1991

Automated inspection of opthalmic [i.e. ophthalmic] needle using machine vision

Man Mohan Sapra
Lehigh University

AUTOMATED INSPECTION OF OPTHALMIC NEEDLE USING MACHINE VISION

by

Man Mohan Sapra

A Thesis

Presented to the Graduate Committee

of Lehigh University

in candidacy for the Degree of

Master of Science

in

Computer Science and Electrical Engineering Department

Lehigh University

1991

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering.

date: Sept 25, 1991

R T Gehton

Advisor in charge

M L Talbert

CSEE Department Chairperson

ACKNOWLEDGMENTS

I owe my sincere thanks and deep sense of gratitude to Professor Richard Denton who always supported and directed me during my research.

I would like to express my thanks to Dr. Emory Zimmers Jr., Director, CIM Lab, and the staff who were very supportive during my research.

I would also like to thank the Reaseach & Development Staff at Alcon Surgical, who were very cooperative and of great help in exchanging ideas.

Table of Contents

LIST OF FIGURES

ABSTRACT

The process of inspecting opthalmic needles for flaws makes demands that no human inspector can easily fulfill: visual acuity varies by the hour, peripheral concerns distort judgement, and repeated appraisals made on the fly tend to become erratic and unreliable. The application of machine vision technology on the manufacturing line inspection station to solve these problems is presented in this thesis. The result is improved quality control and increased product yields due to elimination of inconsistencies.

A specific sequence of techniques in image processing have been developed that bring out the edge detail of the tips and channels of the needles by enhancement and segmentation. These techniques are invariant to rotation and translation of the needle under inspection, thus providing the necessary flexibility in material handling.

Specific methods have been also developed to analyze the shapes of tips and channels that do not use model-matching. Model matching involves comparing the given object with a reference standard. A slight misalignment between the observed image and reference can lead to wrong results. A more flexible approach is shown that involves measuring a set of properties of the image and comparing the measured values with the corresponding expected values, in order to make a good/bad decision on the quality of the needle.

CHAPTER 1

INTRODUCTION

Machine vision systems - computers that can see and recognize have applications in virtually every branch of manufacturing. A broad domain of application is in inspection and quality control. The use of machine vision for in-process manufacturing verification and reduction of product cost reflects the quality control demand for reliable and consistent end product. Quality assurance can be defined as the level of reliability or repeatability that can be expected from a product. While random inspection can provide a reasonably high level of product quality; quality can best be verified through a process of 100 percent inspection, using machine vision.

Detecting defects on the tips and channels of opthalmic needles ,manually, through microscopes is very stressful and unreliable. Machine vision can be applied to automate this task. This is the topic of Chapter 2, where the input and output requirements of the system are discussed.

To detect the profiles of the tips and channels of the needles we must be able to distinguish them from the background. In order to increase the accuracy of edge detection, it is useful to enhance the contrast of the image. Then, suitable high pass filters can be applied to the image to enhance the edge-detail. Finally, the edges can be extracted sequentially from the sketch, in order to analyze their shape. All this is discussed in Chapters 3 and 4.

The analysis of the shape of tips and channels is the basis for making good/bad decisions. A method that finds corners on digital patterns is helpful in segmenting curves and locating features of interest in the image. Tips can be associated with a set of attributes using statistical tools and coordinate geometry. Fractal geometry can be used to characterize the shape of channels. This is the topic of Chapter 5.

The speed at which this inspection is desired requires dedicated hardware which is discussed in Chapter 6. The application of the methods developed in image enhancement, segmentation and analysis are illustrated on the tips and channels of good and bad samples of tips/channels in Chapter 7.

CHAPTER 2

SYSTEM REQUIREMENTS

2.1) INTRODUCTION

Automation has been a vital ingredient of computer integrated manufacturing because it strives for a smooth and continuous manufacturing process with minimum human tending and intervention. It can be very beneficial in areas where extreme care has to be taken in handling the product due to its fragile nature and microscopic dimensions. One such product is an opthalmic needle.

Opthalmic needles are used in performing surgical operations of the eye. On one end, the needle tapers to form the tip and the other end of the needle is called the 'cut-off' region. This is because the needle was cut off at this location from a wire to form the channel. The channel is a hollow region that extends a short distance into the needle. The sutures are attached to the needles by inserting them into this hollow region and then crimping the channel to hold the suture. One of the final stages of the manufacturing process of these needles is the 'cut-off' followed by their inspection for manufacturing defects. The needles are cut from wires ranging from 4 thousandths to 20 thousandths of an inch in diameter. Traditionally, the needles are handled individually by a pair of tweezers and placed into the slot in the die-set, ready to be cut. The machine is manually driven by a clutch to bring the punch down to cut the needle. The pressure applied to the clutch and the tensions acting on the needle while it is held by the tweezers in the slot determines the quality of the cut-off. It is a sensitive operation and requires a great deal of precision. One of the common defects that occur during this cut-off process are 'burrs' on the cut-off region. They can be caused if the needle is not seated properly in the slot. Burrs can also occur if the punch used to cut the needle is worn. Burrs can cut or weaken the suture. They can also tear the skin during surgery. Since defective cut-offs can be very damaging to the eye, the detection of these burrs during inspection, which is the next stage of the process, is very crucial.

The inspection process involves the detection of defects on two regions of the needle - tips and cut-off region. Conventionally, the inspection of the needles is done manually through microscopes. A tray containing a set of needles is scanned visually by moving it under the microscope. On the cut-off region, the human inspector looks for burrs and dirt. On the other end of the needles, he looks for bent, smashed or deformed tips and dirt.

2.2) HUMAN VISION vs MACHINE VISION

The objective of the automated system is to automate the two stages of the manufacturing process of the needles - cutting and inspection, as described above. The automation of the inspection process to detect defects on the needles using machine vision technology will be the focus of this thesis. We will first compare machine vision to human vision on the basis of the following functional parameters which are applicable to the inspection of needles.[1]

Adaptability

This is the capability of the system to automatically adjust or modify its operations according to environmental parameters to achieve a desired result. An example of adaptability is the ability of an individual to take a second look at an object in order to make decision if there is uncertainty about a detail due to poor seeing conditions.

Human vision is much more adaptable than machine vision. Since manufacturing costs are directly affected by the 'tightness of requirements,' it is desirable to make measurements to the necessary level of precision with a high degree of certainty. The nonadaptable characteristic of machine vision systems can be an advantage in industrial vision applications involving performance of identical processes on parallel manufacturing lines since it provides a high degree of measurement certainty.

Decision Making

The human vision systems can be effective in making value judgements for inspection tasks involving features such as color, shape, or odor since it has perceptional and interpretational capabilities. Machine vision systems require the quantification of the measurable parameters utilized in the decision making process. Machine vision systems will be more consistent than human vision systems for factual-based decisions.

Quality of Measurements

Consistency of results and level of precision are two main factors in the quality of measurements. Machine vision systems are clearly superior to human vision systems in the case of applications where the measurement is based on quantified input data. In addition, looking at needles manually through microscopes can be very stressful. The performance of human vision systems will vary over time due to such factors as fatigue, environmental condition, or distraction from other individuals in the work-force. The machine vision system has no random errors due to human fatigue or distraction and the established level of performance will be constant for all practical purposes, through the operating life of the equipment.

Speed of Response

A machine vision system's image acquistion time depends on the size of the image matrix, the processing time of the frame grabbing electronics, and the type of camera. The speed of image acquisition using machine vision systems is much greater than that of the human vision sytems. The ratio of the speed of machine vision to the speed of human vision is increasing with time as the state of the art in electronics improves.

Economic Considerations

The economics of using machine vision for an industrial application include both a capabilities enhancement factor and a direct manufacturing productivity cost factor. Providing the capability to manufacture superior quality products by performing 100% inspection to insure that every item meets specifications is an example of improved capability factor. At the same time, it can also result in increased productivity and reduced unit cost. There are significant savings in replacing a human inspector with an automated vision inspection station.

Taking all these factors into account, the vision system can pay for itself within a short period of time. Therefore, it's not hard to see why vision offers a number of significant advantages. In the next section, we will discuss the requirements of the overall automated system and then focus on the requirements of the machine vision system.

2.3) SYSTEM REQUIREMENTS

In this section, we will specify the basic input and output requirements of the system whose objective is to cut, inspect and sort the needles automatically.

Input Requirements

The system will be presented with carriers, each loaded with a set of needles which are ready to be cut-off and inspected. The needles have to be equally spaced in the carriers within a certain tolerance. The system is capable of holding several carriers at a time, that are stacked on top of each other.

Output Requirements

The system is required to place the needles into appropriate sets of trays labelled good, bad or unknown after they have been cut and inspected in real time. The needles in each tray have to be arranged in an orderly fashion (by rows and columns).

Control Requirements

Here we will briefly describe the sequence of tasks involved in the process of automating the cutting and inpection of needles. The operations are performed sequentially in a specific order. The needle is first positioned to be cut to length. After being cut, the first gripper holding the needle presents it to Camera 1 for channel inspection. After the channel is inspected by the vision system, the first gripper hands off the needle to a second gripper. The second gripper presents the needle to Camera 2 for the tip inspection. After the tip is inspected, the needle is placed in the appropriate tray based on the inspection results of the tips/channels and the cycle is repeated.

2.4) MACHINE VISION REQUIREMENTS

Here we will look at the input and output requirements of the machine vision system.

Input Requirements

Since the vision system is using a fixed window, each needle presented under the camera must be correctly oriented with its tip or channel in the field of view of the camera. The subjects should also be in focus. The background should be noise-free providing good contrast and the illumination should be uniform across the field of view. There should be no vibration affecting the needles during the acquisition of the image by the vision system.

Output Requirements

After the vision system has analyzed the image, it should output a good, bad, or unknown signal based on the result of the analysis.

The techniques used in processing the images and analyzing the shapes of the features in order to determine the quality of the tips and channels of the needles will be the subject of the following chapters.

CHAPTER 3

IMAGE ENHANCEMENT

3.1) INTRODUCTION & BACKGROUND:

Image Enhancement techniques are designed to improve image quality, based on the desires or requirements of the user. The principal objective of these techniques is to process a given image so that the result is more suitable than the original image for a specific application. The techniques that are used for image enhancement are very much problem - oriented. They are dependant on the kind of features that one is trying to extract from the image.

When an image is processed for visual interpretation, the viewer is the ultimate judge of how well a particular method works. This makes the definition of a good image harder to compare with algorithm performance. When the problem is to process images for machine perception, the evaluation task is a little easier. For example, if one were dealing with a character recognition application, the best image processing method would be the one yielding the best machine recognition results.

Therefore, it is important to emphasize that there is no general theory of image enhancement. Even in situations where a clear-cut method of performance can be imposed on the problem, one usually is still faced with a certain amount of trial and error before being able to settle on a particular image processing approach.

Image enhancement operations can be broken into two categories: subjective enhancement and objective enhancement. Before enhancing an image, the question to ask is: are we enhancing the image for an apparent increased quality (subjective) or correcting it for some known degradation (objective)? In the first case, the attempt is made to derive more visual knowledge from the image as it stands, and the operations are applied at the viewer's discretion.

Objective enhancement attempts to correct some known degradation that the image has encountered and hence does not necessarily strive for a more appealing image. For instance, if the original image was of

low contrast, then it may very well be the goal of an objective enhancement to recover the image as it originally existed-- low contrast. The following operations may be applied to either subjective or objective enhancement without distinction other than to its application.

3.2) CONTRAST ENHANCEMENT:

The most basic operation applied to any digital image is that of contrast enhancement. In its most basic form, this class of operations allows an image's gray scale occupancy to be altered. An important image analysis tool is the image histogram. The histogram gives a graphic display of the gray scale occupancy of an image. The two axes are labeled brightness and number of pixels, where the units of brightness range from 0 to 255 in the case of an 8-bit gray scale. This graph allows us to view an image's brightness distribution, helping in the application of an appropriate brightness redistribution technique to increase the contrast of the image.

In any given image, however, pixel brightness may not span the entire available gray scale range (from 0 to 255 for 8-bit pixel). This is often the case when imaging low contrast scenes. For instance, an image with the gray scale distribution lumped in the center of the available gray scale indicates low contrast. The histogram has to be stretched over the entire range of the gray scale in order to improve the contrast. This can be done by a technique known as histogram equalization (or stretching) that is based upon a mapping of the gray levels to achieve a uniform distribution, which is the topic of the following discussion.

Foundation

Let the random variable r represent the gray level of the pixels in the image to be enhanced. For simplicity, it will be assumed in the following discussion that the pixel values have been normalized so that they lie in the range

$$0 \leq r \leq 1 \tag{3-1}$$

where r=0 representing black and r=1 representing white in the gray scale. For any r in the interval [0,1], the following transformation will produce a level s for every pixel value r in the original image:

$$s=T(r), \tag{3-2}$$

It is assumed that the transformation function given in Eq. (3-2) satisfies the following conditions:

(a) T(r) is single-valued monotonically increasing in the interval $0 \leq r \leq 1$, and

(b) $0 \leq T(r) \leq 1$ for $0 \leq r \leq 1$,

Condition (a) preserves the order from black to white in the gray scale, while condition (b) guarantees a mapping that is consistent with the allowed range of pixel values. A transformation function satisfying these conditions is illustrated in Figure 3.1.

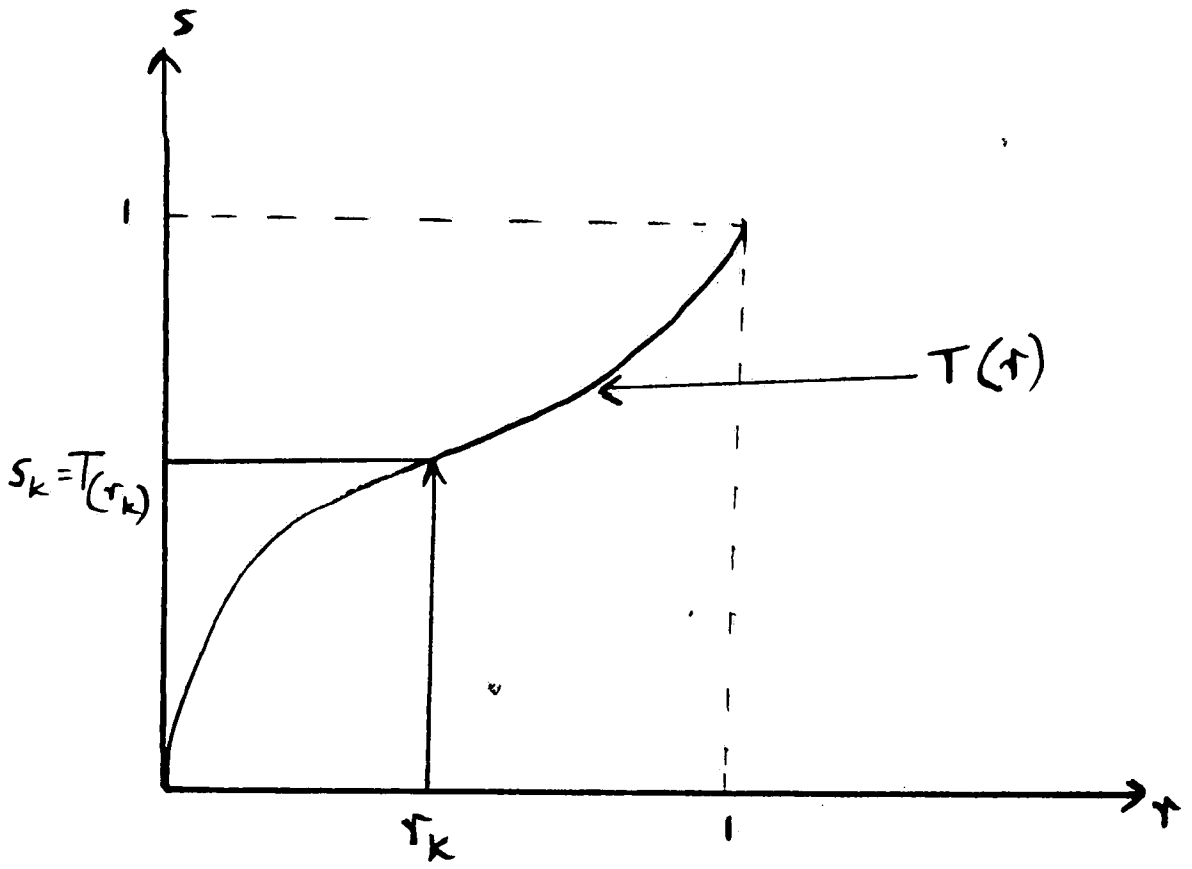


Figure 3.1 A gray-level transformation function

The inverse transformation from s back to r will be denoted by

$$r=T^{-1}(s) \qquad 0\le s\le 1, \tag{3-3}$$

where it is assumed that $T^{-1}(s)$ also satisfies conditions (a) and (b) with respect to the variable s.

The gray levels in an image are random quantities in the interval [0,1]. Assuming for a moment that they are continuous variables, the original and transformed gray levels can be characterized by their probability density functions p(r) and p(s), respectively. One can determine the general characteristics of an image from the density function of its gray values. For example, an image whose gray levels have a density function shown in Figure 3.2a would have fairly dark characteristics since most of the levels are concentrated in the dark region of the grey scale. On the other hand, an image with a density function shown in Figure 3.2b would be bright since majority of its pixels are in the light region of the grey scale.

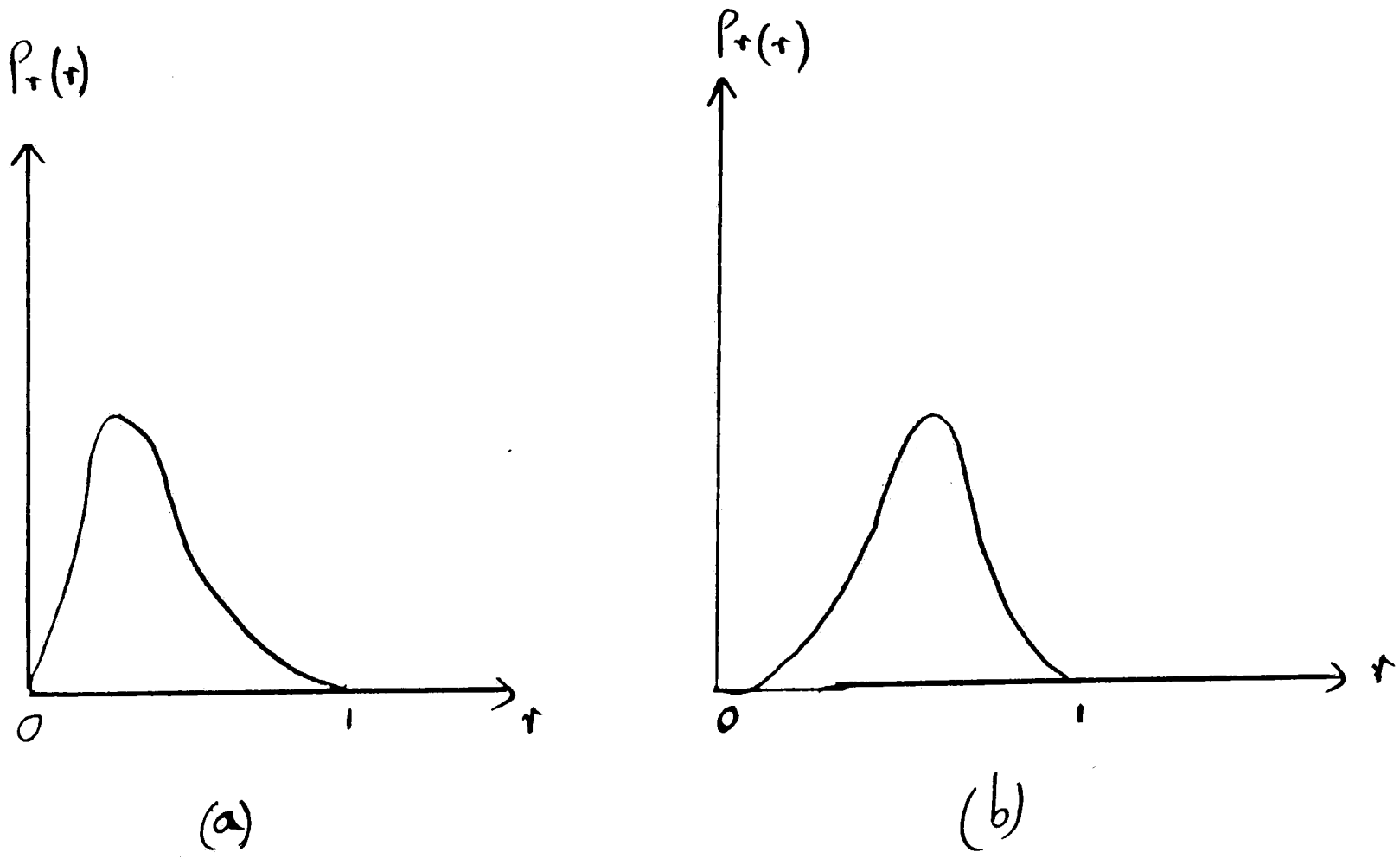


Figure 3.2 Gray level probability density functions of (a) a 'dark' image and (b) a 'light' image

Elementary prabability theory can be used to show that if p(r) and T(r) are known, and T(s) satisfies condition (a) , then the probability density function (pdf) of the transformed gray levels is given by the relation

$$p_s(s) = [p_r(r)\frac{dr}{ds}]_{r=T^{-1}(s)} \tag{3-4}$$

Therefore, T(r) provides a way to modify the pdf of the input image intensities which leads to histogram equalization.

