the number of false edge fragments in the nonmaxima-suppressed gradient magnitude is to apply a threshold to $N[i, j]$. All values below the threshold are changed to zero. The result of applying a threshold to the nonmaxima-suppressed magnitude is an array of the edges detected in the image $I[i, j]$. There will still be some false edges because the threshold τ was too low (false positives), and portions of actual contours may be missing (false negatives) due to softening of the edge contrast by shadows or because the threshold τ was too high. Selecting the proper threshold is difficult and involves some trial and error. A more effective thresholding scheme uses two thresholds.

The double thresholding algorithm takes the nonmaxima-suppressed image, $N[i, j]$, and applies two thresholds τ_1 and τ_2 with, $\tau_2 \approx 2\tau_1$ to produce two thresholded edge images $T_1[i, j]$ and T_2 $[i, j]$. Since image T_2 was formed with a higher threshold, it will contain fewer false edges; but T_2 may have gaps in the contours (too many false negatives). The double thresholding algorithm links the edges in T_2 into contours. When it reaches the end of a contour, the algorithm looks in T_2 at the locations of the 8-neighbors for edges that can be linked to the contour. The algorithm continues to gather edges from T_1 until the gap has been bridged to an edge in T_2. The algorithm performs

edge linking as a by-product of thresholding and resolves some of the problems with choosing a threshold. The Canny edge detection algorithm is outlined in Algorithm 5.1.

5.9 Canny Edge Detection Algorithm

1. Smooth the image with a Gaussian filter.

2. Compute the gradient magnitude and orientation using finite-difference approximations for the partial derivatives.

3. Apply nonmaxima suppression to the gradient magnitude.

4. Use the double thresholding algorithm to detect and link edges.

Chapter 6

RECONSTRUCTION OF X-RAY IMAGES

6.1 Introduction

In medical environment, the x-ray images are very much helpful to the physician in the process of diagnosis and medical treatment. Although it provides to detect human body's invisible parts, blurring and uneven illumination is always occurs. This problem is partially solved by the physicians by lighting the x-rays. Generally, this method is working properly except some cases such as Vesico Ureteral Reflux disease. This may cause loss of some meaningful part of information and failure in diagnosis process. In order to avoid decrease such errors, some computational methods has been developed by means of image processing. Due to its importance, reconstruction of vesicle ureteral reflux disease x-ray images has been chosen as an object of this study.

For analyzing the problem properly, about the disease discussed above, a brief description of it must be done. Aktuğ (Aktuğ, 1995) has briefly described the disease as: *The vesico ureteral reflux is the declining of the complex vesicle ureteral valve mechanism which lets the urine enter into the bladder but it stops the urine go backwards.*

Malfunctioning of vesico ureteral reflux system causes some deformations on calyxes which are the parts of kidney and located inside of it. The disease appears as shape changing on calyxes. Level of changing is related with grade of the disease. This can be seen clearly in Figure 6-2 which has been taken from King's (King, 1994) study. In Figure 6-1 the cross section of kidney is shown.

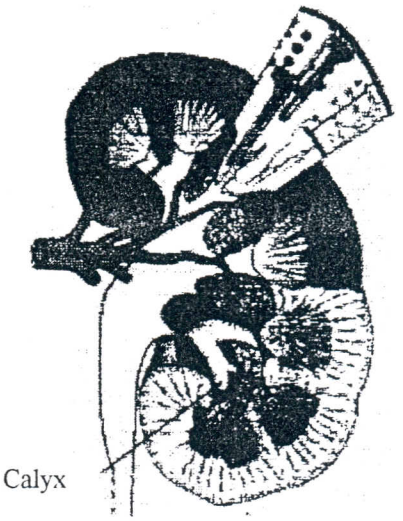


Figure 6-1 Cross section of kidney

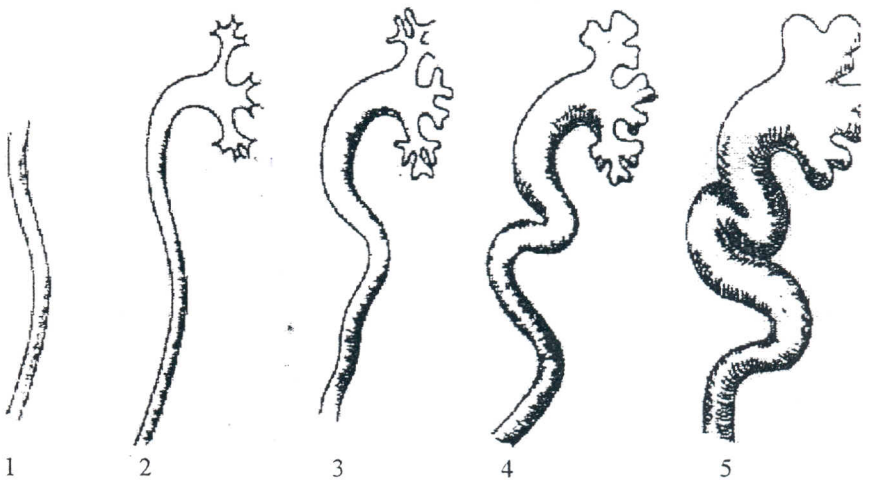


Figure 6-2 Grades of reflux as seen on the voiding cystography.

6.2 Digitizing of X-rays

The object of the study is reconstruction of vesico ureteral reflux disease x-ray images as mentioned above. In medical environment, disease is detected from x-rays which is produced from human's body using the technique voiding cystography. In order to process this materials x-rays must be digitized. Digitizing by using conventional scanners performance is not sufficient because of low of illumination level. Adding some external light sources has solved the problem. Figure 6-3 is illustrating this process.

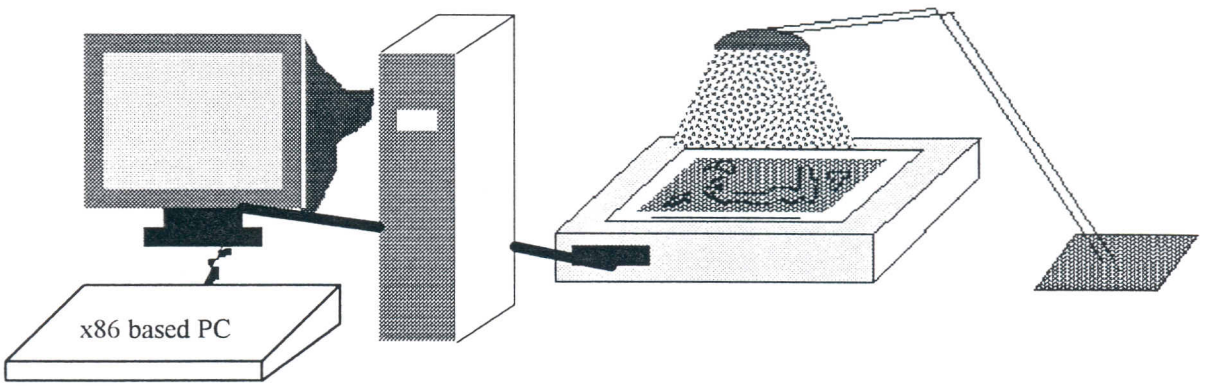
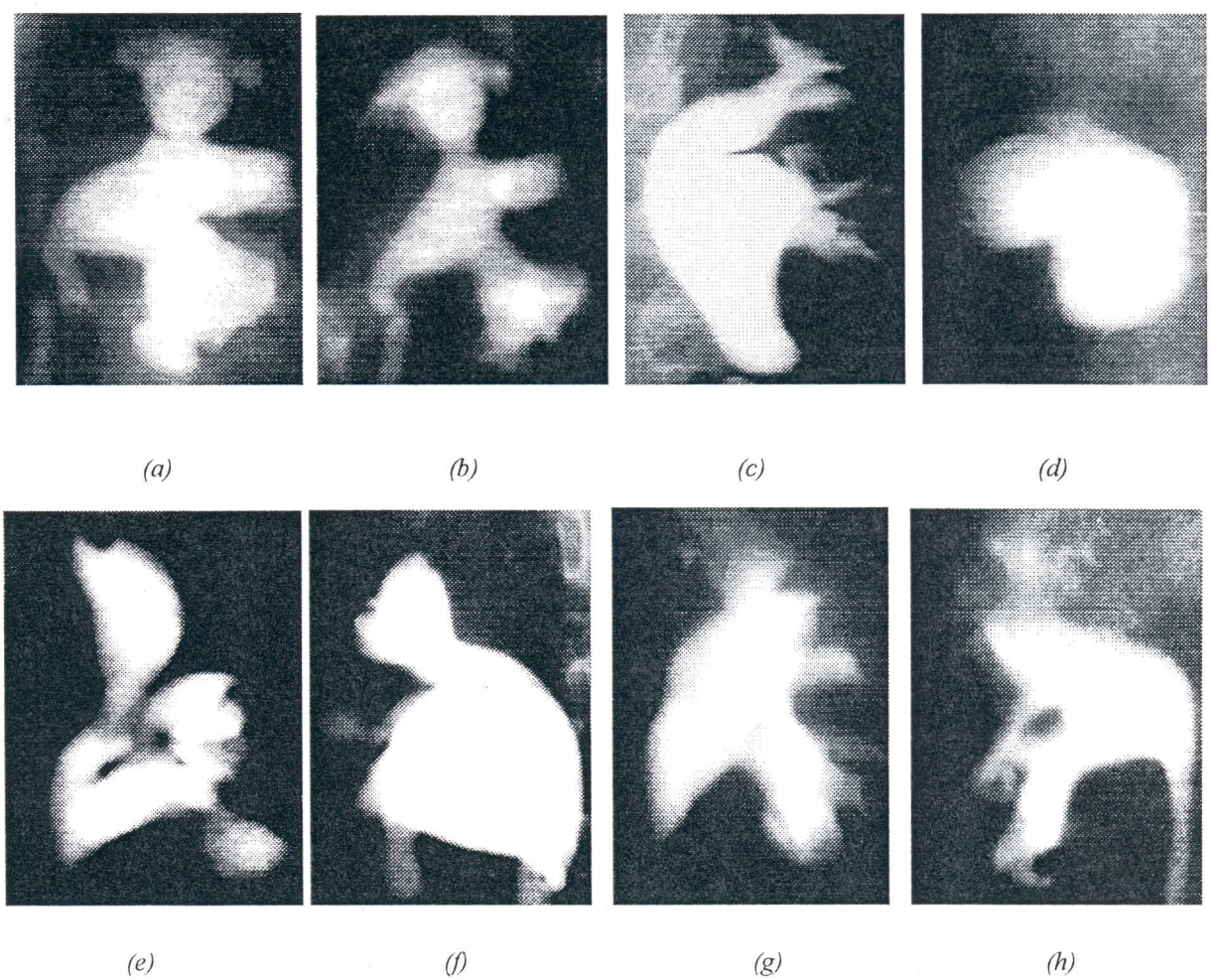
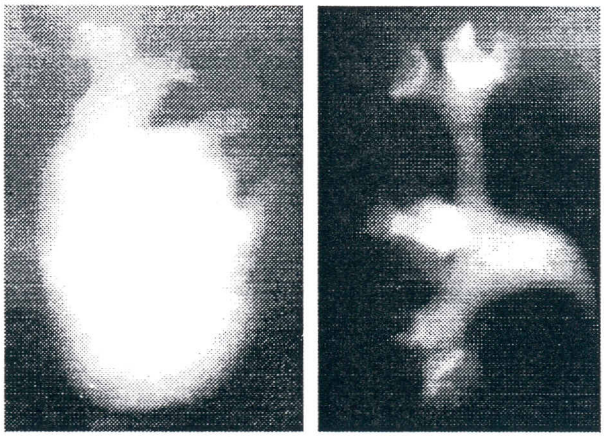


Figure 6-3 X-ray digitizing system

The processed x-rays in the study are supported by Dokuz Eylül University , Faculty of Medicine, Pediatric Surgery Department. In Figure 6-4, it is shown that the digitized form of X-rays.





