EDGES FROM IMAGE

by

Shih Jong Lee

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

Doctor of philosophy

in

Electrical Engineering

Approved:

R. M. Haralick, Chairman

L. G. Shapiro

P. M. Lapsa

K. B. Yu

J. W. Roach

June, 1985

Blacksburg, Virginia

EDGES FROM IMAGE

by

Shih Jong Lee

R. M. Haralick, Chairman

Electrical Engineering

(ABSTRACT)

To simulate the edge perception ability of human eyes and detect scene edges from an image, context information and world constraints must be employed in the edge detection process. To accomplish this, two Bayesian decision theoretic frameworks for context dependent edge detection are developed around the local facet edge detector. The first approach uses all the context in the neighborhood of a pixel. The second approach uses the context of the whole image. The mechanism of the context edge detector then assigns a pixel the most probable edge state which is consistent with its assumed edge context.

We also demonstrate how world constraints can aid the edge detection process with a lighting compensation and a curvature constraint scheme. The context information and world constraints can also be used to evaluate the performance of different edge detectors. A general edge coherence measure, a robust edge thinness measure, and a general edge correctness measure are developed.

Upon comparing the performance of the edge detectors with the context free second directional derivative zero-crossing edge operator, we find that the context dependent edge detector is superior; the world constrained context free edge detectors can also improve the edge result.

Finally, some simple edge detection schemes based on morphologic operations are discussed and evaluated. Although their performances are not as good as the other edge operators described in this dissertation, they are acceptable in the images which have reasonably high signal-to-noise ratio. These morphological edge operations can be realized most efficiently in machine vision systems that have special hardware designed for morphologic operations.

ACKNOWLEDGEMENTS

The author sincerely wishes to thank Dr. R. M. Haralick for his guidance, for his valuable suggestions, and for his financial support. The author also wishes to thank Dr. L. G. Shapiro for her encouragement and suggestions. Especially, the author would like to thank VPI & SU Spatial Data Analysis Laboratory and Machine Vision International Research Laboratory for providing excellent research environments.

The author also gives many thanks to his family for their continuous encouragement and moral support.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

I. INTRODUCTION

I-1. Introduction

Edges in a scene are the consequence of changes in some physical properties of surfaces of the scene, such as illumination (shadows, for example), geometry (orientation or depth) and reflectance. As there is a direct relationship between the edges and physical properties of a scene, much of the scene information can be recovered from an edge image. Thus, edge detection plays a key step in the early processing of a computer vision system. Edge detection converts a greyscale image into a binary form with direction orientation which preserves most of the useful image information. The rest of the vision processes can deal with the simpler form instead of dealing with greyscale image directly.

Image edges occur in places of significant intensity changes on the image. There are many kinds of intensity changes in an image. The most common type of edges are the edges which cause jumps in intensity value. They correspond to local extrema in intensity derivative. We call these edges step edges. The usual aim of edge detection is to locate edges belonging to boundaries of objects of interest. While the human eye performs this task easily, the detection of edges is a complex task to achieve. The difficulties in edge detection are mostly caused by noise, blurring and quantizing effects. This results in a situation in which not all image edges correspond to scene edges. And on the other hand, not all the scene edges correspond to image edges. When people perform edge detection on a given image, they can detect edges selectively. Some of the minor image edges which do not correspond to main scene edges will be ignored and some of the major scene edges will be detected, even though they do not correspond perfectly to image edges. In this way people can incorporate a lot of world knowledge and context information in their edge detection process.

A lot of different approaches for edge detection have been proposed in the past decade. Most of these operators can detect image edges. Although they perform reasonably well on simple noise free images, they tend to fail on the images which are degraded by noise. This is because that as the noise of an image increases, the correspondence between image edges and scene edges becomes weaker and weaker. Thus, an edge detector which can perform well on noisy images is most desirable.

Although it is possible to derive an edge detection algorithm and argue that it is mathmatically optimal under certain ideal image models, the edge detection problem for real scenes has not yet been completely solved. The reason is that there is still no acceptable complete model of the image intensity surface patterns correspond to scene edges. People only select some particular image intensity surface patterns (for example, step or ramp function) and look for an edge operator which can optimally detect these particular image intensity surface patterns. Thus, there is no guarantee that the operator derived from these incomplete patterns can optimally identify all kinds of image intensity changes corresponding to edges of real scenes. In other worlds, any operator which is optimal for image edges is not necessarily optimal for scene edges. However, the edges in the real scene domain are not difficult to define, and they are less ambiguous than the edges defined in the image domain. The ambiguity with image edges is mainly caused by the unknown factors involved in the many to one physics governing image acquisition.

The solution to edge detection on noisy images should not be image smoothing, because image smoothing alone tends to blur edges. The best solution, we believe, is to incorporate world knowledge and edge context information with the edge detection process. Just as in written English one can expect to find certain frequent letter combinations, such as ity," "est", "ion", "ent", so certain patterns of edge distributions in a neighborhood of image positions are likely to occur in the 'context' of others. The former phenomenon has been used to improve character recognition accuracy in text reading machines. We shall demonstrate that the latter can be used to improve accuracy in detecting edges of an image.

In this dissertation, we suggest edge detection schemes which incorporate world knowledge and context information to detect scene edges instead of image edges. In chapter II, we suggest an edge detection scheme based on a local facet image model and a Bayesian decision theoretic framework which uses context information to aid in the detection of edges. Although we incorporate the context approach with only the facet edge detector, the context approach can be extended to be incorporated with any kind of edge detector and should increases the performance of that edge detector.

The local facet model regards the discrete pixel intensity function within a local neighborhood as a sampling of a continuous underlying function. We assume some parametric form for the underlying function and estimate the parameters for the functions around each local neighborhood centered on each pixel position and detect edges based on the estimated parameters.

We formulate the simplest edge detection problem as a decision making problem and prove that under certain approximations the edge detection process can be done independently by local neighborhood. This simplest edge detection scheme does not always provide a good result since it is just a local operator. In view of this fact, we incorporate neighborhood edge context into the edge detecting process so that we can detect connected edge segments instead of edge points alone.

In the first part of chapter II, we introduce neighborhood context information into the edge detection process by assuming that only certain spatial configurations of edge states and edge directions in any neighborhood are likely to occur and assign such patterns equal prior probability. We then assign a pixel the most probable edge state which is consistent with the allowable neighborhood patterns. We set up some of the possible edge patterns in a 3 X 3 neighborhood and for each application we select the most appropriate patterns to work with. The purpose of this part is to show how well the mechanism we introduce can use context information effectively. And it also provides an appropriate way of edge detection on the images for which neighborhood edge context is known.

In the second part of chapter II, we introduce a general scheme which generates edge patterns for each neighborhood by a dynamic programming technique. No neighborhood edge context patterns need to be provided in advance. The scheme can take care of any size neighborhood and it selects the edge context pattern automatically.

The general context approach described here is related to the dynamic programming idea of Montanari (1971) and Martelli (1976) of linking together edge segments. However, we are able to handle multiple boundaries simultaneously and naturally, whereas dynamic programming, as they employed it, can only link one boundary at a time and is basically a postprocessing process whose performance heavily depends on the starting points for linking which are provided for the preprocessing.

In chapter III, we formulate a scheme which for each pixel makes use of the full context of the whole image to detect edges. Thus, the edge detection is no longer a local process. It uses the whole image to assign an edge state of a pixel. We also introduce a clever way of achieving local and global context balance. We introduce a way of making use of strong context information and paying less attention to the weak context constraints, so that we can avoid the possibility of incorrectly guided by the weak and ambiguous context.

In chapter IV, we introduce the method of using world constraints to help the edge detection process. Two schemes are described, the first constraint is based on non-uniform lighting compensation; the second constraint is based on the fact that most of the main scene edges are located at the boundary of two large enough scene surfaces.

Chapter V introduces a general edge evaluator which makes use of local edge context and world constraint to evaluate the performance of different edge detectors. It can evaluate edge detectors based on the local edge coherence of the edge image and besides, it can also evaluate edge position correctness of the given image and its edge detection result. This is the first time a general edge position correctness evaluation scheme has been proposed. In chapter VI, some morphologic edge detection schemes are proposed which can be realized most efficiently in the machine vision systems which have special hardware designed for morphologic operations (Sternberg,1983).

I-2. Related literature review

One conventional approach to the problem of edge detection uses parallel enhancement/thresholding algorithms. These algorithms use spatial operators to enhance the original image to form an edge enhancement strength map. A threshold is then applied to the edge strength map to determine the presence or absence of edge pixels.

Spatial operators can be differential operators or template matching operators(see Rosenfeld et. al,1982). Differential operators implement a discrete gradient like function. The strength of the enhancement map at pixel (r,c) is the magnitude of the gradient at (r,c). The direction (or angle) of the gradient is also obtained at any pixel (r,c). This type of operation is typified by Sobel edge detection. (Duda et.al.,1973)

Template matching operators produce a discrete differentiation in a set of specific directions using a set of templates. The strength of the enhancement map at (r,c) is the maximum output value from these templates. The direction value is that direction associated with the mask which has the largest response. These types of operators include the Prewitt operator, the Kirsch operator and the Robinson 3-level and 5-level operators. Both approaches are extremely widely used, and are quite old in concept (Prewitt, 1970; Kirsch, 1971; and Robinson, 1978). However, the existing masks for these approaches appear rather ad hoc, and their theory is still being developed (Foglein, 1983).

There are a number of edge detectors that involve fitting a function to the image surface. They make the decision as to the presence of an edge using an estimate of its location from the best-fitting surface that approximates the real image surface. These methods usually involve an initial parametric fit of the image surface in terms of some set of basis functions.

One of the earliest examples of this method was the Prewitt operator (Prewitt, 1970), which used a quadratic set of basis functions. Another early work is the Hueckel edge detector (Hueckel, 1971,1973). It uses a basis function with circular support, and tries to fit a single step edge to each circular area. The basis functions are chosen to approximate the Fourier transform of the circular region. Brooks (1977) understood the surface fitting requirement in explaining some of the classical edge operators. Hummel (1979) suggested the use of Karhunen-Loeve principal components for the basis functions required by the edge detector. Morgenthaler's (1980,1981) edge detector used a hybrid local model consisting of step edge superimposed on a low-order polynominal function.