Histogram Equalization

Normally, if the nonzero values of p(r) are clustered about small values of r, it would indicate that the image is excessively darkened and will not provide adequate contrast for further processing. The darkness in the image could be due to insufficient light being used to illuminate the object. A shadow that is overcast on the object under view could also contribute to this problem. On the other hand, if the nonzero values of p(r) are clustered around larger values of r, it would indicate that the image is excessively bright which would also hamper further extraction of useful information from the image. One of the factors that could contribute to this problem is the use of a light source which is set at a higher intensity level than required for the given magnification. This results in a 'washed out' image. Also excessive glare from the surface of the object or background can add to this problem. An incorrect setting of the sensitivity level of the sensor (camera) can be the cause to either cases. Neither of these cases is desirable for further image processing, since the range of the image intensities are quite narrow and therefore the perceived image contrast is poor. Thus an approximately uniform pdf is most desirable because it provides maximum contrast and utilizes the dynamic range of the sensor and display equipment.

Considering the gray-level transformation function

$$s = T(r) = \int_0^r p(u)du \qquad 0 \le r \le 1, \tag{3-5}$$

where u is a dummy variable of integration. The right side of the Eq. (3-5) is recognized as the cumulative distribution function (CDF) of the random variable r. This transformation function satisfies the two conditions that were describer earlier for Eq. (3-2).

Differentiation of Eq. (3-5) with respect to r using Liebnitz's rule yields

$$\frac{ds}{dr} = p(r) \tag{3-6}$$

Thus the pdf of a random variable r is the derivative of the cumulative distribution function, s or T(r).

Substituting dr/ds into Eq. (3-4) yields

$$\begin{aligned} p(s) &= [p(r)\frac{1}{p(r)}]_{r=T^{-1}(s)} \\ &= [1]_{r=T^{-1}(s)} \\ &= 1 \qquad\qquad 0 \le s \le 1 \end{aligned} \tag{3-7}$$

which is a uniform density in the interval of definition of the transformed variable s.[2] [3]

The above development indicates that using a transformation function equal to the cumulative distribution of r produces an image whose gray-levels have a uniform density. The end result is an increase in the dynamic range of the pixels which has a considerable effect on the appearance of an image.

In order to be useful for digital image processing, the concepts developed above must be formulated

in discrete form. For gray levels that assume discrete values,

$$p_r(r_k)=\frac{n_k}{n} \qquad \begin{matrix} 0\le r_k\le 1 \\ k=0,1,\ldots,L-1 \end{matrix} \tag{3-8}$$

where L is the number of levels, $p_r(r_k)$ is the probability of the kth gray level, n_k is the number of times this level appears in the image, and n is the total number of pixels in the image. A plot of $p_r(r_k)$ versus r is usually called a histogram.

The discrete form of Eq. (3-5) is given by the relation

$$\begin{aligned} s_k=T(r_k)&=\sum_{j=0}^{k}\frac{n_j}{n} \\ &=\sum_{j=0}^{k}p_r(r_j) \end{aligned} \qquad \begin{matrix} 0\le r_k\le 1 \\ k=0,1,\ldots,L-1 \end{matrix} \tag{3-9}$$

The inverse transformation is denoted by

$$r_k=T^{-1}(s_k) \qquad 0\le s\le 1, \tag{3-10}$$

where both $T(r_k)$ and $T^{-1}(s_k)$ are assumed to satisfy conditions (a) and (b) of Eq. (3-2).

Since a histogram is an approximation to a probability density function, perfectly flat results are seldom obtained when working with discrete levels. While the equalized histogram is, as expected, not perfectly flat throughout the full range of gray levels, considerable improvement over the original image can be achieved by the spreading effect of the histogram-equalization technique.

CHAPTER 4

IMAGE SEGMENTATION

Segmentation is the process that subdivides an image into its constituent parts or objects. Segmentation is a very important step in automated image analysis because it is during this stage that the features of interest are extracted from an image for subsequent processing. A key element in the task of locating features is that of identifying the location of edges. The algorithms are based on a very basic property of gray values, namely discontinuity. The segmentation operation depends on the capability of the machine vision system to detect the edge or an abrupt change in the gray level value associated with the pixels where the edge is located.

4.1) EDGE ENHANCEMENT

Of geat importance to machine vision systems is the ability to bring out the edge detail in an image. Edges are vital features of an image because they convey central information about the size and shape of objects. Edge enhancement can allow the image processing system to make edge-to-edge boundary distance measurements as well as provide the base of information necessary for automated image understanding. Thus a reliable and accurate technique for detecting edges is crucial to further processing of an image.

Basic Formulation

We define an edge as the boundary between two regions with relatively distinct gray-level properties. In a continuous image, a sharp intensity transition between neighboring pixels would be considered an edge.

Basically, the idea underlying most edge-detection techniques is the computation of a local derivative operator. This concept can be illustrated by examining Figure 4.1. Part (a) of this figure shows an image of a simple light object on a dark background. Proceeding downwards it shows the gray level profile along a horizontal scan line of the image, and the first and second derivatives of the profile. From the profile, an edge is modeled as a ramp, rather than as an abrupt change of gray level. This model indicates the fact

that edges in digital images are always slightly blurred due to sampling.

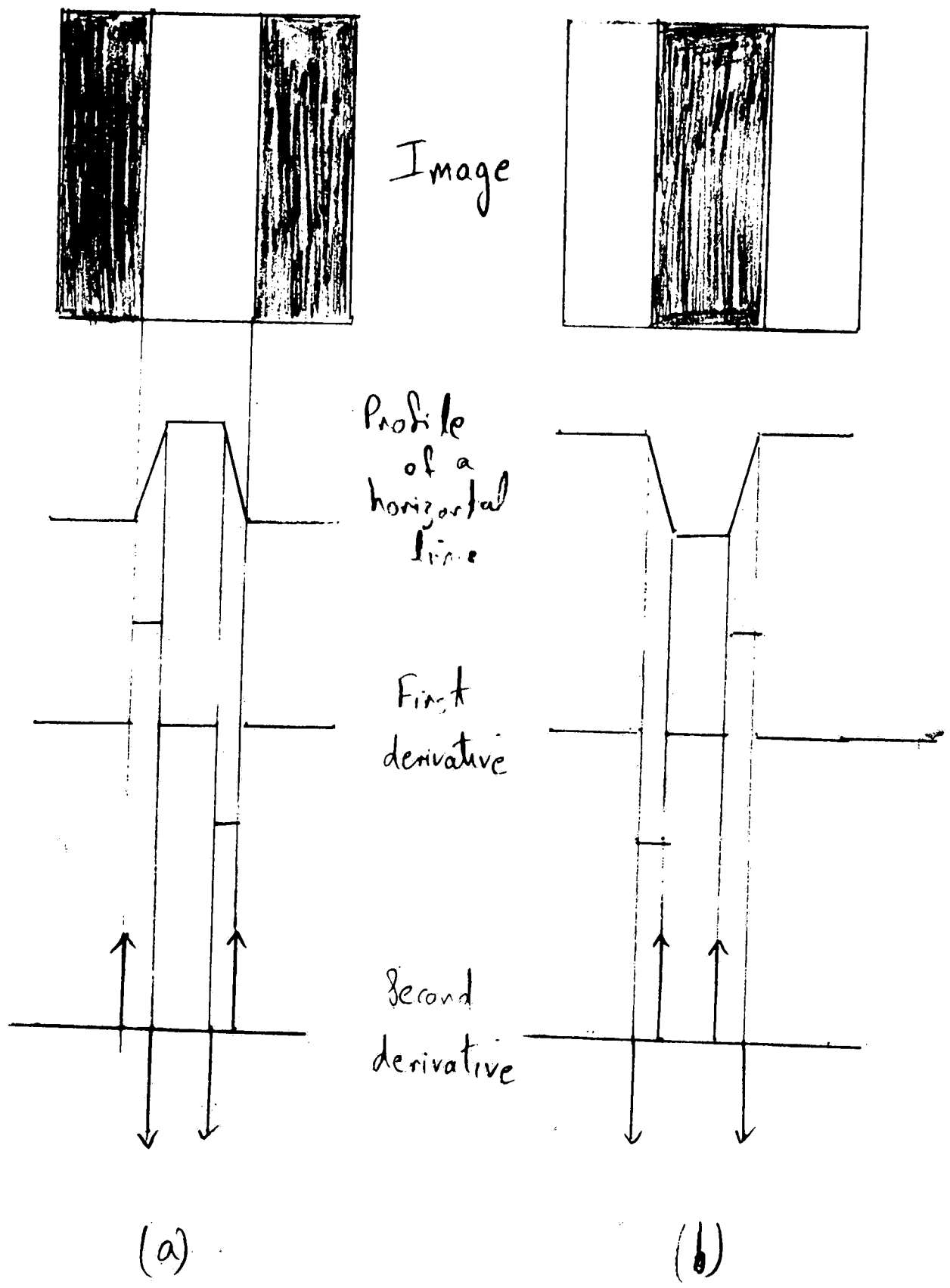


Figure 4.1 Elements of edge detection by derivative operator
(a) Light object on a dark background
(b) Dark object on a light background

The first derivative of an edge modeled in this manner is 0 in all regions of constant gray level, and assumes a constant value during a gray-level transition. The second derivative, on the other hand, is 0 in all locations, except at the onset and termination of a gray level transition. Based on these remarks and the concepts illustrated in Figure 4.1, it is evident that the magnitude of the first derivative can be used to detect the presence of an edge, while the sign of the second derivative can be used to determine whether an edge pixel lies on the dark (background) or light (object) side of an edge. The sign of the second derivative in Fig 1 Part(a) , for example, is positive for pixels lying on the dark side of both the leading and trailing edges of the object, while the sign is negative for pixels on the light side of these edges. Similar comments apply to the case of a dark object on a light background, as shown in Fig 1. Part (b).

Although the discussion thus far has been limited to a one-dimensional horizontal profile, a similar argument applies to an edge of any orientation in an image. We simply define a profile perpendicular to the edge direction at any given point and interpret the results as in the preceding discussion. From experimenting with the first derivative and second derivative operators on images, we observe that the edges are much thicker on the images from the first derivative operators. The edges produced from the application of the second derivative operator are much sharper. It is difficult to work with thick edges especially when one is measuring the irregularity or smoothness of a boundary. If the edge is spread over more than two pixels, an irregularity that is smaller than this will go unnoticed. Sharper edges are always desirable in applications dealing with great precision. Therefore, we restrict our discussion to the methods related to the second derivative operations namely Laplacian. Before we do this we will introduce the concept of spatial filtering. This is the tool used to apply derivative operators.

Spatial Filtering

The term spatial domain refers to the aggregate of pixels composing an image, and spatial-domain methods are procedures that operate directly on these pixels. Image processing functions in the spatial domain may be expressed as

$$g(x,y)=T[f(x,y)], \tag{4-1}$$

where f(x,y) is the input image, g(x,y) is the processed image, and T is an operator on f, defined over some neighborhood of (x,y).

Spatial filters are operations that create an output image based on the spatial frequency content of the input image. An image may be represented as an array of two-dimensional frequency components of varying amplitudes and phases. This representation is known as a Fourier decomposition. Spatial filtering allows the separation of these components in an image. On a pixel-by-pixel basis, an output image is generated based on a pixel's brightness relative to its immediately neighboring pixels. Where a neighborhood's pixel brightness makes rapid transitions from light to dark or vice versa, the image is said to contain high frequency components. A neighborhood of slowly varying pixel brightnesses represents low-frequency components.

A spatial filter attenuates or accentuates the two-dimensional frequency content of an image. These operations may be used to accentuate an image's high-frequency details (edges), yielding a sharper image. Alternatively, the high-frequency details may be attenuated, yielding a low-passed image of little detail.

Spatial filtering is carried out using spatial convolution, an adaptation to two dimensions of the convolution techniques used in signal processing theory. For each input pixel within an image one calculates an output pixel based on the weighted average of it and its surrounding neighbors. Typically, a three pixel by three pixel neighborhood is used for this calculation although larger neighborhoods may be used for added flexibility. With the correct selection of weighting coefficients, we can form highpass, lowpass, and various edge enhancement filters.

The general approach in reference to Figures 4.2 and 4.3 is to let the values of f in a predefined neighborhood of (x,y) determine the value of g at those coordinates. One of the principal approaches in this formulation is based on the use of so called masks (also called windows, templates, filters). Basically, a

mask is a small (eg. 3 X 3) two dimensional array, as shown in Figure 4.2, whose coefficients are chosen to detect a given property in an image. In carrying out a spatial convolution for this array, nine weighting coefficients are defined and labeled w1, w2, w3 .... w9. The center of the mask (labeled w5) is moved around the image, as indicated in Figure 4.3. At each pixel position in the image, we multiply every pixel that is contained within the mask area by the corresponding mask coefficient. The results of these nine multiplications are then summed, producing a new, spatially filtered output pixel brightness. This operation is systematically applied to each pixel in the input image.

w_1 $(x-1, y-1)$	w_2 $(x-1, y)$	w_3 $(x-1, y+1)$
w_4 $(x, y-1)$	w_5 (x, y)	w_6 $(x, y+1)$
w_7 $(x+1, y-1)$	w_8 $(x+1, y)$	w_9 $(x+1, y+1)$

Figure 4.2 A general 3X3 mask showing coefficients and corresponding image pixel locations.

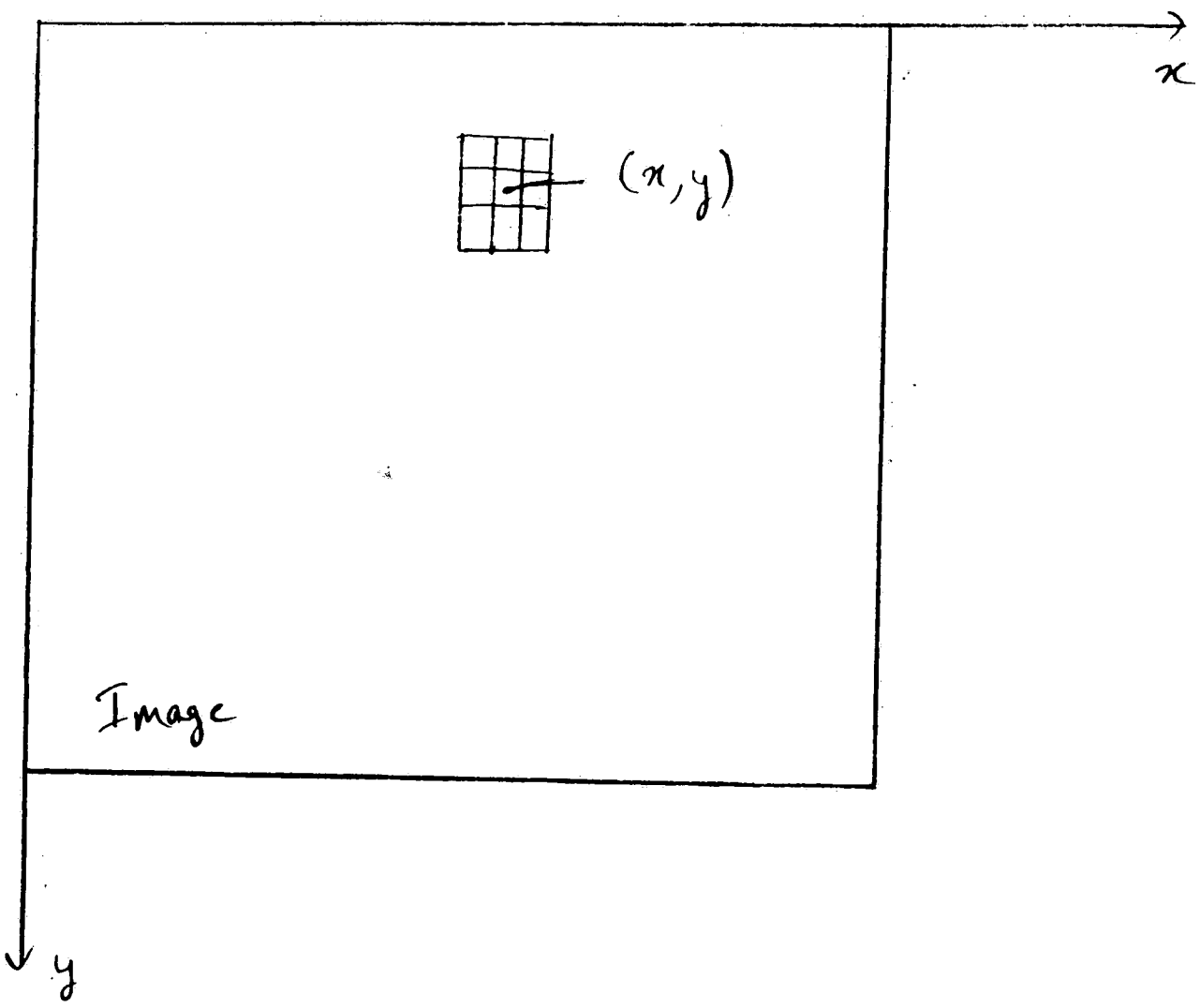


Figure 4.3 A 3X3 neighborhood about a point (x,y) in an image.

We may generalize the preceding discussion as performing the following operation :

$$\begin{aligned} T[f(x,y)] = {} & w_1 f(x-1,y-1) + w_2 f(x-1,y) + w_3 f(x-1,y+1) + \\ & w_4 f(x,y-1) + w5 f(x,y) + w6 f(x,y+1) + \\ & w_7 f(x+1,y-1) + w_8 f(x+1,y) + w_9 f(x+1,y+1) \end{aligned} \tag{4-2}$$

A 3 X 3 neighborhood of (x,y) is assumed. Larger masks are formed in a similar manner.

Now with a background on spatial convolution, we are ready to discuss the application of the second-derivative. The Laplacian operator as mentioned earlier will be used to enhance the edges in an image.

Laplacian Edge Enhancement Operator

The Laplacian operator used in image processing is based on the mathematical second partial derivative Laplacian expression in continuous functions.

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{4-3}$$

This continuous mathematical expression can be approximated by the difference operators in the discrete matching vision digital processing application. The rotationally insensitive Laplacian operator process essentially determines the change in slope of the intensity at the pixel in the x and y directions.[4]

$$L(i,j) = \nabla^2 p(i,j) = \Delta x^2 p(i,j) + \Delta y^2 p(i,j) \tag{4-4}$$

where

$$\Delta x^2 = [p(i-1,j) - p(i,j)] - [p(i,j) - p(i+1,j)] \tag{4-5}$$

$$\Delta y^2 = [p(i,j+1)) - p(i,j)] - [p(i,j) - p(i,j-1)] \tag{4-6}$$

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \qquad \Delta x^2 = [\ d - e\ ] - [\ e - f\ ]$$

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \qquad \Delta y^2 = [\ h - e\ ] - [\ e - b\ ]$$

The collection of terms will result in the following expression for the Laplacian operator.

$$L(i,j) = b + d + f + h - 4e \tag{4-7}$$

The Laplacian L(i,j) can be reduced to the mask

$$\begin{bmatrix} 0 & +1 & 0 \\ +1 & -4 & +1 \\ 0 & +1 & 0 \end{bmatrix}$$

In summary, the Laplacian operator computes the difference between the gray level of the center pixel and the average of the gray levels of the four adjacent pixels in the horizontal and vertical directions. It is a high pass filter because the sum of the coefficients is zero, and it contains both positive and negative coefficients.