(i) (j)

Figure 6-4 Digitized form of x-rays supplied from Dokuz Eylül University

All the digitized images are shown above has 8-bit depth. In other words, all of them are one of the combination of 256 different gray levels. Therefore, all of procedures that will be explained below has been performed 8-bit based image processing.

6.3 Preprocessing of Images

At the beginning of this chapter, we have discussed the difficulties of processing of medical images. As it is mentioned before, this is caused by uneven illuminations. In this section, some image reconstruction techniques for preparing images for further stages are explained.

There has not been only four different stages in preprocessing of medical images, but also all stages use output of previous stage as input. It consists of histogram modification, contrast stretching, exponential filtering, and thresholding.

6.3.1 Histogram Modification

In an image processing context, the histogram of an image normally refers to a histogram of the pixel intensity values. This histogram is a graph showing the number of pixels in an image at each different intensity value found in that image. For an 8-bit greyscale image there are 256 different possible intensities, and so the histogram will

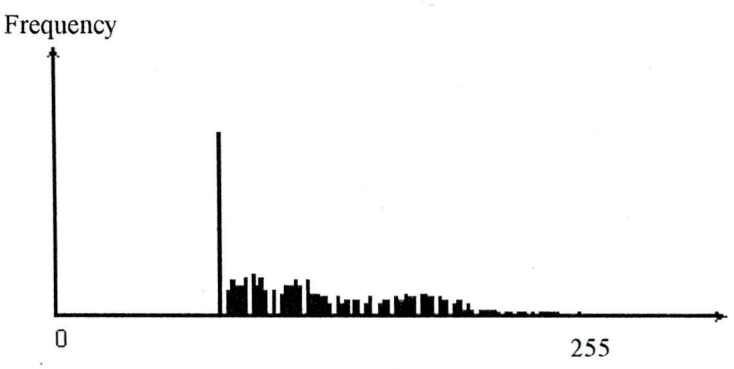
graphically display 256 numbers showing the distribution of pixels among those greyscale values.

Histograms have many uses and contain meaningful information. One of most common is what values are mostly being in image. We have seen that images must have two or three different significant points to process properly. Those points are significant because all of them have a particular information. For example, the point that is next to the intensity value "0", most cases point the background. Similarly, the other(s) indicates the foreground. This was our expectation at the starting point of the study, but it was not. Our researches show us that many of images we have worked on have more than three peaks on their histograms as shown in Figure 6-5.a. Another point that we have detected is the distance between intensity values that refer background is not closed to foreground intensity values. Under the light of this information, we have decided to shift all pixel values with adding a constant value. Let us assume that all pixel values are the result of an exact function $f(x)$ and $g(x)$ denotes that their converted values. The relation between $f(x)$ and $g(x)$ is shown in (6-1).

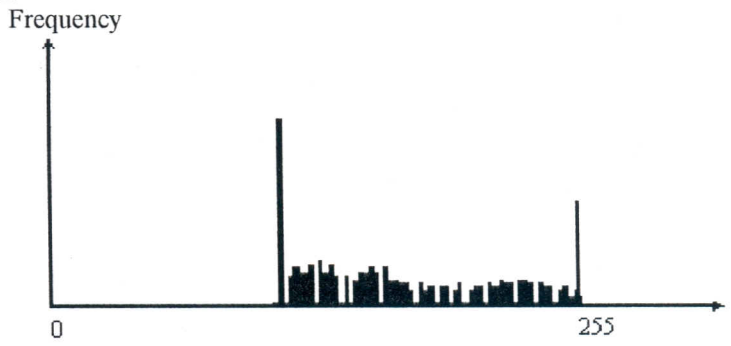
$$g(x)=f(x)+c \qquad \text{c = shifting constant} \qquad (6\text{-}1)$$

Where

$$0 \le f(x) \le 255$$
$$0 \le g(x) \le 255$$
$$0 \le c \le 255$$



(a)



(b)

Figure 6-5 (a) Histogram of original image (b) Histogram of shifted image, where c=70.

As shown in Figure 6-5.b, shifting operation not only reduces peak number, also makes an artificial edge point next to intensity value "255." The shifting constant c is determined from by using mean intensity value of the image. Because of this, sometimes its value is very close to zero and, sometimes it could be 100 or higher.

6.3.2 Contrast Stretching

In our study, histogram modification is not enough for preparing images to further stages generally. However, this does not mean it is not necessary. Quantizing (Teuber, 1989) is one of the method that we have utilized for enrichment of images. Although it is applicable some cases, the results that it has produced is not sufficient in our study. Another comprised operator of the process is called *contrast stretching*, and can be recognized as an additional step for obtaining the best results. As it is mentioned below, the contrast stretching (often called normalization) operator was used in our study to improve the contrast values of some of modificated images .

Contrast stretching is a simple image enhancement technique that attempts to improve the contrast in an image by `stretching' the range of intensity values it contains to span a desired range of values, e.g. the full range of pixel values that the image type concerned allows. It differs from the more sophisticated histogram equalization in that it can only apply a linear scaling function to the image pixel values. As a result the `enhancement' is less harsh. (Most implementations accept a greylevel image as input and produce another graylevel image as output.)

Before the stretching can be performed it is necessary to specify the upper and lower pixel value limits over which the image is to be normalized. Often these limits will just be the minimum and maximum pixel values that the image type concerned allows. For example for 8-bit graylevel images the lower and upper limits might be 0 and 255. Call the lower and the upper limits a and b respectively.

The simplest sort of normalization then scans the image to find the lowest and highest pixel values currently present in the image. Call these c and d. Then each pixel P is scaled using the following function:

$$P_{out} = (P_{in} - c)\left(\frac{b-a}{d-c}\right) + a \qquad (6\text{-}2)$$

The problem with this is that a single outlying pixel with either a very high or very low value can severely affect the value of c or d and this could lead to very unrepresentative scaling. Therefore a more robust approach is to first take a histogram of the image, and then select c and d at, say, the 5th and 95th percentile in the histogram (that is, 5% of the pixel in the histogram will have values lower than c, and 5% of the pixels will have values higher than d). This prevents outliers affecting the scaling so much.

Another common technique for dealing with outliers is to use the intensity histogram to find the most popular intensity level in an image (i.e. the histogram peak) and then define a cutoff fraction which is the minimum fraction of this peak magnitude below which data will be ignored. In other words, all intensity levels with histogram counts below this cutoff fraction will be discarded (driven to intensity value 0) and the remaining range of intensities will be expanded to fill out the full range of the image type under consideration.

Some implementations also work with color images. In this case all the channels will be stretched using the same offset and scaling in order to preserve the correct color ratios.

Normalization is commonly used to improve the contrast in an image without distorting relative graylevel intensities too significantly.

We begin by considering an image which has histogram below

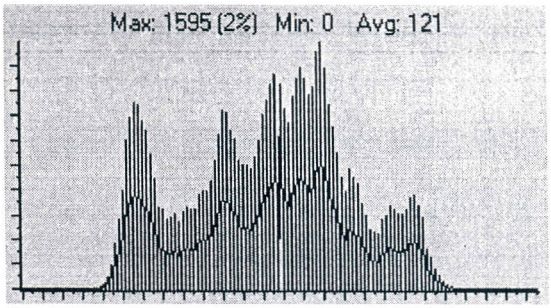


Figure 6-6 Histogram of original image

which can easily be enhanced by the most simple of contrast stretching implementations because the intensity histogram forms a tight, narrow cluster between the graylevel intensity values of 0 - 255 . After contrast stretching, using a simple linear interpolation between c = 0 and d = 255, we obtain

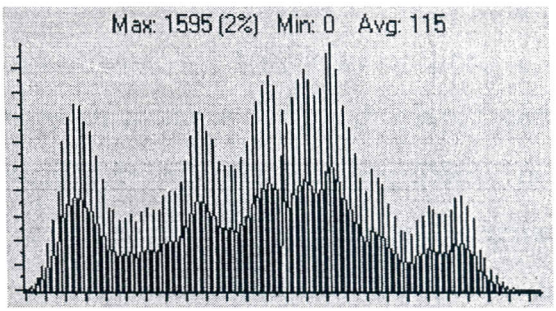


Figure 6-7 Stretched histogram of converted image

While this result is a significant improvement over the original, the enhanced image itself still appears somewhat flat. Histogram equalizing the image increases contrast dramatically, but yields an artificial-looking result.

We can achieve better results by contrast stretching the image over a more narrow range of graylevel values from the original image. This operation has effectively spread out the information contained in the original histogram peak (thus improving contrast in the interesting face regions) by pushing those intensity levels to the left of the peak down the histogram x-axis towards 0.

The minimum and maximum values in 8-bit images are 0 and 255 respectively, and so straightforward normalization to the range 0 - 255 produces absolutely no effect. However, we can enhance the picture by ignoring all pixel values outside the 1% and 99% percentiles, and only applying contrast stretching to those pixels in between. The outliers are simply forced to either 0 or 255 depending upon which side of the range they lie on.

Normalization can also be used when converting from one image type to another, for instance from floating point pixel values to 8-bit integer pixel values. As an example the pixel values in the floating point image might run from 0 to 5000. Normalizing this range to 0-255 allows easy conversion to 8-bit integers. Obviously some information might be lost in the compression process, but the relative intensities of the pixels will be preserved. Stretched values of the image has been used as input to exponential operator during the overall operation.

