1984

Determining object orientation from a single image using multiple information sources

Nigel J. Foster
Carnegie Mellon University

Arthur C. Sanderson

Determining Object Orientation from a Single Image Using Multiple Information Sources

Nigel J. Foster
Arthur C. Sanderson

CMU-RI-TR-84-15

Department of Electrical and Computer Engineering
and The Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

June 1984

This research was supported in part by Westinghouse Electric Corporation and the Carnegie-Mellon University Robotics Institute.

CONTENTS

Abstract

Three techniques are presented which use assumptions about the real world to determine the orientation of objects from a single visual image. The orientation information from each of these techniques is combined to provide a more accurate estimate of object orientation. This algorithm is applied to the task of estimating the orientation of a single transistor against a uniform, but contrasting, background.

Three techniques are proposed for estimating object orientation. *Histogram Template Matching* employs a nearest-neighbor classifier using the normalized correlation function as a distance measure between the histogram of the input image and a set of training histograms. *Binary Connectivity Analysis* analyzes the connectivity of an object's silhouette and uses the resulting image features to determine orientation. *Ellipse Fitting* uses the parameters of an ellipse in the image to specify the orientation of the corresponding circular object surface. Location of the image ellipse is accomplished by exploiting knowledge about object boundaries and image intensity gradients.

The orientation information from each of these three methods is combined using a "plausibility" function. This probability-based, sub-optimal, decision rule employs weighted sums of joint conditional probabilities to enhance robustness.

The combined techniques for estimating orientation were tested on 138 images of transistors. The limitations of this research, and suggestions for further study, are also discussed.

Chapter 1

Introduction

1.1 Objectives

Given a single two-dimensional image of a three-dimensional object, humans can usually infer the three-dimensional shape and orientation of the object in the scene. This feat is accomplished in spite of the fact that there is no *unique* correspondence between an image and its corresponding scene. The ability of humans to infer three-dimensional shape seems to indicate that humans employ assumptions about objects and image formation to aid the vision process. The assumptions made are usually learned from basic physical reasoning and a lifetime of visual experience.

The objective of this study is to analyze a number of techniques which use assumptions about the world in order to estimate the orientation of objects. The orientation information from each of the these techniques is then combined to provide a single, more accurate estimate of object orientation. This algorithm is applied to the task of estimating transistor orientations.

The focus of this work addresses only a limited subset of the problems which are relevant to the field of computer vision, however, the work is not without practical significance. Many industrial applications of computer vision require fast, accurate, classification and orientation of known objects.

1.2 Background

It is the purpose of this research to examine techniques which may be applicable to the determination of 3-dimensional object orientation from a single view. This problem has been addressed in the field of computer vision, and a brief review of the relevant research is in order.

Early attempts at computer vision employed a pattern recognition approach based on simple image features [Rosenfeld 1969] [Levine 1969] [Duda 1973] [Fukunaga 1972]. This approach is designed to *classify* input images into predetermined categories. First, a set of feature values is extracted from an input image and then classification is performed on the basis of statistical decision rules in the feature space. For the purpose of determining orientation, the range of object orientations may be divided into a number of individual orientation classes. Although there is a large body of well-developed theory concerning the creation of optimal decision rules, pattern classification methods are limited in their applicability. In complicated problem domains, it is not always possible to determine a set of useful image features which can be extracted reliably. The application of classification techniques also requires the use of task dependent *a priori* knowledge in order to determine the appropriate decision rule.

The problem of determining object orientation has also been addressed through the use of object modeling [Roberts 1965] [Agin 1973] [Brooks 1981]. If a complete three-dimensional model of the object is available, it is possible to hypothesize a number of object orientations based on low-level image features and then match the model to each of the hypothesized orientations. The orientation which provides the best match is selected as the orientation of the object.

A more general set of methods for determining object orientation can be found in the so called "shape-from" methods. These techniques employ only very general assumptions about the world in an effort to obtain local surface orientations. Shape from shading [Horn 1975] uses the variation of image intensity to infer local surface orientation by exploiting knowledge about the physical process of image formation and by assuming that real world surfaces are discontinuous only at object boundaries. Similar assumptions can be used to infer shape from texture[Kender 1980] or from occluding contours [Marr 1977].

The approach taken in this paper borrows from many of the techniques mentioned above. It has been assumed that some *a priori* knowledge about the object is available, but that a full model of the object is not necessary. The approach used has been specifically designed to be able to perform on real-life images, which often defy simple theoretical frameworks. There has also been an effort to combine the different types of information which may arise from various orientation determining

techniques.

The integration of high-level knowledge is one of the primary goals of the field of artificial intelligence. Currently, knowledge integration is a poorly understood process, but much relevant research is underway. There are a number of "expert systems" which have attempted to combine information through the use of production rules [Davis 1977] or decision trees [Ballard 1976], but each of these systems employs task specific methods to achieve their goals. There is no general theory which can be used to select an optimal method for combining real world information in the face of error and uncertainty.

1.3 Application to Transistor Orientation

The techniques proposed in this study are applied to the task of determining the orientation of discrete transistors from a single grey-level image. The transistor package is a shiny cylindrical can with three leads, and the resulting images are related to orientation in complex ways. This application was selected as a test vehicle for the orientation operators because it represents a real life example with practical significance for robotic assembly systems. After determining the location and orientation of a transistor using vision, a robot arm may be used to acquire the device and insert it into a specified location of a printed circuit board.

The leads of the transistor are small when compared to the transistor body and, because the body itself is very nearly cylindrical, it becomes possible to approximate the entire transistor by a cylinder for the purpose of specifying orientation. The presence of the leads is used only to determine which end of the can corresponds to the "top".

Assuming that a transistor can be represented as a cylinder, it will be useful to establish a coordinate system which can be used throughout the rest of the report. Referring to figure 1.1, the axis of the imaging camera is aligned along the z axis of the world coordinate system. Assume that a separate coordinate system (x_c, y_c, z_c) has been attached to the cylinder such that z_c corresponds to the central axis of the cylinder and the "top" face of the cylinder occurs at a larger value of z_c than the bottom face. The orientation of these cylinder axes with respect to world coordinates may be measured uniquely by the three angles $\angle$tilt, $\angle$rotation and $\angle b$, where $\angle$rotation is defined as the angle between the y axis and L_p — the projection of z_c onto the xy plane.

Since the transistor is symmetric about its central axis (z_c), $\angle$tilt and $\angle$rotation are sufficient to fully specify the three-dimensional orientation of the transistor. This implies that, without loss of generality, it is possible to rotate the transistor about z_c until x_c lies in the xy plane; at this point, $\angle b$ is equivalent (but not equal) to

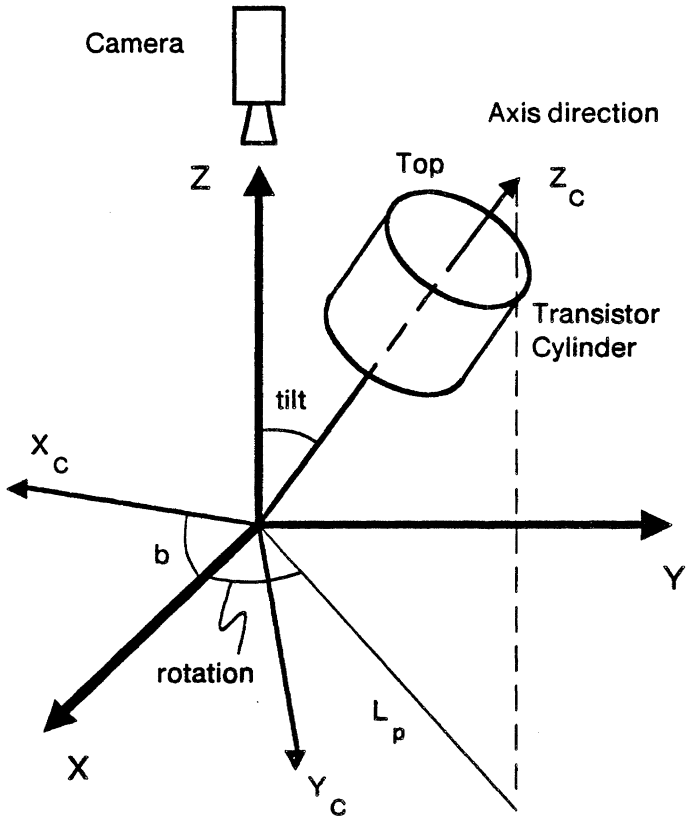


Figure 1.1: Viewing geometry for a cylinder. x, y, z are the world frame coordinates and x_c, y_c, z_c are the coordinate axes associated with the cylinder.

∠rotation and represents redundant information. In this study, ∠rotation is allowed vary from 0° to 359°, and is referred to simply as "rotation". Similarly, ∠tilt is referred to as "tilt", and can take on values between 0° and 90°.

The **Popeye** grey-level vision system [Bracho 1983] was used to implement the required algorithms and perform the necessary evaluation experiments. All of the algorithms were developed under the assumption of orthographic projection. Sample transistor images, and their corresponding orientations, are presented in figures 1.2 through 1.4 to familiarize the reader with the concept of transistor orientation. The picture at the top of figure 1.2 represents a transistor with a tilt angle of 60° and a rotation angle of 150°. This picture is used consistently throughout the report for the sake of illustration. Notice that rotation is measured in a *clockwise* direction because the **Popeye** system frame buffer represents a left-handed coordinate system.

1.4 Overview of this Report

The remainder of this report is divided into five chapters. Each of the first three chapters introduces and analyzes a specific image operation which can be used to deduce object orientation from a single grey-level image. The subsequent chapter examines methods for combining the orientation information from each of these operations. Finally, there is a chapter discussing conclusions and suggestions for further research.

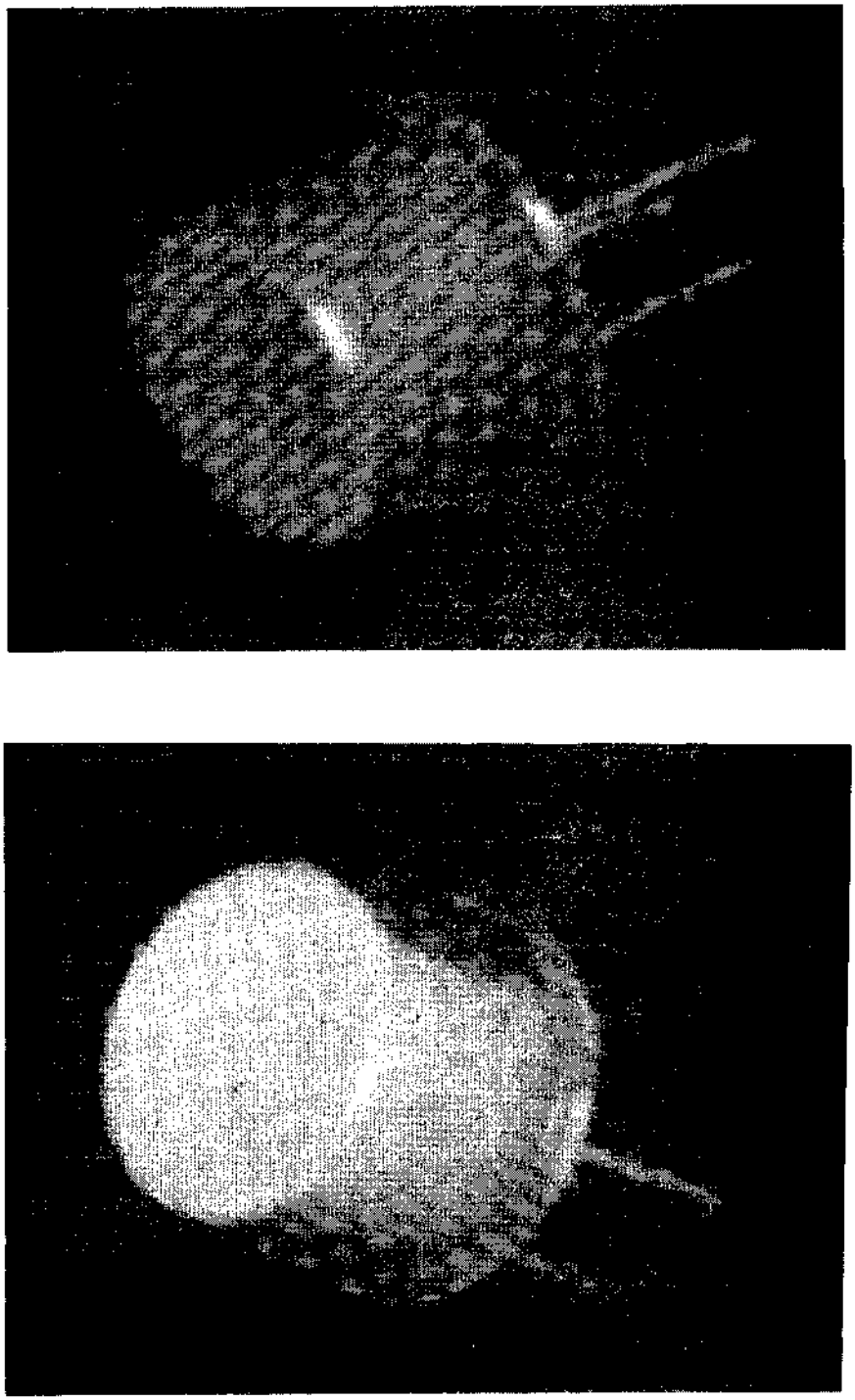


Figure 1.2: Top: Sample transistor image with tilt of 60° and rotation of 150°. Bottom: tilt is 45° and rotation is 195°.

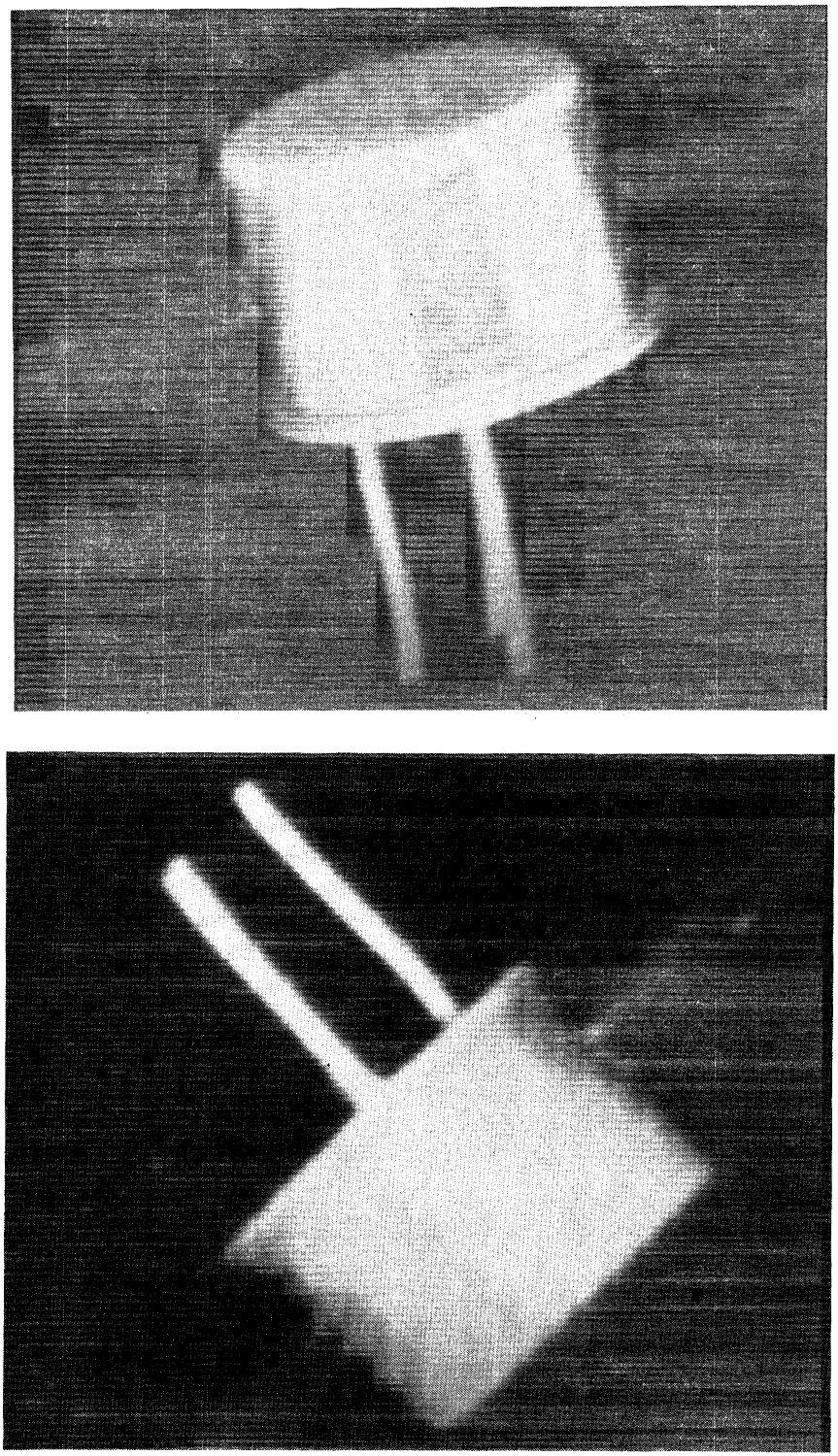


Figure 1.3: Top: Sample transistor image with tilt of 75° and rotation of 255°. Bottom: tilt is 90° and rotation is 45°.

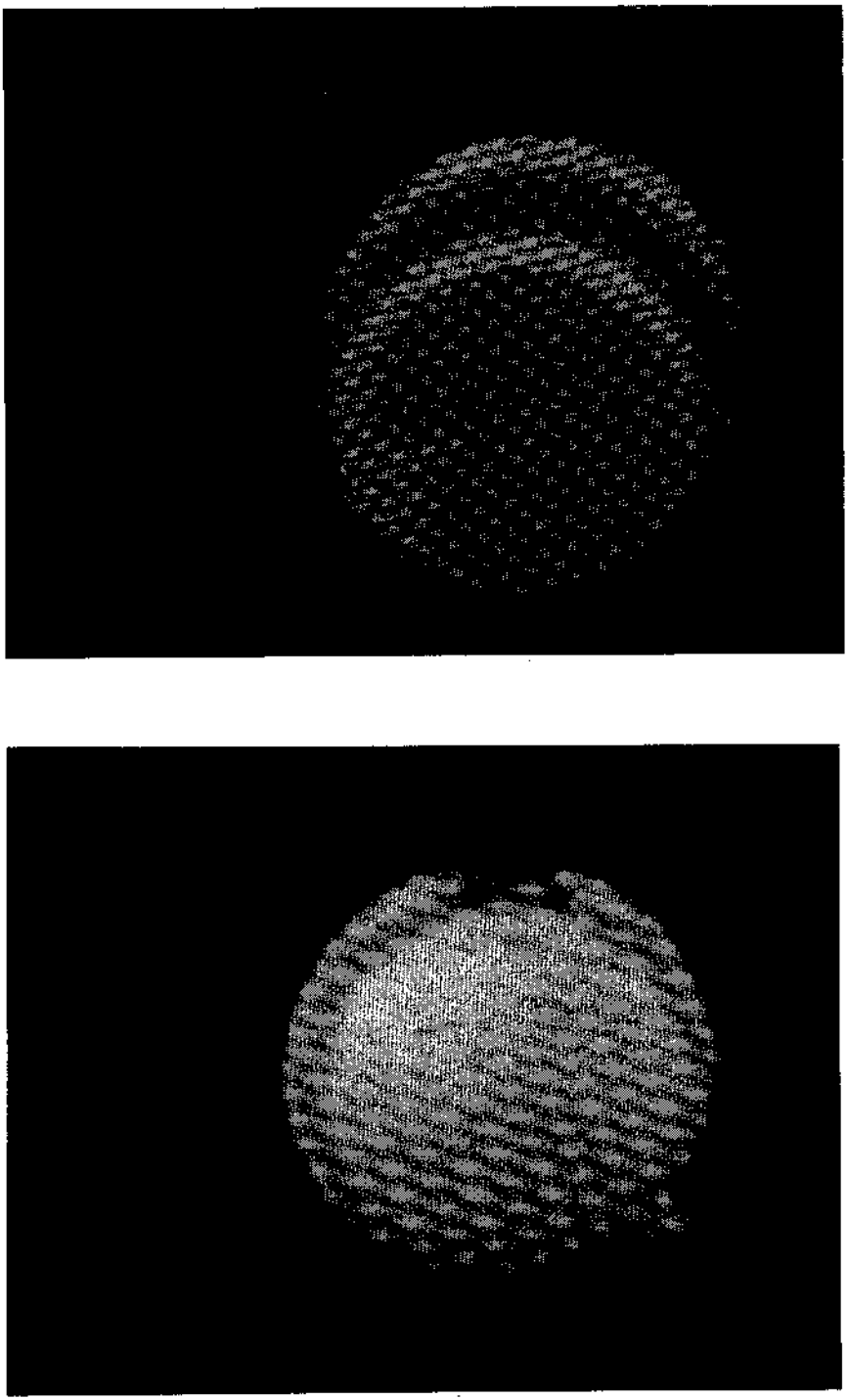


Figure *1A:* Top: Sample transistor image with tilt of 15° and rotation of 90°. Bottom: tilt is 0° and rotation is 0°.

Chapter 2

Histogram Template Matching

2.1 Introduction

The orientation of an object is related to various properties of a corresponding image. By identifying and measuring these relevant image properties it may be possible to deduce object orientation using a pattern recognition approach. In this chapter, the image intensity histogram is used to determine object orientation from an image of a single object on a uniform background. Although the intensity histogram ignores the spatial arrangement of pixels, its structure may consistently reflect changes in grey-level distributions associated with changing object views and, therefore, be a useful monitor of object orientation. The approach is limited to interpretation of highly structured scenes with a training set of views corresponding to different orientations. Under these restrictions the intensity histogram is found to provide useful cues for orientation with a minimal computational cost.

2.2 Histograms and Orientation

2.2.1 Information in Histograms

An image intensity histogram is a discrete estimate of the intensity probability distribution function for an image. This function can be viewed as a highly

compressed, statistical description of the image. The statistical description is not complete, but may contain clues to the orientation of individual objects which make up the scene. For example, consider a scene containing a white cube on a black background. The relative heights of the "white" and "black" histogram peaks will vary in a systematic fashion as the white cube is rotated in the scene and its projected area changes accordingly. The intensity distribution changes smoothly with orientation, but does not uniquely determine orientation. In more complex grey-level scenes, shadows, shading and highlights vary as well as the projected area; these components of the image also vary systematically with orientation.

While the intensity histogram retains only a portion of the original image information, this large data reduction, allows significant simplifications of the problem domain. For example, histograms are independent of rotation in the image plane. Neglecting minor quantization errors, a scene can be rotated arbitrarily about the centerline of the imaging optics without affecting the resulting image histogram. In fact, given a uniform background, orthographic projection and objects which do not overlap, it is possible to individually rotate the objects within a scene while leaving the histogram constant. Even with this kind of ambiguity and significant loss of scene data, a histogram will often still contain some useful information about the orientation of objects within a scene. The process of extracting this information, however, is not simple. Additional knowledge or training on known images is required.

2.2.2 Estimating Orientation Using the Intensity Histogram

One potentially useful source of additional knowledge is *a priori* information about the scene to be imaged. In many image processing applications, the overall structure of a scene and its imaging conditions are known in advance and can be used to aid the information extraction process. For example, the number or type of objects to be imaged and a general description of the background can provide a strong clue to the relationship between a histogram and the orientation of the objects in a scene. The hypothetical example of a white cube on a black background, at the beginning of this section, should make this point clear. While prediction of projected image area from model information is straightforward, relations involving shading and highlights require much more complex models of object geometry, surface properties, and optics. Such complex cases lead to a training based approach.

Techniques of classical pattern recognition (see [Duda 1973]) use a training based approach to represent *a priori* information about the imaging task. A training set of histograms may be obtained by imaging a sample object in all possible orientations. This training set of histograms may be used to derive an orientation classification rule using nearest neighbor or probability of error criteria. A histogram corresponding to an object with unknown orientation is compared to the "oriented histograms"

of the training set to determine the most probable orientation of the object. While such a histogram matching process may not uniquely determine orientation with high statistical confidence, it results in a ranked ordering of likely orientations which may be used in conjunction with other sources of information.

2.3 Histogram Matching

2.3.1 Mathematical Background

Histogram matching uses a similarity measure between pairs of histograms to determine the best match. A number of well-known similarity measures may be used here.

Similarity measures determine the distance between two histograms by combining histogram differences at every grey-level. For example, one possible distance measure between two histograms is obtained by subtracting the histogram heights at each grey-level and then summing the absolute value of these differences. Defining a histogram as a function $H(g)$ where g represents an element of the set of possible grey-levels and $H(g_0)$ represents the histogram height at grey-level g_0, this distance measure can be expressed as:

$$D = \sum_{\text{all } g} |H_1(g) - H_2(g)| \,. \tag{2.1}$$

A more common distance measure is based on the Euclidean distance:

$$E = \sqrt{\sum_{\text{all } g} [H_1(g) - H_2(g)]^2} \,. \tag{2.2}$$

This root mean-square error measure tends to reduce errors introduced by a small number of outlying points.

Equation 2.2 is more conveniently expressed by using the mean-square error [Pratt 1978]:

$$E^2 = \sum_{\text{all } g} [H_1(g) - H_2(g)]^2 \,. \tag{2.3}$$

Multiplying out the term $[H_1(g) - H_2(g)]^2$ yields:

$$E^2 = \left[E_{11}^2 - 2E_{12}^2 + E_{22}^2\right] \,, \tag{2.4}$$

where

$$\sum_{\text{all } g} [H_1(g)]^2,$$

$$H_1(g)H_2(g),$$

$$E_{22}^2 = \sum_{\text{all } g} [H_2(g)]^2 .$$

The terms *Ef*± and *E%2* represent the energy in histogram *Hi(g)* and #2(1?)? respectively. Equivalently, *E*^ and *E*\$_2$ could be viewed as the autocorrelation of *H*\{*g)* and #2(2)- The term *Ef*$_2$ represents the cross-correlation between the two histograms [Oppenheim 1975]. Under the assumption that every histogram has approximately the same energy, the relative square error distance between pairs of histograms is determined by the cross-correlation term alone.

In practice, this strategy does not produce consistent results because different histograms have different energy. One method for circumventing this problem is the normalized cross-correlation distance measure:

$$N_{12} = \frac{}{\sqrt{E_{11}^2 E_{22}^2}} . \quad (2.5)$$

In accord with the Cauchy-Schwarz inequality, the normalized cross-correlation has a maximum value of unity which occurs if and only if *IJi{g)* is a multiple of i?2(sO« This distance measure has many properties which are desirable for histogram matching. It accurately measures the distance between histogram shapes under all conditions and maps the result into the range between zero and one. It provides a practical bonus because a histogram scaled in height correlates perfectly with the original histogram. This feature allows the distance measure to compare shape, not just amplitude, and is useful for comparisons between imaging systems with different numbers of pixels per image. The normalized cross-correlation has been used in the matching experiments described here.

2.3.2 Practical Considerations

It has been assumed that it is possible to obtain a set of training histograms which represent the objects in the scene in every possible position. This is clearly a theoretical impossibility for most objects. A uniform sphere on a neutral background has only one possible orientation, but this is the exception rather than the rule. The training set must represent a quantized subset of all possible orientation histograms, where the coarseness of the quantization is to be determined by the requirements of

the application. The quantized values may be farther apart for convex, lambertian objects where small changes in object orientation tend to produce small changes in the resulting histogram. These objects may also make it possible to use linear interpolation between likely values of orientation. For example, given an unoriented histogram that has a very high correlation with both the first and second members of a training set, it is likely that the true orientation of the object is between the two training orientations.

Another practical problem with histogram matching arises from the illumination of the scene. A system designed to determine object orientations would be more useful if it were able to correctly interpret object orientations under various illuminations. Such a system would be required to differentiate between scene changes due to changes in object orientation and those due to changes in illumination. Changes in illumination can be further broken down into changes in light source position and changes in light source intensity. Changing the number or position of light sources often raises the issue of shadows, which may drastically alter the image of a scene in complex, highly non-linear ways. Such complexities are beyond the scope of this paper. Therefore, changes in illumination will be restricted to uniform changes in the intensity of all light sources.

It is necessary to modify the histogram matching process somewhat to incorporate changes in uniform illumination intensity. One possible modification would be the addition of more training histograms. A complete set of oriented histograms would have to be obtained for every possible scene illumination. Unfortunately, this tactic would greatly increase the required number of oriented histograms. A more intelligent approach can be discovered by examining how a histogram is affected by a change in scene illumination.

The observed intensity of any scene is directly proportional to the intensity of the illuminating source [Horn 1975]. This simple linear result holds true regardless of the object's surface composition and can be used to reduce the number of oriented histograms required. Figure 2.1 illustrates the effects of light intensity changes on a given histogram. As can be seen, there is a linear compression of the histogram as illumination is lowered. Given an oriented histogram, therefore, it is a simple matter to generate the oriented histogram corresponding to the same scene at some multiple of the light intensity. Quantization errors will produce meaningless results for very large or very small multiples of intensity, but there will be a range where this technique is applicable. In this range, it is possible to expand or squeeze each oriented histogram to the intensity value of the unoriented histogram and then proceed with the histogram matching process. The one remaining obstacle is that the illumination intensity of the unoriented histogram is not known *a priori*. It is not actually possible to determine the intensity value of the unoriented histogram unambiguously, but there is a technique which will work in many cases.