4.2) SEQUENTIAL SEGMENTATION

In the segmentation methods described up to now, the processing that was done at each point of the picture did not depend on results obtained at other points. Thus, these methods can be regarded as operating on the picture "in parallel", ie, at all points simultaneously. They are implemented very efficiently on a suitable computer with a parallel architecture. This section deals withe segmentation methods in which we take advantage of the results obtained at previously processed points, in processing of a current point. In these inherently sequential methods, the processing that is performed at a point, and the criteria for accepting it as part of an object, can depend on information obtained from earlier processing of other points, and in particular, on the natures and locations of the points already accepted as parts of the object.

Sequential segmentation methods have a potential advantage over parallel methods, with respect to their computational cost on a conventional, sequential computer. In the parallel approach, the same computations must be performed at every point of the picture, since our only basis for accepting or rejecting a given point is the result of its own local computation. If we want our segmentation process to be reliable, these computations may have to be relatively complex. When using the sequential approach, on the other hand, we can often use simple, inexpensive computations to detect possible object points. Once some such points have been detected, more complex computations can be used to extend or track the object(s). The latter computations need not be performed at every picture point, but only at points that extend objects that have already been detected by the previous parallel segmentation methods.

EDGE AND CURVE TRACKING

Sequential methods are very flexible and can be defined in many ways. We will start by discussing the raster tracking method.

Raster Tracking

Suppose that the objects to be extracted from the given picture are thin, dark, continuous curves whose slopes never differ greatly from 90°. In this case, we can extract the objects by tracking them from row to row of the picture, as we scan the picture row by row in the manner of a TV raster. Specifically, in each row we accept any point whose gray level exceeds some relatively high threshold d; this is our detection criterion. In addition, once a point (x,y) on the yth row has been accepted, we accept any neighbor of (x,y) on the (y-1)st row, ie., we accept any of the points (x-1,y-1), (x,y-1), and (x+1,y-1), provided that the accepted points have gray level above some lower threshold t; this is our tracking criterion.

If we used the d threshold (detection) alone, or the t threshold (tracking) alone, the curves would not be extracted correctly; but when we combine the two thresholds, in conjunction with row-by-row tracking, we are able to extract the curves.

There are many modifications or extensions to this method. The detection and tracking acceptance criteria can involve local property values other than gray level. For example, they can be based on local contrast, as measured, for example, by the value of some derivative operator such as the gradient. The gradient defines not only a degree of contrast, but also a direction (of highest rate of change of gray level at the point). In tracking, one could look for successor points along the perpendicular direction, which should be the direction "along" the curve being tracked. The tracking criterion could also depend on a comparison between the current point (x,y) and its candidate neighbors, so as to discriminate against candidates whose gray levels, etc, do not "resemble" that of (x,y).

In some cases, we may want to accept all candidate points that satisfy the tracking criterion. In other cases, however, it may be preferable only to accept the best one(s). For example, it may turn out that accepting all of them would make a curve thick, which contradicts our a priori knowledge that curves are thin.

The tracking criterion should, in general, apply to some region below each currently accepted point (x,y), rather than just to the point's immediate neighbors on the row below it. Moreover, the criterion can depend on the candidate point's position in this region. For example, it can be more stringent for points that are farther from (x,y), so as to discriminate against large gaps in the curves, or it can be more stringent for points that are less directly below (x,y), so as to discriminate against obliquity.

Similarly, the tracking criterion should depend not just on a single currently accepted point (x,y), but on a set A of already accepted points. This makes it possible, for example, to fit a line or curve to the points in A, and make the criterion more stringent for candidate points that are far from this curve. In this way, we can discriminate against curves that make sharp turns. If desired, we can give greater weight to the more recently accepted points of A than to the "older" points.

Omnidirectional tracking

Up to now we have talked about tracking methods based on a single raster scan. This has the disadvantage that the results depend on the orientation of the raster, and the direction in which it is scanned. For example, if a strong curve gradually becomes weaker (lighter, lower contrast, etc.) as we move from row to row, our tracking scheme may be able to follow it, since the acceptance criteria for a curve that is already being tracked are relatively permissive; but if we were scanning in the opposite direction, so that the curve starts out weak and gets stronger, we might not detect it for quite a while, since our detection criteria are relatively stringent. We can overcome this problem by scanning the picture in both directions, carrying out our tracking procedure for each of the scans independently, and combining the results, but this doubles the computational cost, and the results may not "fit" well.

Further, raster tracking has the disadvantage that it breaks down for curves that are very oblique to the raster lines. For example, if we scan row by row, we cannot track curves that are nearly horizontal, since the crossings of successive rows by such a curve are many columns apart. One way to overcome this problem is to use two perpendicular rasters, eg., the rows and the columns, so that any curve meets at least one of the rasters at an angle between 45° and 135°. If we carry out detection and tracking independently for the two rasters, and combine the results, we should be able to track every part of the curve using at least one of the rasters. Not only does this approach double the computational cost, but it may also cause us to miss parts of the curve, since when it becomes too oblique to be traced by one raster, it will be picked up by the other raster only if it meets the more stringent detection criteria.

These directionality problems can be largely avoided if we use omni-directional curve-following, rather than raster tracking. Given the current point (x,y) on a curve that is being tracked, a curve-following algorithm examines a neighborhood of (x,y) , and picks a candidate for the next point. If the curves being tracked can branch or cross, it may be necessary to pick more than one next

point; in that case, all but one of the chosen points are stored for later investigation, and the tracking proceeds with one remaining point as next point.

The candidates for next point are evaluated on the basis of their satisfying a tracking criterion for acceptance (dark, high contrast, etc). They need not be immediate neighbors of (x,y), since we want to be able to tolerate small gaps in curves; but the tracking criterion should discriminate against points far from (x,y), since a connected curve would normally be preferable to one with gaps. They need not lie on the side of (x,y) directly opposite the already accepted points, since our curves need not be straight lines; but the criterion should discriminate against points that deviate too much from this direction, since a smooth curve is normally preferable to one that makes sharp turns. All of this is, of course, analogous to the raster tracking case, except that all the neighbors of the current point, not just those on some "next row," are candidates for being the next point.

At any stage, the next point must not be a point that has already been accepted at a previous stage. If there are no candidates that have not already been accepted, the tracking terminates. We can now go back to any points that were stored for later investigation, and resume tracking these, to find other curve branches. In particular, when we first detect a curve point, we track that curve in one direction until it terminates, and then go back if possible. When no points remain to be investigated, we can resume. These points must be on curves that do not cross the curves that we have already tracked. In principal, tracked edges should never branch, and should never terminate except by returning to the starting point or running off the edge of the picture.

Omnidirectional tracking, as seen, is much more flexible than raster tracking, but it has the disadvantage that it requires access to the points of the picture in a nonprespecified order, as determined by the shapes of the edges or curves being tracked. It is sensitive to noise and does not fail in a graceful way.[5]

Before we end the discussion on sequential segmentation, we introduce a more compact form of representing the boundary pixels, namely chain coding.

Chain codes

A curve in a digital picture can be represented by a sequence of points $\{(xi,yi)\}$ or, more compactly, by its chain code. The chain code is a slope intrinsic representation for the curve. Typically, this representation is based on the 8-connectivity of the segments, where the direction of each segment is coded using a numbering scheme such as one shown in Figure 4.4. Given any pair of consecutive points on the curve, (x_i,y_i), (x_{i+1},y_{i+1}), there are only eight possible locations for (x_{i+1},y_{i+1}) relative to (x_i,y_i), so that the curve can be represented by a sequence of direction changes. An example of a digital curve and its corresponding chain code is illustrated in Figure 4.5. The starting point is circled and the digital pattern is scanned clockwise to obtain the chain code. The series of coordinates can be obtained back from the chain code without any loss of accuracy.

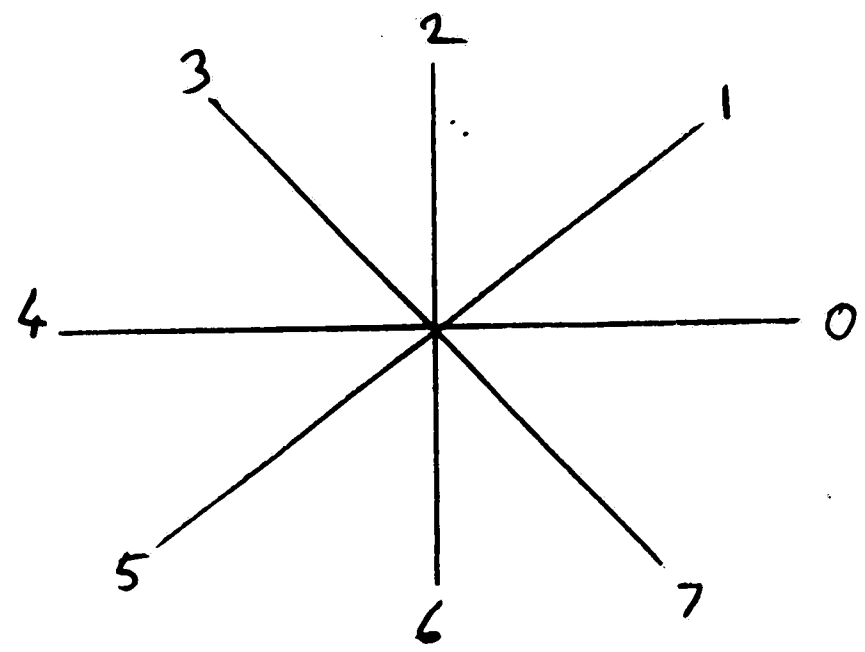
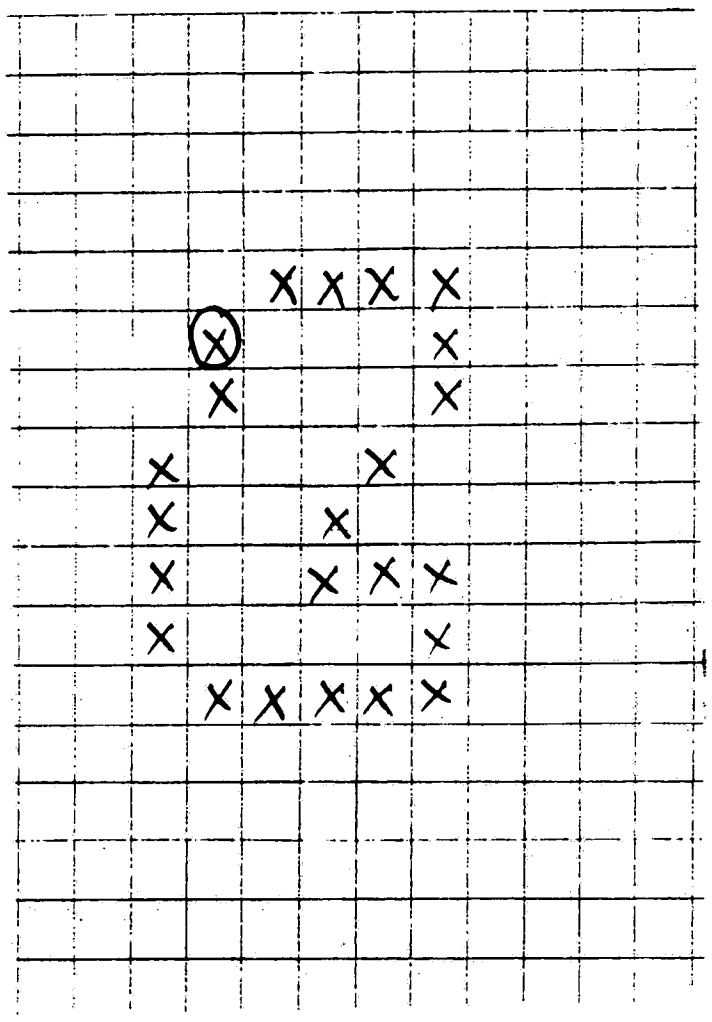


Figure 4.4 Numbering scheme for the chain code



CHAIN CODE: 1 0 0 0 6 6 5 5 6 0 0 6 6 4 4 4 4 3 2 2 2 1 2

Figure 4.5 Digital curve along with its chain code

CHAPTER 5

SHAPE ANALYSIS

The analysis of the shape of features is the basis for recognizing objects or in making measurements of their sizes and shapes. The boundary of a feature is the best known representation of the shape of an object. The purpose of all the previous methods was to extract the borders of the object(s) of interest from the given image. Therefore, the following material assumes that the given digital curve or border is specified by either a series of coordinates or a chain-code. All of the following discussion is aimed at getting a measure of shape. We will start with some properties involving slope and curvature of digital curves, which will be used later in the discussion involving curve segmentation.

5.1) CORNER DETECTION ON DIGITAL CURVES

Slope and Curvature

The shape of a border can be analyzed by examining the turns it makes as we move along it. When we follow a border, we are always moving at an angle that is a multiple of 45°, which is the slope of a chain code at any point. In order to measure a more continuous range of slopes, we must use some type of smoothing. We can define the left and right k-slopes at a point P as the slopes of the lines joining P to the points k steps away along the curve on each side.

Curvature is ordinarily defined as rate of change of slope. Here again if we use difference of successive unit slopes, we always have a multiple of 45° for a chain code. To obtain a more continuous range of values, we must again use smoothing. We can define the k-curvature of P as the difference between its left and right k-slopes. Ofcourse, k-slope is not defined if P is less than k away from an end of an arc. The choice of k depends on the application.

As an example, consider the arc in Figure 5.1 and assume it to be continued indefinitely in both directions. Here the right 1-slope is 0° at some points and 45° at others, but the right 3-slope is $\tan^{-1}(1/3)$ at every point (this is because the arc is periodic with period 3). For $k > 3$, the right k-slopes are not all equal, but as k increases, they all approach $\tan^{-1}(1/3)$. Similarly the k-curvature fluctuates for small k (except that they are all 0 for $k = 3$) and approach 0 as k gets large.

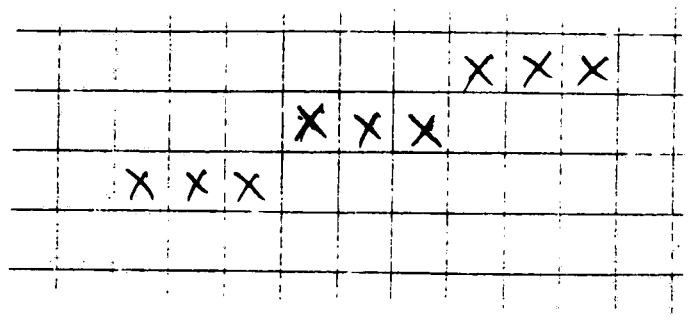


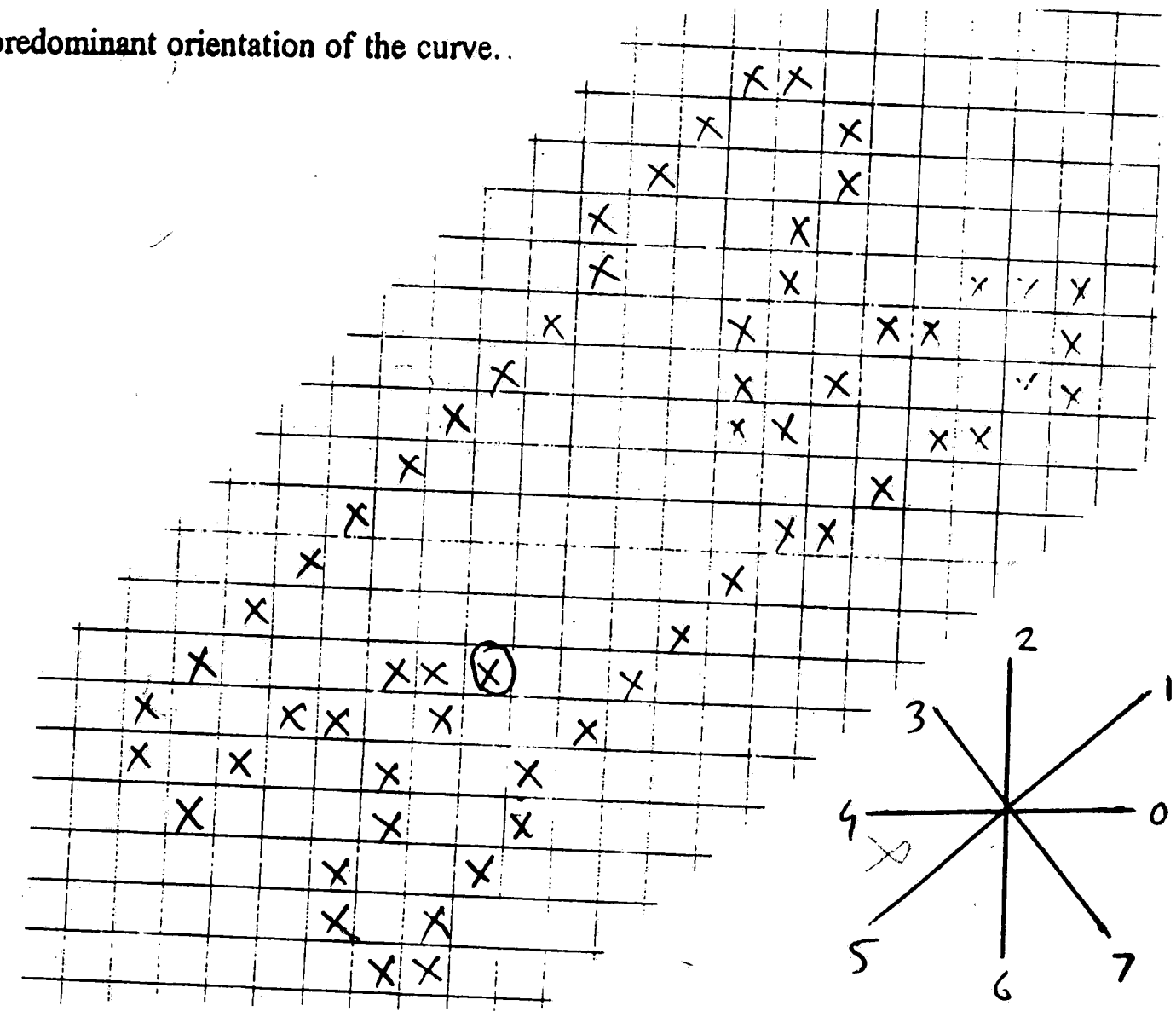
Figure 5.1 Arc showing digital straightness

Curve Segmentation

An important visual descriptor of object shape is the corners the feature shows. Simplified line drawings in which all corners are preserved, but are joined by straight line segments, seem to convey much the same information as the original drawings. This tells us that corners tend to be significant information conveyors in images. Sharp corners are often associated with man-made objects - a road intersection, a house, a cultivated field, but in natural scenes, corners reveal intersections between different objects or surfaces. For this reason, it is interesting to attempt to characterize shape by the corners it possesses. First we will briefly talk about other less common tools in connection with curve segmentation.

The slope histogram of a curve tells us how often each slope occurs on the curve. Ofcourse, it does not tell us how these slopes are arranged along the curve. A squiggle consisting of equal number of

horizontal and vertical segments may have the same slope histogram as a square. However, for noncomplex curves it is reasonable to assume that a peak on the slope histogram provides information about overall orientation. Figure 5.2 shows a closed curve along with its chain code. Figure 5.3 shows this curve's histogram of 1-slopes. The two peaks at directions (1) and (5) which are 180° apart, correspond to the predominant orientation of the curve..



Chain Code: (starting point is circled , scanning anticlockwise)

55656 70211 21111 10110 10224 45455

42212 12345 55655 55555 55671 10100

Figure 5.2 Closed curve along with its chain code

Information about the wiggliness of a curve can be obtained from its curvature histogram. For a smooth curve, the curvatures will be concentrated near 0, whereas for a wiggly curve they will be more spread out. The 1-curvature histogram for Figure 5.2 is shown in Figure 5.4.