Haralick (1980) proposed a fitting of the image by small planar surfaces or 'facets'. Edges are marked at pixels which belong to two such facets when the parameters of the two surfaces are inconsistent. The test for consistency is based on the goodness of fit of each surface within its neighborhood by a χ^2 statistic. In subsequent work on edge detection, Haralick, (1982,1984)) does a least squares fit to a bivariate cubic polynominal in the pixel's central neighborhood and determines directional derivatives from the fitting coefficients. Instead of computing an isotropic form of the second derivative, Haralick computes the second directional derivative in the direction of the gradient. If a pixel contains a high enough negatively sloped zero crossing of the second directional derivative taken in the direction of the gradient, then the pixel is called an edge pixel.

Recently, some edge detectors attempt to enhance edges by filtering. The filters are designed using frequency domain techniques to optimally discriminate step edges from the background. Modestino and Fries (1977) suggested a procedure for detecting edges in noisy pictures using a Winer filtering approach. Their algorithm is derived from a minimum mean squared error estimate of noisy edges, with an a priori assumption that a modified Laplacian operator is the best edge detecting filter. Shanmugan et al (1979) proposed the use of a two-dimensional linear operator that approximates the Laplacian of a Gaussian. Their criteria of optimality was that the band-limited filter maximize the proportion of total output energy confined to a fixed interval when it is convolved with a step edge. Marr and Hildreth (1980) determined edges by first smoothing the image with a Gaussian filter and then taking the Laplacian of the resulting image.

A context dependent edge detection using relaxation labeling scheme was proposed by Zucker et. al. (1977). They associated each pixel's relative edge strength in each direction to a probability by a suitable normalization of the edge strength. The possibility of no edge is also included. Then in a relaxation procedure they permit the probabilities of the directional edges for neighboring pixels to support or weaken each other according to the consistency between them. The process comes to near convergence after a few iterations and results in edge pixels having a high probability in their edge direction and non-edge pixels having a high probability for the no edge label. However, the Zucker scheme does not have a true probability interpolation.

II. CONTEXT DEPENDENT EDGE DETECTION

In this chapter, we formulate the edge detection problem based on a Bayesian model and give a general solution to it. Section II-1 gives the notation conventions of this chapter. Section II-2 introduces the context-free second derivative zero crossing operator. In section II-3 we introduce the context into the edge detection process and discuss how a table look-up scheme can be used to realize context dependent edge detection. Besides, we show how to obtain the joint probability terms from an equal probability assumption for legal edge neighborhood configurations. In section II-4, a general context edge operator based on a dynamic programming technique is introduced. Section II-5 provides some experimental results with the context edge operators we compare and show the effectiveness of the context edge detector over the context-free second directional derivative zero crossing edge operator.

II-1. Notation

A. Notation Conventions

1. Zr: designates row size of an image.

2. Zc: designates column size of an image.

3. R: designates the row index set of an image.

$$R = \{1, \ldots, Zr\}$$

4. C: designates the column index set of an image.

$$C = \{1, \ldots, Zc\}$$

5. (r,c): designates a 2-dimensional image position.

$$(r, c) \in R \times C.$$

6. N(r,c): the set of 3 x 3 neighbors of (r,c), including (r,c).

7. n: designates one image position in a 3 x 3 neighborhood: the central pixel is designated by n=0 and the other positions are designated n=1 to n=8 in a fixed order.

8. ε_i: designates the edge state at position i: it can be either 'edge' or 'no-edge'.

9. θ_i: designates the edge direction (direction of extremum in first directional derivative) at position i.

10. E: designates the two dimensional sequence of edge states of all the image positions.

$$E = (\varepsilon(r, c) | (r, c) \in R \times C)$$

11. θ: designates the two dimensional sequence of edge directions of all the image positions.

$$\theta = (\theta(r,c)|(r,c) \in R \times C)$$

12. ε_N: designates the 9-tuple of edge states of a 3 x 3 neighborhood of image positions.

$$\varepsilon_N = (\varepsilon_n | n \in \{0,\ldots,8\})$$

13. θ_N: designates the 9-tuple of edge directions of a 3 x 3 neighborhood of image positions.

$$\theta_N = (\theta_n | n \in \{0,\ldots,8\})$$

14. E edge: designates all the allowable 3 x 3 neighboring edge patterns under the condition that the central pixel of the pattern is edge.

15. E no-edge: designates all the allowable 3 x 3 neighboring edge patterns under the condition that the central pixel of the pattern is non-edge.

16. P(edge): designates the a priori probability $P(\varepsilon = edge)$ and assumes it is independent of image positions.

17. K: designates the two dimensional sequence of ten-dimensional vectors of the cubic facet coefficients for all image positions.

$$K = (\underline{k}(r,c) | (r,c) \in R \times C)$$

18. $\underline{k}_n$: designates the ten-dimensional cubic facet coefficients at image position n within a neighborhood.

$$\underline{k}_n = (k_{1n}, k_{2n}, \ldots, k_{10n})$$

B. **Superscript Conventions** When a parameter has a superscript star, it designates true value. Parameters without superscripts are either assigned values or estimated values.

1. $\varepsilon^*, \theta^*, k^*$: true values
2. ε, θ: assigned values
3. k: estimated value

II-2. The context-free second derivative zero crossing edge operator

In this section, we introduce how the facet model and directional derivative can be used to obtain the edge state and edge direction of a pixel and formulate a context free edge detector model based on the Bayesian decision theoretic framework.

A. The Facet Model

The facet model states that any analysis made on the basis of pixel values in some neighborhood has its final authoritative interpretation relative to the underlying gray-tone intensity surface of which the neighborhood pixel values are observed noisy samples. To estimate the continuous intensity surface, we use a functional form consisting of a cubic polynomial in the two variables row and column. For greater numerical accuracy this can be calculated in terms of linear combinations of the tensor products of discrete orthogonal polynomials of up to degree three (Haralick, 1981). This form is often used in statistical regression problems (Draper and Smith, 1981). In this paper, the fitting cubic polynomial f in each neighborhood is represented as

$$f(r,c) = k_1 + k_2 r + k_3 c + k_4 r^2 + k_5 rc + k_6 c^2 + k_7 r^3$$

$$+k_8 r^2 c + k_9 rc^2 + k_{10} c^3. \qquad (II-1)$$

where $k_1, k_2, \ldots, k_{10}$ are the estimated coefficients of the facet surface fit f to one neighborhood on the image.

B. The Directional Derivative

In two dimensions, the rate of change of a function f depends on direction. In this section we discuss the relationship between the directional derivatives and the coefficients from the polynomial fit. (Haralick et al., 1983).

The directional derivative of f at the point (r,c) in the direction α is denoted by $f'_\alpha(r,c)$, it is defined by

$$f'_\alpha(r,c) = \frac{\partial f(r,c)}{\partial r}\sin\alpha + \frac{\partial f(r,c)}{\partial c}\cos\alpha \qquad (II-2)$$

The direction angle α in our convention is the clockwise angle from the column axis. We denote the second directional derivative of f at the point (r,c) in the direction α by $f''_\alpha(r,c)$ and it follows that

$$f''_\alpha(r,c) = \frac{\partial^2 f(r,c)}{\partial r^2}\sin^2\alpha + 2\frac{\partial^2 f(r,c)}{\partial r \partial c}\sin\alpha\cos\alpha$$

$$+\frac{\partial^2 f(r,c)}{\partial c^2}\cos^2\alpha \qquad (II-3)$$

Taking f to be the cubic polynomial model represented as (II-1), we can easily obtain $f'_\alpha(r,c)$ at the central position of a neighborhood as

$$f'_\alpha(0,0) = k_2\sin\alpha + k_3\cos\alpha \qquad (II-4)$$

and when $k_2^2 + k_3^2 > 0$ the angle α can be obtained by

$$\sin\alpha = \frac{k_2}{\sqrt{k_2^2 + k_3^2}}$$

$$\cos\alpha = \frac{k_3}{\sqrt{k_2^2 + k_3^2}} \qquad (II-5)$$

At any point (r,c), the second directional derivative in the direction α is given by

$$f''_\alpha(r,c) = (6k_7 \sin^2\alpha + 4k_8 \sin\alpha\cos\alpha + 2k_9\cos^2\alpha)r$$

$$+(6k_{10}\cos^2\alpha + 4k_9\sin\alpha\cos\alpha + 2k_8\sin^2\alpha)c$$

$$+(2k_4\sin^2\alpha + 2k_5\sin\alpha\cos\alpha + 2k_6\cos^2\alpha) \qquad (II-6)$$

We wish to only consider points (r,c) on the line in direction α. Hence, let $r = \rho\sin\alpha$ and $c = \rho\cos\alpha$. Then we have

$$f''_\alpha(\rho) = 6[k_7\sin^3\alpha + k_8\sin^2\alpha\cos\alpha + k_9\sin\alpha\cos^2\alpha + k_{10}\cos^3\alpha]\rho$$

$$+2[k_4\sin^2\alpha + k_5\sin\alpha\cos\alpha + k_6\cos^2\alpha] \qquad (II-7)$$

where ρ is the distance along the direction α between (r,c) and (0,0).

C. The Context-free Digital Edge

We know that edge refers to places in the image where there appears to be a jump in brightness value or a local extremum in brightness value derivative. Since the image can be considered as a discrete array of brightness values and there exists no definition of derivative for a discrete array of brightness values, the way to interpret jumps in value and jumps in derivative is to fit a continuous function f, e.g. the bivariate cubic represented as (II-1), to the discrete array of values in each neighborhood. Sharp discontinuities will reveal themselves in high values for estimates of the first partial derivatives.

With regard to context-free edge detection, we define an edge to occur in a pixel if and only if there are some points in the pixel's area having a negatively sloped zero crossing of the second directional derivative taken in the direction of a non-zero gradient at the pixel's center. This places edges at spatial peaks of gradient values (Haralick,1984), a concept which is the basis of the non-maximal edge surpression algorithms (Rosenfeld,1982) which are sometimes used after edge detection to thin the detected edges.