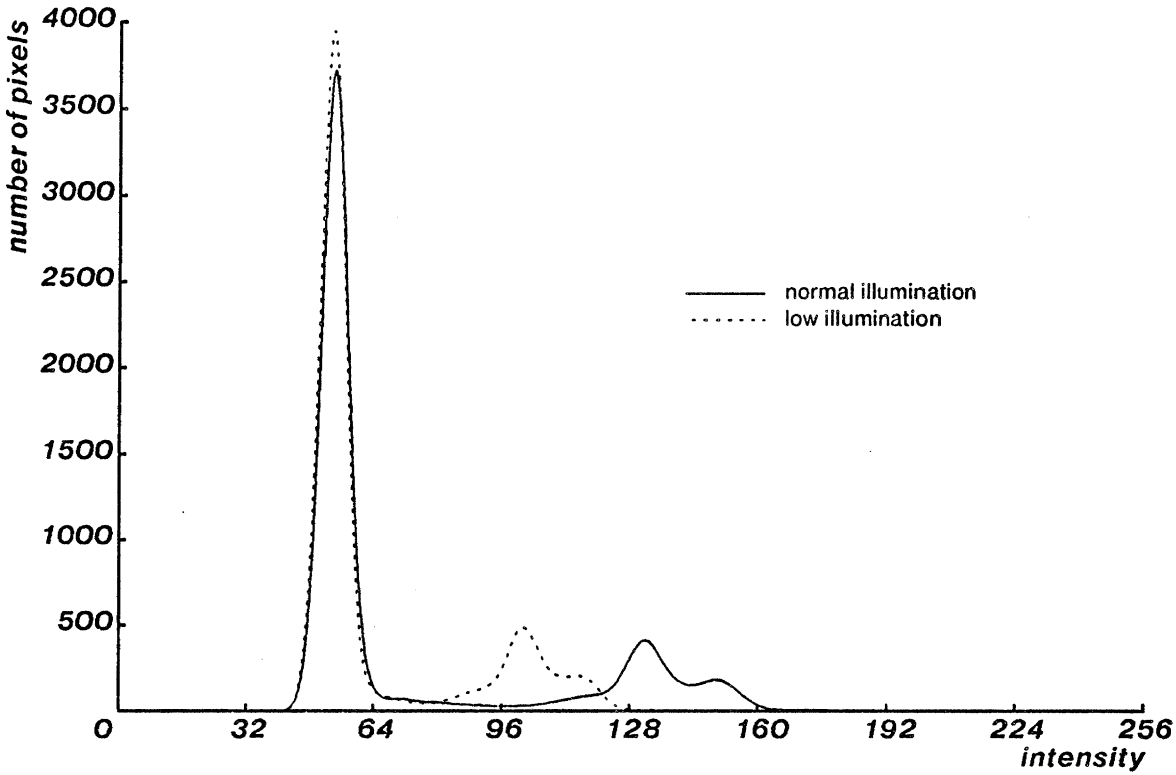


Figure 2.1: Histogram data for one scene at two different illumination intensities. Lowering the intensity compresses the histogram. In principle, the compression should be exactly linear, however, the automatic gain control of the imaging camera provides a slightly non-linear result.

The technique relies on the fact that histograms obtained at a given intensity level often have approximately the same height. For this procedure, each member of the set of oriented histograms is individually expanded or contracted until its highest point is at the height of the highest point on the unoriented histogram. Such a transformation will not always introduce the correct intensity scaling factor because the histogram shapes vary with the orientation of objects in the scene. However, the transformation will produce a nearly correct intensity scaling for the oriented histograms which correspond to scene orientations most closely resembling the scene orientation of the unoriented histogram. Those histograms which are least likely to be scaled correctly are also the ones which are not likely to match the unoriented histogram because they have an inappropriate shape. Conversely, the histograms which are likely to match the unoriented histogram are also the most likely to be scaled correctly, thereby ensuring that they are of low error when evaluating

the difference measure. Such positive reinforcement adds to the robustness of the algorithm, but this procedure involves many simplifying assumptions. It can be thrown off by quantization errors incurred during large swings in intensity or by inherent histogram ambiguity. The real test of the technique will have to come from practical applications.

2.4 Intensity Histograms for Transistor Images

2.4.1 Symmetry and Illumination

The histogram matching technique, along with the other orientation determining methods discussed in this report, was tested by estimating the orientation of a single transistor on a uniform background. The properties of the transistor used were explained and illustrated in chapter 1. Referring to figure 1.1, it would be useful to consider some aspects involved with the symmetry of the problem. Since histograms are independent of rotation about the imaging optics, it is clear that the rotation of the transistor cannot be deduced from the intensity histogram. The only angle remaining is the tilt angle, which is allowed to take values between 0° and 90°. Due to the various symmetries of the problem, tilt is the only orientation information which histograms are capable of determining.

The amount of tilt information contained in histograms is somewhat dependent on the lighting conditions of the environment and the surface material of the object being viewed. The transistor used in this research had a highly specular surface, and the illumination was provided by a single fluorescent "ring" light.

2.4.2 Histogram Regions

Figures 2.2 and 2.3 depict the actual histogram data for a single transistor, against a dark background, at a number of different orientations. The histograms in these figures appear to be divided into 2 or 3 subregions. Each histogram has one region which contains a large number of pixels at low intensity corresponding to the dark background of the scene. Each histogram also has a central region whose shape varies dramatically with changes in orientation. Some histograms contain a peak near the top of the intensity range arising from a specular highlight.

Since the transistor histograms appear to be naturally divided into regions, it may be worthwhile to take a closer look at each of these regions individually. The background section, for example, does not vary much from one transistor orientation to another. Its position and shape are almost constant. The only real change that occurs in the background peak is its height or, equivalently, its area. This change in area is not a useful indication of orientation, however, because a histogram's total

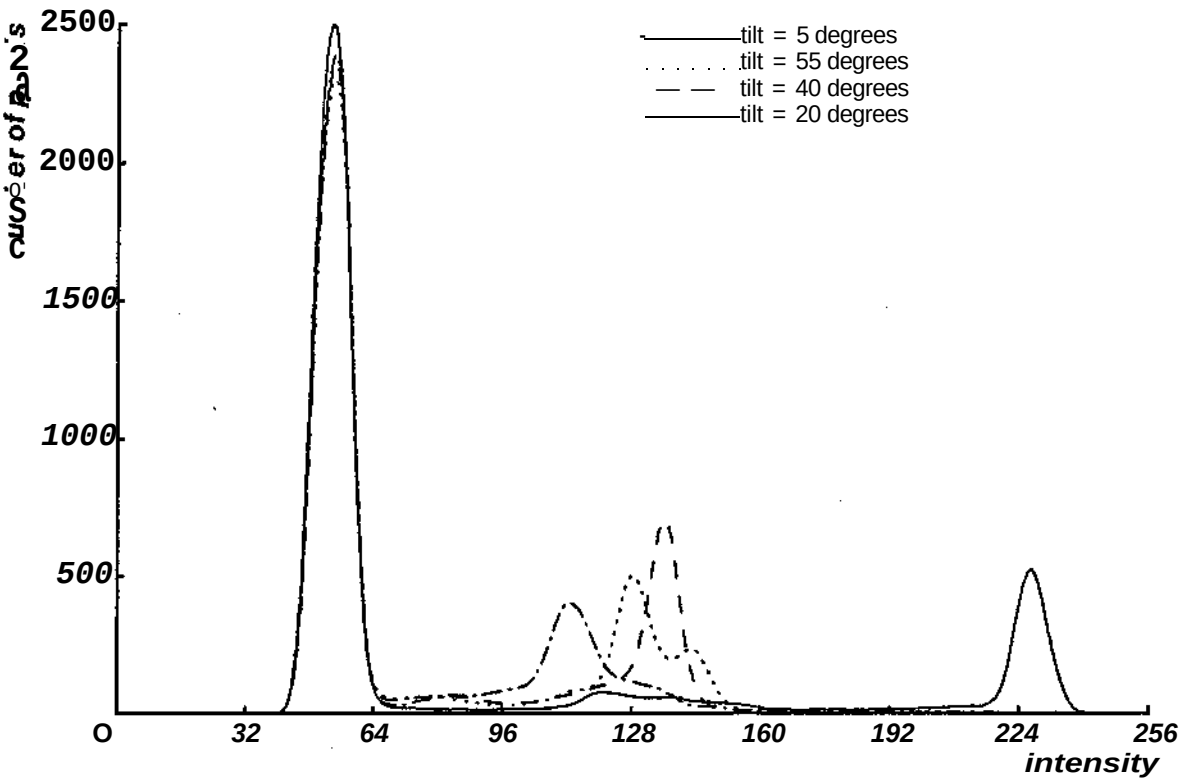


Figure 2.2: Actual histogram data for a single transistor, against a dark background, at four different orientations. There are 256 different grey-levels and a total of 36,864 pixels.

area must be constant. Any change in the area of the background region can be deduced by examining the area of the other histogram sections. In fact, the background region of transistor histograms contains little or no orientation information and can be removed from the histogram with no ill effects to the histogram matching technique. There may even be some positive effects to be gained by removing this high power, low information region of the histogram.

The highlight region is another area of a histogram which deserves further study. This region exists only for those orientations where large areas of the transistor are reflecting illumination from the light source directly into the camera. The simple existence or non-existence of this region constrains the orientation to be within certain ranges. Unfortunately, this constraint is usually not very tight when highlights are absent. For the transistor images, a highlight region is present only for tilts between 0° and approximately 12°. Therefore, knowing that a highlight is not present does not narrow the range of possible tilts by more than 16%. On the other hand, if there is a highlight, this imposes a considerable constraint on the tilt

value.

One note of caution should be mentioned in conjunction with highlights: small changes in the orientation of a specular object might easily produce highlights which will cause large histogram changes. This effect tends to reduce the success rate of matching the intensity histogram unless the training set is very finely quantized and very large.

The remaining region of the histogram, is the central region. This is the region which contains most of the orientation information for all orientations. Due to this fact, it is this region which should be consistently used for the purposes of histogram matching. The highlight region is only useful at certain constrained ranges of tilt and should therefore be considered separately. For this reason, the center histogram section will be known as a "histogram template" and the entire process becomes "histogram template matching".

2.5 Method Summary

Now that the individual elements of histogram template matching have all been discussed, it is time to review how these elements fit together. The first step in the process is to determine the quantization of orientation required for the application. This decision represents a trade-off between high accuracy and high computational costs, in terms of complexity and the storage space required for large training sets. For the transistor application, a quantization of 5° was chosen. This choice implies a relatively small number of training histogrcims ((90° ~-5°) + 1 = 19) and a maximum accuracy of approximately ±2°. The next step is the accumulation of a training set. Obtaining the necessary histograms is a relatively easy task in most applications, but as mentioned previously, the histograms will have to undergo some modifications before they can be used. These modifications are what separates a histogram from a histogram template.

2.5.1 From Histograms to Templates

Histograms can be viewed as discrete estimates of the analog probability distribution function which describes the probability of illumination for any image point (not necessarily a pixel). This estimate may not be very accurate depending on the number and location of samples (pixels) taken from the image. To facilitate the histogram matching process, the estimate can be improved by using a Parzen-window convolution [Parzen 1962] which provides multi-point interpolation through smoothing. In essence, the raw image histogram is convolved with a smoothing filter, known as a Parzen window, to provide a histogram which may better represent the image involved.

Whether the histogram is really improved, or just distorted, depends on the the choice of filtering function. Among other things, the filter must be a legitimate density function whose total area equals one. It is also desirable for the filter to peak at its origin and decrease smoothly in both directions. A typical filter choice which satisfies these criteria is a zero-mean univariate normal density. The variance of such a filter must be chosen carefully to insure that the filtering process does not obliterate useful features of the histogram while still providing a reasonable measure of smoothing [Sanderson 1976]. In general, this variance should become smaller as the number of samples (pixels) grows, but most imaging applications have a constant number of samples so that the variance may be fixed. The variance used to smooth transistor histograms was chosen by trial and error and fixed at $\sigma^2 = 4$, or $\sigma = 2$. The transistor histograms shown in figures 2.2, 2.1 and 2.4 were all filtered by a gaussian with $\sigma = 2$ intensity levels. Figure 2.3 portrays a histogram both before and after smoothing for the purposes of illustration.

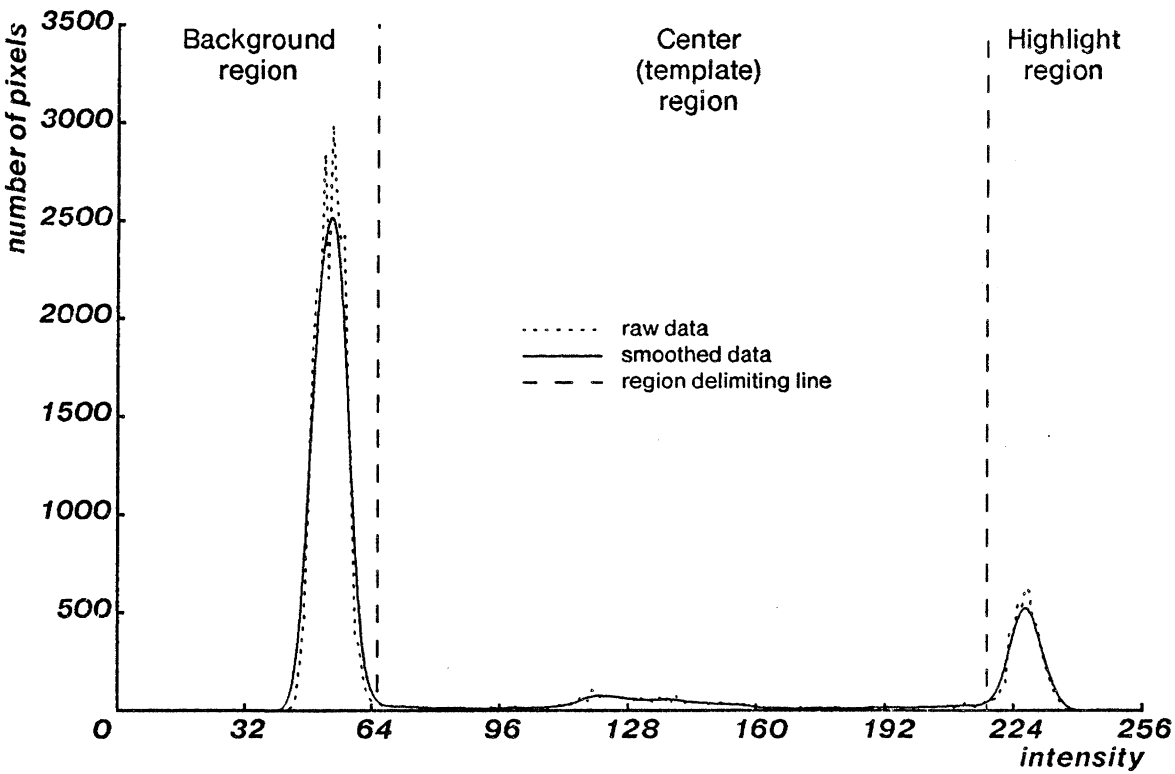


Figure 2.3: Transistor histogram for tilt = 5°. Both the raw and smoothed data are shown along with the 3 different regions of the histogram.

After smoothing, the histogram is partitioned into its component regions using a simple hysteresis peak finding algorithm. Figure 2.3 shows how this algorithm performed on transistor data with tilt = 5°. The highlight and background regions are then removed from the raw histogram and the remaining central region of raw data is smoothed to create the final histogram template. The template formed using the data in figure 2.3 is displayed separately in figure 2.4

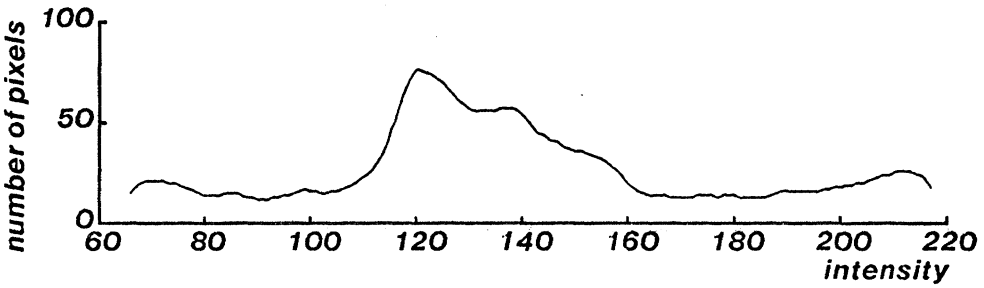


Figure 2.4: Histogram template for tilt = 5°. The template is a smoothed version of the raw data's central region.

2.5.2 Matching

Once all of the training histograms have been converted to templates and stored, it is time to obtain an unoriented histogram and attempt the matching process. The first step in the process involves converting the unoriented histogram to an unoriented template using the steps elucidated in the preceding section. The resulting template is then sequentially compared to all of the oriented histogram templates as described below.

For each template comparison, the template with the greater maximum value is linearly expanded until its height is equal to that of the lower template. This step attempts to correct for any changes in illumination intensity as mentioned in section 2.3.2. Once the expansion is complete, the templates are shifted so that the highest point of each template corresponds to the same intensity value. Equation 2.5 is then applied to produce the normalized correlation value for this pairing of templates. This process is repeated for each training template and the highest of all the correlation values indicates the histogram template of best fit. The orientation of this best-fit histogram is then assigned to the unoriented histogram.

2.6 Results

The histogram template matching technique was used to determine the tilt of 138 transistor images at various orientations. Figure 2.5 displays the results by plotting the estimated tilt against the actual transistor tilt.

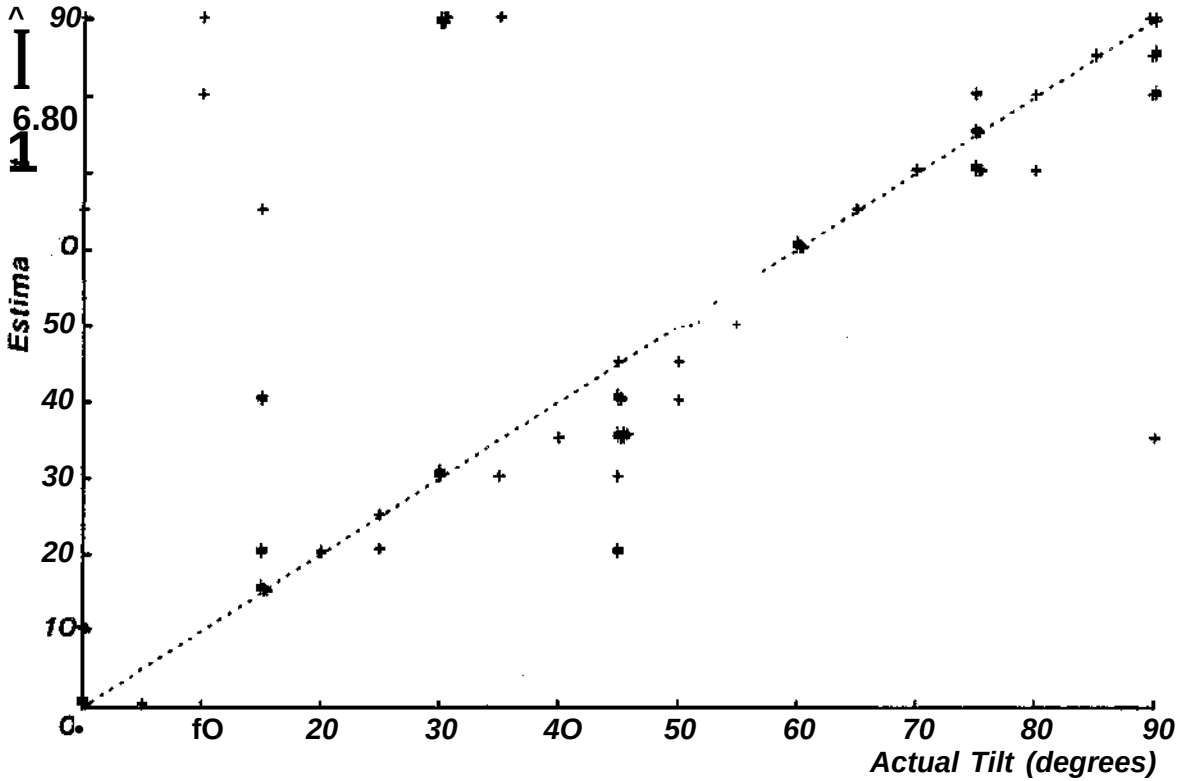


Figure 2.5: Actual transistor tilt vs. estimated tilt obtained from histogram template matching.

On the average, the results exhibit the desired linear relationship between estimated and actual orientation. There is some scatter about the ideal line. The following table provides a concise summary of the magnitude of this scatter.

Error (degrees)	Percentage within error
±0°	**31.9%**
±5°	**60.9%**
±10°	**82.6%**

The data points corresponding to a tilt of 30° exhibit a large systematic error and are worthy of further discussion. Ten of the fourteen images corresponding to a tilt of 30° were estimated to have a tilt of 90°. This error is significant both in the amount of the error, 60°, and the number of points which exhibit such error. The source of this error can be traced, in part, to the shape similarity between the intensity histogram at tilt = 30° and tilt = 90° after correcting for possible changes in illumination. Through sheer coincidence, the histogram of a transistor at a tilt of 30° is quite similar to a low intensity version of the histogram for a tilt of 90°. This type of error occurs because intensity histograms are not guaranteed to display a unique relationship to orientation, especially when arbitrary amounts of expansion or compression are allowed. The camera non-linearities mentioned in figure 2.1 are also partially responsible for the errors of figure 2.5.

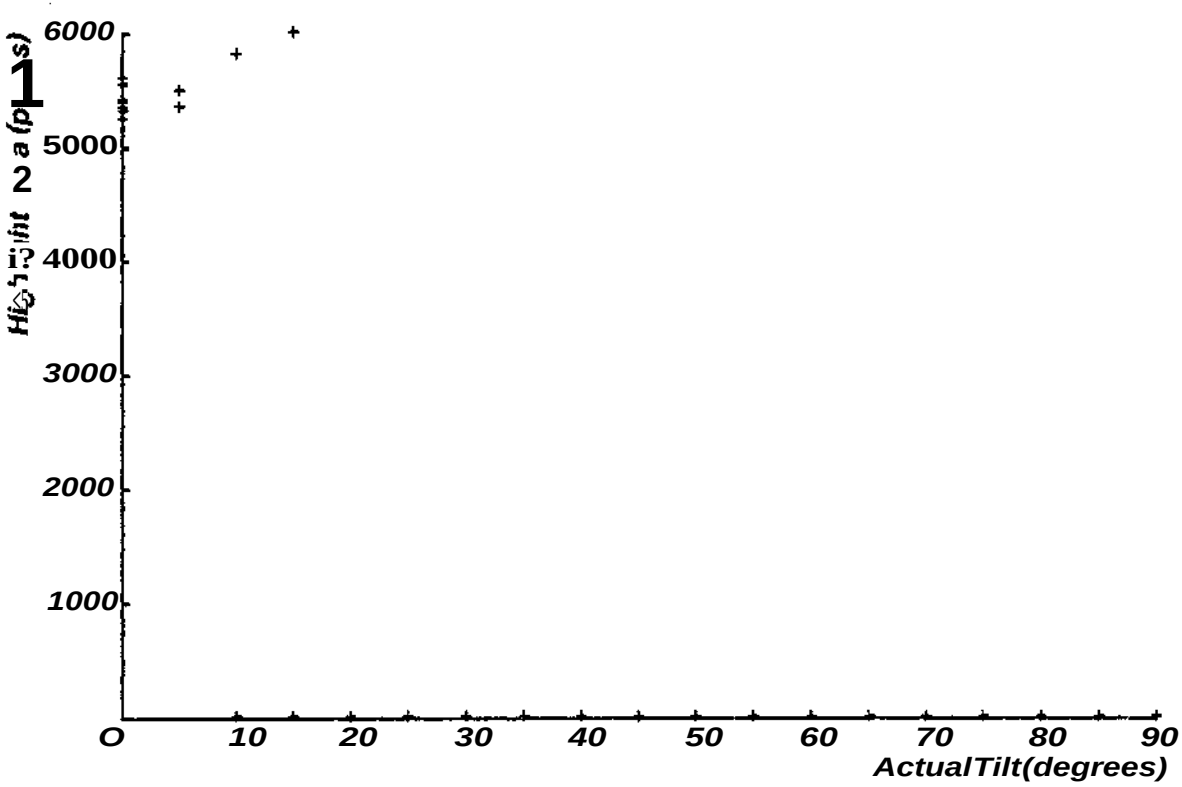


Figure 2.6: Transistor tilt vs. axea of the highlight region. If no histogram highlight region is found, the area is assumed to be zero.

The highlight region of an intensity histogram also provides some measure of the transistor's tilt, as mentioned in section 2.4.2. Figure 2.6 displays the relationship

between tilt and the area of such a highlight region (if any). There is a clear correspondence between low tilt angles and the presence of a highlight region. This relationship is emphasized in the following table.

Tilt (degrees)	Number of Images	Number with Highlight
0°	14	14
5°	2	2
10°	2	1
15°	14	1

The intensity histogram methods presented in this chapter provide a useful measure of transistor orientation. The orientation estimate is imperfect and does not provide information about rotation. In later sections, the histogram template matching technique will be combined with other techniques to significantly improve the accuracy of the orientation estimate.

Chapter 3

Binary Connectivity Analysis

S.I Introduction

Real world objects are composed of surfaces with finite areas. In an image, these surfaces often appear as regions exhibiting some constant image characteristic such as intensity, shading, texture, etc. Each image region, then, represents a portion of a surface, and the shape of that region contains information about the orientation of the surface in the scene. The task of this chapter is to examine a method which will segment these image regions and record their salient features so that orientation information about the underlying surfaces can be extracted. Specifically, this chapter will examine a method to segment the image of a single object on a uniform background based on binary intensity values. A statistical description of the segmented regions is then employed to determine object orientation from a simple heuristic.

The methods of this chapter represent a rather limited subset of region based image segmentation techniques. The technique used in this chapter uses a local measure of similarity to determine connectedness. Other researchers have devised schemes which group pixels based on a global measure of pixel properties [Ohlander 1970]. There are also region merging methods which use boundary "melting" [Brice 1970], or relaxation techniques [Rosenfeld 1978]. See [Fu 1981] for a more complete survey.

The heuristic used to determine the correspondence between segmented region features and orientation is sub-optimal, but computationally inexpensive. The techniques of pattern recognition could be used to determine an optimal decision rule for this case at the cost of increased computation. However, for the transistor application, the increase in performance is slight and does not justify the extra effort,

3.2 Binary Connectivity and Object Orientation

8.2.1 Intensity Based Segmentation

It is generally recognized that a crucial step in the formation of any image description is the determination of the spatial relationships, or connectivity, of pixels which are of the same class [Rosenfeld 1979]. *Connectivity analysis* is a segmentation scheme which describes an image as a collection of regions that are, in turn, composed of connected groups of pixels displaying some common trait. These connected regions, or "blobs", axe then described by aggregate properties of the region's pixels.

The criteria selected to define the division of pixels into classes greatly affects the image description obtained from connectivity analysis. In general, the class description of a pixel may be specified using any combination of primitive, or localized, pixel properties such as brightness, color, local texture, etc. The most commonly used criteria, however, is simply monochromatic intensity. In fact, some research (and most industrial vision) uses only two classes of intensity [Agin 1980] [Ward 1979] [Sanderson 1983] [Kelley 1982]. This simple scheme provides practical benefits because binary images greatly simplify the analysis algorithms. Binary algorithms can often be expressed as logical operations involving only a very local group of pixels and are, therefore, very fast.

In order to apply these binary algorithms to a grey-level image, it is necessary to threshold the grey-level input image to create a binary image. This "binarization" should be performed in a manner which preserves the desired image regions. The selection of an appropriate threshold is the most straightforward in situations where the image contains objects of reasonably uniform brightness against a uniform background of different intensity. In this case, it is a simple matter to select a threshold using the grey-level image histogram. The type of image tinder discussion always yields a bimodal histogram, and a threshold can be chosen on the basis of the minimum histogram point between the two peaks. Histogram smoothing (see section 2,5.1) is usually required before attempting to find the minimum, but this is a simple operation and does not hamper the selection of an appropriate binarization threshold. Images containing non-uniform objects or non-uniform background may require more complex threshold selection methods (see [Weseka 1976]).

3.2.2 From Segmentation to Orientation

Given an appropriate binary image, it is worthwhile to examine how connectivity analysis might describe the various image regions. The description desired should be more compact than an array of intensity values, but should not involve a reduction in critical image information. The contradictory nature of these goals, along with the fact the information which is critical in one application may be irrelevant in another, has prevented any universally accepted optimal representation. A widely used convention for region description involves the use of statistical parameters. Typically, the zeroth, first, and second order moments of inertia of the blob are used. These parameters enable the recovery of axea and position information, as well as the size and orientation of the approximating ellipse.

The statistical description of a blob is not a perfect description, however, because statistical parameters do not uniquely identify the region which they axe trying to describe. One set of parameters could correspond to a large number of differently shaped regions. Using statistical parameters alone, it is impossible to find the smallest rectangle which will enclose the region, let alone determine the shape of the region's boundary. For this reason, the statistical description is usually augmented to include, at least, the minimum and maximum cartesian coordinates of the blob. In this research, the region description also contains a full representation of the region's boundary. This hybrid statistical description captures the regions useful information in a compact, easily accessible format.

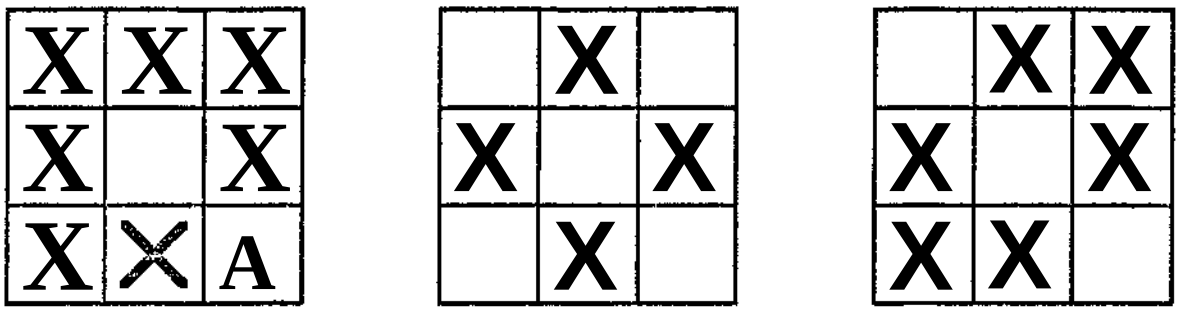
The process of transforming a binary input image to a statistical description of the blobs which compose the image is known as Binary Connectivity Analysis (BCA). This algorithm attempts to capture image information which may be relevant to the determination of object orientation. Finding the exact relationship between the statistical blob descriptions and the orientation of the corresponding object in the scene is, in general, a non-trivial task. The general problem of finding three-dimensional orientation from two-dimensional regions cannot be solved uniquely, and high level domain knowledge is usually required.