6.3.3 Exponential Operator

Although all the operations explained above are used in study to improve the images, it is a common problem that increasing the number of intensity levels in an image makes harder segmentation process. In our case, it shows that only two regions is sufficient to represent original image without any loss. Another word, it is enable to reconstruct kidney's image with two different graylevel. Although it is not common rule for all medical images, it seems to be an applicable way to solve our problem. In the process of our study we have implemented so many well-known image converting algorithms such as thresholding, linear and nonlinear operators. The results show us that the most appropriate approach to change the type of images is *exponential operator*.

The exponential and `raise to power' operators are two anamorphosis operators which can be applied to grayscale images. Like the logarithmic transform, they are used to change the dynamic range of an image. However, in contrast to the logarithmic operator, they enhance high intensity pixel values.

The exponential operator is a point process where the mapping function is an exponential curve. This means that each pixel intensity value in the output image is equal to a basis value raised to the value of the corresponding pixel value in the input image. Which basis number is used depends on the desired degree of compression of the dynamic range. In order to enhance the visibility of a normal photograph, values just above 1 are suitable. For display, the image must be scaled such that the maximum value becomes 255 (assuming an 8-bit display). The resulting image is given by Perkins and Fisher (Perkins and Fisher, 1996) as

$$Q(i,j) = c.b^{P(i,j)} \qquad (6\text{-}3)$$

where P and Q are the input and output images, respectively, b is the basis and c is the scaling factor. As we can see from the formula, Q = 0 yields P = c. To avoid the resulting offset, many implementations subtract 1 from the exponential term before the scaling. Hence, Perkins and Fisher have iterated equation (6-3) to get the following formula:

$$Q(i,j) = c.(b^{P(i,j)} - 1) \qquad (6\text{-}4)$$

Figure 6-8 shows the mapping function for b = 10 and b = 1.01.

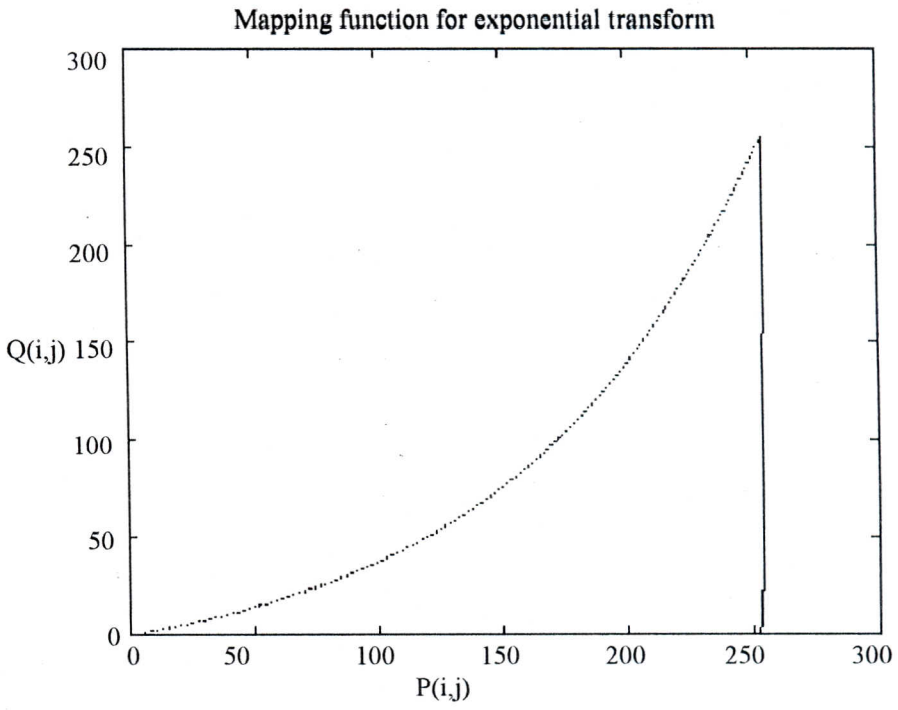


Figure 6-8 Exponential mapping functions for (solid line) and b = 1.01, c = 20.2 (dotted line).

We can see from the figure that the curvature of the base 10 exponential function is far too high to enhance the visibility of normal image. We can control the curvature of the mapping function either by choosing the appropriate basis or by scaling down the image prior to applying the exponential operator. This is because, over a low range of input values, an exponential function with a high basis has the same shape as an exponential function with smaller basis over a larger range of input values.

In the case of the `raise to the power' operator, the pixel intensity values in the input image act as the basis which is raised to a (fixed) power. The operator is defined by the following formula:

$$Q(i,j) = c.P(i,j)^r \tag{6-5}$$

If r > 1, the `raise to power' operator is similar to the exponential operator in the sense that it increases the bandwidth of the high intensity values at the cost of the low pixel values. However, if r < 1, the process enhances the low intensity value while decreasing the bandwidth of the high intensity values. This is similar to the effect achieved with the logarithmic transform.

Examples for r = 0.5, r=2 and r=6 can be seen in Figure 6-9.

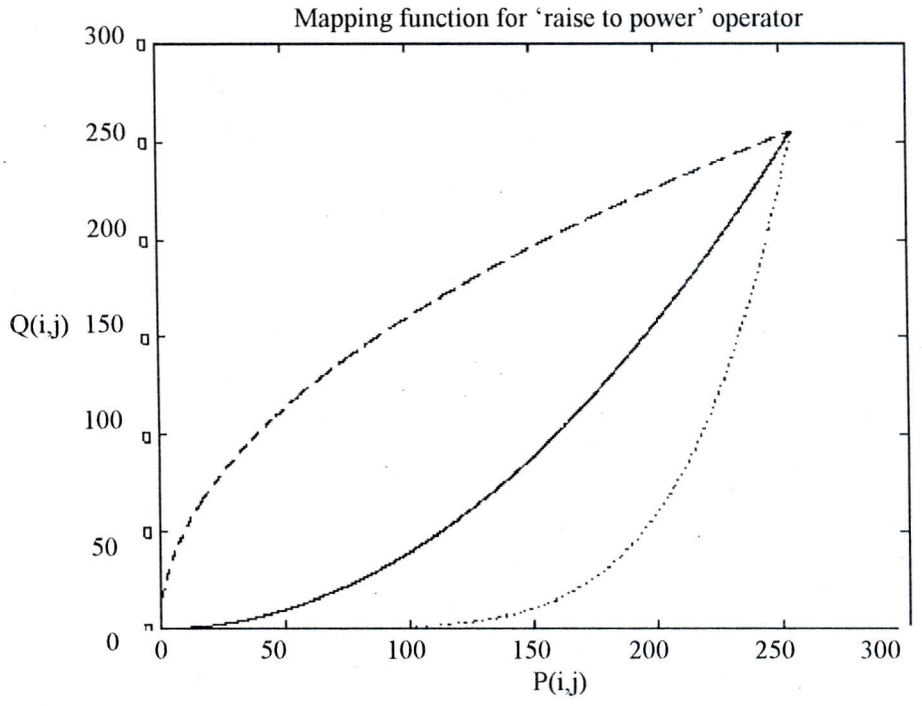


Figure 6-9 Mapping function of 'raise to power' operator for r=0.5 (dashed line), r = 2 (solid line) and r=6 (dotted line).

The exponential operator is the dual of the logarithmic transform. Hence, one application of the exponential operator is to undo changes originating from the logarithmic operator. In this case, the basis for the exponential operator should be the same as it was for the logarithmic transform. The reason for this is summarized in the equation below:

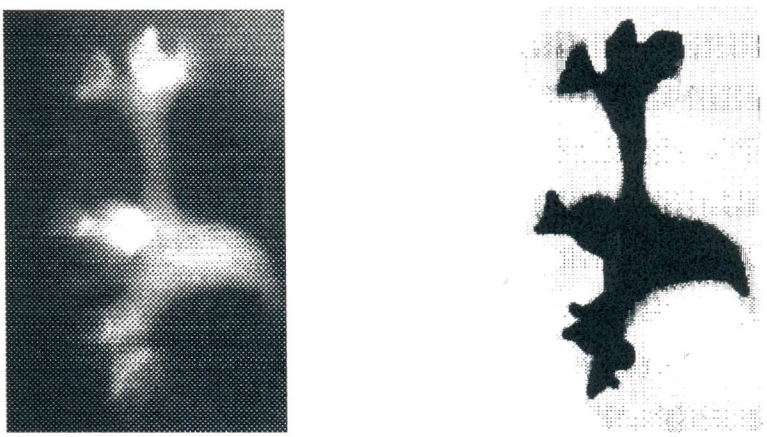
$$Q_2(i,j) = 10^{Q_1(i,j)} = 10^{k.\log(P(i,j))} = P(i,j)^k \tag{6-6}$$

where P is the original image, Q_1 is the image after the logarithmic transform and Q_2 is the final result.

Like the logarithmic operator, the exponential and `raise to power' operators might be used to compress the dynamic range or generally enhance the visibility of an image. Because they can be used to enlarge the range of high intensity pixel values relative to the range of the low intensity values, these operators are suitable for enhancement of images containing details represented by a rather narrow range of high intensity pixel values.

It can easily seen that the contrast in the high pixel values has been increased at the cost of the contrast in the low pixel values. We can see that both exponential and `raise to power' operator perform similarly at this (rather low) rate of compression. One difference, however, is that the `raise to power' operator does not depend on the scale of the image. The exponential operator, on the other hand, compresses the dynamic range more if the image contains pixels with very high intensity values (e.g. an image in floating point format with high pixel intensity values).

The operators above have been used by Aka et al. (Aka et al., 1996) and image produced by using exponential operator and its previous state are shown in Figure 6-10.



(a) *(b)*

Figure 6-10 (a) Original image, (b) where b= 2.718282 , c=255 and P(i,j) = (-1)[P(i,j) P(i,j)/12058]*

6.4 Reconstruction of Preprocessed Images

The point that we have reached shows us that it is not sufficient to use low level algorithms , as we have implemented, without any complementary high level image processing units. All stages that has been explained until now are performed so as to be a basis for the further stages of the processing of medical images.

At the first look both low and high image processing concepts seems to be have the same basis. The difference between them is their area of interest. In general, low level image processing techniques are used for normalization, converting and etc. Rule based segmentation, contour modeling and texture analysis can not be performed with it. At this point, high level image processing methods makes enable to solve this kind of problems.