D. The Edge Detecting Problem and a Context-free Solution

According to the Bayesian framework for decision making (Haralick, 1983), the edge detecting problem can be formulated as one of determining edge image E which maximizes the expected gain

$$\sum_{E^*} G(E, E^*)P(E^*|K, Q) \qquad (II-8)$$

where $G(E, E^*)$ is the gain realized when edge image E is assigned and the true edge image is E^* : $P(E^*|K, Q)$ is the conditional probability that E^* is the true edge image given the cubic facet data K computed from all the observed gray-tone intensity image data and prior information Q about legal edge configurations.

The gain function we use here computes the gain realized when the assigned image is E and the true image is E^* as the sum of the gains realized on a pixel by pixel basis. That is,

$$G(E, E^*) = \sum_{r \in R} \sum_{c \in C} G(\varepsilon_{rc}, \varepsilon^*_{rc}) \qquad (II-9)$$

In the case that the gain function is defined by (II-9), after direct substitution and some mathematical manipulations (II-8) becomes

$$\sum_{r \in R} \sum_{c \in C} \sum_{\varepsilon^*_{rc}} G(\varepsilon_{rc}, \varepsilon^*_{rc})P(\varepsilon^*_{rc}|K, Q) \qquad (II-10)$$

Thus for each (r, c), the edge detecting problem becomes one of finding that value of ε_{rc} which maximizes

$$\sum_{\varepsilon^*_{rc}} G(\varepsilon_{rc}, \varepsilon^*_{rc})P(\varepsilon^*_{rc}|K, Q) \qquad (II-11)$$

We use the zero-one gain function for each pixel. When the assigned edge value and true edge value for a pixel are identical, then the gain is one. If they are different the gain is zero. This gain function is quite simple and yet suitable for edge detection. With the zero-one gain function, the maximization of (II-11) becomes: for each pixel (r, c) independently assign the edge state 'edge' if

$$P(\varepsilon^*_{rc} =' edge' | K, Q) > P(\varepsilon^*_{rc} =' no - edge' | K, Q) \qquad (II - 12)$$

and assign the edge state 'no-edge' otherwise.

There are a couple of conditional independence assumptions which can make the above probabilities only depend on the local neighborhood of (r, c). These assumptions must be considered approximations rather than idealizations and they motivate the use of the neighborhood which under some conditions can be as good as the entire image. The first conditional independence assumption amounts to limited influence. It says that the true facet parameters at a pixel (r, c) only influence the estimated facet data values for pixels in the neighborhood of (r, c) and do not influence the estimated facet parameters for pixels outside the neighborhood of (r, c). Hence

$$P(\underline{k}_{ij} : (i, j) \notin N(r, c) | \underline{k}^*_{rc}) = P(\underline{k}_{ij} : (i, j) \notin N(r, c)) \qquad (II - 13)$$

The second conditional independence assumption amounts to noise independence. It states that given the true facet parameters $\underline{k}^*_{rc}$

at pixel (r,c), the probability of jointly observing facet parameters for the neighborhood of $(r,c), \{\underline{k}_{ij} : (i,j) \in N(r,c)\}$, and for the mutually exclusive image region outside of $N(r,c), \{\underline{k}_{ij} : (i,j) \notin N(r,c)\}$, is equal to the conditional probability of $\{\underline{k}_{ij} : (i,j) \in N(r,c)\}$ given $\underline{k}^*_{rc}$ times the conditional probability of $\{\underline{k}_{ij} : (i,j) \notin N(r,c)\}$ given $\underline{k}^*_{rc}$, that is

$$P(\underline{k}_{ij} : (i,j) \in N(r,c),\ \underline{k}_{lm} : (l,m) \notin N(r,c)|\underline{k}^*_{rc}) =$$

$$P(\underline{k}_{ij} : (i,j) \in N(r,c)|\underline{k}^*_{rc})P(\underline{k}_{lm} : (l,m) \notin N(r,c)|\underline{k}^*_{rc}) \qquad (II-14)$$

By *Appendix 1*, the two conditional independence assumptions imply

$$P(\varepsilon^*_{rc}|K) = P(\varepsilon^*_{rc}|\underline{k}_{ij} : (i,j) \in N(r,c)) \qquad (II-15)$$

In the context free decision making process, the neighborhood $N(r,c)$ just contains the single pixel (r,c) and the edge detecting process becomes one in which a pixel is assigned an edge state 'edge' if

$$P(\varepsilon^*_{rc} =' edge'|\underline{k}_{rc}) > P(\varepsilon^*_{rc} =' no-edge'|\underline{k}_{rc}) \qquad (II-16)$$

A detail discussion about the context free local facet second derivative zero-crossing edge operator is given by Haralick (1984) and the way the conditional probability $P(\varepsilon^*_{rc}|\underline{k}_{rc})$ can be computed is discussed in *Appendix2.*

Let

$$f''_{\alpha}(r,c) = A\rho + B$$

where

$$A = 6[k_7S^3 + k_8S^2C + k_9SC^2 + k_{10}C^3]$$

$$B = 2[k_4S^2 + k_5SC + k_6C^2]$$

and $S = \sin\theta^*$; $C = \cos\theta^*$. Then from *appendix 2* we have

$$P(\varepsilon^*,\theta^*|k_2,\ldots,k_{10}) = \frac{P(k_2^*,k_3^*|\varepsilon^*,\theta^*)P(A,B|\varepsilon^*,\theta^*)}{2\pi P(k_2,\ldots,k_{10})}$$

$$*P(k_4,\ldots,k_{10}|A,B)P(\varepsilon^*) \qquad (II-17)$$

with the conditional distribution for k_2, k_3 given $\varepsilon^* =' edge', \theta^*$ being a normal distribution

$$N\left(H\begin{pmatrix}0\\ \sqrt{\frac{\pi}{2}}\lambda\sigma_{g1}^*\end{pmatrix},\ H\begin{pmatrix}\sigma^2 q & 0\\ 0 & 2\lambda^2\sigma_{g1}^{*2}+\sigma_q^2\end{pmatrix}H'\right) \qquad (II-18)$$

and the conditional distribution of k_2, k_3 given $\varepsilon^* =' no-edge', \theta^*$ being a a normal distribution

$$N\left(H\begin{pmatrix}0\\0\end{pmatrix}, H\begin{pmatrix}\sigma^2 q & 0\\ 0 & 2\sigma_{g1}^{*2}+\sigma_q^2\end{pmatrix}H'\right) \qquad (II-19)$$

where

$$H = \begin{pmatrix}\cos\theta^* & \sin\theta^*\\ -\sin\theta^* & \cos\theta^*\end{pmatrix}$$

The conditional probability $P(A, B|\varepsilon^* =' edge', \theta^*)$ is

$$N\left(\begin{pmatrix}\mu_A^*\\ 0\end{pmatrix},\ T\begin{pmatrix}0 & 0\\ 0 & \sigma_s^{2*}\end{pmatrix}T' + \begin{pmatrix}\sigma_{\eta A}^2 & 0\\ 0 & \sigma_{\eta B}^2\end{pmatrix}\right) \qquad (II-20)$$

and the conditional probability $P(A, B|\varepsilon^* =' no - edge', \theta^*)$ is

$$N\left(\begin{pmatrix}0\\ 0\end{pmatrix},\ T\begin{pmatrix}\sigma_r^{2*} & 0\\ 0 & \sigma_s^{2*}\end{pmatrix}T' + \begin{pmatrix}\sigma_{\eta A}^2 & 0\\ 0 & \sigma_{\eta B}^2\end{pmatrix}\right) \qquad (II-21)$$

where all the variables $T, A, B, \sigma^2, \ldots$, are defined in *Appendix2.* In the context free case, we assign an edge if

$$\int_{\theta^*} P(\varepsilon^* =' edge', \theta^*|k_2, \ldots, k_{10})d\theta^*$$

$$> \int_{\theta^*} P(\varepsilon^* =' no - edge'\theta^*|k_2, \ldots, k_{10})d\theta^* \qquad (II-22)$$

Equation (II-22) takes into account both the requirement of a large enough estimated gradient and the negatively sloped zero crossing of second directional derivative taken in the direction of the gradient. It takes into account the distribution differences of (II-18), (II-19), (II-20) and (II-21). A gradient effect only edge detector would assign an edge if

$$P(\varepsilon^* =' edge'|k_2, k_3) > P(\varepsilon^* =' no - edge'|k_2, k_3).$$

Such an edge detector only uses the distribution differences of (II-18) and (II-19) as the basis for its decision.

II-3. A context dependent edge detection

In this section, we derive the algorithm for context dependent edge detecting process based on the Bayesian model described in section II-2. We illustrate how to implement the approach using an equal probability dictionary based scheme.

A. The context dependent solution

In the neighborhood context decision making process, all the observed data in the neighborhood is used to help assign edge values for the center pixel in the neighborhood. By combining (II-12) and (II-15), the edge detecting solution is now: for each pixel (r, c) assign it edge state 'edge' if

$$P(\varepsilon^*_{rc} =' edge' | \underline{k}_{ij} : (i, j) \in N(r, c))$$

$$> P(\varepsilon^*_{rc} =' no - edge' | \underline{k}_{ij} : (i, j) \in N(r.c)) \qquad (II - 23)$$

and otherwise assign it edge state 'no-edge'.