As mentioned previously, the training based techniques of pattern recognition have been used extensively to incorporate domain knowledge into systems using statistical descriptors. In some cases, however, it becomes possible to use heuristics to transform blob features to the orientation of the object. Such a method lacks generality because of the application-specific knowledge used to generate the heuristics, but in some situations this poses no problems. For the purposes of this research, a simple heuristic was used to relate the statistical blob description to orientation. This heuristic will be described shortly.

3.3 Binary Connectivity Analysis

8.S.I Definition of Connectedness

Before it is possible to analyze an image, a precise definition of connectedness is necessary. Pixels of the same class which are adjacent axe defined as connected, but it is not always clear which pixels should be considered adjacent. For a rectangular grid, the two most natural conventions are 8-connectedness and 4-connectedness (See figure 3.1 (a) and (b)). 4-connectedness is defined by allowing a pixel to be connected to any of its nearest non-diagonal neighbors which belong to the same class. 8-connectedness permits a pixel to be connected to any of the pixel's 8 nearest neighbors, including those along the diagonal directions.



(a) (b) (c)

Figure 3.1: Definitions of (a) 8-connectedness, (b) 4-connectedness, and (c) 6-connectedness. The X's indicate which of the pixels in a 3 by 3 neighborhood may be connected to the center pixel.

Both of these conventions, however, present an undesirable paradox which is illustrated with the aid of figure 3.2 (a). Under 8-connectedness, figure 3.2 (a) consists of a continuous black ring on a white background. Notice that although the ring is considered to be a continuous figure, it does not in any way separate the white area inside the ring from the white area outside the ring; the background is considered to be a single connected area according to the concept of 8-connectedness. This causes an apparent contradiction with real-world objects, which cannot totally surround an area and yet not define that area as a separate region. The problem stems from the fact that pixels may be connected at an infinitely small point, while real-world objects must have some non-zero thickness. The same problem crops up using the definition of 4-connectedness, but in a reversed form. Under 4-connectedness, figure 3.2 (a) depicts three disconnected black figures which somehow manage to divide the white background into two distinct regions.

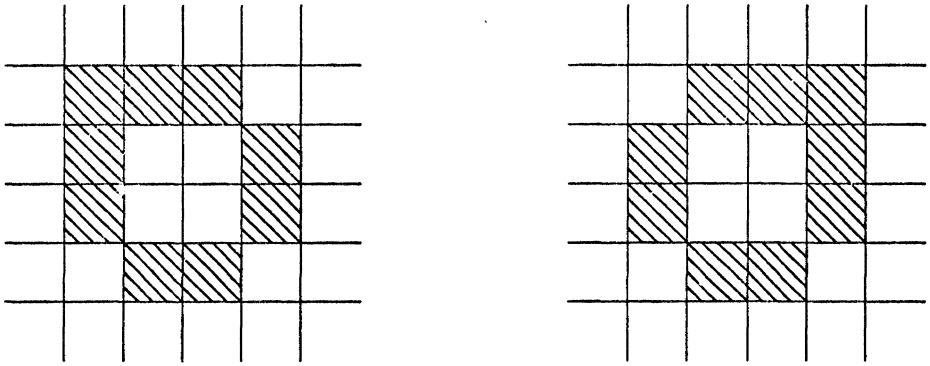


Figure 3.2: Figure (a) demonstrates the connectivity paradox that arises from using the definition of 8-connectedness (or 4-connectedness) on both the black and the white pixels. 6-connectedness resolves this paradox, but produces orientation dependent results. In (a), 6-connectedness defines two disconnected black regions, however when the figure is rotated 90°, as in (b), 6-connectedness discovers only one black region.

There are two generally accepted methods for dealing with this paradoxical situation. The first method involves defining a new concept of connectedness known as 6-connectedness. 6-connectedness is illustrated in figure 3.1 (c) and is guaranteed to produce an interpretation of connectedness which is not paradoxical. Unfortunately, 6-connectedness has some weaknesses. Figure 3.2 (b) shows figure 3.2 (a) rotated by 90°. By applying the definition of 6-connectedness to both versions of the figure, it should be clear that interpretation of the various regions is changed by more than a simple rotation. In fact, the underlying connectedness of the figure is dependent upon the orientation of the figure being segmented. This orientation dependency may be unimportant for some applications because it only occurs for objects which meet at a single point.

There is, however, a second method which gets around the orientation problem completely. This method uses different definitions of connectedness for different classes of pixels. If, for example, all white pixels were grouped using 8-connectedness and 4-connectedness was used to define black blobs, there would never be paradoxes or orientation dependencies. This scheme complicates the connectivity algorithms to some extent, but produces a conceptually clean view of connectedness. For the research conducted in this paper, this "double standard" definition of connectedness is adopted.

3.3.2 Binary Connectivity Algorithm

The connectivity algorithm accepts a binary image as input and produces a statistical description of the image. Since efficient implementations of the algorithm are readily available [Cunningham], detailed discussion will be kept to a minimum. Instead, a general explanation of the algorithm will be provided, followed by an example. Before examining the algorithm, it will be worthwhile to review the elements of the output description.

Each image is described as a hierarchically linked list of blobs. For each blob, the output representation contains all of the zero-th, first and second moments of area. $\sum x$ (the sum of the x coordinates for all pixels in the blob) and $\sum y$ represent the first moments. $\sum xy$, $\sum x^2$, and $\sum y^2$ represent the second moments, while $\sum 1$ (the number of pixels in the blob) represents area, the zero-th moment. The color of each blob (black or white) is also recorded, along with the maximum and minimum values of x and y, and a representation of the blob's boundary. The hierarchical tree structure of the blob list arises because — assuming the image is surrounded by black pixels — every blob in the image is surrounded by a blob of the opposite color. Any blobs contained within a given blob represent the child nodes of the tree structure while the surrounding blob is the parent node.

The connectivity algorithm is a raster scan operation designed to sequentially examine each image pixel in conjunction with three of its neighbors. This process is accomplished by scanning the image with a 2 by 2 image window whose lower right-hand corner always contains the current pixel. Since each of the four pixels being examined can only be colored black or white, there are 16 possible patterns that may appear in the 2 by 2 window. Each of these window states is numbered using the values of the window pixels to form a four bit binary number as illustrated in figure 3.3 (a). This window state number, combined with the partially completed statistical description, is sufficient to specify the action to be taken by the algorithm at each pixel. The correspondence between state number and algorithm action is specified through a software look-up table. The algorithm always assigns the pixel to the blob in which it belongs, and updates the necessary statistical summations; it may also update the maximum or minimum values, the boundary description, or even the hierarchical blob list, depending on the situation.

A simple example, based on figure 3.3 (b), will help to elucidate the procedure. Figure 3.3 (b) contains a 3 by 4 "image", marked by a thick solid rectangle and surrounded by black pixels. The surrounding pixels are not part of the image, but are introduced by the connectivity algorithm to provide a black background for consistent hierarchical structure. The 2 by 2 connectivity window is passed over the figure such that its lower right-hand corner sequentially coincides with the numbered pixels. At the first pixel, the window state number is 1; this signifies that a new blob has been discovered (the background blob was the old blob). This new blob is assigned the label A, to distinguish it from all other blobs, and the appropriate

statistics are updated. The second image pixel corresponds to state number 8. This state number implies that the current pixel belongs to the same blob as the pixel directly above it; therefore, the current pixel is aswsigned to the background blob and the background statistics are updated. The third image pixel corresponds to state 1, as did the first pixel. The third pixel, therefore, must represent a new blob and is assigned the label *B.*

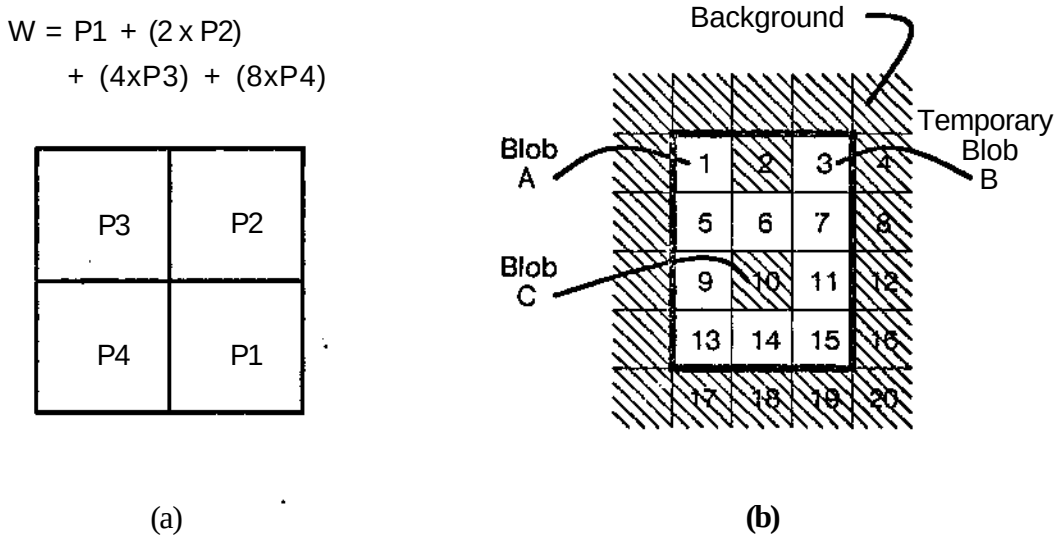


Figure 3.3: Figure (a) illustrates the computation of the window state number, *W,* where P_x can be either 0 or 1 — black or white — for x = 1,2,3,4. Figure (b) is small example image, surrounded by black pixels.

At this point, it appears that the connectivity algorithm has made a mistake. There is only one white blob in the image, but it currently has two labels associated with it. This situation is not, however, an error; it is a necessary part of the algorithm. The raster scan nature of the algorithm makes it impossible to determine that pixels 1 and 3 belong to the same blob until pixel 7 is reached. By the time the algorithm reaches pixel 7, pixel 4 has been assigned to the background, pixels 5 and 6 have been determined to be part of blob A, and the correct statistics have been updated. Pixel 7 is then examined and found to correspond to state number 11 which means that the blob label of the pixel in the upper right-hand corner of the window must be compared to the blob label of the lower left hand pixel. If these labels are different, as in the current example, the two blobs involved both belong to the same blob and must be merged together. The merge is accomplished by adding the statistics of blob *B* to blob *A* and deleting all references to blob £, thereby leaving a single blob with the label *A.*

Image pixels 8 and 9 are easily assigned to the correct blobs using the window state number. Pixel 10, however, corresponds to state 14 which, like state 1, starts a new blob. The new blob receives the label *C.* Pixels 11 through 14 and 16 through 19 correspond to simple window states and are assigned to their appropriate blobs using a simple look-up table. Pixel 15, on the other hand, is another example of state 11 and requires more complex processing. For pixel 15, the blob labels of the upper right and lower left hand window pixels axe both A In this case, blob *A* has completely surrounded blob *C* (the blob corresponding to the pixel in the upper left hand corner of the connectivity window), signifying the termination of blob *C* and the parent-child relationship between *A* and *C.* Pixel 20 corresponds to state 4, but the processing is just the same as it was for pixel 15: the upper right and lower left window pixels are compared. The comparison indicates that the background has closed around blob A, thereby terminating the blob and the algorithm. The resulting linked list of descriptors is ready to be applied to the task at hand.

8.8.8 Determining Blob Features

The zero-th, first, and second moments of area, defined for each blob during binary connectivity analysis, can be used to determine some basic features of the blob. The blob area, for example, is already recorded as the zero-th moment. The centroid of the blob is specified using the first and zero-th moments of area:

$$x_{cent} = \frac{\sum x}{\sum 1} \qquad y_{cent} = \frac{\sum y}{\sum 1}$$

The centroid calculated in this manner is relative to the origin of the image. The moments of inertia, $Y^{x<1}$ and Cl^{2} and the product of inertia, $Y1^{X}V$ are also calculated relative to the image origin.

For many applications, it is useful to calculate the parameters of an ellipse which has the same moments of inertia as the blob under analysis. If the blob is roughly elliptical in shape, this approximating ellipse will be a very good characterization of the shape and orientation of the blob (the size and position will always be the same). To calculate the parameters of the approximating ellipse, it is necessary to first calculate the second moments of the blob with respect to the blob's centroid. The moment of inertia about $y = y_{cent}$ and the moment of inertia about $x = x_{cent}$

(Iy), and the product of inertia about x_{cent},y_{cent} (P_{xy}) are given by:

$$7_{-} - \mathbf{V} \sum_{y}^{\wedge} - \frac{(\mathbf{Ev})^2}{\sum 1}$$

$$I_y = \sum x^2 - \frac{(\sum x)^2}{\sum 1}$$

$$P_{xy} = \sum xy - \frac{\sum x \sum y}{\sum 1}.$$

Using these quantities to construct a Mohr's circle [Beer 1977], it is possible to find the principal axes of inertia about the centroid. The principal axes emanate from the blob centroid and are rotated by an angle *9,* in a counterclockwise direction, from the coordinate axes of the input image array. Where the *9* is given by:

$$\theta = \frac{1}{2}\arctan\left(\frac{2P_{xy}}{I_y - I_x}\right).$$

The principal axes specified by *6* are also the principal axes of the approximating ellipse. Therefore, *$* completely describes the orientation of the major axis of the ellipse. Since the blob's moments of inertia about its principal axes *{Imax* a^d *Imin)* must be equivalent to the approximating ellipse's moments about the same axes, it is possible to calculate the length of the ellipse's major and minor axes. From Mohr's circle, I_{max} and $/_{mzn}$ are given as:

$$\cdot I_{max} = \frac{I_x + I_y + \sqrt{(I_x - I_y)^2 + (2P_{xy})^2}}{2}$$

$$I_{min} = \frac{I_x + I_y - \sqrt{(I_x - I_y)^2 + (2P_{xy})^2}}{2}$$

These quantities are equated with the principal moments of an ellipse to obtain:

$$I_{max} = \frac{}{4} a^3 \&> \qquad Imin = \frac{\ddot{}}{4} ab^3$$

where *a* is the length of the ellipse's semi-major axis and 6 is the length of the semi-minor axis. Solving for a and 6, yields:

$$a = \frac{\sqrt{I_{max}}}{\sqrt[8]{I_{max} I_{min} \left(\frac{\pi}{4}\right)^2}}, \qquad o = \frac{\sqrt{I_{min}}}{\sqrt[8]{I_{max} I_{min} \left(\frac{\pi}{4}\right)^2}}$$

Having calculated a, 6, and 5, the approximating ellipse is now fully specified (the formula for calculating the centroid is given above). The parameters of this ellipse can be used as features that further describe the shape and orientation of the blob.

Another generally useful feature that can be derived from the output of BCA is blob perimeter length. Assuming that pixels represent tiny squares that compose the image, the perimeter of a blob corresponds to.the pixel edges which contact a pixel of the opposite color. The boundary of a blob is described, in part, by the number of horizontal and vertical edge segments $(N^{\wedge}$ and $N_v)^{\wedge}$ as well as the total number of corners (iV_c)? in the perimeter. It would be simple to calculate the perimeter length of a blob by simply adding the number of horizontal and vertical edge segments, however, this measure exhibits considerable orientation dependence. For example, a square measuring 50 pixels a side would have a perimeter length of 200 if exactly aligned with the coordinate frame of the image array. If, on the other hand, this square were rotated 45°, the sum of horizontal and vertical edge segments would be greatly increased due to the "staircase" effect. [Agin 1984] has suggested that the orientation dependence can be reduced by subtracting small amounts from the perimeter length for each corner in the boundary. His empirical formula which provides a good approximation to ideal perimeter length is:

$$\text{Perimeter} = 0.948059 \left(N_h + N_v - \frac{1}{2}\left(2 - y/\pounds\right) N_c\right) .$$

This formula is optimized to produce correct perimeter lengths for circular blobs, and exhibits some length error for other blob shapes.

The features mentioned above are those typically used in industrial vision applications. The output of BCA, however, is general enough to allow the computation of other useful features. These features can then be used to describe a blob and provide clues to the orientation of objects in the underlying scene.

3.4 Applied Binary Connectivity Analysis

In order to apply BCA to the task of determining transistor orientation, it is first necessary to acquire and threshold a grey-level transistor image. The threshold can be chosen on the basis of the intensity histogram as described in section 3.2.1. The transistor image histograms presented in chapter 2 clearly indicate the large intensity difference between the two regions which permits the use of this thresholding technique. The entire transistor is easily distinguished from the background of the image using a single global intensity threshold. Any small specks which transcend the threshold can be rejected after connectivity analysis by ignoring all but the biggest blob. The result of thresholding using histogram information can be seen in figure 3.4. This binary image is then segmented and reduced to a statistical description by BCA.

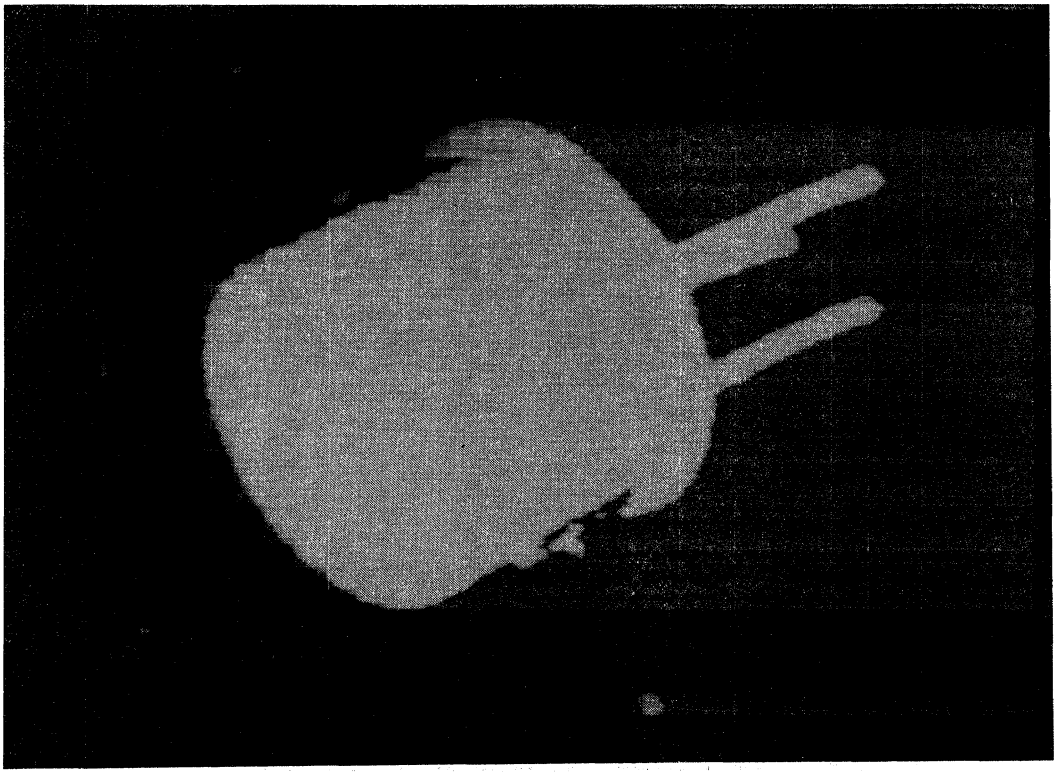


Figure 3.4: Binarized version of original image. Rotation is 150°, tilt is 60°

The relationship between binary blob features and transistor rotation is relatively straightforward to determine because the cylindrical axis of the transistor always corresponds with the long axis of the segmented blob — unless tilt = 0°, where the value of rotation is arbitrary. The relationship between blob features and transistor tilt is not quite as simple to compute and will not be considered for the sake of simplicity. Post-segmentation processing, then, simply involves determining the orientation of the major axis for the blob's approximating ellipse, as mentioned in section 3.3.3. This angle is used as the estimate of transistor rotation.

3.5 Results of Binary Analysis

The methods described in this chapter were applied to 138 images of transistors at different orientations. For each image, an estimate of transistor rotation was extracted. Figure 3.5 displays the results of this operation by graphing the estimated rotations versus the actual transistor rotations.

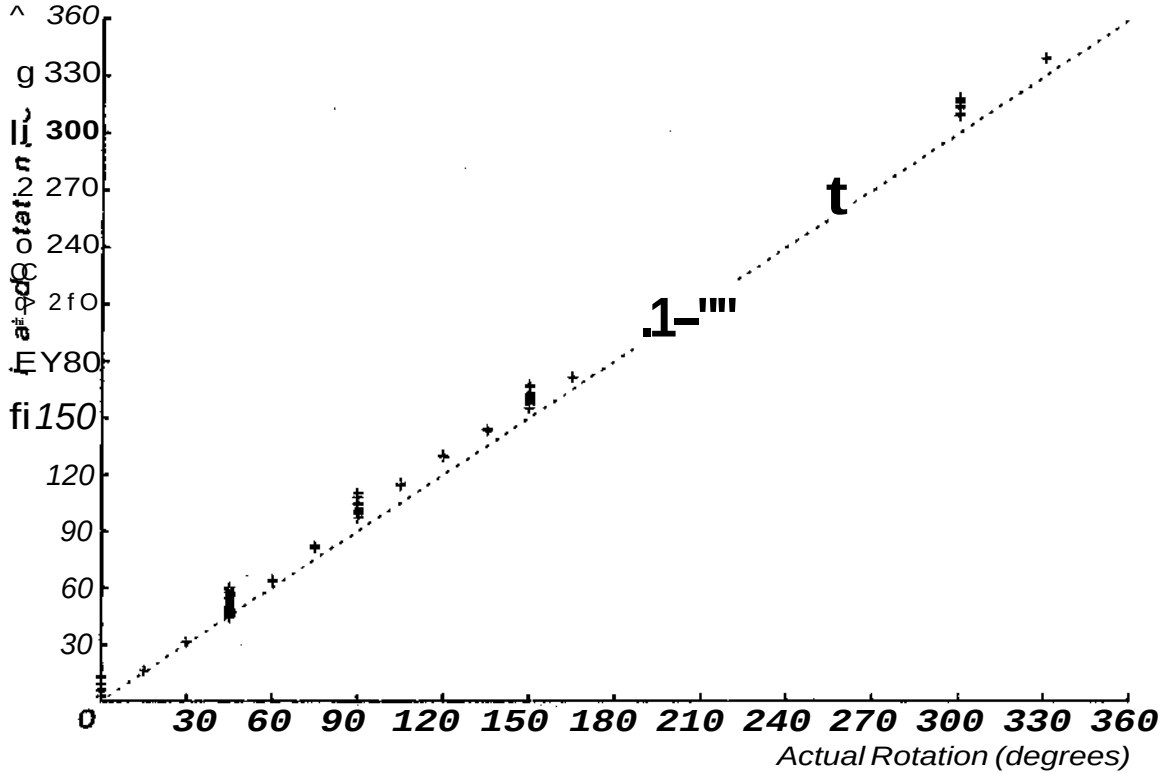


Figure 3.5: Actual transistor rotation vs. rotation estimated from binary connectivity analysis. Estimated rotation values are ambiguous ±180°.

The results of figure 3.5 are generally good. The data displays the desired linear relationship between actual and estimated rotation values, and all of the estimated rotations are within 5% (19°) of their actual values. There is clearly a small systematic error, however, because every estimate is slightly higher than the actual value. The average value of this error is approximately +10°.

This error can be directly attributed to blob asymmetries introduced by the presence of the transistor leads. The heuristic rule used to determine the correspondence between blob features and transistor rotation assumes that the transistor can be approximated by a cylinder. Looking at figure 3.4, this assumption is clearly not completely valid and, therefore, directly affects the results.

It should be noted that data of figure 3.5 is actually ambiguous ±180° due to the inherent ambiguity of the blob's approximating ellipse. Notice also that no rotation values are plotted for transistors with a tilt of 0° because they are meaningless.

Chapter 4

Ellipse fitting

4.1 Introduction

The projection of a three-dimensional circular surface onto a two-dimensional image plane yields an ellipse whose shape is directly related to the surface orientation. This relationship can be exploited to determine the orientation of objects under various imaging conditions. By extracting or enhancing the ellipse edge points, an ellipse may be fit to these points in the image plane. The parameters of this ellipse may be used to estimate three-dimensional surface orientation. This chapter describes an algorithm, based on [Agin 1981], for fitting ellipses to a set of image points. This algorithm is then applied to the problem of estimating transistor orientation.

Ellipse fitting may be applied to determine the orientation of any cylindrical object with visible end-planes. In this case, image preprocessing and the identification of ellipse boundary points are required as a basis for ellipse fitting. Knowledge of the cylindrical geometry of the object may then be used to infer object orientation.

Ellipse fitting may also be used in conjunction with structured lighting techniques, such as light-striping [Agin 1973] [Shirai 1971] [Bolles 1981] [Vanderbrug 1979], to determine object orientation. When a projected plane of light strikes a cylindrical surface, the stripe of illumination appears as a partial ellipse in the image

plane. By estimating the parameters of this partial ellipse, it is possible to deduce the local orientation of the cylindrical surface for a specified camera and light plane geometry.

One of the major difficulties in applying these techniques is the determination of which image points correspond to the perimeter of the ellipse. Some high level information is necessary to determine the region of interest. The problem of fitting an ellipse to a set of data points in a plane will be addressed initially by assuming that some set of "interesting" image plane points has already been determined. The question of how to determine interesting points will then be discussed.

4.2 Fitting Ellipses to Points

4-2.1 The Ellipse Equation

An ellipse which has its center at the origin of the coordinates and which is aligned with the coordinate axes can be represented by the equation

$$\frac{x^2}{a^2} + 6^2 = 1,$$

where a and *b* are the length of the semi-major and semi-minor axes respectively, $a > b$. Such an ellipse is illustrated in figure 4.1a.

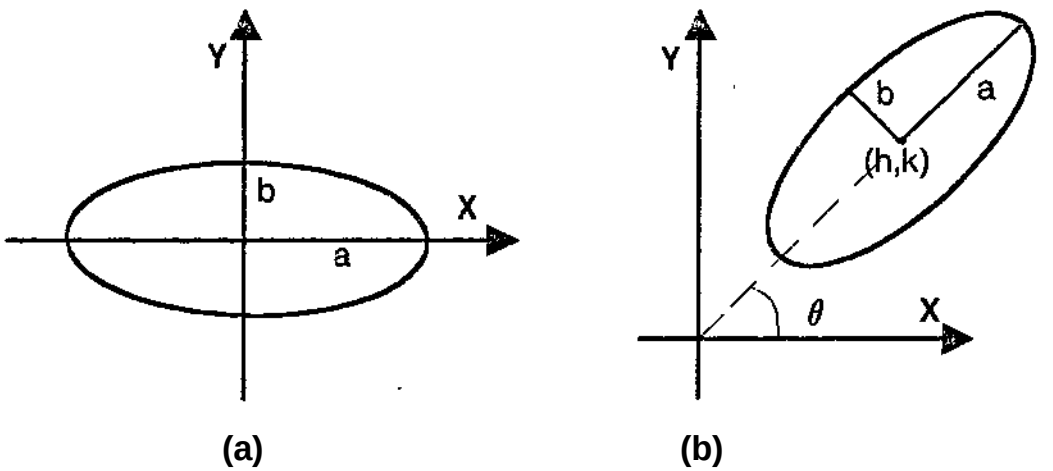


Figure 4.1: (a) An ellipse aligned with the coordinate axes and centered on the origin, (b) An ellipse with arbitrary parameters.

In general, an ellipse may be translated such that its center is at any arbitrary point (h, k) and may be rotated such that it forms an angle θ with the coordinate axes. A rotated and translated ellipse is shown in figure 4.1b, along with the five parameters needed to fully specify its position (a, b, h, k, θ). The equation of an ellipse at any arbitrary rotation and translation is given by

$$\frac{x'^2}{a^2} + \frac{y'^2}{b^2} = 1 \tag{4.1}$$

where

$$x' = (x - h)\cos\theta + (y - k)\sin\theta$$
$$y' = -(x - h)\sin\theta + (y - k)\cos\theta\,.$$

Equation 4.1 can also be written as

$$G(x, y) = Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0\,. \tag{4.2}$$

This form of the ellipse equation is, in fact, the general equation for second order curves and is capable of representing all conic sections [Lynch 1973]. For equation 4.2 to represent an ellipse, it is necessary and sufficient that $B^2 - 4AC < 0$. This paper will assume that this ellipse condition is met; other conics will be ignored (but not rejected by the mathematics). Under this assumption, the coefficients of equation 4.2 can be explicitly expanded as follows:

$$\begin{aligned}
A &= a^2\sin^2\theta + b^2\cos^2\theta \\
B &= (b^2 - a^2)2\sin\theta\cos\theta \\
C &= a^2\cos^2\theta + b^2\sin^2\theta \\
D &= -2h(a^2\sin^2\theta + b^2\cos^2\theta) - k(b^2 - a^2)2\sin\theta\cos\theta \\
E &= -2k(a^2\cos^2\theta + b^2\sin^2\theta) - h(b^2 - a^2)2\sin\theta\cos\theta \\
F &= h^2(a^2\sin^2\theta + b^2\cos^2\theta) + hk(b^2 - a^2)2\sin\theta\cos\theta + k^2(a^2\cos^2\theta + b^2\sin^2\theta)
\end{aligned} \tag{4.3}$$

4.2.2 Generalized Curve Fitting

A generally accepted method for fitting functions such as equation 4.2 to scattered data points is the method of "least squared error" [Wiley 1966]. This method uses an error measure to describe the discrepancy between any given data point and the function being fit. The total error between the function and the data is then computed by summing the squared error from each individual data point. By minimizing this total error, the function parameters which best fit the data can be estimated.

4.2.3 An Error Measure for the Ellipse

It is usually desirable to fit curves in a way which is general, simple to compute, and invariant to translation, rotation, and scaling of the input data. Accomplishment of these goals depends heavily upon the error measure chosen. Intuitively, it seems that the (shortest) perpendicular Euclidean distance from the ellipse to a data point would provide a useful error measure. As described below, however, this error measure leads to intractable analytical results requiring iterative solutions. Therefore, there is a need to consider alternative error measures.