In the case of our study, although all the procedures we have implemented generates two bit depth black and white images, the chest or the other part of the human body may occurs in the scene and appears as a part of the kidney. It is clearly seen that there is an exact segmentation problem which must be solved. In order to eliminate the segmentation problems we have used advanced image processing techniques. In Figure 6-11 it is shown that preprocessed kidney image.

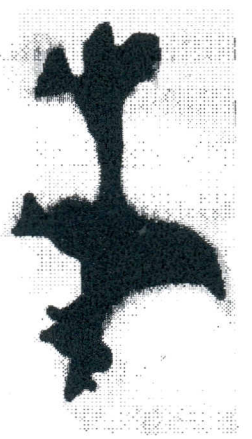


Figure 6-11 Preprocessed kidney image

The algorithm that we have developed which is shown below is explaining our approach. This algorithm is also used in (Güzel et al. ,1996a). It has mainly two different stages. First of all, the algorithm determines the boundaries of kidney and the last recreates of it.

Background cancellation algorithm:

1. Find all edges of the M x N matrix A by using Laplace Edge Detector.
2. Create an another matrix B have an same dimensions (M x N) and assign white color's numerical value to its all elements (pixels).
3. Select a point in the main image which is inside of disease part.
4. If the selected point is black then set pixel which is in the other matrix and located at same coordinates, black.
5. Repeat step 4 recursively until no black pixel traced in the main image

The first step of algorithm is used for to determine the boundaries of different regions located in current image. The result of applying first step to Figure 6-11 is shown in Figure 6-12.

Figure 6-12 Result image of step 1

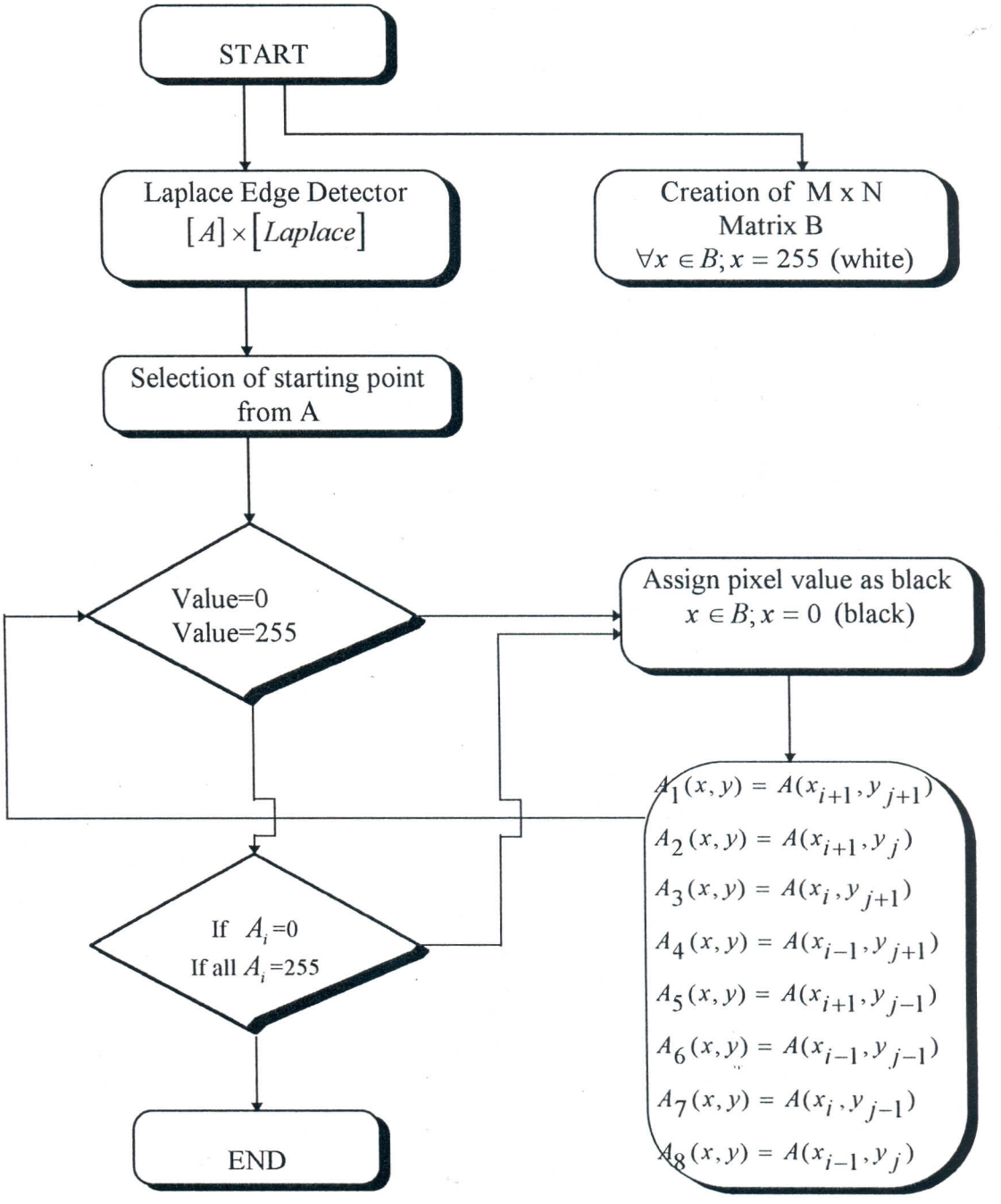


Figure 6-13 The flow chart diagram of background cancellation algorithm

The Figure 6-12 indicates that not only kidney region's boundaries are located, but also two different regions and background noises . Instead of processing the same image, it seems more feasible to create a new image (matrix) that have the same size and to paste the releated area located in the main image onto new scene.

As it is illustrated in Figure 6-13, our approach has performed its tasks perfectly as we expected. The product of using background cancellation algorithm can be seen in Figure 6-14.

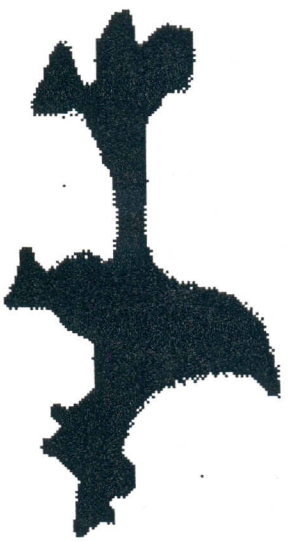


Figure 6-14 Applying of background cancellation algorithm

6.5 Storing Information

Our area of interest in medical image processing has so many different approaches. Above all, we have focused on the part which covers retrieving corrupted data from x-ray images and generating more clear ones.

Storing of huge amount of data is an another problem in our area. There are so many proposed methods such as Discrete Cosine Transform, Discrete Fourier Transform , etc., which are now being used . These are both subdivisions of *digital image compression.* Digital image compression handles every kind of image as an object with a consistent invariant approach. This makes a very flexible environment for researchers much of cases. Although its facilities, it has some trades off. The result of digital image compression techniques always effect as an image, and every image has a header. This makes loss of space on storage device. It is another disadvantage of using these approaches , always

some of the data is lost during the compression process. When we consider the problem from our point of view, it is not very suitable to use digital image compression approach for achieving the best results.

Every kidney image consists of nearly 120.000 pixels and a 1 Kb header (for bitmaps) . With a rough estimation, 121 Kb storage device space must be used for every image. This means at least 250 Kb disk space is necessary for every patient. However, in our case, the boundaries of kidney region are sufficient for diagnosis. In other words, storing the boundaries instead of complete image has the same precision. Under the light of this, we have assumed that if we detect and store the kidney's boundary coordinates on both x and y axis, then we can easily reconstruct the same image whenever we want. Although this was sufficient for our study, we have decided to develop a much more flexible file format by ordering x and y coordinate couples in counter anti-clockwise direction with the same information for further studies such as computer aided diagnosis systems. Güzel et al. (Güzel et al., 1997), have developed a system which classifies grades of VUR disease, can be given as an example.

In order to perform the things above, we have developed an algorithm which has two distinct parts, one of them is edge detection and the other is edge tracing. We have used Laplace edge detection filter for obtaining the boundaries with one pixel width. Although the part of this study will not be explained in this section in order to avoid the repetition, the result of applying this process to Figure 6-14 can be seen in Figure 6-16. The image produced by Laplace Edge Detector is traversed by a new tracing algorithm which is also developed by ourselves. This method includes several steps which can be seen above.

1. Trace the image contours in an ordered manner (anti-clockwise) and find the first white pixel
2. Assign the black value to the found pixel (x[m], y [m])
3. Search the 8 neighborhoods of the found pixel
4. Set the first found neighbor as an initial point and repeat 2 and 3 steps.
5. If no white pixel is left then stop the process.