In order to simplify notation, we make everything relative to each neighborhood. Let

$$P(\varepsilon^*_0 | K_N) = P(\varepsilon^*_{rc} | \underline{k}_{ij} : (i, j) \in N(r, c)) \qquad (II - 24)$$

where for example in a 3 x 3 neighborhood $k_N = (\underline{k}_n | n \in \{0, \ldots, 8\})$ and the indexes number the pixel positions

$$\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 0 & 5 \\ 6 & 7 & 8 \end{array}$$

The technique we discussed, however, is not limited to a 3 X 3 neighborhood. Now the solution to the edge detecting problem becomes for each center pixel position assign ε_0 as 'edge' if

$$P(\varepsilon_0^* =' edge' | K_N) > P(\varepsilon_0^* =' no - edge' | K_N) \qquad (II-25)$$

Now by the definition of conditional probability,

$$P(\varepsilon_0^* | K_N) = \sum_{\theta_N^*} \sum_{\varepsilon_1^*, \ldots, \varepsilon_8^*} P(K_N | \varepsilon_N^*, \theta_N^*) \frac{P(\varepsilon_N^*, \theta_N^*)}{P(K_N)} \qquad (II-26)$$

We make two conditional independence assumptions on the measurement process. The first assumption states that the description process is local. When the pixel n in the neighborhood is being examined, no observed characteristics from any other pixel but pixel n affect the observed data of position n. Hence

$$P(K_N | \varepsilon_N^*, \theta_N^*) = \prod_{n=0}^{8} P(\underline{k}_n | \varepsilon_N^*, \theta_N^*) \qquad (II-27)$$

The second assumption states that the observed facet measurements $\underline{k}_n$ of pixel n depends only upon the true facet parameters associated with pixel n and does not depend upon any true edge data of the other pixels. Hence

$$P(\underline{k}_n | \underline{k}_n^*, \varepsilon_N^*, \theta_N^*) = P(\underline{k}_n | \underline{k}_n^*, \varepsilon_n^*, \theta_n^*) \; n \in \{0, \ldots, 8\} \qquad (II-28)$$

Under these assumptions, we have (see *Appendix 3*)

$$P(K_N|\varepsilon_N^*, \theta_N^*) = \prod_{n=0}^{8} P(\underline{k}_n|\varepsilon_n^*, \theta_n^*) \qquad (II-29)$$

Thus (II-26) becomes

$$P(\varepsilon_0^*|K_N) = \sum_{\theta_0^*} P(\underline{k}_0|\varepsilon_0^*, \theta_0^*) \sum_{\varepsilon_1^*,\theta_1^*,\ldots,\varepsilon_8^*,\theta_8^*} \prod_{n=1}^{8} P(\underline{k}_n|\varepsilon_n^*, \theta_n^*)$$

$$* \frac{P(\varepsilon_N^*, \theta_N^*)}{P(K_N)} \qquad (II-30)$$

The conditional probability term $P(\underline{k}_n|\varepsilon_n^*, \theta_n^*)$ ties the local facet measurement with the true edge state and direction. The joint probability term $P(\varepsilon_N^*, \theta_N^*)$ takes context into account it represents the prior knowledge contextual constraints of true edge state and orientations for a neighborhood.

The context we use in what follows, is a 3 x 3 sized neighborhood. The mathematics for a larger sized neighborhood is similar.

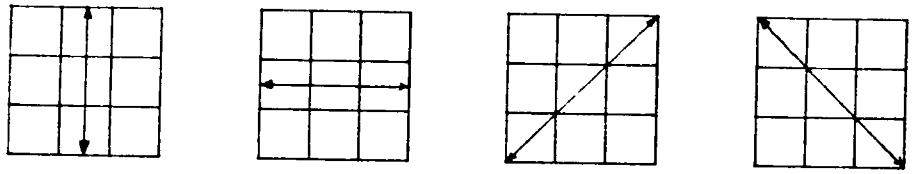
B. The equal probability of legal edge patterns assumption

In each 3 X 3 neighborhood only certain patterns of edge state and edge gradient direction combinations are likely to occur. Thus, all the possible edge patterns in a neighborhood can be divided into an allowable set and a non-allowable set. The allowable context edge patterns are further divided into two sets, one is the set in which the center pixel has edge state 'edge'. We denote this set as *Eedge*. The other one is the set in which the center pixel has edge state 'no-edge'. We denote this set as $Eno - edge$. The equal probability assumption states that all the allowable patterns in the same set have an equal non-zero probability. Those which are non-allowable have zero probability. Thus, each allowable pattern in *Eedge* has probability $\frac{P(edge)}{\#Eedge}$. Similarly, each allowable pattern in $Eno - edge$ has probability $\frac{1-P(edge)}{\#Eno-edge}$. Where $\#Eedge$ designates the total number of allowable patterns in the set *Eedge* and $\#Eno - edge$ designates the total number of allowable patterns in the set $Eno - edge$. This equal probability assumption on allowable context patterns is the equal probability of ignorance assumptions that Haralick (1983) uses with context and corresponds to the way Jayner ( ) suggests for selecting priors.

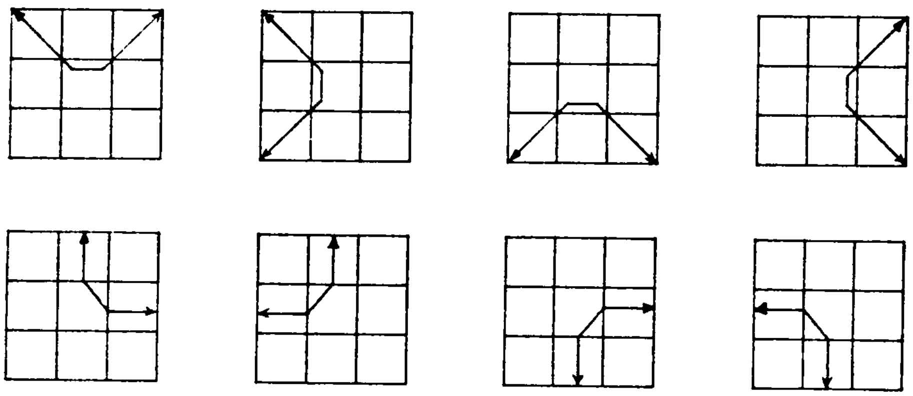
An example of the possible allowable edge patterns in a 3 x 3 neighborhood are shown in *Figure II-1*. Each pattern is labeled either *Edge pattern (set, number)* or *Non-edge pattern (set, number)* to specify the type of class, the set and the number it belongs to. The collection of allowable context edge patterns must be selected from the groups shown in *Figure II-1* according to the prior knowledge about the contextual characteristics of the image being processed.

Thus, the context edge patterns are dependent on what are the prior expectations of what configurations are likely to be focused in region of the pixel. Depending on the kinds of higher level vision control processer, these configurations can be the same for all the image or be different region by region.

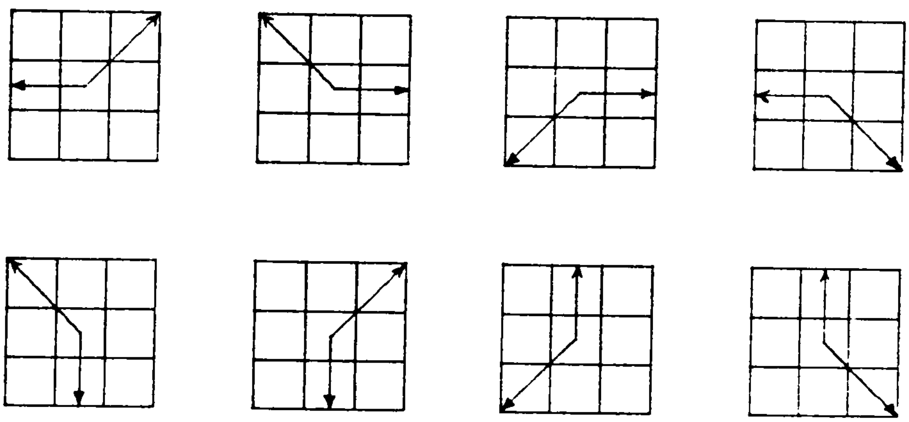
Edge pattern (1,1) - (1,4)



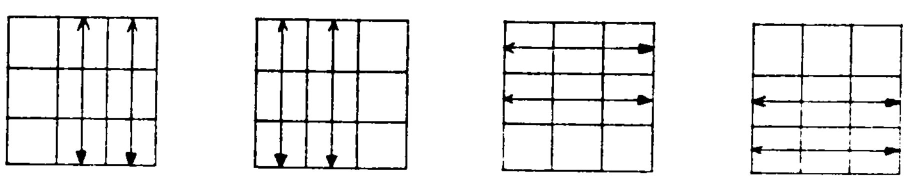
Edge pattern (2,1) - (2,8)



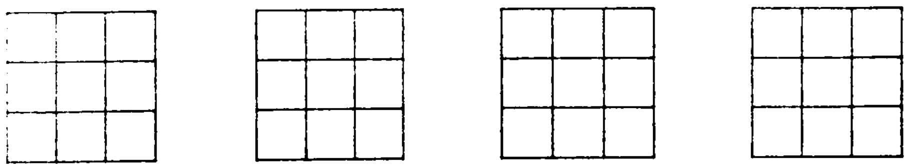
Edge pattern (3,1) - (3,8)



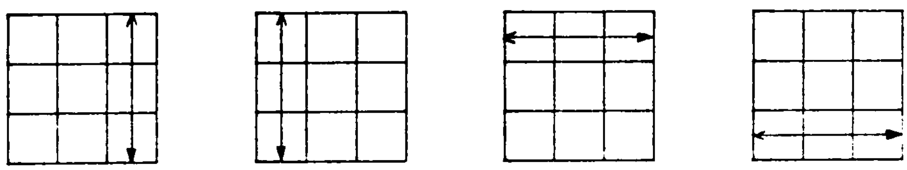
Edge pattern (4,1) - (4,4)



No-edge pattern (1,1)



No-edge pattern (2,1) - (2,4)



No-edge pattern (3,1) - (3,4)

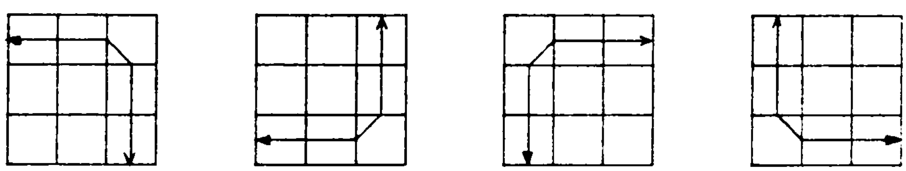


Figure II-1. Examples of some allowable edge and non-edge context patterns.

From equation (II-30) we know that the prior probability used in the edge detection process is the joint probability $P(\varepsilon_N^*, \theta_N^*)$. We now illustrate how to obtain this term from the context edge patterns by showing two examples: one for *Edge pattern (2,1)* which belongs to *Eedge*; one for *Non-edge pattern (3,2)* which belongs to $Eno - edge$.