An examination of the problem of fitting a straight line to data points in a plane may provide some suggestions for such an alternative error measure. In the straight line case, the *perpendicular* distance between a point $P\backslash = (zi,yi)$ and a line $ax + by + c = 0$ is given by

$$\text{Distance} = \frac{|ax_1 + by_1 + c|}{\sqrt{a^2 + b^2}}.$$

The validity of this equation is easily verified by viewing the line $ax + by + c = 0$ as the intersection of the plane $z = ax + by + c$ with the xy plane, as illustrated in figure 4.2. The line-to-point distance d is equal to the height h of the plane at Pi,

$$h = \backslash axi + byi + c\backslash ,$$

divided by the maximum slope of the plane,

$$\|\nabla z\| = \sqrt{\left(\frac{\partial z}{\partial x}\right)^2 + \left(\frac{\partial z}{\partial y}\right)^2} = \sqrt{a^2 + b^2}.$$

This distance measure can be used to obtain a least squared error fit for the straight line case. The distance measure is convenient because it yields a closed form solution which is not degenerate for vertical lines. Traditional linear regression methods which use vertical distance, rather than perpendicular distance, as a measure of error also provide a closed form solution, but are degenerate for vertical lines.

In an attempt to apply a similar distance measure to the ellipse equation, an ellipse can be viewed as the intersection of an elliptic paraboloid with the xy plane, as shown in figure 4.3. The equation of an elliptic paraboloid is obtained by simply removing the restriction on equation 4.2, $G(x,y) = 0$, so that $G(x_1y) = z$ (under the assumption that $J3^2 - 4AC < 0$). Now, the distance from the ellipse $G(x> y) = 0$ to a point Pi can be obtained approximately as

$$\text{Distance} \approx \frac{G(x_1, y_1)}{\|\nabla G(x,y)\|\Big|_{(x_1,y_1)}}. \qquad \textbf{(4.4)}$$

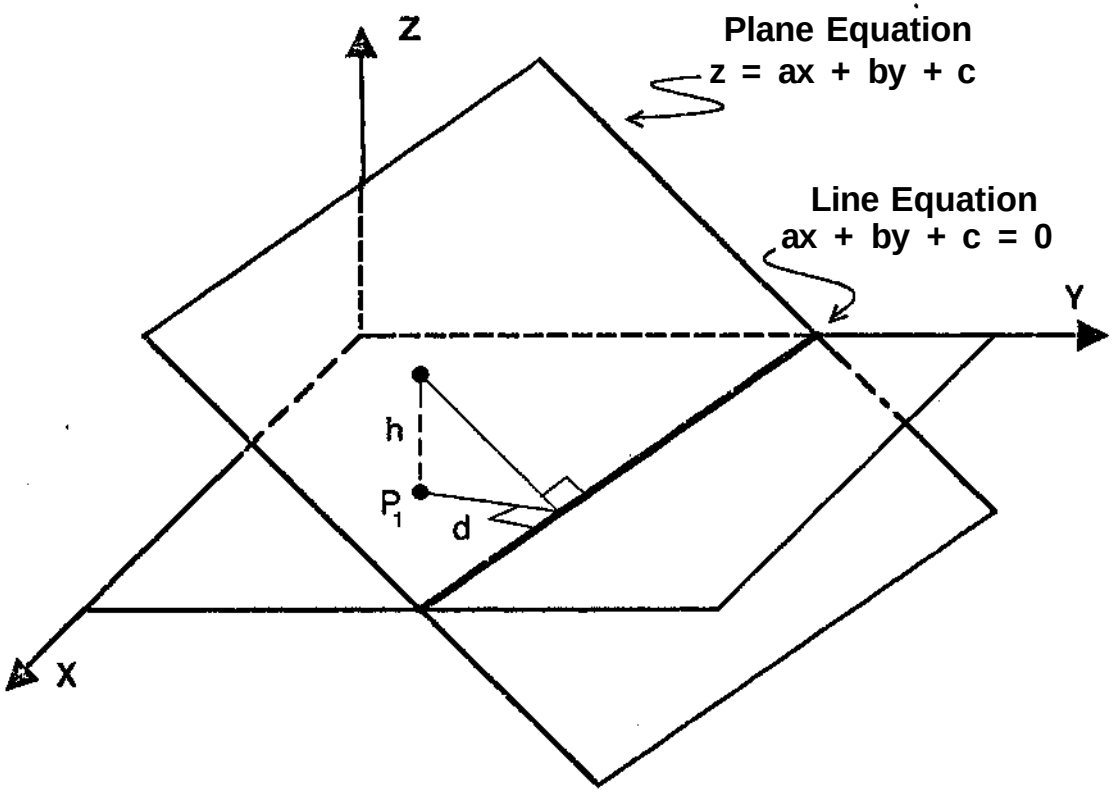


Figure 4.2: The line $ax+by+c = 0$ is represented by the intersection of the plane $z = ax + by$-\-c with the xy plane. The distance d from the line to point $P\backslash = (zi,yi)$ in the xy plane is equal the height h of the plane at rci,j/i divided by the magnitude of the plane gradient, ||VJS?||.

Although this distance measure is approximate, it has been found to provide good curve fits even for difficult cases such as data from very eccentric ellipses [Agin 1981]. The distance measure still requires iterative methods for the minimization of total square error. Many people [Paton 1970] [Albano 1974] [Cooper 1976], therefore, use a less accurate, but computationally less expensive measure of error which can provide a solution using only linear algebra techniques.

The error measure which is usually used is the equation of the elliptic paraboloid itself, |G(x,t/)|. This measure retains a rough qualitative resemblance to that of equation 4.4 since |G(x,y)| increases monotonically with distance along any line in the xy plane which is perpendicular to the ellipse. The error function does not, however, relate distance directly. In fact, the rate at which error increases with distance will vary depending upon which perpendicular line is chosen.

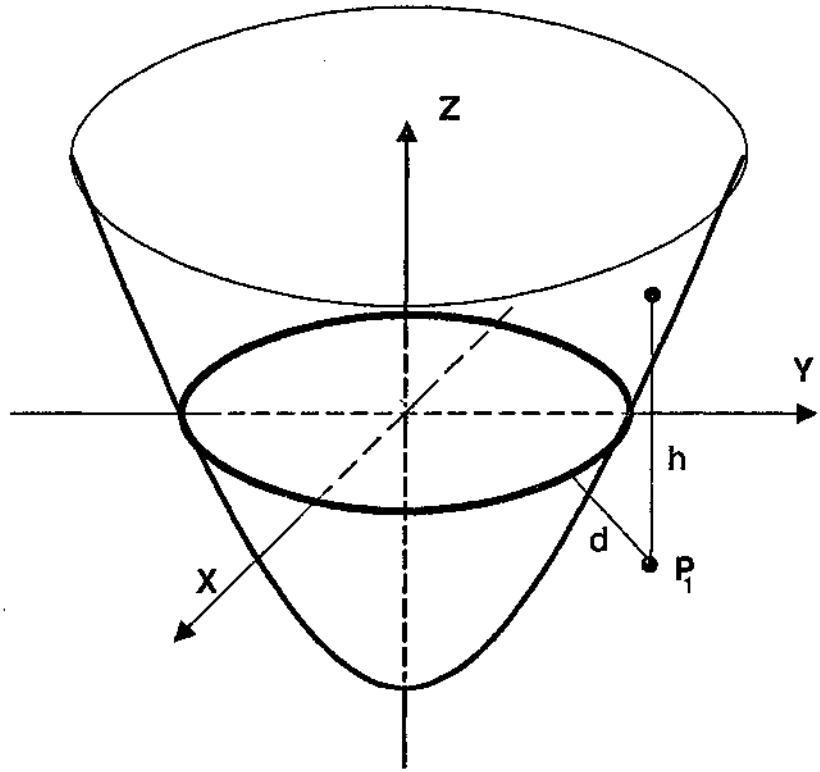


Figure 4.3: An ellipse can be represented as the intersection of an elliptic paraboloid with the *xy* plane. The distance *d* from the ellipse to a point Pj in the *xy* plane is monotonically, but non-linearly, related to the height *h* of the paraboloid above Pi.

4-2-4 Constraining the Minimization

Using \G{x, y)\ as the pointwise error measure, the total squared error, 3, is given by

$$\Xi = \sum_{\text{all } (xi \wedge i)} \left(G(x_i, y_i)\right)^2. \qquad (4.5)$$

To determine the best fit ellipse, S must be minimized with respect to the parameters of *G(x,y): A,B,C, D,E,* and *F.* Such a minimization is usually accomplished by taking the partial derivatives of 3 with respect to A, JB, C, D, 2£5 and *F* and then setting these equal to zero. In this case, directly applying such operations yields the trivial solution *A~B~C_= D = E = F —-0.* The reason for this rather disappointing result is shown in figure 4.4. An infinite number of elliptic paraboloids can be used to describe a single ellipse in the *xy* plane; therefore, the pointwise error (and the total error), can be made arbitrarily small by choosing an appropriately shaped error function. To avoid this problem, it is necessary to place a constraint

on the coefficients of $G(x, y)$ so that the error function can have only one shape for a given ellipse.

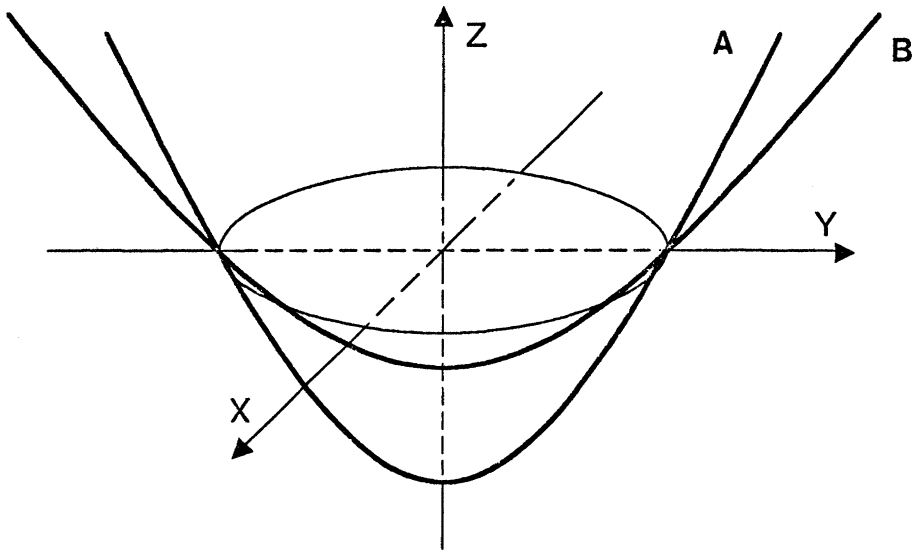


Figure 4.4: Elliptic paraboloids A and B both represent the same ellipse in the xy plane. In fact, an infinite number of such paraboloids are capable of describing this ellipse.

The constraint used must be chosen carefully so that the shape of the error function does not change if the ellipse data is rotated or translated in the xy plane. More generally, the constraint should be chosen so that the error measure and the resulting curve fit are invariant with respect to the equiform group of transformations in the Euclidean plane: rotation, translation and scaling [Bookstein 1979]. This means that if an equiform coordinate transform is performed on the data, the new best-fit curve must be exactly equal to a transformed version of the curve which best fit the original data.

One constraint which satisfies the invariance requirement [Bookstein 1979] is

$$A^2 + B^2/2 + C^2 = \text{constant}.$$

By substituting the explicit expressions for A, B, and C from equation 4.3, it is easy to show that this constraint is equivalent to

$$a^4 + b^4 = \text{constant}$$

which is clearly invariant under rotation and translation. Another constraint which satisfies the invariance requirement is the "average gradient constraint" [Agin 1981].

The average gradient constraint requires that the mean-square gradient of $G(x,y)$, as measured at all of the data points, is equal to unity. This can be expressed as

$$\frac{\sum_{\text{all } (x_i,y_i)} \|\nabla G(x,y)\|^2 \Big|_{(x_i,y_i)}}{n} = 1, \tag{4.6}$$

where n is the number of data points. This constraint was chosen because "on the average" it tends to reestablish the approximate link between $G(x,y)$ and Euclidean distance, as shown in equation 4.4. It does not, however, provide any theoretical increase or decrease in curve fit error when compared with the first constraint.

4.2.5 Minimizing the Total Squared Error

For the sake of illustration, this paper will follow the notation of [Agin 1981] and use the average gradient constraint to constrain the minimization. A shift to a more compact notation will make the exposition easier. Equation 4.2 can be rewritten as

$$G(x,y) = V^T X = X^T V,$$

where

$$V = \begin{pmatrix} A \\ B \\ C \\ D \\ E \\ F \end{pmatrix} \quad \text{and} \quad X = \begin{pmatrix} x^2 \\ xy \\ y^2 \\ x \\ y \\ 1 \end{pmatrix}.$$

The total squared error Ξ, from equation 4.5, then becomes

$$\begin{aligned} \Xi &= \sum_{\text{all } (x_i,y_i)} \Big(G(x_i,y_i)\Big)^2 = \sum_{\text{all } (x_i,y_i)} \Big(V^T X X^T V\Big) \\ &= V^T \left\{ \sum_{\text{all } (x_i,y_i)} \Big(XX^T\Big) \right\} V = V^T P V, \end{aligned} \tag{4.7}$$

where P is now a matrix containing the sums of x_i, y_i raised to various powers. Using vector notation to express the average gradient constraint, equation 4.6 becomes

$$\sum_{\text{all } (x_i,y_i)} \|\nabla G(x,y)\|^2 \Big|_{(x_i,y_i)} = n,$$

where

$$\|\nabla G(x,y)\|^2 = \left(\frac{\partial G}{\partial x}\right)^2 + \left(\frac{\partial G}{\partial y}\right)^2 = \left(V^T \frac{\partial X}{\partial x}\right)^2 + \left(V^T \frac{\partial X}{\partial y}\right)^2.$$

Defining

$$X_X = \frac{dX}{dx} = \begin{pmatrix} 2x \\ y \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad Xy = \frac{dX}{dy} = \begin{pmatrix} 0 \\ x \\ 2y \\ 0 \\ 1 \\ 0 \end{pmatrix},$$

the average gradient constraint can be written in its final form

$$\sum_{\text{all } (x_i, y_i)} V^T \left(X_X X_X^T + X_Y X_Y^T \right) V$$

$$= V^T \left\{ \sum_{\text{all } (x} 1 \quad \{X_X X_X^T + X_Y X_Y^T) \right\} V$$

$$= V^T Q V = \text{n}.$$

Here, Q is a matrix much like P which is composed of the sums of powers of x and y. For any given set of data, P and Q are constant.

Now that the error equation and its corresponding constraint have been expressed in matrix form, it is time to get down to the business of minimizing $H = V^T P V$ under the constraint $V^T Q V$ = n. Such problems in constrained minima are traditionally solved using the method of Lagrange multipliers [Lynch 1973]. Lagrange's method states that at the constrained minima the following condition must be true:

$$\nabla_V \left(V^T P V \right) = \lambda \nabla_V \left(V^T Q V \right).$$

After taking the required gradients, this condition reduces to

$$PV = XQV. \tag{4.8}$$

The solution to this equation is clearly an eigenvalue problem, but because Q is singular, some rather unusual methods are required for the final solution f. The method used by [Agin 1981] yields five eigenvalues and five eigenvectors. The eigenvector corresponding to the smallest eigenvalue represents the coefficient vector V which is the minimum error solution. This fact can easily be deduced by premultiplying both sides of equation 4.8 by V^T and then realizing that the result is equivalent to S = An. Since n is a constant, total error is minimized for the smallest eigenvalue A.

f If $A? + B^2/2 + C^2$ = constant is used as the minimization constraint, the eigenvalue problem can be solved using standard techniques.

At this point, it may be worthwhile to look at one concrete example of ellipse fitting in order to obtain a qualitative estimate of algorithm performance. Figure 4.5 depicts a set of data points in the plane (represented by "+") and the ellipse which was fit to those points using the technique described above. Qualitatively, the fit appears excellent; it is not, however, perfect. In general, ellipse fitting based on the minimization of the general conic (equation 4.2) is inherently inaccurate due to the non-linearity of the error measure. These inaccuracies tend to favor ellipses which are more elongated than expected [Ballard 1982]. While this tendency is not very noticeable in the example of figure 4.5, the problem gets worse for scattered data generated by partial or very eccentric ellipses.

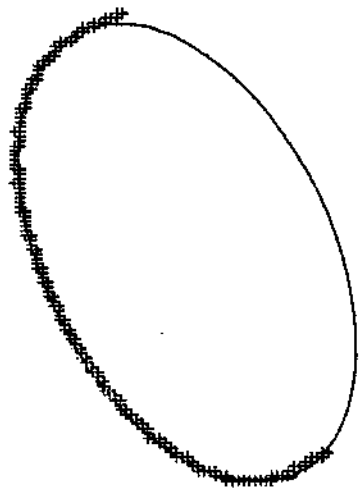


Figure 4.5: A sample ellipse fit. The data points to which the ellipse was fit are denoted by "+".

Given this method for fitting ellipses to collections of points on the plane, it is still necessary to determine which image points are to be fit. The next section will describe one possible method for the determination of candidate ellipse points and its use in the transistor application.

4.3 Applied Ellipse Fitting: Determining Orientation

*4*3.1 Object Boundary Methods*

Real world images may contain many combinations of features which might be described as elliptical, "partially" elliptical, "approximately" elliptical, etc. Fur-

thermore, these features may correspond to changes of intensity, texture, distance, reflectance, *ad infinitum* in the scene. It is clearly beyond the scope of this paper to examine any number of these cases in depth, therefore, only some simplified subset of real world images will be considered.

One possible simplification is to examine only those portions of the image which correspond to the boundary of an object in the scene. Object boundaries are often easy to determine because the discontinuity in scene distance creates distinctive features in the image. Object boundaries also provide some measure of an object's shape, hence they may be able to suggest elliptically shaped edges. Finally, object boundaries vary with the viewing orientation of the object. The combination of the above properties make object boundaries an ideal choice for a simple feature to which ellipses can be fit with the ultimate goal of determining object orientation. The determination of orientation, however, may require some *a priori* information about the object to which the ellipse is being fit.

4.3.2 Transistor Boundaries

The transistor images provide an excellent example. As discussed in chapter 3, it is easy to segment the image region corresponding to the transistor from the background, thereby obtaining the transistor image boundary. Figure 4.6 illustrates the results of performing this operation on a typical transistor image.

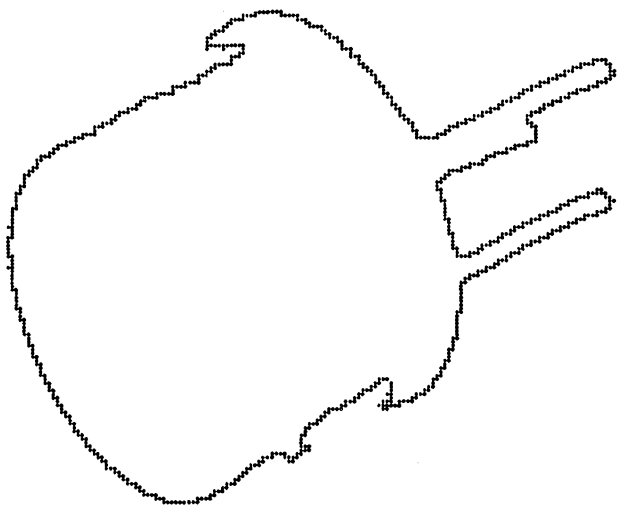


Figure 4.6: The object boundary for a transistor with tilt = 60° and rotation = 150°.

The transistor outline of figure 4.6 clearly exliibits curve segments which are approximately elliptical in shape. The task now is to identify these curve segments and use the corresponding points as input for ellipse fitting. In the simplified domain of object boundaries, it is possible to compute a curvature measure for any given curve segment. This curvature measure will provide enough information to identify boundary regions which may correspond to elliptical edges.

4-3.3 The Mathematics of Curvature

A curve in the *xy* plane, such as the one shown in figure 4.7a, can be described in a number of equivalent ways. The usual formulation for such a curve is $y = /(z)$, however, the curve might also be described parametrically as *(x(t),y(t))* using the independent variable t. Another possible formulation of the curve equation is $a = 7(5)$, where *a* is the direction of the curve with respect to the *x* axis and *s* represents *arc length*, the distance traveled along the curve. This form of the curve equation requires the specification of a starting point in order to represent a unique curve, but is otherwise equivalent to the more standard curve equations. This last form of the curve equation also proves to be the most natural for describing the concept of curvature.

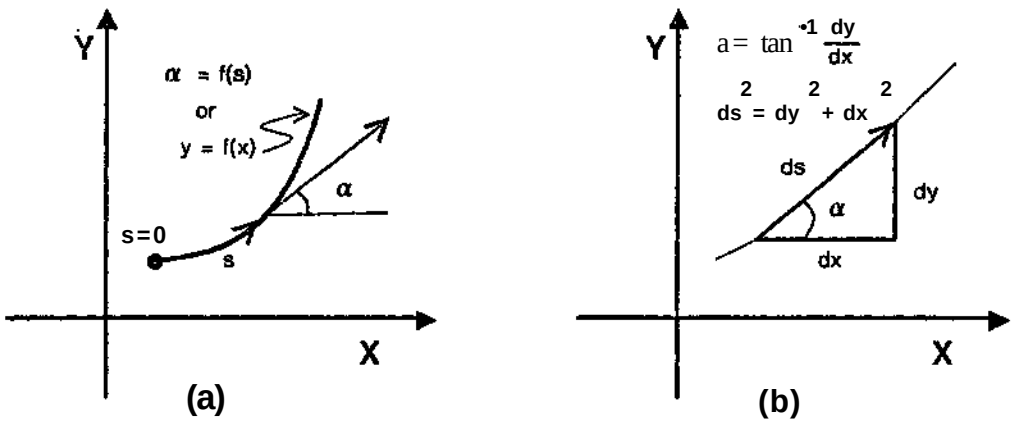


Figure 4.7: (a) The definition of a curve as either $y = f(x)$ or $a = f(s)$. (b) The relationship between *dy^dx* and *a, ds.*

Curvature is defined as the rate of change of direction along a curve. In other words, curvature, &, is a measure of how fast *a* changes in relation to *s.* This definition is described mathematically by

$$k(s) = \frac{da}{ds}. \qquad \textbf{(4.9)}$$

As one can see, it is mathematically straightforward to define a curvature measure for a curve described by $\alpha = f(s)$. Unfortunately, many curves are represented as $y = f(x)$. In order to describe curvature as a function of x, it is necessary to make the following transformations, as illustrated in figure 4.7b:

$$\alpha = \arctan\left(\frac{dy}{dx}\right)$$

and

$$ds = \sqrt{dx^2 + dy^2} = dx\sqrt{1 + \left(\frac{dy}{dx}\right)^2}.$$

Noting that

$$\frac{d\alpha}{dx} = \frac{1}{1 + \left(\frac{dy}{dx}\right)^2}\frac{d^2x}{dy^2}$$

and

$$\frac{ds}{dx} = \sqrt{1 + \left(\frac{dy}{dx}\right)^2},$$

curvature can be expressed as

$$k(x) = \frac{d\alpha/dx}{ds/dx} = \frac{d^2x/dy^2}{\left(1 + \left(\frac{dy}{dx}\right)^2\right)^{3/2}}. \tag{4.10}$$

4.3.4 Curvature of Real Object Boundaries

Referring to figure 4.6, it is obvious that neither equation 4.9 nor, equation 4.10 are immediately applicable to the curve which represents the transistor boundary. Firstly, the chain-coded transistor boundary is not in the form of an analytic function of any type and, therefore, can not simply be plugged into one of the curvature equations. Secondly, even if the boundary were in some analytical form, the boundary equation would be almost useless for the purposes of measuring curvature because noise and quantization errors would be greatly accentuated by the differential operations [Bennett 1975]. To get an estimate of the curvature for a real boundary requires the use of some special curvature estimator.

One technique for estimating the curvature of chain-coded boundaries is to fit low order polynomials, usually parabolas, to small subsections of the boundary [Smith 1984]. These polynomials can then be differentiated analytically and the

results plugged into equation 4.10 to determine the curvature of that boundary section. This method reduces noise and quantization errors by fitting a curve to the average trend of the boundary points.

A somewhat more direct approach is suggested by figure 4.8. In figure 4.8a, a straight line L_2 is fit to a small subsection of the curve (between points P_a and P_b) in order to estimate α directly. Another straight line is fit to a section of boundary points at a distance Δs further along the curve to obtain a different α value, as illustrated in figure 4.8b. From equation 4.9 the curvature can be estimated directly by subtracting the two α's and dividing the result by Δs, $k \approx \Delta\alpha/\Delta s$. Once again, smoothing is obtained during line fitting by averaging the error from a number of points.

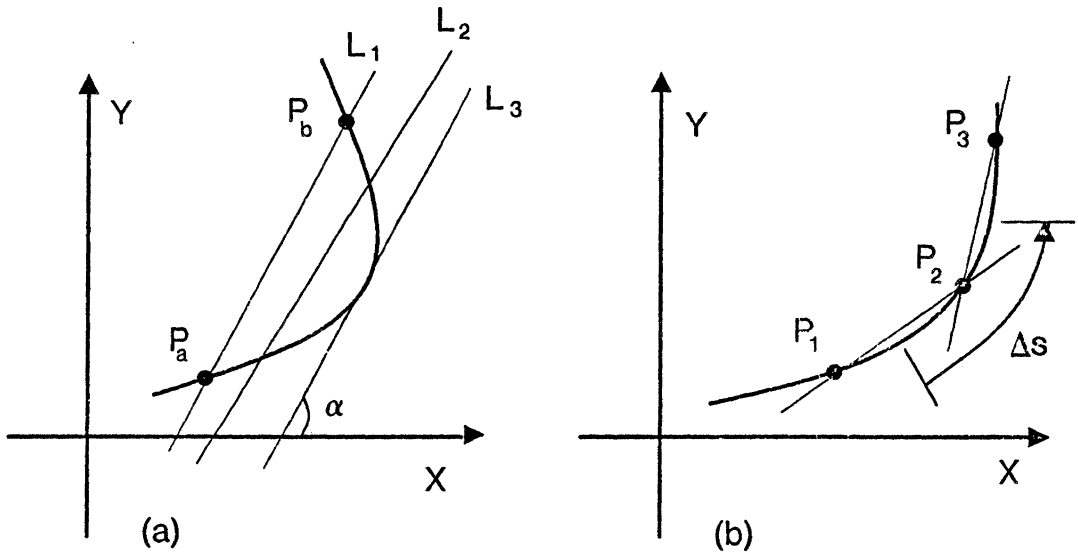


Figure 4.8: (a) L_1 and L_2 illustrate two different methods to estimate α for the curve section between P_a and P_b. L_3 is a line parallel to L_1. (b) Curvature can be estimated by obtaining estimates of α at two different curve positions and then subtracting these estimates and dividing by Δs.

The algorithm actually used to estimate the curvature of the transistor boundary represents a slightly simplified version of the previous method. In order to increase processing speed, each estimate of α was accomplished using a straight line fit to only the end points of the boundary subsection. This is illustrated by L_1 in figure 4.8a. This simplified method is clearly more susceptible to errors due to noise and quantization effects, but these errors can be removed by smoothing later on.

The results of applying this quick but noisy curvature operator to the transistor boundary can be found in figure 4.9.

It should be noted that the method of curvature estimation depicted in figure 4.8b assumes that *As* is equivalent to the distance between the midpoints of the curve subsections. In principle, *As* should represent the arc length between the curve points whose directions are given by the two estimates of *a.* Referring to figure 4.8a, the mean value theorem states that somewhere between P_a and $P^{\wedge}$ there must be a line L3 which is tangent to the curve direction and parallel to $L\backslash$. If the curvature is constant between P_a and P&, then L3 will be tangent to the midpoint of the curve. If the curvature is non-uniform, then £3 may be tangent to the curve at a point other than the midpoint. In this case, the true value of *As* will be different from the distance between the two curve midpoints, thereby distorting the curvature estimate.

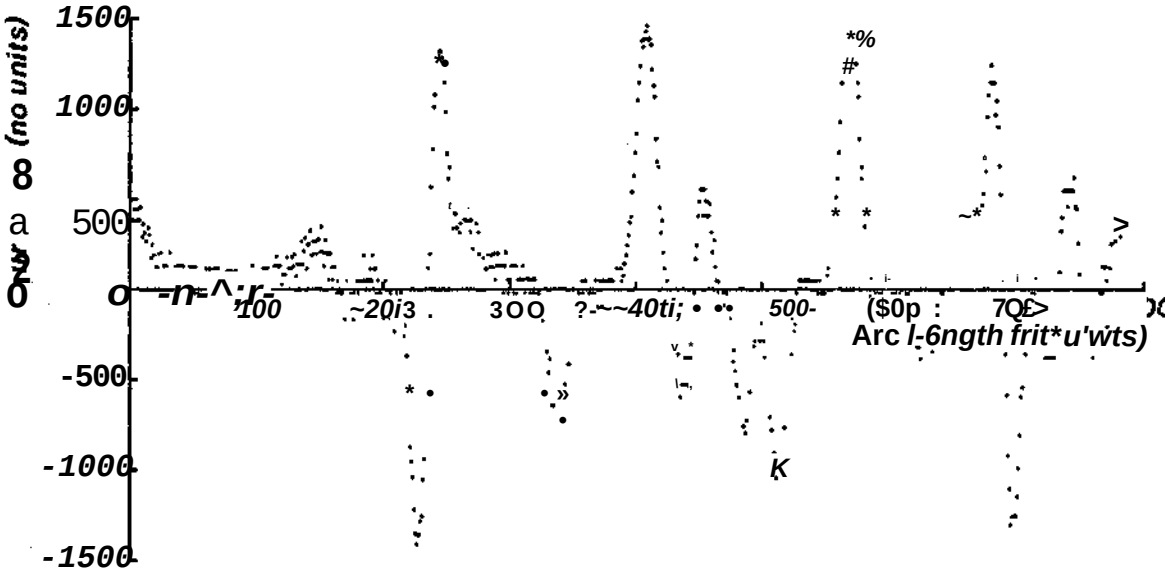


Figure 4.9: A graph of curvature vs. arc length for the transistor outline shown in figure 4.6. Positive values of curvature indicate convex curves, negative values indicate concave curves. Arc length is 0 at the bottom-most pixel of the boundary and increases in a clockwise direction.

As expected, figure 4.9 exhibits considerable noise patterns. By smoothing the curvature estimate with a 1-dimensional gaussian filter, most of this noise can be eliminated, as in figure 4.10.