Slope	:	0	1	2	3	4	5	6	7
Fequency	:	7	15	8	1	5	18	4	2

Figure 5.3 Histogram of 1-slopes for the curve in Figure 5.2

Difference	:	-3	-2	-1	0	+1	+2	+3
Frequency	:	1	1	13	23	18	4	0

Figure 5.4 Histogram of 1-curvatures for the curve in Figure 5.2

The slope histogram and the curvature histogram described above can be used to obtain some knowledge about the complexity of the shape of an object. They give no specific information about the locations and hence cannot be used alone in segmenting a curve. The idea behind segmenting a curve is to locate portions of the object's shape on its periphery whose properties are already known. Once located, these portions and their neighborhoods can be further analyzed to get a measure of the orientation of the object in the image or to see if it is of acceptable quality as required by the application. Locating corners is a good starting point. Therefore, it is important to have a reliable technique that finds corners on digital patterns which is the topic of the following discussion.

Information on a contour is concentrated at points having high curvature. Any computer model of shape perception should include a component that detects angles. A digital model for detecting and locating points of high curvature on a digital curve is essential. By a 'corner' we mean the chain node with which we can associate an identifiable discontinuity in the mean curvature of the curve. The detection of a corner

is a function of the magnitude of the discontinuity, its abrptness, and the curve regions on either side of it over which the mean curvature can be considered to be uniform and free of discontinuities.

In the real Euclidean plane, curvature is defined as the rate of change of slope as a function of arc length. For the curve y = f(x), this can be expressed in terms of derivates as

$$\frac{\frac{d^2y}{dx^2}}{(1+(\frac{dy}{dx})^2)^{\frac{3}{2}}} \qquad (5\text{-}1)$$

If we deal with digital curves, however, it is not immediately clear how to define a discrete analog of curvature. Suppose that a digital curve is defined as a sequence of integer coordinate points p_1, p_2, ..., p_n, where p_{i+1} is a neighborhood of p_i (modulo n), $1 \leq i \leq n$ and n representing the total number of points defining the perimeter of the curve. Thus if $p_i = (x_i, y_i)$, we have $|x_i - x_{i+1}|$ and $|y_i - y_{i+1}|$, both less than or equal to 1 , but not both 0. If we define the curvature at pi by simply replacing the derivatives in Eq. (5-1) by differences, the difficulty arises that successive slope angles on the digital curve can only differ by multiple of 45°, so that small changes in slope are impossible.

Most features observed in real images have 'corners' that are quite variable. Some measure of how prominent a corner is must be used along with the means of detecting them. This difficulty can be reduced by using a smoothed slope measurement, eg., defining the slope at p_i as $(y_{i+k} - y_i)/(x_{i+k} - x_k)$ for some $k > 1$, rather than simply using the first difference (ie $k = 1$). But it is not clear how to choose the smoothing factor k.

We describe a procedure for defining significant curvature maxima (and possibly also points of inflection) on a digital curve, using a variable degree of smoothing. This is a 'parallel' procedure in the sense that the results at each point do not depend on results previously obtained at other points.[6] [7]

We define the k-vectors at p_i as

$$a_{ik} = (x_i - x_{i+k},\ y_i - y_{i+k})$$

$$b_{ik} = (x_i - x_{i-k},\ y_i - y_{i-k})$$

and the k-cosine at p_i as

$$c_{ik} = \frac{(a_{ik}.b_{ik})}{|a_{ik}|\ |b_{ik}|} \qquad (5\text{-}2)$$

This c_{ik} is the cosine of the angle between a_{ik} and b_{ik}. Thus $-1 \leq c_{ik} \leq 1$, where cik is close to 1 if a_{ik} and b_{ik} make an angle near 0°, and c_{ik} is close to -1 if a_{ik} and b_{ik} make an angle near 180°; in other words, c_{ik} is larger when the curve is turning rapidly, and smaller when the curve is relatively straight.

We next describe how to select an appropriate value of k at each point p_i of the curve. We begin by computing $c_{i1}, c_{i2}, \ldots, c_{im}$, where m can be arbitrarily chosen to be n/10, ie 1/10 of the perimeter of the curve. This choice can vary depending on the application. Let h be such that

$$c_{im} < c_{i,m-1} < \ldots < c_{ih} \not< c_{i,h-1}$$

(where we can define $c_{i0} = -1$ to insure that there always is such an h between m and 1). We call cih the cosine at p_i, and denote it by c_i. Finally , we say there is a curvature maximum at p_i if

$c_i \geq c_j$ for all j such that $|i-j| \leq h/2$.

This results of applying this procedure can be seen in Figure 5.5. The corners on the digital pattern of the chromosome are indicated by arrows.

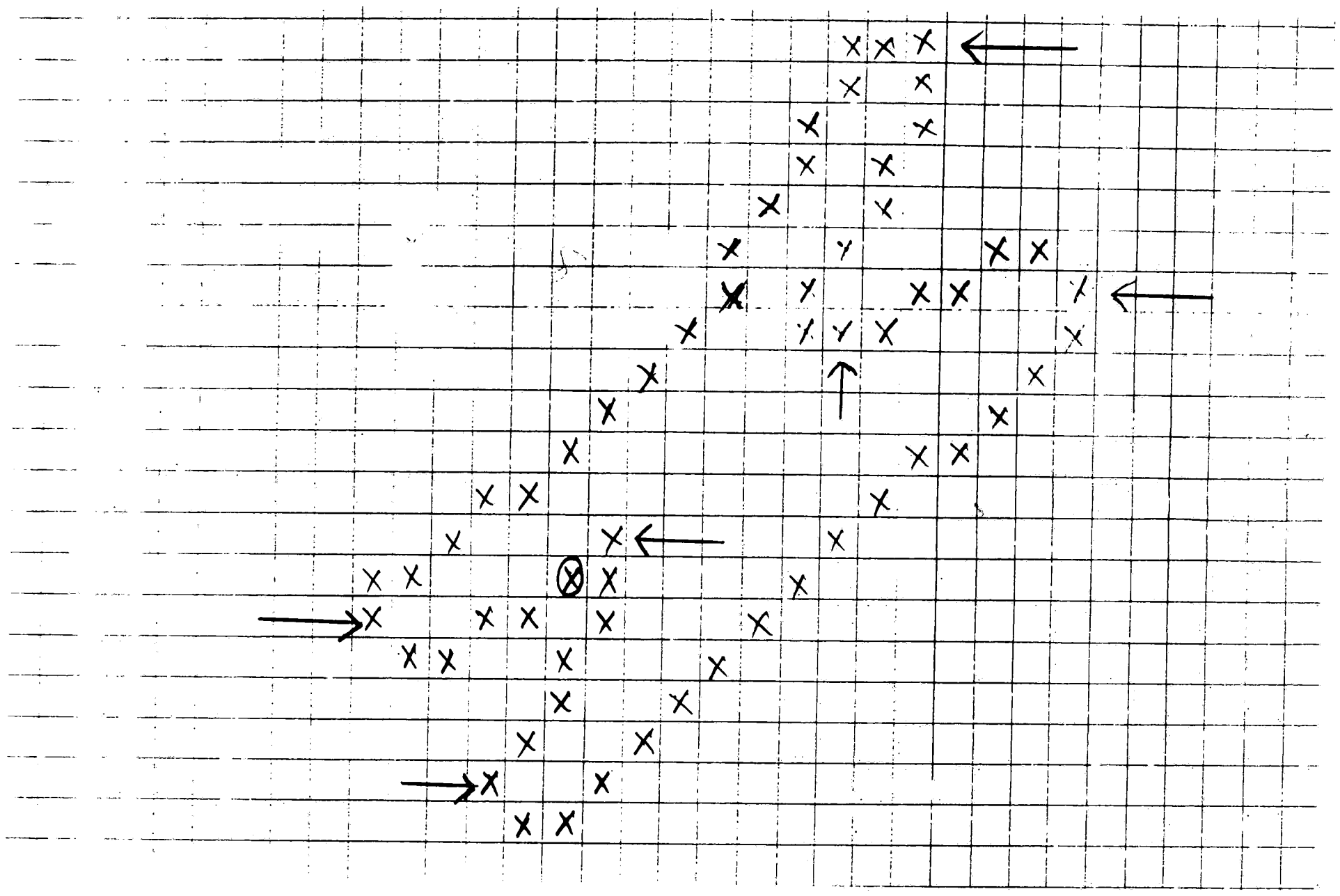


Figure 5.5 Digital Pattern of a chromosome showing the corners detected (indicated by arrows)

5.2) SHAPE REPRESENTATION OF TIPS

It is necessary to represent the shape of a tip by a set of numbers, in order to arrive at a conclusion about the quality of the tip. This is done through attribute measurement. The goal is to define a specific

set of attributes which will distinguish good tips from bad ones. These attributes are constructed by experimenting with needles which are known to have good or bad tips. By examining the data results from each attribute, its effectiveness in detecting bad tips can be analyzed. Some attributes work better than others in detecting certain type of defects. Based on the frequency of occurrence of particular defects, the order of applying each attribute can be prioritized. It is advantageous to first apply those attributes that identify the more common defects, particularly if their calculations are complex or require a lot of computation time. Speed of response from an inspection system is very critical in real-time applications. By calculating and comparing the attribute values for the tip under inspection with predefined values, an accept or reject signal can be given. Before we start defining the set of attributes used for the tips, we will first describe a statistical tool used to fit straight lines through a set of data points.

Regression

Regression analysis is typically performed by means of the least-square technique. It is used to determine a linear relationship between two variables x and y. In our application, this data represents the series of coordinate pixels along the boundary of a feature outline. Consider the sample data in Figure 5.6. The linear relationship between x and y is determined by finding the straight line that best fits the set of data presented in this figure. One way to do this would be to fit the line to the data 'bye eye'. Different analysts would come up with different lines. To overcome this variation in judgement, we use the least squares criterion. This criterion states that the "best fit" is the line that minimizes the sum of the squared errors between data points and the line. To elucidate this statement, the equation for a straight line is

$$y = A + Bx$$

where B = a constant representing the slope of the line

A = a constant representing the intercept of the line with the y-axis (at x = 0)

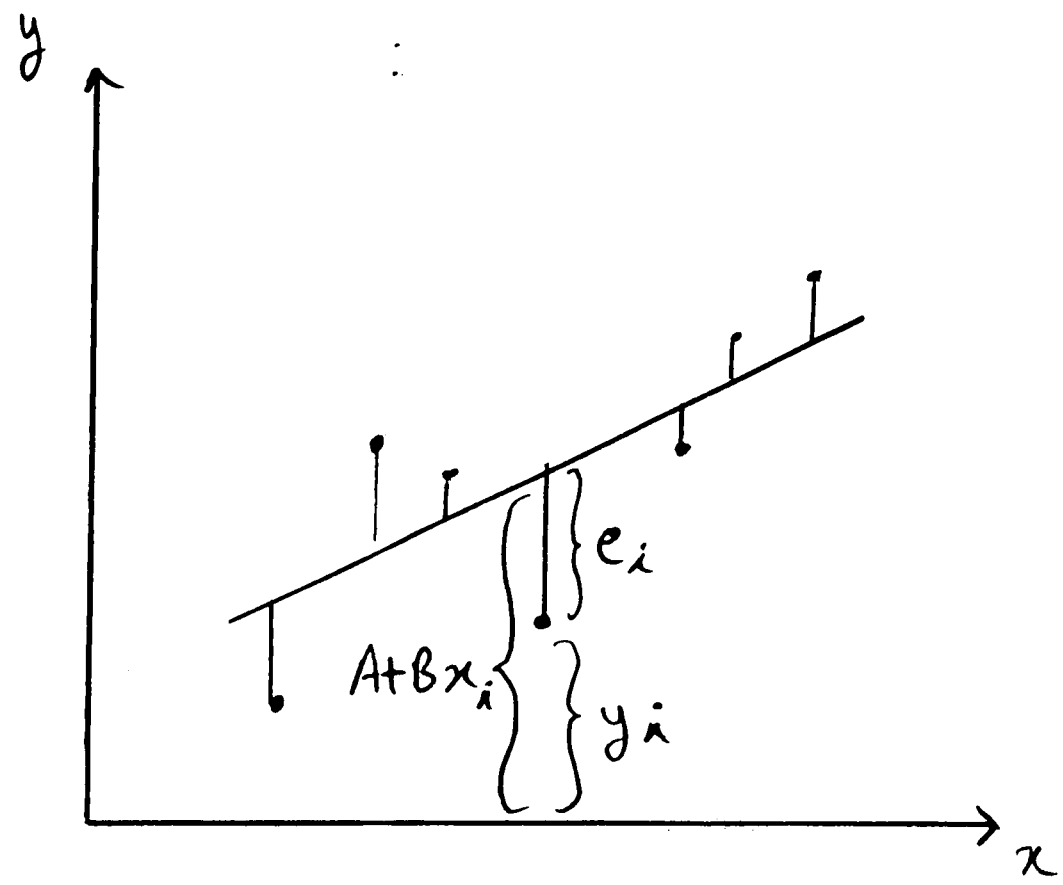


Figure 5.6 Plot showing "best-fit" line through set of data points

One possible line that follows this general equation is shown in Figure 5.6. The vertical distance between the straight line and any given data point (defined by x_i,y_i) is termed the error e_i. That is,

$$e_i=y_i-(A+Bx_i) \tag{5-3}$$

Mathematically expressing the criterion of least squares, the objective is to find values of A and B that

$$\textit{minimize} \quad \sum_{i=1}^{n} e_i^2=\sum_{i=1}^{n} [y_i-(A+Bx_i)]^2 \tag{5-4}$$

where n is the number of points in the data set.

To find the values of A and B that satisfy this objective, the partial derivatives of Eq. (5-4) with respect to A and B can be found and set equal to zero.

$$2\sum_{i=1}^{n}[y_i-(A+Bx_i)](-1) = 0 \qquad (5\text{-}5a)$$

$$2\sum_{i=1}^{n}[y_i-(A+Bx_i)](-x_i) = 0 \qquad (5\text{-}5b)$$

These equations can be written in a more convenient form by separating the terms being summed as follows:

$$\sum_{i=1}^{n} y_i = An + B\sum_{i=1}^{n} x_i \qquad (5\text{-}6a)$$

$$\sum_{i=1}^{n} x_i y_i = A\sum_{i=1}^{n} x_i + B\sum_{i=1}^{n} x_i^2 \qquad (5\text{-}6b)$$

Equations (5-6a) and (5-6b) are called the normal equations. By solving these two equations simultaneously, the values of A and B can be computed which provide the "best fit" to the data by the least-squares criterion. It is desirable to examine some of the statistical characteristics of the regression equation which indicate how good the relationship is. Among these characteristics is the standard error of estimate (se) .

The se statistic for a regression equation gives the average value of the e_i errors. It is not an arithmetic average, but rather a root-mean-square average corrected for the number of degrees of freedom. It is defined as

$$se = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n-2}} = \sqrt{\frac{\sum_{i=1}^{n} [y_i - (A + Bx_i)]^2}{n-2}} \qquad (5\text{-}7)$$

The squared term inside the radical sign shows the root-mean-square averaging process. The (n-2) term is the number of degrees of freedom. A more convenient way of representing se for calculation is given by:

$$se = \sqrt{\frac{S_{xx}S_{yy} - (S_{xy})^2}{n(n-2)S_{xx}}} \qquad (5\text{-}8)$$

where

$$S_{xx} = n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2$$

$$S_{yy} = n\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2$$

$$S_{xy} = n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)$$

The sign of a "good fit" is when the value of se is low relative to the values of y in the data set. If se = 0, this indicates that the equation fits the data perfectly.

SHAPE ATTRIBUTES FOR TIPS

As we mentioned in the beginning of this section, each tip under inspection will be associated with a set of attributes whose values will indicate the quality of that tip. We will define and analyze each attribute.

Linearity of tip edges

Ideally the two edges that taper to form the tip should be straight lines. This attribute checks to see if there are any abnormalities along a portion of the edges on either side of the tip. Any sharp nicks, cuts or dents on the edges will affect the linearity of the lines. The procedure is to take (n) of points on either side of the tip. So we will have two sets of data points, one for the top edge $\{ (x_{t1},y_{t1}) (x_{t2},y_{t2}) \ldots (x_{tn},y_{tn})\}$ and one for the bottom edge $\{ (x_{b1},y_{b1}) (x_{b2},y_{b2}) \ldots (x_{bn},y_{bn}) \}$. The value of n can be chosen according to the size of the tip or its magnification. The first point in both the above sets is the same since it represents the coordinate of the tip pixel. Therefore, $x_{t1} = x_{b1}$ and $y_{t1} = y_{b1}$. Next, using Eq. (5-8), we find the standard error of estimate of the top edge (se_{top}) for the n points of the bottom data set. We also find the standard error of estimate of the top edge (se_{bot}) for the n points of the bottom data set. Next, se_{top} is compared with y_{tavg} and sebot is compared with y_{bavg}.

where

$$y_{tavg} = \frac{\sum_{i=1}^{n} y_{ti}}{n}$$

and

$$y_{bavg} = \frac{\sum_{i=1}^{n} y_{bi}}{n}$$

To make this comparison task easier and to arrive at a single value for the linearity of the edges, we find the percentages (T_{per} and B_{per}) of the standard error of estimates (se_{top} and s_{ebot}) with respect to the average value of y (y_{tavg} and y_{bavg}) as follows

$$T_{per} = (se_{top} \text{ X } 100) / y_{tavg}$$

$$B_{per} = (se_{bot} \text{ X } 100) / y_{bavg}$$

Finally, T_{per} and B_{per} give a measure of the "goodness of fit" of the straight lines through the two data sets. If T_{per} and B_{per} are high, this indicates that there are some imperfections along the edges. If the top edge is not linear, T_{per} will be high. If the bottom edge is not linear, B_{per} will be high.

Tip Offset

Here we will predict the location of the tip pixel by assuming that it is a perfect tip. Then the deviation of this predicted location of the tip pixel from the actual location will be given by the tip offset. By examining the bent tip in Figure 5.7 , it can be seen that the value of this offset should be higher for a bent tip relative to a good tip.

