1989

# Edge Contours

Donna J. Williams
dwilliam@stetson.edu

# EDGE CONTOURS

By
Donna J. Williams

1989

UCF

UNIVERSITY OF CENTRAL FLORIDA

OFFICE OF GRADUATE STUDIES

DISSERTATION APPROVAL

DATE: July 10, 1989

BASED ON THE CANDIDATE'S SUCCESSFUL ORAL DEFENSE, IT IS RECOMMENDED

THAT THE DISSERTATION PREPARED BY Donna J. Williams

ENTITLED Edge Contours

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF Doctor of Philosophy

FROM THE DEPARTMENT OF Computer Science

IN THE COLLEGE OF Arts and Sciences

# EDGE CONTOURS

by

DONNA J. WILLIAMS

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
the Department of Computer Science at
the University of Central Florida
Orlando, Florida

August 1989

Major Professor: Mubarak A. Shah

UNIVERSITY OF CENTRAL FLORIDA

OFFICE OF GRADUATE STUDIES

DEFENSE OF DISSERTATION

THE UNDERSIGNED VERIFY THAT THE FINAL ORAL DEFENSE OF THE DOCTOR OF PHILOSOPHY DISSERTATION OF Donna J. Williams

HAS BEEN SUCCESSFULLY COMPLETED ON July 10, 1989

TITLE OF DISSERTATION: Edge Contours

MAJOR FIELD OF STUDY: Computer Science

## ABSTRACT

The accuracy with which a computer vision system is able to identify objects in an image is heavily dependent upon the accuracy of the low level processes that identify which points lie on the edges of an object. In order to remove noise and fine texture from an image, it is usually smoothed before edge detection is performed. This smoothing causes edges to be displaced from their actual location in the image. Knowledge about the changes that occur with different degrees of smoothing (scales) and the physical conditions that cause these changes is essential to proper interpretation of the results obtained. In this work the amount of delocalization and the magnitude of the response to the Normalized Gradient of Gaussian operator are analyzed as a function of $\sigma$, the standard deviation of the Gaussian. As a result of this analysis it was determined that edge points could be characterized as to slope, contrast, and proximity to other edges. The analysis is also used to define the size that the neighborhood of an edge point must be in order to assure its containing the delocalized edge point at another scale when $\sigma$ is known.

Given this theoretical background, an algorithm was developed to obtain sequential lists of edge points. This used multiple scales in order to achieve the superior localization and detection of weak edges possible with smaller scales combined with the noise suppression of the larger scales. The edge contours obtained with this method are significantly better than those achieved with a single scale. A second algorithm was developed to allow sets of edge contour points to be represented as active contours so that interaction with a higher level process is possible. This higher level process could do such things as determine where corners or discontinuities could appear. The algorithm developed here allows hard constraints and represents a significant improvement in speed over previous algorithms allowing hard constraints, being linear rather than cubic.

To

my husband

Gareth

## ACKNOWLEDGEMENTS

I would like to express my appreciation to my advisor, Dr. Mubarak Shah, for his guidance and encouragement in this work. It has been a privilege to work under him. I would like to thank my committee members, Dr. Robert C. Brigham, Dr. J. Michael Moshell, and Dr. Harley R. Myler, for their time and assistance. My thanks also go to the administrative and technical staff of the Department of Computer Science for their support, and Susanne Payne for her editorial assistance.

My fellow students at UCF have made my stay here most enjoyable. I would like to mention especially Teresa Haynes, Greg Schaper, Viva Wingate, Chris Walker, Ranga, Jennifer Burg, Julie Carrington, Krishnan, Shiva Kumar, Jay Hackett, Kristine Gould, Bill Hughes, and Rick Weddleton.

I would like to extend a special appreciation to my husband, Gareth, and my sons, Brian and Jeff, who have cheerfully endured many inconveniences during the time I have been a student.

The research reported here was supported by the Center for Research in Electro Optics and Lasers (CREOL), University of Central Florida under grant number 20-52-043 and by DOD/PMTRADE under contract number N61339-88-G-0002/6. The content of the information herein does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

TABLE OF CONTENTS

# 1. INTRODUCTION

Computer vision is a new field currently undergoing tremendous growth along with the fields where its primary applications lie: space, medicine, and robotics. It is concerned with arrays of numbers called images. The values represent the magnitude of some value such as light, heat (infrared radiation), or distance from the camera at a point. In cases where stereo, motion, or color information is available, each image may actually be a set of images. The goal of computer vision is to interpret what objects are present in the image, and perhaps determine the shape, orientation, or motion of the objects. The detail to be obtained from an image depends on the application. For instance, a system looking for defects in parts on an assembly belt has a very narrow range of objects which it must be able to identify, but it must be able to interpret an image in great detail in order to determine if there is a defect. A system analyzing satellite images in order to determine, for example, the strength of air defenses, must be able to recognize the types of features visible from the air such as roads, rivers, buildings, and airfields. When an airfield has been identified, the types of airplanes on the airfield must be determined. Many more models in more orientations must be identified, but the type of detail needed is different from that needed for the assembly line. The task of a mail cart robot is much easier. It must be able to follow a predefined path, and recognize when there is an obstacle in its path. The requirements of all these systems are very different.

All the above systems depend on being able to identify objects in an image. The accuracy with which they are able to do this is heavily dependent upon the accuracy of what are referred to as low level processes. These processes are the initial work done on an image and produce information which is suitable for the higher level,

application dependent processing. Two early processes are called segmentation and edge detection. The first is concerned with identifying regions in the image which belong to the same object based on similarity of color, texture, etc. Its complement is edge detection, which has the goal of identifying points in an image where the intensity is changing rapidly and determining which of these correspond to the edges of objects. An outline thus obtained can be matched to a projection of a three-dimensional model of an object.

Some of the difficulties involved in this task can be seen by considering the example in Figures 1 and 2. The top graph in Figure 1 is a slice across a two-dimensional image. The image is 128 pixels wide. The location of five major edges or near vertical portions of the curve are identified by the arrows at the top. Several smaller edges can also be seen between the two large edges on the left. When an edge occurs, the transition from one intensity to another takes place over an interval of several pixels. Three approaches have been used to detect edges. First, a model edge can be matched to the image, and the points with the best matches identified as edge points. Another approach is to fit a surface to the image at each point, and compute the gradient of the surface fitted. The points with largest gradient magnitude are the edge points. A third approach is to apply some operator to estimate the gradient directly at each point. This is the approach advocated by Canny [8] and used in the work presented here. An example of this is given in Figure 2. The derivative at each point is estimated by taking the difference of a weighted average of points to the left and to the right of the point. Points farther from the point where the derivative is being estimated have a smaller weight. In Figure 2 the size of the neighborhood used to estimate the derivative increases from top to bottom. It is 3 pixels on either side for the top row and increases to 7, 10, and 14 for the other rows. The operation of taking a weighted average is called smoothing. The effect of smoothing the original image directly is

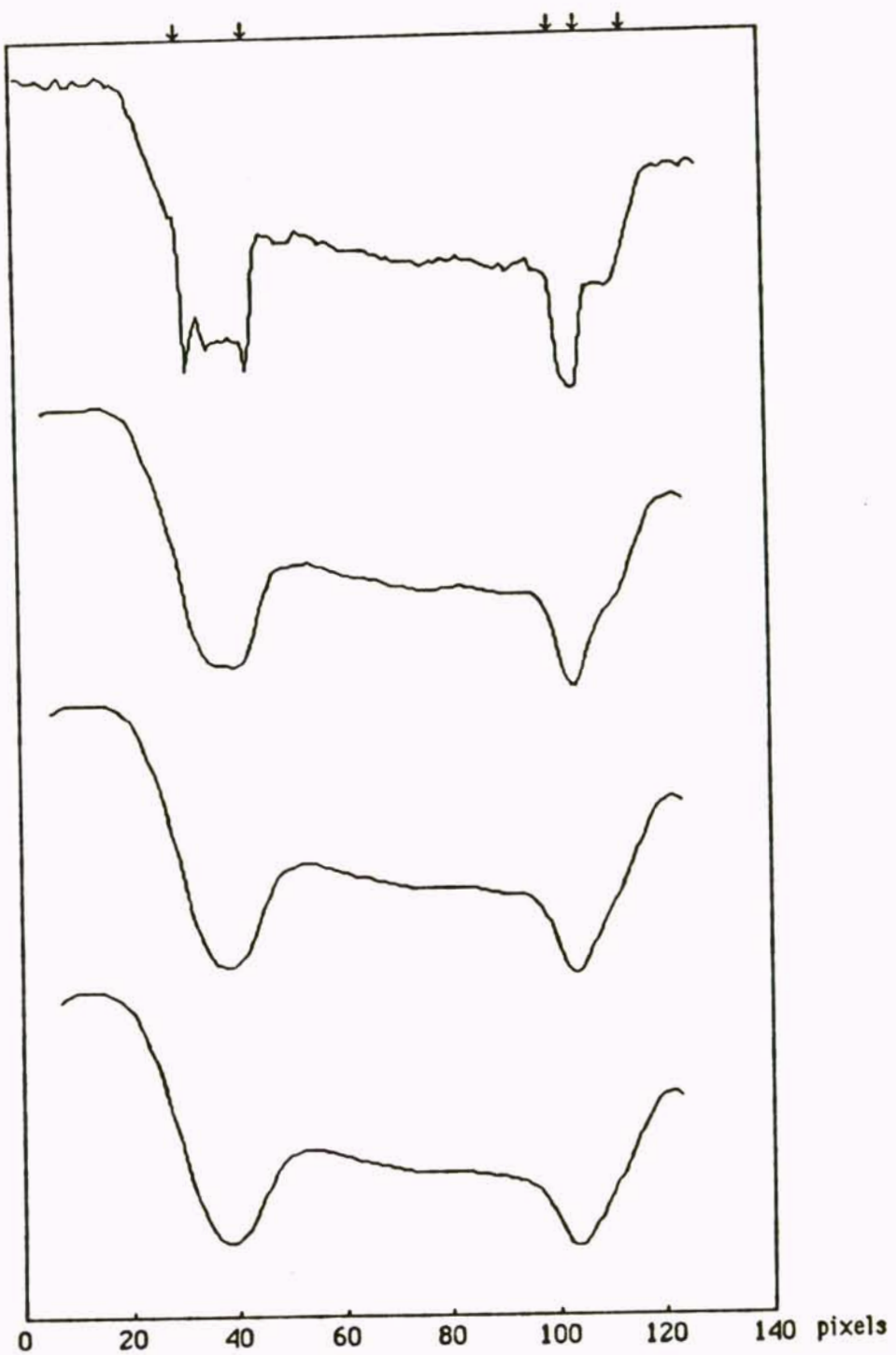



Figure 1: One row from an image (top graph) and the result of smoothing the image at different scales. Scale (degree of smoothing) increases from top to bottom.

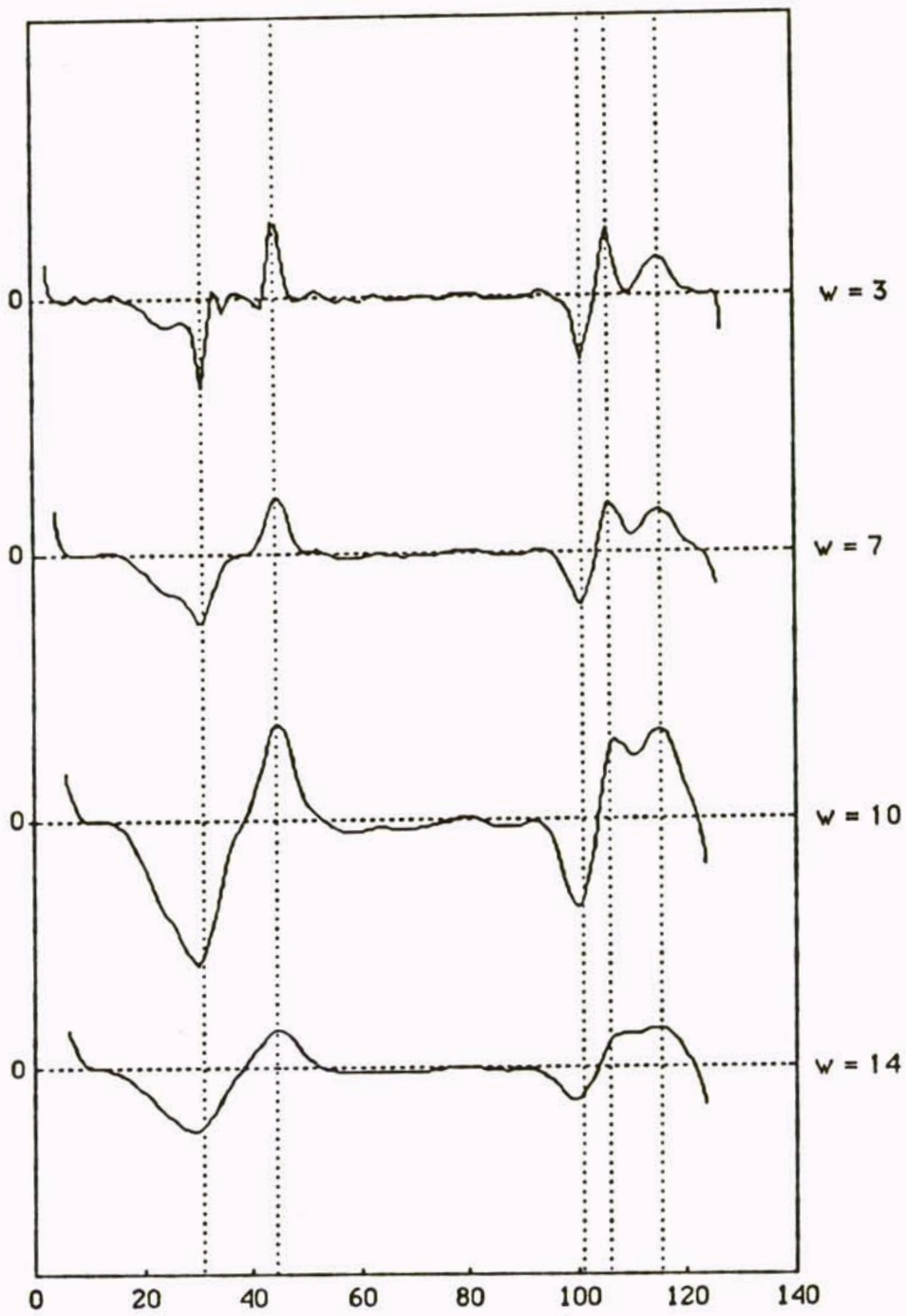



Figure 2: The derivative of the image of Figure 1. The width of the neighborhood for smoothing from top to bottom is 3, 7, 10, 14 pixels on either side of a point. The dotted lines mark the original location of the edges.

shown in rows 2-4 of Figure 1. The positive peaks and negative valleys in the first derivative graph correspond to the positions in the image where the intensity change is highest.

Using techniques of calculus, the locations of maxima and minima can be computed by taking the derivative and finding points where this is 0. Since the previous graph was of a first derivative, this is actually a second derivative of the original image. Points where the second derivative is 0 are referred to as *zero-crossings* in vision. This is because when a continuous curve is sampled at discrete intervals, it is highly unlikely that the exact point where a curve crosses 0 will be one of the points sampled. Points sought will be those where the sign of the curve changes from positive to negative or vice versa. On the right in Figure 2 there is a minimum which is not negative. This corresponds to the flat area between the two rightmost steps in Figure 1 and indicates a point where the intensity change is smallest. Points such as this will also have a zero second derivative.

As the degree of smoothing increases, some of the peaks disappear and some of them are displaced slightly. Thus one of the problems of edge detection is to determine how much smoothing should be done in order to smooth out the small variations in the image and at the same time cause the smallest delocalization while not removing any significant edges. This is the problem of scale. Because determining the optimum scale for smoothing is difficult, and the goals of removing noise and minimizing delocalization are in conflict, more than one scale is often employed in order to obtain the sensitivity of the small scale combined with the noise resistance of the larger scales.

The example presented above was a one-dimensional slice of a two-dimensional image. In two dimensions the problem of edge detection becomes more complicated. There is no direct equivalent of the first and second derivatives.

If the image is considered to be the two-dimensional function $f(x,y)$, then the partial derivatives of $f$ with respect to $x$ and $y$ are the components of the gradient vector, $(f_x, f_y)$, and give the rate of change of intensity in the $x$ and $y$ directions respectively. The direction of maximum rate of change at each point can be computed by $\theta = tan^{-1}(f_y/f_x)$, $0 \leq \theta < 360°$. The magnitude of the rate of change is $M(x,y) = \sqrt{f_x^2 + f_y^2}$. $M(x,y)$ is also equivalent to the directional derivative of $f$ in the direction of greatest change. The second derivative can be estimated by a function such as the Laplacian, $f_{xx} + f_{yy}$, or by $f_{xx} - 2f_{xy} + f_{yy}$. Rather than use a second derivative operator, a process called non-maxima suppression can be used to identify the points which have a gradient magnitude larger than that of neighbors in the direction of greatest change. This avoids the problem of having to eliminate points which are actually gradient minima as well as that of having to decide on a second derivative operator which does not accurately reflect the quantity of interest. In noisy images there is also the problem that the higher the derivative being estimated, the more it is affected by the noise. Thus whenever possible it is preferable to use lower order derivatives.

Once points which are potentially edges have been identified, the ones which form coherent edge contours should be combined, and a method found to represent the higher level structure. The research reported in this dissertation is concerned with methods to combine points which have been identified as gradient maxima into lists of edge points, and how to determine meaningful information from the points thus obtained. Chapter 2 gives the background to the research. In Chapter 3 a method is presented of tracing gradient maxima contours and linking the points into coherent edge segments using the direction and magnitude information. Gradient maxima points which are not in a contour having length at least 3 pixels are eliminated. This removes isolated noisy points. The original algorithm is then extended to one

which uses multiple scales in order to improve the detection of edges which are weak or near other edges, hence undetectable at scales large enough to remove the noise. The coarse to fine algorithm ensures that fine scale noise will not be included. Since there are occasional jogs in the contours obtained with this algorithm, due to the fact that edge points at the larger scales have been delocalized more than at fine scales, another algorithm was developed. This one again uses multiple scales, but combines them during the step which identifies gradient maxima. By matching points which are being identified at different scales, it is able to remove much of the delocalization effects, resulting in more accurate contours, again without including fine scale noise. Thus it achieves the localization and edge detection properties of the small scale with the noise reduction property of the large scale.

Because both of the multiple scale algorithms must search the vicinity of an edge point at one scale to find the same point at another scale, it is necessary to know how far an edge point may move as a function of the degree of smoothing. Qualitative studies have been performed, e.g., [39] indicating that edges having opposite polarity will move apart as the scale increases while those having the same polarity will move closer together, and become one edge, as the scale increases. This combining of two edges can be observed in the rightmost two edges of Figure 1. Chapter 4 presents a quantitative analysis of how much movement will occur as a function of the polarity, relative strength, and distance between neighboring edges. It is shown that a limit can be placed on the size neighborhood to be searched, or alternately, the distance between smoothing scales for a fixed size neighborhood, so that the same edge at two different scales can be identified.

As the image is smoothed by different amounts, the response to the gradient operator also changes. It has been shown [10] that when the operator used is the gradient of Gaussian (which will be described in detail in the next chapter), the response will

decrease as scale increases for gradient maxima, and will increase for gradient minima. However, if this operator is *normalized* by multiplying by the proper factor, the response will be constant for an isolated step edge, one where image intensity changes from one constant value to another in one pixel. The response will increase and decrease under certain other circumstances. The behavior of this normalized operator is examined in Chapter 5, and based on this analysis it is determined that the characteristics of slope, contrast, and optimum scale can be computed for an edge. This is demonstrated for a series of images. It is also shown that as adjacent edges interact they satisfy the condition called conservation of contrast.

Finally, when a set of points has been determined to lie on a single edge segment, it is possible to do further processing to make the information about the segment more meaningful. A common practice is to fit straight line segments to the points. However, straight lines do not give a unique representation of curved lines. These contours can be smoothed themselves using one-dimensional equivalents of the techniques for smoothing the original two-dimensional image. It has been suggested [17] that contours should be represented in such a manner that the degree of smoothing can be determined by the demands of the higher level processing. Two algorithms which had previously been developed to solve this problem are discussed in Chapter 6. One uses techniques of variational calculus, while the other uses dynamic programming to introduce flexibility into the kind of constraints used, at the expense of being much slower. These algorithms are examined, and a new one presented which combines the flexibility of the dynamic programming solution with much faster run times. A discussion is also included on methods of approximating the curvature of contours which are represented by discrete samples.

# 2. LITERATURE SURVEY

## 2.1 Edge Detection

Edge points are usually identified by some type of gradient operator, since edges of objects are points in an image where the intensity values are changing rapidly. Methods must then be developed to identify which points will be considered significant edges. A crude method of doing this is to simply threshold the gradient image, and identify points above some value as edge points. This results in wide lines and isolated points.

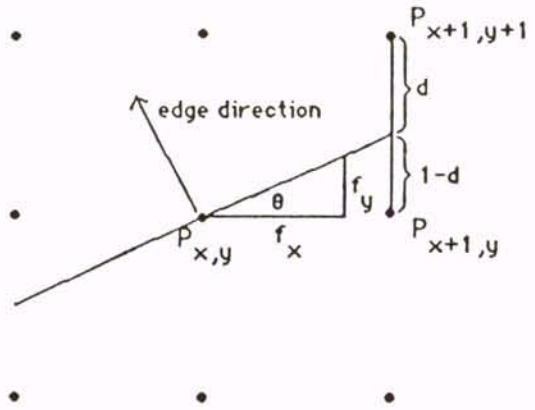
At the present, one of the best edge detectors is recognized to be that of Canny [7, 8]. In this method, the image is first convolved with a gradient of Gaussian operator. The two-dimensional Gaussian with standard deviation $\sigma$ is defined by the function

$$g_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2})$$

This can be separated into the product of two one-dimensional functions

$$g_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{x^2}{2\sigma^2})$$

The derivative is $g'_\sigma(x) = -\frac{x}{\sigma^2}g_\sigma(x)$. Convolving the image $f(x,y)$ in the horizontal direction with $g'_\sigma(x)$, then in the vertical direction with $g_\sigma(y)$ gives the partial derivative of $f$ with respect to $x$, $f_x$, while at the same time smoothing the image in both directions with the Gaussian. The degree of smoothing depends on the value of $\sigma$ used, larger values giving more smoothing. Similarly convolving with $g'_\sigma(y)$ in the vertical direction followed by $g_\sigma(x)$ in the horizontal direction gives the partial of




$$M_\theta(x,y) = d\,M(x+1,y) + (1-d)\,M(x+1,y+1)$$

$$M_{\theta+180}(x,y) = d\,M(x-1,y) + (1-d)\,M(x-1,y+1)$$

Figure 3: Non-maxima suppression.

$f$ with respect to $y$, $f_y$. Performing both of these operations produces the gradient vector $(f_x, f_y)$ introduced in the previous chapter.

The next step in the Canny edge detector is to perform non-maxima suppression. This is done by computing the direction of the gradient vector, $\theta$, at each point and interpolating between the values of the two eight-neighbors having direction nearest to $\theta$. The same thing is done in the $\theta + 180°$ direction. If the gradient magnitude at the point is not greater than both of the other values (call them $M_\theta(x,y)$ and $M_{\theta+180°}(x,y)$), then it is designated as a non-edge point. Otherwise it is designated as a possible edge point. See Figure 3. In this research it is assumed that the unit of measure is the (uniform) interpixel distance.

The third step is to scan the image for possible edge points. When one is found, the magnitude is checked at that point. If it is above a user defined high threshold, it is marked as an edge point and a search is initiated to all eight neighbors. If the neighbor is a possible edge point and has magnitude above a low threshold, then the

point is marked as an edge point and the search continues among all neighbors. If a point is not a possible edge (marked non-edge in the non-maxima suppression step or has already been marked as an edge) or is below the low threshold, the search in that direction stops. This produces a set of edge points. The use of two thresholds introduces a hysteresis effect and helps avoid gaps in edge contours.

## 2.2 Non-Maxima Suppression

Non-maxima suppression was first introduced by Rosenfeld and Thurston [37]; they called it *sharpening*. The concept was further developed and was called non-maxima suppression by Rosenfeld and Kak [36]. This is a method of identifying the points in a gradient image where the intensity is a local maximum. A number of variations are possible, all causing anomalies under certain circumstances. Most methods divide the gradient angles into four directions: horizontal, vertical, and the two diagonal directions. Then neighbors in the 3 by 3 neighborhood of a point in the direction of the gradient are examined. Nevatia and Babu [31], in their line detection algorithm, use what might be called a *neighbor* directed method. They examine the center point and the two neighbors in the gradient direction, and if the center point is larger than both neighbors, the center is marked as a possible edge and the neighbors are marked as not edge points. Also, to be an edge point the neighbors must have directions within $\pm 30°$ of the center pixel.

Most others (Canny [8], Schunck [38], Rosenfeld and Kak [36]) use a *self* directed method. In this method, if the center point is larger than the two neighbors, it is marked as an edge point; otherwise it is marked as non-edge. The other points are only marked when they are the center point being examined. The methods have the same effect if all three points being compared have the same gradient direction. However, differences arise when the center point and the two neighbors have different

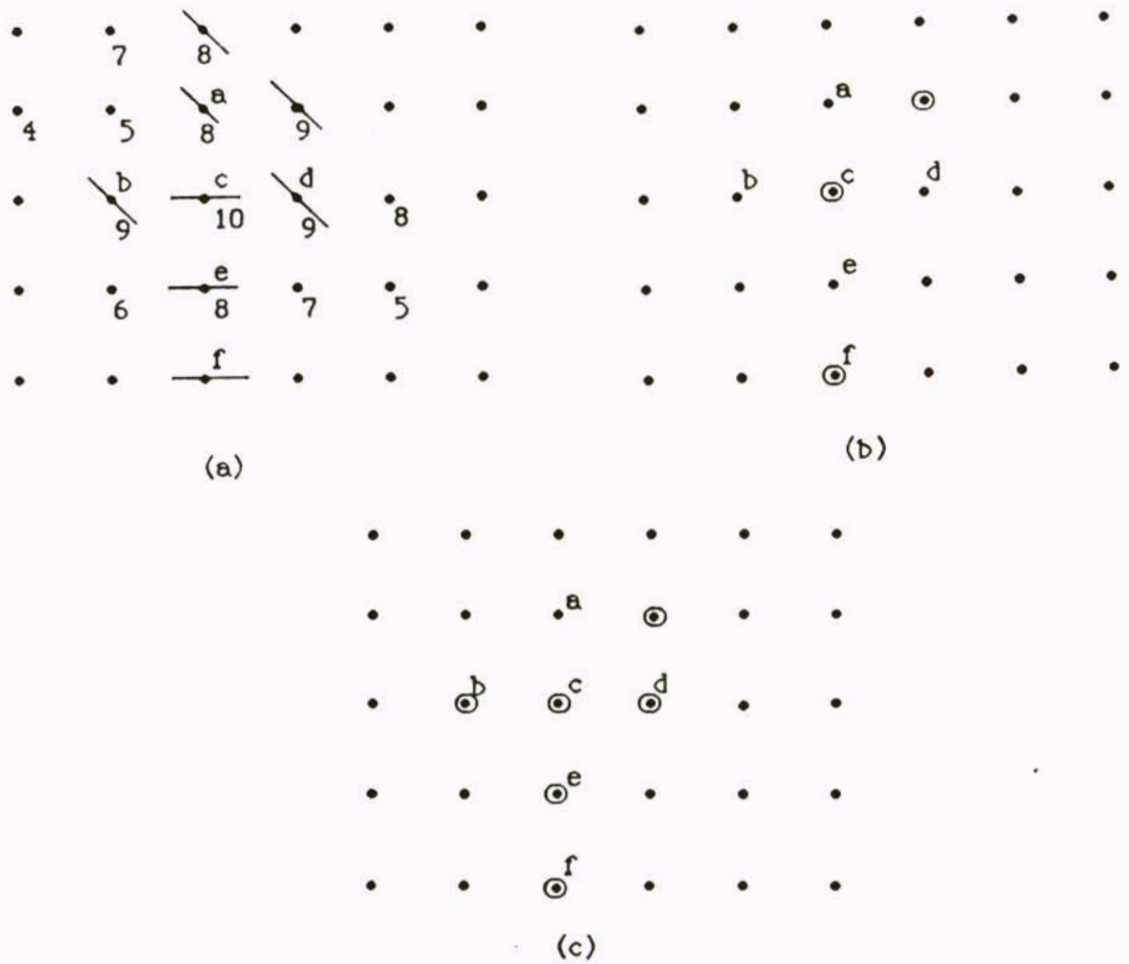



Figure 4: Self vs. neighbor directed non-maxima suppression. (a) Gradient magnitude and direction, (b) Neighbor directed, (c) Self directed.

directions. This is especially likely to happen when an edge is traversing the area at an angle near the boundary of the direction segments (e.g., 20-25°, halfway between horizontal and diagonal) or when an edge is curving. For an example of the differences, consider Figure 4. The lines through the points in Figure 4a indicate the direction of non-maxima suppression. The numbers near each point are magnitudes.

Using the neighbor directed suppression, $b$ causes $e$ to be marked non-edge, $c$ unmarks $b$ and $d$, and $d$ unmarks $a$, leaving $c$ with no edge neighbors in horizontal

and vertical directions and causing a gap between $c$ and $f$ (Figure 4b). Nevatia and Babu were specifically interested in line detection, so this might be satisfactory if the difference were caused by a curve in the edge, but not if the directions were caused by a line having slope near a direction segment boundary. Using the self directed suppression, on the other hand, would mark points $e$, $b$, $c$, and $d$ as edges (Figure 4c). Thus, the one method leaves a gap in the edge segment, while the latter provides an edge that is more than one pixel wide. Schunck [38] states that non-maxima suppression produces a ridge which is only one pixel wide. It can be seen from the examples given that this will be true only when the ridge through the region is near the direction of non-maxima suppression (horizontal, vertical or diagonal), and when the ridge is not curving.

As described in the previous section, Canny [8] does not reduce the gradient direction to the one of the four primary directions nearest the gradient direction. Rather, he interpolates between the two neighbors in the directions nearest the gradient direction to get a value with which to compare the center point (see Figure 3). He states that this gives better results than comparing directly with the points nearest to the gradient direction, but does not explain what the improvement is. To see the type of difference this would make, consider Figure 5. Using interpolation, point $a$ would be compared to the interpolated value $x_1$ and $b$ would would be compared to $x_2$. If interpolation were not used, $a$ and $b$ would be directly compared and one of them would be eliminated; however, if $x_1 < b$ and $x_2 < a$ it is possible that $a$ and $b$ would both be marked as edges. Since the interpolated value will always be smaller than the larger of the two values being interpolated, using this method will more often cause extra edge points to be included. In the example given, both $a$ and $b$ are eight-neighbors of $c$ and $d$, so marking just one would not cause a gap in the edge segment; however, there could be cases where it would.

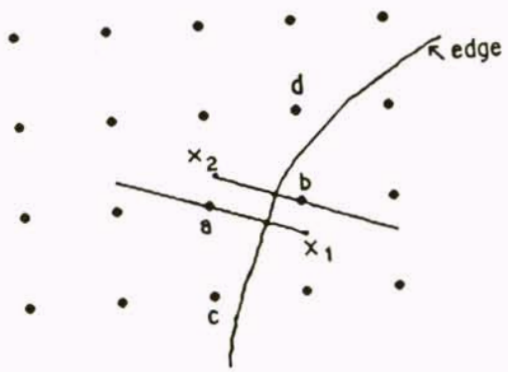



Figure 5: Non-maxima suppression using interpolation.