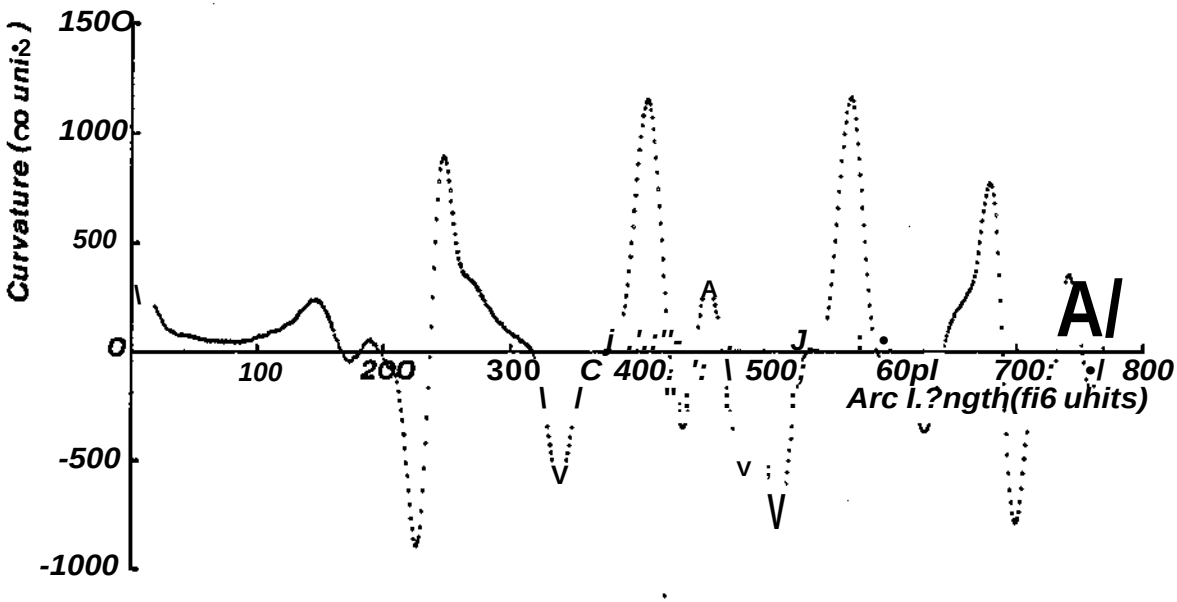


Figure 4.10: A smoothed version of figure 4.9.

4-3.5 Interpretation of the Curvature Estimate: Where's the Ellipse?

Now that a smooth estimate of tiie transistor's boundary has been obtained, it must be used to segment those boundary regions which may have arisen from elliptical edges in the image. The difficulty of this task depends heavily on the task domain and the availability of *a priori* information about the types and sizes of ellipses that may be present. If very small ellipses may be present, then almost any convex boundary section might belong to such an ellipse. Similarly, it is easy to confuse straight lines with very elongated ellipses because they both have sections of approximately zero curvature.

For a transistor boundary, the areas of interest are those which correspond to the top and bottom edge of the transistor cylinder. In principle, the top and bottom edges should be equally useful for determining orientation. In practice, the top edge is usually easier to find and more useful for future orientation operations. In the curvature graph, the elliptical edge generated by the transistor top should be visible as a long region of positive curvature. With this powerful *a priori* constraint, it is possible to generate a simple, but effective curvature segmentation rule: select the boundary region which exhibits the longest stretch of positive curvature.

Applying this criterion to the curvature graph of figure 4.10 generates the result shown in figure 4.11. The points represented by "+" in figure 4.11 are the points to which an ellipse is to be fit using the methods enumerated in the first sections

of this chapter. The results of such an ellipse fit have already been presented in figure 4.5. The fit appears remarkably good when compared to the input points which represent the partial ellipse. When compared to the transistor image seen in figure 4.12, however, there is a distinguishable error between the fit ellipse and the top of the transistor cylinder. The minor axis of the fit ellipse is 19% larger than it should be to accurately describe the transistor top. This error will create significant inaccuracies in the orientation estimate.

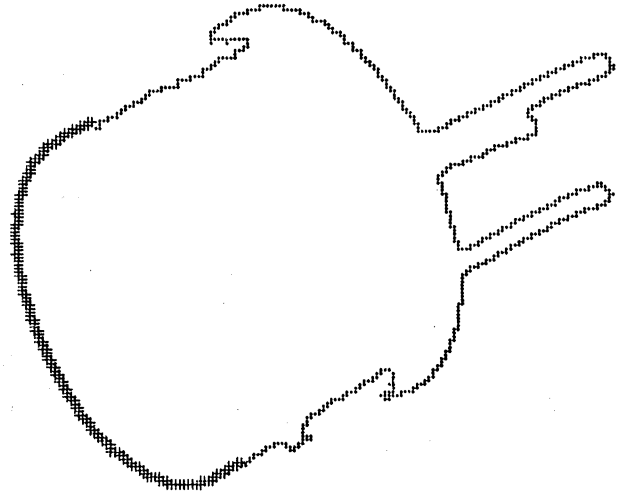


Figure 4.11: The section of the transistor boundary marked by "+" is selected for ellipse fitting, based on its curvature profile.

4.3.6 Using Intensity Gradient Information for Ellipse fitting

In some ways, it is not surprising that there should be such an error in the ellipse fit result. Of all the information contained in a real world image such as the transistor image, only object boundary information was used for the purposes of ellipse fitting. This simplification represents a tremendous amount of data reduction. In most cases, the transistor boundary only contains half of an elliptical boundary. The other half of the ellipse is ignored for the sake of simplicity. Now it is time to consider some of the image information corresponding to this missing half.

The most obvious image feature which can be used to deduce the complete position of the ellipse corresponding the transistor top is the edge where the top meets the transistor body (see figure 1.2). This highlight region defines a segment of the ellipse which can not be obtained from the transistor boundary. If it were

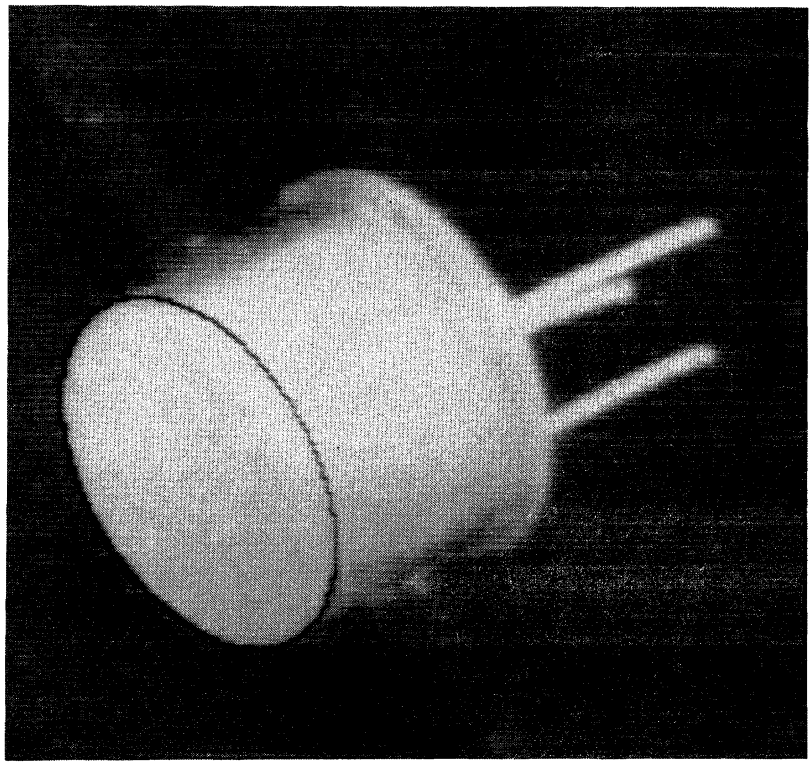


Figure 4.12: The ellipse which was fit using selected boundary points is shown on the original transistor image. Tilt = 60°, rotation = 150°

possible to identify this region and use those points as input to the ellipse fitting algorithm, a more accurate fit would result.

Identifying the region of interest is a difficult task. One possible tool for accomplishing this task is an "edge operator", such as the one suggested by I. Sobel [Duda 1973]. Such an operator is used to pick out the image areas which exhibit high intensity gradients. The results of applying the Sobel edge operator to the original transistor image can be seen in figure 4.13. Figure 4.13 clearly illustrates the image highlight which denotes the edge between the top and the side of the transistor. The boundary of the transistor is also outlined, as well as a number of other interior points. To determine candidate ellipse points, the "edge image" obtained using the Sobel operator is thresholded using the method of [Schlag 1983] and then the ellipse of figure 4.12 is superimposed on the resulting image. The points in the edge image which are near to to the ellipse represent image features which correspond to the ellipse in the scene. These points arise both from the curved portion of the

transistor boundary and from the brightly lit corner which marks the the opposite side of the ellipse in the image. All of these points should be used for ellipse fitting.

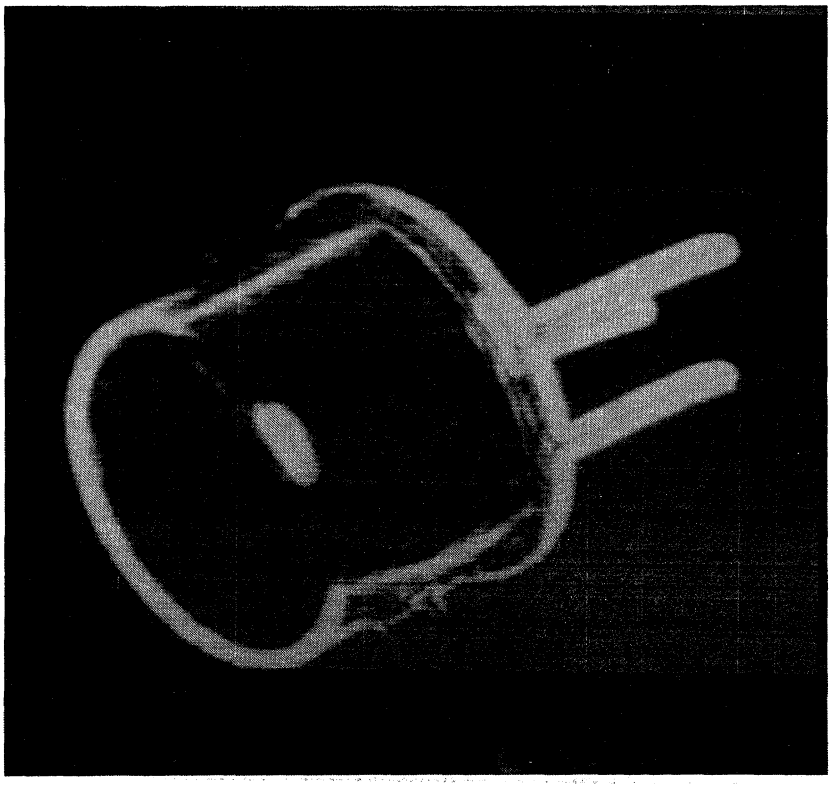


Figure 4.13: Edge image of original transistor obtained using the Sobel edge operator.

To collect the data for all the points of interest, the ellipse which was fit using only boundary information is expanded and then repositioned on the binarized edge image. All of the edge image points which lie inside this expanded ellipse can then be used as input to the ellipse fitter. Figure 4.14 depicts the superposition of the original ellipse, the expanded ellipse, and the gradient image. Figure 4.15 illustrates the improved ellipse fit obtained using this method. The minor axis of the improved ellipse is only 2% larger than it should be to accurately describe the transistor top. This can be compared to the 19% error of the first ellipse, which is also shown in figure 4.15.

4.3.7 Orientation From Ellipses

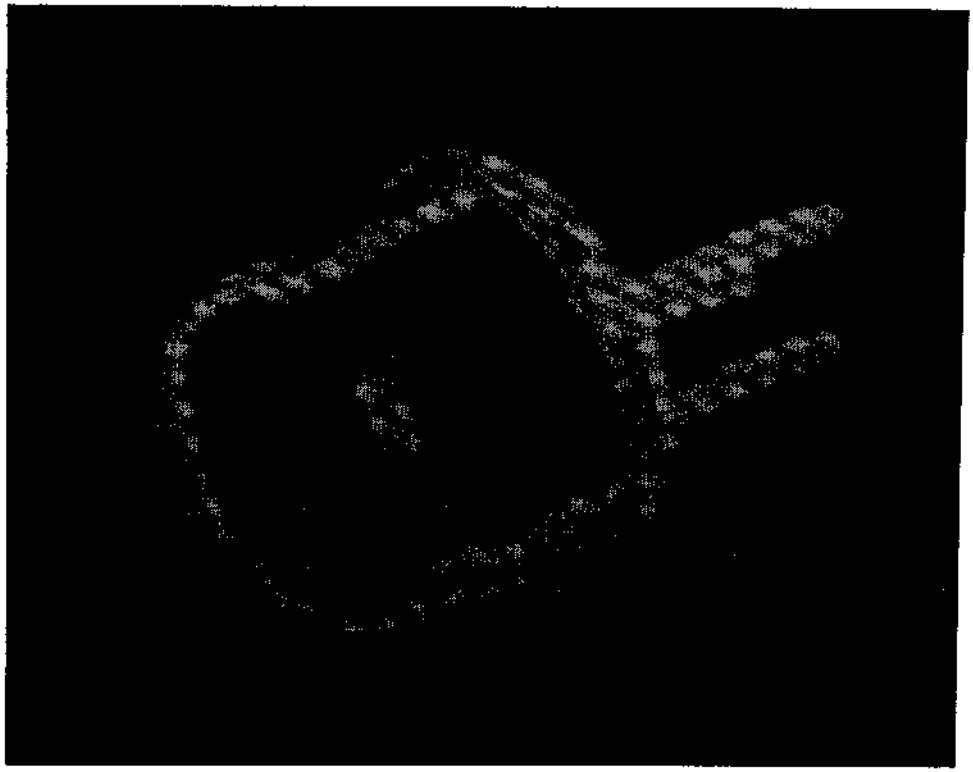


Figure 4,14: The ellipse of figure 4.12 is expanded and then superimposed on the binarized edge image. The original ellipse of figure 4.12 is also shown for comparison.

Knowledge of which ellipse parameters best fit the image data does not intrinsically allow the determination of orientation. Something must be known about the object to which the ellipse is fit before any useful orientation information may be extracted.

For the transistor, the image ellipse is generated by the perspective view of the circular transistor top. The relative lengths of the major and minor axes will then determine the tilt of the transistor according to the following formula:

$$\text{Tilt} = \arccos \left(\frac{b}{a} \right) .$$

The rotation of the transistor corresponds to the orientation of the minor axis. The estimate of rotation will be ambiguous ±180° because any ellipse rotated by 180° is equivalent to the original ellipse. In terms of the standard ellipse parameters,

$$\text{Rotation} = \theta + 90° \quad \pm 180° .$$

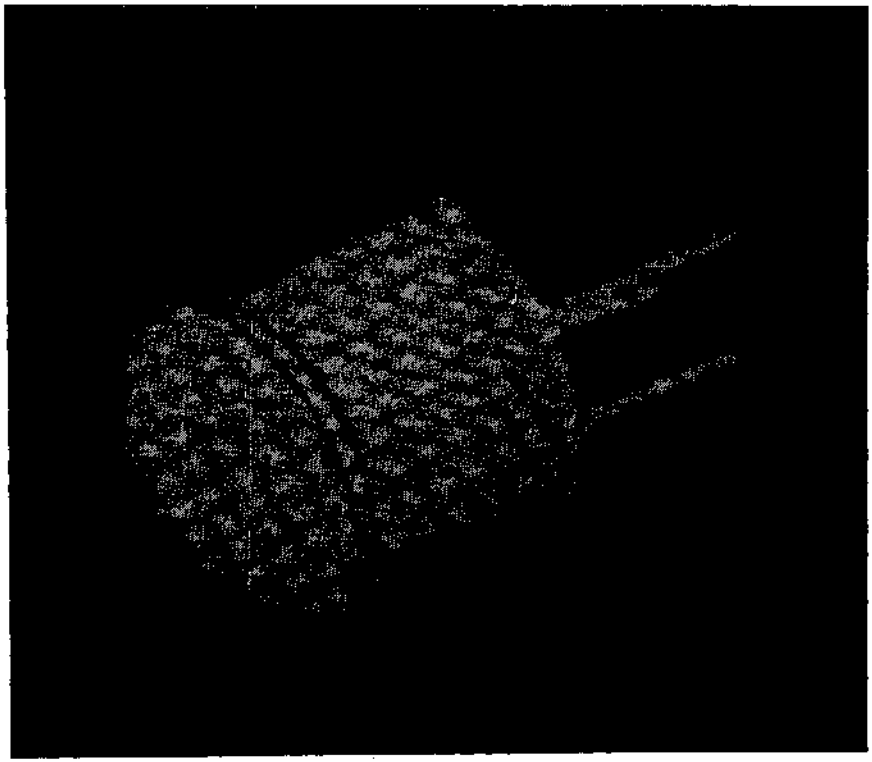


Figure 4J5: The ellipse fit using edge information is shown on the original transistor image. The larger, less accurate, ellipse of figure 4.12 is also shown for comparison.

The ellipse fitting algorithm mentioned in this chapter provides a solution in terms of the parameters A, *B,* C, D, E_y and *F.* For completeness, the equations for converting from these parameters to the standard ellipse parameters a, 6, /i,/c, and *6* are provided below:

$$h = \frac{BE - 2CD}{4AC - B^2} \qquad \text{and} \qquad k = \frac{2EA - BD}{B^2 - 4AC}$$

$$\theta = \frac{*}{2} \arctan\left(\frac{-B}{C - A}\right).$$

Using *h, k,* and *9* from above,

$$a = \sqrt{\frac{J(\sin^2\theta - \cos^2\theta)}{C\sin^2\theta - A\cos^2\theta}} \qquad \text{cind} \qquad b = \sqrt{\frac{J(\cos^2\theta - \sin^2\theta)}{C\cos^2\theta - A\sin^2\theta}},$$

where $J = h?A + hkB + k^2C - F$.

J_h3.8 Application Summery

The task of fitting ellipses to images has been described in this chapter as a two part process. First, there is the general question of how to fit an ellipse to a set of points in the coordinate plane. This problem has received much study, and a number of general purpose solutions exist. Secondly, there is the question of which image points should be selected as a group to represent an ellipse.

It is this second question for which no general sohition exists. This paper has considered the simplified image domain of object boundaries and determined that curvature methods can be used to identify curve segments which may correspond to elliptical edges. For the specific case of the transistor application, some simple *a priori* knowledge about the outlines of cylindrical objects was sufficient to uniquely identify the ellipse shaped segment of the object boundary. Such curve segments provide an ellipse solution which is a good first approximation to the original image data, but one which could use some improvement.

The ellipse solution obtained using boundary methods was then used as a guide for selecting candidate ellipse points from the more complicated domain of intensity gradients. The extra information extracted from the gradient domain was found to provide a more accurate ellipse fit for the sample transistor image. The parameters of the ellipse were then converted to an estimate of orientation using the knowledge that the elliptical region of the image was generated by the perspective transformation of a circle.

4.4 Results of Ellipse fitting

In order to objectively analyze the methods of the previous sections, these methods were applied to 138 different images of transistors. The orientation of each transistor was measured using standard mechanical techniques. Each image v/as then labelled with an estimate of orientation derived from the methods put forth in this chapter. The corresponding performance graphs are shown in figures 4.16, 4.17 and 4.18. Each graph shows the actual orientation versus the estimated orientation.

Figure 4.16 illustrates the estimates of tilt obtained using *only* boundary curvature to select ellipse points. The errors evident in this graph arise from (at least) three different independent error mechanisms.

- The smoothing function used to smooth the curvature graph (see figure 4.10) tends to blur the sharp change in curvature between the transistor top and side. Because of this, the boundary region selected for ellipse fitting is always a few

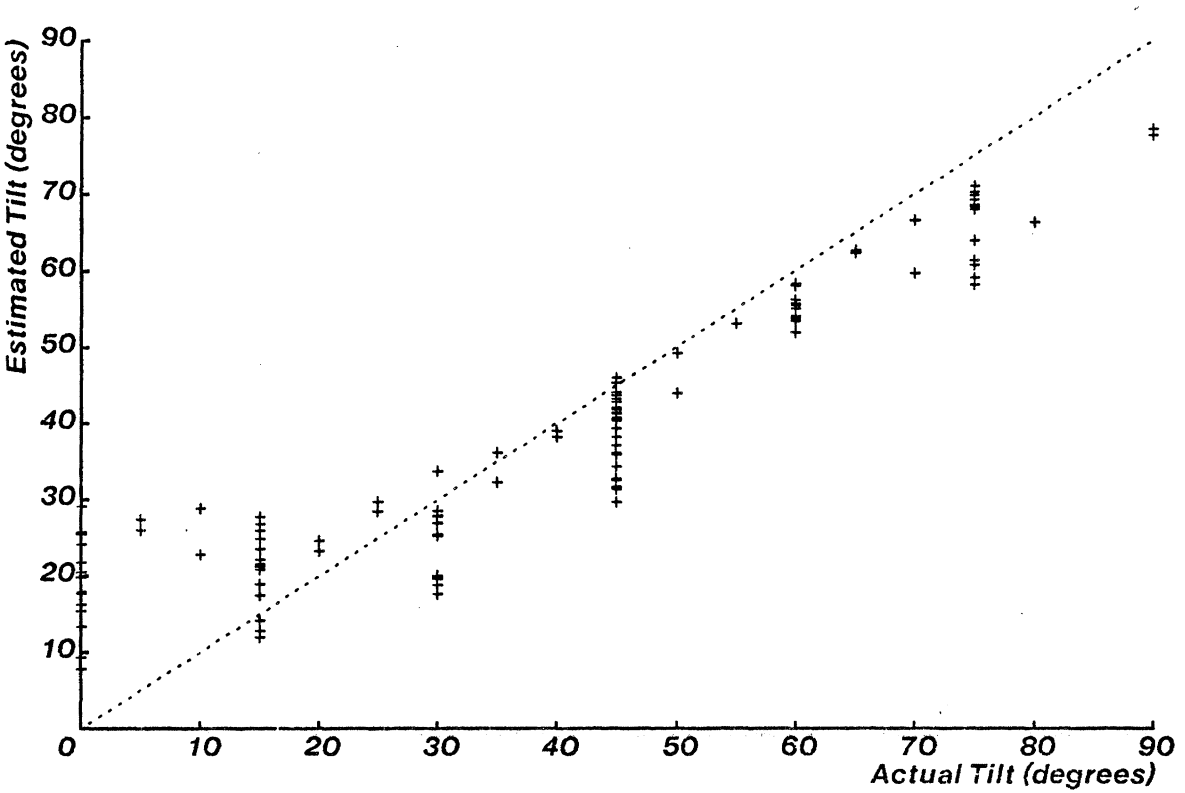


Figure 4.16: Actual transistor tilt vs. tilt estimated from ellipse parameters. Ellipse fit was obtained using *only* selected transistor boundary points.

pixels too long, resulting in low tilt values. This type of inaccuracy is most critical for high values of tilt and is the overriding cause of error for tilts greater than 30°.

- The estimation of tilt is very sensitive for tilt values near 0° (see section 4.3.7). At low tilts, a 2% error in the estimation of a or b introduces a 10° error in tilt value. For the images used in this research, a 2% error corresponds to about one pixel at tilt = 0°. This helps to explain the large errors at low tilts.

- As previously discussed, ellipse fitting based on minimization of the general conic (equation 4.2) tends to favor ellipses which are more eccentric than expected. Because of the two points mentioned above, these erroneously eccentric ellipses cause the most problems at low tilt angles. Figure 4.16 displays a clear tendency to overestimate tilt for values of tilt less than 30°.

Ellipse fitting does not always work as well as figure 4.16 might seem to indicate. At certain orientations (those with high tilt values), the elliptical shape of the

transistor boundary is nonexistent, or difficult to discern. In these cases, the curve fitting method of the previous sections decides that a hyperbola fits the data better than any ellipse. In practical terms, this indicates a total failure of the ellipse methods and such points are not shown in figures 4.16, 4.17 and 4.18. There are 16 of these disaster points, but they are confined to transistors with high tilt, as shown by the following table.

Tilt Angle (degrees)	Number of attempts	Number of Failures
90	14	12
85	2	2
80	2	1
75	14	1

It is useful to note that the heuristic used to segment transistor boundaries usually finds the boundary section corresponding to the top end of the transistor. This occurs because the outline of the bottom is broken by the outline of the legs. For low tilt angles, however, the legs are occluded by the transistor body and the ellipse is often fit to the bottom edge of the transistor. While this does not pose any problems for determining the orientation of the ellipse in figure 4.16, it is still important to be able to recognize which end of the transistor is the top.

The use of intensity gradient information as described in section 4.3.6 actually assumes that the transistor top has been located. If this assumption is invalid, then a very large error is reported by the ellipse fitting algorithm when the edge image is used. This large error is used to recognize the "wrong end" condition and corrective measures are applied. In principle, corrective measures might include reanalyzing the curvature graph and determining the top end correctly. In this research, corrective measures correspond to using the ellipse which was fit to the bottom boundary and then simply remembering that this ellipse corresponds to the bottom end.

Figure 4.17 depicts the performance for the tilt estimate obtained using information from the edge image; figure 4.18 depicts the performance for the rotation estimate using the same method. Notice that there is a 180° ambiguity in the rotation value since the ellipse parameters do not distinguish between the top and bottom ends of the transistor.

Comparing figure 4.17 with figure 4.16, it is clear that intensity gradient information significantly improves results. The errors that do exist can be explained by the following points

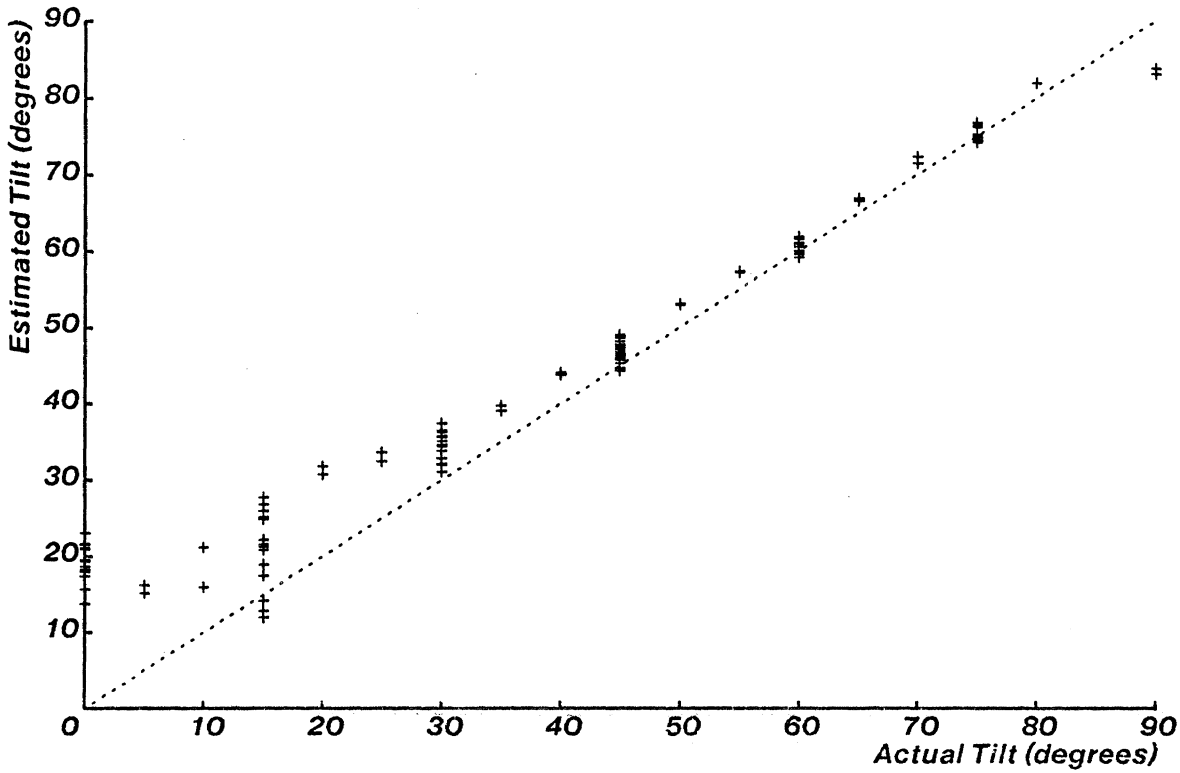


Figure 4.17: Actual transistor tilt vs. tilt estimated from ellipse parameters. Ellipse fit was obtained using intensity gradient information, as well as selected transistor boundary points.

- As for figure 4.16, there is an inherent tendency for all ellipses to be more eccentric than desired causing all tilt estimates to be on the high side. This error has a greater effect at low tilts because the required arccosine operation is very sensitive around 0°.

- At low tilts, especially around 15°, the ellipse is first fit to the bottom end of the transistor. When the edge data is used to improve the initial ellipse estimate, the "wrong end" condition is detected and no ellipse improvement occurs. The use of somewhat cruder ellipses results in large scatter of the tilt estimate at 15°.

- At tilt = 90° there is theoretically no ellipse in the image. This usually results in a failure of the ellipse method as mentioned earlier. In some cases, however, the transistor boundary is seen as *slightly* curved and an ellipse is fit. The tilt estimated from such an ellipse can never be the actual value of 90°, but must be some smaller value.