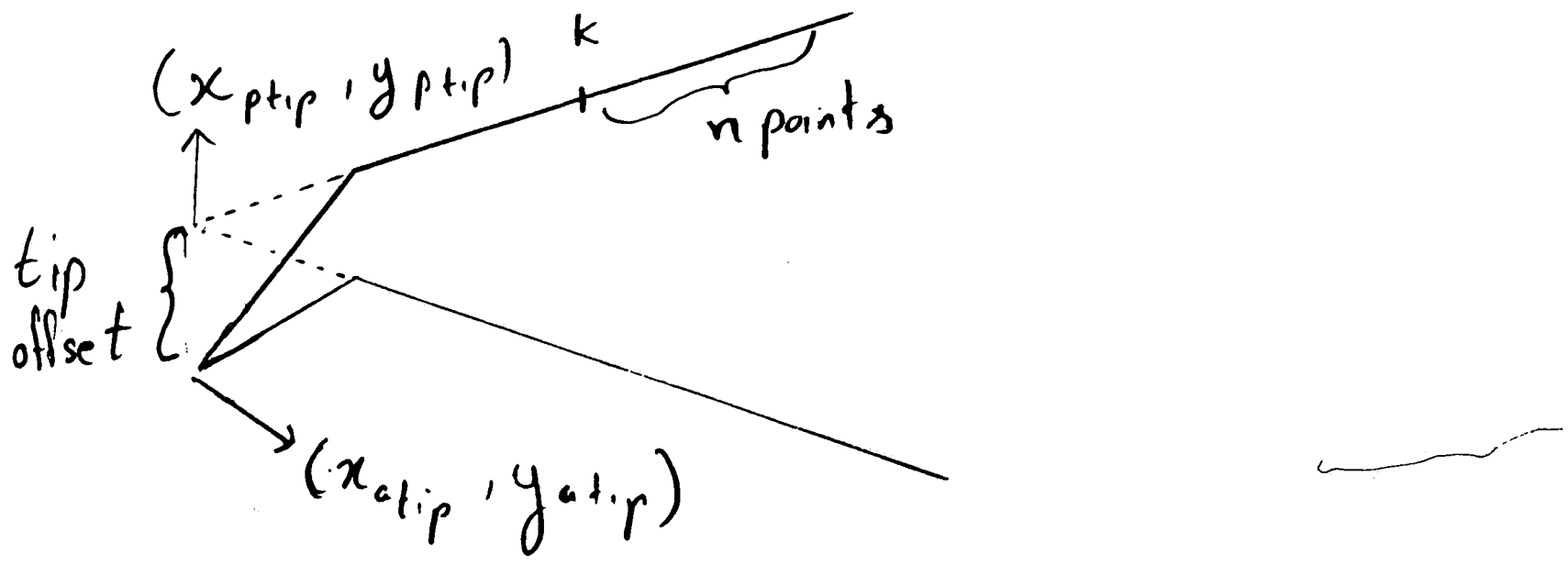


Figure 5.7 Bent tip illustrating the calculation of tip offset

The procedure is to take n points starting from the kth point on either side of the tip (k should be chosen large enough to ensure that it does not fall in the defective area of the tip). Using Equations (5-6a) and (5-6b), we 'best fit' straight lines to both the data sets that correspond to the top and bottom edges. Let the equation of the top fitted straight line be given by

$$y_{top} = A + Bx \qquad (5\text{-}9a)$$

and the one for the bottom line be given by

$$y_{bot} = C + Dx \qquad (5\text{-}9b)$$

The predicted location of the tip is the intersection of the two lines in Equations (5-9a) and (5-9b). The coordinate of the predicted tip (x_{ptip}, y_{ptip}) can be calculated as follows

$$x_{ptip} = (C - A) / (B - D)$$

$$y_{ptip} = A + B\, x_{ptip}$$

Assuming that the coordinate of the actual location of the tip under inspection is (x_{atip}, y_{atip}), the offset is given by

$$offset = \sqrt{(x_{atip} - x_{ptip})^2 + (y_{atip} - y_{ptip})^2} \qquad (5\text{-}10)$$

The value of this offset will be higher for bent tips relative to the offset for good tips.

Measure of the bend angle

If we examine the bent tip in Figure 5.8, it is seen that each line on either side of the tip, instead of being straight, makes an angle which is less than 180° at some point. The value of this angle is another indication of the magnitude of the defect. The procedure to detect this angle is the same for the top and bottom edges of the tip. The angles whose values are to be found are indicated by alpha and beta in Figure 5.8.

We take (n) points starting from the tip coordinate. Using Equations (5-6a) and (5-6b), we best-fit a straight line through these points and get vector a pointing towards the tip, as shown in Figure 5.8. We repeat this procedure for another (x) points starting from the kth point and get vector b pointing away from the tip (again k should be chosen large enough to avoid the defective region; the choice of n and x is

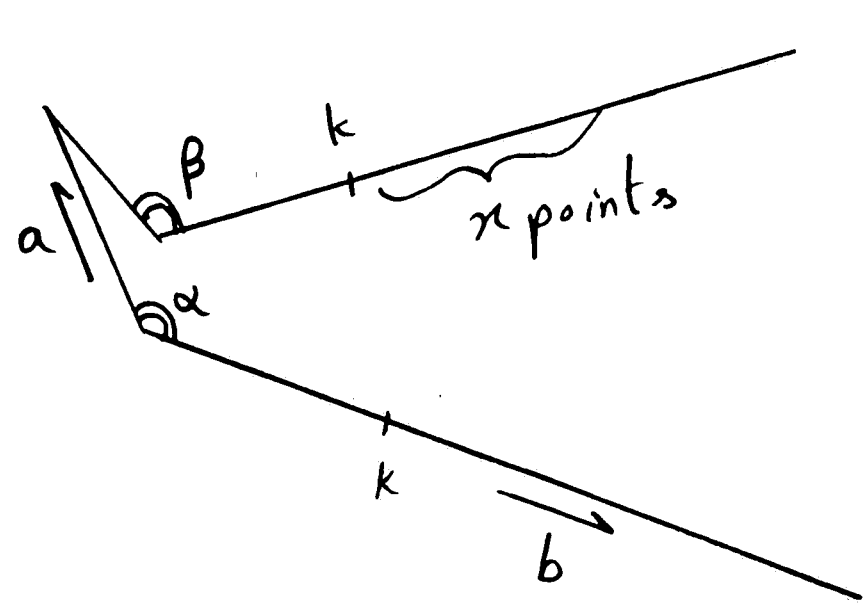


Figure 5.8 Bent tip illustrating the measure of bend angle

dictated by the size of the tip, type of defects and magnification). Then, the cosine of the angle between these two vectors can be found as follows:

$$cosine = \frac{(a.b)}{|a|.|b|}$$

where $-1 \leq cosine \leq 1$. The value of cosine gives a measure of the amount of bend on the edges. If the value of the cosine is close to -1, it indicates there is no angle smaller than 180° and the edges are straight.

Defective Pixel Count

Last but not least, is an extension to the procedure described earlier in calculating the attribute - tip offset. First, we "best-fit" straight lines to the top and bottom edges that join to form the predicted location of the tip. Then we look for the area that lies outside the region enclosed by these two "best-fit" lines. By examining Figure 5.9, it is seen that if a tip is defective there will always be a region lying outside the two lines that join to form the predicted tip location. The shaded portions the defective tips shown in Figure 5.9 represent the defective areas that we are looking for. By counting the edge pixels of the tip that fall in these defective regions, we get a good measure of the degree of deformity of the tip. This procedure is described in the following paragraph.

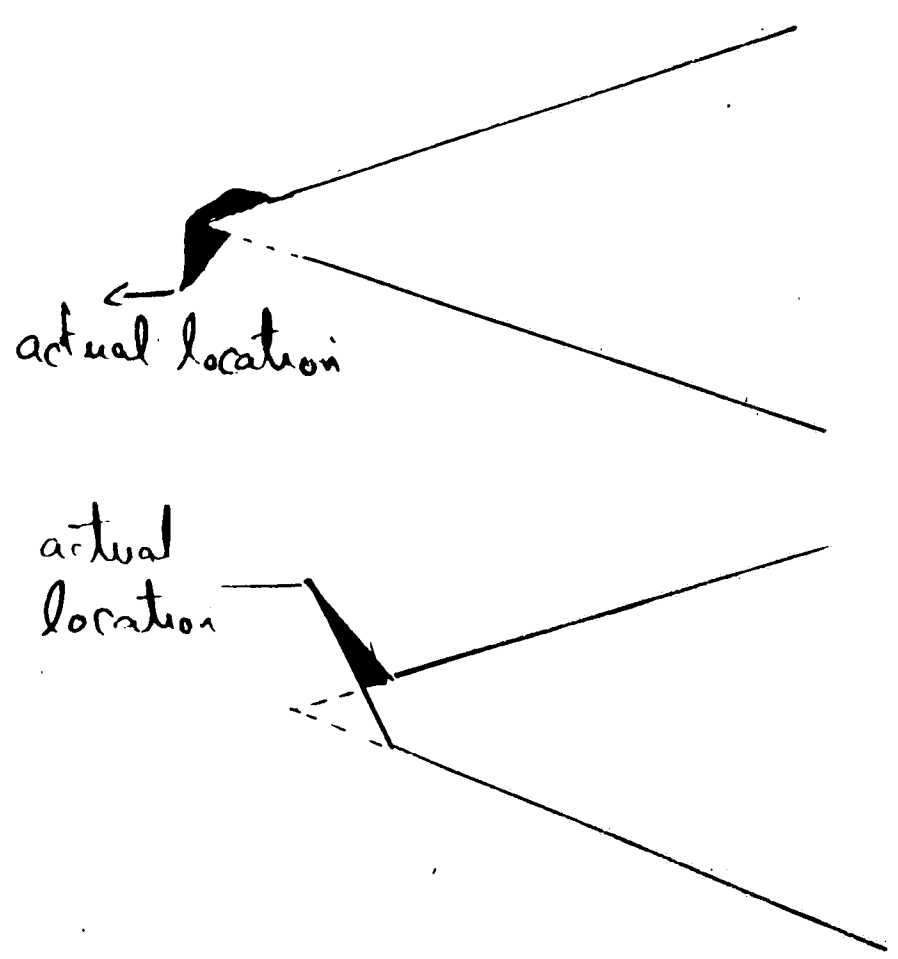


Figure 5.9 Deformed tips illustrating the calculation of Defective Pixel Count

We arrive at Equations (5-9a) and (5-9b) representing the top and bottom "best-fit" lines similarly as described in the procedure for calculating the tip offset. Our task now is to count the number of edge pixels that fall outside the area enclosed by the two lines in Equations (5-9a) and (5-9b). This means that,

starting from the actual tip pixel (x_{atip},y_{atip}), we scan n edge points on either side of the tip to see if they lie outside the region enclosed by the predicted lines. We count the number of edge - points that lie above the line in Eq. (5-9a), representing the top edge. We also count the number of edge pixels that lie below the line in Eq. (5-9b), representing the bottom edge. The total count will be higher for a deformed tip relative to the count for a good tip. The value of the count is dependant on the size of the deformity.

5.3) FRACTAL DIMENSIONS

In applications involving inspection, it is often necessary to determine the amount of roughness or irregularity on feature profiles. The roughness on profiles could be a result of certain type of defects occurring on the objects undergoing inspection that have to be detected. The use of fractal geometry for describing irregular shaped objects in automated image analysis will be discussed in this section. In order to understand the concept of fractal dimensions, we will first talk briefly about self-similarity.

Self-Similarity

In nature, most boudaries are not straight lines forming a polygon, or simple smooth curves, but are instead rough and irregular. Furthermore, the amount of roughness and irregularity that we see is generally limited by our image resolution, and if we could increase the magnification, the amount of perimeter we would see in the image would increase. There seems to be no limit to the increase in detail made visible through increased magnification. This particular type of behaviour that is of most interest to us here has a peculiar form called self-similarity.

Self-similarity simply means that at any magnification at which we view a line or surface, it looks the same. Whatever measurements we can make to describe the roughness and its scale will be independant of the scale. Another way of understanding this is if one were to take a portion of the perimeter of a feature and look at it under a microscope, the magnified portion would look exactly the same as the original

large part of the boundary. This means that if we were to look at a picture of a portion of an ideal fractal boundary, we would not know the magnification of the photograph from the ruggedness of the boundary, since it always looks the same at every magnification. This 'look alike' feature of an ideal fractal boundary at various magnifications is described as self-similarity. When the complexity of the structure increases with magnification, it is useful to use fractal dimensions to describe the structure of the particle. Therefore, the procedure for determining the fractal dimension of a feature outline will be the subject of the following discussion.[8] [9]

Concept

The length of the boundary is determined by swinging along the boundary with dividers set at some arbitrary distance. Then if the divider distance, or stride length, is reduced, and the same operation repeated, the measured length is greater because the measurement is able to follow more of the irregularities of the profile. Repeating the operation with finer and finer steps will cause the length to continuously increase, so that in effect, the length of the outline would be expected to become infinity at a fine enough scale. Furthermore, it is noted that over a considerable range of stride lengths, and for a variety of borders, the slope of the plot of measured length versus stride length is constant on a log scale. If the increase in measured length with improvement in measuring resolution is uniform (a straight line on a log plot), the feature is said to be self-similar. The fractal dimension is then obtained from the slope of this plot. This procedure can be illustrated by exploring the simpler, randomly drawn rugged profile shown in Figure 5.10.

Illustration

A pair of compasses is used to construct a polygon of side length (λ) by walking the compasses around the profile. Thus, as a start of the procedure the compass point is placed at point A. One now swings the compasses from outside of the profile until it makes contact with the profile. This contact point becomes the reference point for drawing a straight line of length (λ). The compass point is then moved to put the

pivot on point B, and the next stride is taken along the outline to make contact at point C. One now 'walks' around the profile until one is now almost back at the starting point. Usually, the perimeter thus measured is not a perfect integral multiple of the stride length, and some partial length is left over at the end. A simple and unbiased method of completing the polygon is to join the point representing the last complete stride to the starting point. The length of this short side needed to complete the polygon is then taken to be a fraction (α) of the stride length (λ). The polygon drawn around the profile is now taken to be the estimate of the perimeter P, at the resolution (λ).

$$P = n\lambda + \alpha\lambda \tag{5-11}$$

When plotting graphs from the data generated in exploring the profile, it is useful to convert the perimeter P and the stride length (λ) to a dimensionless form by dividing both by some reference length. One useful reference length for converting the perimeter and stride length used to explore the rugged profile to a dimensionless form is the maximum projected length of the profile (L). This quantity is shown in Figure (5.10). This process of converting quantities such as 'length of a profile' into dimensionless form by dividing by a reference length is described as normalization of the variable. Once we have normalized the perimeter and stride lengths, we can compare the shapes of different profiles without being worried about the actual units. If the perimeters (P) are plotted against the stride lengths (λ) on a log-log scales, we obtain the plot shown in Figure 5.11. Once the log plot has been obtained, the fractal dimension (δ) of the boundary is obtained from m, the slope of the linear portion of the plot as

$$\delta = 1 + |m| \tag{5-12}$$

The quantitative evaluation of boundaries is a complex problem which involves not only the magnitude of the fractal dimension (δ) used to descibe the system but also the range of resolution (λ) over which that fractal dimension is an operative description parameter. Usually the magnitude of the fractal dimension increases with roughness, ie the slope of the plot is higher for rougher profiles. Thus, keeping the same magnification, the fractal dimension can be used to characterize the complexity of the profiles which in turn is useful in determining the quality of the object under inspection.

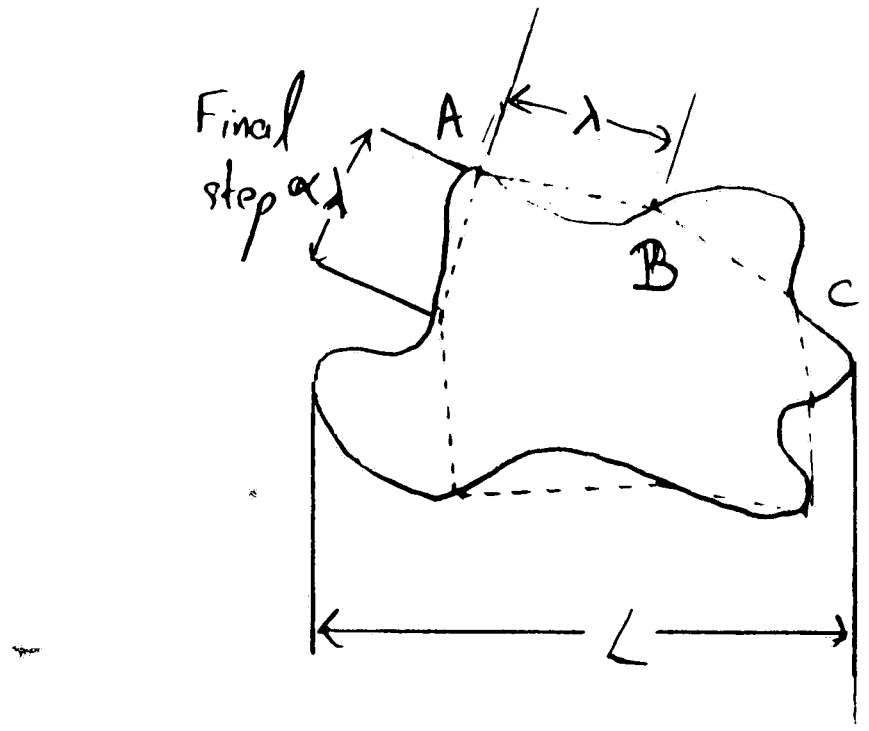


Figure 5.10 Rugged profile to illustrate the calculation of fractal dimension

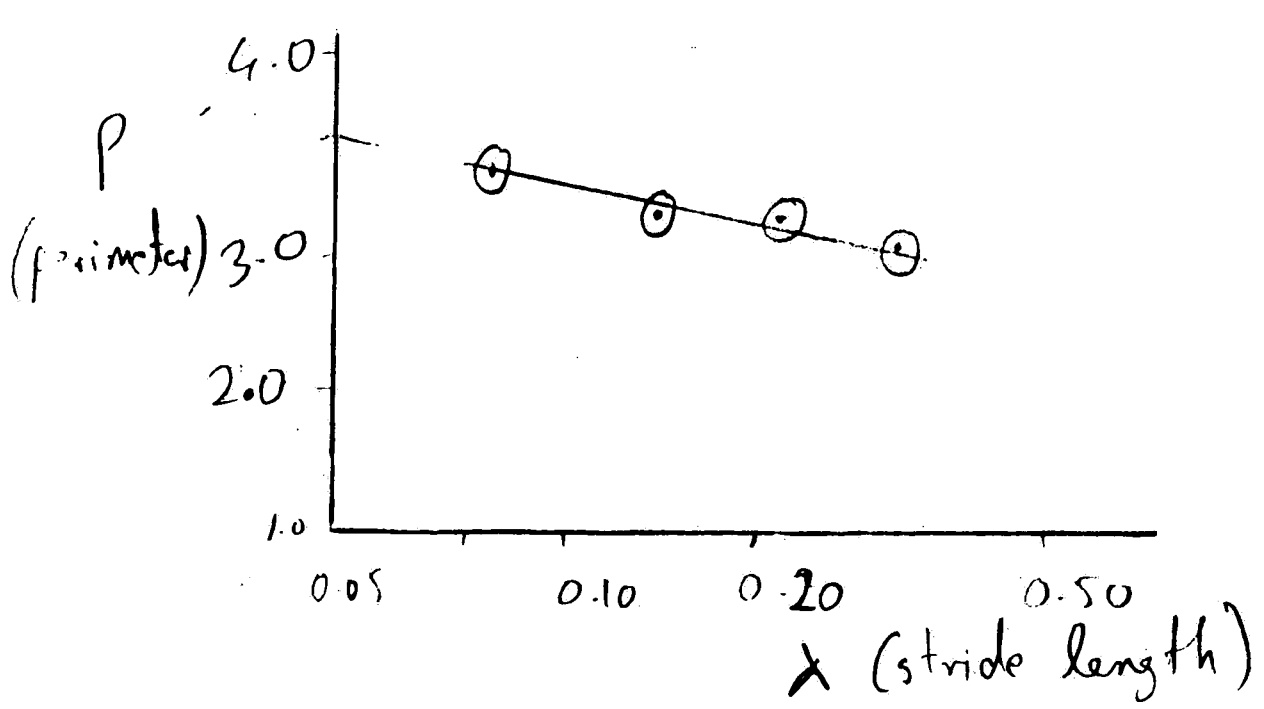


Figure 5.11 Plot of stride length vs perimeter

CHAPTER 6

HARDWARE OVERVIEW OF THE AUTOMATED SYSTEM

6.1) MACHINE VISION HARDWARE

A machine vision system is comprised of all the elements necessary to obtain a digital representation of a visual image, to modify the data, and to present the digital image data to the external world. Figure 6.1 shows the block diagram of the hardware components of a machine vision system. Each one of these components will be discussed in this section. [10]

Illumination

Lighting is a critical aspect of any machine vision application. Choosing the proper lighting scheme can result in increased accuracy, system reliability and response time. A poorly designed illumination system can produce glare which may saturate the camera, shawdows which may hide defects, low contrast or uneven illumination making the inspection task unnecessarily difficult to perform. The four most commonly used forms of vision system lighting are back lighting, front lighting, structured lighting and strobe lighting. Our application used ring light front lighting. Front lighting employs light reflected from the object. The illumination source and the camera are both on the same side of the object. Ringlights provide intense shadow-free lighting and are especially useful when imaging specular (shiny) objects like needles.

Sensor

In the context of machine vision systems, a sensor refers to an electro-optical device which converts an optical image to a video signal. A charge couple device (CCD) was used for our application, which is a type of solid-state matrix camera. It is composed of multiple rows and columns of photosites. It can thus produce two dimensional images. The basic concept underlying solid state image sensors is that a

separate electrical signal is produced for each pixel or area in the sensor.