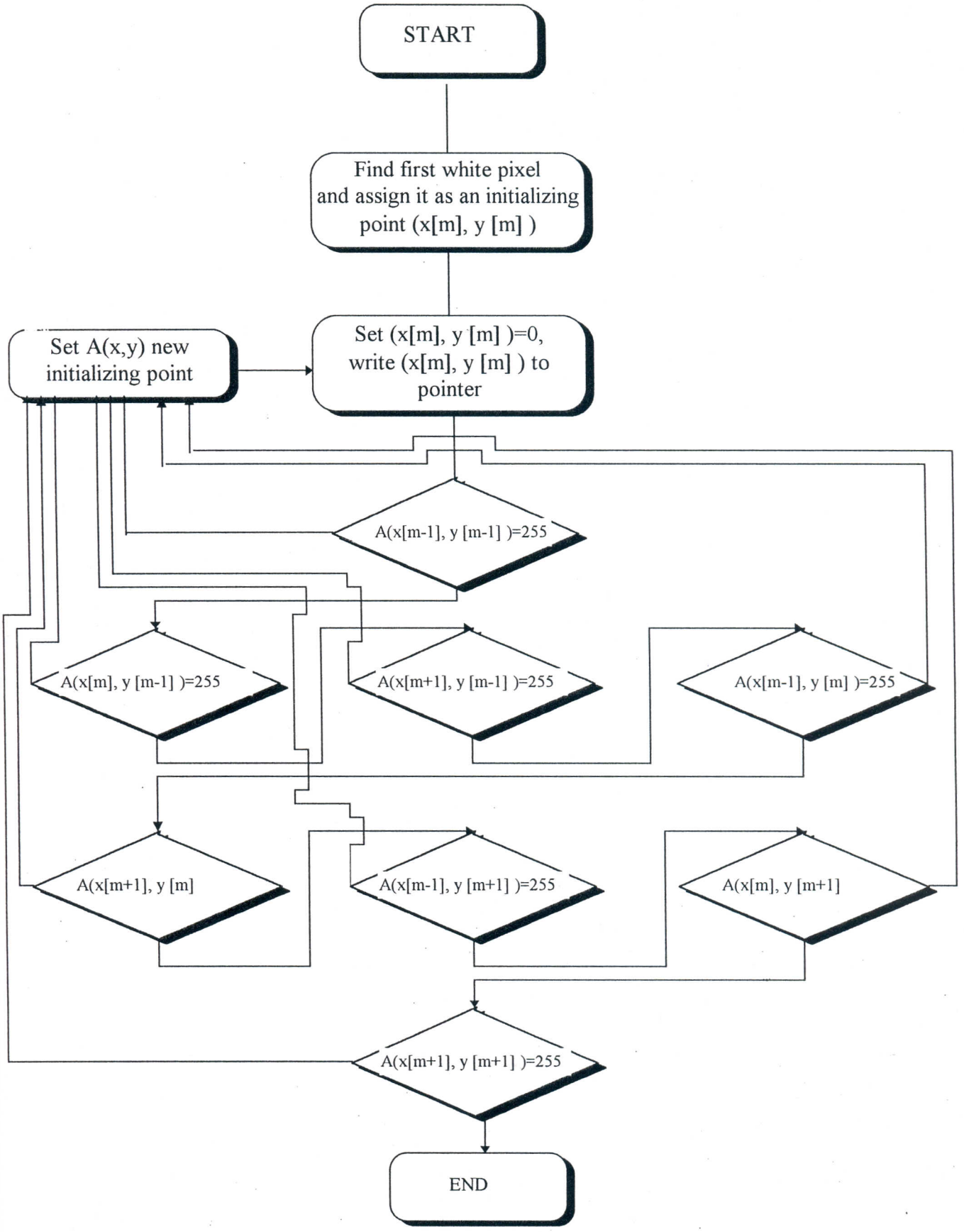


Figure 6-15 Flow chart diagram of edge tracing algorithm

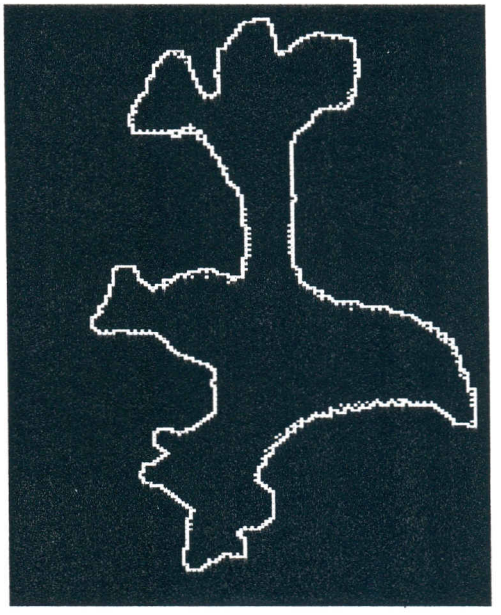


Figure 6-16 The production of Laplace edge detector

We have experimented on several kidney images for testing the robustness of this approach. Our experiments show us that although the algorithm has worked correctly, it has some trades off. The most important thing at the initializing point so as to eliminate these trades off, focused object's boundary edge points must be in the form of closed curve and edges must be one pixel thickness. Otherwise, the algorithm does not work as it is expected.

7

Chapter 7

CONCLUSIONS AND FUTURE WORKS

7.1 Conclusions

We have presented an integrated approach in retrieving, reconstructing and storing kidney images achieved from noisy X-rays. We have begun by defining the problem of extraction, then started to generate multi-step solutions. We have experimented so many different approaches for every single step. In each experiment, sometimes we have reached to a better solution and sometimes we have not. Thus, we have exerted every effort to optimize the solution for each step. Although we have done our best, as it is always faced in multi-functional systems, some of the steps have not been compatible to the others. So, we have applied many different combinations of these solutions to achieve a complete system. Although we have worked with limited sample images, due to the problem of retrieving x-rays which is seen in our case, the results show us that, all the samples that we have processed, could have been reconstructed and stored as we have expected.

At the stage of thresholding, we have applied so many different algorithms. Because of uneven illuminations occurred on the images, it has appealed that the conventional threshold, which is one of the filter that we have used, is not appropriate for our case. Adaptive thresholding is another method we have applied, although it has generated more clear images, some of cases it has failed due to the smoothness located at the edges. As we have mentioned before, we have obtained the best results with using the exponential filters.

Another problem which must have been solved, is the thickness of edges during the process of edge detection. There were several edge detection filters we have used to obtain the best result, which means one pixel thickness edge point, in our case. Although the Sobel operator is one of these and has generated edge points very clearly, it has produced edges with two pixel of thickness. So it was not an appropriate approach for us. We have also applied Laplacian of Gaussian second derivative edge detection operator. Although it has performed its task perfectly, it has required so much processing time due to its complexity. At last, we have focused on Laplace edge detector, which is also a second derivative operator. The results that it has produced were very similar to the Laplacian of Gaussian, but its performance time was nearly half of that.

The other parts of the system has designed to perform their tasks according to outputs of steps discussed. For example, developed edge tracing algorithm although perform its task as it is expected, also has some trades off. As it is discussed at last chapter, it uses the one-thickness pixels. This property of it sometimes makes tracing hardener. Another way, it has a step approach which enables to find next boundary pixel at the next of the current. If it is not, the algorithm recognizes that the process has been completed. This property also has an inheritance of that the pixels located on the

boundary must be in the form of an closed curve. It is very important that the terms last we have discussed at the point of system integrity. That is why we had always mentioned that the all layers must be compatible.

7.2 Future Works

A system which reconstructs X-ray images has been developed through this study. The subject which has been focused on was the X-rays of vesicle ureteral reflux disease. Storing X-ray data and patient data has been worked on although they were not the focus points of this study. From this point of view, an object oriented multimedia database can be developed to store and query these kinds of data. Such systems may supply physicians with a data pool. This study also provides educational advantages. If the designed database is opened to the Internet or intranet, millions of medical students can have access to these data (in the course of this study it has been observed that finding V.U.R. data is too difficult) and, they can improve their ability of diagnosis on this disease. What is more, detecting human face contours and profiles might be another application area of this study, and this might contribute to plastic surgery operations, too.

References

1. **[Aka et al. 1996]** H. C. Aka, C. Güzel, H. Püskülcü, S. Aytaç, T. Aktuğ, "Vesiko üreteral reflü hastalığının yapay sinir ağları yardımıyla sınıflandırılması", Proceedings of Biyomut- 96, 1996 Biomedical Engineering National Congress, pp. 99-106, 1996.

2. **[Aktuğ 1995]** T.Aktuğ, "Çocuk Cerrahisi",Güleç Kırtasiye ve Matbaacılık, 1995

3. **[Barzohar and Cooper 1996]** M. Barzohar and D.B. Cooper, "Automatic finding of main roads in aerial images by using geometric-stochastic models and estimation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 7, 1996

4. **[Chien and Mortensen 1996]** S.A. Chien and H.B. Mortensen, "Automating image processing for scientific data analysis of a large, image database", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 8, 1996

5. **[Choe et al. 1993]** Z. Choe, J.P. Jones, M. Singh, "Foundations of Medical Imaging", John Wiley & Sons Inc., New York, 1993, pp. 3-17.

6. **[Geiger et al. 1995]** D. Geiger, A. Gupta, L.A. Costa, and J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 3, March 1995

7. **[Gigus and Malik 1991]** Z. Gigus and J. Malik "Detecting curvilinear structure in images." ,Technical report CSD-91-619, University of California Berkeley, CS, 1991.

8. **[Greisen 1988]** E. W. Greisen , "The astronomical image processing system" National Radio Astronomy Observatory ,September, 1988

9. **[Güzel et al. 1996]** C. Güzel, H.C. Aka, H.Püskülcü, S. Aytaç, T. Aktuğ, "Reconstruction and recognition of V.U.R disease X-ray images", Proceedings of Third National Symposium on Biomedical Science and Technology, 1996.

10. **[Güzel et al. 1997]** C. Güzel, H. C. Aka, H. Püskülcü, S. Aytaç, "Yapay sinir ağları kullanarak şekil tanıma", Signal Processing and its Applications Conference, Proceedings of the SIU-97, vol.2 pp. 563-568, 1997.

11. **[Han et al. 1995]** Y. Han, G. Bilbro, R. Whitaker, and S. Pizer, W. Snyder, "Image relaxation: restoration and feature extraction", IEEE Transactions on Pattern Analysis and Machine Intelligence ,Vol. 17, No. 6, June 1995

12. **[Heijen 1995]** F.Heijden, "Edge and line feature extraction based on covariance models", IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 17, No. 1, January 1995

13. **[Iverson and Zucker 1995]** L.A. Iverson and S.W. Zucker, "Logical/linear operators for image curves", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 10, October 1995

14. **[Jain et al. 1995]** R. Jain, R. Katsuri, B.G. Schunck, "Machine Vision", McGraw-Hill International Editions, Singapore, 1995

15. **[King 1994]** L.R. King, " Vesicoureteral Reflux", Scientific Foundations of Urology, pp. 66-71, 1994

16. **[Law 1996]** T.Law "Image filtering, edge detection, and edge tracing using fuzzy reasoning", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No. 5, May 1996

17. **[Lindeberg 1996]** T. Lindeberg, "Edge detection and ridge detection with automatic scale selection" In CVPR'96 : IEEE Conf on Computer Vision and Patten Recognition, pp. 465--470, IEEE Computer Society Press, 1996

18. [Liu and Ehrich 1995] X. Liu and R. W. Ehrich, "Subpixel edge location in binary images using dithering", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 6, June 1995

19. [Low 1991] A. Low, "Introductory Computer Vision and Image Processing", McGraw-Hill Book Company, 1991.

20. [Merlet and Zerubia 1996] N.Merlet and J. Zerubia, "New prospects in line detection by dynamic programming", IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 18, No. 4, April 1996

21. [Nordstrom 1989] N. Nordstrom, "Biased anisotropic diffusion--a unified regularization and diffusion approach to edge detection." ,Technical report CSD-89-514, University of California Berkeley, CS, 1989

22. [Perkins and Fisher 1996] S. Perkins, B. Fisher, A. Walker, E. Wolfart, "HIPR - Hypermedia Image Processing Reference", Wiley, 1996

23. [Picton 1984] P.D. Picton, "Comment on elimination of redundant operations for a fast Sobel operator", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-14, No: 3, pp. 560-561, 1984

24. [Pratt 1978] W. K. Pratt, "Digital Signal Processing", John Wiley and Sons, 1978.