Another point relates to the method of comparison. If the magnitude is represented as integer values to decrease computation time, there is a possibility that the top of the ridge in the gradient values could have two equal values. This might happen if the edge were midway between the two edge points, or if the edge were more of a ramp than a step. Then, if the suppression marked only points strictly greater than their two neighbors in the gradient direction, both of two equal points would be marked as non-edge, leaving a gap in the edge, even though there is a maximum in the gradient. On the other hand, if a point were marked as an edge when equal to one neighbor and greater than the other, points on the shoulder of a ridge might be marked as edge points, introducing more noise. In Figure 6, showing a graph of the gradient magnitude along a one-dimensional slice across a ridge in the direction of greatest gradient magnitude, points $a$ and $b$ appear the same if only their nearest neighbors are compared. This indicates that in the case of a point being equal to its neighbor, the search should be extended in that direction until a non-equal neighbor is found.

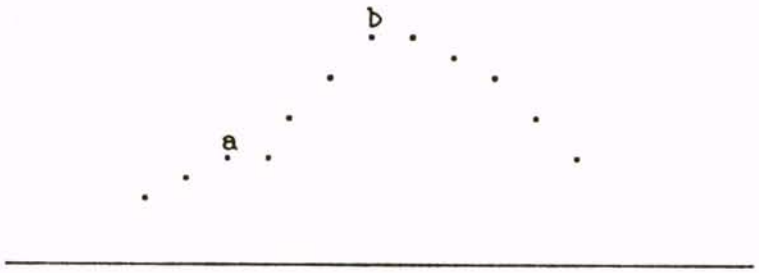



Figure 6: Gradient magnitude - points $a$ and $b$ appear the same.

Schunck [38] recommends non-maxima suppression over larger neighborhoods for another reason. At each point he first multiplies the gradient values computed at different scales (degrees of smoothing). This increases the strength of strong edges and greatly reduces that of weak edges nearby. By performing non-maxima suppression over a larger neighborhood, weak edges near stronger edges are removed. Comparing over a larger neighborhood introduces the question of which points will be compared with the center point. This is because neighborhoods larger than 3 by 3 contain points which do not fall on lines in the four primary directions. One possibility would be to choose smaller intervals for the comparison directions. For example, in a 5 by 5 neighborhood the comparison could be done in directions differing by approximately 22.5° rather than 45° as is done in 3 by 3 neighborhoods. This would introduce some anomalies. For example, in Figure 7 angle A is not the same as angle B, so the size of the sectors for comparison direction would not be the same in all directions. Another is that in the vertical, horizontal and diagonal directions, there are two points in each direction to compare to the central point, while in the other directions there is only

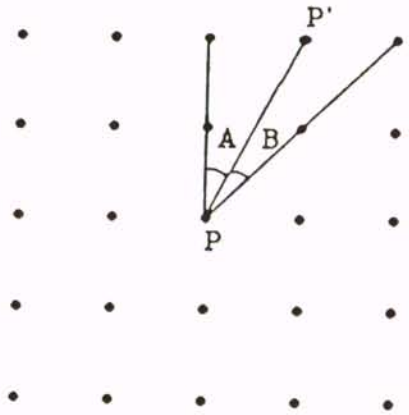



Figure 7: Non-maxima suppression over larger than 3 by 3 neighborhoods.

one point. If the decision is made to compare in only the four original directions, then what about the magnitude of point P'? Its value will never be compared with the value of P. As the neighborhoods for suppression become larger, there will be more of these points to consider. Schunck's approach is to compare only in the four primary directions, apparently assuming that if there is a ridge higher than the center point in the sector, it will be detected in one of the four directions examined.

## 2.3 Edge Linking

Montanari [29] presented a technique for using dynamic programming to recognize a line embedded in a noisy picture. If the length of the line and its characteristics such as curvature and location of corners are known, the entire image is searched to find the best instance of that line appearing in the image. Because of very large time and storage requirements, this method could be used only for small images, for example a small region where it was known that a character was written.

Martelli [28] produced a set of points linked into edge segments. His approach organized the points into a graph and found a best path through the graph. Arcs in

the graph connected a node to its nearest neighbor in the direction of the edge and to the two nearest neighbors on either side of that point. The node was given its gradient magnitude as a weight. The decision could be made to not include a node in the graph if its weight was below a threshold. Similarly, an arc could be omitted if the directions of the two nodes it joined differed by more than 90°. A positive cost was assigned to each arc based on magnitude of nodes and direction agreement of the two nodes joined by the arc. He then applied Nilsson's heuristic search algorithm (A*) [32, 33]. This can be performed if some estimate of the cost of a path joining two points is available. If $x_A$ and $x_B$ are the first and last points, let $f(x_i) = s(x_i) + g(x_i)$, where $s(x_i)$ is the cost of the path from $x_A$ to $x_i$ and $g(x_i)$ is the estimated cost from $x_i$ to $x_B$. The algorithm proceeds as follows:

1. Put the successors of the start node in list $S$ with pointers to $x_A$.

2. Remove the node $x_i$ with minimum $f$. If $x_i = x_B$ STOP.

3. Put successors of $x_i$ in $S$ with pointers to $x_i$.

4. Return to step 2.

The information to estimate $g(x_i)$ comes from some knowledge about the edges being sought. If there is no way to make an estimate, $g(x_i) = 0$ and the algorithm is equivalent to Dijkstra's algorithm for a path through a graph.

Ashkar and Modestino [2] implemented a variation of the heuristic search using edge strength, curvature, proximity to a known approximate curve, and distance from the goal to assign weights. They were working with medical images (e.g., lung x-rays) so they had information about approximate shape and length of curves sought.

Fischler and Wolf [14] detected lines, although the method could easily be adapted to edges. They assigned a local and a global weight to intensity maxima and linked

the points into a graph based on proximity. They then found a minimum spanning tree through the points. Short branches were pruned and longest paths through the tree marked. No details on how the arc weights were assigned was given.

Nevatia and Babu [31] first applied directional 5×5 gradient masks, then performed non-maxima suppression to identify edge points. The direction of a point was compared to those of its neighbors to determine its successors and predecessors in an edge. Then, beginning with points having no predecessors, the points were linked. When a point had two successors, the primary one, based on proximity and magnitude, was linked first. The secondary successor was considered the beginning of a separate edge segment. The choice of which points to link was a local decision, being based only on directions and edge characteristics of a point's neighbors.

## 2.4 Use of Multiple Scales

If points where there is rapid intensity change are to be detected, more than one point must be examined. An operator that examines a small neighborhood may not return the same value as one that examines a large neighborhood. The problem thus arises of what size neighborhood, or scale, to use. Small scales respond more to noise and fine texture, while larger scales may use neighborhoods that extend over more than one edge or average out weak edges.

Rosenfeld and Thurston [37] were among the first to suggest the use of multiple scales in edge detection. Their method compares average intensities in pairs of nonoverlapping neighborhoods meeting at a point. The orientation of the pairs determines the direction of the edge being detected. They suggest using a number of different sized neighborhoods, ranging in powers of 2 from 2 to a "size comparable to that of the entire picture." The edges would then be "sharpened" by suppressing the value at any point which had a larger value appearing within half the diameter of

the neighborhood in the direction being examined. The scale used at a point would be the largest such that the largest value found in the sharpening would not be significantly larger at the next smaller scale. This corresponds to the size of the largest neighborhoods that lie inside a uniform region on both sides of the edge. In order to detect edges at different angles, pairs of neighborhoods having different orientations would also be compared at each point. They recommended a parallel implementation and gave an O(log n) algorithm for computing the set of neighborhood averages for square neighborhoods.

Marr and Hildreth [26] were motivated by psychophysical findings in the late seventies that there were four channels, or scales, for detecting edges in the human eye. In determining the best operator to use, they wished to reduce the frequency range of intensity changes, while at the same time having an operator with the smallest localization possible in order to avoid interactions between nearby edges. Leipnik [21] had shown that the Gaussian is the only function to minimize both of these conflicting properties. They also desired to reduce the computation involved in applying multiple operators at different orientations. Thus they used the isotropic Laplacian of Gaussian ($\nabla^2 G = G_{xx}^2 + G_{yy}^2$) operator and detected zero-crossings in its output to identify maxima in intensity changes. In order to combine the edge information detected at different scales, they developed the spatial coincidence assumption. This states that a zero-crossing segment that is present in a contiguous set of different scales is due to a single physical phenomenon.

Eklundh, Elfving and Nyberg [12] extended the Marr-Hildreth method. They used three different scales and considered the points where there were zero-crossings at all three scales. They then compared magnitude and direction at the three scales and suppressed the point if the differences exceeded a threshold. In the output from a single scale, they also compared nearby zero-crossing points and tested the differences

in magnitude and direction against a threshold. These steps produced gaps in the contours, so they next detected the ends of contours and linked the most likely ones based on the distance between end points and the differences between the angles formed by the curves near their ends. The problem with this method was determining the several thresholds needed.

Witkin [44] took a different approach. Instead of detecting zero-crossings of the $\nabla^2 G$ at several discrete scales, he plotted the location of the zero-crossings against $\sigma$ as it varied continuously from zero to a value where all significant features had been smoothed out. He called this graph scale space. The graph contained a set of arches closed at the top and open at the bottom and a few lines that did not form arches (see Figure 8a). Some arches are nested inside others. It is theoretically possible for arches to intersect [18, 39], but because the situations where that would happen are unstable, in practice the phenomenon is not observed [9]. Witkin defined the scale of an event to be the apex of the arch corresponding to its contour. The location where the contour crossed the $x$-axis was considered to be its location in the image. From this graph he constructed what he called an interval tree. Vertical lines were drawn at the location of the events occurring at the largest scale. A horizontal line was drawn at that value of $\sigma$. Decreasing $\sigma$ in each interval, whenever the apex of an arch was reached a horizontal line was drawn across the interval in which it occurred and vertical lines were drawn from the horizontal line to the intersections of the bases of the arches with the $x$-axis (Figure 8b). Then the vertical intervals which had the longest undivided extent were considered the most significant features.

Bischof and Caelli [6] used an approach which is a hybrid of the Witkin and the Marr-Hildreth methods. Instead of considering the longest unsubdivided interval in the interval tree, they assigned a stability index to a point which indicates the longest vertical extent of a zero-crossing at the same location. This is like Marr-Hildreth in

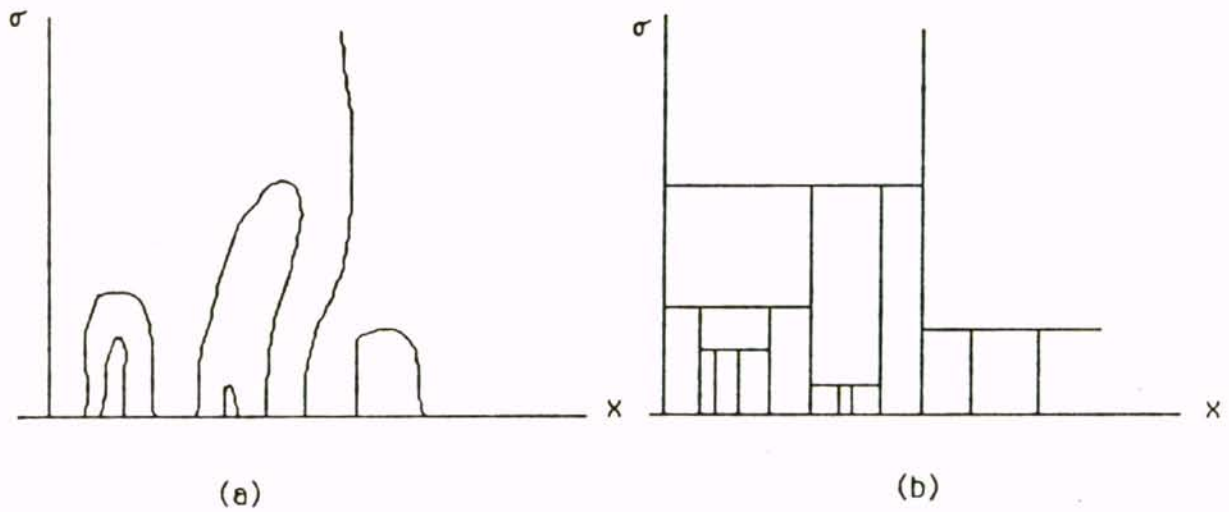



Figure 8: Witkin's (a) scale space and (b) interval tree.

that spatial coincidence is considered significant, but like Witkin in that a discrete approximation to continuous scale space was used, rather than just two to four scales as in Marr-Hildreth.

Schunck [38] combined multiple scales in yet a different manner. He used a gradient of Gaussian operator, rather than the $\nabla^2 G$ as the above have done. This was applied at two or more scales. Then the gradient magnitude at all scales at a point was multiplied to give a single magnitude array. He showed that if a weak edge is near a stronger edge, then the weak edge becomes much weaker in the combined array, with little effect on the stronger edge. Also the location of the stronger edge appears nearer its location at the smaller scale than that of the larger. This method removes much of the unwanted fine texture and noise from the edge map.

Canny [8] combined multiple scales using a technique he called feature synthesis. Beginning at the smallest scale, he marked edge points that were above a threshold based on the signal to noise ratio. Then he computed what the response of the edges detected would be at a higher scale. This synthesis step was performed by

convolving with a Gaussian normal to the edge direction computed from the current filter response. The standard deviation of this Gaussian is the same as that of the next larger filter to be examined. Then actually applying the larger scale filter, he compared the result to the predicted response. Points returning a significantly higher value than the predicted were marked as edge points. These correspond to "fuzzy" edges superimposed on sharper edges, or edges that are appearing at a higher scale. Other points giving a response near or below the expected were considered to be the same points detected at the smaller scale. The edges detected at the larger scale were mostly due to shading, shadows, and edges between textured regions.

Multiple scales were used by Bergholm [4] in a technique he called *edge focusing*. He applied a gradient of Gaussian edge operator at a coarse scale, and from this produced an edge map identifying the edge points at the finest scale that could be tracked from a coarse scale edge point. This removed the effects of delocalization and over smoothing that occurred at larger scales. However, for diffuse edges, e.g., shadows, the edge was replaced by noise and fine texture edges that were nearby.

## 2.5 Analysis of Scale Space

Shah, Sood and Jain [39] examined the scale space representation of idealized edge models, the staircase and the pulse. These are combinations of a unit step function defined by the equation

$$U(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{otherwise} \end{cases}$$

They showed that the convolution of this with the Gaussian is

$$u_\sigma(x) = \int_{-\infty}^{x} g_\sigma(u)du$$

and the second derivative is

$$u''_\sigma(x) = -(x/\sigma^2)g_\sigma(x)$$

The ramp edge having slope $m$ and width $w$ is defined by

$$R(x) = \begin{cases} 0 & \text{if } x < 0 \\ mx & \text{if } 0 < x < w \\ mw & \text{if } x > w \end{cases}$$

Convolving with the first and second derivatives of the Gaussian gives

$$r'_\sigma(x) = m\left[\int_{-\infty}^{x} g_\sigma(u)du - \int_{-\infty}^{x-w} g_\sigma(u)du\right]$$

$$r''_\sigma(x) = m[g_\sigma(x) - g_\sigma(x-w)]$$

The staircase and pulse are the sum and difference of two step functions at 0 and $w$,

$$S(x) = U(x) + bU(x-w)$$

$$P(x) = U(x) - bU(x-w)$$

Convolving with the second derivative of Gaussian gives

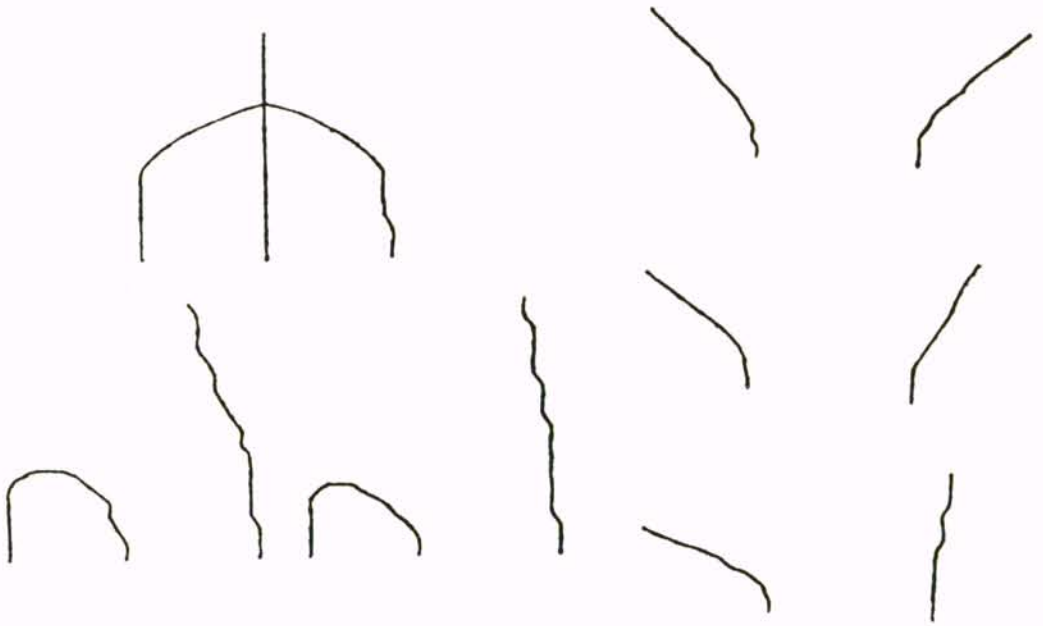
$$s''_\sigma(x) = -\frac{x}{\sigma^2}g_\sigma(x) - b\frac{x-w}{\sigma^2}g_\sigma(x-w)$$

$$p''_\sigma(x) = -\frac{x}{\sigma^2}g_\sigma(x) + b\frac{x-w}{\sigma^2}g_\sigma(x-w)$$

Plotting the zeros of the second derivatives gives the scale space images of the staircase and pulse. These are shown in Figure 9.

They also developed equations for a staircase and pulse in two dimensions and investigated their behavior. Peich [35] showed that the scale space of the two-dimensional staircase and pulse have a cross-section identical to that of the one-dimensional edge models.

Katz [18] also analyzed the pulse and staircase and showed that in the scale space image of the staircase with equal steps at $a$ and $-a$ the two side contours collapse into one when $\sigma = a$. In addition he showed that the contours of a pulse with equal steps at $a$ and $-a$ converge to $\sigma = \pm x$. He also discussed combinations of more than two steps.



(a) (b)

Figure 9: Scale space of (a) staircase with b = 1, 2, 4 and (b) pulse with b = 1, 2, 8.

Most edge detectors assume that the edge being detected is straight and the end does not fall within the support of the edge detection operator. Berzins [5] examined the behavior when these conditions are not met, for example if the line curves or has a corner. He examined the output of the Laplacian of Gaussian operator and showed that the edge contour goes through an isolated corner, but is rounded near the corner. He concluded that displacement at a corner is 0, and near the corner is less than $\sigma$ when the angle is greater than 15° for the Laplacian operator. See Figure 10. The edges of a square show similar distortion at the corners, but as the sides of the square become less than $4\sigma$, the detected edges move outward from the corner and the contour approaches a circle of radius $\sqrt{2}\sigma$. The detected edge of a large circle shows little displacement when the radius is above $4\sigma$, but as the radius becomes smaller, that is, the curvature is larger, the displacement approaches $\sqrt{2}\sigma$ as for the square.

Bergholm [4] also examined the behavior of corners and small closed curves in

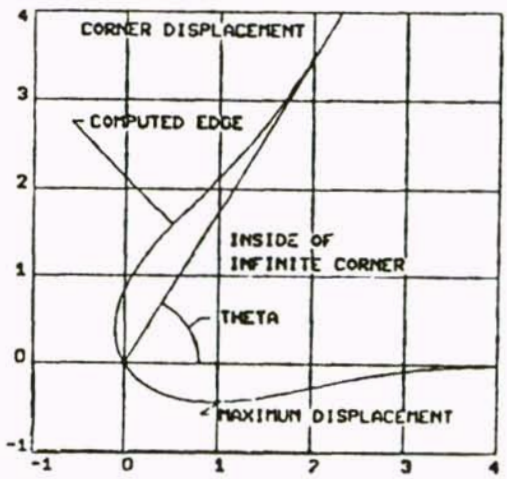



Figure 10: Displacement of edge near a corner.

scale space. He analyzed the gradient of Gaussian rather than the Laplacian. Under this operator, a corner becomes rounded inside the original corner. He showed that displacement of a corner is less than $2\sigma$ when the angle is greater than 22°. A small closed curve expands and assumes a circular shape, while the edges of a pulse composed of two *equal* step edges having opposite parity move apart. The rate of movement in all these cases is less than $\Delta\sigma$, the difference in the value of $\sigma$. He used this information to determine the size of the smallest neighborhood of an edge point detected at one scale of Gaussian that would be assured to contain that same edge at a different scale.

Clark [10] analyzed scale space using the perspective of catastrophe theory. The gradient of Gaussian is an example of a contrast function. He showed that when an image is smoothed with a contrast function, the point at which a gradient maxima and gradient minima meet and disappear is an example of a bifurcation. He analyzed the behavior of the contrast function and showed that at gradient maxima points the gradient value will decrease in value as scale increases, while the value at gradient

minima points will increase. This behavior can be used to classify the type of point detected. Korn [19] suggests using a two-dimensional gradient of Gaussian operator which has been normalized by multiplication with the factor $\sigma\sqrt{2\pi}$. This no longer has the properties of a contrast function and can be used to detect edge interaction.

## 2.6 Fitting Curves to Data Points

Edge detectors return sets of points that are affected by noise and delocalization. The next step in image analysis usually involves determining a higher level representation for these points. The simplest method for doing this is to use the least squares method to fit straight lines or curves to the points. Since only two points are required to fit a straight line, having a set of points containing more than two points produces an over-constrained system. The least squares procedure is designed to find a best solution to this system.

The Hough transform [3] can also be used to fit a line or other curves to a set of data points. For a given point $(\bar{x}, \bar{y})$ in the set, any line that passes through it must be of the form $\bar{y} = m\bar{x} + c$. Thus a single point comprises an under-constrained system, and more information must be used to find a solution. The equation above defines a line in the $m - c$ plane, $c = -\bar{x}m + \bar{y}$. Both the $m$ and the $c$ axes are subdivided into intervals and a two-dimensional array set up, representing each rectangular interval thus defined in the $m - c$ plane. The line is plotted by adding a one into the array location corresponding to each rectangle through which it passes. This is done for each point in the set. Then the rectangle having the highest count is considered to be the intersection of the most lines in the $m - c$ plane, and thus gives the best value of $m$ and $c$ for that set of data. Since values of $m$ between 0 and 1 represent a 45° range for the slope and 1 to $\infty$ also represents 45°, 0 to 1 must be divided into much smaller intervals than 1 to $\infty$ in order to get meaningful results. For this reason the

polar equation of a line is often used: $\bar{x}\cos\theta + \bar{y}\sin\theta = r$, and the $r,\theta$ graph, which gives a sinusoidal curve, is plotted. This method can also be extended to other curves and even irregular shapes.

Both of these methods have their shortcomings. For example, the Hough transform may indicate that a point far separated from a linear set belongs on a line because it accidentally lines up with the line. Both of these methods require that before fitting the curve a decision must be made as to what type of curve will best approximate the data points, e.g., a straight line, circle, cubic polynomial, etc. Unfortunately, in practice it is often not possible to decide on the type of curve beforehand.

Fischler and Bolles [13] developed a method of fitting lines or curves which started with a system which was neither over- nor under-constrained. This was done by choosing randomly a set of points just large enough to solve the system, for example two points for a line and three for a circle. Then the curve was fitted to the points and the number of points compatible with the curve was counted. If the set of compatible points was big enough, this was considered the correct curve. If the set was not large enough, another initial set of points was chosen and the curve refitted. This method involved thresholds for deciding which points were compatible, how big a set would be considered large enough, and how many times to try before giving up.

Duda and Hart [11] developed the recursive linear segmentation algorithm for fitting straight line segments to a set of data points. They started with a line joining the first and last points in the list. The set was segmented at the point which lay farthest from the line and the line was replaced by the two new line segments. The point farthest from the new lines was then chosen and its segment divided into two. This was repeated until all points were close enough to the curve. Lowe [23, 24] extended this method. The length of a line segment divided by the largest distance of a point from the line was computed as a scale independent significance measure for

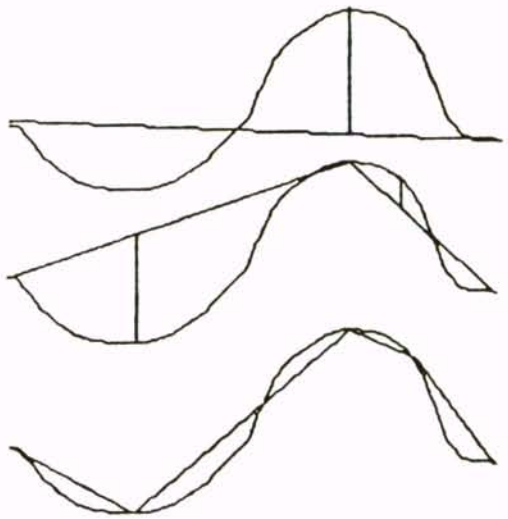


Figure 11: Subdivision of a set of data points into significant line segments.

the line. The set of points was then subdivided at the point lying the largest distance from the line as above, but the process was repeated until the line segments were no more than four pixels long. As he worked back up from shorter to longer lines, the significance of each of the shorter subsegments was compared to the complete segment. If either was higher than the significance of the complete segment, the shorter subsegments were returned, otherwise the single segment was returned. Both of these worked without having to know the values of tangents or curvature before the segmentation was done. See Figure 11. Lowe also suggested a method using the angular distance between the line connecting a point to the nearest end point and the line joining the endpoints as an error measure for a point, and computed a significance measure for each line segment based on the angular distance of all points from the line.

A different approach is to determine some characteristics which are desired in a curve, such as continuity and smoothness, then define a functional which will be

minimized when these conditions are met. The set of points giving the minimum value is determined by techniques of variational calculus.

Lee and Pavlidis [20] minimized the discrete functional

$$J(f) = \sum_{i=1}^{n} \alpha_i \left[ \frac{f_{i+1} - f_i}{x_{i+1} - x_i} - \frac{f_i - f_{i-1}}{x_i - x_{i-1}} \right]^2 + \beta \sum_{i=1}^{n} (y_i - f_i)^2$$

where $f_i = f(x_i)$. When the data points are equally spaced this reduces to

$$J(f) = \frac{1}{u^2} \sum_{i=1}^{n} \alpha_i [f_{i+1} - 2f_i + f_{i-1}]^2 + \beta \sum_{i=1}^{n} (y_i - f_i)^2$$

where $u = x_{i+1} - x_i$. The first term is a measure of curvature while the second measures the distance of the computed set of points from the original data points. To determine the values of $\alpha_i$ the solution was obtained with all $\alpha_i = 1$, then at each point $e_i = [f_{i+1} - 2f_i + f_{i-1}]^2 + (y_i - f_i)^2$ was computed. The value of this term will be high if the difference of the forward difference and backward difference (which approximates curvature) was large and/or if the point is far from the fitted curve. When this value was large, $\alpha_i$ at this point was set to 0 and the curve refitted until the largest value of $e_i$ was not much larger than the next larger $e_i$. If a single $\alpha$ is set to zero, a corner can develop at that point. If two adjacent $\alpha$'s, $\alpha_i$ and $\alpha_{i+1}$, are set to 0, this will produce a discontinuity in the curve.

Kass, Witkin, and Terzopoulos [17] minimized

$$E = \int_0^1 E_{int}(v(s)) + E_{ext}(v(s))ds$$

where $v(s) = (x(s), y(s))$. The internal energy term, $E_{int}$, enforces continuity and smoothness in the curve and is written

$$E_{int} = (\alpha |v_s(s)|^2 + \beta |v_{ss}(s)|^2)/2$$

$E_{ext}$ measures some image force, for example intensity or gradient magnitude. They solved the Euler equations of the system using an iterative procedure. Their approach

differed from that of Lee and Pavlidis in that the "error" term is not the distance of the curve from the original data points, but is a measure of the chosen image force. Thus the data points serve only to initialize the computation and the image forces determine the movement of the curve during the course of the iteration procedure which solves for the minimum.

There is no way to introduce *hard* constraints, such as minimum distance between points, in the iterative method used to solve the system of Kass, Witkin, and Terzopoulous. In order to allow their introduction, Amini, Tehrani, and Weymouth [1] propose solving the system using dynamic programming techniques. While this approach is much slower, it allows much more flexibility in controlling the course of the convergence to the minimum value.

Mackworth and Mokhtarian [25] have studied the properties of smoothing two-dimensional parametric curves with Gaussians. They determined the zeroes of curvature at different scales of smoothing and used these to obtain a scale space representation of a curve which they used to match a model curve being sought. However, there is a problem when smoothing parametric curves with a Gaussian; they shrink as the degree of smoothing increases. Lowe [22] presented a correction factor which can be applied as the curve is smoothed to remove the shrinking effect. He then segmented the curves at points where the rate of change of curvature was high.

# 3. EDGE LINKING

Edge point detectors identify potential edge points, those where the image intensity is changing rapidly. These typically return an edge map identifying the location of points where the intensity gradient is high, together with some gradient and direction information. In order to use this information in higher level processing, the next step is to identify those points which should be grouped together into edge segments. Several authors have developed algorithms for linking edge points into segments. None of the methods presented in Chapter 2 use more than one scale of smoothing. In addition, they are either not applicable to general images, or depend on local information only. Montanari [29] found a globally maximum line through an image using dynamic programming, but this used the entire image as a search space and is impractical to use because of the excessive execution time. In this chapter we present a method which chooses a best path based on global information through a restricted search space. Two extensions to this algorithm will be given that use multiple scales.

The single scale algorithm first uses a gradient of Gaussian operator to determine gradient magnitude and direction, followed by a non-maxima suppression step to identify ridges in the gradient map. This process gives an edge map identifying points whch are gradient maxima. Canny [8] has shown that this is a near optimal edge detector. He then uses a method involving two thresholds to identify a subset of these points as edge points. There are several problems with this set of points.

1. There is no structure. They are simply a set of unconnected edge points.

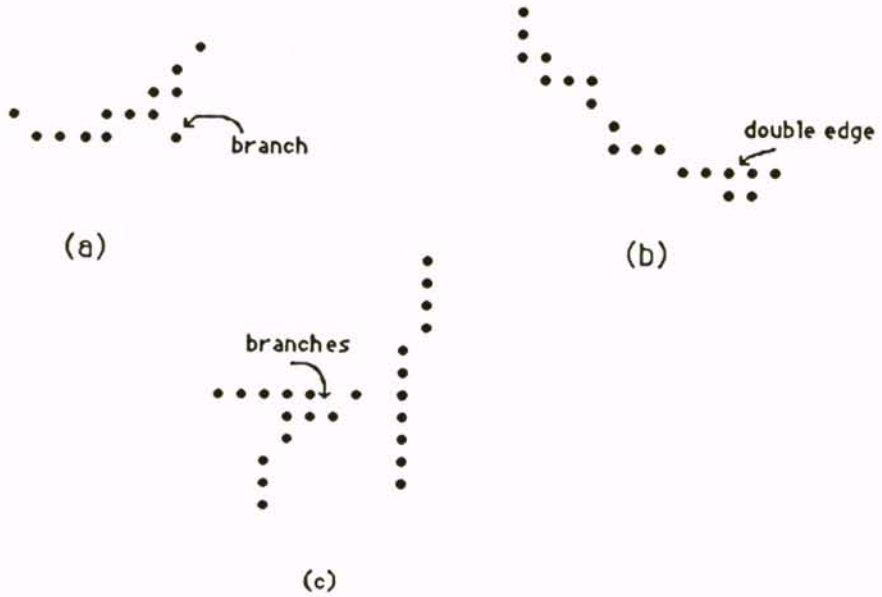



Figure 12: Edges having branches and double width.

2. The points appear as a path with side branches, some branches only one or two pixels long (see Figure 12a and 12c).

3. Even when there is a single, well defined edge, the edge may appear more than one pixel wide (see Figure 12b).

Note in Figure 12c that when the edge points form a T-junction, the non-maxima suppression will usually create a gap, so that the two contours are separated. This eliminates many major branches in the contours.