Vision Optics

The image of a part is formed on the sensor with the aid of a lens or lens system. Four important parameters associated with optical lens of the vision systems are magnification, focal length, depth of field and lens mounting. Our lens system was equipped with a zoom control which changed the magnification (focal length) of the lens between two finite limits. The zoom lens offered most flexibility by adjusting the magnification based on the size of the needle under inspection (8 wire - 20 wire). The space above and below the object plane where the lens maintains the focus of the image within acceptable limits is the depth of field. The depth of field was an important optical consideration in our application. A lens system with a greatest depth of field was desired since we were looking at the needles with very high magnifications (250X - 350X). The depth of field would correct for slight variations in positions during fixturing (due to mechanical tolerances of the grippers and arms) by keeping the needles in focus. The lens had a C-mount which attached the CCD camera.

Vision Processing Hardware

The image processing architecture includes a high speed pipelined digital processor. Pixels are processed at a rate of 10 MHz (100 nanoseconds/pixel). The processor can perform two look-up table transforms, a multiplication, a logical or arithmetic operation, and data normalization to all 262,144 pixels (512 X 512) in an entire frame in real-time (1/30th second). Several processing elements are connected via a software controllable data flow architecture (Pipelined Pixel Processor), giving maximum power and applications range to a single image processing system. The image processor contains A/D and D/A converters for analog video signal I/O. Up to four video cameras, VCR's, or other RS-170 standard video devices can be connected to the processor. (2 camera inputs were used, one CCD for the tips and the other CCD for channels). Ultra-stable phase lock loop circuitry insures reliable locking the video input signal, which is low-pass filtered and digitized to 8-bit resolution (256 grey levels) at a 10 MHz rate. Up to three

full 512 X 512 X 8-bit images can be simultaneously stored in the internal frame buffers of the processor. This triple image memory supports the data recirculation techniques which are so vital to high speed image processing. Two memories can be combined to store a 16-bit deep processed image. The vision system is designed to be connected to an AT-compatible host computer. An AT with a 33 MHz clock rate was used. The software that drives the entire vision system is resident in this host computer.

Sec 6.2 OVERALL SYSTEM HARDWARE

In Sec 2.3 of Chap 2, we listed the sequence of tasks involved in automating the cutting and inspection of needles. Here we will biriefly list and describe the function of the hardware components used in the automated system. Figure 6.2 shows a block diagram of the main components and how they are connected to each other.

A programmable logic controller (PLC) is an assembly of solid-state digital logic elements designed to make logical decisions and provide outputs. Push buttons at the control panel are used to power up, start, home or stop the system. The opening and closing of grippers, the lowering of the punch, the movement of the rotary actuators are all controlled by the PLC through solenoids, whose valves are connected to air cylinders. The PLC sends an output to turn on a solenoid which moves the piston that causes an action to take place. The PLC also senses inputs from various locations such as grippers, rotary actuators, die-set punch etc. These sensors inform the PLC that the current task has been completed (the grippers has been shut, the punch has been lowered, etc) and it can proceed to the next sequential task. The PLC also sends and receives signals from motor controllers. Motor controller 1 is used to push the carrier containing the needles that are ready to be cut. Motor controller 2 positions the X-Y table on which the inspected needles are placed. Proximity limit switches are used to govern the starting, stopping and reversal of motors. They sense the presence or absence of metal objects without physical contact.

Vision Interface

The vision system host computer is interfaced with the PLC via an 8255 parallel interface adapter. The host computer receives signals from PLC that tell it to start the image acquistion on camera 1 or camera 2. The vision computer sends signals back to the PLC with the inpection results for camera 1(channels) or camera 2 (tips).

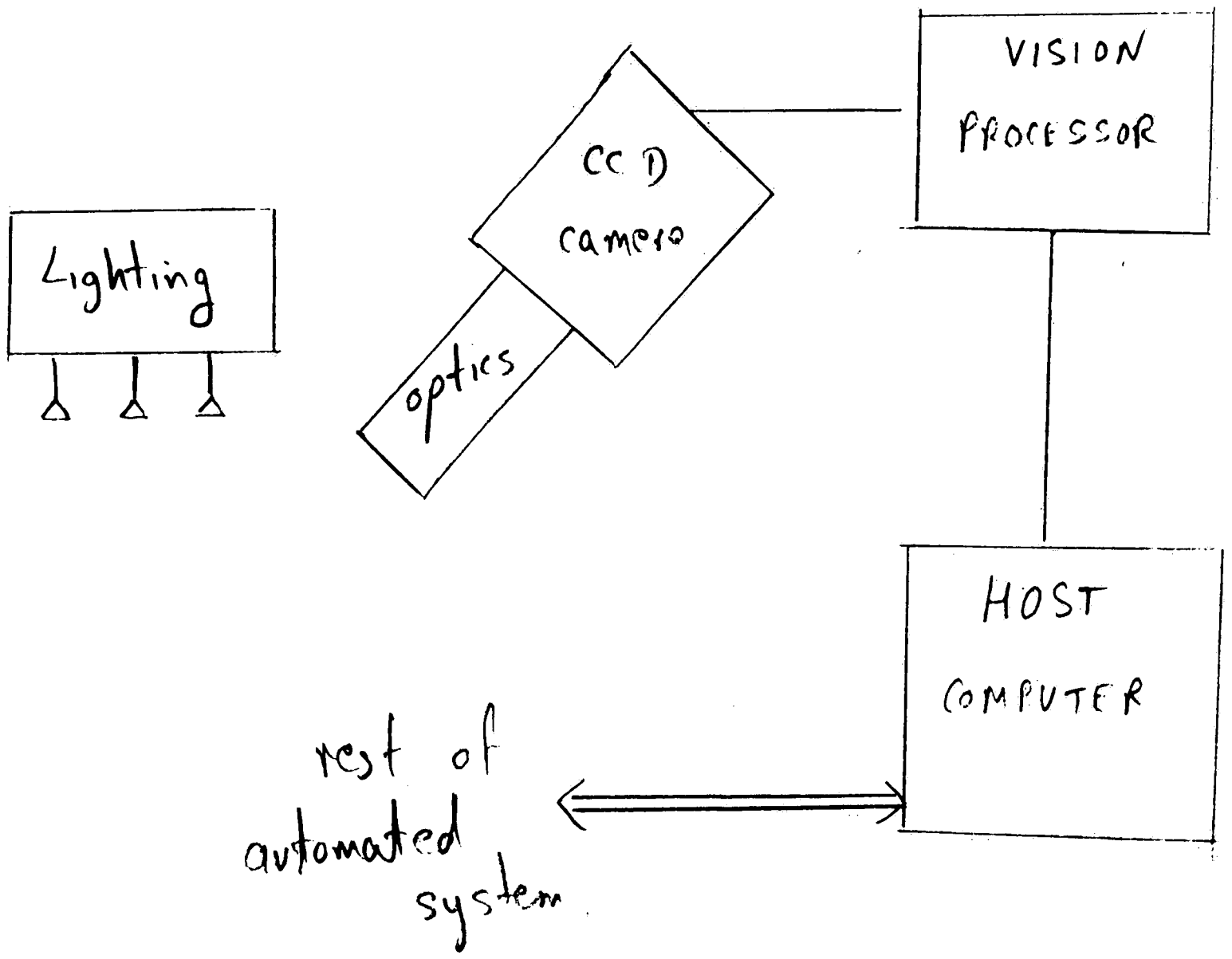


Figure 6.1 Hardware components of a machine-vision system

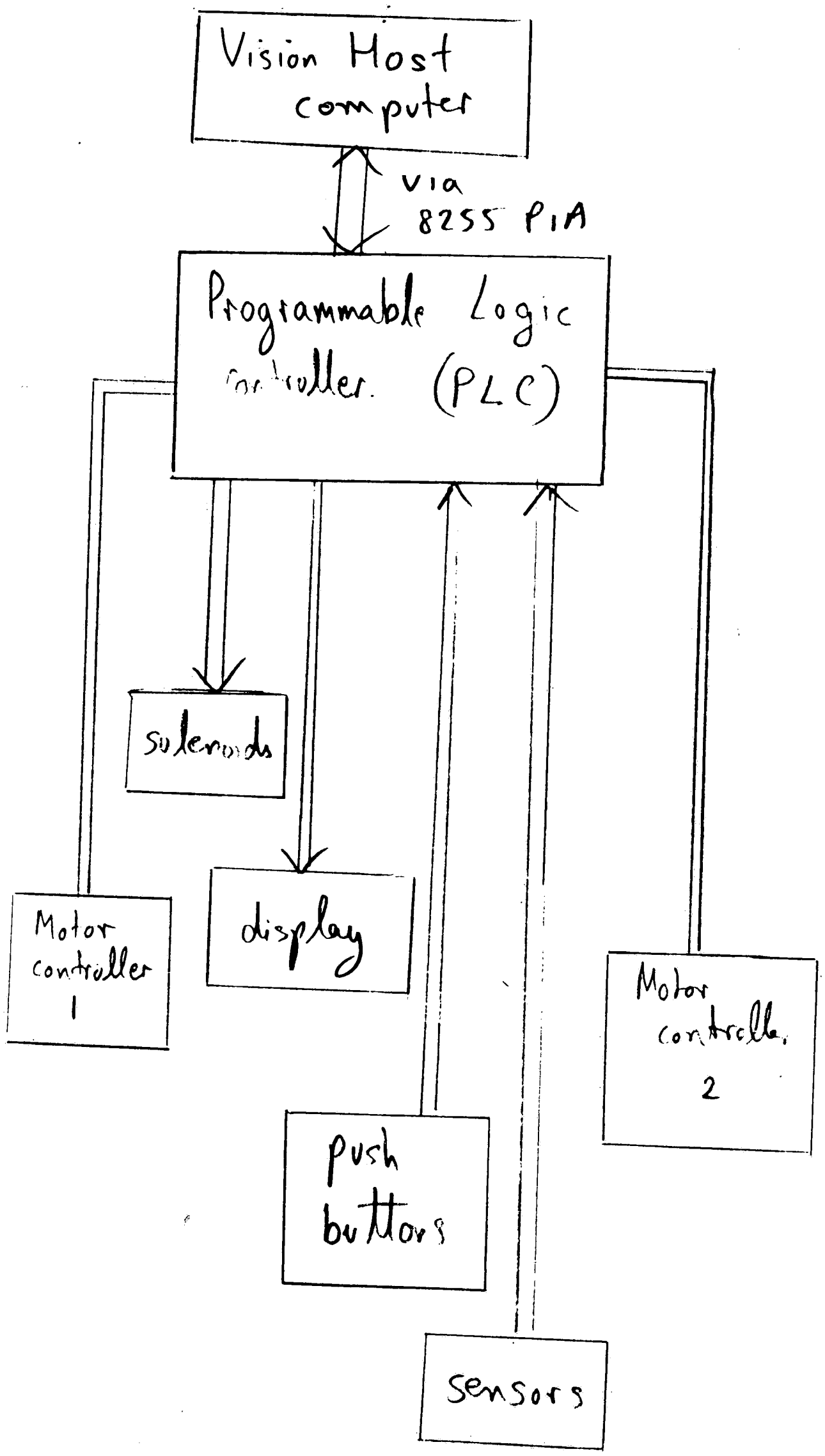


Figure 6.2 Block diagram of the main hardware components of the overall automated system

CHAPTER 7

APPLICATION AND ANALYSIS OF THE TECHNIQUES

In this chapter we will demonstrate the specific techniques applied to extract the edges of the tips/channels and analyze their shapes, using the methods developed in the previous chapters. We will start by going through the operations involved in extracting the edges.

7.1) EDGE DETECTION OF TIPS/CHANNELS

The imaging workstation can be used not only to digitze an image, but to extract only the edge information, leaving a result suitable for computer analysis of the shape. Here we will show how to capture a picture, enhance the contrast, find the edges, and eliminate all non-edge information, leaving a sketch outline of the object. The sequence of operations performed in obtaining the profile is the same for the tips and channels.

Plate 1.0 shows an original frame of a sample tip that has been acquired. A fragment of this image is also indicated in this plate by a small rectangle, that partly overlaps the tip surface. This fragment spans a small area of the image with rows ranging from 254 to 263 and columns ranging from 295 to 306, out of a total of 512 X 512 rows and columns. This fragment is chosen to illustrate the changes in the gray values of the pixels after each step of the process. Table 1.1 lists the gray values of all the 120 pixels (10 rows X 12 columns), occuring within this fragment of the original image of Plate 1.0. The gray values range from 0 to 255 (8-bit grey scale resolution). The gray values occuring in the top half of Table 1.1 are close to 255 and correspond to the white background area within the fragment. The gray values occuring in the bottom half are much lower that correspond to the darker region on the needle surface within the fragment. Each step of the edge-detection process operates on the gray values of the image. After each operation, the new image will be shown in the plates. A table containing the new gray values occuring within the fragment will also be shown after each operation. The location of the fragment will remain

fixed. The plates will help to visualize the effect of each operation on the appearance of the image. The corresponding tables will indicate the effect of the operation on the gray values of the image. As we follow each of the following steps, we will see how the sketch of the tip of Plate 1.0 finally emerges.

Step I Contrast Enhancement

As discussed in Chap 3, contrast enhancement is done by 'stretching' or 'equalizing' the histogram. This step will increase the magnitude of all the edges in the picture, making it easier to detect the edges later. The mechanism is to make a histogram that counts the number of pixels at each possible grey level, then to integrate that curve, and use the resulting cumulative distribution as a grey-scale look-up table. This results in a picture that has, as closely as possible, equal numbers of pixels at each possible grey level. Plate 1.1 shows the picture of the same original image in Plate 1.0, along with a graph of the cumulative distribution function (CDF) of all the image values. The input values of the function cover the range of grey scale values of the image (0 to 255). The output value of the function is the percentage of the image that has an intensity equal to or below the input value. Plate 1.2 shows the image after contrast enhancement, along with the graph of the CDF. The transformed image has a more uniform CDF compared to the CDF in Plate 1.1, resulting in increased contrast. Table 1.2 lists the gray values of the fragment in Plate 1.2., that have a uniform density.

Step II Edge Detection

Next, the edges of the object should be isolated from the rest of the picture. This is accomplished with a non-directional edge detection convolution, as discussed in Sec 4.1. It gives neutral gray output on solidly colored areas, and brightest possible output for isolated black spots. Its effect on edges is to turn each one into a black line immediately adjacent to a white line as seen in Plate 1.3.

The following 3X3 spatial high pass filter is applied to the image using the method of spatial filtering discussed in Sec 4.1.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The result of applying this mask to the gray values in Table 1.2 is shown in Table 1.3. It produces a 16-bit signed image. Table 1.3a lists the least significant byte of the 16 bit value of each pixel. The negative gray values adjacent to positive ones reflect the appearance of the black line adjacent to white ones. Table 1.3b contains the most significant byte of the 16 bit value of each pixel.

Step III Edge Selection

Since we only need one line for each edge, we choose to eliminate the black lines. All values below zero in the image are immediately clipped to 0, leaving greys and whites, but eliminating blacks. Plate 1.4 shows the result of this clipping. Table 1.4 shows the fragment values of Plate 1.4. It can be seen that there is only one line for the edge in Plate 1.4.

To increase the dynamic range, now is a good time to change the image to an unsigned image, mapping neutral gray (signed 0) to black (unsigned 0). This is accomplished by taking the absolute value of the frame of Plate 1.4. The result is shown in Plate 1.5. Table 1.5 shows the fragment values of Plate 1.5.

Step IV Edge Amplification

The edges of the image are now seen as light-colored pixels. Ideally, they should be totally white, with the background totally black. The first step towards this goal is to square all pixel values, to heighten the contrast. The result of this operation is shown in Plate 1.6. Table 1.6a lists the least significant byte of the 16 bit result. Table 1.6b lists the most significant byte of the 16 bit result.

Any values that went over 255 upon being squared are brought back into range by high clipping the 16 bit frame of Plate 1.6 to the value of 255. Plate 1.7 show the resultant image. This frame now

contains 255 where the original image had an edge. Table 1.7 shows its fragment values. Plate 1.7 also shows some low frequency noise at the top left of the tip. Apparently, the edge-detection steps have enhanced the noise which can be seen as a blob. The sources of these noise components could be dirty lenses, non-uniform illumination or shadows. They should be avoided as far as possible since they can interfere with the edge-extraction and give wrong results.

Step V Sequential Edge extraction

From the edge map of Plate 1.7, the coordinates of the edge pixels are extracted sequentially using the method of omnidirectional tracking discussed in Sec 4.3. Column 400 is scanned downwards to find a edge pixel with a gray value of 255, which will be the starting point of the tracking procedure. Once a starting point is located its 8 immediate neighbors are scanned for the next edge pixel.

7.2) SHAPE ANALYSIS OF TIPS AND CHANNELS

In this section, we will see the results of applying the techniques of Chap 5. to distinguish good tips/channels from bad ones. We will start by examining samples of tips, whose edges have already been extracted.

Analysis of Tips

Plate 3.0 shows the sequentially extracted edge image of a good tip sample. The first step is to locate the tip in this image. To do this, we first find all the corners using the corner-detection algorithm discussed in Sec 5.1. The detected corners are highlighted in Plate 3.1. The corners are number anticlockwise starting from the top rightmost corner. The tip is picked from these corners by its sharpness.

Once the tip is located, the next step is to measure the attributes used to distinguish good tips from bad ones as discussed in Sec 5.2. Next, we will list the attribute values calculated for three samples of tips (one good and two defective).

A set of 4 attributes will be associated to the tips. The first attribute is the Linearity of the tip edges (A) that indicates how well we can fit straight lines on either side of the tip. We will calculate T_{per} and B_{per} (percentages of the standard error of estimate relative to the average value of y) for each tip. The second attribute is the Tip Offset (B) which gives the deviation of the predicted location of the tip from the actual location. The third attribute is Measure of Bend Angle (C) that gives the cosine of the bend angle of the top and bottom edges. The fourth attribute is the Defective Pixel Count that gives the number of edge pixels that fall in the defective region of the tip. The values of these four attributes will follow for the samples of tips(one good and two defective).

1. Attribute values for the good tip shown in Plate 3.2 :

A> Linearity of tip edges: T_{per} - 0.150763 %

B_{per} - 0.141589 %

B> Tip Offset: 1.854760

C> Measure of Bend Angle : top-cosine -0.999291 bottom-cosine -0.999946

D> Defective Pixel Count : 0

2. Attribute values for defective tip shown in Plate 4.2 whose profile is shown in Plate 4.0. Plate 4.1 gives a visual illustration of the attributes.

A> T_{per} - 0.212326 % B_{per} - 0.182130 %

B> Tip Offset: 6.934889

C> Measure of Bend Angle: top-cosine -0.855324 bottom-cosine -0.890671

D> Defective Pixel Count: 10

3. Attribute values for the defective tip shown in Plate 5.0. Plate 5.1 gives a visual illustration of the attributes.

A> T_{per} 0.238847 % B_{per} 0.887841 %

B> Tip Offset 6.314860

C> Measure of Bend Angle top-cosine -0.963432 bottom-cosine -0.927521

D> Defective Pixel Count 7

By examining the values of the attributes of the three tips, it can be seen that the tip in Plate 4.2 possesses the highest magnitude of defect. The tip in Plate 5.0 is defective to a lesser degree compared with the tip in Plate 4.2. If we compare the attributes for the defective tips in Plates 4.2 and 5.0 with those of the good tip in Plate 3.2, there is a large difference in the values. Attribute D (Defective Pixel Count) seems to be the most reliable and powerful in distinguishing good tips from bad ones. The effectiveness of the other attributes vary depending on the magnitude and type of defect. By examining the attribute values for a larger number of good and bad samples, we can deduce a threshold value for each attribute which will be used to label the tips as good or bad. We arrived at the following threshold values for the four attributes, but they are based on a specific type of needles and can vary with the size and the magnitude of defect.

Threshold Values for the Attributes:

A> Tper = Bper = 0.17 %

B> Tip Offset = 2

C> Measure of Bend Angle: top-cosine = bottom-cosine = -0.98

D> Defective Pixel Count 2

Analysis of Channels

Here we will discuss two approaches used to distinguish good channels from bad ones. The first approach is based on the corner detection algorithm discussed in Sec 5.1. The second approach uses the concept of fractal dimensions discussed in Sec 5.3.

Approach 1.