25. [Ridder et al. 1995] C. Ridder, O. Munkelt, and H. Kirchner, "adaptive background estimation and foreground detection using Kalman-filtering", Proceedings of International Conference on Recent Advances in Mechatronics, ICRAM'95 volume I, pp. 193-199, 1995

26. [Ryser 1963] H. J. Ryser, "Combinatorial Mathematics", John Wiley and Sons, 1963.

27. **[Schalkoff 1989]** R.J. Schalkoff, "Digital Image Processing and Computer Vision", John Wiley & Sons Inc,1989.

28. **[Stone and Isard 1995]** J.V. Stone and S.D. Isard, "Adaptive scale filtering: a general method for obtaining shape from texture", IEEE Transactions on Pattern Analysis and Machine Intelligence ,Vol. 17, No. 7, 1995

29. **[Teuber 1989]** J. Teuber, "Digital Image Processing", Prentice Hall, Cambridge, 1972

30. **[Wang and Rao 1996]** Z. Wang and K.R. Rao, "Optimal ramp edge detection using expansion matching", IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 18, No. 11, November 1996

31. **[Zhu and Chirlian 1995]** P. Zhu and P.M. Chirlian, "On critical point detection of digital shapes" ,IEEE Transactions on Pattern Analysis and Machine Intelligence Vol. 17, No. 8, August 1995

Appendix A

In this part of the study, you will find *ProVISION* which is developed for physicians by us to help them in their diagnosis process. OSF/Motif and C programming languages have been used onto IBM AIX operating system during the development of this project. It consists of medical imaging and patient registration tools. In Figure A-1, it can be seen that the security frame of the application. It is for to protect the patients data from unauthorized persons. It provides two-layered protection; Application *register no* and *password*.

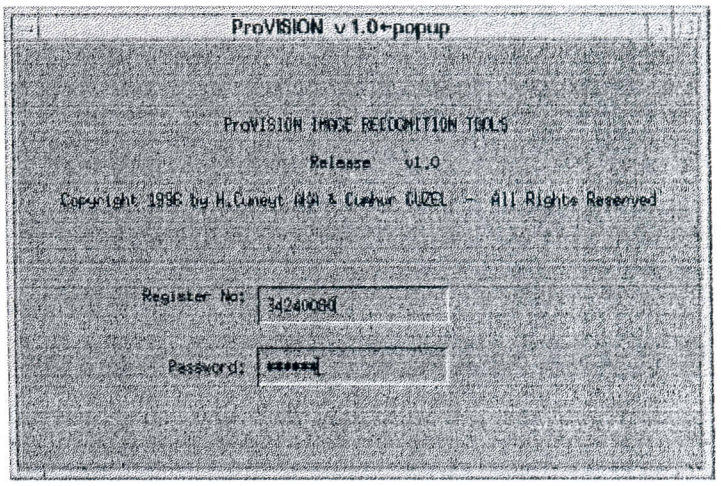


Figure A-1 Login frame

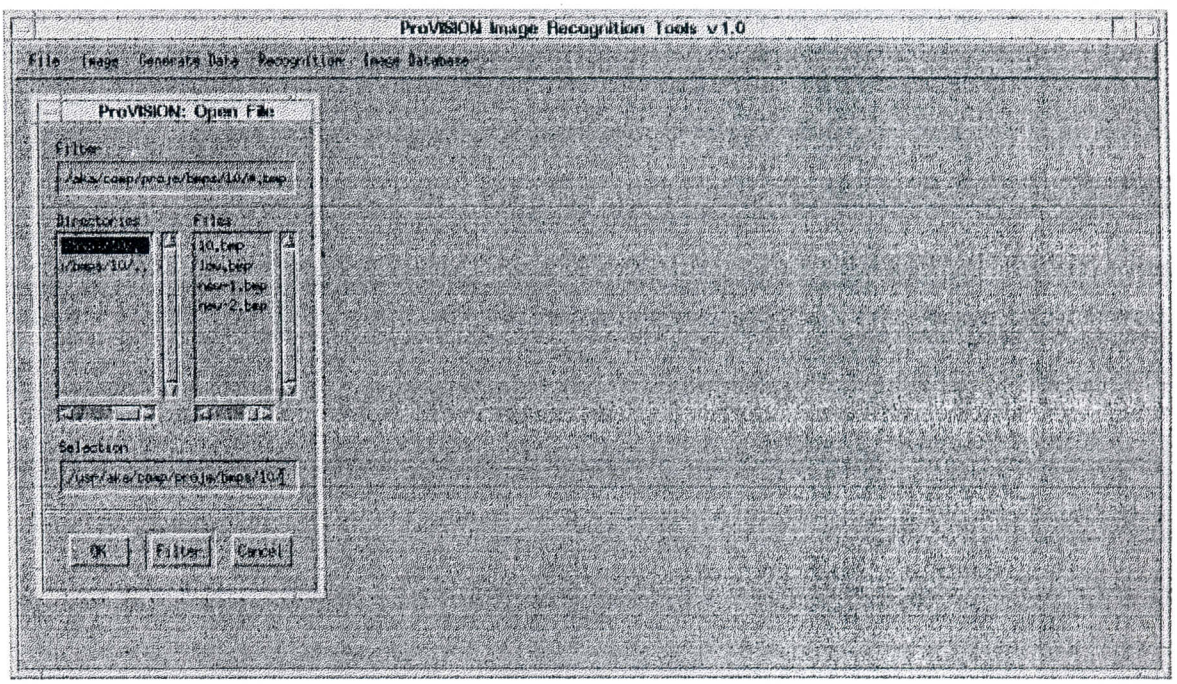


Figure A-2 File openning frame

The main frame consists of five different pull-down menu item. File menu is used for all file operations such as open, close, save and exit. Figure A- 2 shows the main frame and file opening process. It must be noticed that this is an prototypye and can only open and save bitmap formatted images. A sample view of an openned image can be seen in Figure A-3.

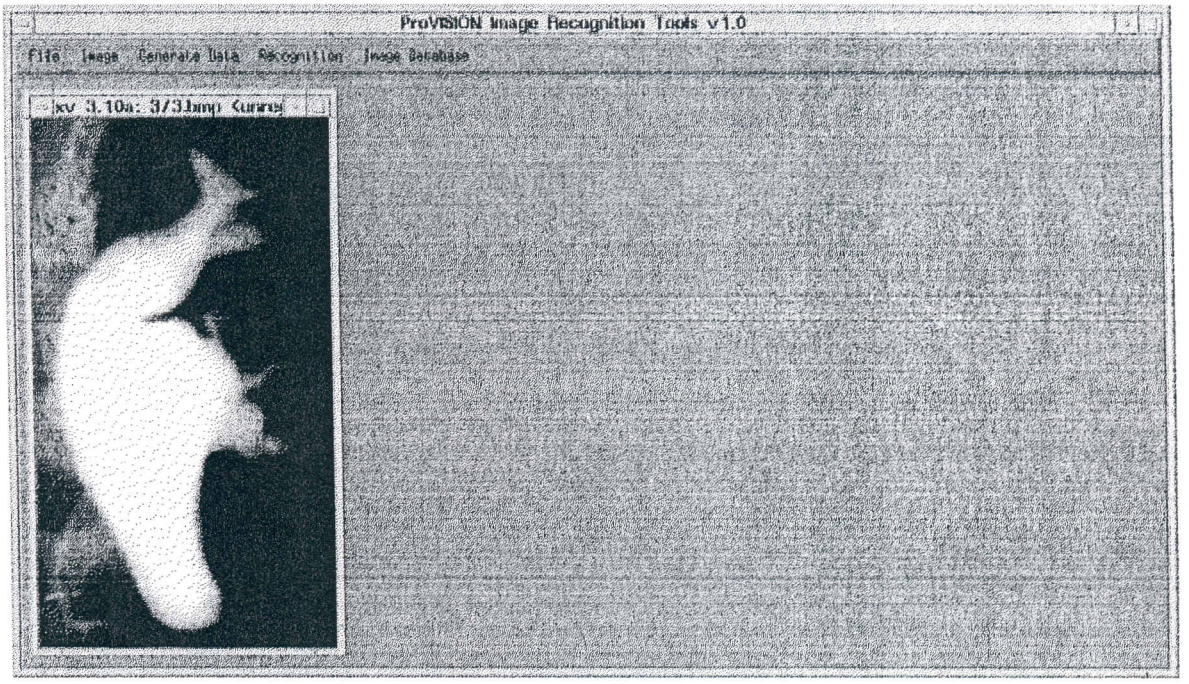


Figure A-3 ProVISION uses the program Xviev to visualize the medical images.

It provides to end users two different way to reconstruct or to change their images. They can use *Image* menu's different submenus which provides them freely process their images or *Generate Data* menu's *Process Image* option.