The points could easily be grouped into distinct segments by keeping a linked list of the edge points as they are marked. There would be a separate list for each connected set of edge points. However, this connected set would not be in sequential order along the edge, and additional processing would be needed to make it useful.

The second problem is caused by the algorithm used to mark the edge points. It searches in all directions from each point marked as an edge to find other points to mark. Thus it is possible to have branches on the path that have no points above the high threshold, but because they are branches off a path having points above the threshold they are kept. However, there is information available that is not being used by this algorithm, specifically *the direction of maximum gradient* at each point and the *magnitude* of that gradient. This information could be used, and a single best path through the set of points found. Any side branches that have points above the threshold would become separate edge point segments. Any branches having no points above the threshold would be considered as noise and eliminated. This would also solve the third problem, as a path only one pixel wide would be chosen through the points.

Grouping the points which belong to a single edge into a contour only one pixel wide gives a more meaningful data structure. Thus the algorithm presented in Section 3.1 thins the points to a contour one pixel wide and links them in sequential order. The algorithm assigns weights based on several factors, and then chooses the set of points giving the largest average weight. The weight at each point is computed using local direction and magnitude information and a global length weight. The set of points having the highest average weight is chosen, so global information is used here also. This is in contrast to Nevatia and Babu [31], who used only information in the $3 \times 3$ neighborhood of a point to determine to which point it should be linked.

This single scale edge linking algorithm is then extended to one which uses multiple scales to obtain better edges with little increase in the response to noise. The choice of the size neighborhood, or scale, to use in smoothing an image and determining the gradient is a difficult problem. When using the Gaussian function to smooth images, $\sigma$, the standard deviation of the Gaussian, is the scale parameter. Smaller scales

result in too much noise and fine texture being retained while larger scales result in delocalization of edges and loss of significant edge points. Background on the use of multiple scales in edge detection to reduce the conflicting goals of noise suppression and accurate localization was discussed in Chapter 2.

Pseudo-code for the algorithms is given in Section 3.5.

## 3.1 Single Scale Edge Detection and Linking

In this section an algorithm is presented for linking gradient maxima points into an edge contour which is a single pixel wide and has no side branches. In this method, the image is first convolved with a gradient of Gaussian operator. Then non-maxima suppression is performed, producing a set of potential edge points. Since the algorithms presented here make no attempt to extend edge contours across a gap where there is no point marked as a potential edge, options are chosen in the non-maxima suppression that will provide the largest possible set of points. Specifically, when a point is marked as a potential edge, no neighbors are disqualified from being edge points. Also, points are retained that are greater than or equal to their neighbors. Interpolation is used in the first two algorithms, but was not possible in the third algorithm. Suppression is performed in a 3 by 3 neighborhood.

All of the methods in Chapter 2 except that of Nevatia and Babu [31] had a large search space, examining all points with gradient or intensity above some threshold. The algorithm presented here finds a single good path through points which are gradient maxima. This reduces the number of points that will be examined since the gradient maxima ridges are only one pixel wide in many cases, and only two or three pixels wide at the greatest, with occasional short side branches.

The set of potential edge points is placed in a priority queue with the edge point having largest magnitude on the top. Thus the strongest edge points will be extended

into contours first. This step is performed because in some cases the order in which the points are processed could make a difference in the results obtained. For example, if three edge segments meet at a point, two of them will probably be linked into a single contour, while the third branch will become a separate contour if it has a point above the threshold on it. Thus if the search begins on a noisy spur there will be one contour consisting of the spur and one strong branch, and another contour consisting of the other strong branch. The desired contour would link the two strong branches. The use of a priority queue tends to avoid this problem. The situation of three edges meeting at a point rarely occurs except in the case where one of the three segments is a noisy spur, because the non-maxima suppression step usually causes a gap between the main contour and a side branch.

The search for points to assign to a contour proceeds as follows. The first edge point that is not already on a contour is retrieved from the queue. Its gradient magnitude and direction, $\theta$, are determined. Then the direction is used to determine the direction, $\theta + 90°$, of the next edge point. This assumes that the edge will be at a right angle to the direction of greatest intensity change. The angle is converted into an integer 0 through 7, each representing a 45° range. Zero corresponds to an edge in the range $-22.5°$ to $22.5°$, with the subsequent integers going counterclockwise from 0. The point in the computed direction is examined first, then those in the adjacent directions on either side of it. Each branch is followed to the end and a weight assigned at each point based on four factors. The four factors are

1. Is the edge point being examined in the direction determined by the gradient, or in the direction next to it?

2. How much does the direction of the next edge point differ from that of the current edge point?

3. What is the magnitude of the gradient?

4. How long is the contour extending from this point?

The maximum weight assigned is forty, with each of the four factors contributing a maximum of 10 each. First, if the point being examined is in the direction pointed to by the previous point, the weight is 10. Otherwise it is 45° on one side or the other and the weight is 5. In Figure 13a the previous point, p, with its direction is on the left. The three points that are possible successors are on the right together with their weights for factor one. For the second factor, the direction of the current point, p′, is compared to the direction of the previous point, p. Ten is assigned for a difference of 0, 8 for a difference of 1, 0 for a difference of 2, -8 and -10 for a difference of 3 and 4 respectively. In Figure 13b the previous point with its direction is on the left. The current point, with weights for possible directions, is on the right. There is little penalty for a difference of 1 as this often occurs on curves and lines near the boundary of the direction regions (e.g., 20-25°). However, sharp corners are penalized. This factor is important especially at the end of a contour where the location of an edge is more inclined to drift. The third weight is based on gradient magnitude and is the ratio of the point's magnitude to the maximum magnitude in the image, multiplied by 10. The fourth factor is designed to penalize short edges and is equal to the length of the edge constructed from this point to the end, if the length is less than 10. Otherwise the weight is 10.

The weights are designed to favor the longest, strongest, straightest path. The search is organized as a post order tree traversal. That means that beginning with the root, each node's subtrees are examined, then the node. Thus when a node is being processed it can choose the subtree which represents the path having largest weight among its children, update the weights to include itself, and return that value

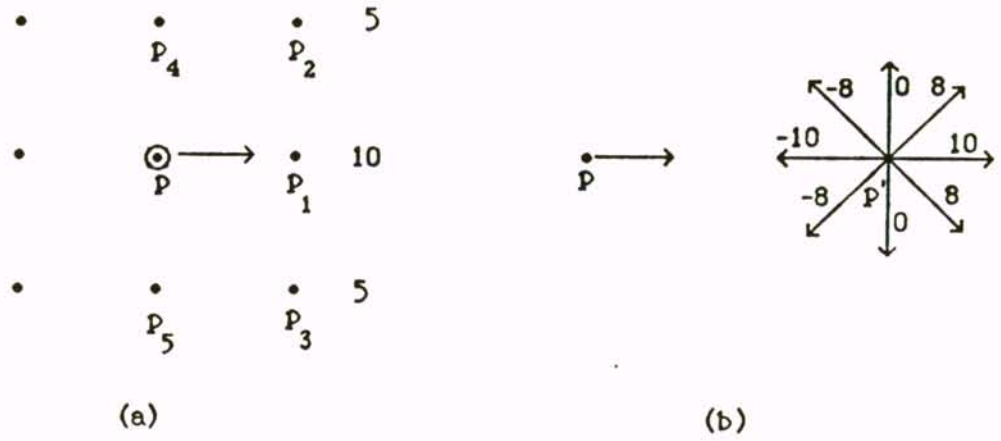



Figure 13: (a) Weights for factor one. Previous point, p, is on the left and possible successors, $p_1$, $p_2$, $p_3$, with their weights on the right. (b) Weights for factor two. Previous point, p, with its direction on the left and current point p′ on the right with the weights determined by comparing its direction with that of p. Ten points for a difference of 0; 8 for a difference of 1; 0, -8, -10 for differences of 2, 3, 4, respectively.

to its parent node. There are a maximum of three children examined for each node of the tree, but due to the non-maxima suppression step, many nodes will have only one child which is a potential edge, while most others will have only two. An occasional node has three. This reduces the search space to reasonable levels.

After searching from the initial point in one direction, a similar search is conducted in the opposite direction unless a closed contour has been formed. The two branches are combined to form one contour. Contours having three or fewer pixels are discarded. Then the next point is chosen from the queue, and the search continues for the next contour until there are no more edges in the queue having magnitude greater than a threshold expressed as a per cent of the maximum magnitude. Values in the range 5% to 10% gave good results on the images presented here.

For purposes of analyzing the complexity of the above algorithm, it can be considered as a tree search. If the tree searched has height n, a depth first algorithm

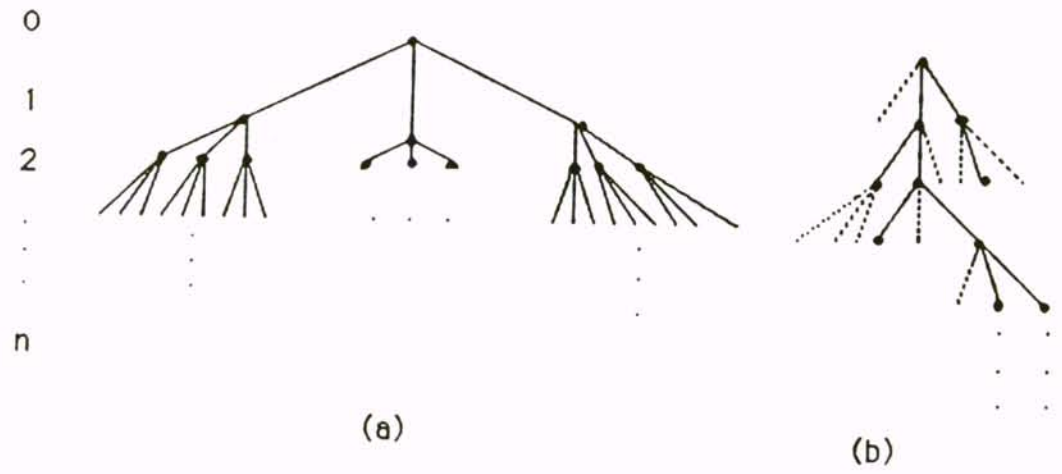



Figure 14: (a) Full tree, (b) Typical tree developed by algorithm.

is usually exponential. Let the height of the tree, which corresponds to the length of the contour being marked, be n. The tree has three branches at each node so the total number of nodes in the tree is $\frac{3^{n+1}-1}{2}$. Thus the complexity of the search is $O(3^n)$. However, the use of non-maxima suppression produces a ridge no more than two pixels wide at any point. Thus the total number of points is less than $2n$. Since no point is allowed to be on more than one branch of the path, the algorithm is actually linear (see Figure 14).

## 3.2 Multiple Scale Linking

This section deals with the extension of the algorithm presented in the previous section to an algorithm which uses multiple scales in order to produce improved detection of weak edges. It is known that at higher scales edges are delocalized. In our algorithm for multiple scales we need to know how large the delocalization is in order to determine the size of neighborhood of an edge point at one scale which must be searched to ensure finding the edge at another scale. A theoretical analysis

of the movement of idealized edges undergoing Gaussian smoothing is presented in Chapter 4.

The algorithm given in the previous section is extended to one using multiple scales, as follows. Initially the image is convolved with gradient masks at three scales: $\sigma, \sqrt{2}\sigma$, and $2\sigma$, where $\sigma$ is input by the user. Any number of scales could be used, but three was found to provide a significant improvement in contours. The fewer scales needed, the more efficient the process. The scale should be chosen so that the largest scale removes most of the unwanted noise without losing significant edges.

The search for a contour proceeds as for the single scale, using the largest scale, until a best partial contour at that scale has been found. Then the next finer scale is chosen and the neighborhood around the end point of the contour is examined to determine if there are potential edge points at the new scale having a direction similar to the end point of the contour. In this case a difference of two in the directions is considered close enough to continue the edge contour at the smaller scale. This is because the direction may change slightly at a different scale due to the fact that there is less interaction between edges at smaller scales. The neighborhood searched is only one pixel in each direction, based on the analysis given in Chapter 4. This analysis shows that the maximum delocalization of an edge point is $\sigma$ and is usually less than that. Thus an edge detected at one scale, $\sigma_1$, should appear no further away than $|\sigma_1 - \sigma_2|$ when the scale is $\sigma_2$. When the largest value of $\sigma$ is 4, most edges will be found in a neighborhood with radius one pixel, and when the largest value is 2, all will be in this neighborhood. The original algorithm is then followed for each of the points satisfying the above condition, and the best is chosen as an extension to the original edge. While extending the edge, if any point is discovered to be a potential edge point at a coarser scale, the search scale is increased to that value.

When the contour cannot be extended further, the scale is decreased to the next finer scale, and the process is repeated until the contour cannot be extended at the finest scale. The edge segment is then extended in the same manner at the other end. This algorithm resulted in a considerable improvement in the detection of some of the incomplete edge contours, with almost no degradation due to inclusion of noisy edge points.

### 3.3 Multiple Scale Non-maxima Suppression

When a contour at one scale ends and is continued at a finer scale, in the above algorithm, there is sometimes a jog in the contour due to the differences in delocalization at the two scales. This can be seen in the upper right corner of the object in Figure 15d. Thus an algorithm was sought which would remove the delocalization which occurs at larger scales, eliminating any jogs in the contour.

This algorithm combines the gradient information computed at several scales in the non-maxima suppression step rather than the linking step. Non-maxima suppression was performed in the usual manner for the coarsest scale and the potential edge points were marked. Then non-maxima suppression was performed at successively smaller scales. If a point was being marked as a gradient maximum and an adjacent point normal to the edge was a maximum at a coarser scale, but not at the present scale, then the label for the coarser scale was moved to the present point. This had the effect of shifting a delocalized edge point to its location at the finer scale. When performing the edge linking step, an additional weight was used, based on the number of scales at which a point had been detected. Thus an edge detected at three scales would have a larger weight than a point marked at only one or two scales. This is similar to the Marr-Hildreth spatial coincidence assumption; however, the marks for delocalized edges have been moved to their location at a finer scale increasing the effect of spatial coincidence.

## 3.4 Experimental Results

The results of the algorithms applied to several real 128 × 128 images is presented here. The values of $\sigma$ used were 1, $\sqrt{2}$, and 2 and a threshold of .08 was applied. For comparison, the images were also processed using the Canny operator. The threshold used for the Canny algorithm was chosen to return approximately the same number of contours as the edge linking algorithms. The results for three images, Part, Tiwanaku, and Bananasplit, are shown in Figures 15, 16, and 17. The original image is a, the Canny operator is b, the single scale algorithm is c, the multiple scale algorithm is d and using the multiple scale non-maxima suppression is e.

In the part image, there were several fairly well defined edges, but quite close to each other, so that at scales which were large enough to remove noise, the nearness of the edges had caused some of them to disappear. The main difference between the Canny operator and the single scale edge linking algorithm was that the double edges were replaced by single pixel edges, and some small spurs on the edges were eliminated. More improvement was achieved with the multiple scale algorithm. Looking at the fourth contour from the center, the multiple scale algorithm was able to join three partial contours and extend the right side into an almost complete contour. The third contour was also extended across the bottom of the image. Note also, in the shadow edges at the top and bottom of the image, that small fragmentary edge contours have been combined into longer, much more well defined contours. The contours using the multi-scale non-maxima suppression give even better results. The corner where the shadow edge joins the object is much clearer and the edges appear at their location at the finest scale. In this area of the image, the outer contour of the part has an edge of opposite parity to its left, thus they repel each other, pushing the object edge to the right. The shadow edge outside the part has the same parity as the object edge, thus pulling the edge to the right. With one edge pushing the contour to the right

while another pulls, the edge of the object has been moved outward in b through d. This delocalization has been removed in e.

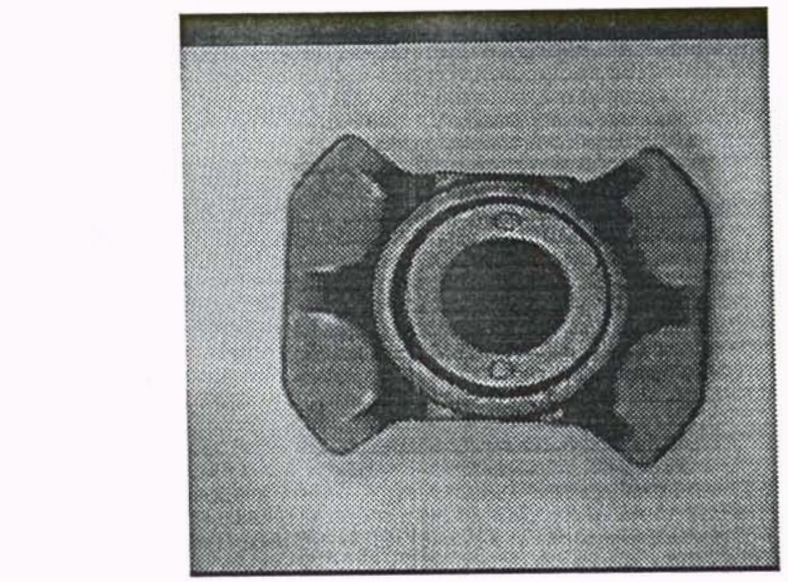
In the Tiwanaku image the single scale algorithm shows less noise than the Canny, probably due to the difference in the thresholds which were difficult to compare. Again, some spurs and double edges have been removed. The multiple scale algorithm produces slight extensions of the vertical lines on the large stone, and the bases of the smaller stones are more well defined. Some of the texture edges in the background and the clouds have been combined into longer contours as well. Again, the results in e are better. The corners are squarer, and the delocalization at the tops of the two small stones on the right has been removed, defining the edges more clearly. Also, the edge between the small stones has been much improved.

The Bananasplit image is different from the other two in that most edges fall into one of two categories. They are either very well defined, as in the sides of the post and the floor-wall joint that are well marked by all the algorithms, or they are very poorly defined, with few intermediate edges of the type most improved by the multiple scale algorithm. For example, the specularity on the left side of the stool interrupts the bottom edge and the gap cannot be detected at any scale. The horizontal lines on the stool are only one pixel wide, thus the edges are too close together to be detected by any of the methods. The multiple scale algorithm did complete one contour on the left. The base of the stool is much clearer in e, as well as the shape of the top.

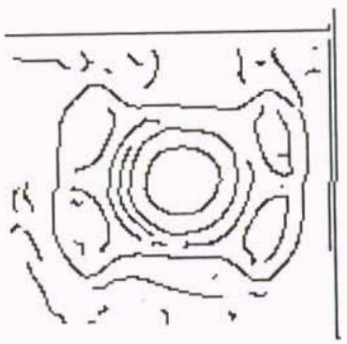
The conclusion is that the single scale edge linking algorithm cleans up the Canny edges and produces a set of linked lists corresponding to the contours found. In addition the multiple scale algorithm is able to improve detection of edges that are close together and interact at scales which are large enough to remove noise and fine texture. It also improves detection of weak, but well defined edges, such as those of the shadows in the Part image. Best results in all cases occurred with the multiple

scale non-maxima suppression algorithm. Edges which had been delocalized were moved back to their location at a smaller scale, separating edges which had become too close together to differentiate. Because of this, some contours were extended farther than with the multiple scale linking algorithm.

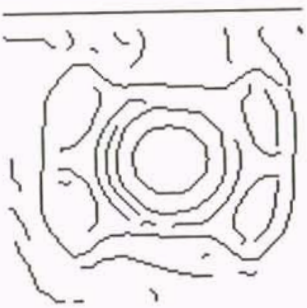
The weights were chosen heuristically. Experiments varying the weights indicated that the actual values did not seem to be extremely critical as long as higher weights were given to the points in the primary direction, having the same direction as the current point, high magnitude, and longer length contour. Experiments in which each one of the factors in turn was removed, however, indicated that no three gave as good results as using all four. This was interesting especially in relation to weights 1 and 2 which were both determined using direction information. The weight for factor 1 is higher when the point is noise free and the curvature is small, but noise seems to be the most important factor. Factor 2 has a higher weight when the curve has no sharp turns. The ends of the contours particularly drifted when this weight was removed. Removing the length factor allowed one or two strong points, perhaps lying next to the main edge on a ridge two pixels wide to be chosen, rather than a longer contour which extended into a weaker portion of the edge.



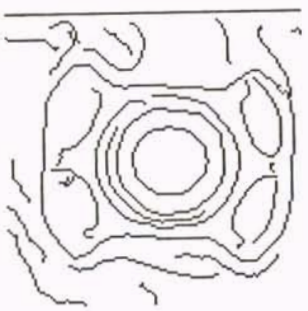
(a)



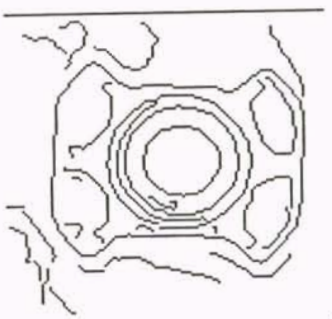
(b)



(c)

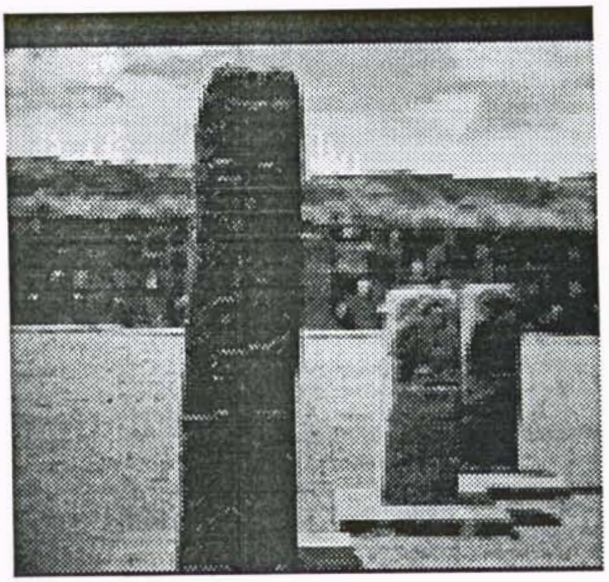


(d)

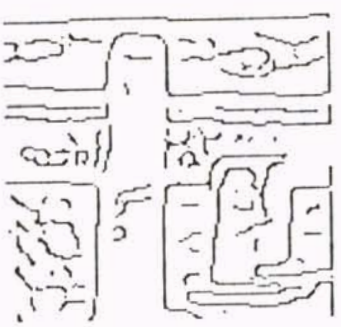


(e)

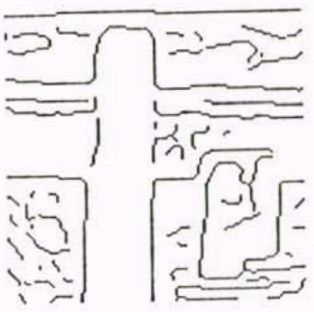
Figure 15: Part (a) Original Images, (b) Canny Operator, (c) Single Scale Edge Linking Algorithm, (d) Multiple Scale Algorithm, (e) Multiple Scale Non-maxima Suppression.



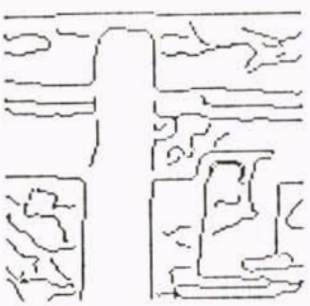
(a)



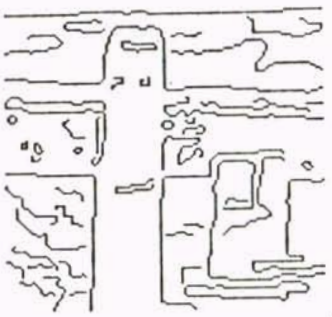
(b)



(c)

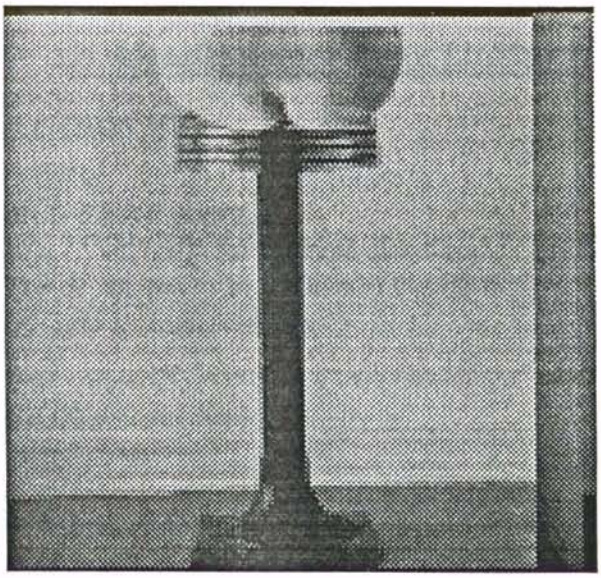


(d)

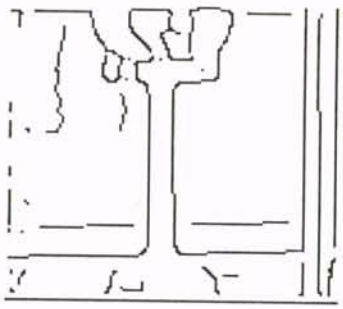


(e)

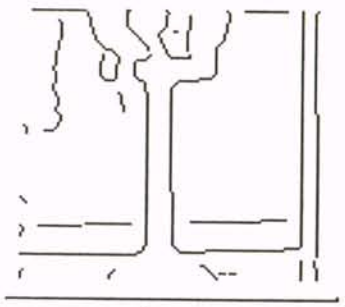
Figure 16: Tiwanaku (a) Original Images, (b) Canny Operator, (c) Single Scale Edge Linking Algorithm, (d) Multiple Scale Algorithm, (e) Multiple Scale Non-maxima Suppression.



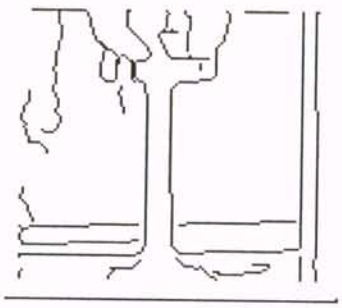
(a)



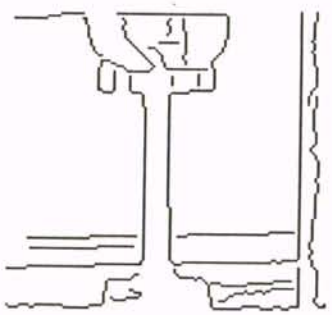
(b)



(c)



(d)



(e)

Figure 17: Bananasplit (a) Original Images, (b) Canny Operator, (c) Single Scale Edge Linking Algorithm, (d) Multiple Scale Algorithm, (e) Multiple Scale Non-maxima Suppression.

## 3.5 Algorithms for Edge Linking

**SINGLE SCALE ALGORITHM:**

procedure FIND CONTOUR

1. get next POSSIBLE EDGE point, $p$
2. FOLLOWEDGE($p, C, wt, len$) in forward direction
3. FOLLOWEDGE($p, D, wt, len$) in backward direction
4. COMBINE($C, D$) into a single contour

procedure FOLLOWEDGE($p, C, wt, len$)

input: $p$, point to begin contour
output: $C$, contour points
$wt$, average weight of $C$
$len$, number of pixels in $C$

1. **if not** POSSIBLE EDGE($p$) **then** $wt = len = 0$, **return**
2. **for** $i = 1, 2, 3$ **do** FOLLOWEDGE($p_i, C_i, wt_i, len_i$)
   /* see Figure 13a for numbering of points */
3. $max = i$ where $wt_{max}$ is the largest of $wt_i$
4. **if** $wt_{max} = 0$ **then** /* $p$ has no continuation */
   $wt = factor1\_wt + factor2\_wt + factor3\_wt + 1$
   $len = 1$
   $C$ = PUSH($\Phi, p$) /* Create a new contour $C$ containing $p$ */
   **return**
5. /* continuation found */
   $wt = (len_{max} * (wt_{max} - len_{max}) + factor1\_wt + factor2\_wt + factor3\_wt)/$
   $(len_{max} + 1) + factor4\_wt$
6. $len = len_{max} + 1$
7. $C = PUSH(C_{max}, p)$ . /* add $p$ to $C_{max}$ to give new contour $C$ */
8. **return**

**MULTIPLE SCALE ALGORITHM:**

procedure FIND CONTOUR

1. get next POSSIBLE EDGE point, $p$
2. FOLLOWEDGE($p, C, wt, len$) in forward direction
3. FOLLOWEDGE($p, D, wt, len$) in backward direction
4. **if not** CLOSED($C, D$) **then** EXTEND($C$)
5. **if not** CLOSED($C, D$) **then** EXTEND($D$)
6. COMBINE($C, D$) into a single contour

```
procedure FOLLOWEDGE(p, C, wt, len)
    input: p, point to begin contour
    output: C, contour points
            wt, average weight of C
           len, number of pixels in C
    1. if not POSSIBLE EDGE(p, scale)
            /* true if p is possible edge at scale or coarser. */
            scale = largest scale for which p is a POSSIBLE EDGE
            then wt = len = 0, return
    2. for i = 1 ... 5 do FOLLOWEDGE(p_i, C_i, wt_i, len_i)
            /* see Figure 13a for numbering of points */
    3. max = i where wt_max is the largest of wt_i
    4. if wt_max = 0 then   /* p has no continuation */
          wt = factor1_wt + factor2_wt + factor3_wt + 1
          len = 1
          C = PUSH(Φ, p) /* create new contour C containing p */
          return
    5. /* continuation found */
          wt = (len_max * (wt_max - len_max) + factor1_wt + factor2_wt + factor3_wt)/
            (len_max + 1) + factor4_wt
    6. len = len_max + 1
    7. C = PUSH(C_max, p)       /* add p to C_max to give new contour C */
    8. return

procedure EXTEND(C)
    1. while (true)
    2.   p = end point in contour C
    3.   scale = largest scale at which p is a POSSIBLE EDGE
    4.   if scale is smallest possible then break
    5.   scale = scale - 1   /* continue end at next smaller scale */
    6.   for i = 1, 2, 3, 4, 5 /* see Figure 13a for labeling of points */
           if POSSIBLE EDGE(p_i, scale) and directions of p and p_i differ by no more than 2
           then FOLLOWEDGE(p_i, C_i, wt_i, len_i)
    7.   max = i where wt_max is largest of wt_i   /* find best extension */
    8.   if wt_max > 0 then add contour C_max to contour C
              else break   /* no extension*/
    9. repeat
```

# 4. EDGE MOVEMENT

It is well known that smoothing an image with a Gaussian operator causes delocalization of edges. The larger the standard deviation of the Gaussian, the greater the delocalization. If multiple scales (values of $\sigma$) are used in the analysis of an image it is useful to know how far from its original position, or how far from its position in another scale, a zero-crossing appears. For example, when performing edge linking using multiple scales, as in the previous chapter, the edges may be at different locations for different scales. When extending an edge contour at one scale with points at a different scale it is necessary to know how far the points may have been displaced to determine the search diameter and ensure proper linking. Also when matching a 2-D model to an image, if the maximum possible movement at different scales is known, this information can be used to compute error bounds for the location of the object edges in the image.

In this chapter we analyze the movement of the zero-crossings of the second derivative for ideal edge pairs as they are smoothed with Gaussian operators having different standard deviations, $\sigma$. Adjacent edges in an image will have either the same parity, i.e., both of increasing or both of decreasing intensity; or opposite parity, one increasing and the other decreasing. The edge model chosen for this analysis was the step edge, and the two combinations examined are the staircase (adjacent steps having the same parity) and the pulse (opposite parity). The relative size of the steps of the two edges is allowed to be arbitrary. The distance between the edges in the pair also affects the characteristics of the movement. Thus the effect of changing this parameter will also be examined.