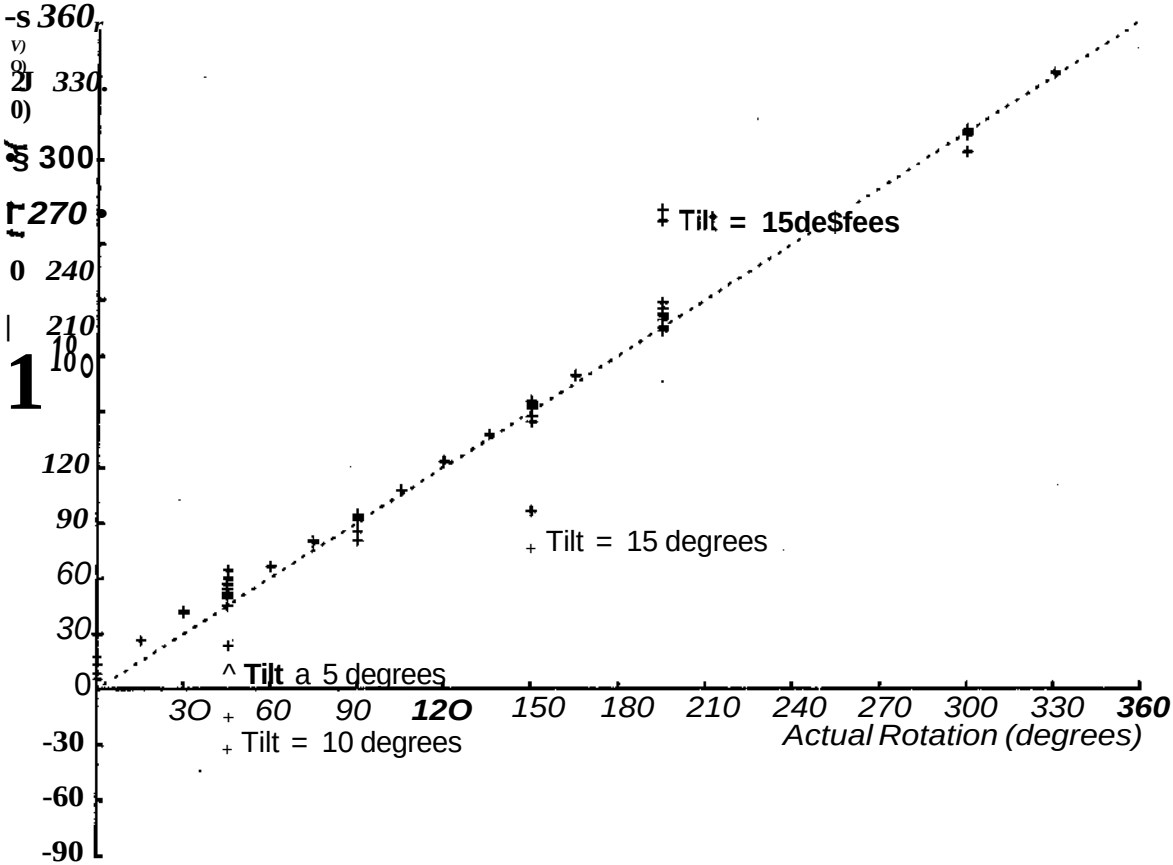


Figure 4,18: Actual transistor rotation vs. rotation estimated from ellipse parameters. Four pairs of outlying points are labelled with their corresponding tilt value. Estimated rotation values are ambiguous ±180°. Ellipse fit was obtained using intensity gradient information, as well as selected transistor boundary points.

The results of the ellipse rotation estimate are displayed in figure 4.18. 92.6% of the rotation estimates are within ±25° of the actual rotation values, but there are four distinct pairs of outlying points. Each pair of outlying points is labelled with the actual transistor tilt value for that pair, and it is clear that all of the outlying points were generated by transistors with very low tilt values. There are three major causes for this systematic error.

- Very low tilts correspond to almost circular ellipses which axe inherently degenerate in *6.*

- As previously discussed, low tilt angles often result in fitting an ellipse to the bottom end of the transistor which yields less accurate ellipse parameters.
- © The Sobel edge image generated by low tilt transistors is often more "cluttered" than the edge image of high tilt transistors. This increases the chance that improper edge points will be used to determine the ellipse.

It should be noted that figure 4.18 does not contain any points corresponding to failed ellipses or to those ellipses with an actual tilt of 0° because the concept of rotation is nonsensical in these cases. It is also worth noting that the points in. figure 4.18 have a ±180° ambiguity.

Chapter 5

Combining Orientation Information

5.1 Introduction

In previous chapters, three methods have been presented for determining the three-dimensional orientation of objects from a two-dimensional image. Each of the methods has its own weaknesses and conditions of applicability, but each has proven to be useful for determining orientation. Intuitively, it seems that the information from each of these methods should be mutually reinforcing, and that some combination of the methods should be able to provide a better estimate of orientation than any of the methods alone.

The concept of combining information sources has been generally recognized and is used in many well-known AI expert systems [Ballard 1976] [Buchanan 1969] [Davis 1977] [Brooks 1981] [McDermott 1982] [Pople 1982]. Unfortunately, such use of multiple independent knowledge sources in practical problem solving is often very application specific. Due to the diverse types of knowledge, no generally useful theory has been developed to combine different information concerning a specific task. There are techniques, however, which show promise for combining information about certain problem domains.

5.2 Representing and Combining Knowledge

5.2.1 *Knowledge Representation*

Information may be represented in a number of different formats, and often the complexity of real world information requires several levels of representation. For example, consider the image of a sphere. Information concerning the presence of the sphere in the image can simply be represented as a boolean value: True or False. On the other hand, information concerning the image intensity values of the sphere could be represented as a surface in a real-valued three dimensional space or by an equation which describes this surface.

Using the definitions of [Ballard 1982], the knowledge necessary for computer vision tasks can be divided into three categories of representation. Each of the three representations are illustrated in the sphere example and will be further described here.

- *Analogical* representations are used to model real-valued "analog" phenomena, such as the three dimensional description of an image's surface. Analogical representations are continuous and often related to the structure of the represented situation. Most importantly, analogical information is typically manipulated by complex computational procedures.
- *Propositional* representations contain knowledge about high-level concepts such as whether there is a sphere present in an image. Propositional models are usually discrete abstractions. Propositional representations take on values like True or False and are manipulated using some type of inference engine, such as predicate calculus.
- *Procedural* representations store information about "how-to" perform some complex activity. This information is stored as a sequence of program steps or "procedures". A procedure, for example, might explain *how* to use a formula which represents the surface of a sphere. There is some question as to whether procedural information is really just a type of analogical or propositional information, but that need not be of concern here.

The ideal knowledge representation depends heavily upon the problem domain and the solution desired. Some problems are inherently analogical while others are propositional; many problems contain both analogical and propositional information. The ideal representation also provides for ease of combining or comparing information.

5.2.2 *Knowledge Combination*

The solution for many problems requires the combination of information from diverse knowledge sources. Due to the complexity of real world problems, a number

of different techniques for combining information have been proposed. Some of these techniques will be illustrated in this section.

Pattern recognition techniques are well-known methods for combining purely analog information [Duda 1973] [Fukunaga 1972]. Here, the information from the scene is represented in terms of "features" which are usually modeled as gaussian probability distribution functions. Analysis of the features using Bayesian decision theory yields a solution based on minimum probability of error. Such techniques, however, require that all domain knowledge be represented in a common feature space. This is an inadequate description for many high-level tasks.

Production systems l^ave been used to draw inferences from propositional knowledge [Davis 1977] [McDermott 1982]. A production system consists of a number of inference "rules" which specify some conclusion which can be inferred given that some number of conditions axe satisfied. A simple example of such a rule might be: "IF a section of the image is green THEN that section is grass". This type of rule is useful when the required high-level propositional information has been extracted, but is impractical for low-level analog processing.

It is important to note that information may interact in subtle and complex ways. Often, one piece of information is used to help deduce another. This interdependence must be represented as part of the knowledge about the problem. There must also be provisions for uncertainty or error* If an erroneous piece of data is used to derive a proposition which is, in turn, used to make higher level deductions, serious errors could result. Most importantly, the goal of most problems is to determine the globally best solution. This requires that there is some way to prevent local errors from propagating throughout the information network.

To account for these complex issues, practical systems usually use some combination of analogical and propositional techniques. A classical example is the MYCIN system [Buchanan 1984]. MYCIN is an expert system that tries to diagnose the cause of bacterial blood infections. MYCIN employs production rules to represent "expert knowledge" about blood bacteria, but *certainty factors* — ranging from zero to one — are attached to each piece of information and each production rule. These analog certainty factors are necessary because a correct diagnosis must be made in a domain where deductions are not certain.

The exact application of the certainty factors depends on the type of production rule involved.

- *AND* rules require that a number of conditions *(all)* be satisfied before the corresponding conclusion can be reached. For these rules, the certainty of the conclusion is computed by multiplying the smallest certainty factor of the conditions by the certainty of the rule itself.

- *OR* rules require that (*only*) one, or more, condition need be satisfied before the corresponding conclusion can be reached. For OR rules, the certainty of the conclusion is equal to the certainty of the production rule multiplied by C_t, where

$$C_t = C_1 + C_2 + \cdots + C_n - C_1 C_2 \cdots C_n ,$$

and C_i is the certainty of the i^{th} satisfied condition for $i = 1, \ldots, n$.

MYCIN, therefore, operates on analogic and propositional information in parallel. Data from a doctor is used by the production system to deduce ever higher levels of propositional abstraction while analogic certainty measures are used to weed out erroneous propositions and find the globally "best" final conclusion. This flow of information is depicted in figure 5.1.

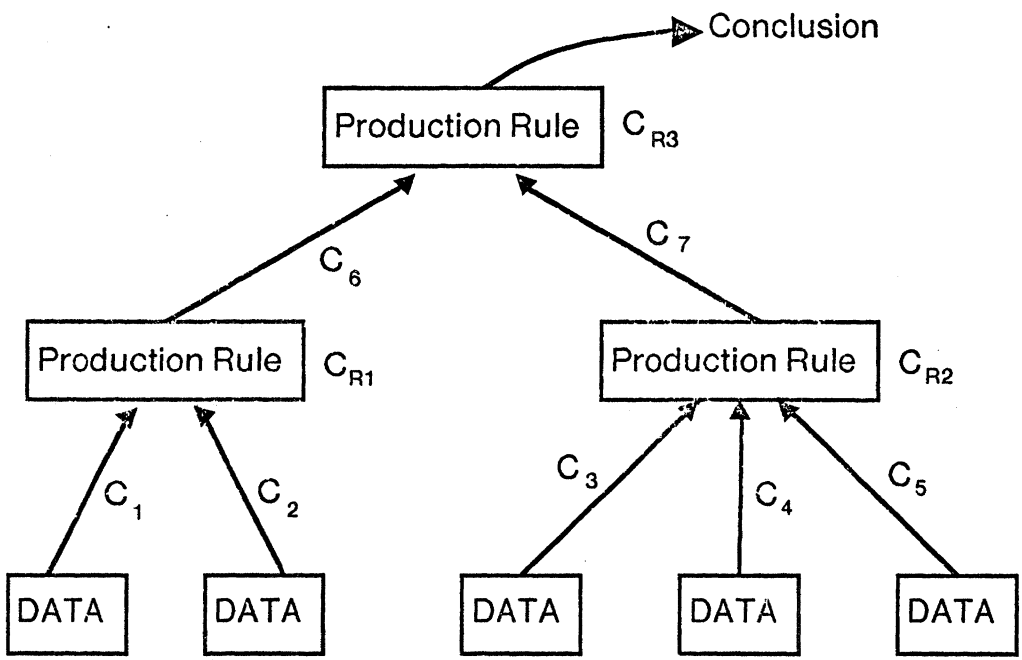


Figure 5.1: Information flow in the MYCIN system. C_i represents the certainty factor associated with a piece of information, and C_{Ri} is the certainty corresponding to a given production rule.

It should be noted that this method of combining information can not be called optimal. No theoretically optimal set of rules and certainty factors is available because the problem is not fully understood even by the best humans. The basic structure of the productions and the corresponding certainties are based upon

the subjective knowledge of experts. Such a heuristic approach is typical of most systems which attempt to solve real world problems at a high level of abstraction.

5.2.3 Plausibility Functions

The problem of combining multiple sources of unreliable information raises fundamental issues in estimation theory. When accurate probability models of observations are available, there exist several strategies — including maximum likelihood and maximum *a posteriori* probability — to optimally estimate underlying model parameters. In practice, such models may themselves be unreliable, and unreliable observation of any one measurement may be catastrophic for the estimation process. In many cases, one would, perhaps, prefer a less optimal estimate in exchange for a more graceful mode of failure in the face of unreliable models. In this study, the use of *plausibility functions* is introduced as an estimation technique which weights observation likelihoods according to the reliability of the underlying probability models. The resulting expressions are weighted averages of possible likelihood estimates and offer sub-optimal estimates in any given regime, but are robust with respect to failure of the underlying models.

As an example, consider a robot with N different range sensors, S_i for $i = 1, \ldots, N$. The robot is a distance d from an obstacle, and each sensor S_i yields a distance measurement x_i. The distance estimated by a sensor will vary about the true distance d depending on the surface properties and geometry of the obstacle. Assuming that a *reliable* probabilistic model can be obtained for sensor S_i, the measurement x_i will have a known probability density function $p(x_i|d)$.

Simultaneous observation of n reliable sensors is described by the joint conditional pdf $p(x_1, \ldots, x_n|d)$. If all sensor models are reliable, and if the reliable observations are also statistically independent, the joint pdf of the x_i given d is

$$p(\overline{x}|d) = p(x_1, \ldots, x_N|d) = \prod_{i=1}^{N} p(x_i|d) \,.$$

The maximum likelihood estimate, $\hat{d}_{ML}$, is that d for which $p(\overline{x}|d)$ is a maximum:

$$\hat{d}_{ML} = \max_d \left[\prod_{i=1}^{N} p(x_i|d) \right] \,.$$

If some *a priori* information $p(d)$ is available, Bayes Rule,

$$p(d|\overline{x}) = \frac{p(\overline{x}|d)p(d)}{p(\overline{x})} \,,$$

can be employed to deduce the *a posteriori* estimate of *d.* The maximum *a posteriori* estimate, $\hat{d}_{MAP}$ is that *d* for which $p(d|\vec{x})$ is a maximum:

$$\hat{d}_{MAP} = \max_{d} \left[p(d) \prod_{i=1}^{N} p(x_i|d) \right] .$$

Given reliable sensor models, $\hat{C}_{IMAP}$ is determined by finding the maximum value of the function resulting from the multiplication of the sensor probability density functions with the *a priori* pdf of *d.* Unfortunately, this scheme does not degrade gracefully in the presence of unreliable models. If just one of the robot's distance sensors were to fail in such a way that the sensor reported $p(x_i|d) = 0$ for $X\{ == (i,$ then the maximum likelihood estimate $\hat{C}_{IML}$ would never be accurate. This poor result would occur even if all of the other N — 1 sensors provided perfectly correct estimates of *d.*

The pdf of a reliable model must, therefore, indicate that such serious errors may occur. In practice, the determination of a pdf which reliably deals with all possible failure modes may not be feasible. This leads to the use of probability-like density functions and, therefore, sub-optimal combination rules.

One possible method for the combination of these unreliable pdfs is suggested by analyzing the failure mode of the classical probability approach. The maximum likelihood estimate breaks down in the presence of serious error because it is an intersection operation. In other words, *all* of the sensors must be accurately modeled by their probability density functions. In the case of unreliable estimates, it may be useful to define a sub-optimal, but more robust, *plausibility function* which is a linear combination of sub-optimal likelihood estimates:

$$\begin{aligned} pl\{x_u \ldots, x_N|d) \quad = & \sum_{i=1}^{N} W_i \, p(x_i|d) \\ & + \sum_{\substack{i,j \\ j<i}} W_{ij} \, p(x_i, x_j|d) \\ & + \sum_{\substack{i,j,k \\ k<j<i}} W_{ijk} \, p(x_i, x_j, x_k|d) \\ & \vdots \\ & + W_{1 \ldots N} \, p(x_1, \ldots, x_N|d) . \end{aligned}$$

Each possible joint pdf is multiplied by some weighting factor $W\backslash$ which is known as a *plausibility factor,* and then summed to obtain the plausibility function. The

resulting most plausible estimate is obtained as:

$$\hat{d}_{MPL} = \max_{d} [pl(x_1, \ldots, x_N|d)] \ .$$

The estimate provided by the equation above reduces to the maximum likelihood estimate when all observations are reliable and all of the plausibility factors are equal to zero, except $W_{1\ldots N} = 1$. The plausibility function incorporates joint probability densities, which are most useful for estimation when they are reliable, but which are also the most sensitive to unreliable observations. In practice, these joint probability densities are the most difficult to estimate. It is of particular practical interest to examine the case where joint density models are unobtainable or independence assumptions are uncertain. In this case all joint probability models are regarded as unreliable. The resulting *single factor plausibility function*:

$$pl(x_1, \ldots, x_N|d) = \sum_{i=1}^{N} W_i p(x_i|d)$$

describes the average likelihood given mutually exclusive observation of x_i, for $i = 1, \ldots, N$.

The single factor plausibility function is closely related to a number of heuristic linear weighted probability schemes which have been previously proposed [Buchanan 1984]. The development presented here relates these approaches to more traditional estimation strategies and shows that maximum likelihood estimation may be regarded as a special case of the general plausibility function method. The single factor plausibility function is used as the basis for estimation in this study due to the difficulty in estimating joint densities and in determining joint plausibility factors which may be functions of the model parameter d.

5.3 Integrating Information About Transistor Orientation

5.3.1 Representing Transistor Orientation Information

In this paper, the task is to combine the estimates of tilt and rotation obtained from the methods of Chapters 2, 3, and 4, in order to obtain the most plausible single interpretation of transistor orientation. Before this can be accomplished, it is first necessary to establish a consistent representation for orientation information. The representation selected should capture the critical aspects of the problem domain while allowing easy comparison and manipulation of the information. An analogical representation seems the most natural to satisfy these constraints because

of the analog nature of orientation estimation — tilt and rotation are both naturally expressed as smoothly varying real-valued parameters. It will be useful to see what types of information are available from the orientation estimates of the previous chapters and how this information can be converted to an analog representation with the desired properties.

The binary connectivity analysis of chapter 3 provides two estimates of transistor rotation which are 180° apart. This information appears to be a poor choice for analog representation because the information simply consists of two numbers (only one of which is unique). A little thought, however, will show that the two estimates actually contain analog information when viewed properly. The rotation estimates represent *the most probable* estimate. Due to the effects of noise and measurement error, it is only slightly less probable that the actual transistor rotation is 1° above or below this estimate. In fact, the estimate of rotation represents the center of an analog probability density function. As discussed above, the shape of the pdf must be determined by *a priori* knowledge or empirical studies, but given such a function, it is possible to create an orientation representation which models the information quite well. For the purposes of this research, a gaussian density function is used. The two single-valued estimates of rotation obtained from binary connectivity aimlysis can now be transformed to a rotation probability density function by convolving each value with a gaussian filter. The tilt and rotation estimates from ellipse fitting can also expanded to probability functions through convolution with a gaussian density function.

The estimates from histogram template matching, however, represent a different form of information and must be handled differently. As discussed in chapter 2, both the center section and the highlight section (if any) of a transistor histogram already generate a probability measure for each 5° of tilt. The probability measure from the center histogram section corresponds to the degree of correlation between the input histogram and the current histogram from the training set. The probability measure generated by the highlight section of the histogram depends on the presence (or absence) of a highlight region in both the input histogram and the current histogram from the training set. In order to create an analog probability function from each of these discrete probability functions, linear interpolation is used.

All of the tilt and rotation estimates discussed in the preceding chapters have now been represented in a uniform analog format. Each estimate has been transformed in a way which attempts to model the real world considerations of noise and error while preserving the underlying structure of the knowledge. Examples of these functions will be illustrated in the next section.

5.3.2 Combining Transistor Orientation Information

The rotation estimates obtained from binary connectivity analysis and ellipse fitting both suffer from a $\pm 180°$ ambiguity. In order to alleviate this problem, another (unambiguous) estimator of rotation is used. The orientation of a line drawn between the center of the binary blob and the center of the ellipse is a unique estimator for the rotation of the transistor. This estimate is convolved with a gaussian density function to create a probability function.

The results for such an estimator are displayed in figure 5.2. The graph exhibits generally linear behavior with a few outlying points. The three points which exhibit the greatest error have been labelled with their appropriate tilt values. The pair of points with a tilt of 5° display the greatest error because the rotation estimator is inherently degenerate for tilts approaching 0°. This degeneracy is responsible for most points which display significant error. There is, however, one outlying point with a tilt of 80°. In this case, the ellipse which was fit to the data had an approximately correct length to width ratio, but it was twice the size of the desired ellipse. It is more difficult to fit ellipses to transistors with high tilt because the ellipse becomes non-existent at 90° and the corresponding edge image is quite "cluttered".

With this new rotation estimate, there are a total of six orientation estimators: three estimates of transistor tilt and three estimates of rotation. To create a single estimate of tilt (rotation), some method for combining the three tilt (rotation) estimates is necessary. As discussed in the preceding section, one combination method involves scaling each pdf by an appropriate plausibility factor and then summing the resulting plausibility functions. The final orientation estimate corresponds to the the most plausible point of the total plausibility function.

The plausibility factors are intended to provide a "fair" balance between the individual estimates. The plausibility factors specify the relative importance of the given estimate to the final conclusion, and are closely related to the certainty factors of MYCIN. As in MYCIN, the values of these scaling factors must be selected on the basis of experience. The scaling factor for the orientation estimates from ellipse fitting can also be modified by the error of the ellipse fit: the higher the error, the smaller the scaling factor.

Figures 5.3 through 5.10 illustrate the entire process of representing and combining information for a transistor with tilt = 60° and rotation = 150°. Figures 5.3, 5.7, 5.8 and 5.9 show the result of convolving a gaussian density function with one, or two, single-valued orientation estimates and then scaling the result by a plausibility factor to obtain the appropriate plausibility function. Figures 5.4 and 5.5 were obtained by applying linear interpolation to the two discrete probability functions provided by histogram template matching and then scaling the result. Figure 5.6 depicts the result of summing the tilt plausibility functions, and figure 5.10 repre-

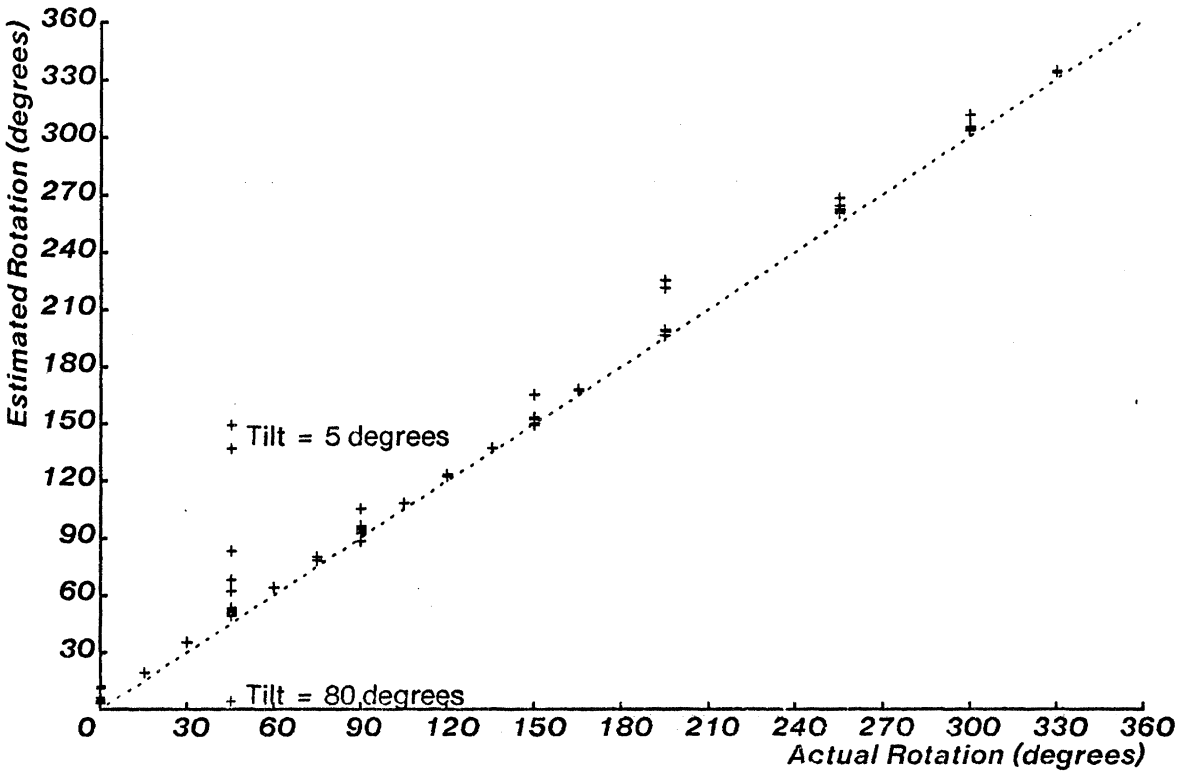


Figure 5.2: Actual transistor rotation vs. unambiguous rotation estimated from the orientation of a line drawn between the center of binary blob and center of ellipse. The graph does not contain points for which the transistor tilt was 0°, or for which ellipse fitting failed.

sents the summation of the rotation plausibilities. The most plausible orientation value from each of these plots is selected as the orientation of the transistor. In this case, tilt = 59° and rotation = 154°.

The method of summed plausibility functions illustrated by figures 5.3 through 5.10 has a number of useful properties. First, the representation used fits in well with the problem domain and allows for easy combination of information. Second, there is no need to attach separate certainty factors to each estimate of orientation; the certainty of any estimate can be included as part of the plausibility representation through a scaling factor. Third, this method allows all the information to come together in a single summing operation thereby providing a globally most plausible solution. This is possible because there is no need to deduce an orientation estimate from information gained using another method. These important points are summarized in figure 5.11 and may be compared to the MYCIN system (figure 5.1).

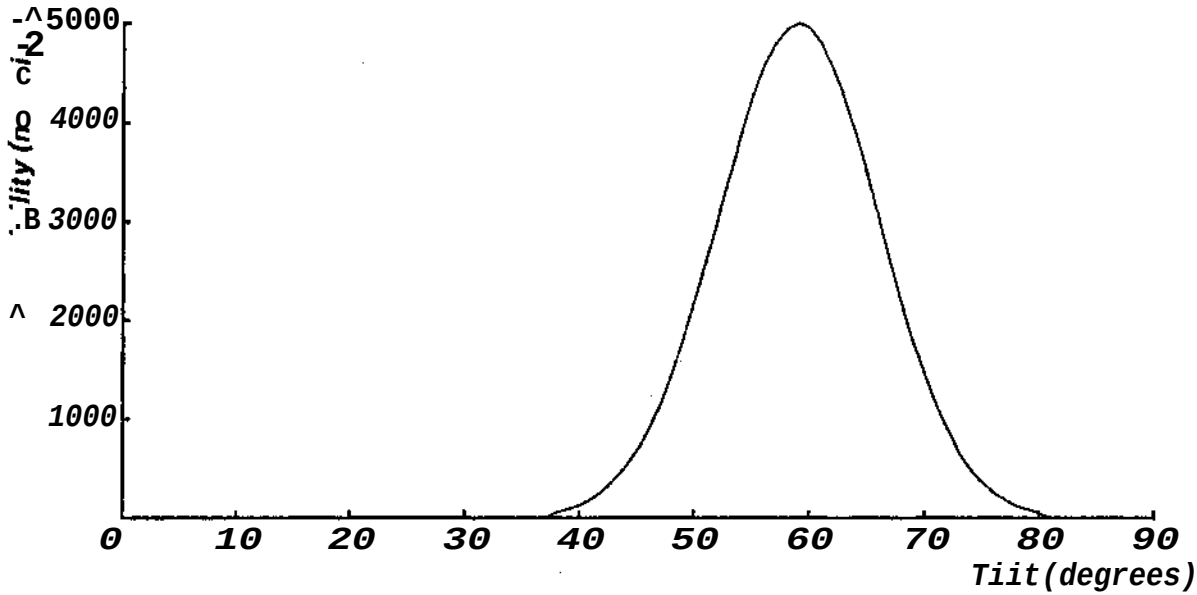


Figure 5.3: Tilt plausibility function obtained using ellipse fitting. Maximum plausibility occurs at tilt of 59.3°. The actual transistor tilt is 60°.

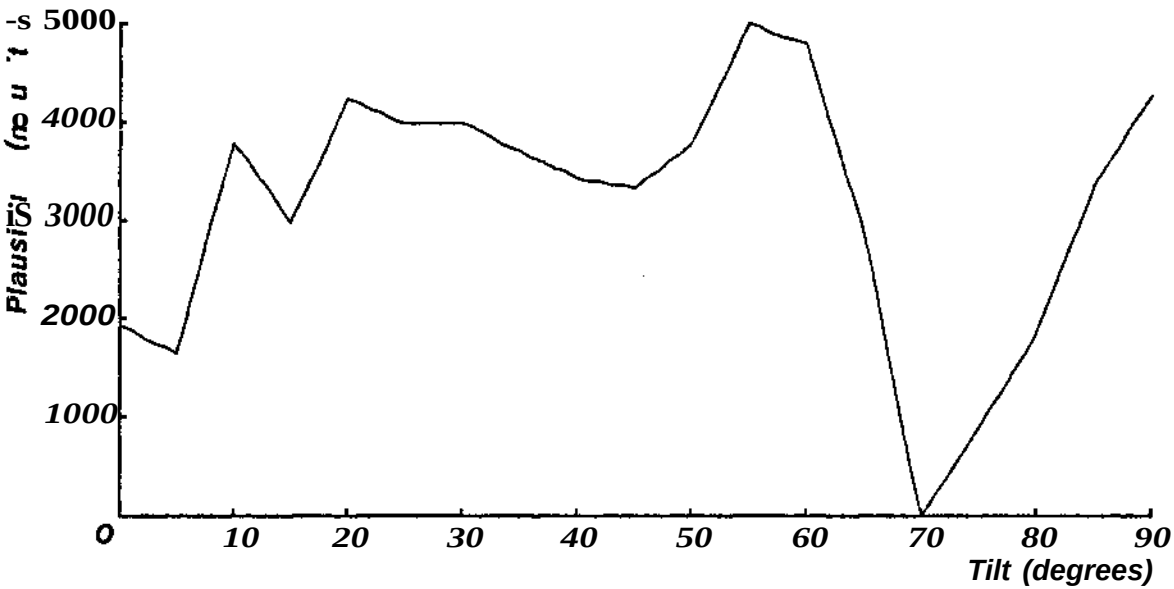


Figure 5.4: Tilt plausibility function obtained from histogram template matching. Maximum plausibility occurs at tilt of 55°. The actual transistor tilt is 60°.