Since the joint probability $P(\varepsilon_N^*, \theta_N^*)$ of an *Edge pattern (2,1)* can be written explicitly as

$$P(\varepsilon_N^*, \theta_N^*) = P(\varepsilon_{0,1,3}^* =' edge', \varepsilon_{2,4,5,6,7,8}^* =' no - edge',$$

$$\theta_0^* = \genfrac{}{}{0pt}{}{90^o}{270^o}, \theta_1^* = \genfrac{}{}{0pt}{}{45^o}{225^o}, \theta_3^* = \genfrac{}{}{0pt}{}{135^o}{315^o}, \theta_{2,4,5,6,7,8}^*) \qquad (II-31)$$

Where $\theta_{2,4,5,6,7,8}^*$ means that $\theta_2^*, \theta_4^*, \theta_5^*, \theta_6^*, \theta_7^*, \theta_8^*$ can vary freely in angle range from 0^o to 360^o and they are not necessarily equal. Since

$$P(\theta_{2,4,5,6,7,8}^*, \varepsilon_{0,1,3}^* =' edge',$$

$$\varepsilon_{2,4,5,6,7,8}^* =' no - edge', \theta_0^* = \genfrac{}{}{0pt}{}{90^o}{270^o}, \theta_1^* = \genfrac{}{}{0pt}{}{45^o}{225^o}, \theta_3^* = \genfrac{}{}{0pt}{}{135^o}{315^o})$$

$$= P(\theta_{2,4,5,6,7,8}^* | \varepsilon_{2,4,5,6,7,8}^* =' no - edge')$$

$$*P(\varepsilon_{2,4,5,6,7,8}^* =' no - edge', \varepsilon_{1,3}^* =' edge', \theta_0^* = \genfrac{}{}{0pt}{}{90^o}{270^o},$$

$$\theta_1^* = \genfrac{}{}{0pt}{}{45^o}{225^o}, \theta_3^* = \genfrac{}{}{0pt}{}{135^o}{315^o} | \varepsilon_0^* =' edge') P(\varepsilon_0^* =' edge')$$

We assume that given a group of non-edge pixels the conditional probability of their gradient directions are independent. And in general the conditional probability density of edge direction is uniform for non-edge pixels. Thus we give all the possible θ_i^* equal conditional probability. The prior probability for edge state = 'edge' is a constant P(edge). Thus, we can simplify the joint probability to

$$P(\varepsilon_N^*, \theta_N^*) = \left(\frac{1}{2\pi}\right)^6 * \frac{1}{\#Eedge} * P(edge)$$

$$= \frac{1}{(2\pi)^6} * \frac{P(edge)}{\#Eedge} \qquad (II-32)$$

Similarly, the joint probability $P(\varepsilon_N^*, \theta_N^*)$ of a *Non-edge pattern (3,2)* can be written explicitly as

$$P(\varepsilon_N^*, \theta_N^*) = P(\varepsilon_{3,5,6,7,8}^* =' edge', \varepsilon_{0,1,2,4}^* =' no-edge',$$

$$\theta_{3,5}^* = \frac{180^o}{0^o}, \theta_8^* = \frac{135^o}{315^o}, \theta_{6,7}^* = \frac{90^o}{270^o}, \theta_{0,1,2,4}^*)$$

$$= P(\theta_{0,1,2,4}^* | \varepsilon_{0,1,2,4}^* =' no-edge')$$

$$P(\varepsilon_{1,2,4}^* =' no-edge', \varepsilon_{3,5,6,7,8}^* =' edge',$$

$$\theta_{3,5}^* = \frac{180^o}{0^o}, \theta_8^* = \frac{135^o}{315^o}, \theta_{6,7}^* = \frac{90^o}{270^o} | \varepsilon_0^* =' no-edge')$$

$$P(\varepsilon_0^* =' no-edge')$$

$$= \prod_{i \in \{0,1,2,4\}} P(\theta_i^* | \varepsilon_i^* =' no-edge') \frac{1}{\#Eno-edge}(1-P(edge))$$

$$= \frac{1}{(2\pi)^4} * \frac{[1-P(edge)]}{\#Eno-edge} \qquad (II-33)$$

Following the same procedures we can obtain the joint probabilities $P(\varepsilon_N^*, \theta_N^*)$ for all the edge patterns illustrated in Figure II-1. In summary, the joint probability for an edge pattern can be assigned according to the following rule:

$$P(\varepsilon_N^*, \theta_N^*) = \frac{1}{(2\pi)^{\#non-edge\ pixels}} * \frac{P(pattern)}{\#Epattern}$$

where pattern can be either $Eedge$ or $Eno-edge$.

In the context dependent case, we assign an edge if

$$\sum_{(\theta_N^*, \varepsilon_N^*) \in Eedge} \prod_{n=0}^{8} P(\underline{k}_n | \varepsilon_n^*, \theta_n^*) P(\varepsilon_N^*, \theta_N^*)$$

$$> \sum_{(\theta_N^*, \varepsilon_N^*) \in Eno-edge} \prod_{n=0}^{8} P(\underline{k}_n | \varepsilon_n^*, \theta_n^*) P(\varepsilon_N^*, \theta_N^*) \qquad (II-34)$$

II-4. A General Context Dependent Edge Detection

The context dependent edge detection scheme we discussed in section II-3 is based on a dictionary of the allowable edge patterns in a 3 X 3 neighborhood. In order to further improve the performance of the context operator, it is important to have the flexibility of increasing the neighborhood size of the context edge patterns. However, increasing the size of context neighborhood will greatly increase the number of possible edge patterns. The huge number of edge patterns prohibit the dictionary-based scheme to be carried out within a reasonable computation and storage expense. To reduce the unmanagable amount of computation cost and avoid the need of prior information about context edge patterns, a dynamic programming scheme which generates the most feasible edge patterns in a given neighborhood and use only these patterns for edge detection is developed as a solution. The idea is to consider only the most feasible edge patterns of each neighborhood in the edge detecting process. This is a more general and computational effecient edge detecting process.

As we mentioned before, the dictionary-based context edge detection process is based on the consistency between the given edge neighborhood and the predetermined allowable edge pattern set *Eedge* and non-edge pattern set $Eno-edge$. We assign an edge if *equation(II-34)* holds.

In this section we discuss how to determine the sets $Eno - edge$ and *Eedge* automatically neighborhood by neighborhood for any size of context support. The number of patterns produced by this scheme in either *Eedge* or $Eno-edge$ are independent of the selected context neighborhood size and are less than the average number of patterns in *Eedge* and $Eno - edge$ of the 3X3 dictionary based context edge detection(Lee, 1985).

We rewrite the joint probability term $P(\varepsilon_N^*, \theta_N^*)$ in equation (II-34) by $P(pattern_i) = P(\varepsilon_n^*, \theta_n^* \ \forall n \in pattern_i)$ where $pattern_i$ can either belong to $Eedge$ or $Eno - edge$ and the number of pixels of $pattern_i$ and the number of pixels of $pattern_j$ for $i \neq j$ may not be identical. We first consider the case that $pattern_i \in Eedge$. In this case, $P(pattern_i)$ can be written as $P(\varepsilon_n^* =' edge', \theta_n^*; n \in pattern_i)$. In our development, the pixels explicitly specified by $pattern_i$ consists of only pixels which have edge state 'edge'. Let $pattern_i$ have k+1 pixels which are addressed by n, where $n \in \{0, \ldots, k\}$. The term $P(pattern_i)$ can be decomposed into the following form

$$P(\varepsilon_n^* =' edge', \theta_n^*; n \in pattern_i) =$$

$$P(\theta_k^* | \varepsilon_n^* =' edge', n \in \{0, \ldots, k\}, \theta_m^*, m \in \{0, \ldots, k-1\}) *$$

$$\vdots$$

$$P(\theta_0^*, \varepsilon_n^* =' edge', n \in \{1, \ldots, k\} | \varepsilon_0^* =' edge') *$$

$$P(\varepsilon_0^* =' edge') \qquad (II-35)$$

By assuming that all the patterns belong to $Eedge$ have equal probability, we have

$$P(\theta_0^*, \varepsilon_n^* =' edge', n \in \{1, \ldots, k\} | \varepsilon_0^* =' edge') = \frac{1}{\#Eedge} \qquad (II-36)$$

where $\#Eedge$ designates the total number of patterns in $Eedge$.

In an edge line it is reasonable to assume that the edge angle of a pixel depends only on the edge angle of its immediate adjacent edge pixel and the relative position between them. Thus

$$P(\theta_i^*|\varepsilon_n^* =' edge', n \in \{0,\ldots,k\}, \theta_m^*, m \in \{0,\ldots,i-1\}) =$$

$$P(\theta_i^*|\varepsilon_n^* =' edge', n \in \{0,\ldots,k\}, \theta_{i-1}^*)$$

Due to the low cumulative curvature requirement of an edge curve in a small neighborhood, it is reasonable to assume that the conditional probability $P(\theta_i^*|\varepsilon_n^* =' edge', n \in \{0,\ldots,k\}, \theta_i^* - 1)$ has maximal value when the edge direction at the immediate adjacent neighbor agrees with that at the center. And the expected edge direction of the center pixel based on its position relationship with respect to the adjacent neighbor agrees with the center edge direction. To satisfy these requirements, we approximate the probability by

$$f(\theta_i^*, \theta_{i-1}^*) = P(\theta_i^*|\varepsilon_n^* =' edge', n \in \{0,\ldots,k\}, \theta_{i-1}^*) =$$

$$d(\theta_i^*, \theta_{i-1}^*) * d(\theta_i^* + \frac{\pi}{2}, \frac{M\pi}{4}) = f(\theta_i^*, \theta_{i-1}^*) \qquad (II-37)$$

where d is defined as

$$d(\alpha, \beta) = \frac{\cos(\alpha - \beta) + 1}{2}$$

and M is the adjacent edge position index with respect to the center pixel according to the following order

3	2	1
4	0	8
5	6	7

The range of function d is the closed interval [0, 1]. The reason why we select this nonlinear function for edge coherence probability is that as α approaches β the function has less penality (higher value) than the absolute difference between these two angles. Conversely, it gives more penality when the angle difference is large. Thus the angle quantization effect due to the rectangular grid layout of the pixels will tend to be minimized. By combining (II-35), (II-36) and (II-37) we have

$$P(pattern_i) = [\prod_{n=0}^{k} f(\theta_n^*, \theta_{n-1}^*)] * \frac{P(edge)}{\#Eedge} \qquad (II-38)$$

where for the purpose of notational convenience we define $f(\theta_0^*, \theta_{-1}^*)$ as 1 although θ_{-1}^* does not exist.