Shah *et al.*, [39] developed equations for these step pairs convolved with the Gaussian and its derivatives and showed the general shape of the scale space curves. That work is extended here to develop the equation of the scale space curves and analyze quantitatively the amount of the delocalization that occurs as images containing these steps are convolved with Gaussians having different values of $\sigma$. In some cases an edge location approaches a certain limiting position as $\sigma$ increases. The equations for these positions are also developed. Bergholm [4] examined a pulse having equal steps and showed that the speed with which two edges move apart is limited by $\Delta\sigma$. This work considers general pulses as well as staircases, and focuses on the maximum possible movement as a function of $\sigma$.

The movement of each of the two edges in the staircase and pulse models are analyzed as a function of the relative strengths of the two edges, the distance apart, and the degree of smoothing. It is determined that for the staircase model, the maximum movement of an edge occurs when the two edges of the staircase are $2\sigma$ apart and the intensity change for the two edges is equal. Movement decreases rapidly from this maximum when the distance between the edges is either larger or smaller than $2\sigma$ or the steps become unequal in size. For the pulse model the maximum movement occurs when the two edges have the same step size and are very close together. As with the staircase, the movement decreases rapidly for edges that are farther apart, and when edges are $4\sigma$ apart the movement is negligible.

## 4.1 Scale Space of Ideal Edges

The ideal step edge was chosen as the edge model in this analysis both because of its simple form, and because of its use by previous authors. In addition, because the step edge is the most extreme edge, its movement gives an upper bound on the movement of edges which are more nearly ramp shaped. This is demonstrated in

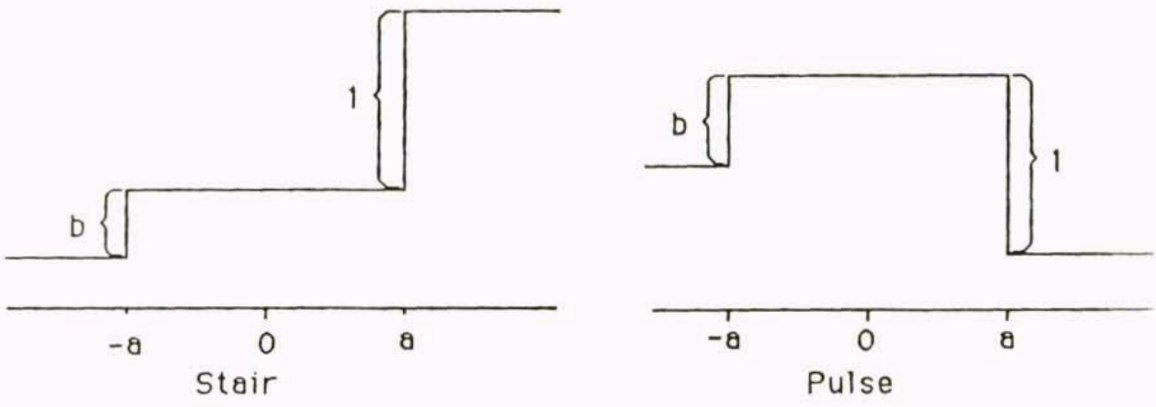



Figure 18: Staircase and Pulse Edges.

Table 2 of Chapter 5. Korn [19] showed that given a ramp smoothed with a Gaussian it is possible to find a Gaussian having a different standard deviation, $\sigma$, that will produce the same result for another ramp having a different slope (assuming total contrast is the same). The ramp with the larger slope will have a larger value of $\sigma$, hence greater movement. The roof edge, which is two adjacent ramp edges of opposite parity, was analyzed and the results compared to the pulse. The characteristics of the movement were very much like that of the pulse; however, the equations for the pulse yielded an equation for the scale space. This was not possible with the roof.

The analysis will be performed for the one-dimensional case although images are two dimensional. This makes the computation much more straightforward and can be justified by the work of Peich [35] who showed that with a suitable change of coordinate system, the perpendicular cross section of a pulse or staircase in two dimensions is identical to the one-dimensional models. This result is valid when two edges in the same neighborhood are parallel. The unit step edge is represented by the equation

$$U(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{otherwise} \end{cases}$$

Then the staircase with edges located at $-a$, and $a$ is represented by

$$S_a(x) = bU(x+a) + U(x-a)$$

while the pulse with edges at $-a$, and $a$ is represented by

$$P_a(x) = bU(x+a) - U(x-a)$$

See Figure 18.

The one-dimensional Gaussian having standard deviation $\sigma$ is defined by the function

$$g_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{2\sigma^2})$$

The first derivative is

$$g'_\sigma(x) = -\frac{x}{\sigma^2} g_\sigma(x)$$

The convolution of the derivative of the Gaussian with the staircase is

$$\int_{-\infty}^{\infty} g'_\sigma(u) S_a(x-u) du$$

Since $U(x-u+a) = 0$ when $u > x+a$ and $U(x-u-a) = 0$ when $u > x-a$ this is equal to

$$b\int_{-\infty}^{x+a} g'_\sigma(u) du + \int_{-\infty}^{x-a} g'_\sigma(u) du$$

thus

$$s'_{\sigma,a}(x) = bg_\sigma(x+a) + g_\sigma(x-a) \tag{1}$$

The equation for the convolution with the second derivative is

$$s''_{\sigma,a}(x) = bg'_\sigma(x+a) + g'_\sigma(x-a)$$

Similarly, the equation of the pulse convolved with the derivative of the Gaussian is

$$p'_{\sigma,a}(x) = bg_\sigma(x+a) - g_\sigma(x-a)$$

and convolved with the second derivative is

$$p''_{\sigma,a}(x) = bg'_\sigma(x+a) - g'_\sigma(x-a)$$

Edge points are those points where the magnitude of the convolution of the image function with the first derivative of the Gaussian has a maximum value. These are the points where there is a positive maximum or negative minimum. They can be found by determining where the convolution of the function with the second derivative has zero values. Since images are discrete functions, the zero value may not fall on a grid point. Thus, a point near the zero where the value changes sign will be called a zero-crossing. However, positive minima and negative maxima will also give zero-crossings. These points are where the intensity change is smallest. These are the points referred to as *phantom* edges. When they occur in the analysis, their presence will be pointed out.

It will be assumed that $b$, which represents the relative heights of the two steps, satisfies $0 < b \leq 1$. Thus we are considering the weaker edge to be at $x = -a$. (Similar results could be developed for the weaker edge at $x = a$.) The equation of the scale space, which plots $\sigma$ as a function of $x$ at the zero-crossings, for the staircase is

$$\sigma = \left[\frac{2ax}{\ln(b(a+x)/(a-x))}\right]^{1/2}$$

for $-a < x < 0$ and $a(1-b)/(1+b) < x < a$. The equation of the scale space for the pulse is

$$\sigma = \left[\frac{2ax}{\ln(b(x+a)/(x-a))}\right]^{1/2}$$

for $-\infty < x < -a$ and $a < x < a(1+b)/(1-b)$. The derivation of these equations can be found in the appendix. The graphs for several values of $b$ are given in Figures 19 and 20. These will be referred to as the scale space images. For a more complete discussion on the scale space images of pulse and staircase edge pairs see [18, 39].

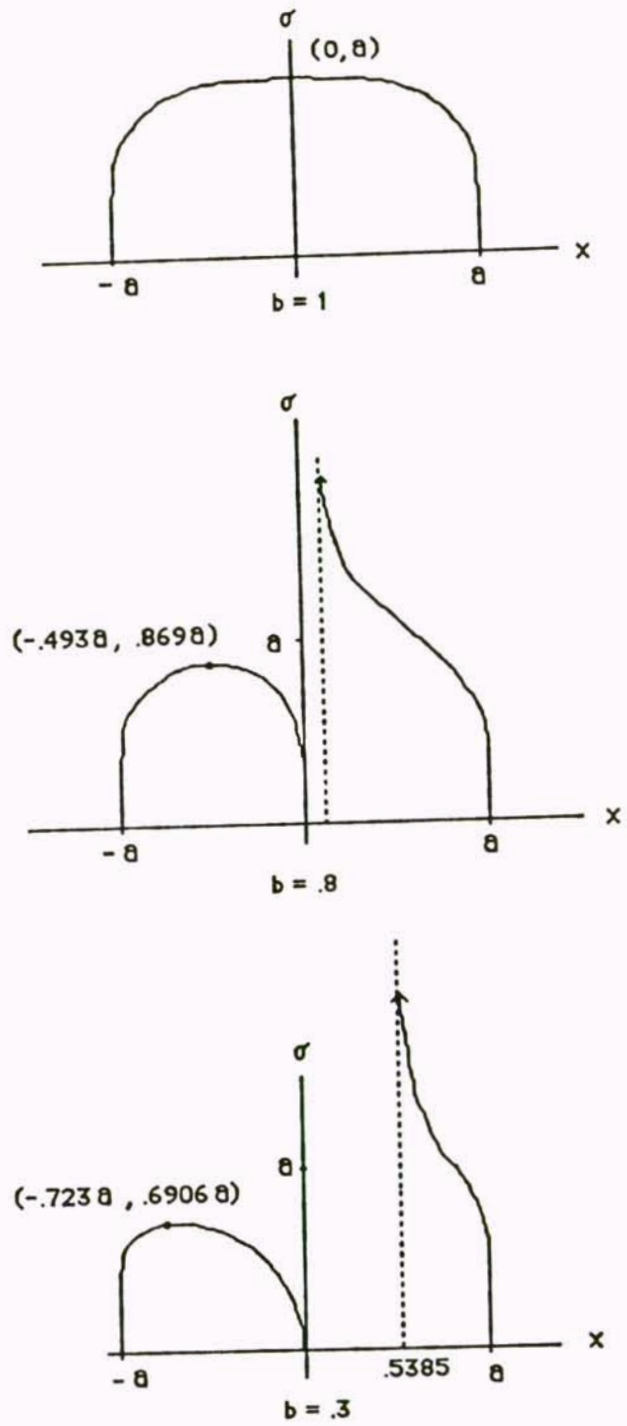



Figure 19: Scale Space Image: Location of zero-crossings for the staircase when (a) b = 1, (b) b = .8, (c) b = .3.

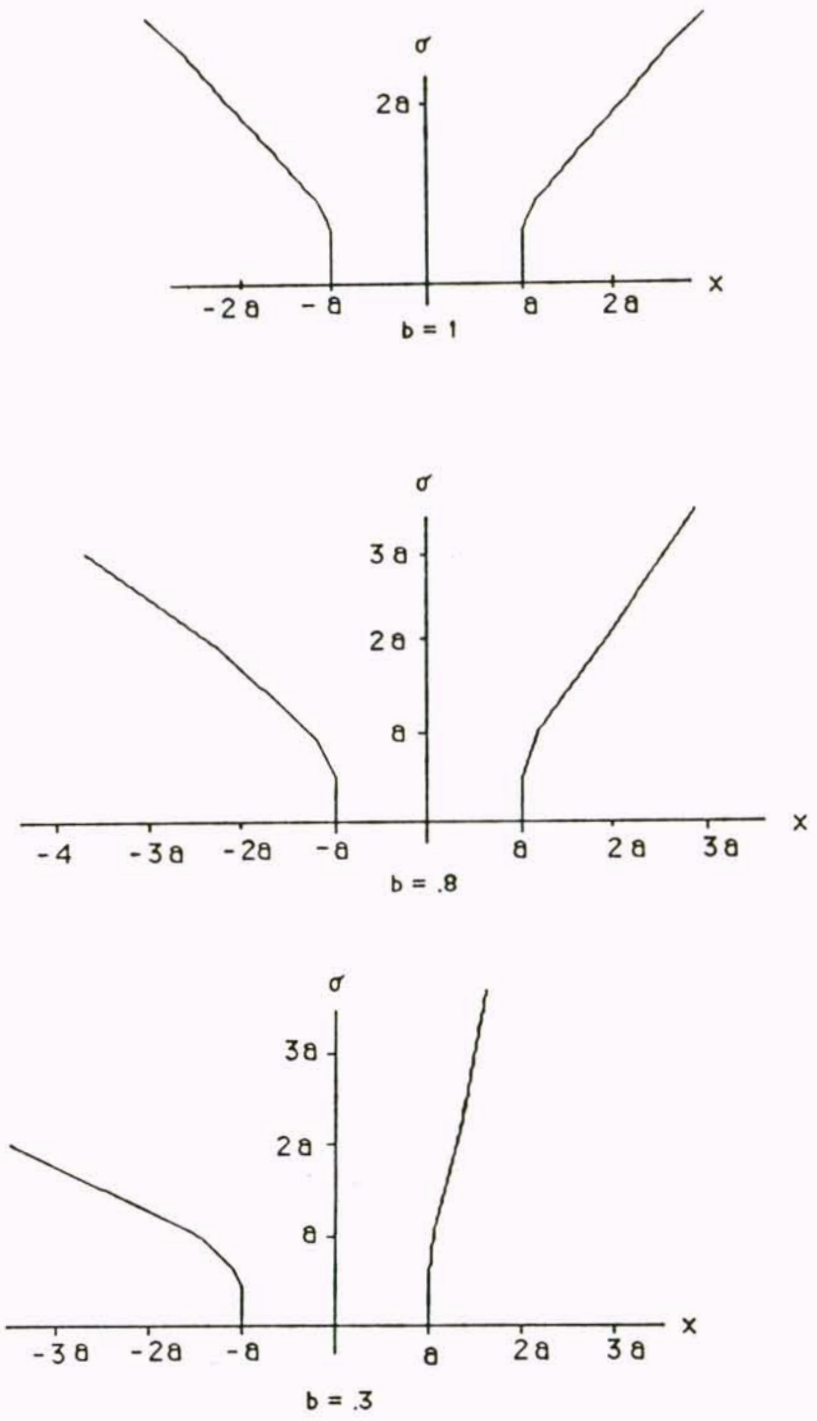



Figure 20: Scale Space Image: Location of zero-crossings for the pulse when (a) b = 1, (b) b = .8, (c) b = .3.

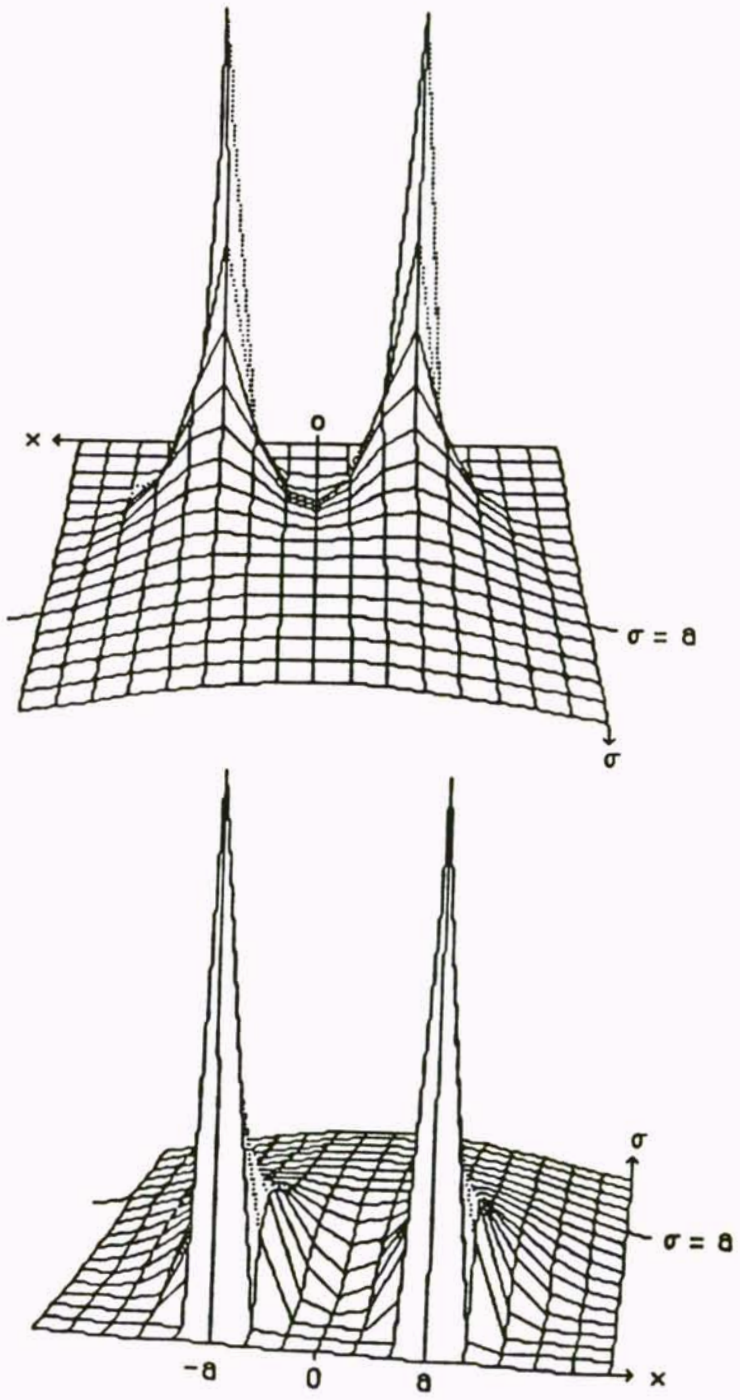



Figure 21: 3-D plot of gradient magnitude of staircase, $s'_{\sigma,a}(x)$, when $b = 1$.

## 4.2 Movement of Ideal Edges

Having the scale space curves which give the location of the edges at different scales, it is now possible to determine how far an edge has moved for a particular value of $\sigma$. First we will consider the staircase. Notice that if $b = 1$ the two edges at $a$ and $-a$ move together as $\sigma$ increases until they meet when $\sigma = a$. For values of $\sigma$ larger than $a$, only one edge exists at $x = 0$. There is also a zero-crossing at $x = 0$ for $\sigma < a$ which corresponds to an inflection point in the smoothed staircase; Clark [9] calls this type of zero-crossing a *phantom edge*. Thus the scale space curve has a pitchfork shape. For $\sigma < a$ there are two edges separated by a phantom edge. These three combine when $\sigma = a$ and there is only the one edge for larger values of $\sigma$. Clark has shown that when there is an arch in a scale space curve, an actual edge and a phantom edge meet at the top and the pair do not appear for larger values of $\sigma$. In Figure 21 a 3-D plot of Equation 1, the gradient of the staircase, clearly shows the pitchfork form of Figure 19a. The two ridges corresponding to the edges and the valley corresponding to the phantom edge move closer together and combine at $\sigma = a$; for larger values of $\sigma$ there is one ridge. Another aspect of the behavior described by Clark for phantom and actual edges can also be observed. For $\sigma < a$, the gradient magnitude for the actual edges is decreasing as $\sigma$ increases while the magnitude of the phantom edge is increasing (the valley is getting higher). Thus the profile along $x = 0$ shows the magnitude increasing until $\sigma = a$, then decreasing.

When $b < 1$ the stronger edge moves toward the middle and approaches the asymptote $a(1 - b)/(1 + b)$ as $\sigma$ approaches $\infty$. See the appendix for the derivation of the asymptote. For the case of $b < 1$, when $\sigma$ becomes sufficiently large the zero-crossing corresponding to the weak edge on the left combines with the zero-crossing of a phantom edge and the pair disappears. Thus when $b = .8$ the maximum movement of the weak edge occurs at the top of the arch and is $(1 - .493)a = .507a$. When

$b = .3$ the maximum movement is $(1 - .723)a = .277a$.

The units on both axes in the scale space image are marked in units of $a$. This can be done because if $x$ and $\sigma$ are both multiplied by $a$, then

$$s''_{a\sigma,a}(ax) = bg'_{a\sigma}(ax + a) + g'_{a\sigma}(ax - a)$$

But

$$g'_{a\sigma}(ax \pm a) = -a(ax \pm a)/(a^2\sigma^2\sqrt{2\pi}\, a\sigma)\exp(-(ax \pm a)^2/2a^2\sigma^2)$$

$$= (-x \pm 1)/(\sqrt{2\pi}\, a\sigma^3)\exp(-(x \pm 1)^2/2\sigma^2) = (1/a)g'_\sigma(x \pm 1)$$

Thus

$$s''_{a\sigma,a}(ax) = (1/a)s''_{\sigma,1}(x)$$

The function has been multiplied by a constant, but the location of the zero-crossing will occur at the same locations, and one graph can be used to represent all values of $a$.

In practice when an image is being examined, $\sigma$ is known, but $a$ is not. Thus, instead of considering $a$ constant and $\sigma$ as the variable in Figures 19 and 20, $\sigma$ can be fixed and $a$ can be allowed to vary. For a fixed value of $\sigma$, different points on the vertical axis will then correspond to different values of $a$. In this discussion, whenever a fixed value of $\sigma$ is being considered, $\bar{\sigma}$ will be used instead of $\sigma$ to indicate this.

Refer to Figure 19 for an example of how this works. If $\sigma$ has the fixed value 2, the point $2a$ on the vertical, $\sigma$, axis will correspond to $a = 1$, ($\sigma = 2 = 2a$, thus $a = 1$), while the point $a$ will correspond to $a = 2$, and $.5a$ will correspond to $a = 4$. In general, if $\hat{\sigma}a$ is a point on the $\sigma$ axis, then $\bar{\sigma} = \hat{\sigma}a, a = (1/\hat{\sigma})\bar{\sigma}$. Thus for fixed $\sigma$, points on the vertical axis having larger coefficients of $a$ correspond to smaller values of $a$, while points having smaller coefficients correspond to larger $a$. If $(\hat{x}, \hat{\sigma})$ are the coefficients of $a$ for a point on the scale space curve, the distance that the edge point

has moved from its original location at $a$ will be $(1 - \hat{x})a = (1 - \hat{x})(1/\hat{\sigma})\bar{\sigma}$ for the strong edge and $(1 + \hat{x})a = (1 + \hat{x})(1/\hat{\sigma})\bar{\sigma}$ for the weaker edge.

Since movement ($m$) depends on $a$, and both $m$ and $a$ can be expressed in terms of $\bar{\sigma}$, this suggests plotting $m$ versus $a$. This is shown in Figure 22. On both axes the units are $\bar{\sigma}$. Notice that if $b = 1$, when two edges are $2\bar{\sigma}$ apart ($a = \bar{\sigma}$) each will move $\bar{\sigma}$ and combine to become one edge. If the edges are closer together (smaller value of $a$), they will move $a$ pixels each to come together, but because they were closer to begin with, the distance moved will be smaller. Thus for $a \leq \bar{\sigma}$, $m = a$. If the distance apart is greater than $2\bar{\sigma}$ ($a > \bar{\sigma}$) they will move closer together, but remain distinct. When $a > 2\bar{\sigma}$ the amount of movement is negligible. This corresponds to the part of the scale space image where the curve is near vertical. In this situation the edges are far enough apart to have little interaction, so there is little movement. This is a result of the fact that 99% of the support of a Gaussian filter having standard deviation $\sigma$ falls within $3\sigma$ of the mean, so there is effectively no interaction when edges are this far apart. The interaction begins slowly as the edges become closer together. When $a = 2\bar{\sigma}$, the movement is only about $0.0014\bar{\sigma}$, or less than 0.1% of the distance between the edges. As $a$ becomes smaller, the movement increases rapidly to the maximum at $(\bar{\sigma}, \bar{\sigma})$, then decreases until $a$ reaches 0.

For $b < 1$, the movement of the stronger edge will be largest for some value of $a$ between 0 and $\bar{\sigma}$. Figure 23 shows a graph of the maximum movement possible, in terms of $\bar{\sigma}$, for different values of $b$, for the stronger and weaker edges in the staircase. Always the largest movement of $\bar{\sigma}$ will occur for equal edges which are $2\bar{\sigma}$ apart. For example, if an image is convolved with the gradient of Gaussian having $\sigma = 2$, then maximum movement is 2 pixels and occurs when $b = 1$ and $a = 2$. However, if it were known that most neighboring edges had relative strength 0.5, then their greatest movement would be about $0.46 \times 2 = 0.92$, or less than 1 pixel. Similarly, most edge

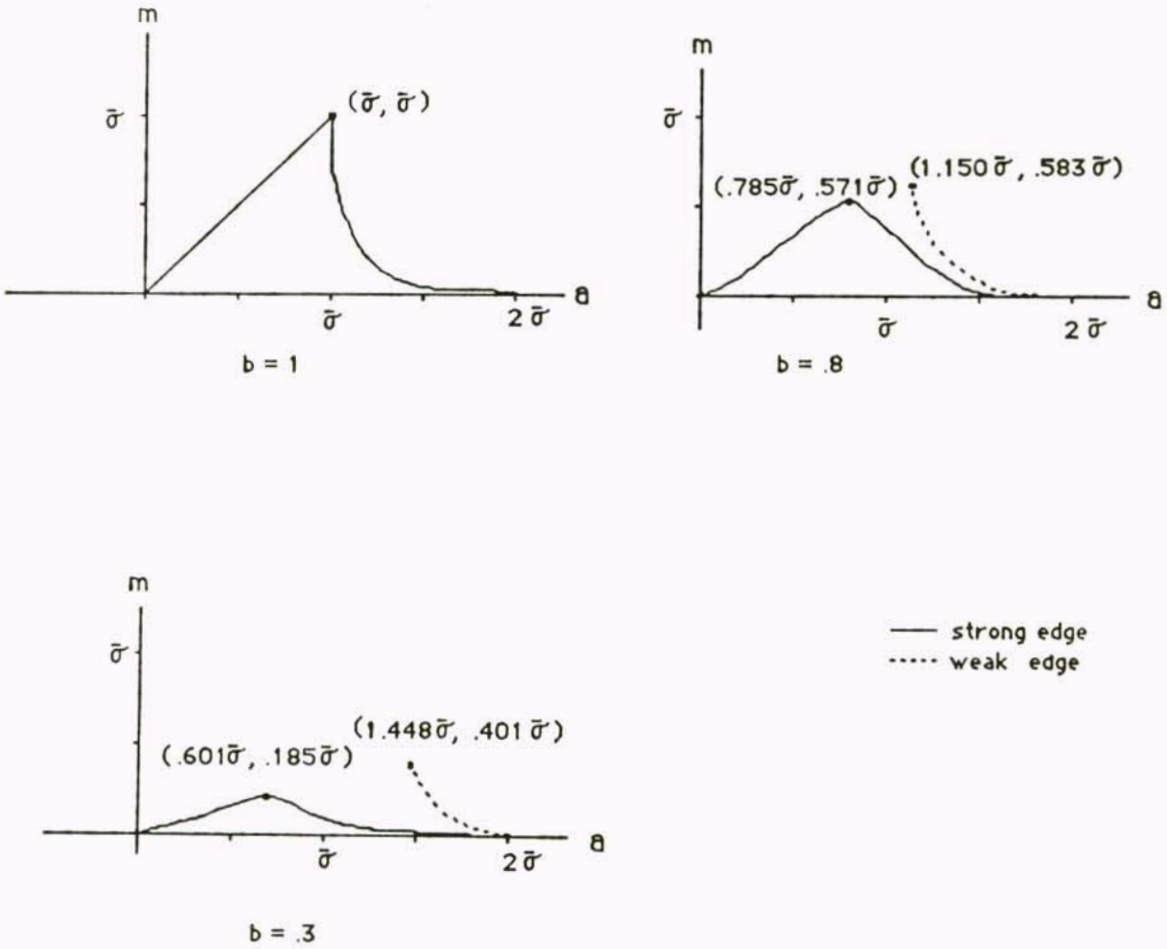



Figure 22: Movement vs. Distance between edges, Staircase.

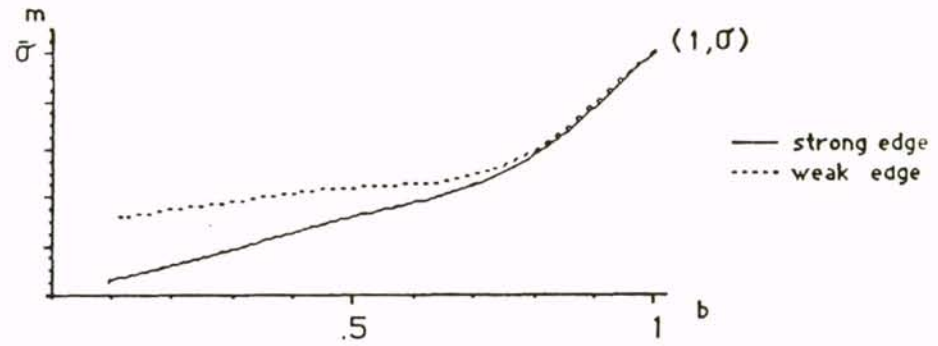



Figure 23: Maximum movement in terms of $\bar{\sigma}$ vs. b for weaker and stronger edges in a staircase.

pairs will not be exactly four pixels apart, thus movement in these cases will also be less than the maximum two pixels.

A similar analysis can be performed for a pulse. Figure 24 shows the movement versus the distance between edges for a pulse. The maximum movement, when $b = 1$, is $\bar{\sigma}$ as it was for the staircase, but this value is now the limiting value as the edges become closer together. This can be seen by examining the scale space graphs for the pulse (Figure 20). When $b = 1$ the curve approaches the lines $\sigma = \pm x$ [18]. Movement is $(|\hat{x}| - 1)a = (|\hat{x}| - 1)\bar{\sigma}/\hat{\sigma}$. Thus

$$\lim_{\hat{x}\to\infty} m = \lim_{\hat{x}\to\infty} \frac{\hat{x} - 1}{\hat{\sigma}}\bar{\sigma}$$

Since the scale space curve approaches $\hat{x} = \hat{\sigma}$ for large $\hat{x}$ this is equal to

$$\lim_{\hat{x}\to\infty} \frac{\hat{x} - 1}{\hat{x}}\bar{\sigma} = \bar{\sigma}$$

But for large $\hat{x}$, $a = \bar{\sigma}/\hat{\sigma} = \bar{\sigma}/\hat{x}$ and $\bar{\sigma}$ is fixed so as $\hat{x} \to \infty$, $a \to 0$ and

$$\lim_{a\to 0} m = \bar{\sigma}$$

When $b < 1$ the strong edge in the scale space image approaches the vertical asymptote $a(1 + b)/(1 - b)$ and displays a well-defined maximum movement as in the

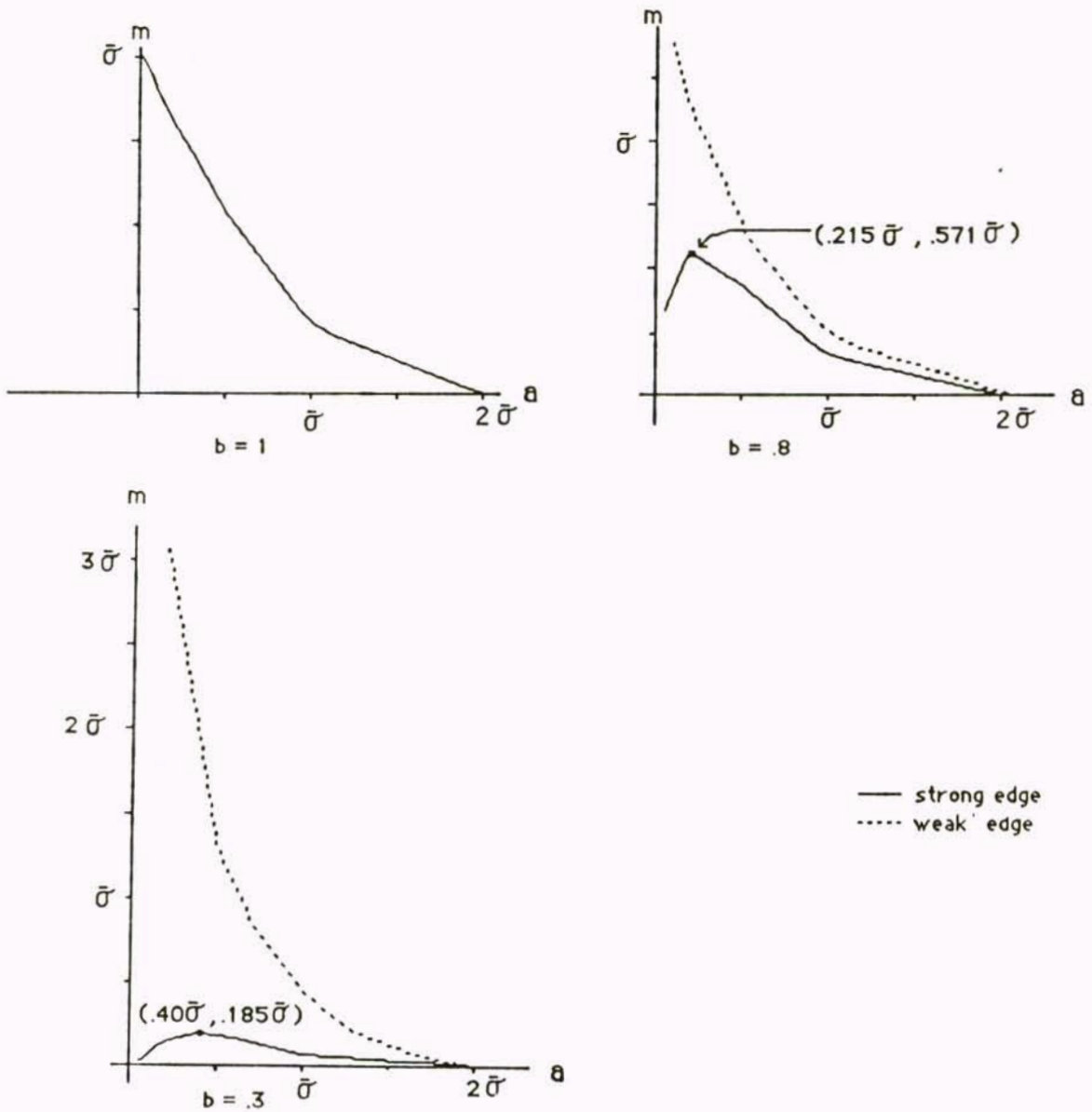



Figure 24: Movement vs. Distance between edges, Pulse.

staircase, when $a$ is between 0 and $\bar{\sigma}$. But the weak edge can move indefinitely as $a$ becomes smaller. In the scale space image the weak edge approaches the horizontal parabola $x = \frac{\ln b}{2a}\sigma^2$. But $p'_{\sigma,a}(\frac{\ln b}{2a}\sigma^2) = 0$ for all values of $\sigma$, thus the parabola gives the location where the gradient value crosses zero as it goes from the positive step of the pulse to the negative. The rate of change of position as $\sigma$ changes is near that of the parabola. Thus $\frac{dx}{d\sigma} = \frac{\ln b}{a}\sigma$ will be large for small $b$ or small $a$. See the appendix for the derivation of the asymptotes.