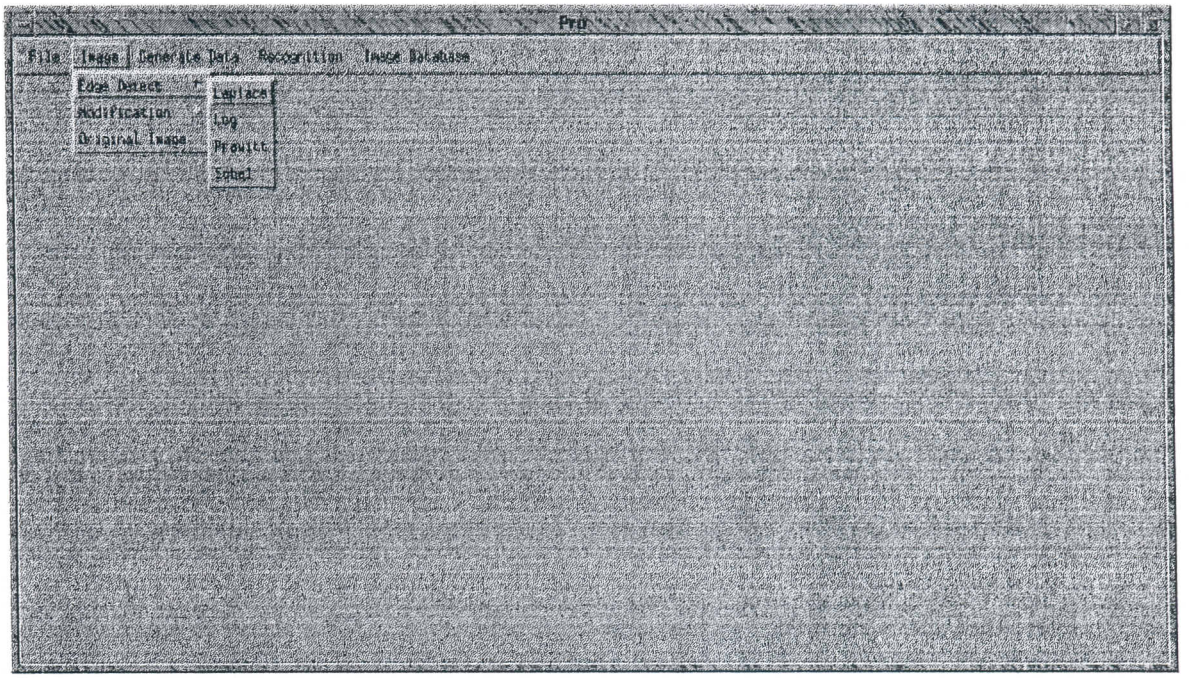


Figure A-4 Image menu

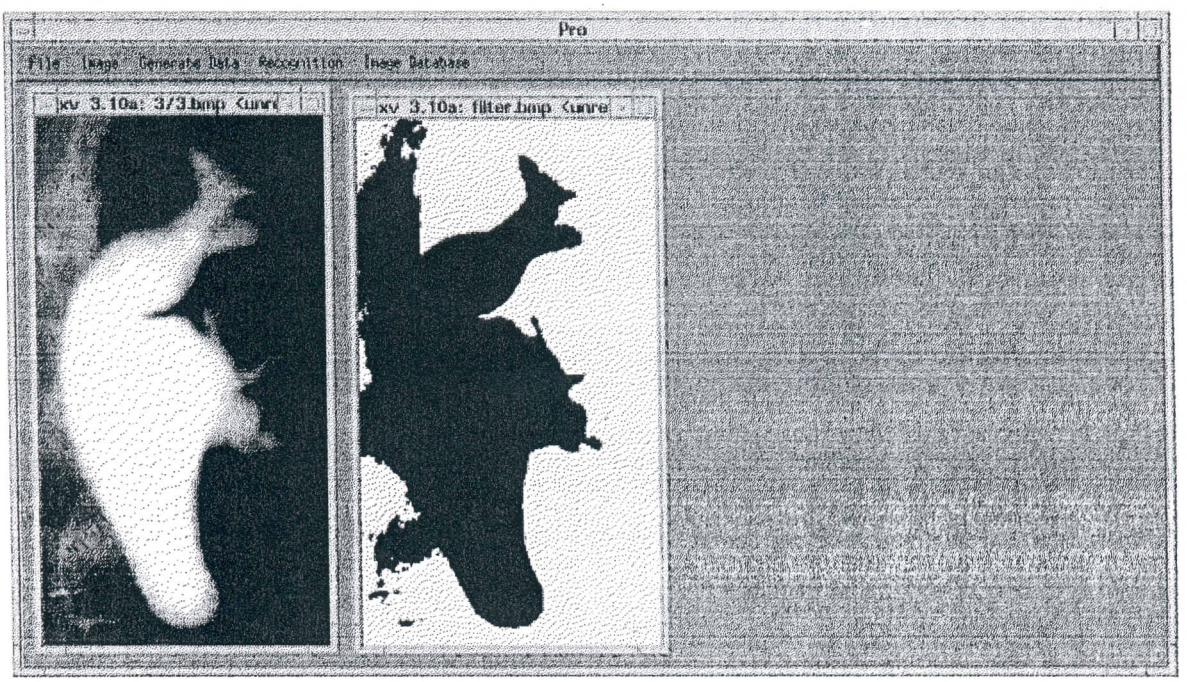


Figure A-5 Applying of threshold algorithm onto opened image.

If the physician uses the second option, the result will be appealed like below.

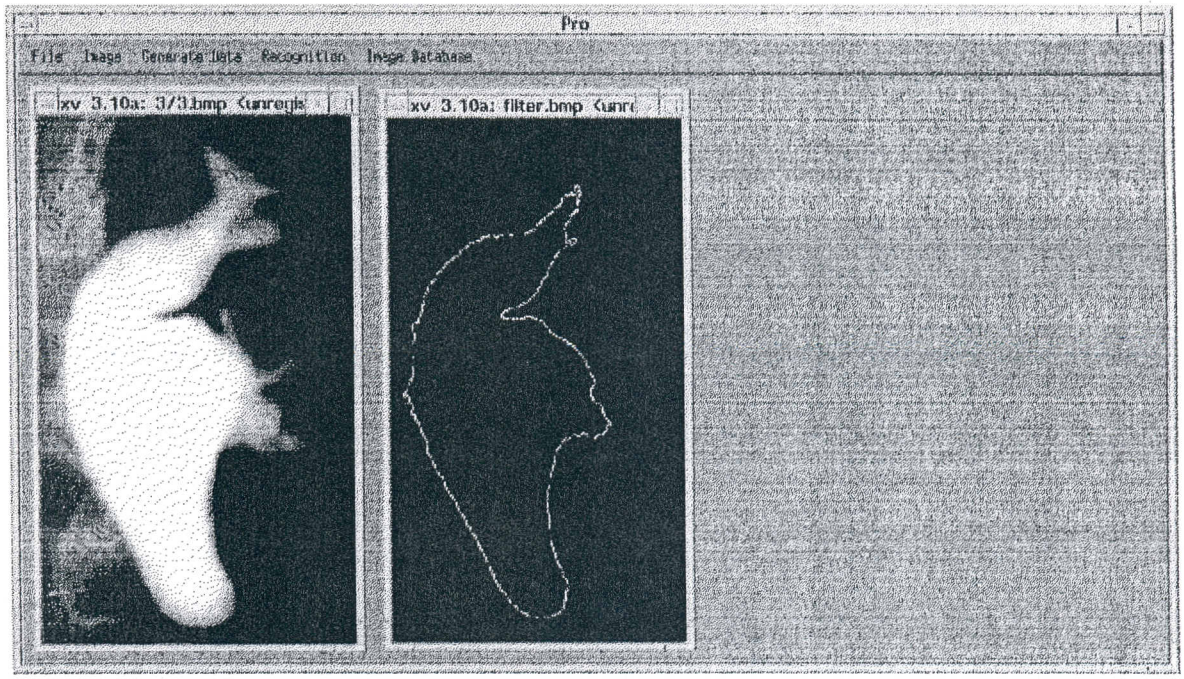


Figure A-6 The result of using Process Image option.

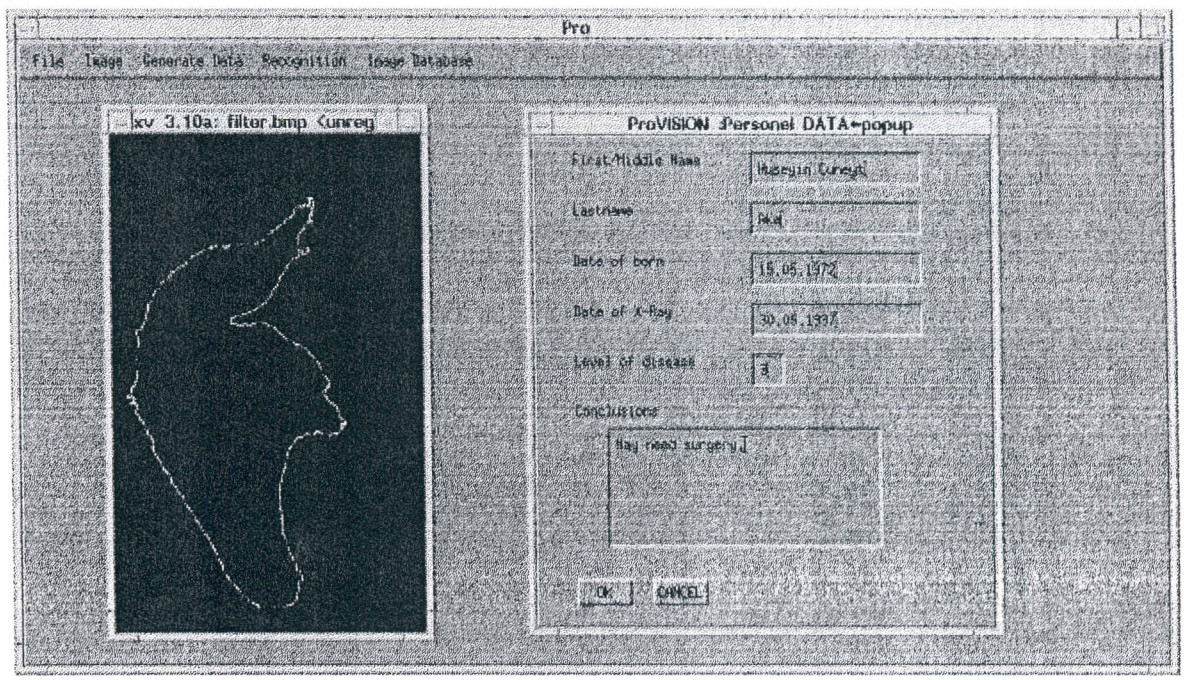


Figure A-7 Stroring information

As we have pointed that, ProVISION is not only an image reconstruction tool but also has the registration facility. It stores two different information about every image. The first one is stored for to rebuild handled image. This information is strored as an text file includes x and y coordinates for every different image. The other is used for to associate with the paitent's personel info and processed x-ray image. This is an indexed file and includes all the patients personel data.