The idea behind this approach is that if a defect occurs on the channels, there will be points of high curvature on the profile of the channel. The corner-detection algorithm will be able to pick these defect points on the curve. Then the local angle at these points can give us further information on the curvature at that point. Plate 6.0 shows an example of a good channel. Its profile is shown in Plate 6.1. The first step is to find all the corners using the corner-detection algorithm discussed in Sec 5.1. The smoothing factor of the corner-detection algorithm was adjusted for the shape of channels so that it detects very prominent corners. The detected corners are highlighted in Plate 6.1. Next, the local angle at each corner is found. Although the channel makes a circular turn, the local angles should be more or less straight lines. The low values of the angle reflect the defects occuring on the channels. We will list the local angle values for three samples of channels (one good and two defective).

Note: Corners are numbered anti clockwise and the two rigtmost corners of the channels are not considered.

1. *Local angle values of the detected corners for the good channel in Plate 6.0, whose profile is in Plate 6.1 :*

   Corner 0 : 174.39°

   Corner 1 : 164.51°

   Corner 2 : 167.79°

   Corner 3 : 160.47°

   Four corners were detected, whose values are close to 180°. This indicates that the channel is good.

2. *Local angle values for the defective channel in Plate 7.0, whose profile is in Plate 7.1 :*

Corner 0 : 171.65°

Corner 1 : 51.17°

Corner 2 : 41.19°

Three corners were detected. Corner 1 and 2 have very low values which indicates that there is a defect.

3. *Local angle values for the defective channel in Plate 8.0, whose profile is in Plate 8.1 :*

Corner 0: 117.30°

Corner 1: 78.53°

Corner 2: 164.91°

Three corners were detected. Corner 1 has a low value indicating a defect.

Approach 2

This approach uses the concept of fractal dimensions to give an estimate of the roughness of the channel profiles. The stride length is varied and the perimeter is calculated for each value of the stride length. The fractal dimension is found from the slope of the plot (stride length vs perimeter). Plates 9 and 10 demonstrate the use of fractal dimensions, discussed in Sec 5.3, to distinguish good channels from bad ones. Plate 9 shows a profile of a good channel, along with a fractal plot at the top left corner. Plate 10 shows a profile of a bad channel along with the fractal dimension plot. It can be seen that the slope of the bad channel plot in Plate 10 is much higher than that of the good channel plot in Plate 9.

CONCLUSION

This effectiveness of the techniques developed to detect edges depend on the quality of the original image. They have performed very well under the conditions used to inspect the tips and channels of the needles as demonstrated by the pictures. Some of the operations involved in the edge enhancement and

detection process can be fine tuned to the conditions of the application. The techniques developed to analyze the tips and channels have also been satisfactory in distinguishing good parts from bad ones. They can also be fine tuned according to the size and shape of the profile under analysis.

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	255	255	255	255	255	255	255	255	255	255	255	255
255:	255	255	255	255	255	255	255	255	255	255	255	255
256:	255	255	255	255	255	255	255	255	255	255	255	255
257:	244	252	255	255	255	255	255	255	255	255	255	255
258:	194	200	206	207	207	209	211	212	219	225	227	230
259:	162	171	178	182	179	180	181	183	187	192	198	199
260:	134	141	145	148	145	142	139	140	141	141	145	149
261:	117	123	127	129	128	124	122	124	123	119	122	125
262:	100	105	106	106	106	104	103	103	106	105	106	106
263:	92	95	98	96	94	95	95	95	99	104	104	102

TABLE 1.1

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	253	253	253	253	253	253	253	253	253	253	253	253
255:	253	253	253	253	253	253	253	253	253	253	253	253
256:	253	253	253	253	253	253	253	253	253	253	253	253
257:	27	27	253	253	253	253	253	253	253	253	253	253
258:	24	24	25	25	25	25	25	25	26	26	26	26
259:	22	22	23	23	23	23	23	23	23	24	24	24
260:	17	19	19	20	19	19	18	18	19	19	19	20
261:	13	15	16	16	16	15	15	15	15	14	15	15
262:	10	10	11	11	11	10	10	10	11	10	11	11
263:	8	8	9	9	8	8	8	8	9	10	10	10

TABLE 1.2

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	86	-113	-57	0	0	0	0	0	0	0	0	0
257:	-90	-32	28	85	85	85	85	85	85	85	85	85
258:	0	57	112	-87	-87	-87	-87	-87	-88	-88	-88	-88
259:	-3	-1	-2	-2	-2	-2	-2	-2	-1	-3	-2	-2
260:	0	-2	0	-2	0	0	1	1	-1	0	0	-1
261:	0	-2	-2	-2	-2	-1	-2	-1	-1	1	-1	1
262:	0	2	1	2	1	2	2	2	0	3	1	2
263:	0	1	-1	-1	0	0	0	1	0	0	2	4

TABLE 1.3a

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	-1	-1	-1	0	0	0	0	0	0	0	0	0
257:	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
258:	0	0	0	0	0	0	0	0	0	0	0	0
259:	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
260:	0	-1	0	-1	0	0	0	0	-1	0	0	-1
261:	0	-1	-1	-1	-1	-1	-1	-1	-1	0	-1	0
262:	0	0	0	0	0	0	0	0	0	0	0	0
263:	0	0	-1	-1	0	0	0	0	0	0	0	0

TABLE 1.3b

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	0	0	0	0	0	0	0	0	0	0	0	0
257:	-90	-32	0	0	0	0	0	0	0	0	0	0
258:	0	57	112	-87	-87	-87	-87	-87	-88	-88	-88	-88
259:	0	0	0	0	0	0	0	0	0	0	0	0
260:	0	0	0	0	0	0	1	1	0	0	0	0
261:	0	0	0	0	0	0	0	0	0	1	0	1
262:	0	2	1	2	1	2	2	2	0	3	1	2
263:	0	1	0	0	0	0	0	1	0	0	2	4

TABLE 1.4

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	0	0	0	0	0	0	0	0	0	0	0	0
257:	90	32	0	0	0	0	0	0	0	0	0	0
258:	0	57	112	87	87	87	87	87	88	88	88	88
259:	0	0	0	0	0	0	0	0	0	0	0	0
260:	0	0	0	0	0	0	1	1	0	0	0	0
261:	0	0	0	0	0	0	0	0	0	1	0	1
262:	0	2	1	2	1	2	2	2	0	3	1	2
263:	0	1	0	0	0	0	0	1	0	0	2	4

TABLE 1.5

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	0	0	0	0	0	0	0	0	0	0	0	0
257:	164	0	0	0	0	0	0	0	0	0	0	0
258:	0	177	0	145	145	145	145	145	64	64	64	64
259:	0	0	0	0	0	0	0	0	0	0	0	0
260:	0	0	0	0	0	0	1	1	0	0	0	0
261:	0	0	0	0	0	0	0	0	0	1	0	1
262:	0	4	1	4	1	4	4	4	0	9	1	4
263:	0	1	0	0	0	0	0	1	0	0	4	16

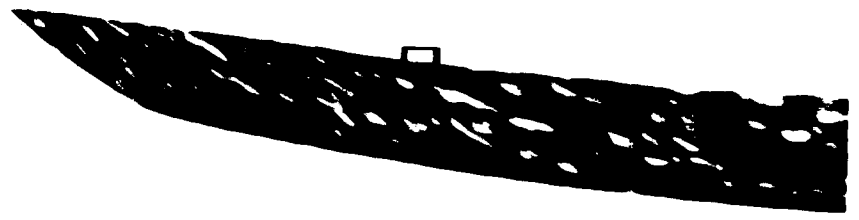
TABLE 1.6a

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	0	0	0	0	0	0	0	0	0	0	0	0
257:	31	4	0	0	0	0	0	0	0	0	0	0
258:	0	12	49	29	29	29	29	29	30	30	30	30
259:	0	0	0	0	0	0	0	0	0	0	0	0
260:	0	0	0	0	0	0	0	0	0	0	0	0
261:	0	0	0	0	0	0	0	0	0	0	0	0
262:	0	0	0	0	0	0	0	0	0	0	0	0
263:	0	0	0	0	0	0	0	0	0	0	0	0

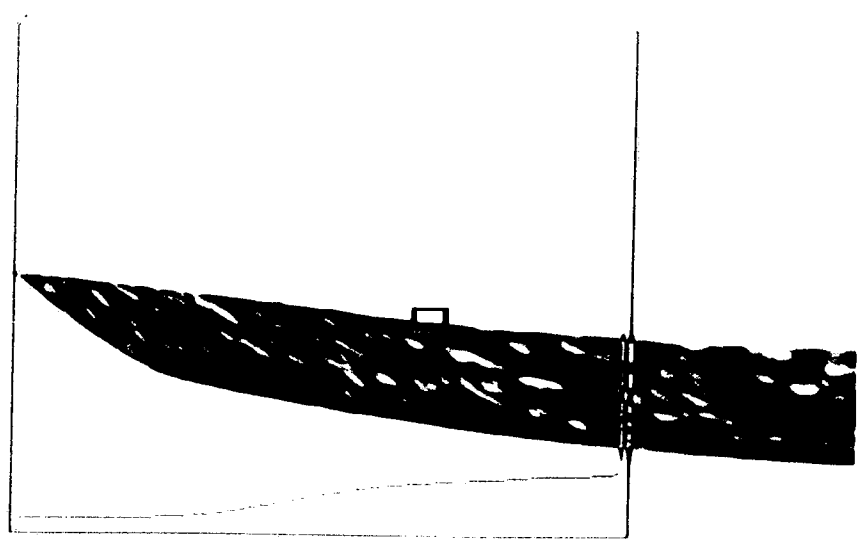
TABLE 1.6b

Row:\Col:	295:	296:	297:	298:	299:	300:	301:	302:	303:	304:	305:	306:
254:	0	0	0	0	0	0	0	0	0	0	0	0
255:	0	0	0	0	0	0	0	0	0	0	0	0
256:	0	0	0	0	0	0	0	0	0	0	0	0
257:	255	255	0	0	0	0	0	0	0	0	0	0
258:	0	255	255	255	255	255	255	255	255	255	255	255
259:	0	0	0	0	0	0	0	0	0	0	0	0
260:	0	0	0	0	0	0	1	1	0	0	0	0
261:	0	0	0	0	0	0	0	0	0	1	0	1
262:	0	4	1	4	1	4	4	4	0	9	1	4
263:	0	1	0	0	0	0	0	1	0	0	4	16

TABLE 1.7



Courtesy of Rhon Surgical & CIM Lab, Lehigh University



Courtesy of Rhon Surgical & CIM Lab, Lehigh University

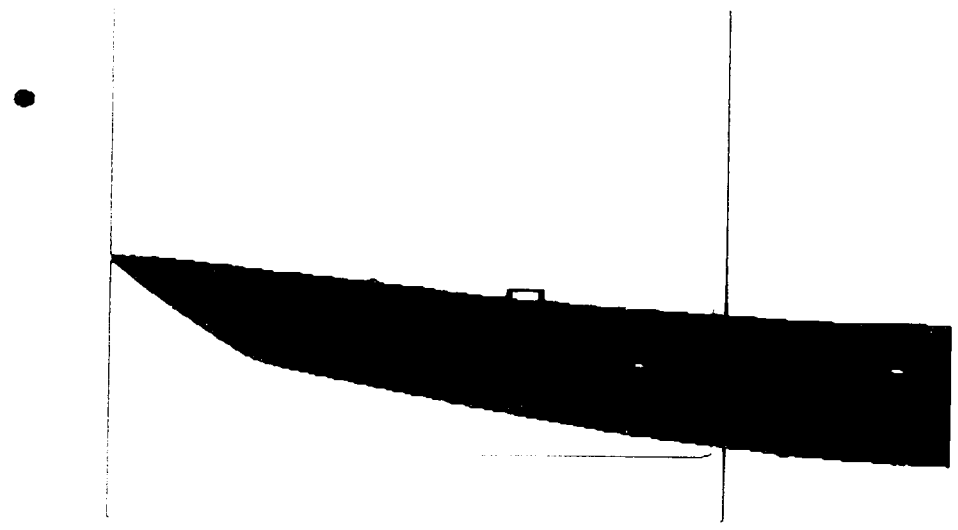
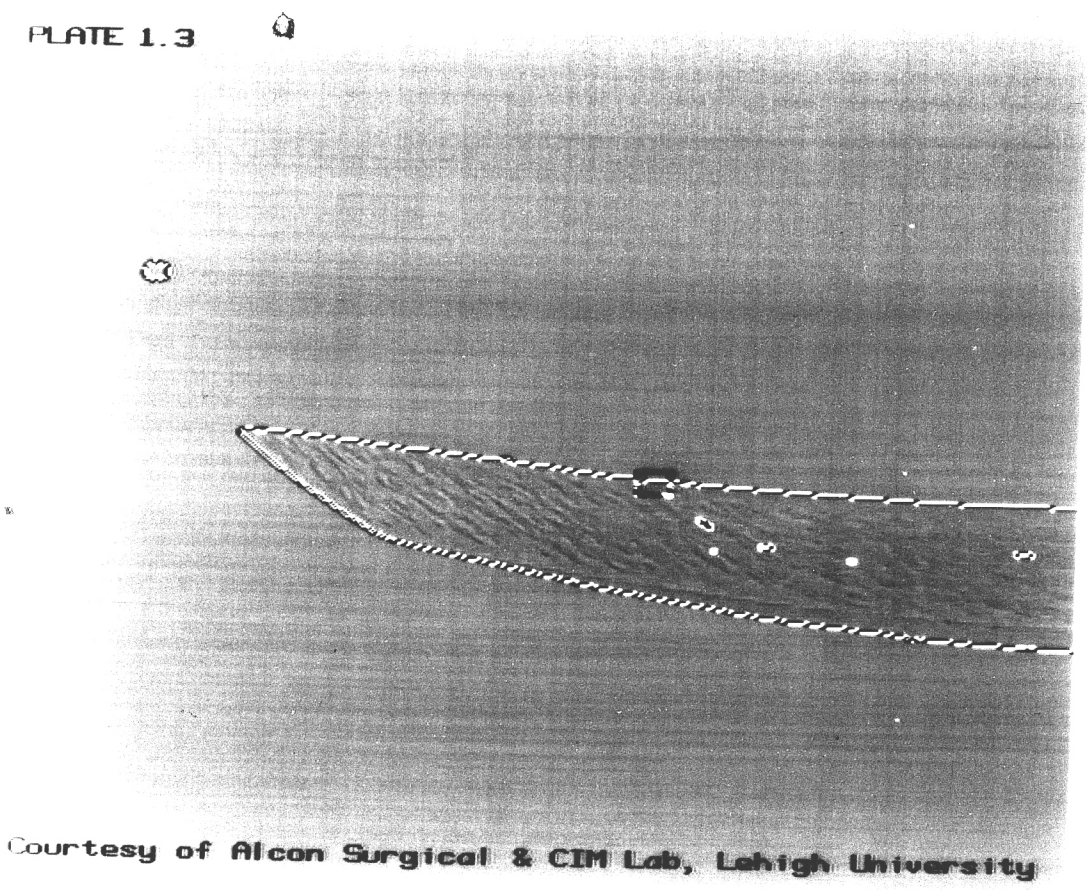


PLATE 1.3



Courtesy of Alcon Surgical & CIM Lab, Lehigh University

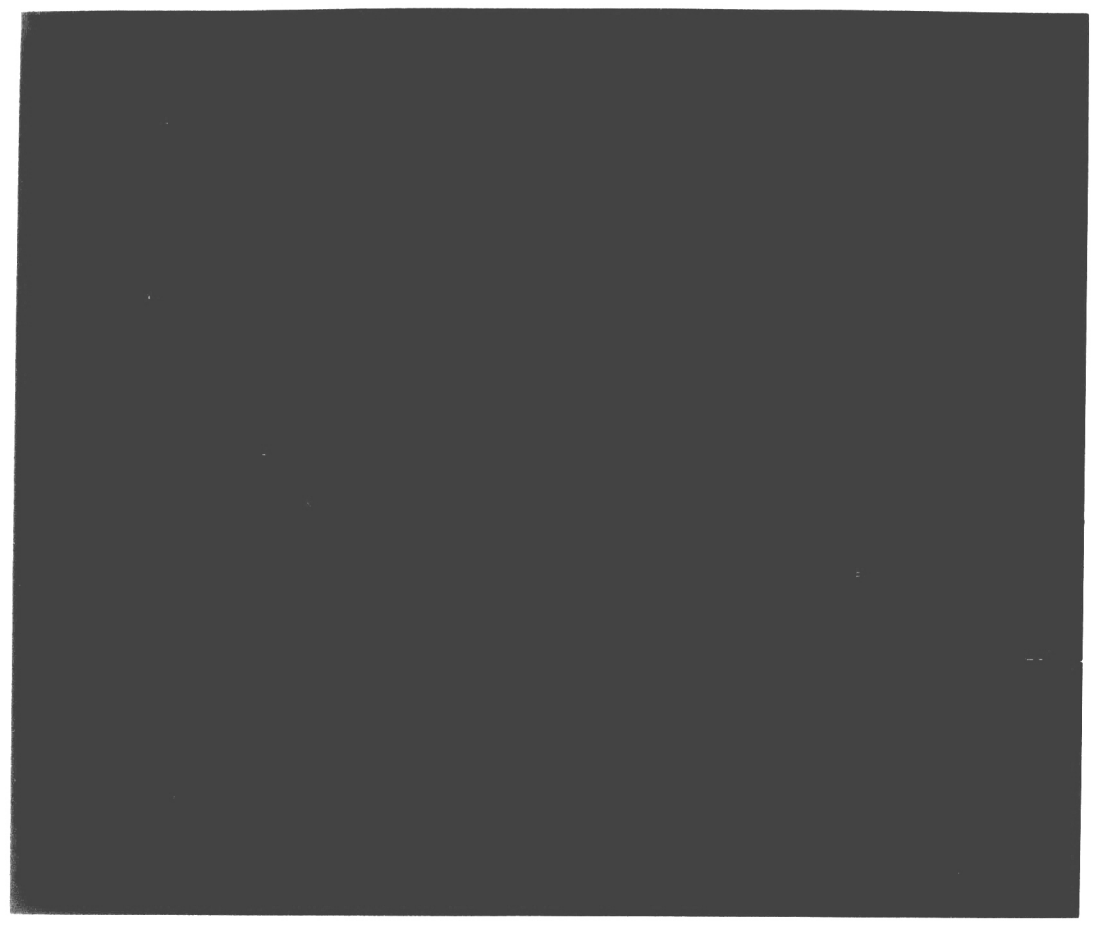
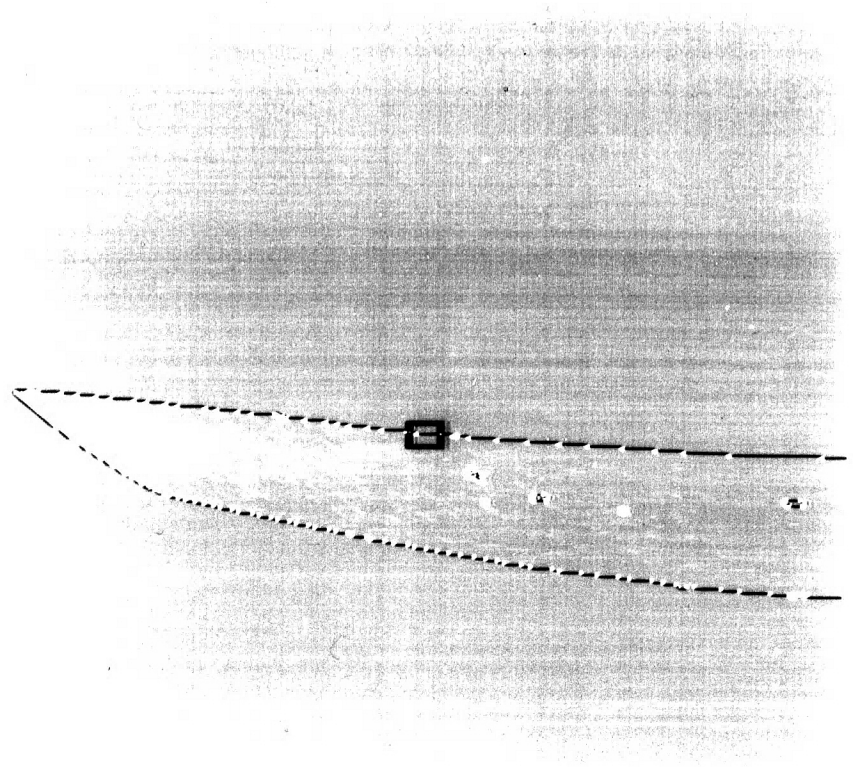