There are two practical considerations limiting the amount of movement of the weak edge. The first is that as $a$ becomes smaller and the weak edge is moved farther, its location becomes closer to the parabola above, thus its gradient value becomes smaller, and at some point falls below any threshold being used. The other limiting factor relates to the sampling theorem. In the case of the pulse defined above, the wavelength is $4a$. The distance unit being used is the interpixel distance, which is assumed to be the same as the sampling distance in the original image. The sampling theorem states that the sampling interval $\delta$ should be less than $\lambda/2$ where $\lambda$ is the wavelength of the highest frequency. Since $\delta$ is 1, $\delta < \lambda/2$ means $2 < \lambda = 4a$. Therefore $a > 1/2$. Thus for $0 < a < 1/2 = (1/2\bar{\sigma})\bar{\sigma}$ the conditions of the sampling theorem are not met. Figure 25 gives an example when $b = .8$ and $\bar{\sigma} = 2$. Then $a = (1/2\bar{\sigma})\bar{\sigma} = (1/4)\bar{\sigma}$ is the cutoff point and maximum movement is $.5675\bar{\sigma}$ for the strong edge and $1.004\bar{\sigma}$ for the weak edge.

Figure 26 gives a graph of maximum movement for the stronger edge of a pulse as $b$ varies.

The numerical calculations presented in this section were performed on a Macintosh SE computer using Borland's *Eureka* package to solve equations and find maxima and minima. The three-dimensional plots in Figure 21 were done using *MacFunction* by Think Technologies.

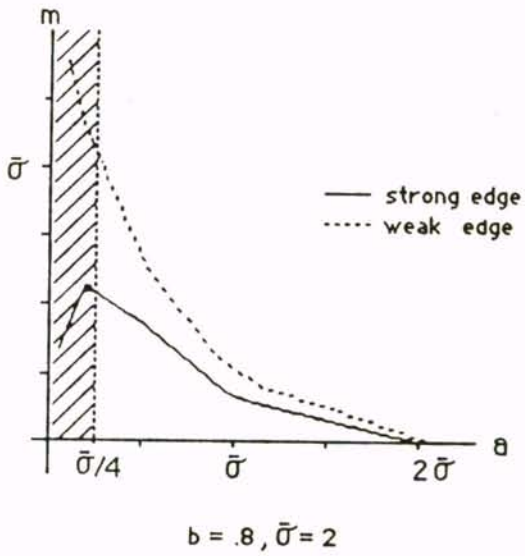



Figure 25: Maximum movement when b = .8 and $\bar{\sigma} = 2$.

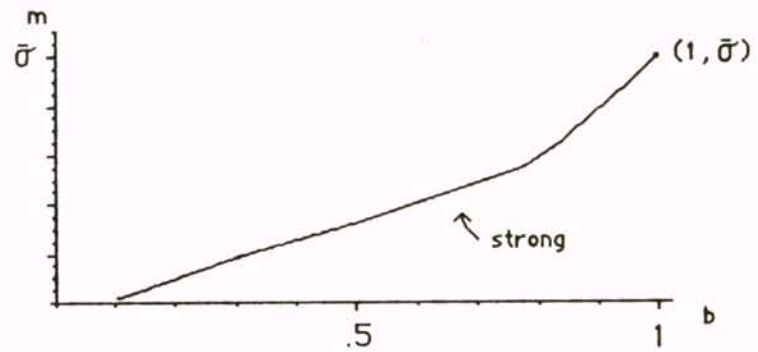



Figure 26: Maximum movement in terms of $\bar{\sigma}$ vs. b for stronger edge in a pulse.

## 4.3 Conclusions

The movement of edges modeled as adjacent step edges of the same or opposite parity has been examined. It was determined that the greatest movement was $\sigma$ and occurred for the staircase when the edges were $2\sigma$ apart and of equal contrast, and for the pulse when the edges were very close together and of equal contrast. For edge pairs not satisfying these extreme conditions the movement was considerably smaller. The graphs given depict the movement under different combinations of distance between edges, relative strength, and degree of smoothing. The equations for the scale space of the pulse and staircase were written as functions of $\sigma$, and the domain and asymptotes for the functions were determined.

# 5. NORMALIZED EDGE OPERATOR

When an image is smoothed with a Gaussian operator to remove noise and fine texture, the edge points are displaced. The amount of delocalization is a function of $\sigma$, the scale of the operator, as discussed in the previous chapter. However, the magnitude of the gradient at an edge point also changes with scale. In this chapter an analysis is presented of how the magnitude changes for different ideal edge types. The gradient is shown to contain much more information than is typically used in gradient based edge operators. The knowledge obtained from this analysis is used to characterize edge points, each point being assigned values for slope, stepsize, and scale. It can also be determined what type of edge interaction is occurring.

When considering an image smoothed at one scale, operations such as non-maxima suppression and zero-crossing detection are concerned only with the comparative magnitudes of the gradient at different points, or with points where the Laplacian of the Gaussian has zero values. Thus the normalizing factor of the n-dimensional Gaussian, $(\sqrt{2\pi}\sigma)^{-n}$, is often omitted or replaced by a more convenient scaling factor. See, e.g., [16, 38, 39]. However, when examining more than one scale, the choice of factor is important. Clark [9, 10] shows that the contrast of an authentic zero crossing decreases as $\sigma$ increases, while that of a phantom zero crossing increases. The magnitude of the gradient of the Gaussian is an acceptable contrast function, and thus exhibits this behavior. However, omitting the $(\sqrt{2\pi}\sigma)^{-n}$ term gives a function for which this result does not hold. For this function the magnitude of the response increases and decreases as a result of edge profile and edge interaction. Korn [19] suggests using a two-dimensional gradient of Gaussian operator which has been normalized by multiplying with the factor $\sqrt{2\pi}\,\sigma$. He defines the scale of an edge to be the scale at

which the magnitude of the gradient vector obtained with this operator first reaches its maximum value.

In this discussion the two-dimensional operator is separated into the product of a one-dimensional normalized gradient of Gaussian operator and a one-dimensional Gaussian. The behavior of the magnitude of the response to the normalized gradient operator is analyzed as scale changes. This operator is of interest because the response of an ideal step edge is constant for all values of $\sigma$, as will be shown in the next section. This provides a basis for comparing edge responses at different scales, and makes certain types of information about the edges more accessible. While it is very important to know where edge points occur, it is also important to know other characteristics of the gray level function at the edge, such as its total contrast, steepness or slope, and spatial extent or width. This information is useful in classifying edges as to type (shadow, surface markings, occlusion, etc.), and in matching problems such as stereo and motion. Hildreth [16] used the slope of the zero-crossing, a third derivative, at two scales to determine the slope and width of a ramp, but did not consider interaction of nearby edges.

In Section 5.1 the operator is defined, then in Section 5.2 a discussion is given of the behavior of idealized edges and combinations of edges under the normalized gradient of Gaussian operator and a definition of the circumstances under which the gradient magnitude will increase or decrease. It is also shown that with the information derived, a single small scale is sufficient to determine the slope of a ramp edge, and that for isolated edges, the stepsize and width of the edge can be determined by the behavior of the gradient. Further, the stepsize of an edge undergoing interaction with its neighbors can be estimated using the methods developed in this chapter. Simulations of the operator applied to ideal edges and a demonstration of how the theoretical results of Section 5.2 can be applied to obtain information about edges in

real images are presented in Section 5.3.

## 5.1 The Normalized Operator

The two-dimensional Gaussian function is defined by the equation

$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(x^2 + y^2)}{2\sigma^2}\right)$$

This can be separated into the product of two one-dimensional Gaussians:

$$g(x, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{2\sigma^2})$$

The multiplicative factor ensures that the area under the curve is 1.

The derivative of the one-dimensional Gaussian is

$$g'(x, \sigma) = \frac{-x}{\sqrt{2\pi}\sigma^3} \exp(-\frac{x^2}{2\sigma^2})$$

The area between the curve and the $x$-axis is in two parts, that for $x < 0$ is above the axis, while that for $x > 0$ is below. The areas of the two parts are the same and are equal to

$$\int_{-\infty}^{0} g'(x, \sigma)dx = \frac{1}{\sqrt{2\pi}\sigma}$$

The $\sigma$ term in the denominator ensures that the area under each half of the curve will decrease as $\sigma$ increases. Normalizing consists of multiplying the one-dimensional gradient of Gaussian by the factor $\sqrt{2\pi}\sigma$ to make the area under the curve constant and equal to one. Then the product of the Gaussian with the normalized gradient operator gives a two-dimensional gradient operator, $\sqrt{2\pi}\sigma g'(x, \sigma)g(y, \sigma)$, which has volume under the surface of 1 for the negative and positive parts and gives the gradient in the $x$ direction. The gradient in the $y$ direction is computed similarly. Since the gradient operator is separable, the normalized one-dimensional derivative

operator can be examined to determine the behavior of cross sections of edges in two dimensions. The normalized gradient operator will be defined as

$$G'(x,\sigma) = \sqrt{2\pi}\,\sigma g'(x,\sigma) = -\frac{x}{\sigma^2}\exp(-x^2/2\,\sigma^2)$$

and for consistency,

$$G(x,\sigma) = \sqrt{2\pi}\,\sigma g(x,\sigma) = \exp(-x^2/2\,\sigma^2)$$

## 5.2 Ideal Edge Models

In this section the behavior of certain ideal edges is examined as the normalized edge operator is applied.

### 5.2.1 Step Edge

The unit step edge at $x = 0$ is represented by the equation

$$U(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{otherwise} \end{cases}$$

A step of arbitrary height, $c$, is given by $c\,U(x)$. Convolving an arbitrary step with $G'$ gives

$$\mathcal{U}'(x) = c\,U(x) * G'(x,\sigma) = c\,U'(x) * G(x,\sigma) = c\,G(x,\sigma)$$

since $U'(x)$ is the impulse function and convolution with it gives the original function [15, page 82]. The maximum value of $G(x,\sigma)$ is 1 and occurs at $x = 0$. Thus the maximum value of $\mathcal{U}'(x)$ is $c$ and always occurs at the location of the step, $x = 0$. Further

$$\frac{\partial \mathcal{U}'}{\partial \sigma} = c\frac{x^2}{\sigma^3}\exp(-\frac{x^2}{2\sigma^2})$$

This expression is equal to 0 when $x = 0$ and $\sigma \neq 0$. Therefore the response of a step to the normalized gradient operator is constant, and since $\mathcal{U}'(0) = c$, its value is the stepsize.

### 5.2.2 Ramp

A ramp edge is represented by the equation

$$r(x) = \begin{cases} 0 & \text{if } x < 0 \\ mx & \text{if } 0 \leq x \leq w \\ mw & \text{if } x > w \end{cases}$$

where $m$ is the slope of the ramp and $w$ is its width. Its derivative is given by

$$r'(x) = \begin{cases} m & \text{if } 0 < x < w \\ 0 & \text{otherwise} \end{cases}$$

$$= m(U(x) - U(x - w))$$

Convolving with the normalized Gaussian gives

$$R'(x) = r'(x) * G(x, \sigma) = m \int_{x-w}^{x} G(u, \sigma) du$$

The integral has its largest value when the limits of integration are centered on $u = 0$, i.e., at $x = w/2$. Thus the isolated ramp will always be detected at its midpoint:

$$R'(w/2) = m \int_{-w/2}^{w/2} G(u, \sigma) du$$

When $\sigma$ is small enough that most of the support of the Gaussian falls inside the interval $(-w/2, w/2)$ (see Figure 27a), the value of this integral will be $m\sqrt{2\pi}\,\sigma$. Thus as $\sigma$ increases, the value of the gradient will increase linearly with $\sigma$ until $\sigma$ becomes large enough for the ends of the ramp to be included in the support of the operator. Since 98% of the support of the Gaussian falls in a $5\sigma$ interval around 0, the linear behavior will be apparent for edges separated by a distance larger than this.

Since

$$\lim_{\sigma \to \infty} G(x, \sigma) = \lim_{\sigma \to \infty} \exp(-\frac{x^2}{2\sigma^2}) = 1$$

it follows that

$$\lim_{\sigma \to \infty} R'(w/2) = \lim_{\sigma \to \infty} m \int_{-w/2}^{w/2} G(u, \sigma) du = mw$$

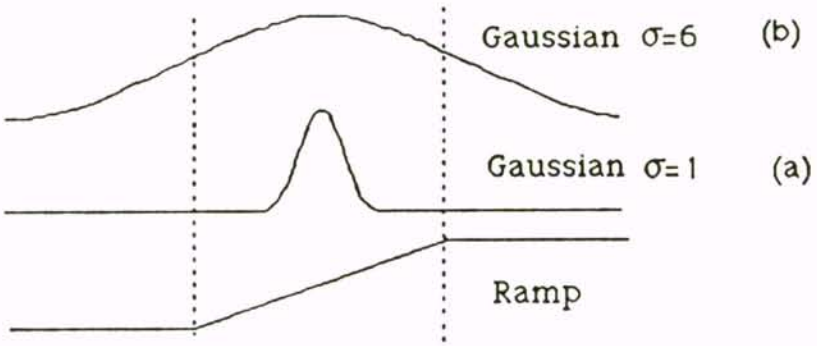



Figure 27: (a) Support of Gaussian falls completely in ramp, (b) Gaussian includes ends of ramp.

which is the stepsize. Taking the partial derivative of $R'$ with respect to $\sigma$ gives

$$\frac{\partial R'}{\partial \sigma} = \frac{m}{\sigma^3} \int_{x-w}^{x} u^2 G(u, \sigma) du$$

which always has the sign of $m$ since $\sigma$ and the area under the integral are positive. This means that $R'$ is a monotonic function. In summary, if a ramp is isolated, the gradient magnitude will increase linearly with $\sigma$ until the ends of the ramp begin to influence the value, then the rate of increase will slow, but the value will continue to increase, approaching the limit $|mw|$.

### 5.2.3 Staircase

The staircase having steps of the same parity at $x = a$ and $x = -a$ and relative heights $b$ is given by the equation

$$s_a(x) = U(x + a) + bU(x - a)$$

The smaller edge, having stepsize $b$, is at $x = a$. Convolving with the normalized gradient of Gaussian gives

$$S'_a(x, \sigma) = G(x + a, \sigma) + bG(x - a, \sigma)$$

When $\sigma = 0$, there is no interaction between the two edges and $S'_a(x,0) = 0$ when $x$ is not equal to $a$ or $-a$. At these points

$$\lim_{\sigma \to 0} S'_a(a,\sigma) = \lim_{\sigma \to 0} G(2a,\sigma) + b = b$$

and

$$\lim_{\sigma \to 0} S'_a(-a,\sigma) = 1$$

Thus the individual edges behave like isolated step edges. When $\sigma$ is greater than 0 a valley or phantom edge will exist between the two step edges. As $\sigma$ becomes larger the two edges move together, and the weaker edge and the phantom edge disappear, while the stronger one remains [39]. For large values of $\sigma$, $\lim_{\sigma \to \infty} S'_a(x,\sigma) = 1 + b$. Thus, for large $\sigma$, the staircase appears like a single step having stepsize equal to the sum of the separate stepsizes, and from the results of Chapter 4, its location will be $x = \frac{a(b-1)}{b+1}$. This behavior is demonstrated for a sample edge in Figure 28.

### 5.2.4 Pulse

A pulse is defined as two neighboring steps of opposite parity. The equation of the pulse is

$$p_a(x) = U(x+a) - bU(x-a)$$

The equation of the pulse convolved with the normalized gradient of Gaussian operator is

$$P'_a(x,\sigma) = G(x+a,\sigma) - bG(x-a,\sigma)$$

When $\sigma = 0, P'_a(x,0) = 0$, for $x \neq a, -a$. When $\sigma = 0$ and $x = a$,

$$\lim_{\sigma \to 0} P'_a(a,\sigma) = \lim_{\sigma \to 0} G(2a,\sigma) - b = -b$$

When $x = -a$,

$$\lim_{\sigma \to 0} P'_a(-a,\sigma) = \lim_{\sigma \to 0}(1 - bG(-2a,\sigma)) = 1$$

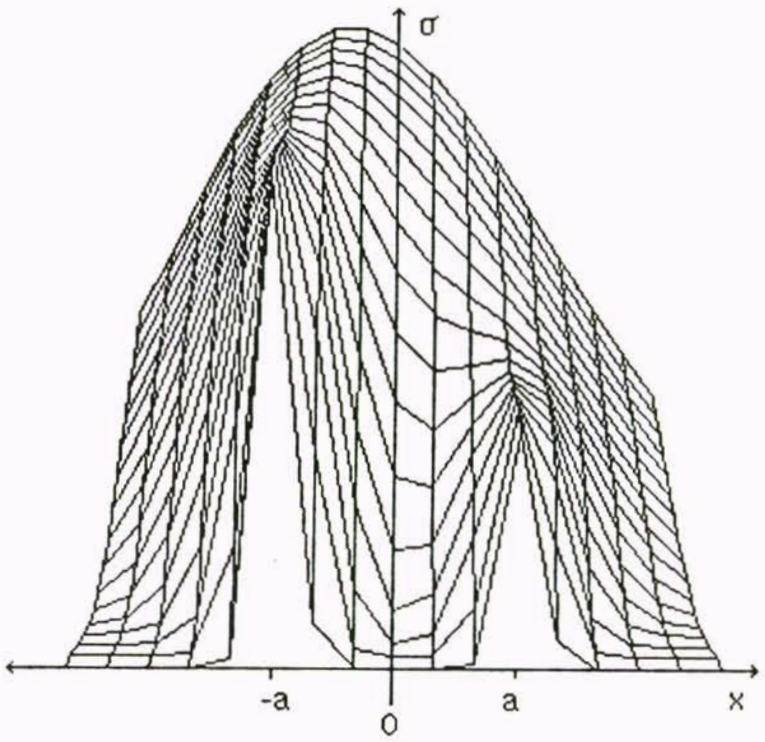



Figure 28: Three-dimensional plot of gradient magnitude for staircase having $b = 0.5$.

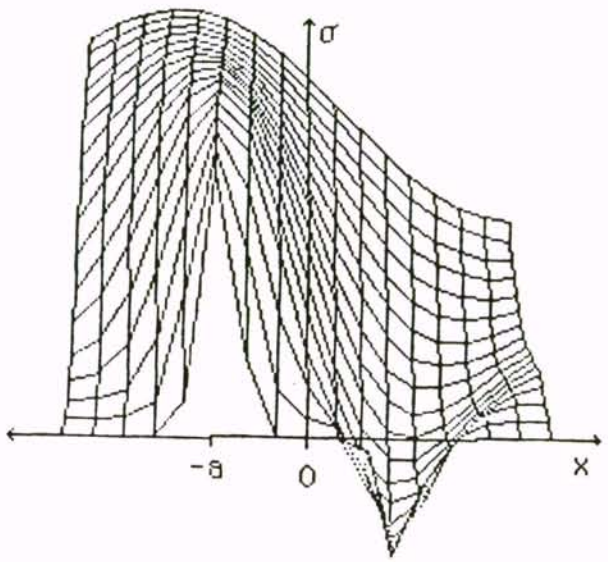



Figure 29: Three-dimensional plot of gradient magnitude for pulse having $b = 0.5$.

Thus, as expected, for small values of $\sigma$ the two edges behave as step edges. As the edges begin to interact, they move apart. When $\sigma$ becomes large, $P'_a(x, \infty) = 1 - b$; the pulse appears as a single step edge having contrast equal to the difference of the two steps. This edge will appear at the location $x = \frac{a(b+1)}{b-1}$ and is the stronger edge. The weaker edge approaches the horizontal parabola $x = -\sigma^2 \ln b/2a$ as $\sigma$ increases [40]. But $P'_a(-\frac{\sigma^2 \ln b}{2a}, \sigma) = 0$; thus this is the track of the zero gradient that appears between the positive gradient of the left edge and the negative gradient of the right edge. Since the weak edge approaches this parabola, the gradient value of the weak edge must also approach 0. This behavior is demonstrated for a sample edge in Figure 29.

### 5.2.5 Conservation of Contrast

Notice that when adjacent edges of either parity interact the contrasts of the two combine. For the staircase the total contrast of the two edges when $\sigma = 0$ is $1 + b$,

and the contrast for the single combined step approaches $1 + b$ as $\sigma$ approaches $\infty$. Similarly for the pulse, the combined contrast of the two edges when $\sigma = 0$ is $1 - b$, and the limit as $\sigma$ approaches $\infty$ is $1 - b$. In the limit none of the contrast has been lost. Thus the following theorem has been proved.

**Conservation of contrast:** If two adjacent step edges have contrast $b$ and $c$, then the sum of the gradient maxima will be $b + c$ when the image is smoothed with the normalized gradient of Gaussian operator having $\sigma = 0$. The gradient maximum for the stronger edge will also approach $b + c$ when $\sigma \rightarrow \infty$. If $b$ and $c$ have the same sign, then the maximum for the weaker edge will disappear for some value of $\sigma$, or the two maxima will combine to become one if $b = c$. If $b$ and $c$ have opposite signs, then the gradient maximum of the weaker edge will approach 0. When $b = c$ the gradient maxima for both edges approach 0.

It is interesting that although this principle holds for $\sigma = 0$ and $\infty$, experimental results indicate that it does not hold for intermediate values. As can be seen in Table 2b and c, after the weaker edge has disappeared (in the case of a staircase), or become very small (in the case of a pulse), the magnitude of the stronger edge continues to change, approaching the limiting value. Thus an edge near a stronger one continues to influence the response of the stronger one even at scales at which it cannot itself be detected.

### 5.2.6 Summary

The behavior of the above edges leads to the following observations. A pure step edge has a constant gradient magnitude equal to the stepsize for all values of $\sigma$. The magnitude of ramp edges and any combination of adjacent ramp edges and step edges having the *same* parity will increase as $\sigma$ increases, approaching the sum of their

| edge type | magnitude of normalized gradient | | location of edge as $\sigma$ increases | change in grad. mag. as $\sigma$ increases |
|---|---|---|---|---|
| | when $\sigma = 0$ | when $\sigma = \infty$ | | |
| step | contrast | contrast | at step | constant |
| ramp | $m\sqrt{2\pi}\,\sigma = 0$ | contrast | middle of ramp | up |
| staircase (strong) | contrast | sum of contrasts $(1 + b)$ | at step, moving to $\frac{a(b-1)}{b+1}$ | up |
| staircase (weak) | contrast | — | moves toward 0, then disappears | up |
| pulse (strong) | contrast | difference of contrasts, $(1 - b)$ | at step, moving to $\frac{a(b+1)}{b-1}$ | down |
| pulse (weak) | contrast | 0 | at step, moving to $\frac{\sigma^2 \ln b}{2a}$ | down |
| ramp staircase | $m\sqrt{2\pi}\,\sigma = 0$ | sum of contrasts | same as staircase, $a$ is ramp mid pt. | up |
| ramp pulse | $m\sqrt{2\pi}\,\sigma = 0$ | difference of contrasts | same as pulse, $a$ is ramp mid pt. | up then down |

Table 1: Summary of gradient behavior. In formulas, $\sigma$ is the scale, $m$ is slope of ramp edges. Step and ramp edges are at 0, pairs of edges in staircases and pulses are at $a$, $-a$, with the step at $-a$ having contrast 1, that at $a$ having contrast $b$, $0 < b \leq 1$.

contrasts. A pulse edge will begin as two step edges and the magnitude will decrease, that of the stronger edge to the difference of the two edges, and that of the weaker edge to zero. A pulse formed of ramp edges will have the most complicated behavior, increasing linearly with $\sigma$ until the ends of the ramp are within the support of the operator, then at a slower rate until the operator includes the step having opposite parity. Then the magnitude of both edges will decrease, the weaker one to 0, the stronger one to the difference of the contrast of the two edges. Table 1 summarizes these results.

### 5.3 Simulations and Experimental Results

Simulations were performed on a number of synthetic one-dimensional images of the ideal edges examined theoretically. The normalized gradient operator was also applied to real two-dimensional images. For these the slopes of the edges were

| Ramp, $m$=10, $w$=20 | | | | Ramp, $m$=10, $w$=10 | | | | Smooth Ramp, $m$=10, $w$=20 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\sigma$ | loc | gradient | $m$ | $\sigma$ | loc | gradient | $m$ | $\sigma$ | loc | gradient | $m$ |
| 1 | [-7,7] | 25 | 9.97 | 1 | [-2,2] | 25 | 9.97 | 1 | [-4,4] | 25 | 9.97 |
| 2 | [-3,3] | 50 | 9.99 | 2 | 0 | 49.6 | 9.89 | 2 | 0 | 50.1 | 9.99 |
| 4 | 0 | 98 | 9.82 | 4 | 0 | 78.9 | 7.87 | 3 | 0 | 74.2 | 9.87 |
| 8 | 0 | 157 | 7.85 | 8 | 0 | 93.5 | 4.66 | 4 | 0 | 95.6 | 9.53 |
| 20 | 0 | 189 | 3.77 | 20 | 0 | 97.4 | 1.94 | 20 | 0 | 185.7 | 3.70 |

(a)

| Pulse, location -5, 5, stepsize = 100 | | | | stepsizes = 100, -60 | | | | |
|---|---|---|---|---|---|---|---|---|
| $\sigma$ | locations | | gradient | $\sigma$ | locations | | gradient | |
| 1 | [-6,-5] | [5,6] | ±91.2 | 1 | [-5,-6] | [5,6] | 91.2 | -54.7 |
| 2 | [-6,-5] | [5,6] | ±97.85 | 2 | [-5,-6] | [5,6] | 97.9 | -58.7 |
| 4 | -6 | 6 | ±97.66 | 4 | -6 | 6 | 98.2 | -58.0 |
| 16 | -17 | 17 | ±39.8 | 16 | -12 | 30 | 58.6 | -12.7 |
| 20 | -21 | 21 | ±32.2 | 30 | -17 | 45 | 47.5 | -.9 |

(b)

| Staircase, location -5, 5, stepsize = 50 | | | | stepsizes = 40, 60 | | | | |
|---|---|---|---|---|---|---|---|---|
| $\sigma$ | locations | | gradient | $\sigma$ | locations | | gradient | |
| 1 | [-6,-5] | [5,6] | 45.6 | 1 | [-5,-6] | [5,6] | 36.5 | 54.7 |
| 2 | [-6,-5] | [5,6] | 48.9 | 2 | [-5,-6] | [5,6] | 39.1 | 58.7 |
| 4 | -5 | 5 | 51.0 | 4 | -5 | 5 | 41.4 | 60.6 |
| 8 | 0 | 0 | 78.5 | 8 | - | 2 | - | 79.9 |
| 20 | 0 | 0 | 94.7 | 20 | - | 1 | - | 94.9 |

(c)

| Ramp Pulse, loc -10, 10, $m$ = 10 contrast = 100 | | | | | Ramp Staircase, loc -10, 10, $m$ = 5 contrast each, 50 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\sigma$ | locations | | grad. | $m$ | $\sigma$ | locations | | grad. | $\Delta gr./\Delta\sigma$ | $m$ |
| 1 | [-12,-8] | [8,12] | 25 | 9.97 | 1 | [-12,-8] | [8,12] | 12.5 | - | 4.99 |
| 2 | -10 | 10 | 49.6 | 9.89 | 2 | -10 | 10 | 24.8 | 12.3 | 4.94 |
| 4 | -10 | 10 | 78.9 | 7.87 | 4 | -10 | 10 | 39.5 | 7.3 | 3.94 |
| 8 | -11 | 11 | 88.6 | 4.42 | 8 | -8 | 8 | 50.3 | 2.7 | 2.51 |
| 10 | -12 | 12 | 84.6 | 3.37 | 12 | 0 | 0 | 70.2 | 4.9 | 2.33 |
| 20 | -21 | 21 | 54.2 | 1.08 | 16 | 0 | 0 | 80.9 | 2.7 | 2.02 |

(d) (e)

Table 2: Results of applying normalized gradient to ideal edges.