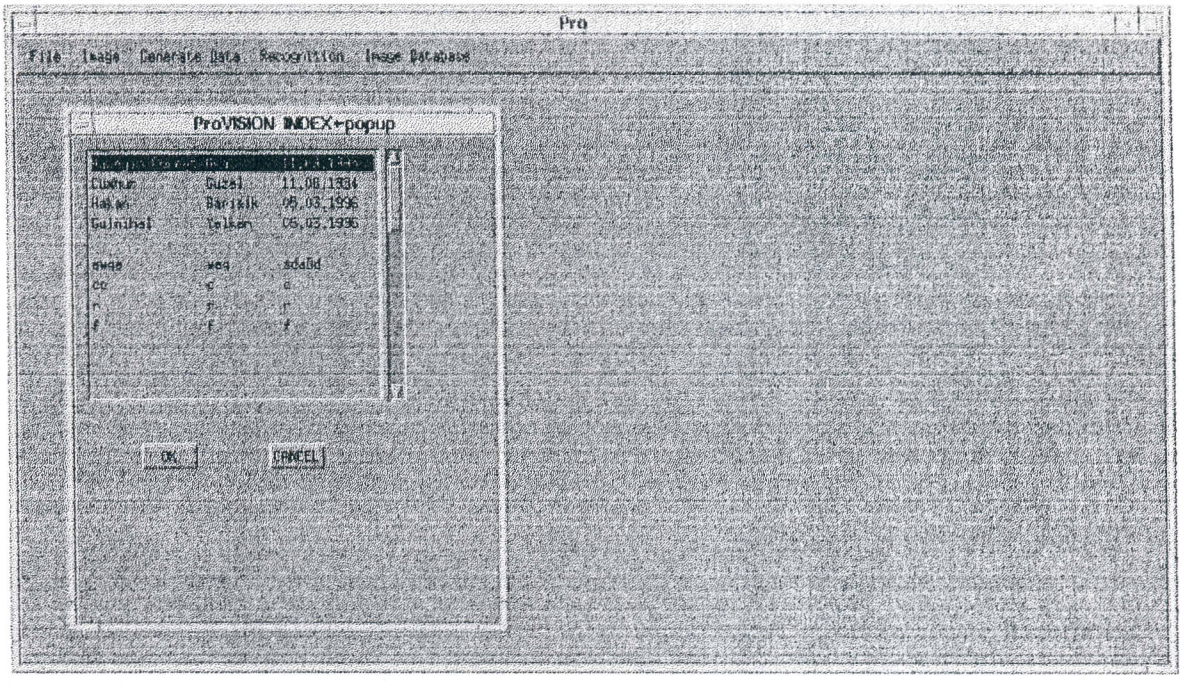


Figure A-8 Retrieving a record from file

Resulting Images of ProVISION

Figure A-9.a

Figure A-9.b

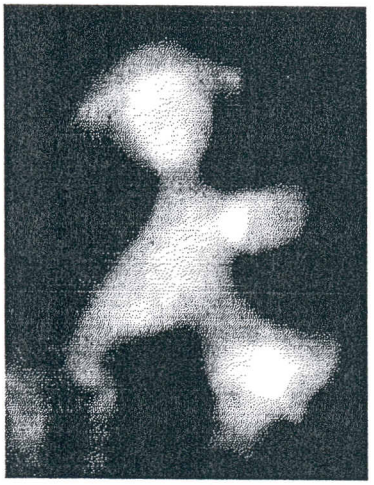


Figure A-10.a

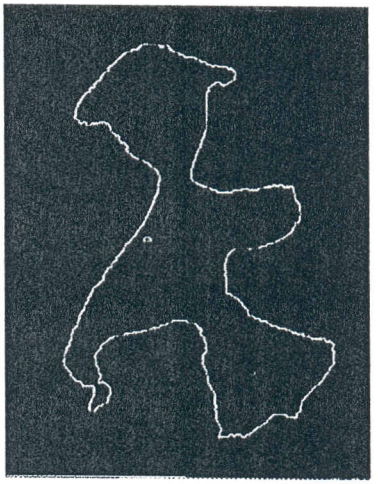


Figure A-10.b

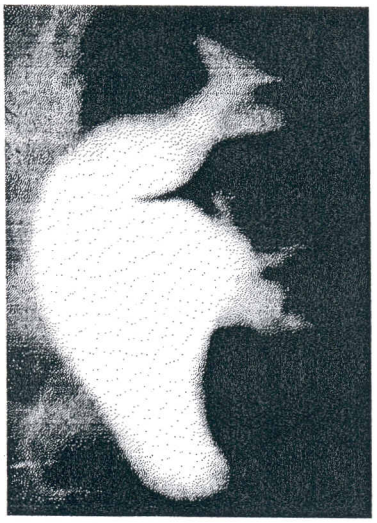


Figure A-11.a

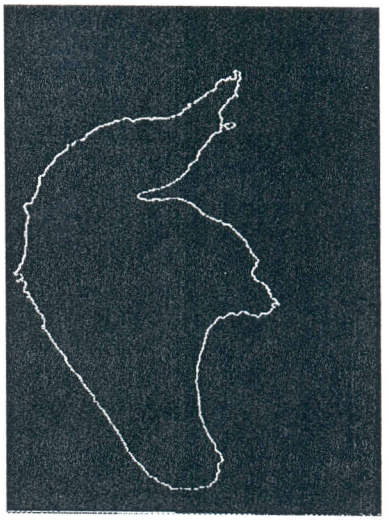


Figure A-11.b

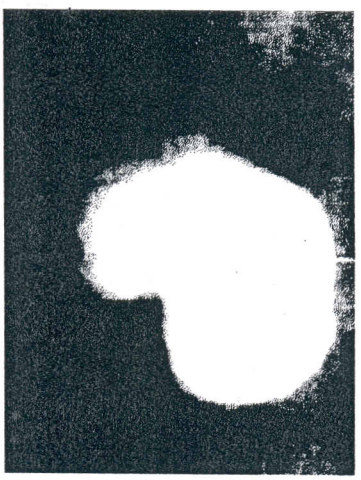


Figure A-12.a

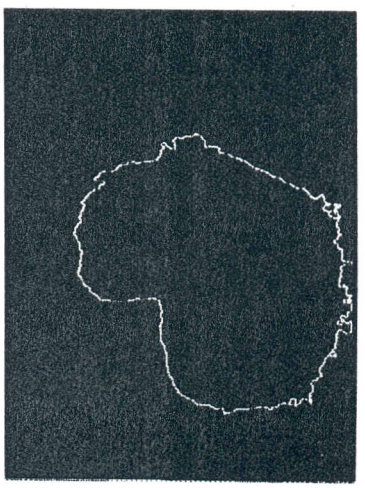


Figure A-12.b

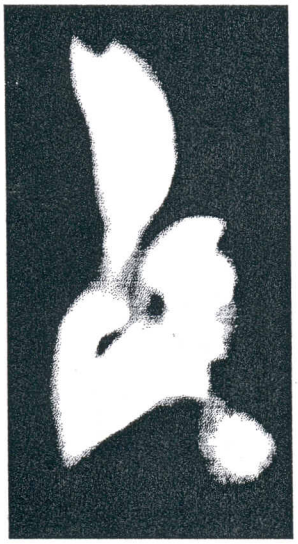


Figure A-13.a

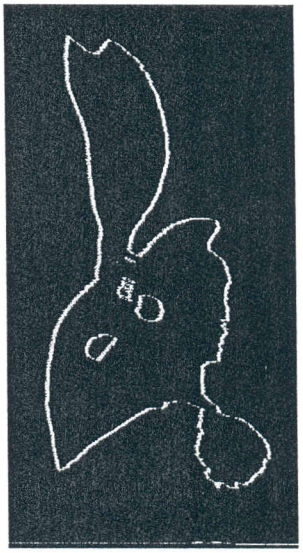


Figure A-13.b

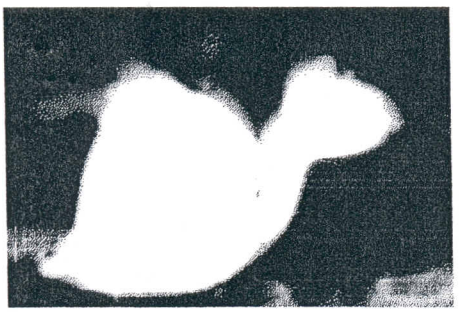


Figure A-14.a

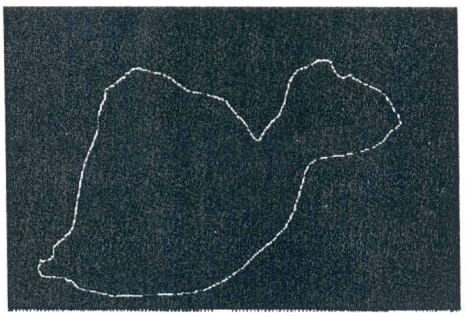


Figure A-14.b

Figure A-15.a

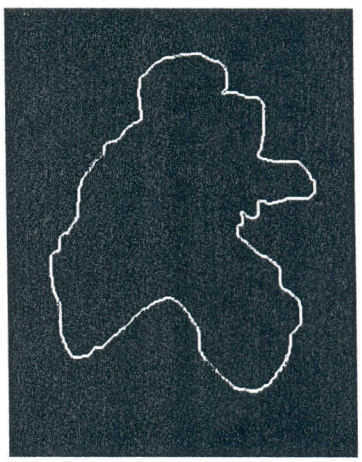


Figure A-15.b

Figure A-16.a

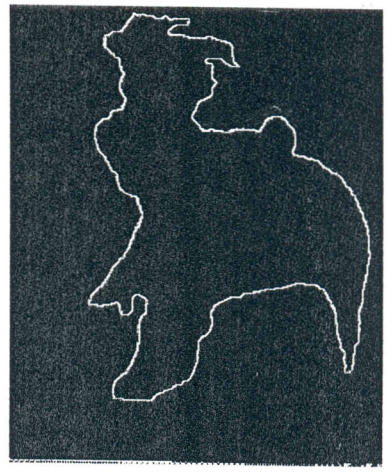


Figure A-16.b

Figure A-17.a

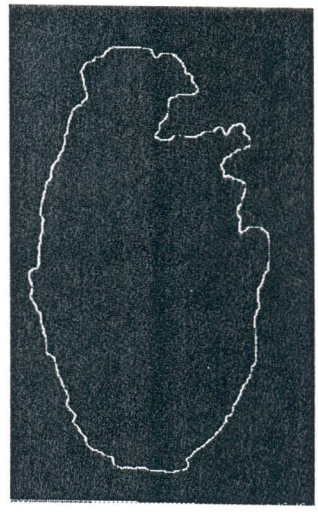


Figure A-17.b

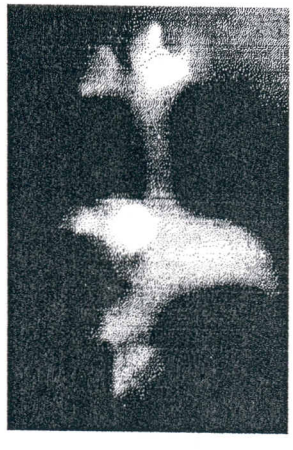


Figure-18.a

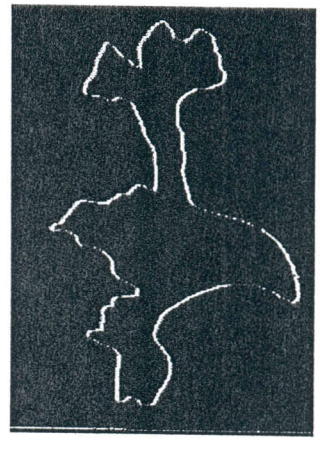


Figure-18.b