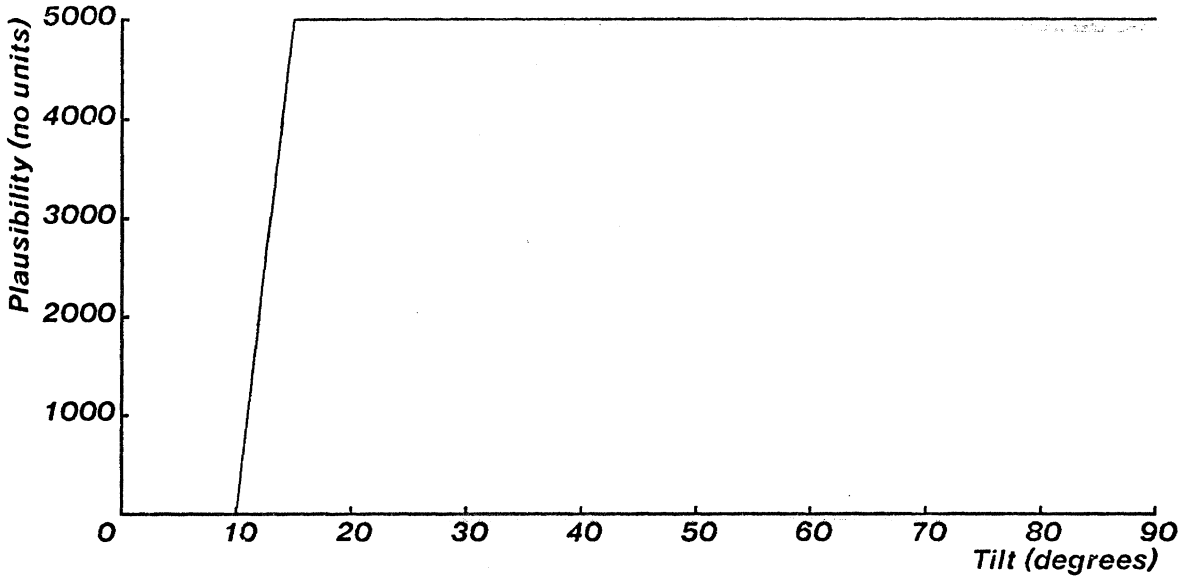


Figure 5.5: Tilt plausibility function obtained from histogram highlight matching. Maximum plausibility occurs for all values of tilt between 15° and 90°. The actual tilt is 60°.

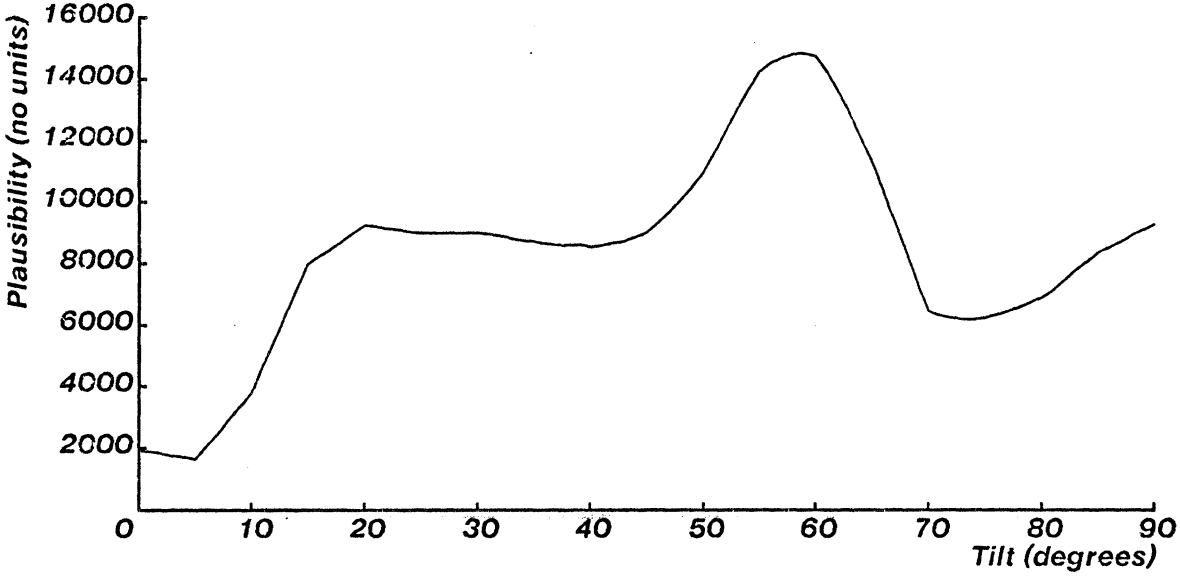


Figure 5.6: Tilt plausibility function obtained by summing the three previous plausibility graphs. Maximum plausibility occurs at tilt of 59°. The actual tilt is 60°.

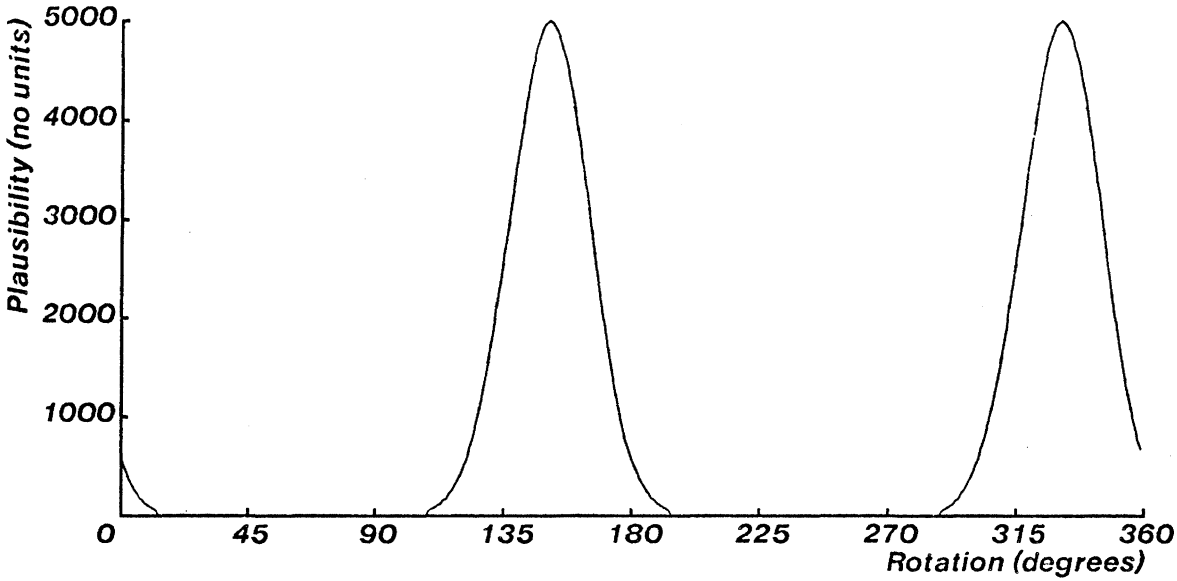


Figure 5.7: Rotation plausibility function obtained from ellipse fitting. Maximum plausibility is at a rotation of 151° or 331°. The actual tilt is 150°.

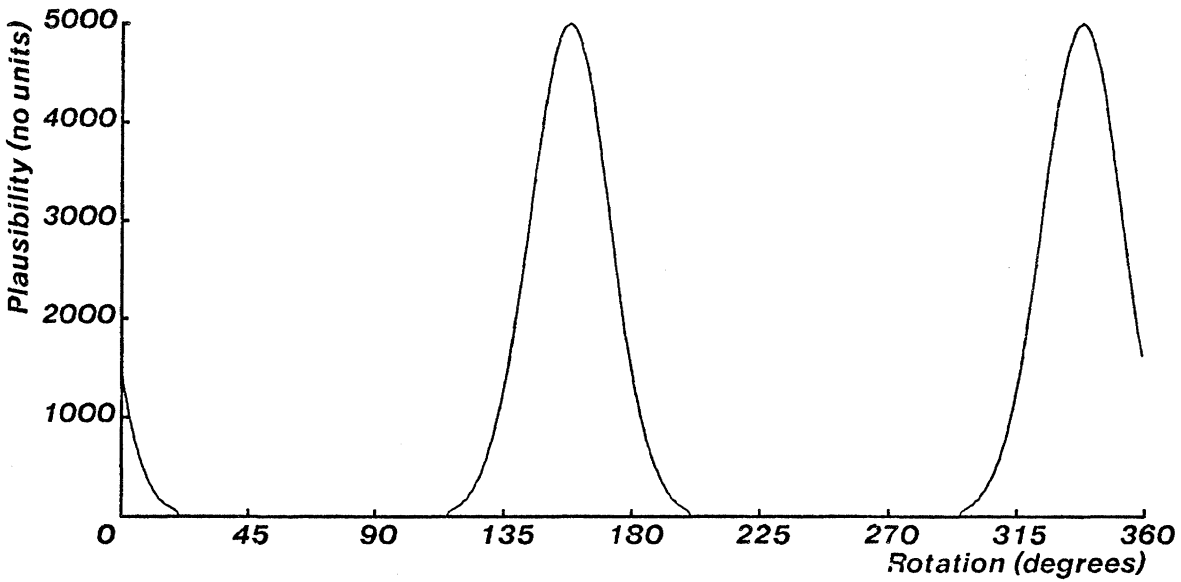


Figure 5.8: Rotation plausibility function from binary connectivity analysis. Maximum plausibility occurs at a rotation of 159° or 339°.

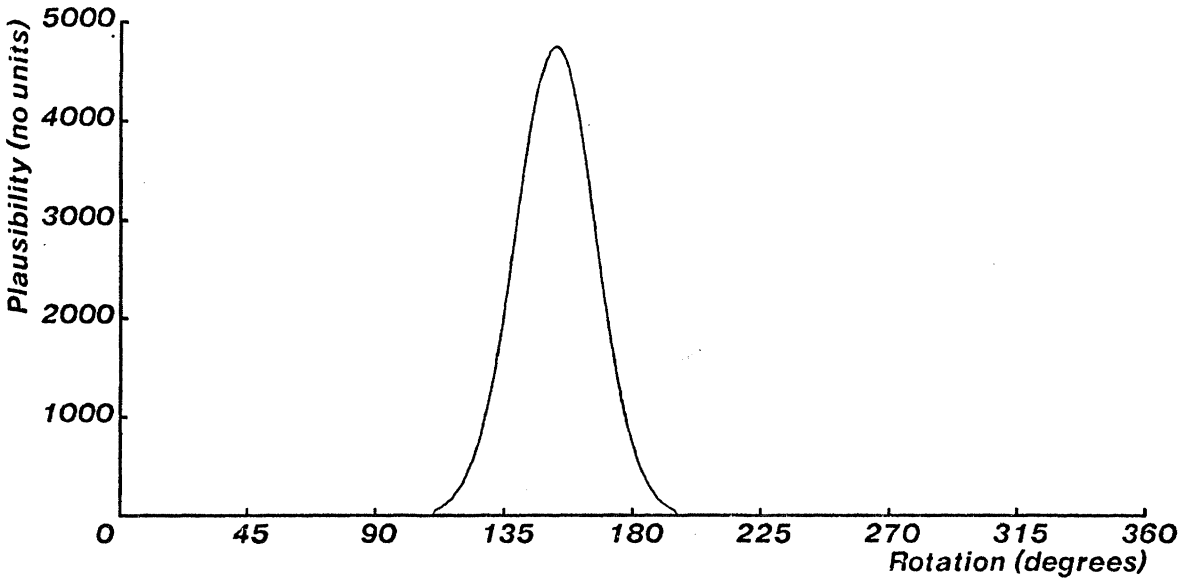


Figure 5.9: Rotation plausibility function obtained from unambiguous rotation estimator. Maximum plausibility occurs at a rotation of 152°. The actual transistor rotation is 150°.

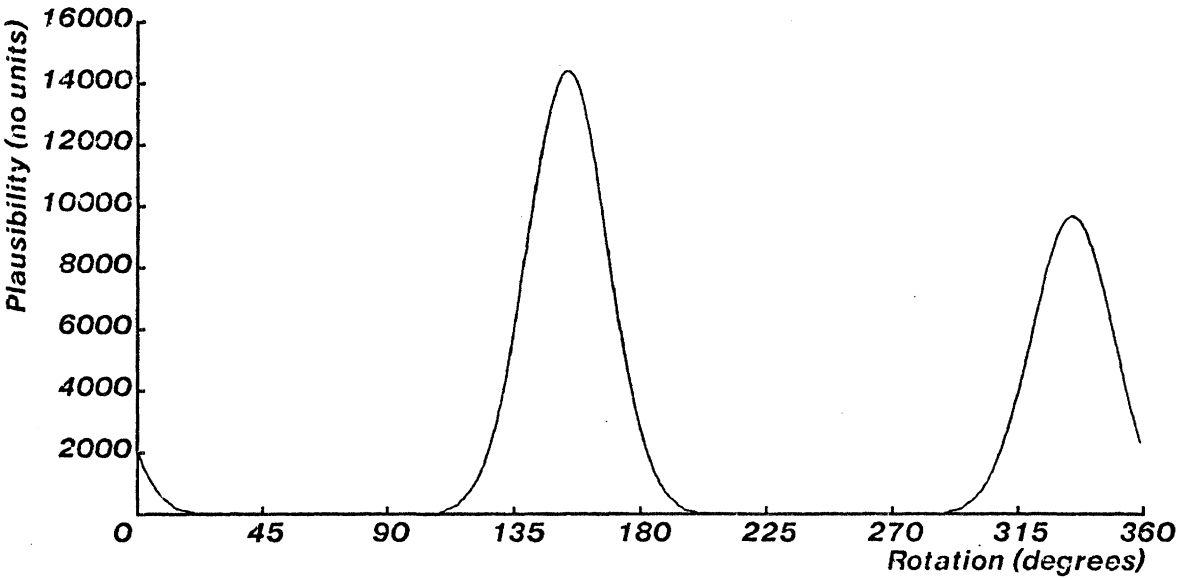


Figure 5.10: Summation of the three previous plausibility graphs. Maximum plausibility occurs at a rotation value of 154°. Actual rotation is 150°.

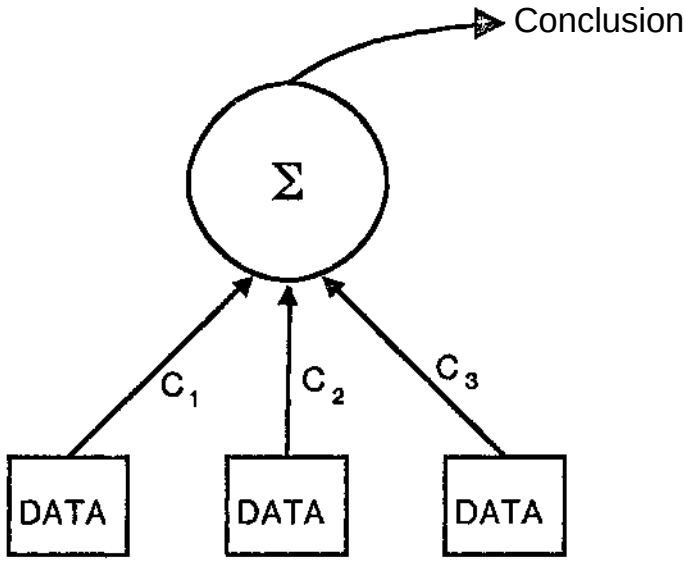


Figure 5.11: Information flow in the transistor system. The C_i are scaling factors analogous to MYCIN's certainty factors.

5.3.3 Unusual Ellipse Situations

A number of unusual situations may arise which affect the ellipse fit and the subsequent plausibility function. These situations and their effects axe addressed here.

If the ellipse fitting algorithm fails (i.e. the data fits best to some conic other than an ellipse), the relevant probability density functions are generated as a special case. For the rotation estimate, every angle is equally probable, so a flat probability function is generated. For tilt, it is most probable that the actual tilt is close to 90° since this is where the ellipse usually fails (as explained in section 4.4). These synthesized probability functions are then multiplied by very small plausibility factors to indicate a low degree of confidence. Finally, the unambiguous rotation estimator described at the beginning of the previous section requires the presence of an ellipse to function properly. When the ellipse is non-existent, the estimator measures the orientation of a line drawn between the center of the binary blob and the center of the points to which the ellipse fit was attempted.

If the ellipse is accidentally fit to the bottom end of the transistor, the tilt and rotation estimates from the ellipse will be unchanged. The unambiguous rotation estimate, however, will be off by 180°. In this case, 180° is added to correct the estimate. The plausibility factor for the ellipse orientation estimates is also reduced somewhat because ellipses fit to the transistor bottom are inherently less accurate,

as mentioned in chapter 4.

5.4 Results

To test the performance of the total system, the system was used to estimate transistor orientation for a number of transistor images. Forty-nine transistor images were used, each image corresponding to a different orientation. The 49 different orientations used were determined by trying all possible permutations of 7 rotations (0°, 45°, 90°, 150°, 195°, 255°, 300°) and 7 tilts (0°, 15°, 30°, 45°, 60°, 75°, 90°). The resulting estimates of tilt and rotation are displayed in figure 5.12 and figure 5.13, respectively.

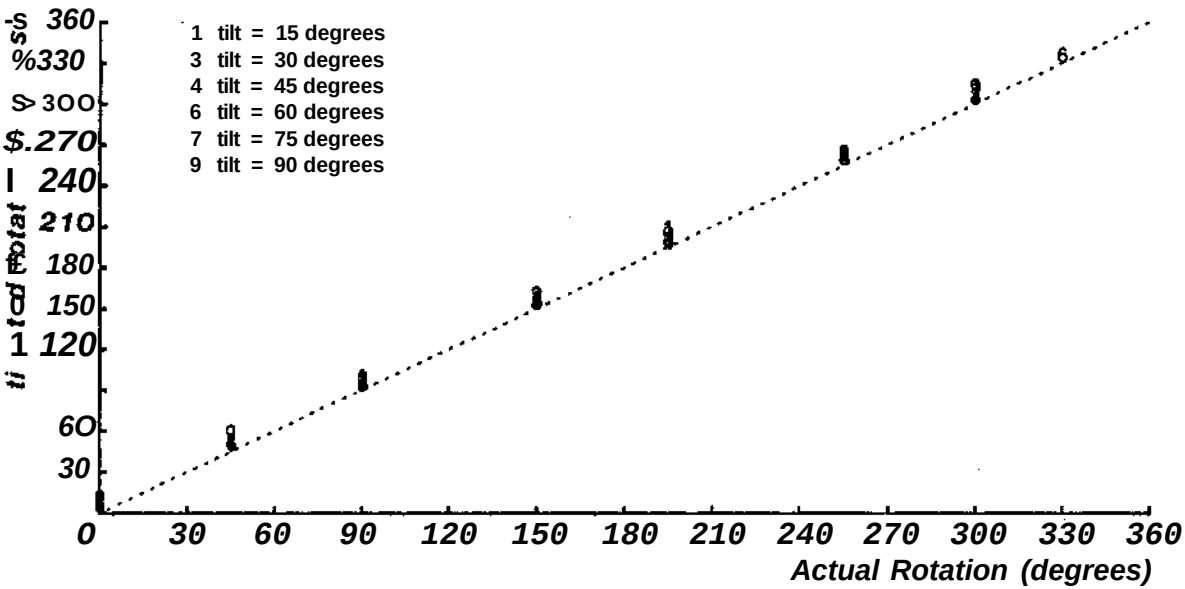


Figure 5.12s Actual vs. estimated rotation from combined system. Data is labelled by tilt for 49 images under normal illumination.

Figure 5.12 exhibits the desired straight line trend between actual and estimated orientation. On the average, there is a tendency for the rotation estimate to be a few degrees too high. This systematic error is caused by the position of the transistor legs as discussed in chapter 3. Including these errors, every rotation estimate is still within *4%* (14°) of the actual rotation. Two minor points to notice are:

1. This graph does not include any data points which correspond to tilt = 0° because such points have no meaningful rotation value.

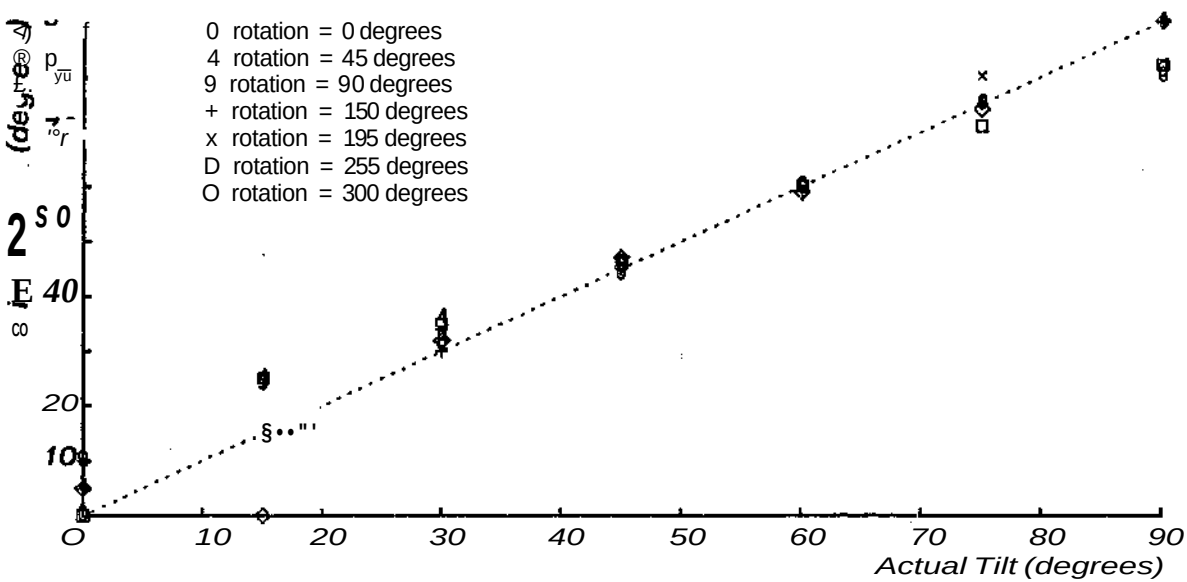


Figure 5.13: Actual transistor tilt vs. estimated tilt from combined system. Input data is labelled by rotation for 49 different transistor images under normal illumination.

2. The single data point at tilt = 60°, rotation = 330° is not one of the possible permutations mentioned above. This data point was accidentally included instead of the point at tilt = 60°, rotation = 300°.

The tilt estimates (figure 5.13) exhibit a larger percentage error than the rotation estimates. With the exception of the single data point corresponding to tilt = 15° and rotation == 330°, every tilt estimate is within 11% (10°) of the actual value. The various sources of error have been explained in the relevant sections of the previous chapters, however, some additional explanation is warranted for the outlying point at **tilt** = 15° and rotation == 330°.

This point had a tilt of 24.6° according to the best fit ellipse. Its histogram correlated best with a tilt of 60°, but 20° and 0° correlated well also. The real problem occurred with the highlight portion of the histogram; according to the histogram highlight this data point should have a tilt of 0°, 5° or, 10°. This incorrect information could not be corrected by the other tilt estimates because they were not very accurate either.

It is important to note that the histogram tilt estimators play a more important role in estimating small tilts. Referring to figure 4.17, it is clear that the ellipse tilt estimate suffers from systematic errors at low angles and that these errors must be overcome by the histogram estimators. A quick glance at figure 5.13 suggests

that the histogram tilt estimators were indeed able to reduce these errors. Figure 2.5 further suggests that much of the error reduction was supplied by the highlight portion of the histograms.

Using the same 49 transistor orientations, a set of low intensity images was collected to test the system's sensitivity to changes in illumination intensity. These darker images were actually obtained under the same lighting conditions as the first set; the lens aperture was just closed two f-stops. The result of using these darker images can be seen in figures 5.14 and 5.15.

Figure 5.14 is almost identical to figure 5.12 and all of the same analysis is applicable. The only discrepancy worth mentioning is the point at tilt = 90° and rotation = 300°. This point is the only point with an error greater than 4% (14°), but it is not an obvious "outlier".

Similarly, figure 5.15 bears a very close resemblance to figure 5.13. The outlying point in figure 5.15 occurs, in part, because the ellipse fitting algorithm fails to find any ellipse at all. The histogram tilt estimator then selects 35° as the most plausible tilt value and there is no information available to correct the mistake. Ignoring this one data point, every tilt estimate is within 11% (10°) of the actual value.

Eight extra images were used to further test the response of the rotation estimators. The results of this test are shown in figure 5.16. Low intensity versions of these eight images were used to generate figure 5.17. Both of these figures exhibit extremely low error; the maximum error in either graph is 1% (4°).

Twelve extra images were used to further test the response of the tilt estimators. The results of this test are shown in figure 5.18. Figure 5.19 shows the results obtained using twelve extra low intensity images. Both figures exhibit some error in the 0° to 40° range due to the ellipse fitting errors discussed in chapter 4. Figure 5.19, however, also displays errors for very high tilt angles. These errors arise from histogram correlation. In either case, the maximum error is 11% (10°).

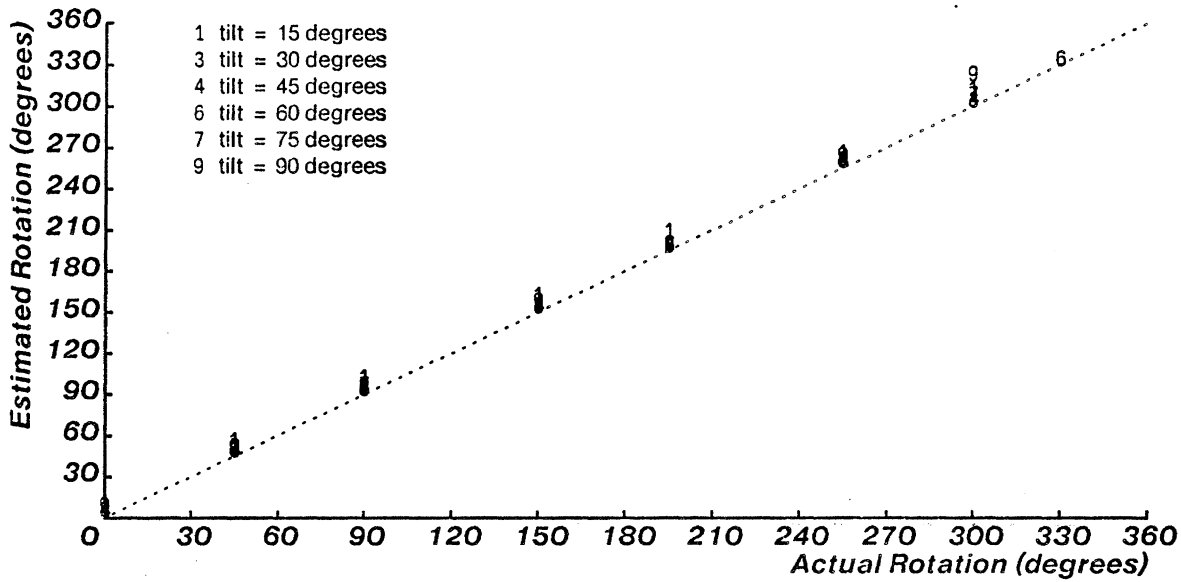


Figure 5.14: Actual transistor rotation vs. estimated rotation from combined system. Points are labelled by tilt for 49 different transistor images under low illumination.

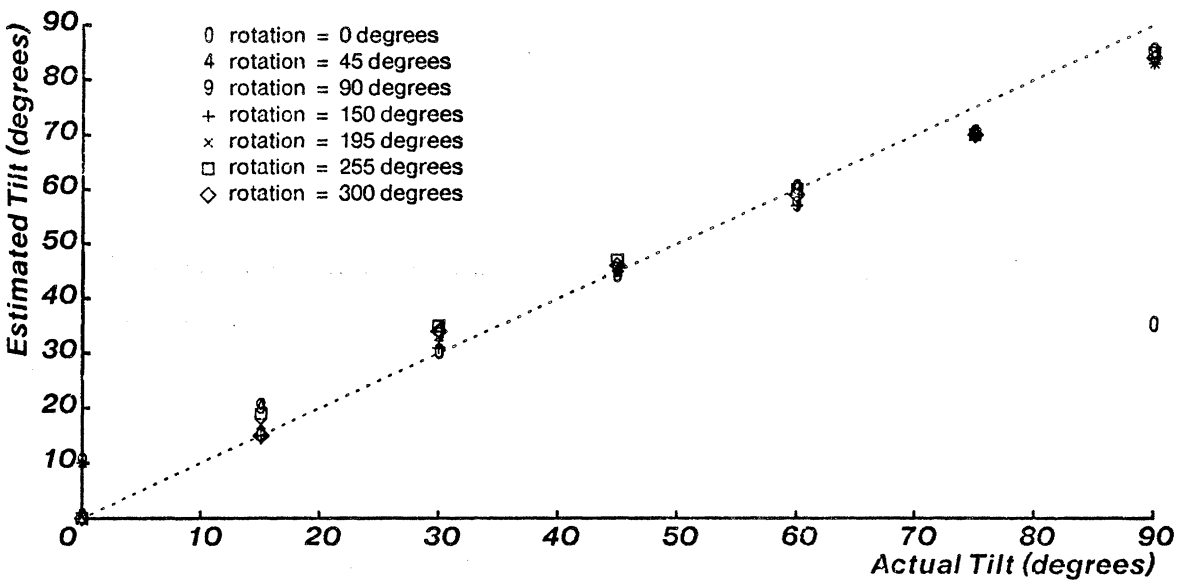


Figure 5.15: Actual vs. estimated tilt from combined system under low illumination. Points are labelled by rotation value.

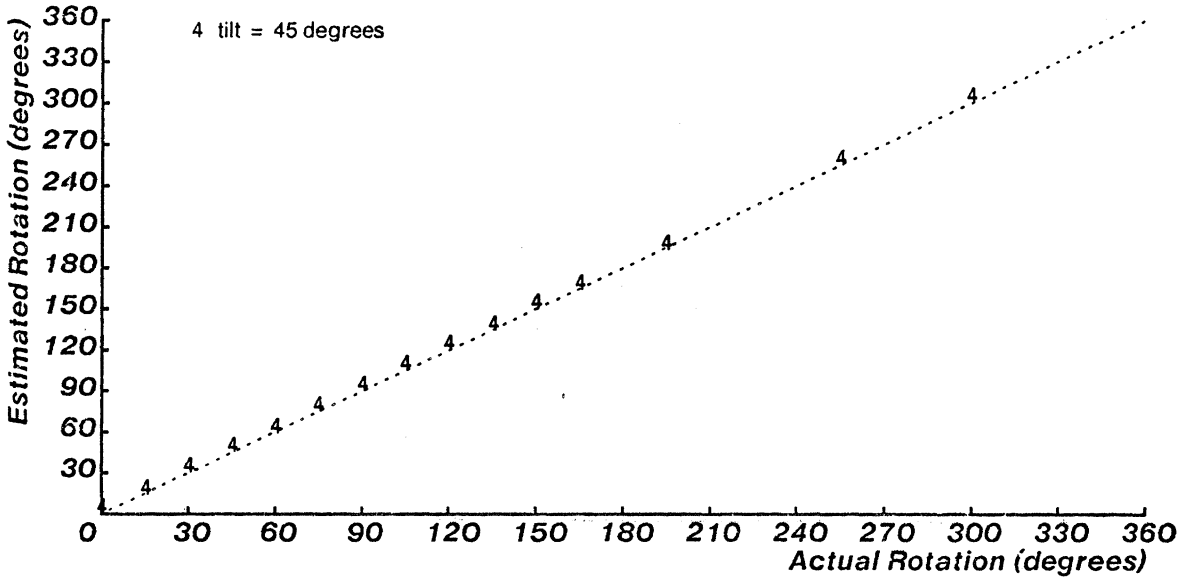


Figure 5.16: Actual transistor rotation vs. estimated rotation from combined system. Points are labelled by tilt for 15 different transistor images under normal illumination.