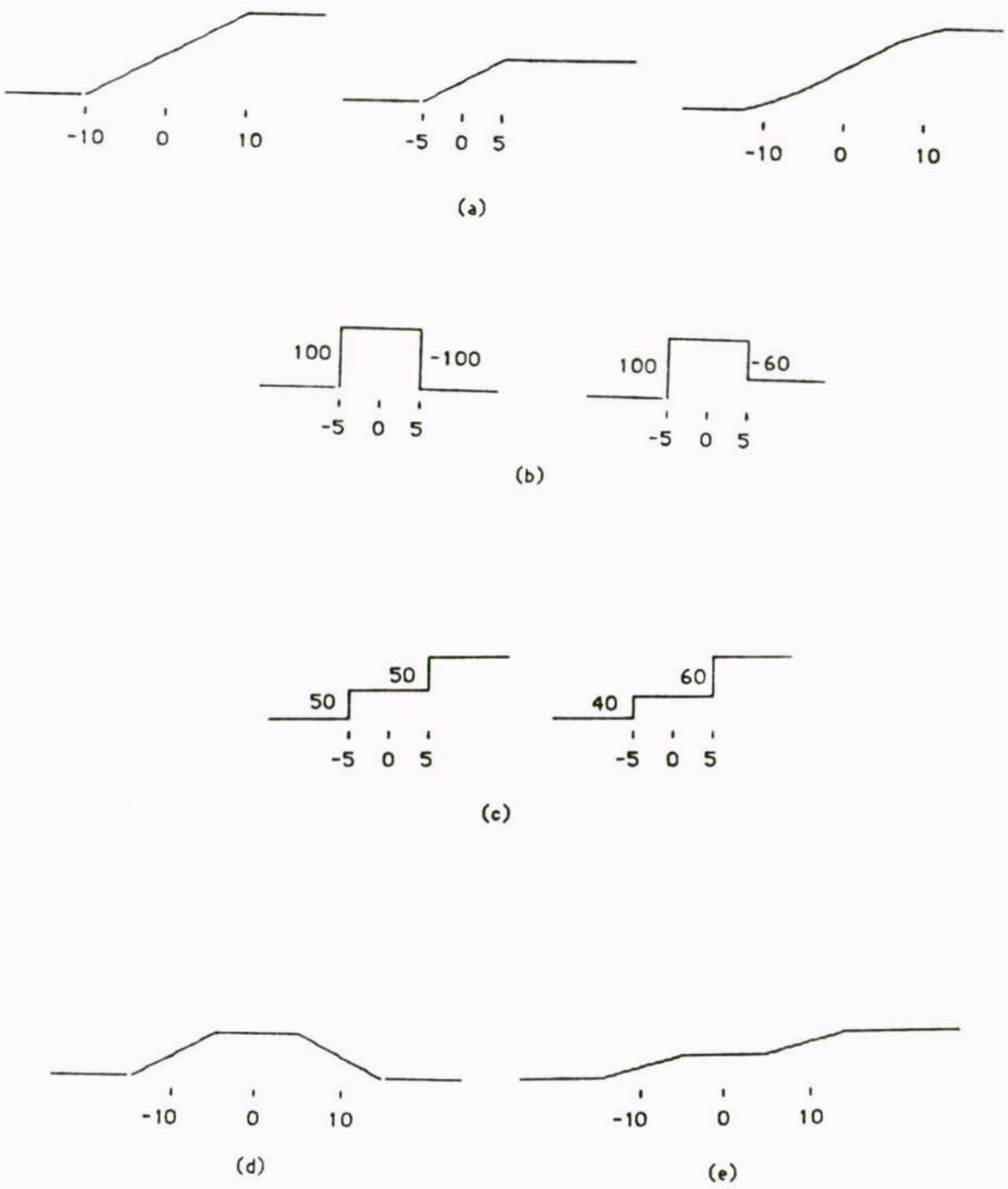



Figure 30: Graphs of edges in Table 2.

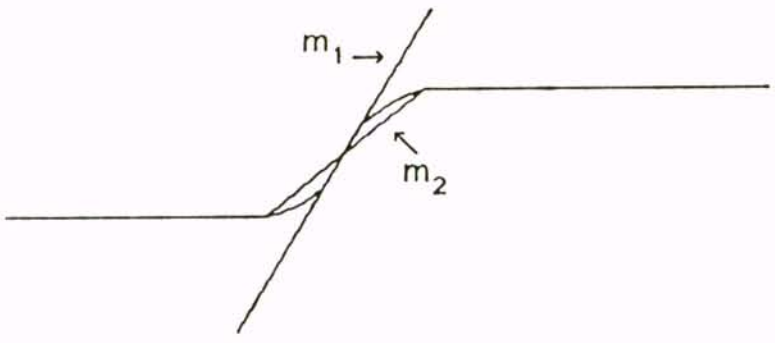



Figure 31: Two ways of computing slope for an edge.

estimated and the edges were characterized by the behavior of the gradient maxima at different scales.

### 5.3.1 Simulations

A number of simulations are reported in Table 2 to demonstrate the behavior of the ideal edges described above as the normalized Gaussian operator is applied. Figure 30 shows graphs of the edges used in the simulation. Values of the gradient for $\sigma = 1$ appear lower than expected due to the fact that the discrete gradient mask does not accurately approximate the continuous one. When a location is given as a closed interval, then the entire interval has the same gradient value. The first two columns of Table 2a give ramps having the same slope and different widths, to demonstrate how the interaction with the ends of the ramp develop. Note that the estimate of the slope for the ramps in a, d, and e are very close to the actual values when $\sigma = 1, 2$ then begin to decrease. In a when $w = 10$ the slope estimate decreases slightly from $\sigma = 1$ to $\sigma = 2$, in contrast to the case where $w = 20$. This indicates that at about $\sigma = w/5$ (in this case 2), with less than 2% of the Gaussian outside the ramp, the estimate is just beginning to be affected. The width of the ramp can be estimated by finding the value of $\sigma$ at which the slope first decreases, then using the formula $w = 5\sigma$.

For an edge with the profile given in Figure 31 there are two possible estimates of slope, $m_1$, the slope at the middle of the ramp, and $m_2$, the average slope over the entire edge. The estimate given in this chapter is of $m_1$. The third column of Table 2a is for a ramp similar to that in Figure 31 having a linear segment of fourteen pixels with slope 10 in the middle. The ends of the ramp have smaller slope, decreasing to 0. The overall width is 44, the contrast is 200, giving a slope over the entire smoothed ramp of about 4.5. The slope estimates for small $\sigma$ are the same as for the ramp, but begin to decrease sooner.

Table 2b gives results for a pulse with equal and unequal stepsizes. The decrease in the gradient value can be seen as the two steps of opposite parity interact. The difference of the two gradient values in the case where stepsizes are unequal is near 40 for small values of $\sigma$ where there is little interaction. However, for larger values of $\sigma$, the difference of the two becomes greater than 40. When the smaller edge has almost disappeared, the gradient magnitude of the stronger edge continues to decrease. This phenomenon of an undetectable edge continuing to affect a nearby edge can be seen more strikingly in c, a staircase with equal and unequal steps. The weaker edge combines with the phantom edge between $\sigma = 4$ and $\sigma = 8$ and disappears as a gradient maximum, but is still present as an inflection point in the gradient magnitude graph. For larger values of $\sigma$ the magnitude of the stronger edge continues to increase.

In d, a pulse composed of ramps rather than steps is given. The gradient value begins low, then when $\sigma = 8$ a maximum estimate of contrast, 88.6, is reached. A more accurate value for contrast could be obtained by first estimating the width of the ramp by the method described above. The value of $\sigma$ at which the slope estimate begins to decrease is 2. Multiplying by 5 gives an estimated width of 10. This is multiplied by the largest slope estimate (9.97) to give a very good estimate of 99.7

for contrast.

Finally, e gives a staircase composed of ramps rather than steps. Of special interest here is the rate of change of gradient value. Between $\sigma = 8$ and $\sigma = 12$, the rate increases. For an isolated ramp edge the rate of increase should be a decreasing function. The increase indicates that there is interaction with an edge of the same parity. In this case the best estimate of contrast is given by the gradient value at $\sigma = 8$ and is 50.3. Higher scales cause the two edges to be combined.

### 5.3.2 Edge Characterization

Because edges in real images often appear like ramp edges or smoothed ramp edges, applying the smallest practical Gaussian to the image will give an estimate of the slope of the edges. Many of the edges detected for a small Gaussian are noise or unwanted fine texture. Thus larger scales should be used to determine which edge points are significant. Since the width of a ramp can be determined by finding the scale at which the nonlinear behavior in $\sigma$ ceases, multiple scales can be applied at or near points of interest to detect this behavior.

In order to determine the best scale at which to estimate the stepsize two tests were used. The first is based on the fact that the gradient value of a ramp is an increasing function with respect to $\sigma$. Thus if the value of the gradient fails to increase with $\sigma$, either the edge is isolated and the gradient has reached a value close to the actual stepsize and becomes constant, or there is interaction with an edge of opposite parity and the gradient decreases. Thus the scale chosen to estimate stepsize is the largest for which the gradient value increases. The second test involves the rate of increase. Since the gradient increases to a finite bound, the rate of increase must be negative. If the rate of increase becomes positive, that indicates interaction with an edge having the same parity. Thus the scale chosen is the largest for which the rate of increase

is negative. An edge can be characterized according to which test determined the scale. The first test would indicate that its most influential neighbor had opposite parity, the second test that the neighbor had the same parity. Edges having small scale will be those having nearby edges, which includes much of the noise, and those having a small width. Diffuse edges, for example those due to illumination gradients or shadows, will have larger scale.

### 5.3.3 Real Images

The tests above have been applied to real images and the results of the characterization at gradient maxima points have been displayed as intensity images. In Figures 32 through 35, the original picture is a; the slope is given in b, with brighter points corresponding to steeper slope. The stepsize is in c with brighter points corresponding to larger stepsizes. Finally, the scale at which the stepsize was estimated is given in d with the brightest points corresponding to the smallest scale. The image in Figure 32 is 128 × 128 pixels, while the others are 256 × 256. In Figures 32 through 34 the values of $\sigma$ used were 1, 2, 3, and 4. In Figure 35 the values used were 1, 2, 4, and 8. Most of the edges had a scale of 4 or smaller, thus these scales were considered sufficiently large.

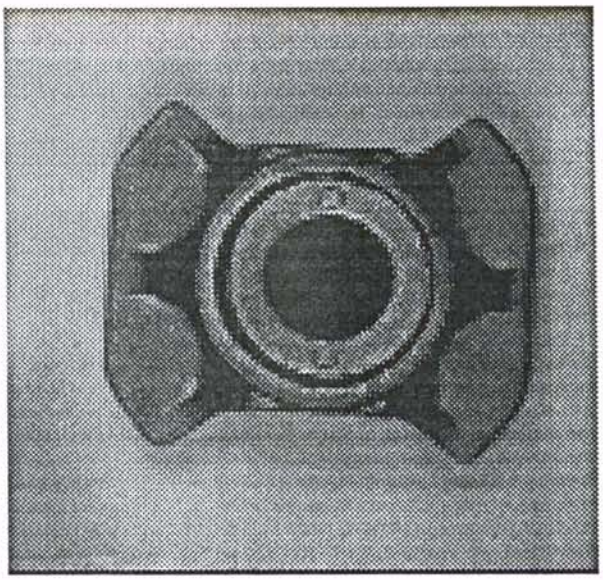
Note in the upper right part of Figure 32 there is a shadow edge. Its slope is smaller than the object edge casting the shadow, but its stepsize is close to that of the object. Notice also that the slope of the rather isolated shadow edge in the middle bottom is small, while the scale chosen to estimate its size is large due to its isolation and width.

The jet image, Figure 33, has a noisy background characterized by small scale and small slope. Some points have fairly large stepsize, indicating that this is not as accurate an indication of noise as the other characteristics. The bottom of the nose of

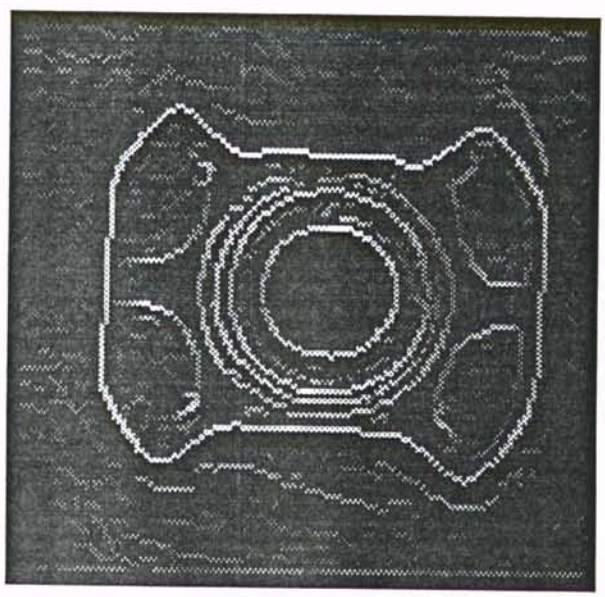
the plane is largely in shadow and has small slope, moderate stepsize and has a large scale because it is diffuse. The top edge of the plane is also at a large scale, while the stripes, being sharper and closer together, have larger slope and smaller scale. The shadows cast by the right engine are of interest as well. The stepsize is fairly constant along the edge, but the slope is smaller toward the forward edge of the wing.

Figure 34 is of a race car driver. The well defined, isolated edges, for example around the window, are characterized by large slope, stepsize, and scale. Where the knee of the driver comes closer to the instrument panel, the edge has a finer scale than does the more isolated part of the edge. The slopes of the edges across the face mask are smaller than the window edge although the stepsize is fairly large. The parallel lines on the console and gear lever have large contrast and slope, and small scale.

In the bottle image the noise is characterized by fine scale, small slope and to a lesser extent small contrast. The well defined outside edges of the bottle have large slope and stepsize. The shadows have smaller slope and larger scale than the object edges.

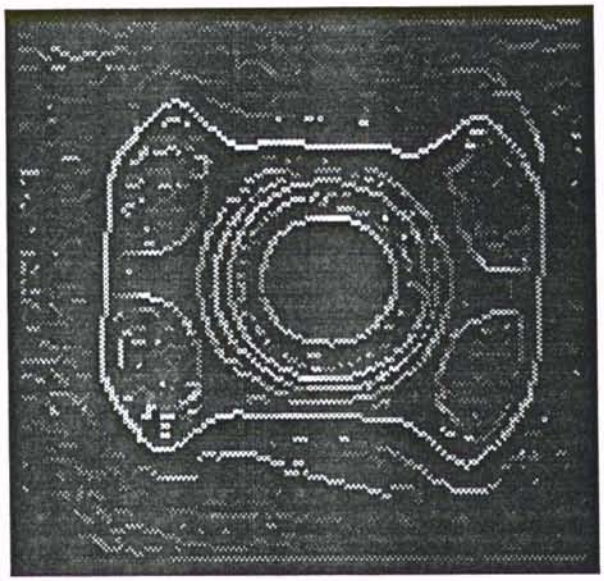


(a)

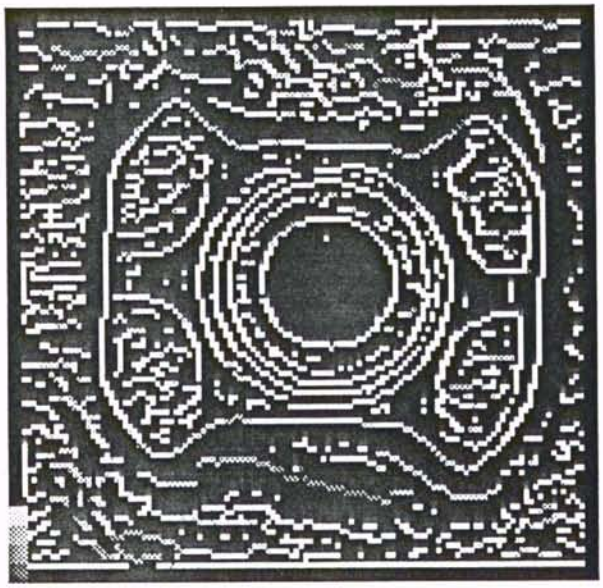


(b)

Figure 32: Part image. (a) original image, (b) slope.



(c)



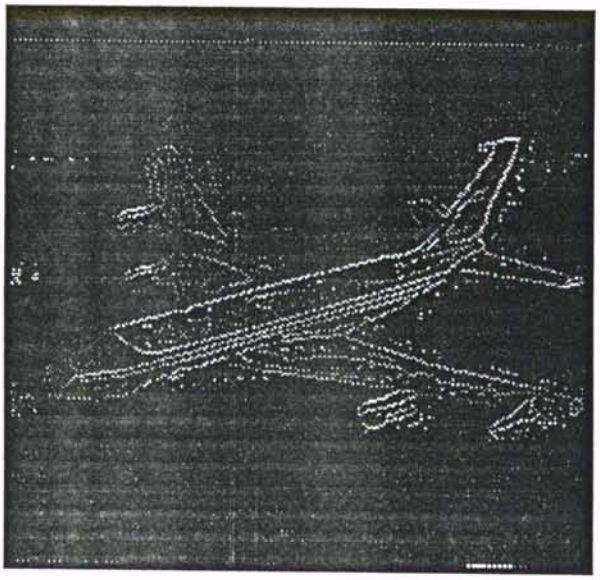
(d)

Figure 32: (continued) Part image. (c) stepsize, (d) scale for estimating stepsize.

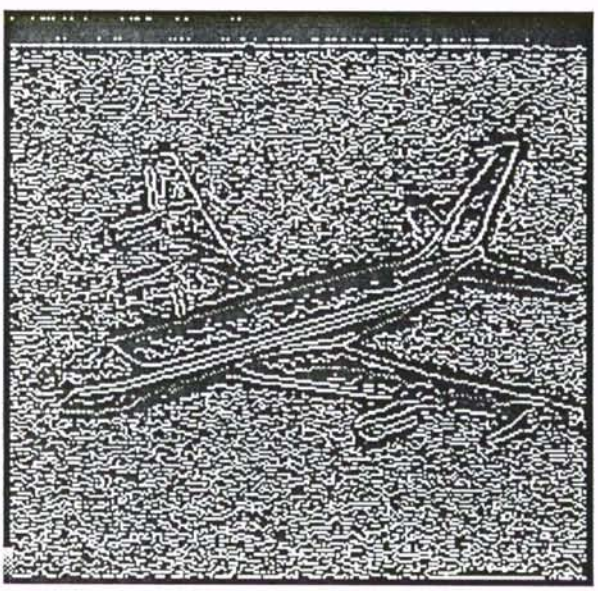
(a)

(b)

Figure 33: Jet image. (a) original image, (b) slope.



(c)



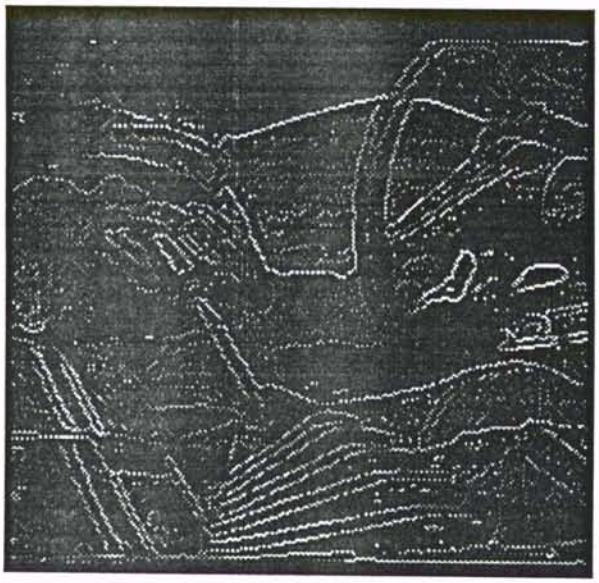
(d)

Figure 33: (continued) Jet image. (c) stepsize, (d) scale for estimating stepsize.

(a)

(b)

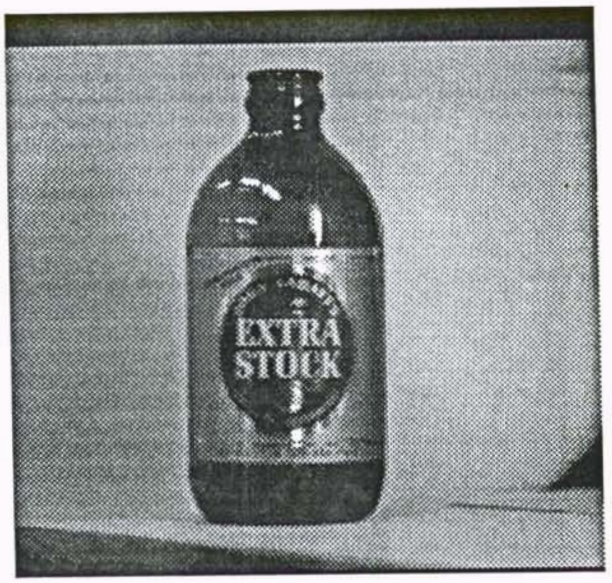
Figure 34: Driver image. (a) original image, (b) slope.



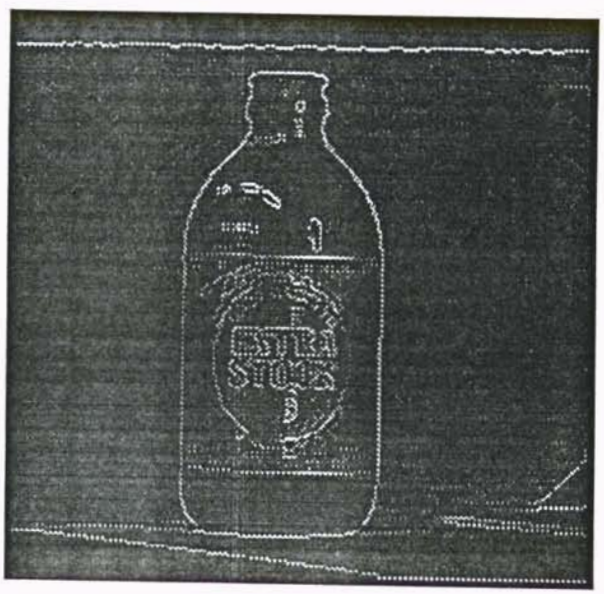
(c)

(d)

Figure 34: (continued) Driver image. (c) stepsize, (d) scale for estimating stepsize.



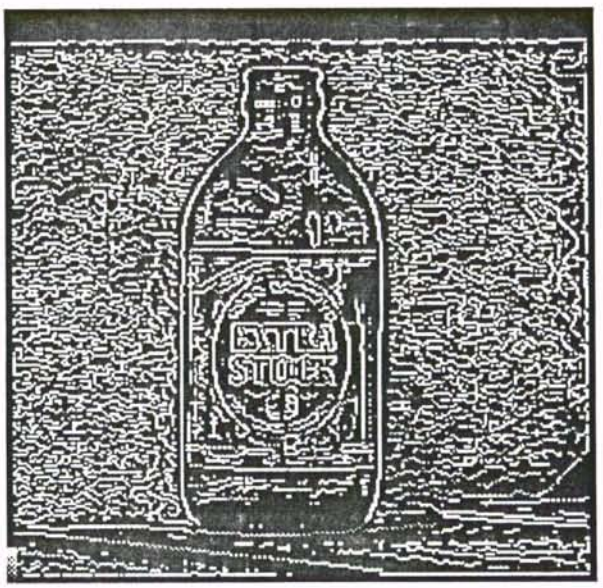

(a)



(b)

Figure 35: Bottle image. (a) original image, (b) slope.

(c)



(d)

Figure 35: (continued) Bottle image. (c) stepsize, (d) scale for estimating stepsize.

# 6. EDGE REPRESENTATION

The problem of how to represent a set of points which have been determined to lie on an edge is a challenging one. In this chapter a representation of this problem is given and two methods that have been proposed for solving it are discussed. A new algorithm is then presented which solves some of the problems present in the previous algorithms. The concept of curvature is basic to this discussion, so formulas for approximating curvature of discrete curves are presented and evaluated.

One of the difficulties in the representation of curves is the problem of scale. This can refer to the degree of smoothing that is performed on an image before a process such as edge detection is applied. The term is also applied to a contour which has been identified in an image or to a surface which has been fitted to an image. The degree to which these have been smoothed is also referred to as scale.

## 6.1 Minimum Energy Contours

At present it is common to use more than one scale to detect edges or represent contours. Rather than combine the information derived at the different scales into a unified "best" representation of the information, another approach is to attempt to keep the information at different scales available so that higher level processes can use the most meaningful representation. This was one of the goals of Kass, Witkin, and Terzopoulos [17] when they developed their Active Contour Models (called snakes). They developed a controlled continuity spline which can be operated upon by internal contour forces, image forces, and external forces which are supplied by an interactive user, or potentially by a higher level process.

In their work, Kass *et al.*, represented a contour by a vector, $\mathbf{v}(s) = (x(s), y(s))$ having the arc length, $s$, as parameter.[1] They defined an energy functional of the contour and described a method for finding contours which correspond to local minima of the functional. The energy functional is written as:

$$\begin{aligned} E^*_{snake} &= \int_0^1 E_{snake}(\mathbf{v}(s))ds \\ &= \int_0^1 E_{int}(\mathbf{v}(s)) + E_{image}(\mathbf{v}(s)) + E_{con}(\mathbf{v}(s))ds \end{aligned} \tag{1}$$

$E_{int}$ represents the internal energy of the contour due to bending or discontinuities, $E_{image}$ is the image forces, and $E_{con}$ is the external constraints. The image forces can be due to various events. The ones presented by Kass *et al.*, are lines, edges, and terminations. The internal spline energy is written:

$$E_{int} = (\alpha(s)|\mathbf{v}_s(s)|^2 + \beta(s)|\mathbf{v}_{ss}(s)|^2)/2 \tag{2}$$

The above equation contains a first-order term which will have larger values where there is a gap in the curve, and a second-order continuity term which will be larger where the curve is bending rapidly. The values of $\alpha$ and $\beta$ at a point determine the extent to which the contour is allowed to stretch or bend at that point. The relative sizes of $\alpha$ and $\beta$ can be chosen to control the influence of the corresponding constraints. For instance, a large value of $\beta$ would make the second-order continuity term larger than the other terms, thus the minimum value of E* would occur when the curve is smoother, approaching a circle for a closed contour, and a straight line for a contour which is not closed. If $\alpha$ is 0 at a point, a discontinuity can occur at that point, while if $\beta$ is 0, a corner can develop, because large values of these terms would not be included in the total. The minimum energy contour was determined using techniques of variational calculus.

[1] Lower-case, bold letters like **v** will be used to denote vectors when they are interpreted as points, while lower-case, bold letters with an arrow above ($\vec{\mathbf{u}}$) will be used when the quantity represented is a vector from one point to another.

Amini, Tehrani, and Weymouth [1] point out some of the problems involved in this method of solution and propose that the contour having minimum energy be determined using dynamic programming rather than variational calculus. This allows the introduction of constraints that cannot be violated, called *hard* constraints, as well as the first- and second-order continuity constraints which are inherent in the problem formulation. These latter are known as *soft* constraints because they are not satisfied absolutely, only to a certain degree.

At this point it would be meaningful to examine the advantages and disadvantages of the problem formulation itself, and with the two proposed methods of solution. A "+" by an item on the list indicates that this is a positive feature, while a "–" indicates that this is a drawback. First we will consider advantages and disadvantages which apply to the statement of the problem and to both methods of solution.

+ A closed contour which is placed around an object can span gaps in the edge map. Similarly, if an object with texture has edges which make it appear as several smaller objects, the contour can outline the object as a whole, giving a continuous edge contour for the entire object. See Figure 36 for an example of this.

+ Information from a higher level process can be used to set the values of $\alpha$ and $\beta$, allowing corners where they are expected, for example, and seeing how that affects the contour obtained.

– No guidelines are given in either method for determining the values of $\alpha$ and $\beta$. Also both methods apparently use the same value for $\alpha$ and $\beta$ at every point, and no discussion or examples are given explaining how changing these values affects the contours. It happens that the values are critical, and must be chosen carefully to obtain meaningful results.

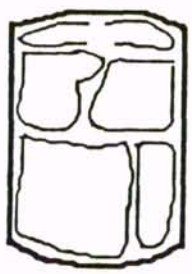


Figure 36: Contour outlines entire object, rather than following texture edges on the surface of the object.

- Related to the previous item, if $\beta$ is constant, corners will not be well defined. There is also a problem if points are far apart and a corner falls between two points on a contour.

- The first derivative term in Equation 2 is approximated by a finite difference, $|\mathbf{v}_s|^2 \approx (x_i - x_{i-1})^2 + (y_i - y_{i-1})^2$. This is equivalent to minimizing the distance between points, and has the effect of causing the contour to shrink.

- Points can move along the contour as well as perpendicular to it, thereby allowing points to bunch up in segments of the contour where the image forces are higher. The hard constraints provided for in the method of Amini *et al.*, can be used to minimize this problem.

The following list applies to the Kass method only.

+ Forces can travel large distances along the contour in one iteration, allowing faster convergence.

- Image forces and constraints need to be differentiable in order to guarantee convergence. Thus it is not possible to include hard constraints, such as minimum

distance between points.

- Intermediate results are not meaningful. The contour does not smoothly approach the minimum value. It was for this reason that the name snakes was given to the contours.

The next list gives characteristics of the Amini method.

+ Hard constraints can be introduced into the method.

+ Points are moved on the discrete grid, as opposed to the Kass method which computes point coordinates as real numbers, allowing points to fall between the discrete coordinates.

+ This method is numerically stable.

- Memory requirements are large, being $O(n * m^2)$, where $n$ is the number of points on the contour and $m$ is the number of possible locations to which a point may move in a single iteration.

- The method is very slow, being $O(n * m^3)$.

## 6.2 Curvature Estimation

Both Kass *et al.*, and Amini *et al.*, approximate the derivatives in Equation 2 by finite differences. If $\mathbf{v}_i = (x_i, y_i)$ is a point on the contour, the following approximations are used:

$$\left|\frac{d\mathbf{v}_i}{ds}\right|^2 \approx |\mathbf{v}_i - \mathbf{v}_{i-1}|^2 = (x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 \tag{3}$$

and

$$\begin{aligned}\left|\frac{d^2\mathbf{v}_i}{ds^2}\right|^2 &\approx |\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}|^2 \\ &= (x_{i-1} - 2x_i + x_{i+1})^2 + (y_{i-1} - 2y_i + y_{i+1})^2 \end{aligned} \tag{4}$$

Note that two assumptions have been made here. The first assumption is that the points on the contour are evenly spaced. If the points are evenly spaced, then the index $i$ can be considered as an approximation of arc length. In that case the expression in Equation 3 should be divided by $d^2$, where $d$ is the distance between points, and that of Equation 4 by $d^4$. The values of $\alpha$ and $\beta$ can be chosen to include these factors in order to reduce computation. Then $d$ will have to be made available to any higher order process which is attempting to assign values to $\alpha$ and $\beta$ automatically.

If the points are not evenly spaced, the first derivative term will be incorrect by a factor of ${d_i}^2$, where $d_i$ is the distance between points $i$ and $i-1$. This will cause the first-order continuity term in the energy expression to be larger for points which are farther apart. The second derivative term will also have quite different values, depending on the distances of the two end points from the middle point. Whether this is a desirable or undesirable property will be discussed later.

The second assumption is related to the interpretation of the $|\mathbf{v}_{ss}|$ term. The parameter $s$ is arc length, so the quantity measured is curvature, and has a very intuitive application to curves: corners are points of high curvature and are to be discouraged. However, when the parameter is not arc length, the curvature is given by

$$\kappa = \frac{|x'y'' - x''y'|}{(x'^2 + y'^2)^{3/2}} \tag{5}$$

for a parameter $t$ where $x' = dx/dt$, $x'' = d^2x/dt^2$, $y' = dy/dt$ and $y'' = d^2y/dt^2$. The quantity $|\mathbf{v}_{tt}| = \sqrt{x''^2 + y''^2}$ does not have a clear geometrical interpretation when the parameter $t$ is not arc length.

It is not clear what measure of curvature is the best reflection of the geometric situation depicted by the contour. The mathematical definition of curvature is $d\theta/ds$

where $\theta$ is the angle between the positive $x$-axis and the tangent vector to the curve. This is a coordinate independent measure, as the same value will be obtained for $d\theta$ when any line is substituted for the $x$-axis; thus the measure is invariant under rotation. That is a desirable feature for model matching. Another desirable feature which is not present in curvature is scale invariance. A circle with radius $r$ has curvature $1/r$ at each point. Thus when the radius is doubled, the curvature is halved.

The remainder of this section presents five possible measures of curvature in discrete contours, and discusses the characteristics of each. In order to demonstrate the difference in the results obtained by these different approximations, they were all applied to the two situations displayed in Figure 37 and the results are displayed in Table 3. In each case $\mathbf{v}_{i-1}$ is point $a$, $\mathbf{v}_i$ is point $b$, while $\mathbf{v}_{i+1}$, the third point necessary in the curvature estimate, can be any one of the points $c_1 \ldots c_5$. The first section of the table is the situation in Figure 37a where $a$ and $b$ are on a horizontal or vertical line. The lower section is the case where $a$ and $b$ lie on a diagonal line. When the external angle is $0$, $\pi/2$, or $\pi$, the distances from $b$ to its two neighbors are equal, being 1 for the horizontal case and $\sqrt{2}$ for the diagonal case. When the angle is $\pi/4$ or $3\pi/4$ the two distances are not equal, being 1 and $\sqrt{2}$.