PLATE 1.4

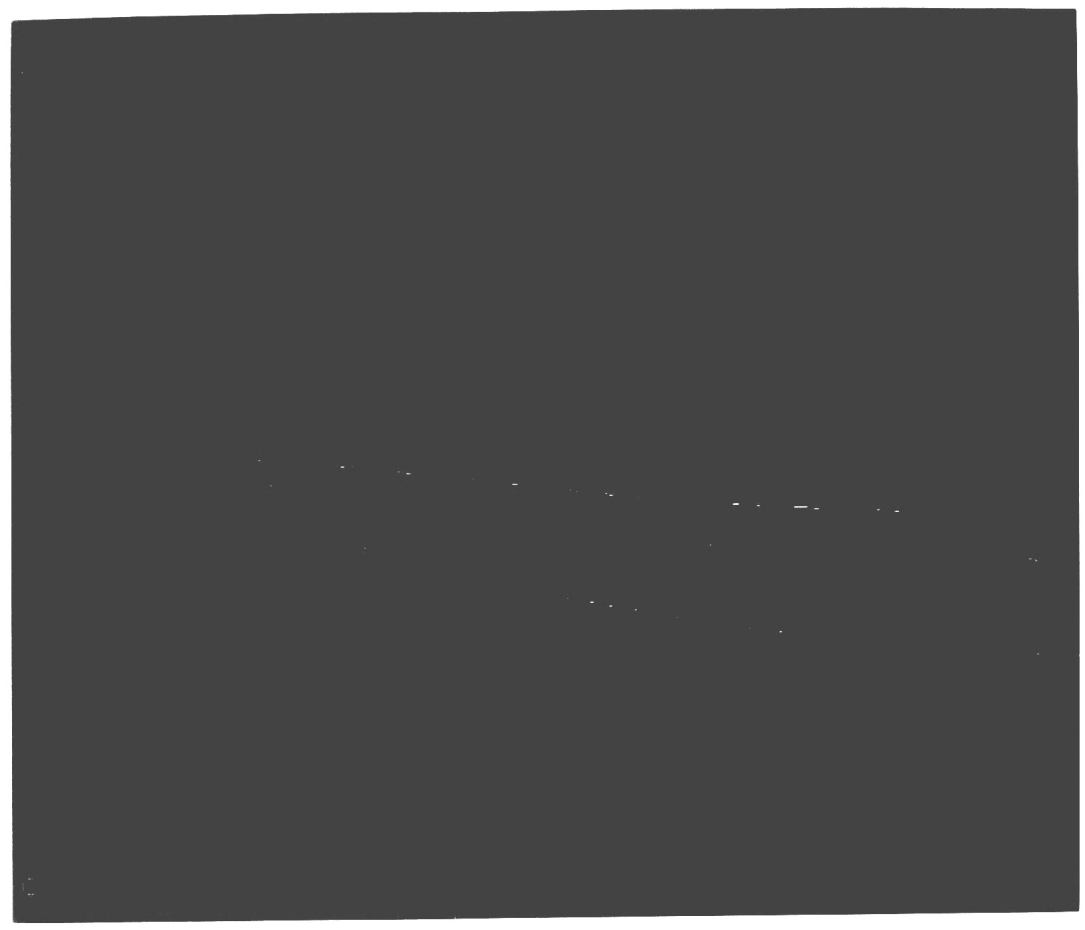


Courtesy of Alcon Surgical & CIM Lab, Lehigh University

[redacted]

[redacted]

[redacted]



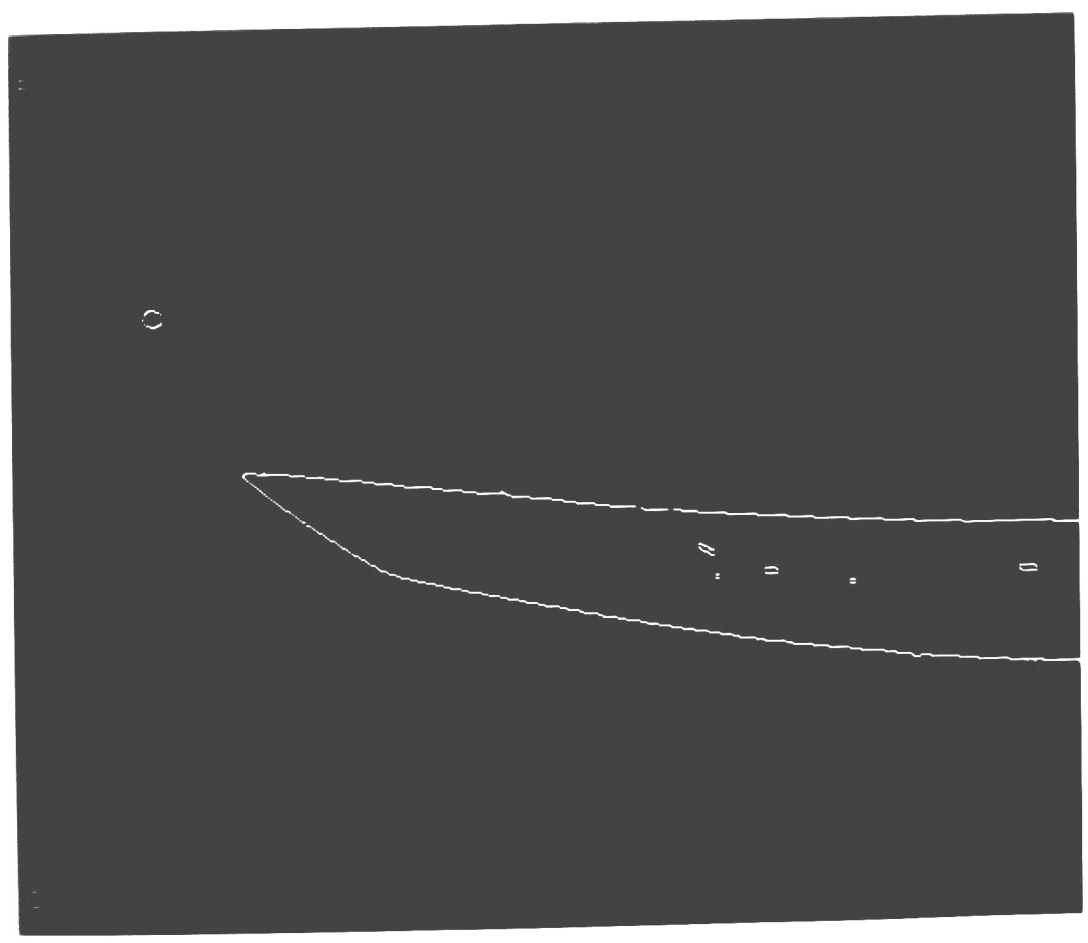
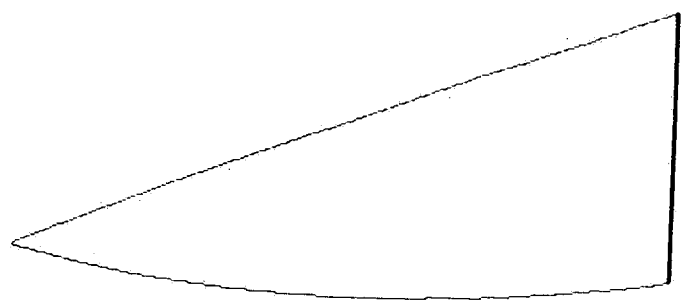


PLATE 3 0

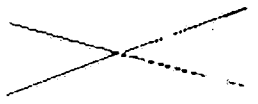


Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 3 1

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 3 2



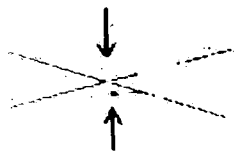
GOOD TIP

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 4 0

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 4.1



DEFECTIVE TIP

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

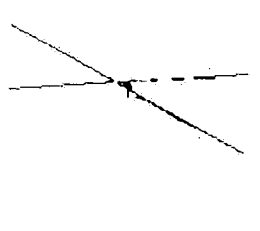
PLATE 4.2

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 5 0

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

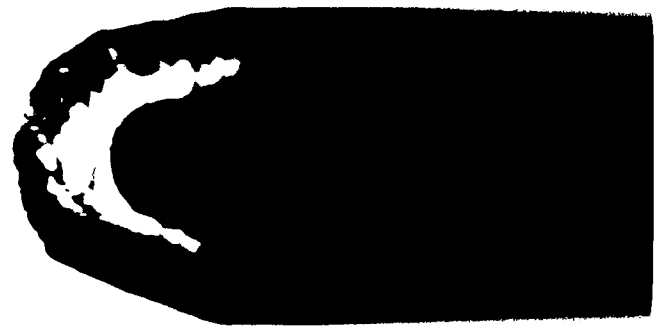
PLATE 5 1



DEFECTIVE TIP

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 6.0



Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 6.1

GOOD CHANNEL

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

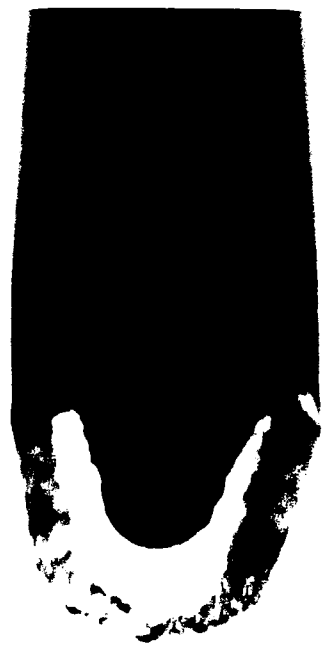
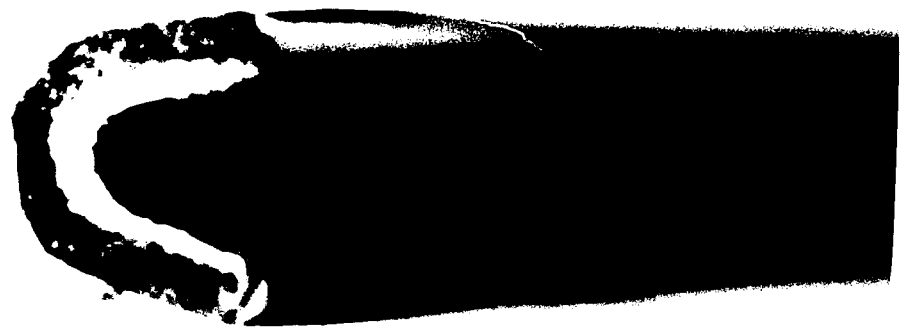


PLATE 7 0



Courtesy of Alcon Surgical & CIM Lab, Lehigh University

PLATE 7 1

DEFECTIVE CHANNEL

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

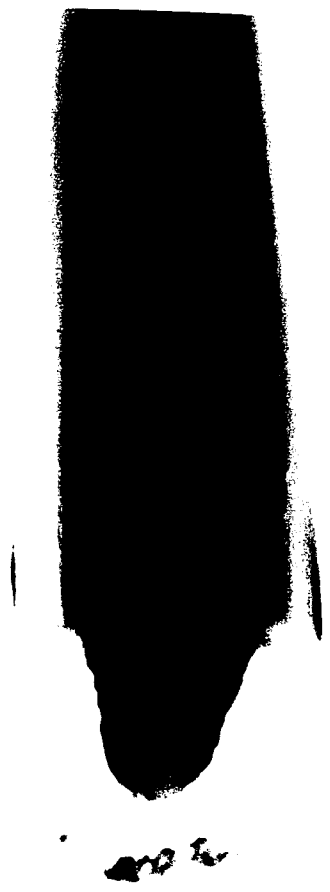


PLATE 8 0

Courtesy of Alcon Surgical & CIM Lab, Lehigh University

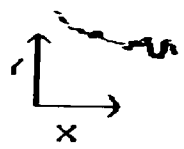
PLATE 8 1

DEFECTIVE CHANNEL

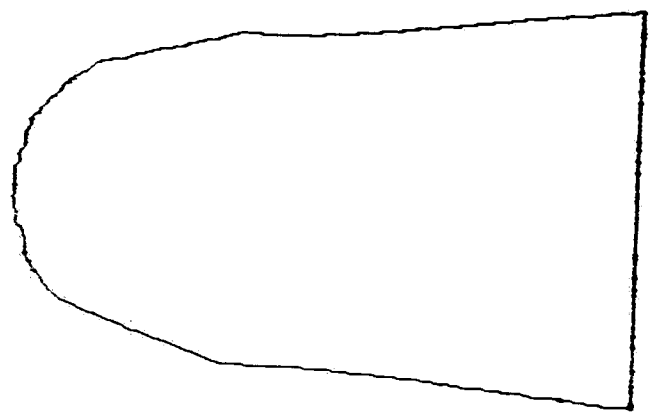
Courtesy of Alcon Surgical & CIM Lab, Lehigh University

Needle size 8 wire

PLATE 9.0



Fractal Dimension Plot



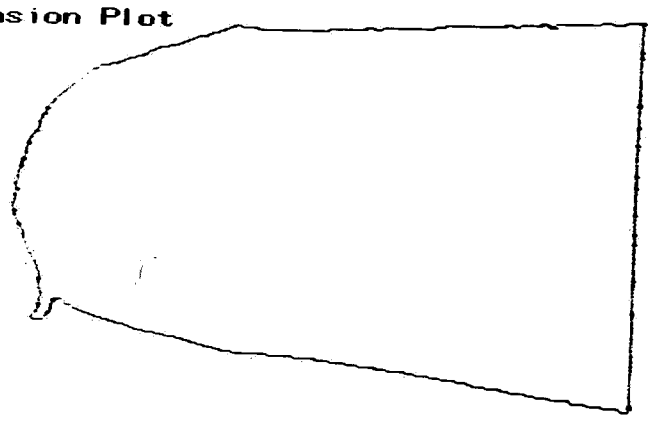
Wed Aug 21 17:08:17 1991 Needle number 46

PD - RESEARCH & DEVELOPMENT CIM Lab, Lehigh University

Needle size 8 wire

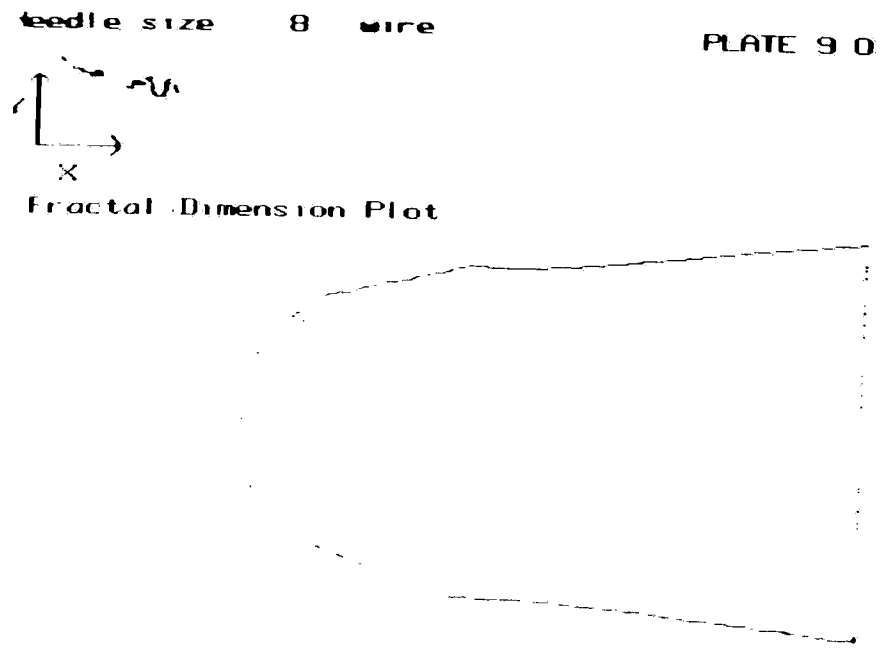
PLATE 10

Fractal Dimension Plot

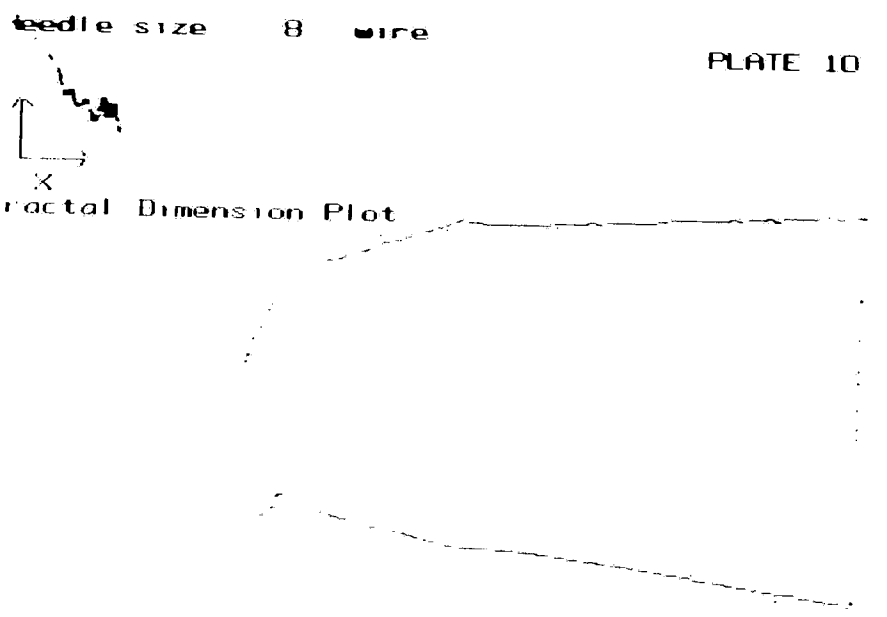


Wed Aug 21 17:07:10 1991 Needle number 45

PD - RESEARCH & DEVELOPMENT CIM Lab, Lehigh University



ed Aug 21 17 08 17 1991 Needle number 46
PD RESEARCH & DEVELOPMENT CIM Lab, Lehigh University



ed Aug 21 17 07 10 1991 Needle number 45
PD RESEARCH & DEVELOPMENT CIM Lab, Lehigh University

REFERENCES

1. Galbiati, Louis J., Machine Vision and Digital Image Processing Fundamentals, New Jersey, Prentice Hall, Inc., 1990.

2. Hall, Ernest L., Computer Image Processing and recognition, New York, Academic Press, Inc, 1979, pp. 166-173.

3. Beaumont, G.P., Probability and random variables, England, Ellis Horwood Limited, 1986.

4. Rosenfeld, Azriel and Kak, Avinash C., Digital Picture Processing, New York, Academic Press, Inc, 1982, pp. 241-245

5. Schalkoff, Robert J., Digital Image Processing and Computer vision, New York, John Wiley & Sons, Inc., 1989, pp 262-269

6. Davis, Larry S., "Understanding Shape: Angles and sides" IEEE Transactions on Computers, Vol C-26, No. 3, pp. 236-242

7. Rosenfeld, Azriel and Johnston Emily., "Angle Detection on Digital Curves" IEEE Transactions on Computers, Sep 1973, pp. 875-878.

8. Kaye, Brian H, A Random Walk through Fractal Dimensions, New York, VCH Publishers, 1989.

9. Russ, John C, Computer-Assisted Microscopy, New York, Plenum Press, 1990, pp. 161-168.

10. Zeuch Nello, Machine Vision : Capabilities for industry, Machine Vision Association of SME, 1986.

VITA

The author, Man Mohan Sapra, was born on August 17, 1966 to Mr. Vishwanath Sapra and Mrs. Krishna Guliani. His undergraduate study was done at Florida Institute of Technology, Melbourne, Florida where he received a B.S. in Computer Engineering in 1987. His graduate study was completed at Lehigh University where he was involved in implementing an automated inspection system for a firm that manufactures opthalmic needles. He will receive a M.S. from Lehigh University in Electrical Engineering in October 1991.