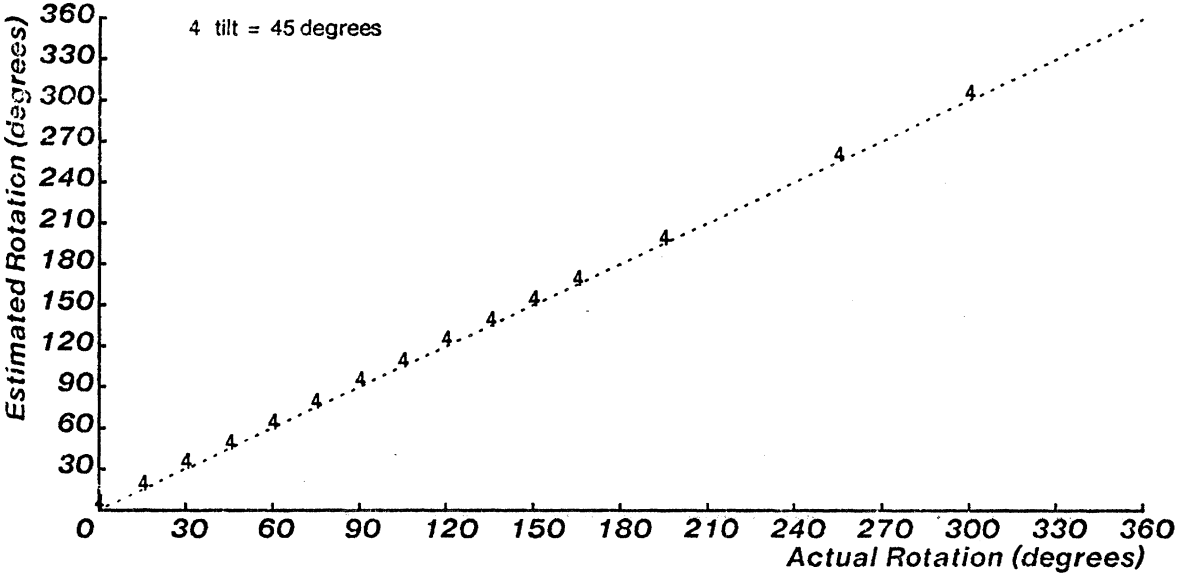


Figure 5.17: Actual vs. estimated rotation of combined system. Points are labelled by tilt for 15 images under low illumination.

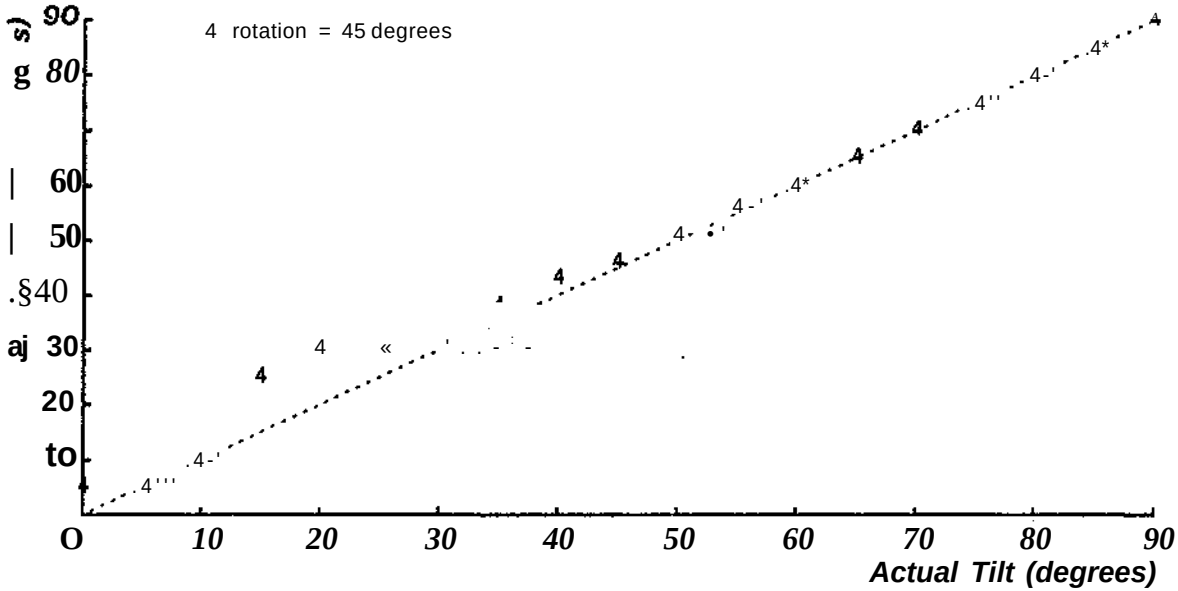


Figure 5.18: Actual transistor tilt vs. estimated tilt from combined system. Points are labelled by rotation for 19 different transistor images under normal illumination.

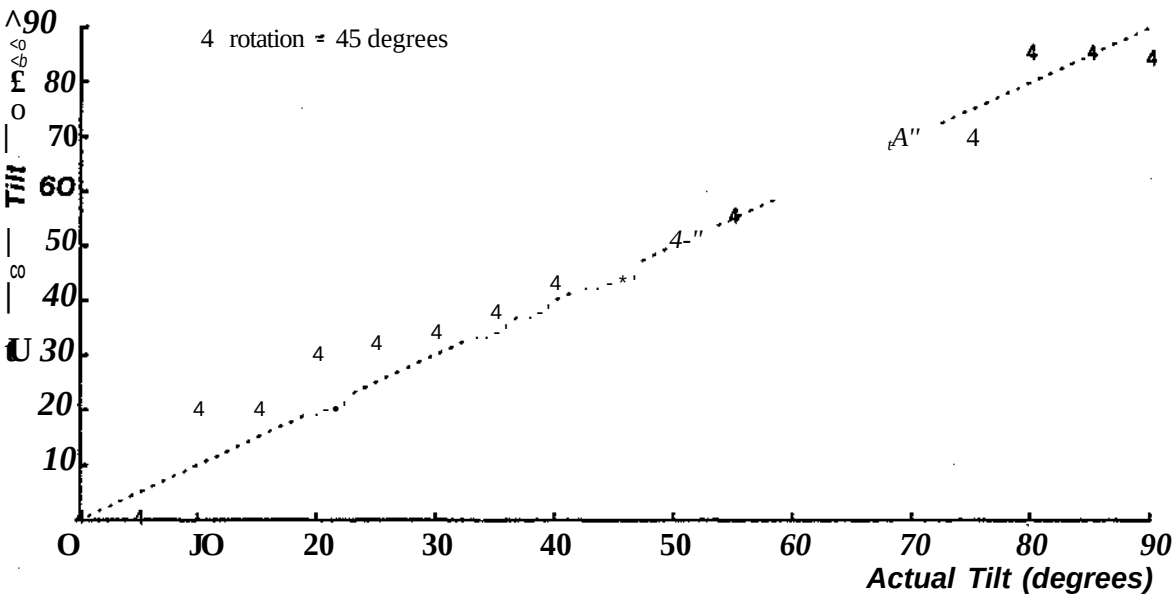


Figure 5*19: Actual vs. estimated tilt from combined system for 19 images under low illumination. Rotation = 45°.

Chapter 6

Conclusion

6.1 Summary

One general goal of computer vision is to be able to produce a semantic description of a scene from a single image consisting of a two-dimensional array of intensity values. The desired description will often consist of a list of the objects in the scene as well as the orientation of those objects and their spatial relationship. This study has examined the problem of determining object orientation from simple real-life images using a limited amount of *a priori* knowledge about the scene being observed.

The emphasis of the work was directed toward devising methods for extracting object orientation information from a single image. The methods were designed to be independent of changes in the illumination intensity of the scene and changes in object size. The general problem of locating the object in the image has been treated only briefly as a prelude to the analysis of orientation. To facilitate this approach, it was assumed that input images would contain a single object on a uniform, but contrasting background. It was further assumed that complete mathematical models of the imaged objects would not be necessary, although a number of sample images and some general description of the object would be available.

Three techniques were presented for estimating object orientation.

- Histogram template matching employed a nearest-neighbor classifier using the normalized correlation function as a distance measure between the histogram of the input image and a set of training histograms.
- Binary connectivity analysis assumed that the object silhouette could be obtained by thresholding the image. The silhouette was then analyzed for connectivity, and the features of the resulting image region were used, in conjunction with some *a priori* knowledge, to determine orientation.
- Ellipse fitting used knowledge about the shape of circles to determine orientation. A circle in a scene corresponds to an ellipse in the image, and the parameters of the ellipse can be used to specify the orientation of the circular surface. Location of the image ellipse was accomplished by exploiting knowledge about object boundaries and image intensity gradients.

The orientation information from each of these three methods was then combined using the plausibility function. This probability-based, sub-optimal, decision rule employed weighted sums of joint conditional probabilities to enhance robustness.

The combined techniques for estimating orientation were tested on 138 images of transistors with good results. Of the 138 estimates for transistor rotation, 137 (99.3%) were within 14° (4%) of the actual rotation value. For the tilt estimates, 136 (98.6%) of the 138 estimates were within 10° (11%) of the actual tilt values. These results demonstrate the feasibility of accurately determining object orientation, using grey-level computer vision, for certain simple classes of monocular images.

6.2 Limitations

The orientation determining methods presented in this report have proved to be quite effective for estimating the orientation of transistors from images of single transistors against a uniform dark background. The methods used, however, were able to employ some simplifying assumptions that exploited the natural constraints inherent in the problem domain. Although these assumptions were stated explicitly wherever they were used, it may be useful to examine the limitations of these methods.

Histogram template matching is limited almost exclusively to scenes which are highly structured. This technique assumes that changes in the image intensity histogram vary smoothly and systematically with object orientation, and that there is a unique correspondence between orientation and histogram shape. The method also requires a set of training histograms. Even under these stringent conditions, histogram matching is incapable of determining rotation about the imaging axis.

Binary connectivity analysis requires that the features of the thresholded image regions vary smoothly and uniquely with object orientation. This requirement is reasonably easy to satisfy (possibly with a small number of ambiguous orientations) for most objects, if it is possible to binarize the grey-level image in such a way that the binary image regions correspond to specific object surfaces. A correspondence between binary regions and object surfaces is unlikely for most complex scenes, but in many industrial applications the imaging environment can be structured so that such a correspondence exists.

The method of ellipse fitting is restricted to those scenes which may generate elliptical features in the image plane. The extraction of the ellipse perimeter points is a difficult task which may involve significant computational cost. The methods presented in this study are optimized to extract the points corresponding to the edge of the unoccluded end face of a protruding cylinder. These methods may not be applicable to the determination of perimeter points for a circular hole, or other object features which generate an ellipse in the image. Once the ellipse is fit, it is necessary to know the relationship between ellipse parameters and object orientation in order to determine the orientation of the object involved.

The method of plausibility functions is limited to representing and combining information which can be expressed as a probability density function. Purely propositional knowledge will not fit into the required framework and is best represented by production rules or some similar construct for performing propositional deductions and inferences. In general, the plausibility function will not yield an optimal solution, and it may require non-trivial amounts of effort to choose a set of plausibility factors which provide a "good" sub-optimal behavior.

6.3 Further Study

6.3.1 Potential Improvements

The results obtained for the transistor application indicate that orientation methods used in this study are useful, but imperfect. Some of these imperfections are explained by violations of the inherent limitations stated above; others are not. Each of these errors has been analyzed and explained in the relevant section of this report. Here, potential remedies will be suggested for the more serious errors.

Histogram template matching suffers from two serious problems which merit further discussion. The first problem stems from the inherent limitation that histogram shape must correspond uniquely to object orientation. In general, this assumption will be false, and the problem is exacerbated by the use of arbitrary amounts of histogram scaling and expansion in order to preserve illumination and

scaling independence. Many erroneous matches are the result of unrealistic amounts of histogram expansion, or the result of non-linear expansion due to the imaging camera. These errors could be reduced using a more linear camera and reasonable heuristic assumptions about the possible deviation of illumination and object area.

Another source of histogram error occurs because histograms do not vary uniformly with changes of orientation. Histogram shape may vary rapidly for some ranges of orientation and quite slowly over other ranges. These problems could be ameliorated by quantizing the training set in narrow intervals over the range of orientations where histograms vary rapidly and by quantizing in wider intervals over the ranges where histograms change slowly.

The major drawback of binary connectivity analysis is the use of binary images. Many scenes do not easily lend themselves to segmentation based on a single threshold. As discussed in chapter 3, there are many alternate region-based segmentation schemes which employ more sophisticated techniques. Such sophisticated methods may be used to separately segment the top, leads and body of a transistor, thereby greatly increasing the available information and enhancing orientation estimates. The use of pattern recognition techniques to analyze region features would also increase the accuracy of the orientation estimation.

The potential improvements for ellipse fitting fall into three categories. The first category of improvement is in the area of mathematical methods for ellipse fitting. A more accurate ellipse fit can be obtained using a more exact error measure for determining the distance between a data point and an ellipse. This modification, however, will require iterative solutions and will therefore be computationly more expensive.

The second potential improvement is the use of additional heuristic constraints specific to the transistor application. The heuristic used to determine ellipse shaped boundary sections could be improved by employing constraints on the shape or area of the desired section of the curvature graph. The ellipse error conditions could also be exploited to greater advantage. Ellipse "failure" provides significant information about transistor tilt, but this information is currently ignored. Similarly, the "wrong-end" condition could be used, in conjunction with the original curvature graph, to search for the boundary section corresponding the transistor top, thereby improving the ellipse fit.

The process of ellipse fitting might also be improved by replacing the ellipse fitting algorithm used in this study with a Hough transform algorithm. The Hough transform method does not require the extraction of candidate ellipse points and may be more useful for some applications. The Hough method may, however, be more expensive in terms of computational complexity and memory requirements since it requires finding local maxima in a five dimensional accumulator array.

Finally, there is the subject of combining knowledge. The *a priori* pdfs used to represent the individual estimators of transistor orientation are rather simplistic models for the estimation process. These models could be greatly enhanced by including the notion of failure, so that no single model could incorrectly rule out a parameter value. The values of the plausibility factors could then be changed to reflect some of the joint density terms which would provide a more accurate final estimate of orientation given the improved models.

6.3.2 Other Areas of Application

The problem of determining object orientation from a single view is a task of great practical interest. Many potential industrial applications require a vision system to determine the location and orientation of an object so that the object may be inspected, or acquired using a robot manipulator. The "bin-picking" task is a typical example of a useful industrial application [Kelley 1982]. The problem of loading industrial machines from bins of randomly oriented parts requires that the location and orientation of an appropriate part be determined so that it may be grasped.

The orientation estimating techniques described in this report have been demonstrated by applying them to the problem of determining transistor orientation, and it may be useful to consider whether these techniques are also relevant to the bin-picking problem. The general limitations of each method, as discussed in the previous section, can be used to determine whether the methods will work well for any given application. However, discussion of specific points may help to clarify the concepts.

The major difficulty introduced by considering a bin of parts is the task of locating a single object in the image. This report has emphasized methods for determining orientation given that it is "reasonably easy" to locate a single object in the image. This assumption is no longer valid for the bin of parts problem, and new techniques must be developed to segment the image into regions which correspond to individual objects. The ease with which this can be accomplished is highly task-specific and will not be addressed here except to note that a number of segmentation techniques have been proposed in the literature [Fu 1981].

The remaining difficulties lie in the interactions between the stacked parts. Changes in the lighting conditions of individual objects may be caused by shadows or reflections from the surfaces of other objects. The occlusion of objects is also a new factor to consider. Given a viable segmentation scheme, however, each of the orientation techniques may be modified for use in the new problem domain.

Histogram template matching, for example, could be used by obtaining the

histogram of one segmented region and comparing it to a number of oriented histograms as explained in chapter 2. This modified technique will fail if the lighting conditions in the bin are radically different from the lighting conditions used to obtain the histogram training set. In an industrial setting, however, it may be possible to structure the lighting to minimize such effects.

The use of binary connectivity analysis is also possible given a good segmentation scheme. The segmented image region is passed to the binary connectivity algorithm to determine the relevant features of the region. The orientation analysis would have to be changed to handle (or ignore) the silhouettes of occluded objects, but typically there will be a number of unoccluded objects at the surface of the bin.

Given objects with circular surfaces, the use of ellipse fitting is viable, subject to approximately the same limitations as binary connectivity analysis. In other words, the region boundaries can be used as the initial input to the ellipse fitting algorithm, as described in chapter 4. Some modifications would be required to deal effectively with the presence of partial object boundaries due to occlusion.

The use of plausibility functions remains unchanged for the bin-picking application because they represent a general method for combining the information from individual orientation estimators. It is this combination of sometimes unreliable information which accounts for the power of the results presented.

References

[Agin 1973] Agin, G.J. and Binford, T.O., "Computer Description of Curved Objects," *International Joint Conference on AI 3*, 1973, pp. 629-640

[Agin 1979] Agin, G.J., "Sequential Strategies for Finding and Identifying Objects with a Plane of Light," *SRI International 9th Report on Machine Intelligence Research Applied to Industrial Automation*, D. Nitzan et al, ed., SRI International, Menlo Park, CA, 1979, pp. 109-121

[Agin 1980] Agin, G.J., "Computer Vision Systems for Industrial Inspection and Assembly," Advances in Computer Technology - 1980, Vol. 1 sponsored by ASME Century 2, San Francisco, CA, August 12-15, 1980, pp. 1-7

[Agin 1981] Agin, G.J., "Fitting Ellipses and General Second-Order Curves," Technical Report CMU-RI-81-5, CMU Robotics Institute, July, 1981

[Agin 1984] Agin, G.J., "Vision Systems," in *Handbook of Industrial Robotics*, Shimon Nof, ed., Wiley, New York, 1984 (in press)

[Albano 1974] Albano, A., "Representation of Digitized Contours in Terms of Conic Arcs and Straight-Line Segments," *Computer Graphics and Image Processing,* vol. 3, 1974

[Ballard 1976] Ballard, D.H., and Sklansky, X, "A ladder-structured decision tree for recognizing tumors in chest radiographs," *IEEE Trans. Computers,* vol. 25, no. 5, May, 1976, pp. 503-513

[Ballard 1982] Ballard, D.H., and Brown, CM, *Computer Vision,* Prentice-Hall, N.J., 1982

[Barrow 1981] Barrow, H.G. and Tennenbaum, J.M., "Computational vision," *IEEE Proceedings,* vol. 69, no. 5, May, 1981, pp. 572-595

[Bennett 1975] Bennet, J.R. and Mac Donald, J.S. "On the Measurement of Curvature in a Quantized Environment," *IEEE Transactions on Computers,* Vol. C-24, no. 8, August, 1975, pp. 803-820

[Beer 1977] Beer, F.P. and Johnston, E.R., Jr., *Vector Mechanics for Engineers Statics and Dynamics,* McGraw Hill Book Company, New York, 1977

[Bolles 1981] Bolles, R.C., Kremers, J.H. and Cain, R.A., "A simple sensor to gather three-dimensional data," Tech. Note 249, SRI International, Inc., Menlo Park, CA., 1981

[Bookstein 1978] Bookstein, F.L., "The Measurement of Biological Shape and Shape Change," *Lecture Notes in Biomathematics,* vol. 24, no. 8, 1978

[Bookstein 1979] Bookstein, F.L., "Fitting Conic Sections to Scattered Data," *Computer Graphics and Image Processing,* vol 9, no. 1, January, 1979, pp. 56-71

[Bracho 1983] Bracho, R., Schlag, J. and Sanderson, A.C., "POPEYE: A Gray-Level Vision System for Robotics Applications," Technical Report CMU-RI-TR-83-6, Robotics Institute, CMU, May, 1983

[Brice 1970] Brice, C.R. and Fennema, C.L., "Scene Analysis Using Regions," *Artificial Intelligence*, vol. 1, 1970, pp. 205-226

[Brooks 1981] Brooks, R.A., "Symbolic Reasoning among 3-D Models and 2-D Images," *Artificial Intelligence*, vol. 17, August, 1981

[Buchanan 1984] Buchanan, B. and Shortliffe, E.H., *Rule-Based Expert Programs: the MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley, Reading, MA, 1984

[Buchanan 1969] Buchanan, B., Sutherland, G. and Feigenbaum, E.A., "Heuristic DENDRAL: A Program for Generating Explanatory Hypotheses in Organic Chemistry," *Machine Intelligence J[*, American Elsevier, New York, 1969

[Cooper 1976] Cooper, D.R., Yalabik, N., "On the computational cost of approximating and recognizing noise-perturbed straight lines and quadratic arcs in the plane," *IEEE Trans. Computers*, C-25, 1976

[Cunningham 1981] Cunningham, R., "Segmenting binary images," *Robotics Age*, July/Aug., 1981, pp. 4-19

[Davis 1975] Davis, L.S., "A survey of edge detection techniques," *Computer Graphics and Image Processing*, vol. 4, April, 1975, pp. 248-270

[Davis 1977] Davis, R., Buchanan, B. and Shortliffe, E., "Production Rules as a Representation for a Knowledge-Based Consultation Program," *Artificial Intelligence*, vol. 8, 1977

[Duda 1973] Duda, R.O. and Hart, P.E., *Pattern Classification and Scene Analysis*, Wiley-Interscience, New York, 1973

[Duda 1976] Duda, R.O., Hart, RE. and Nilson, N.J., "Subjective Bayesian Methods for Rule-Based Inference Systems," TR-124, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, CA., 1976

[Eccles 1977] Eccles, M.J., McQueen, M.P.C. and Rosen, D., "Analysis of the digitized boundaries of planar objects," *Pattern Recognition,* vol. 9, January, 1977, pp. 31-41

[Fu 1981] Fu, K.S. and Mui, J.K., "A survey on image segmentation," *Pattern Recognition,* vol. 13, 1981, pp. 3-16

[Fukunaga 1972] Fukunaga, K., *Introduction to Statistical Pattern Recognition,* Academic Press, N.Y., 1972

[Gonzalez 1978] Gonzalez, R.C. and Thomason, M.G., *Syntactic Pattern Recognition,* Addison Wesley, 1978

[Hall 1982] Hall, E.L., Tio, J.B.K., McPherson, C.A. and Sadjadi, F.A., "Measuring curved surfaces for robot vision," *IEEE Computer,* vol. 15, no, 12, December, 1982, pp. 42-54

[Hanson 1978] Hanson, A. and Riseman, E., *Computer Vision Systems,* Academic Press, 1978

[Horn 1975] Horn, B.K.P., "Obtaining Shape from Shading Information," in *The Psychology of Computer Vision,* (P.H. Winston, ed), McGraw-Hill, New York, 1975

[Horowitz 1976] Horowitz, S.L., and Pavlidis, T., "Picture segmentation by a tree traversal algorithm," J. *ACM,* vol. 23, no. 2, April, 1976, pp. 368-388

[Huffman 1975] Huffman, D,A.9 "Curvature and Creases: A Primer on Paper," *Proceedings of Conference on Computer Graphics, Pattern Recognition and Data Structures,* May, 1975

[Hummel 1975] Hummel, R.A., "Histogram modification techniques," *Computer Graphics and Image Processing,* vol. 4, 1975, pp. 209-224

[Kanade 1979] Kanade, T., "Recovery of the three-dimensional shape of an object from a single view.," Technical Report CMU-CS-79-153, Computer Science Department, Carnegie-Mellon University, October, 1979

[Kashioka 1977] Kashioka, S., Takeda, S., Shima, Y., Uno, T., and Mamoda, T., "An approach to the integrated intelligent robot with multiple sensory feedback: visual recognition techniques," *Proc. Seventh Int. Symposium on Industrial Robots*, Tokyo, Japan, October, 1977, pp. 531-538

[Kelley 1982] Kelley, R.B., Birk, J.R., Martins, H.A. and Tella, R., "A Robot System Which Acquires Cylindrical Workpieces from Bins," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-12, no. 2, March/April, 1982, pp. 204-213

[Kender 1980] Kender, J.R., "Shape from Texture," Ph.D. Thesis, Carnegie-Mellon University, November, 1980

[Levine 1969] Levine, M.D., "Feature Extraction: A Survey," *Proceedings of IEEE*, vol. 57, August, 1969, pp. 1391-1407

[Lynch 1973] Lynch, R.V., Ostberg, D.R., and Kuller, R.G., *Calculus with computer applications*, Xerox College Publishing, Lexington, Massachusetts, 1973

[Marr 1977] Marr, D., "Analysis of Occluding Contour," *Proceedings of the Royal Society of London, B-197*, 1977, pp. 441-475

[Marr 1982] Marr, D., *Vision*, W.H. Freeman and Co., San Francisco, CA, 1982

[McDermott 1982] McDermott, J., "R1: A Rule-Based Configurer of Computer Systems," *Artificial Intelligence*, vol. 19, no. 1, 1982

[Ohlander 1975] Ohlander, R.B., "Analysis of Natural Scenes," Doctoral Dissertation, Computer Science Department, Carnegie-Mellon University, 1975

[Oppenheim 1975] Oppenheim, A.V., *Digital Signal Processing*, Prentice-Hall, New Jersey, 1975, pp. 28-29

[Papoulis 1965] Papoulis, A., *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, NY, 1965

[Parzen 1962] Parzen, E., "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, 1962, pp. 1065-1076

[Paton 1970] Paton, K.A., "Conic Sections in chromosome analysis," *Pattern Recognition*, vol. 2, 1970

[Pavlidis 1978] Pavlidis, T., "A review of algorithms for shape analysis," *Computer Graphics and Image Processing*, 1978, pp. 243-258

[Pednault 1981] Pednault, E.P.D., Zucker, S.W. and Muresan, L.V., "On the Independence Assumption Underlying Subjective Bayesian Updating," *Artificial Intelligence*, vol. 16, no. 2, 1981

[Pople 1982] Pople, H.E., Jr.. "Heuristic Methods for Imposing Structure on Ill-Structured Problems: The Structuring of Medical Diagnostics," in *Artificial Intelligence in Medicine*, edited by P. Szolovits, Westview Press, Boulder, CO, 1982

[Pratt 1978] Pratt, W.K., *Digital Image Processing*, Wiley-Interscience, New York, 1978

[Proc 1979] "Special Issue on Pattern and Image Processing," *IEEE Proceedings*, vol. 67, no. 5, May, 1979, pp. 705-880

[Proc 1981] "Special Issue on Image Processing," *IEEE Proceedings*, vol. 69, no. 5, May, 1981, pp. 497-672

[Roberts 1965] Roberts, L., "Machine Perception of Three-Dimensional Solids," in it Optical and Electro-Optical Information Processing, J.Tippett, ed., MIT Press, Cambridge, MA, 1965, pp. 159-197

[Rosenfeld 1969] Rosenfeld, A., *Picture Processing by Computer*, Academic Press, New York, 1969

[Rosenfeld 1978] Rosenfeld, A., "Some Recent Results Using Relaxation-Like Processes," *Proceedings of the Image Understanding Workshop*, 1978, pp. 100-104

[Rosenfeld 1979] Rosenfeld, A., and Davis, L.S., "Image segmentation and image models," *IEEE Proceedings*, vol. 67, no. 5, May, 1979, pp. 764-772

[Rosenfeld 1981] Rosenfeld, A., "Image pattern recognition," *IEEE Proceedings*, vol. 69, 1981, pp. 596-605

[Sanderson 1976] Sanderson, A.C. and Kobler, B., "Sequential interval his-, togram analysis of non-stationary neuronal spike trains," *Biol. Cybernetics*, vol. 22, 1976, pp. 61-71

[Sanderson 1983] Sanderson, R.J., "A Survey of the Robotics Vision Industry," *Robotics World*, February, 1983, pp. 22-31

[Schlag 1983] Schlag, J., Sanderson, A.C, Neuman, C.P., and Wimberly, F.C., "Implementation of Automatic Focusing Algorithms for a Computer Vision System with Camera Control," Technical Report CMU-RI-TR-83-14, CMU Robotics Institute, August, 1983

[Shirai 1971] Shirai, Y. and Suwa, M., "Recognition of Polyhedrons with a Rangefinder," *International Joint Conference on AI 2*, 1971, pp. 80-87

[Smith 1984] Smith, D.R., Ph.D. Thesis, Carnegie-Mellon University, 1984 (in preparation)

[Tsai 1980] Tsai, W-H. and Fu, K-S., "Attributed grammar — A tool for combining syntactic and statistical approaches to pattern recognition," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 10, no. 12, December, 1980, pp. 873-885

[Vanderbrug 1979] Vanderbrug, G.J., Albus, J.S. and Borkmeyer, E., "A Vision System for Real-Time Control of Robots," *Proceedings of the Ninth International Symposium and Exposition on Industrial Robots*, 1979, pp. 213-231

[Ward 1979] Ward, M.H., Rossol, L. and Holland, S.W., "CONSIGHT: A Practical Vision-Based Robot Guidance System," 9th International Symposium on Industrial Robots, Society of Manufacturing Engineers and Robot Institute of America, Washington, DC, March, 1979, pp. 195-212

[Weseka 1976] Weseka, J.S., "A Survey of Threshold Selection Techniques," *Computer Graphics and Image Processing*, vol. 5, 1976, pp. 382-399

[Winston 1979] Winston, P.H., *Artificial Intelligence*, Addison-Wesley Publishing Company, Reading, MA., 1979

[Wylie 1966] Wylie, C.R., *Advanced Engineering Mathematics*, McGraw-Hill Book Company, New York, 1966