It will be necessary to estimate the value of differentials in the following discussion. The usual convention will be to use the *backward* difference for this estimate. That is, $dx$ at the point $\mathbf{v}_i$ is approximated by $x_i - x_{i-1}$ and is denoted $\Delta x_i$. Occasionally when the backward difference might vary substantially from the *forward* difference, $(x_{i+1} - x_i)$, the *centered* difference will be used instead. It is given by $(x_{i+1} - x_{i-1})/2$. Whenever this is done, it will be pointed out. Similar notation for finite difference estimation of differentials will be used for all variables.

The first possibility for approximating curvature is to apply the definition of cur-

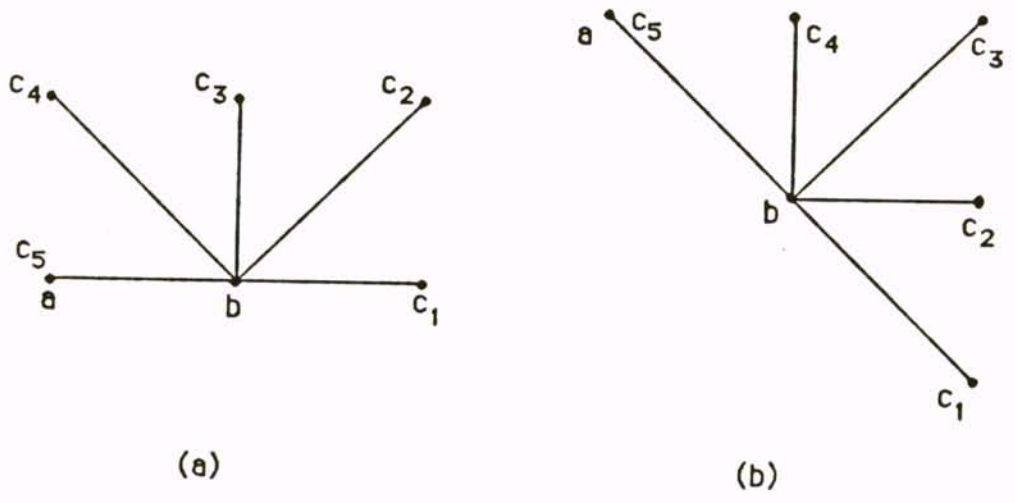



Figure 37: Arrangement of points $a$, $b$, and $c$ for Table 3.

| | (1) | (2) | (3) | (4) | (5) |
|---|---|---|---|---|---|
| c | $(d\theta/ds)^2$ | $\kappa^2$ | $\lvert \mathbf{v}_{ss} \rvert^2$ | $\lvert \vec{u}_i - \vec{u}_{i+1} \rvert^2$ | $\left\lvert \frac{\vec{u}_i}{\lvert\vec{u}_i\rvert} - \frac{\vec{u}_{i+1}}{\lvert\vec{u}_{i+1}\rvert} \right\rvert^2$ |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 0.42 | 0.512 | 0.40 | 1.0 | 0.59 |
| 3 | 2.47 | 8.0 | 2.0 | 2.0 | 2.0 |
| 4 | 3.80 | 64.0 | 2.34 | 5.0 | 3.41 |
| 5 | 9.87 | $\infty$ | 4.0 | 4.0 | 4.0 |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 2 | 0.42 | 0.512 | 0.40 | 1.0 | 0.59 |
| 3 | 1.23 | 4.0 | 1.0 | 4.0 | 2.0 |
| 4 | 3.80 | 64.0 | 2.34 | 5.0 | 3.41 |
| 5 | 4.93 | $\infty$ | 2.0 | 8.0 | 4.0 |

Table 3: Comparison of estimates of the square of the curvature using different methods. The first section of the table is the horizontal situation of Figure 37, the lower section is the diagonal case. Column one gives $(d\theta/ds)^2$, column two is $\kappa^2$ using Equation 5, and column three gives $|\mathbf{v}_{ss}|^2$. Column four is $|\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}|^2$ and column five gives $[\Delta x_i/\Delta s_i - \Delta x_{i+1}/\Delta s_{i+1}]^2 + [\Delta y_i/\Delta s_i - \Delta y_{i+1}/\Delta s_{i+1}]^2$.

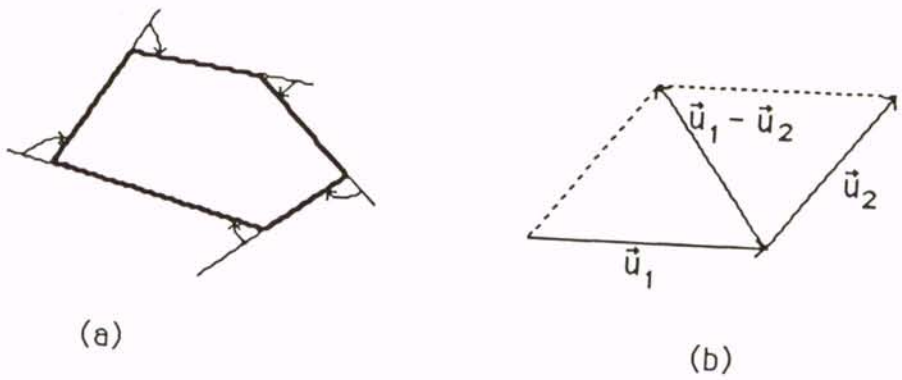



Figure 38: (a) External angles for a closed polygon. Their sum is $2\pi$. (b) The difference of two vectors.

vature directly. If a discrete approximation of $d\theta/ds$ is computed for evenly spaced points, it has the property that it depends linearly on the angle $\Delta\theta$ between the two vectors, $\vec{\mathbf{u}}_i = (x_i - x_{i-1},\, y_i - y_{i-1})$ and $\vec{\mathbf{u}}_{i+1} = (x_{i+1} - x_i,\, y_{i+1} - y_i)$. The formula for $\Delta\theta$ is given by

$$\Delta\theta = \cos^{-1} \frac{\vec{\mathbf{u}}_i \cdot \vec{\mathbf{u}}_{i+1}}{|\vec{\mathbf{u}}_i||\vec{\mathbf{u}}_{i+1}|} = \cos^{-1} \frac{(x_i - x_{i-1})(x_{i+1} - x_i) + (y_i - y_{i-1})(y_{i+1} - y_i)}{\sqrt{[(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2][(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2]}}$$

Given a closed polygon and a direction, $\Delta\theta \approx d\theta$ is the external angle as the circumference is traversed. The sum of the external angles is $2\pi$, as shown in Figure 38a. The centered difference, $(\Delta s_{i+1} + \Delta s_i)/2 = (|\vec{\mathbf{u}}_i| + |\vec{\mathbf{u}}_{i+1}|)/2$, averages the distance from point $i$ to its two neighbors and thus gives the best estimate of $ds$. The smallest value for $\Delta s$ is 1 and the largest value of $\Delta\theta$ is $\pi$; thus values of $\Delta\theta/\Delta s$ all fall in the interval $[0, \pi]$. This is not true for continuous curves, where rapidly bending curves can have very large curvature. However, when a curve is digitized, a limit is placed on the curvature. Although giving intuitively satisfying results, this measure requires a lot of computation, including 5 multiplications, 2 divisions, 2 square roots, and an

inverse cosine. Column one of Table 3 gives values of $(d\theta/ds)^2$. The values are all in the interval $[0, \pi^2]$ when $\Delta s$ is one, and in a smaller interval when $\Delta s$ is larger than one.

Evaluating the expression for curvature in Equation 5 should give results identical to that of $d\theta/ds$ for continuous curves. However, this is not the case for discrete curves. When the angle between $\vec{\mathbf{u}}_i$ and $\vec{\mathbf{u}}_{i+1}$ becomes large, $\Delta x_i$ has a value near $-\Delta x_{i+1}$ and $\Delta y_i$ is near $-\Delta y_{i+1}$. Thus when the centered difference is used to estimate $dx$ and $dy$, these values become very small, giving a value for curvature which is unbounded, as it is for continuous curves. Column two of Table 3, the discrete approximation to Equation 5, is comparable to the other estimates for small angles, but as $(\Delta x_i + \Delta x_{i+1})/2$ and $(\Delta y_i + \Delta y_{i+1})/2$ grow smaller, the estimate of $\kappa^2$ becomes very large.

Converting the parameter to arc length and then computing the second derivative is theoretically equal to the two previous measures for continuous curves. The discrete approximation is given by

$$|\mathbf{v}_{ss}| = \frac{1}{\Delta s}\sqrt{\left[\left(\frac{\Delta x_i}{\Delta s_i} - \frac{\Delta x_{i+1}}{\Delta s_{i+1}}\right)^2 + \left(\frac{\Delta y_i}{\Delta s_i} - \frac{\Delta y_{i+1}}{\Delta s_{i+1}}\right)^2\right]} \tag{6}$$

where $\Delta s$ is $(\Delta s_i + \Delta s_{i+1})/2$. The third column of Table 3 gives the square of the discrete estimate of the second derivative vector. Notice in the diagonal case for column three that the curvature for $c_4$ is larger than for $c_5$, even though the path $a-b-c_5$ actually doubles back on itself, and should intuitively have higher curvature.

Another possible measure of curvature which has the advantage of being computationally efficient is given by the expression in Equation 4. If $\vec{\mathbf{u}}_1$ and $\vec{\mathbf{u}}_2$ are the vectors shown in Figure 38b, this is equivalent to $|\vec{\mathbf{u}}_2 - \vec{\mathbf{u}}_1|^2$. It reflects not only the difference between the directions of the two vectors, but also the difference in length. This produces unintuitive results when the distances between points are not equal.

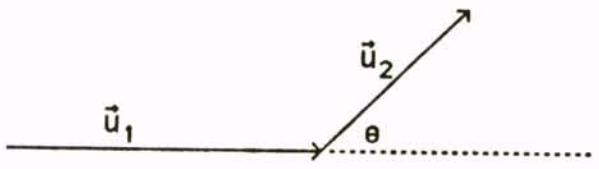



Figure 39: Difference in direction of two vectors.

The difference of two vectors is shown in Figure 38b. The fourth column in the table is the square of the curvature estimate using Equation 4. Notice that in column four, $c_4$ has the largest value for the horizontal case as it does for column three in the diagonal case.

Normalizing the two vectors before taking the difference removes the length differential, and the measure depends solely on relative direction. Thus it will be bounded, with values in the interval $[0, 2]$. The length of $\frac{\vec{u}_{i+1}}{|\vec{u}_{i+1}|} - \frac{\vec{u}_i}{|\vec{u}_i|}$ is given by $2\sin(\theta/2)$ where $\theta$, $0 \leq \theta \leq \pi$, is the difference in direction of the two vectors as shown in Figure 39. Column five of the table gives the values obtained by this formula.

It is interesting that the last three measures are closely related. Multiplying the discrete approximation of $|\mathbf{v}_{ss}|$ by $\Delta s$ gives the difference of the normalized vectors (column five). When the points are evenly spaced, multiplying by $\Delta s$ again gives the expression in Equation 4 (column 4).

There is a sixth method of approximating curvature at a point, that of fitting a circle through the point and its two neighbors (e.g., [34]). The radius of the circle will give a good estimate of the radius of curvature if a circle is a good approximation of the curve through the three points. However, this only gives a reasonable estimate when the angle between the two vectors is large and when the points are evenly spaced

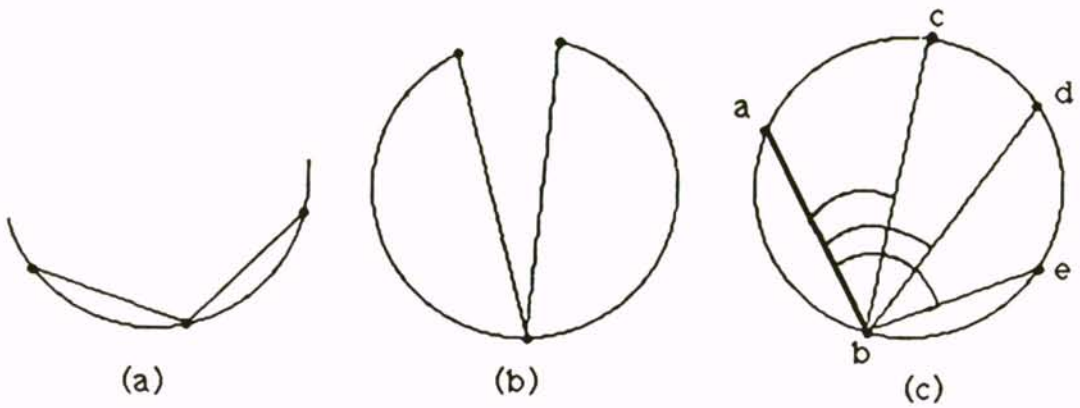



Figure 40: Estimation of curvature by fitting a circle to three points. (a) Angle between vectors is large, hence fit is good. (b) Small angle means a circle is not a good approximation to the curve through the three points. (c) When distance between points is not equal, circles having the same radius will go through $\{a, b, c\}$, $\{a, b, d\}$, and $\{a, b, e\}$.

(Figure 40a). When the angle between the two vectors is small, the circle does not give a good approximation to the curve through the three points, and the curvature estimate will be too small (Figure 40b). If the points are not evenly spaced, very different situations, which do not seem to have the same curvature, will give a circle having the same radius. For example, the sets of points $\{a, b, c\}$, $\{a, b, d\}$, and $\{a, b, e\}$ would have a circle of the same radius fitted through them (Figure 40c) even though the curvature of the curve through the different sets does not appear the same. Thus, this method does not seem to have general enough application to consider here.

## 6.3 Greedy Algorithm

In this section a greedy algorithm will be presented which allows a contour with controlled first and second order continuity to converge on an area of high image energy, in this case edges. This algorithm allows the inclusion of hard constraints as

described by Amini *et al.*, [1] but is much faster than their $O(nm^3)$ algorithm, being $O(nm)$, for a contour having $n$ points which are allowed to move to any point in a neighborhood of size $m$ at each iteration. The algorithm is not guaranteed to give a global minimum, but produced good results on the images for which it was tested.

The quantity being minimized by this algorithm is

$$E = \int (\alpha(s)E_{cont} + \beta(s)E_{curv} + \gamma(s)E_{image})\, ds \tag{7}$$

The form of this equation is similar to Equation 1. The first and second terms are first- and second-order continuity constraints and will be described in detail later. They correspond to $E_{int}$ in Equation 1. The last term measures some image quantity such as edge strength or intensity and is the same as the middle term of Equation 1. No term for external constraints was included, although it would be possible to do so.

The proposed algorithm is iterative, as are those of Kass and Amini. During each iteration, a neighborhood of each point is examined and the point in the neighborhood giving the smallest value for the energy term is chosen as the new location of the point. Only closed contours are being considered, so all index arithmetic is modulo $n$. The first-order continuity term uses the value of $\mathbf{v}_{i-1}$, for which a new value has already been computed during the current iteration when $i > 0$. The curvature term uses $\mathbf{v}_{i-1}$ and $\mathbf{v}_{i+1}$. The latter has the value of the previous iteration; thus one old and one new point is used in the evaluation of the curvature term. For $i = 0$, only old values are used. For this reason $\mathbf{v}_0$ is processed twice, once as the first point in the list, and once as the last point. This helps make its behavior more like that of the other points.

The parameters $\alpha$, $\beta$, and $\gamma$ are used to balance the relative influence of the three terms. In the examples given below, $\alpha = 1$, $\beta$ is either 0 or 1, and $\gamma = 1.2$, so that

the image gradient will have more importance than either of the continuity terms in determining where points on the contour move.

Evaluation of the first term in Equation 7, the continuity term, presents some difficulties. Using $|\mathbf{v}_i - \mathbf{v}_{i-1}|^2$ causes the curve to shrink, as this is actually minimizing the distance between points. It also contributes to the problem of points bunching up on strong portions of the contour. These effects are even worse with a greedy algorithm where each point is moved based on local considerations. The tendency is for points to always be moved nearer the previous point, which also moves a point farther from the following point. This causes a chain reaction, moving all points toward the previous one. In observing the behavior of the given algorithms, it became apparent that a term which encouraged even spacing of the points would reflect the desired behavior of the contours more than one which caused shrinking. The original goal of encouraging first-order continuity is still satisfied. Thus the algorithm presented here uses the difference between the average distance between points, $\bar{d}$, and the distance between the two points under consideration: $\bar{d} - |\mathbf{v}_i - \mathbf{v}_{i-1}|$. Thus points having distance near the average will have the minimum value. The value is normalized by dividing by the largest value in the neighborhood. At the end of each iteration a new value of $\bar{d}$ is computed.

The second term in Equation 7 is curvature. Since the formulation of the continuity term causes the points to be relatively evenly spaced, $|\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}|^2$, the formula in column four of Table 3, gives a reasonable estimate of curvature multiplied by a constant. The constant term is not significant since this term, like the continuity term, is normalized by dividing by the largest value in the neighborhood, giving a number from 0 to 1. This formula has the advantage that it is the most computationally efficient of the ones discussed in the previous section.

The third term in Equation 7, $E_{image}$, is the image force which is gradient magni-

tude. Gradient magnitude at each point in the image is input as an eight bit integer, with values 0–255. There is a significant difference between a point with gradient magnitude 240, and one having magnitude 255. This is not reflected when the values are normalized by division by 255. Thus the maximum and minimum gradient in each neighborhood is determined, and $(min - mag)/(max - min)$ is used for the normalized edge strength term. This gradient magnitude term is negative so that points with large gradient will have small values. If $max - min < 5$ then $min$ is given the value $max - 5$. This prevents large differences in the value of this term from occurring in areas where the gradient magnitude is nearly uniform. For example, when all points in the neighborhood being examined had values 47, 48, and 49, the gradient magnitude term would be 0, -0.5, or -1.0 for points with essentially the same gradient magnitude. Thus a point would have a strong tendency to stay at a point with gradient magnitude 49, even though it is not a strong edge point. Having a minimum of 5 in the denominator would give -0.6, -0.8, or -1.0 for the gradient term, more accurately reflecting the similarity of the points. Near an edge this situation does not normally arise, but if the contour has points that begin fairly far from the final edge or span regions where there are gaps in the edge, points on the contour may resist moving without this constraint.

At the end of each iteration, a step is included which determines the curvature at each point on the new contour, and if the value is a curvature maximum, sets $\beta_i = 0$ for the next iteration. This step functions as a primitive high level process giving feedback to the energy minimization step. Curvature is computed at each of the $n$ points by $[\Delta x_i/\Delta s_i - \Delta x_{i+1}/\Delta s_{i+1}]^2 + [\Delta y_i/\Delta s_i - \Delta y_{i+1}/\Delta s_{i+1}]^2$. This is the measure given in column five of Table 3, which is related to the angle between the vectors. This formula requires more computation than the one used in the main computation of the algorithm, but is computed fewer $(n)$ times and is used because determining a

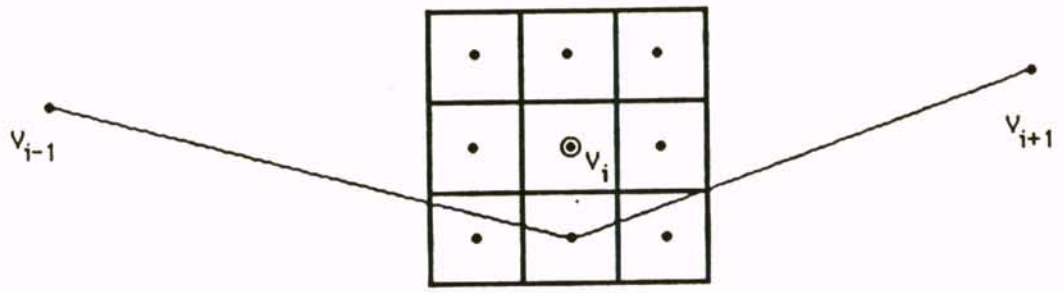



Figure 41: The energy function is computed at $\mathbf{v}_i$ and each of its eight neighbors. The point before and after it on the contour are used in computing the continuity constraints. The location having the smallest value is chosen as the new position of $\mathbf{v}_i$.

meaningful threshold is easy. Non-maxima suppression is then performed on curvature values along the contour, and curvature maxima points having curvature above a threshold are considered corner points for the next iteration. A further condition for designating a point as a corner is that the gradient magnitude must be above some minimum value. The reason for this final condition is that as the contour begins to converge toward the final location, a point which is not near an edge, in a region where the gradient is relatively uniform, may move more slowly, causing a corner to form. These will not be considered corners, because the corner disappears as the contour forces from the movement of neighboring points increase and pull the point toward its final position. Thus $\beta$ is set equal to zero at the points satisfying the above three conditions, allowing a corner to form there, and reducing the curvature in the segments between these points.

Figure 41 demonstrates how the algorithm works. The energy function is computed for the current location of $\mathbf{v}_i$ and each of its neighbors. The location having the smallest value is chosen as the new position of $\mathbf{v}_i$. $\mathbf{v}_{i-1}$ has already been moved to its

new position during the current iteration. Its location is used with that of each of the proposed locations for $\mathbf{v}_i$ to compute the first-order continuity term. The location of $\mathbf{v}_{i+1}$ has not yet been moved. Its location, along with that of $\mathbf{v}_{i-1}$, is used to compute the second-order constraint for each point in the neighborhood of $\mathbf{v}_i$.

## 6.4 Pseudo-Code for Greedy Algorithm

Index arithmetic is modulo $n$.

Initialize $\alpha_i$, $\beta_i$, and $\gamma_i$ to 1 for all $i$.

**do**

/* loop to move points to new locations */

**for** $i = 0$ **to** $n$ /* point 0 is first and last one processed */

$E_{min} = \text{BIG}$

**for** $j = 0$ to $m - 1$ /* $m$ is size of neighborhood */

$E_j = \alpha_i E_{cont,j} + \beta_i E_{curv,j} + \gamma_i E_{image,j}$

**if** $E_j < E_{min}$ **then**

$E_{min} = E_j$

$jmin = j$

Move point $\mathbf{v}_i$ to location $jmin$

**if** $jmin$ **not** current location, $ptsmoved+ = 1$ /* count points moved */

/* process determines where to allow corners in the next iteration */

**for** $i = 0$ **to** $n - 1$

$c_i = \left| \frac{\vec{\mathbf{u}}_i}{|\vec{\mathbf{u}}_i|} - \frac{\vec{\mathbf{u}}_{i+1}}{|\vec{\mathbf{u}}_{i+1}|} \right|^2$

**for** $i = 0$ **to** $n - 1$

**if** ($c_i > c_{i-1}$ **and** $c_i > c_{i+1}$ /* **if** curvature is larger than neighbors */

**and** $c_i > threshold1$ /* **and** curvature is larger than threshold */

**and** $mag(\mathbf{v}_i) > threshold2$) /* **and** edge strength is above threshold */

**then** $\beta_i = 0$ /* relax curvature at point $i$ */

**until** $ptsmoved < threshold3$

## 6.5 Experimental Results

In order to demonstrate the performance of the algorithm described in the previous section, results are given for the greedy algorithm developed above, for the original variational calculus solution and for the dynamic programming algorithm. These programs were run on one synthetic image, a Square (Figure 42), and three real images, Box (Figure 43), Bottle (Figure 44), and Cup (Figure 45). The Cup image tested the behavior of the algorithm when the contour spanned a region where the edge was weak or missing. The initial contour for the Square was produced by the edge linking algorithm developed in Chapter 3, thus was quite good to start with. In all the image figures, the points on the contour that satisfied the conditions of high curvature are marked with larger squares. At these points the second-order continuity restraint was relaxed. The neighborhood examined at each point consisted of the point itself and its eight neighbors. Thus the neighborhood size, $m$, was 9. In the image figures, (a) shows the beginning contours, all of which had 40-60 points spaced a distance of approximately 4-6 pixels apart. The threshold for setting $\beta = 0$ was 0.25, corresponding to approximately 29°.

Part (b) shows the result of allowing the original contour to converge to the edge around the object using the variational calculus method proposed by Kass *et al.*

Part (c) shows the result of the dynamic programming algorithm for the four pictures. In order to reduce the tendency to bunch up at strong points on the contour, one of the hard constraints prohibited movement perpendicular to the direction of maximum gradient. This did not prevent the points not currently on the edge from moving toward a strong edge point which was not the nearest point to the current location, but did prevent edges moving along the contour to higher points once they had reached the edge. Movement along the contour also extended the convergence time when this constraint was not included. The threshold given was the number of

points which moved during the iteration. Usually the number of points being moved in each iteration dropped sharply when the contour approached the edge location. Notice that the edge points are more closely spaced on the strong portions of the contour while in locations like the bottom of the cup handle there are no points.

Part (d) in each figure shows the results of the greedy algorithm. The results achieved by all three of the methods presented are comparable, one giving slightly better results in one image, while a diferent method gives better results in another image. The greedy algorithm has removed some of the small jogs from the inside of the square to the outside, but the dynamic programming algorithm has removed more of them. The original contour was very good, so there was very little change using any of the algorithms. Corners are not set with the Kass method, so it gives contours that are more rounded at the corners. The results on the Box are almost identical for the three algorithms, with the upper left edge being better with the greedy algorithm, while the upper right edge is slightly better with the dynamic programming. In the Bottle image, two points become very close together at the top and at the right-hand side with the dynamic programming, but remain evenly spaced in the greedy algorithm because of the different form of the first-order continuity constraint. The edge points do not follow the neck of the bottle as well in the greedy algorithm. As expected, the contour does not follow the right side of the cup well with any of the methods. Where the cup edges are not strong, points belonging to the background appear to be the points converged to with the greedy algorithm. All three converged to the shadow edge at the right-hand bottom corner of the cup rather than to the cup itself, since that was the first edge encountered as the contour approached the cup.

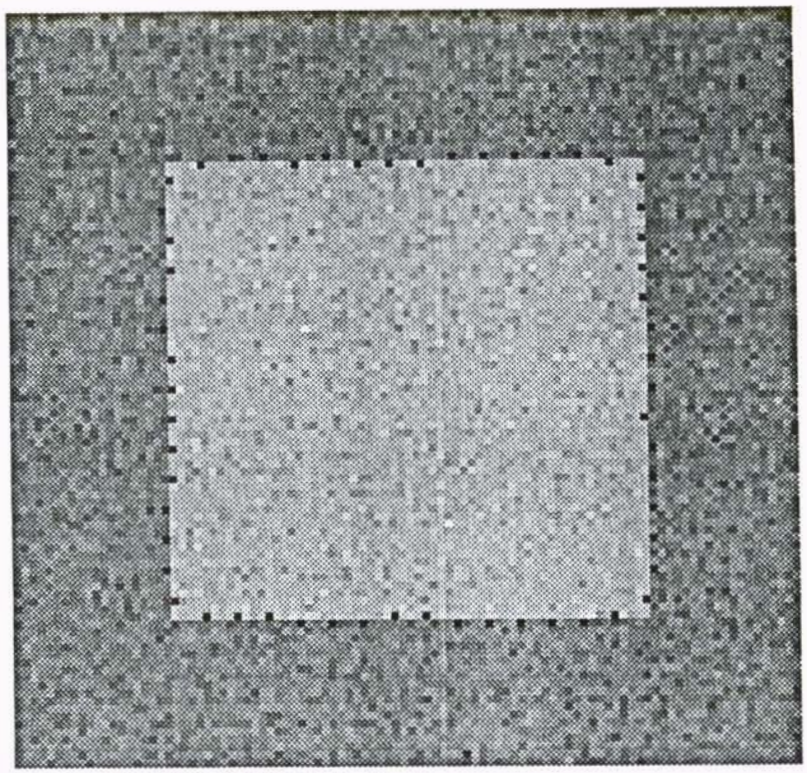
Table 4 gives the number of points in each contour, the threshold used for convergence, the user times in seconds, the number of iterations required to converge, and the number of points of high curvature at which the second-order continuity con-

| | | | Greedy algorithm | | | Dynamic Prog. | | | Variational Calc. | |
|---|---|---|---|---|---|---|---|---|---|---|
| image | size | th. | secs. | iter. | cor. | secs. | iter. | cor. | secs. | iter. |
| square | 58 | 3 | 0.250 | 2 | 3 | 12.162 | 3 | 4 | .350 | 4 |
| box | 56 | 1 | 1.867 | 15 | 5 | 25.157 | 6 | 8 | .466 | 4 |
| bottle | 50 | 1 | 1.217 | 11 | 4 | 29.465 | 8 | 5 | 1.499 | 12 |
| cup | 46 | 3 | 0.700 | 7 | 3 | 34.153 | 10 | 6 | .300 | 4 |

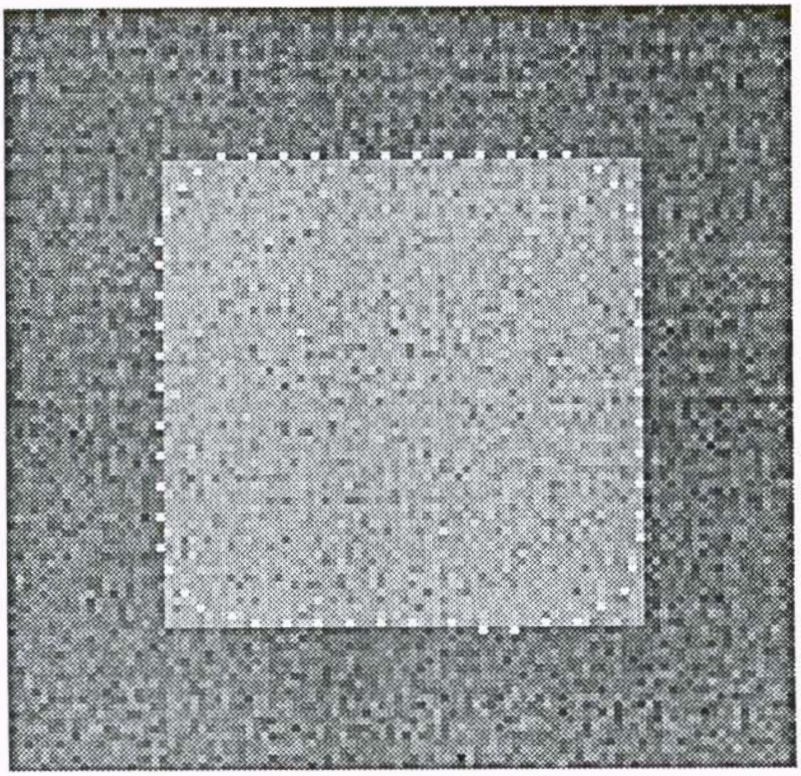
Table 4: Comparison of runtime in seconds, number of iterations, and number of second-order discontinuities (corners) marked for the greedy, dynamic programming, and variational calculus methods.

straint was relaxed by setting $\beta$ to 0. The algorithms were implemented in C on a Harris HCX9 minicomputer. Using the greedy algorithm, the speedup over dynamic programming was significant in all cases, varying from a factor of 13 for the Box, to 48 for the Cup and Square. Neither method was significantly better in the number of iterations required, with the Square and Cup having fewer iterations with the greedy algorithm, while the Box and Bottle required fewer iterations with the dynamic programming algorithm. The results of the contours obtained with the greedy algorithm are at least as good as those of the dynamic programming algorithm, and the run times are much better. The variational calculus approach required time comparable to the greedy method for each iteration, but usually converged in fewer iterations. If values of $\beta$ were allowed to change between iterations, the inverse of a pentadiagonal matrix would need to be computed, slowing its speed.

Figure 46 shows a sequence of images produced as the contour converges to the edge of the bottle, using the greedy algorithm. The edges all move smoothly toward the bottle except for one point at the right-hand bottom corner which is initially stationary, then as the curvature and continuity energy becomes large in that area it begins to move as well. In the final image the contour has settled nicely around the edges of the bottle.