Note that for different patterns of *Eedge* set, the number of the edge pixels, thus k value, may be different. In order to take care of the difference in k on different patterns, we normalize $P(pattern_i)$ to single pixel basis. That is we define the normalized probability as

$$P_n(pattern_i) = [\prod_{n=0}^{k} f(\theta_n^*, \theta_{n-1}^*)]^{\frac{1}{k+1}} * \frac{P(edge)}{\#Eedge} \qquad (II-39)$$

Similarly, in the case that $pattern_j \in Eno-edge$, $P(pattern_j)$ can be written as

$$P(pattern_j) = \frac{(1-P(edge))}{(2\pi)^{k+1} * \#Eno-edge} \qquad (II-40)$$

We assume that given a group of non-edge pixels the conditional probability of their gradient directions are uniform and independent. We also assume that the conditional probability density of edge direction is uniform for non-edge pixels. That is

$$P(\theta_n^* | \varepsilon_n^* =' no-edge') = \frac{1}{2\pi} \; \forall n \in \{0, \ldots, k\}$$

Similar as in the $pattern_i \in Eedge$ case, the normalized $P(pattern_j)$ can be expressed as

$$P_n(pattern_j) = \frac{(1-P(edge))}{(2\pi) * \#Eno-edge} \qquad (II-41)$$

Since we can normalize the conditional probability term of equation (II-34) into single pixel basis as

$$[\prod_{n=0}^{k} P(\underline{k}_n | \varepsilon_n^*, \theta_n^*)]^{\frac{1}{k+1}}$$

the normalized version of equation (II-34) becomes assign ε_0 as an edge pixel if

$$\sum_{pattern_i \in Eedge} [\prod_{n \in pattern_i} P(\underline{k}_n | \varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)]^{\frac{1}{\#pattern_i}}$$

$$* \frac{P(edge)}{\#Eedge}$$

$$> \sum_{pattern_i \in Eno-edge} [\prod_{n \in pattern_i} P(\underline{k}_n | \varepsilon_n^* =' no-edge', \theta_n^*)]^{\frac{1}{\#pattern_j}}$$

$$* \frac{(1 - P(edge))}{2\pi * \#Eno-edge} \qquad (II-42)$$

where $\#pattern_i$ and $\#pattern_j$ are the numbers of pixels which are 'edge' in $pattern_i$ and 'no-edge' in $pattern_j$, respectively. To perform the edge detection scheme we have to determine the sets $Eedge$ and $Eno-edge$. As we mentioned before the number of patterns in $Eedge$ and $Eno-edge$ are kept as small as possible and are independent of the neighborhood size. Thus, we only pick up some edge patterns which maximize the term

$$[\prod_{n \in pattern_i} P(\underline{k}_n | \varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)]^{\frac{1}{\#pattern_i}} \qquad (II-43)$$

and put them in the set $Eedge$. For example, we only pick up the pattern among all the possible single edge curve patterns which maximizes (II-43) as one of the patterns in $Eedge$. The set $Eedge$ can include one single edge curve pattern, one edge junction (two edge curves merge at the center point) pattern, one edge corner pattern, and one crossing edge curve (two edge line meet at the center point) pattern.

To find the patterns which maximize (II-43), a dynamic programming based technique is introduced. The dynamic programming technique determines four optimal edge curves (possibly merging into three or two edge curves) in four directions emanating from the center pixel. The four optimal edge curves are the optimal edge curves starting from the center pixel toward the upward , downward, left and right boundaries of the selected neighborhood, respectively. *Figure II-2* shows an example of these curves in a 9 X 9 neighborhood.

Consecutive edge pixels in any optimal edge curve must be 8-connected. Under this constraint, both the optimal edge curves toward the upward and downward boundaries may in some special cases coincide with the optimal edge curves toward the left or right boundaries. That is, curve (1) could coincide with curve (3) or curve (4). Similarly, curve (2) could also coincide with curve (3) or curve (4).

When determining the allowable edge pattern set *Eedge*, the candidates for the edge patterns belonging to *Eedge* are all of the possible combinations of the four optimal edge curves emanating from the center pixel. The patterns selected among these candidate patterns for the edge detecting process are in accordance with prior knowledge about the image being processed. The prior knowledge includes the belief about: the existence of a single edge curve, two crossing edge curves meeting at the center pixel, edge corners or edge junctions (three edge curves meeting at the center pixel) in a neighborhood. For example, if the single edge curve pattern is chosen, the pattern

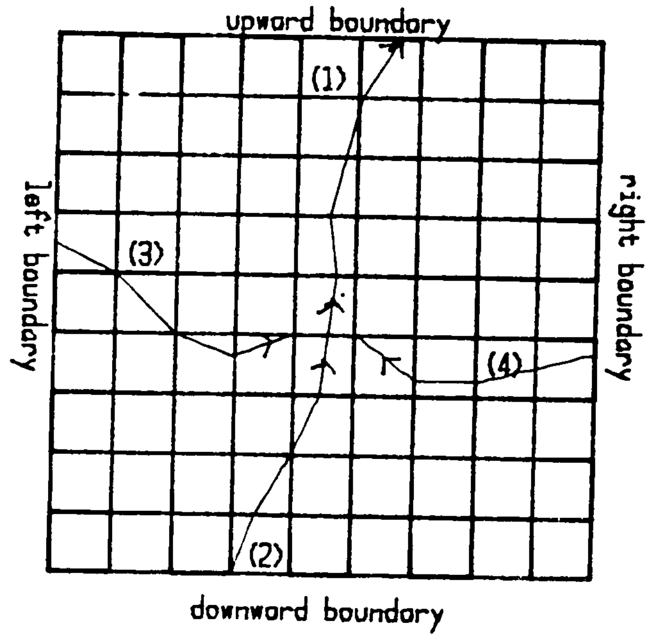


Figure II-2. Optimal edge line in a 9X9 neighborhood.

between (*curve(1)* ∪ *curve(2)*) and (*curve(3)* ∪ *curve(4)*) of *Figure II-2* and which maximizes (II-43) is put into the set *Eedge*. Similarly, if the edge corner pattern is chosen, the pattern which has highest probability among (*curve(1)* ∪ *curve(4)*), (*curve(1)* ∪ *curve(3)*), (*curve(2)* ∪ *curve(4)*) and (*curve(2)* ∪ *curve(3)*) is included in *Eedge*. The same process holds for two crossing edge lines and edge junction cases as well. Furthermore, to reduce the possible bias caused by using context alone, we also put the pattern which has only center pixel in it and has edge state 'edge' as a pattern of *Eedge*.

Now, we show how we construct the set $Eno - edge$. The process of construction goes through ten 3X3 non-edge neighborhood patterns (see *Figure II-3*) and picks up among them four patterns. In general, a local neighborhood of an image should be smooth. If the center pixel in a neighborhood has true edge state 'no-edge', it is most probable that the pixels belonging to the eight-connected neighborhood surround this pixel are also non-edge pixels. Therefore, the pattern which has all non-edge pixels in a 3 X 3 neighborhood surround a center non-edge pixel is included in the allowable pattern set $Eno - edge$. (see *Figure II-3(a)*).

In order to take care of the possibility of non-smooth neighborhoods such as the non-edge pixels immediately adjacent to an edge curve, the pattern among the four patterns of *Figure II-3(b)* which maximizes (II-43) is put in the set $Eno - edge$. The patterns of *Figure II-3(b)* are the patterns which include six 4-connected non-edge pixels around the center pixel in a 3 X 3 neighborhood. That is, we take care of the cases of a non-edge pixel either vertically or horizontally adjacent to an edge line. Similarly, we put in the set $Eno - edge$ the pattern which maximizes (II-43) among all the patterns belonging to

Figure II-3(c). Finally, we introduce the context free effect into the edge detecting process by including the pattern which has a single non-edge center pixel as a pattern of $Eno-edge$. The inclusion of the context free pattern reduces the possibility of bias caused by the context effect alone. It is noted that no matter how large the context neighborhood is selected, the non-edge pattern set only includes the patterns we described above. All the pixels within the patterns which are specified by 'X' are considered as *'don't care'* pixels. The 'don't care' pixel can be either 'edge' or 'no-edge'. Since the set $Eno-edge$ has only four patterns. It requires little computational and storage efforts when performing the edge detection process.

As we mentioned before the edge pattern which maximizes the term

$$[\prod_{n \in pattern_i} P(\underline{k}_n|\varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)]^{\frac{1}{\#pattern_i}}$$

is the optimal edge pattern and shall be put in the set $Eedge$.

We now describe how we can use a dynamic programming technique to find the optimal edge curves starting from the center pixel toward the upward, downward, left and right boundaries of the selected neighborhood.