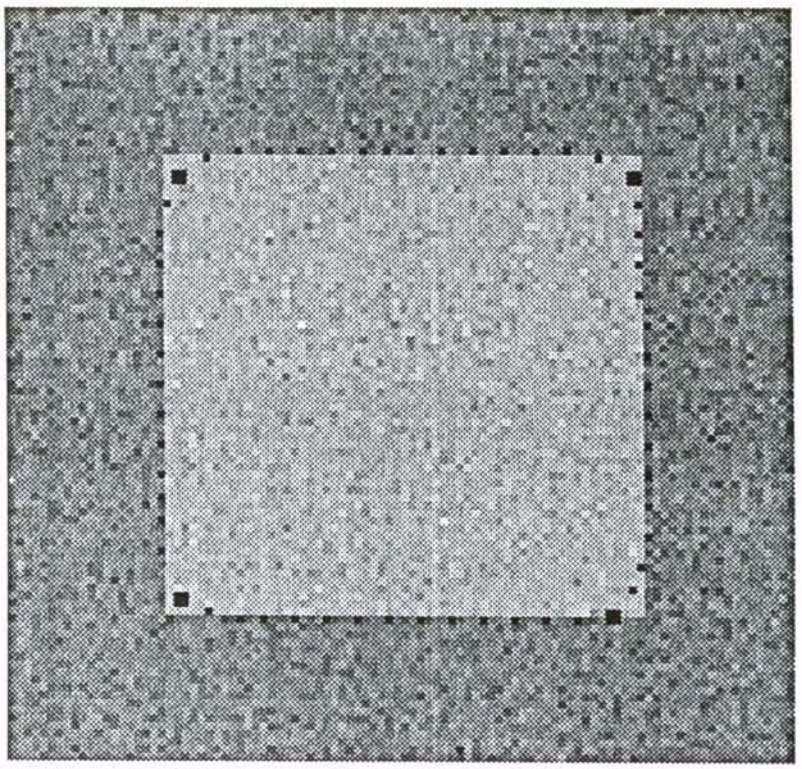


(a)

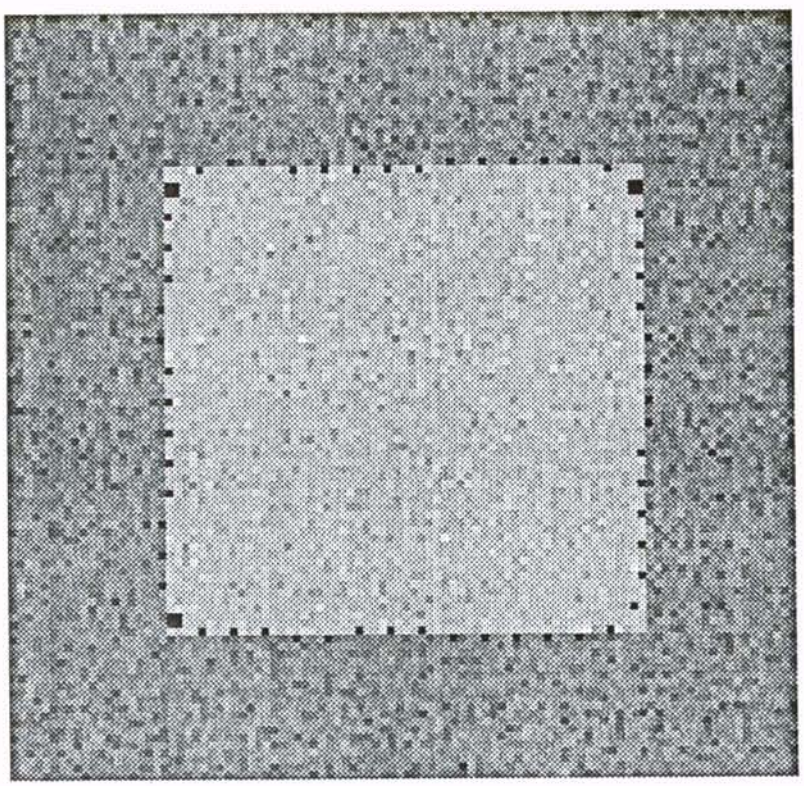


(b)

Figure 42: Square. (a) Original contour, (b) Kass method.

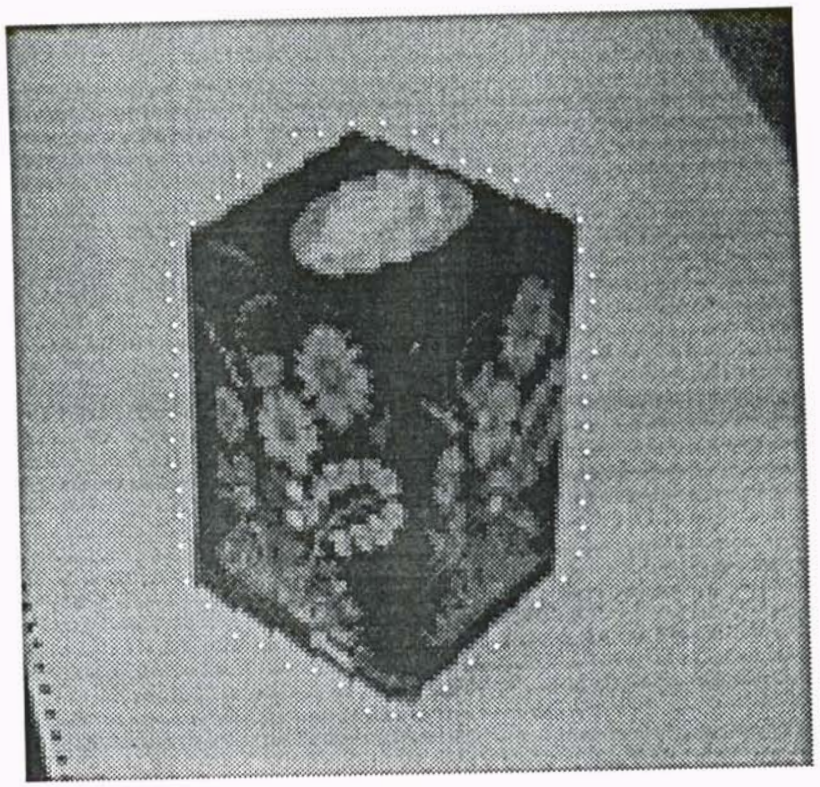


(c)

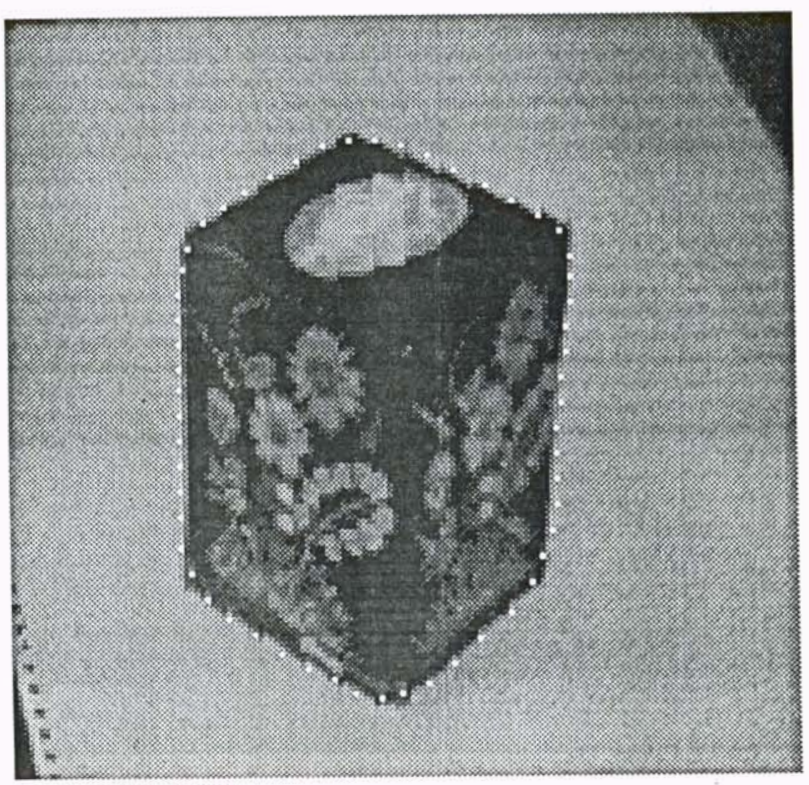


(d)

Figure 42: (continued) Square. (c) Dynamic programming algorithm, (d) Greedy algorithm.

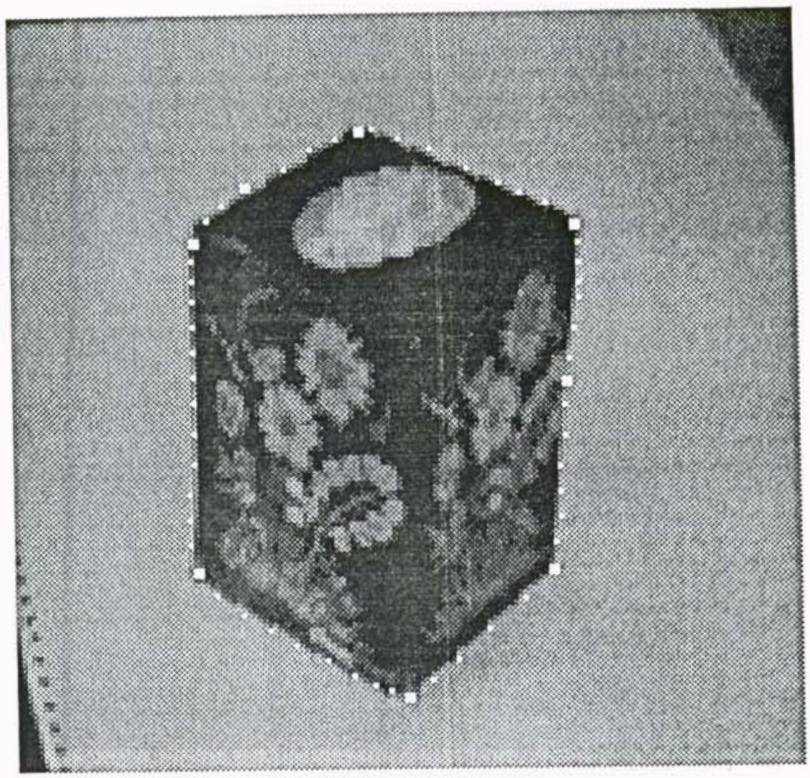


(a)

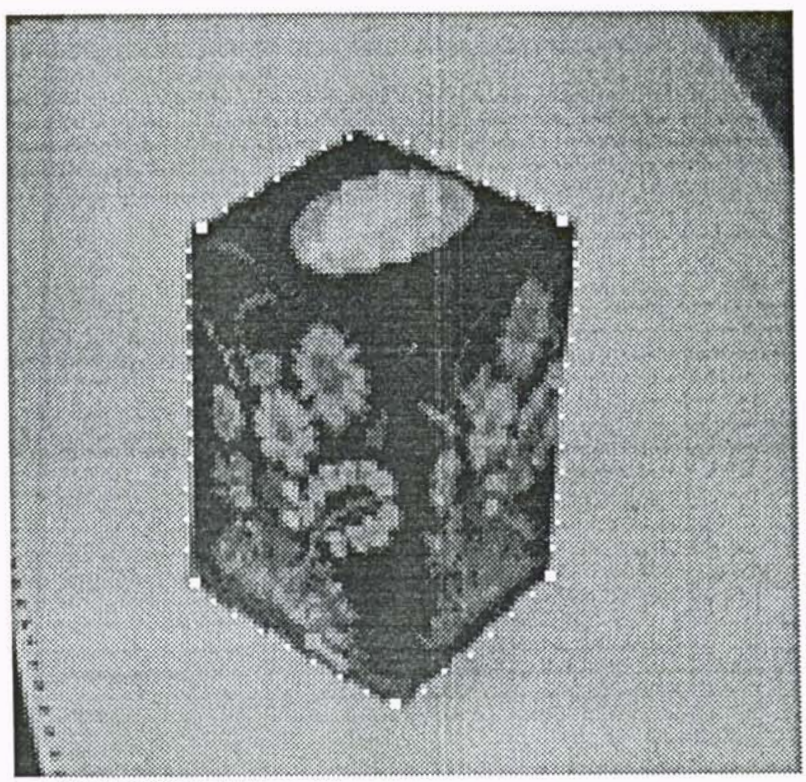


(b)

Figure 43: Box. (a) Original contour, (b) Kass method.



(c)



(d)

Figure 43: (continued) Box. (c) Dynamic programming algorithm, (d) Greedy algorithm.


(a)


(b)

Figure 44: Bottle. (a) Original contour, (b) Kass method.


(c)


(d)

Figure 44: (continued) Bottle. (c) Dynamic programming algorithm, (d) Greedy algorithm.

(a)

(b)

Figure 45: Cup. (a) Original contour, (b) Kass method.

(c)

(d)

Figure 45: (continued) Cup. (c) Dynamic programming algorithm, (d) Greedy algorithm.

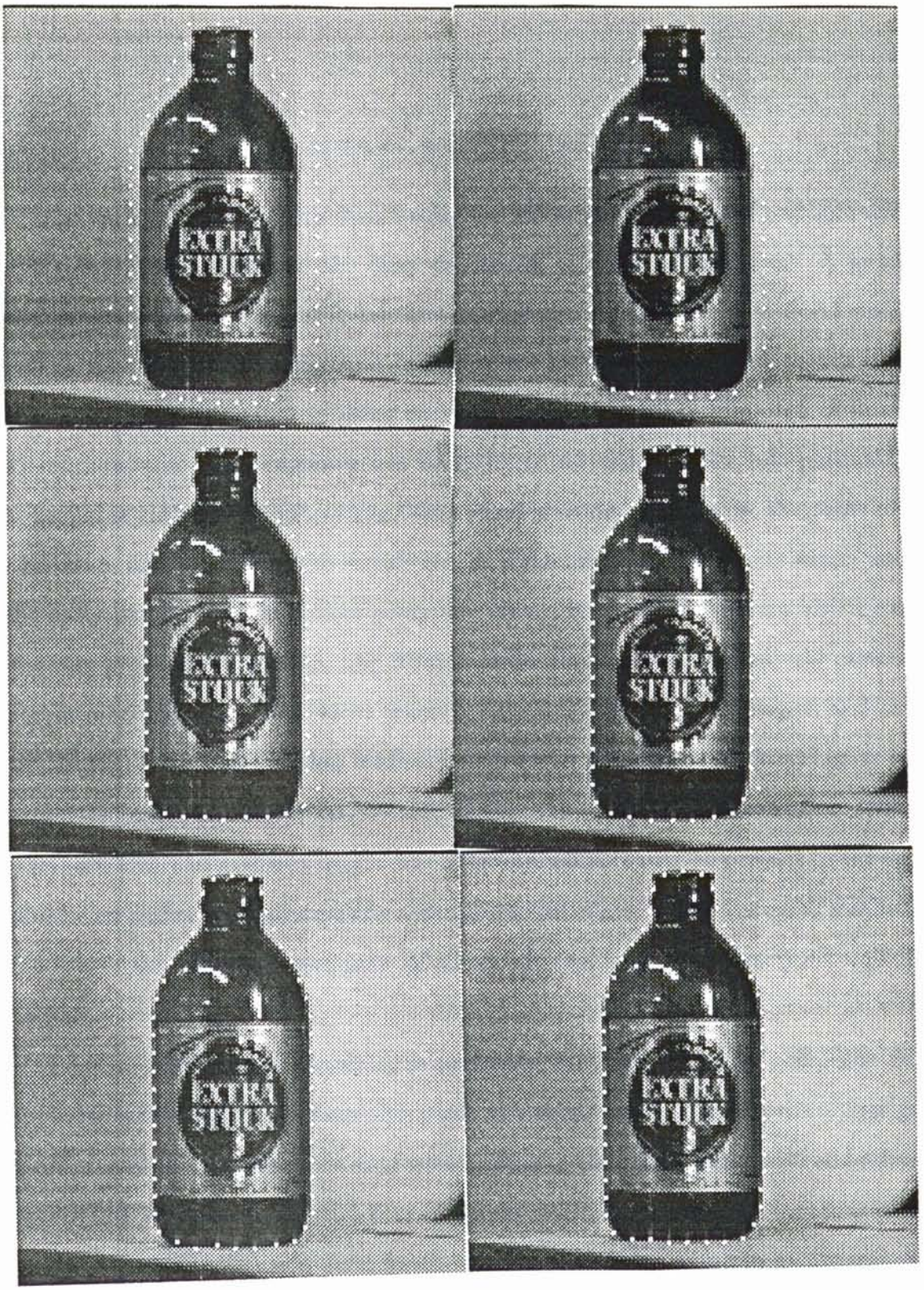



Figure 46: Sequence showing convergence of a contour to edges of bottle using the greedy algorithm.

# 7. SUMMARY AND FUTURE WORK

The work presented in this dissertation has investigated the use of multiple scales, or degrees of smoothing, in the early processing of visual information. A number of techniques that are useful in low level image processing have been developed, as well as the theoretical background justifying the methods used. This theory is also of use to others interested in using multiple scales in the interpretation of images. Throughout the work, emphasis has been placed on using all the information that is available rather than extracting some small portion, perhaps in the form of an edge map, and discarding the remainder, as is commonly done. David Marr, one of the early pioneers in the field of computer vision, insisted that human vision was a one-way process, with information from lower level processes progressively passed to higher level processes [27]. More recently this idea has been challenged, and there are those [17] who believe that feedback, or transfer of information from higher level processes back to lower levels, occurs. Thus, the lower level information such as edge direction and gradient magnitude at several scales could be present in the same process as higher level information about potential edges. These thoughts motivated the use, in the work presented here, of all information available whenever it would be helpful.

First an edge linking algorithm is presented that uses the data computed from application of a gradient of Gaussian mask at a single scale. This algorithm uses gradient magnitude and direction of greatest intensity change together with smoothness constraints to define sequential lists of points which comprise edge contours in the image. This is a more useful form for the data than the edge map commonly returned by edge detectors. While this gave good edges, the primary reason for its development

was to extend it to use multiple scales. By switching to a finer scale whenever an edge faded out, it was possible to fill in gaps in edge contours and produce longer, more coherent edge contours. However, the effects of delocalization of edge points, caused by the smoothing at different scales, created occasional jogs in the contours. This observation led to an improved algorithm which combined the information from the multiple scales at an earlier level and consequently was able to remove most of the delocalization effects.

The introduction of the problem of delocalization, together with decisions which had to be made about the size of neighborhood to be considered when making correspondence between edge points detected at different scales, led to a theoretical analysis of the behavior of edge points detected by the gradient of Gaussian operator. The amount of movement which occurs in the smoothing step was examined as a function of $\sigma$, the smoothing parameter. The edge models examined were combinations of step edges, which are the ones displaying the most extreme movement. Thus these edges set an upper limit on the distance that an edge point can move. It was determined that for most cases the movement is limited to a distance equal to $\sigma$, and that this maximum movement occurs under narrowly limited conditions, the movement being considerably smaller under other conditions. The one case where this limit does not apply is when two adjacent edges have opposite parity and unequal contrast. In that case the weaker edge can exhibit extreme movement. However, the gradient magnitude of such an edge declines rapidly; thus the edge will not be detected when it has moved far from its original location.

As the degree of smoothing increases, and the detected edge moves, the magnitude of the response to the gradient operator at all gradient maxima decreases. However, the rate of decrease is not the same in all conditions (e.g., the weak edge of a pulse has a more rapid rate of decrease than some others). Multiplying the gradient operator

by a factor of $\sqrt{2\pi}\,\sigma$ gives an operator called the *normalized* gradient of Gaussian, which has more interesting behavior than the original gradient of Gaussian. While examining this new operator, it was determined that information about the slope and width of isolated edges can be estimated. In addition, it is possible to obtain information about edge interaction by examining the behavior of the response to the gradient operator as $\sigma$ increases. These data can be used to characterize an edge as to steepness, width, the distance to nearby edges, and the relative parity of the nearby edges.

Once having obtained a good list of edge points the next level of processing included extracting information about the contour as a whole. The points obtained by the edge linking process can be used as initial data for a process which treats the contour as a unit, and applies constraints such as smoothness and continuity. By varying the parameters, the contour can be made to conform closely to the actual edge points, or to exhibit some other properties such as small curvature. The latter requirement could be used to replace a jagged edge by one having a much straighter profile, which would be more useful in model matching. An efficient algorithm for solving this problem was presented. The method developed here allows the inclusion of constraints on the solution, leading to a more robust process. The introduction of the concept of curvature highlighted the problem of how to approximate curvature when a curve is represented by a set of discrete points. The advantages and disadvantages of a number of different methods of approximation were pointed out.

In summary, the contributions of this research are:

1. Development of an edge linking algorithm using multiple scales.

2. Analysis of movement of edges under Gaussian smoothing and derivation of the scale space equations for staircase and pulse edges.

3. Analysis of the magnitude of the response of edges to the normalized gradient of Gaussian operator and characterization of edges using the results.

4. Development of a greedy algorithm for active contours which combines speed with flexibility. Analysis of discrete curvature approximations.

There are several areas in which the above work could be extended. Most of them are as higher level processes using or interacting with the information provided by the low level processes described here. For example, stereo matching is not reliable when performed on primitive structures, such as image intensities or edge points. Matching of edge contours is much more reliable. The edge linking algorithms developed here give a good contour structure. The characteristics of the edge points determined by using the normalized gradient of Gaussian operator can be used to characterize a contour. A confidence measure can then be attached to matching pairs of contours, based on the similarity in the characteristics of the contours. This could be combined with the other measures such as proximity and shape which are commonly used.

The techniques for handling active contours were primarily developed to facilitate interaction with higher level processes. In that chapter, a simple corner detector was applied in order to determine values of $\beta$, but more sophisticated processes could be developed. These might not only supply information such as values of $\alpha$ and $\beta$ to the contour process, but could analyze the results that changing these values had on the contour.

It would also be interesting to experiment with a process which examines the contours returned by the edge linking algorithm, and attempt to determine which edge contours belong to a composite object. A complete contour could then be placed around all the contours in the group, and an outline of the composite object obtained.

On the theoretical side, it would be interesting to extend the type of analysis performed in Chapters 4 and 5 to other edge models. These models might include the hyperbolic tangent proposed by Nalwa and Binford [30], or the roof, which is two adjacent ramp edges having opposite parity.

# APPENDIX

The derivation of the asymptote for the stronger edge in a staircase is developed as follows. The equation of the staircase convolved with the second derivative of the Gaussian is

$$s''_{\sigma,a}(x) = b\, g'_\sigma(x+a) + g'_\sigma(x-a)$$

Since we seek an equation for the zero-crossings, expand and set this equal to 0.

$$\frac{b(x+a)}{\sigma^3\sqrt{2\pi}} \exp\left(-\frac{(x+a)^2}{2\sigma^2}\right) + \frac{x-a}{\sigma^3\sqrt{2\pi}} \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right) = 0$$

Multiplying both sides by $\sigma^3\sqrt{2\pi}$ and combining $x$ terms

$$x\left[b\exp\left(-\frac{(x+a)^2}{2\sigma^2}\right) + \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right)\right] +$$

$$a\left[b\exp\left(-\frac{(x+a)^2}{2\sigma^2}\right) - \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right)\right] = 0$$

When $\sigma \to \infty$, the exponential terms all approach 1 so in the limit

$$x(b+1) + a(b-1) = 0$$

$$x = \frac{a(1-b)}{1+b}$$

The equation for the asymptote for the stronger edge of a pulse is similar to that for the staircase. The equation for the second derivative of the pulse convolved with the Gaussian becomes

$$b(x+a)\exp\left(-\frac{(x+a)^2}{2\sigma^2}\right) - (x-a)\exp\left(-\frac{(x-a)^2}{2\sigma^2}\right) = 0 \qquad (1)$$

which expands to

$$x\left[b\exp\left(-\frac{(x+a)^2}{2\sigma^2}\right) - \exp\left(-\frac{(x-a)^2}{2\sigma^2}\right)\right] +$$

$$a\left[b\exp\left(-\frac{(x+a)^2}{2\sigma^2}\right)+\exp\left(-\frac{(x-a)^2}{2\sigma^2}\right)\right]=0$$

When $\sigma \to \infty$, the exponential terms go to 1, so the limit is

$$x(b-1)+a(b+1)=0$$

$$x=\frac{a(1+b)}{1-b}$$

When $b = 1$, $x$ is undefined. This is expected because the curve approaches $x = \sigma$ as has already been mentioned; thus there is no vertical asymptote. When $b < 1$, $x$ approaches a finite limit, $a(1 + b)/(1 - b)$ as $\sigma \to \infty$.

To determine the asymptotic behavior of the weaker edge, go back to equation 1 for the second derivative. Rearranging terms

$$\exp\left(\frac{-(x-a)^2}{2\sigma^2}\right)\exp\left(\frac{(x+a)^2}{2\sigma^2}\right)=\frac{b(x+a)}{x-a}$$

Expanding the exponents on the left side and canceling terms gives

$$\exp(\frac{2ax}{\sigma^2})=\frac{b(x+a)}{x-a}$$

Taking the logarithm of both sides

$$\frac{2ax}{\sigma^2}=\ln\frac{b(x+a)}{x-a}$$

or

$$\frac{x}{\sigma^2}=\frac{1}{2a}\ln\frac{b(x+a)}{x-a} \tag{2}$$

Then as $x \to -\infty$, $x/\sigma^2 \to (1/2a)\ln b$. Thus the curve approaches the parabola $x = \frac{\ln b}{2a}\sigma^2$.

Solving equation 2 for $\sigma$ gives the equation of the scale space image for the pulse.

$$\sigma=\left[\frac{2ax}{\ln b(x+a)/(x-a)}\right]^{1/2}$$

This equation is defined when $-\infty < x < -a$ and $a < x < a(1+b)/(1-b)$.

Similarly, the equation giving the scale space image graph for the staircase is

$$\sigma = \left[\frac{2ax}{\ln(b(a+x)/(a-x))}\right]^{1/2}$$

when $-a < x < 0$ and $a(1-b)/(1+b) < x < a$. The right portion of the arch in the graph obtained when this is plotted corresponds to the phantom edge produced by the point of inflection in the original smoothed image rather than to a gradient maximum.

## REFERENCES


[1] Amini, A. A., Tehrani, S., and Weymouth, T. E. Using dynamic programming for minimizing the energy of active contours in the presence of hard constraints. In *Proc. Second International Conference on Computer Vision*, pages 95–99, 1988.

[2] Ashkar, G. P. and Modestino, J. W. The contour extraction problem and biomedical applications. *Computer Graphics and Image Processing*, 7:331–355, 1978.

[3] Ballard, D.H. and Brown, C.M. *Computer Vision.* Prentice Hall, Inc., Englewood Cliffs, N.J., 1982.

[4] Bergholm, F. Edge focusing. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-9(6):726–741, 1987.

[5] Berzins, V. Accuracy of Laplacian edge detectors. *Computer Vision, Graphics, and Image Processing*, 27:195–210, 1984.

[6] Bischof, W. F. and Caelli, T. Parsing scale-space and spatial stability analysis. *Computer Vision, Graphics, and Image Processing*, 42:192–205, 1988.

[7] Canny, J. F. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, November 1986.

[8] Canny, J. F. *Finding Edges and Lines in Images.* Master's thesis, MIT, 1983.

[9] Clark, J. J. Singularities of contrast functions in scale space. In *Proc. First International Conf. on Computer Vision*, pages 988–1000, 1987.

[10] Clark, J. J. Singularity theory and phantom edges in scale space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-10(5):720–727, 1988.

[11] Duda, R. and Hart, P. *Pattern Classification and Scene Analysis.* John Wiley, New York, 1973.

[12] Eklundh, J., Elfving, T. E., and Nyberg, S. Edge detection using the Marr-Hildreth operator with different sizes. In *ICPR-6*, pages 1109–1112, 1980.

[13] Fischler, M. A. and Bolles, R. C. Random sample concensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[14] Fischler, M. A. and Wolf, H. C. Linear delineation. In *Proc. IEEE Conf. on Computer Vision and Image Processing*, pages 351–356, 1983.

[15] Gonzalez, R. C. and Wentz, P. A. *Digital Image Processing.* Addison-Wesley Publishers, Inc., Reading, MA, 1987.

[16] Hildreth, E. C. *Implementation of a Theory of Edge Detection.* Master's thesis, MIT, 1980.

[17] Kass, M., Witkin, A., and Terzopoulos, D. Snakes: Active contour models. In *Proc. of First International Conf. on Computer Vision*, pages 259–269, 1987.

[18] Katz, I. *Coaxial Stereo and Scale-Based Matching.* Technical Report, Laboratory for Computational Vision, Department of Computer Science, University of British Colombia, 1985.

[19] Korn, A. F. Toward a symbolic representation of intensity changes in images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-10(5):610–625, 1988.

[20] Lee, D. and Pavlidis, T. One-dimensional regularization with discontinuities. In *Proc. of First International Conf. on Computer Vision*, pages 572–577, 1987.

[21] Leipnik, R. The extended entropy uncertainty principle. *Inf. Control*, 3:18–25, 1960.

[22] Lowe, D. G. Organization of smooth image curves at multiple scales. In *Proc. Second International Conference on Computer Vision*, pages 558–567, 1988.

[23] Lowe, D. G. *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, Boston, 1985.

[24] Lowe, D. G. The viewpoint consistency constraint. *International Journal of Computer Vision*, 1(1):57–72, 1987.

[25] Mackworth, A. K. and Mokhtarian, F. The renormalized curvature scale space and the evolution properties of planar curves. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 318–326, 1988.

[26] Marr, D. and Hildreth, E. Theory of edge detection. In *Proc. R. Soc. Lond.*, pages 187–217, 1980.

[27] Marr, David. *Vision.* Freeman, San Francisco, 1982.

[28] Martelli, A. An application of heuristic search methods to edge and contour detection. *Communications of the ACM*, 19(2):73–83, 1976.

[29] Montanari, U. On the optimal detection of curves in noisy pictures. *Communications of the ACM*, 14(5):335–345, 1971.

[30] Nalwa, V. S. and Binford, T. O. On detecting edges. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):699–714, 1986.

[31] Nevatia, R. and Babu, K. R. Linear feature extraction and description. *Computer Graphics and Image Processing*, 13:257–269, 1980.

[32] Nilsson, N. J. *Principles of Artificial Intelligence.* Morgan Kaufmann Publishers, Los Altos, CA, 1980.

[33] Nilsson, N. J. *Problem Solving Methods in Artificial Intelligence.* McGraw-Hill, New York, 1971.

[34] Pavlidis, T. *Algorithms for Graphics and Image Processing.* Computer Science Press, Rockville, MD, 1982.

[35] Peich, A. M. Comments on fingerprints of two-dimensional edge models. *Computer Vision, Graphics, and Image Processing*, 42:381–386, 1988.

[36] Rosenfeld, A. and Kak, A. *Digital Picture Processing.* Academic Press, San Diego, 1982.

[37] Rosenfeld, A. and Thurston, M. Edge and curve detection for visual scene. *IEEE Trans. on Computers*, C-20(5):562–569, 1971.

[38] Schunck, B. G. *Gaussian Filters and Edge Detection.* Technical Report GMR-5586, General Motors Research Laboratories, 1986.

[39] Shah, M., Sood, A., and Jain, R. Pulse and staircase edge models. *Computer Vision, Graphics, and Image Processing*, 34:321–341, June, 1986.

[40] Williams, D. and Shah, M. *Edge Contours.* Technical Report CS-TR-88-18, University of Central Florida, Computer Science Department, Sept., 1988.

[41] Williams, D. and Shah, M. Edge contours using multiple scales. *Computer Vision, Graphics, and Image Processing*, in press.

[42] Williams, D. and Shah, M. Multiple scale edge contours. In Trivedi, M. M., ed., *Applications of Artificial Intelligence VII*, pages 13–24, SPIE, 1989.

[43] Williams, D. and Shah, M. *On Normalized Edge Detection.* Technical Report CS-TR-89-04, University of Central Florida, Computer Science Department, March, 1989.

[44] Witkin, A. Scale-space filtering. In *Proc. Seventh International Joint Conf. on A.I.*, pages 1019–1021, 1983.