The dynamic programming scheme we use to find the optimal edge curve is subjected to the constraint that the consecutive edge

```
0 0 0
0 0 0
0 0 0
```

(a)

```
0 0 0    X X X    0 0 X    X 0 0
0 0 0    0 0 0    0 0 X    X 0 0
X X X    0 0 0    0 0 X    X 0 0
```

(b)

```
0 X X    0 0 X    X X 0    X 0 0
0 0 X    X 0 0    X 0 0    0 0 X
X 0 0    X X 0    0 0 X    0 X X
```

(c)

```
X X X
X 0 X
X X X
```

(d)

Figure II-3. The non-edge neighborhood patterns. '0' designates non-edge pixel,'x' designates *'don't care'*.

pixels in any optimal edge line must be 8-connected neighbors. Let x_0 be the center edge pixel which is the starting point of the optimal edge curve. Let x_1 be the successive edge pixel of x_0 and x_2 be the successive edge pixel of x_1, etc. Then the constraint is

$$\|x_k - x_{k+1}\| \leq \sqrt{2}$$

Since logarithm is an increasing function, the optimal edge line which maximizes the probability of (II-43) is the same as the optimal edge line which maximizes

$$\frac{1}{\#pattern} \sum_{n \in pattern} log[P(\underline{k}_n | \varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)]$$

For a given neighborhood, the $\#pattern$ value is a constant, Therefore the cost function we use for optimization can be simply

$$\sum_{n \in pattern} log[P(\underline{k}_n | \varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)]$$

Let's order the pixels belonging to the edge line and designate them as $x_0, x_1, \ldots, x_l$. Thus, for a selected neighborhood of size $(2l+1)$ by $(2l+1)$, x_0 represents the starting pixel which is the center of the neighborhood; x_1 represents the first layer pixel belonging to the set of 8-connected neighbors of x_0; x_2 represents the second layer pixel belonging to the set of pixels in one side of boundary of the neighborhood. Thus, the optimal curve generating problem becomes finding $x_i, i \in \{1, \ldots, l\}$ such that the following term is maximized:

$$\sum_{i=1}^{l} log[P(\underline{k}_n | \varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)] \qquad (II-44)$$

Let's define

$$h_i(x_{i-1}, x_i) = log[P(\underline{k}_n | \varepsilon_n^* =' edge', \theta_n^*) f(\theta_n^*, \theta_{n-1}^*)]$$

Then (II-44) becomes

$$\max_{x_i} \sum_{i=1}^{l} h_i(x_{i-1}, x_i)$$

We also define

$$g_m(x_m) = \max_{x_i, \ldots, x_{m-1}} \sum_{i=1}^{m} h_i(x_{i-1}, x_i)$$

It is easy to verify that

$$g_m(x_m) = \max_{x_{m-1}} [g_{m-1}(x_{m-1}) + h_m(x_{m-1}, x_m)] \qquad (II-45)$$

and

$$g_0(x_0) = \log[P(\underline{k}_{x_0} | \varepsilon_{x_0}^* =' edge', \theta_{x_0}^*)] \qquad (II-46)$$

we can then carry out the maximization by a dynamic programming process. The solution for (II-43) is finally

$$\max_{x_l} g_l(x_l)$$

where

$$g_l(x_l) = \max_{x_{l-1}}[g_{l-1}(x_{l-1}) + h_l(x_{l-1}, x_l)]$$

Let the coordinate of x_0 be (r_0, c_0), then in the case of an 8-connected edge line starting from the center pixel ending at the upward boundary. We have the domain of the adjacent edge candidates in layer i, $i \in \{1, \ldots, l\}$ be

$$x_i \in \{(r_0 - i, c_0 - i), \ldots, (r_0 - i, c_0), \ldots, (r_0 - k, c_0 + l)\} \quad (II - 47)$$

Similar situations hold for the cases of edge curves toward the bottom, left or right boundaries. A standard dynamic programming scheme can be employed to solve the maximization problem defined by (II-46) under the constraint of (II-47).

The scheme we describe in this section is related to the dynamic programming scheme of Montanari (1971) and the heuristic search scheme of Martelli (1976). However, both Montanari's and Martelli's scheme are postprocessing schemes for linking edge segments. They heavily depends on the initial point selected for linking process and they can only take care of a single edge line passing through the selected edge point. The dynamic programming scheme in this section does not depend on the selection of initial point and can take care of not only single edge lines but also edge corners, edge junctions and two crossing edge lines.

The heuristic search scheme proposed by Martelli is just a brute force depth-first search. If the edge consistent cost function is pairwise, a dynamic programming scheme can give a more efficient solution. Besides, the solution found by dynamic programming is optimal while the solution found by heuristic search can be suboptimal.

II-5. Experimental results

To understand the performance of the context dependent edge detectors we first examine the behavior of the dictionary look up 3 X 3 neighborhood context edge operator on three well structured simulated images so that we can see how the mechanism we employed can use context information effectively. We then apply the general dynamic programming context edge operator on three noisy object images and compare its performance with the context free operator to see how the context scheme can achieve better object identification. The first simulated image is a checkerboard of size 100 x 100 pixels with a check size of 20 x 20 pixels. The dark checks have gray tone intensity 75 and the light checks have gray tone intensity 175. To this perfect checkerboard, we add independent Gaussian noise having mean zero and standard deviation 35. The perfect and noisy checkerboards are shown in *Figure II-4*.

In order to compare the performance of different edge operators, we use the conditional probability of assigned edge on state 'edge' given the true edge states 'edge' of an image, $P(E'|E^*)$, and the conditional probability of true edge states 'edge' given assigned edge states 'edge' of an image $P(E^*|E')$. The adjustable parameters of each edge operator are chosen to equalize these two conditional probabilities. The quality of the edge operator is determined by the value of $P(E'|E^*) = P(E^*|E')$. Although this performance measure is not in general applicable on all kinds of images, it is well suitful for the simulated checkerboard image.

We apply context dependent edge detectors and the context free second directional derivative zero-crossing edge operator (Haralick, 1984) on the noisy checkerboard and compare the performance in terms of $P(E'|E^*)$ and $P(E^*|E')$. In both cases, the image is first fitted by cubic polynomials with neighborhood size 5 x 5. Table II-1 lists the test results of these edge operators. The results show that the conditional probabilities are improved from 82.47 percent to 96.10 percent by the context dependent edge operator.

Figure II-5 shows the edge images of these edge operators. A visual evaluation also leaves the impression that the context dependent edge detector produces much better edge continuity and has less noise than the context free detector.

As described in section II-2 the edge detecting scheme takes into consideration both edge gradient and zero crossing of second-directional derivative and treat both effect equally. Therefore, it can detect a low gradient edge pixel if the pixel has high evidence of a zero crossing of second derivative. It is also noted that the context free second directional derivative zero-crossing edge operator is basically a

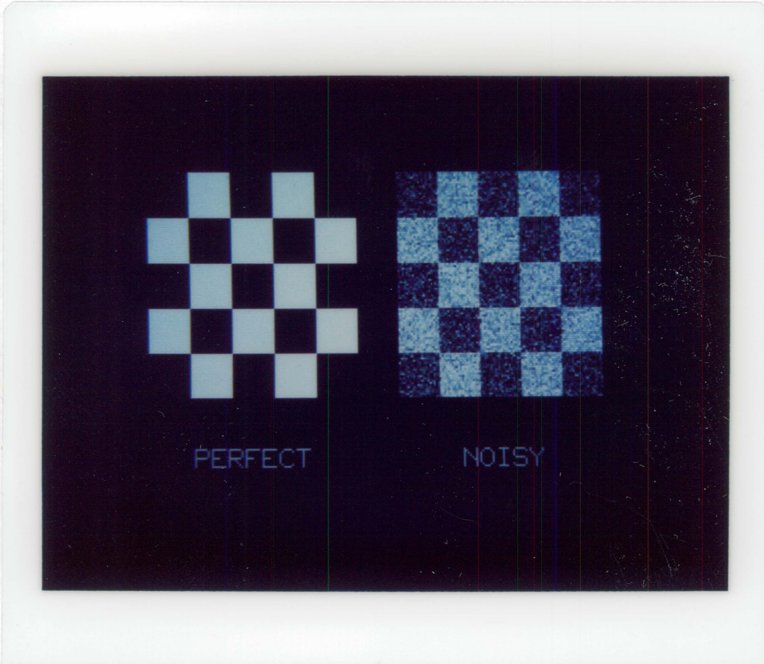


Figure II-4. Shows the perfect checkerboard image and the noisy image with $\sigma_n = 35.0, \mu_n = 0.0$. The constrast between the bright and dark checks is 100.

Table II-1. $P(E'|E^*)$ and $P(E^*|E')$ values of context free second directional derivative edge operator and context operators on a noisy checkerboard image with 5 x 5 cubic facet window size.

prob \ operator	*context free operator*	*context operator*
$P(E'\|E^*)$	0.8247	0.9610
$P(E^*\|E')$	0.8247	0.9610

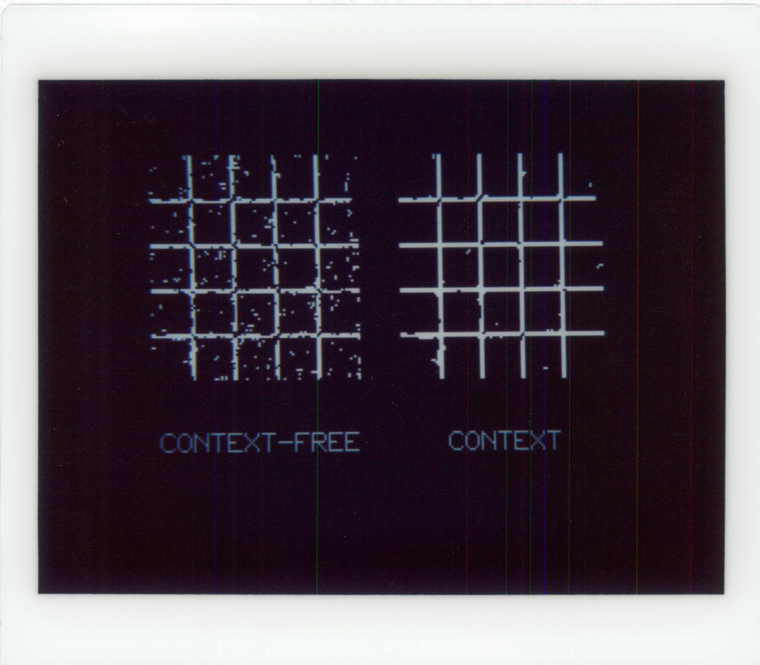


Figure II-5. The edge images of the context free second directional derivative zero crossing edge operator and the context edge operator. The selected context pattern sets are Edge pattern set 6 and Non-edge pattern sets 1 and 2. The other parameters are selected to equalize the probabilities $P(E'|E^*)$ and $P(E^*|E')$.

gradient based operator. It detects an edge pixel if the pixel has both high gradient and zero-crossing of second derivative in the direction of gradient. If the pixel does not have enough gradient strength, it will not be classified as an edge pixel even though it has zero-crossing of second derivative. The combined gradient and second derivative edge detecting scheme can detect non-obvious edges which have less than average gradient and which are actually edge pixels.

In order to see the effect of context, a context free edge image generated by equation (II-22) is shown in Figure II-6. From this figure, we can see that the context free edge image has fewer edges than the context dependent edge image. The context dependent operator, then, produces a strong clean edge image with little noise compared with the context free edge image.

Unlike most of the previous reported edge operators, the context edge operator does not use a pure thresholding scheme. The adjustable parameter λ which is the ratio between the standard deviation of cubic facet parameter k_2 between edge and non-edge pixels. From equation (II-18) and (II-19) we know that increasing λ will increase both mean and variance of the probability $P(k_2, k_3|\varepsilon^* =' edge', \theta^*)$ and simultaneously decrease the variance of $P(k_2, k_3|\varepsilon^* =' no - edge', \theta^*)$. Thus, increasing λ may increase the total number of edges and also may decrease the total number of

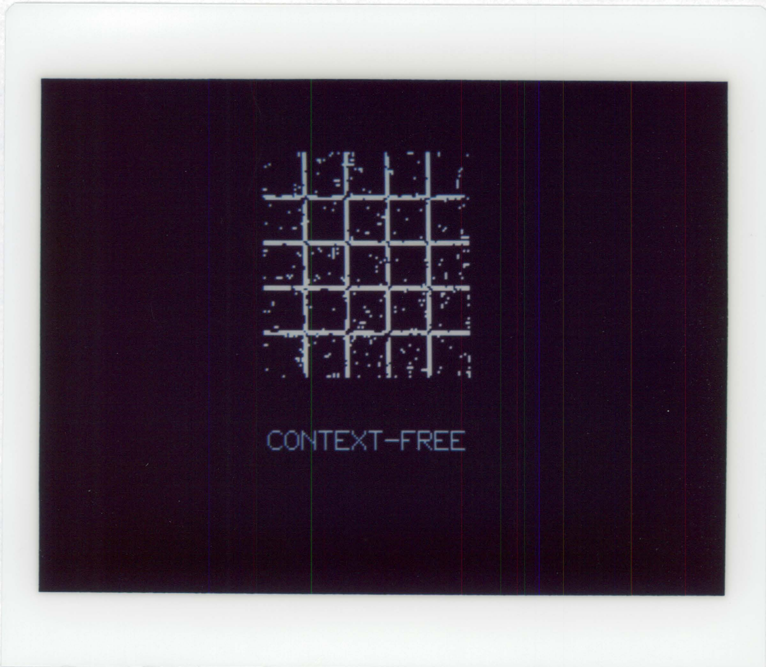


Figure II-6. The edge image of the context free edge operator with same λ and same cubic facet window size as the context dependent one of *Figure II-5(b)*.

edges depending on the distributions of the conditional probabilities. Then this scheme avoids the drawback of a pure thresholding which assigns pixels falling above the threshold as edge pixels while those below the threshold as non-edge pixels. Since the gradient of edge boundaries are often relative, a gradient which is higher than the average gradient of a homogeneous area of an image may be much lower that the average gradient in a textured area of the same image.

As we mentioned before, the edge operator based on combined gradient and second derivative can detect non-obvious edges in a homogeneous image area which have less than average gradient values. To see this effect, we generate a bar image of size 40 x 60 pixels. There are six vertical bars on this image. Each bar has length 40 pixels and width 10 pixels. The gray tone intensities of the bars are, from left to right, 216, 125, 64, 27, 8, and 4 respectively. We then add a signal dependent zero mean Gaussian noise to the image. The standard deviation of the noise is, from left bar to right bar, 36, 21, 10, 4, 0 and 0 respectively.

By applying the context free second directional derivative zero-crossing edge operator on the noisy image with cubic facet neighborhood size 5 x 5, we find that no satisfactory result can be obtained. *Figure II-7(a)* shows the noisy bar image and (b) - (e) show the results of the context free edge operator with different threshold values. The results illustrate that large threshold values tend to miss edge boundaries in the right portion of the image, while small threshold values introduce lots of noise in the left portion of the image. We then apply the combined gradient and second derivative context dependent edge operator on the same image. The result shows (see *Figure II-7(f)*) that we have a superior edge image to the images of *Figure II-7(b) - (e)*.

In order to see the performance of the edge detectors in the non-vertical and non-horizontal edge features, we rotate the noisy checkerboard image of *Figure II-4* by 45 degrees and apply both context free second directional derivative zero-crossing edge operator and the context dependent edge operator on this rotated checkerboard image (see *Figure II-8*).

We also set up two more edge pattern sets and two more non-edge pattern sets for the context dependent edge operator so that it can better take care of the diagonal edge features. These patterns are named as Edge pattern (5,1) through Edge pattern (6,4) and Non-edge pattern (4,1) through Non-edge pattern (5,4). A diagram of these patterns is shown in *Figure II-9.* The test results of both operators with 5 x 5 cubic facet window size are listed in terms of the conditional probabilities $P(E'|E^*)$ and $P(E^*|E')$ in *Table II-2.* Due to the effect that we use a square neighborhood for cubic polynominal fitting,

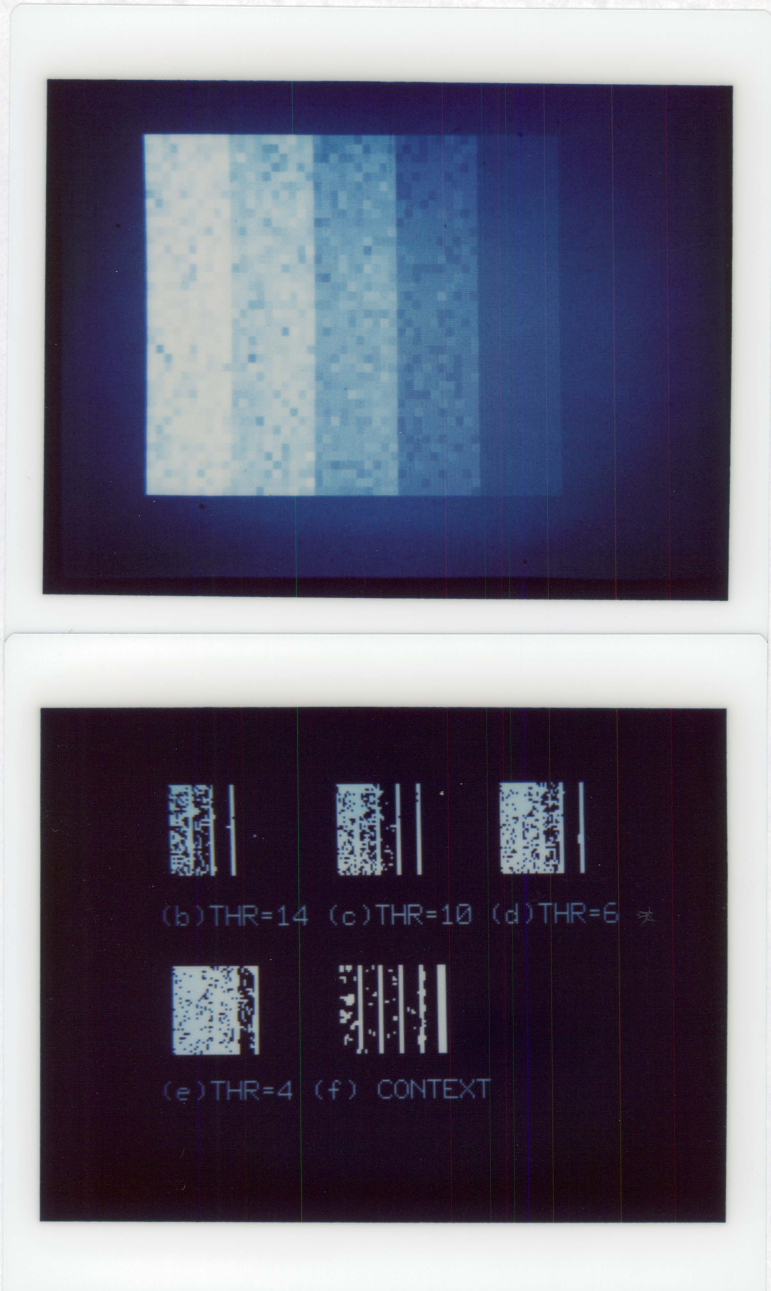


Figure II-7. (a) The noisy bar image and context free edge images with (b) threshold = 14; (c) threshold = 10; (d) threshold = 6; (e) threshold = 3. (f) the context dependent edge image with selected edge context patterns, Edge pattern set 6 and Non-edge pattern sets 1 and 2.

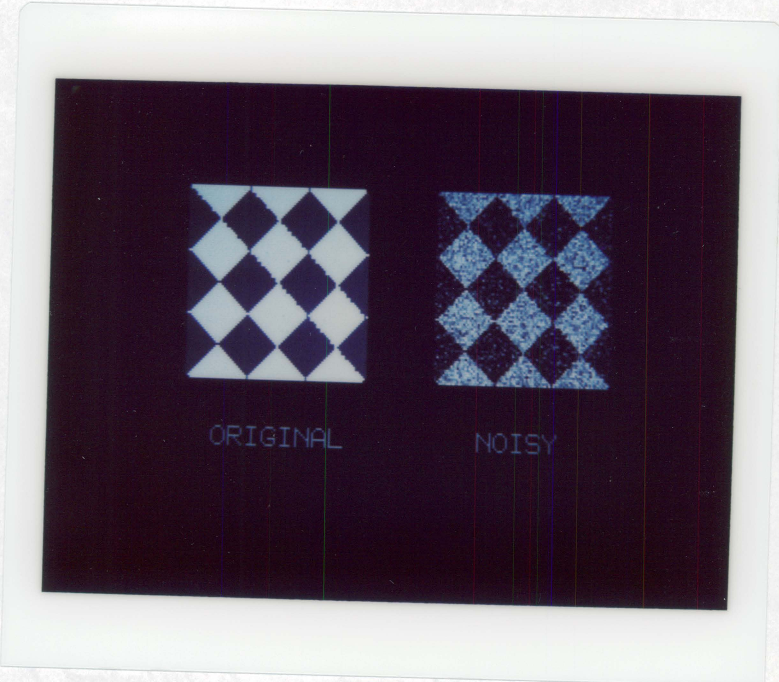


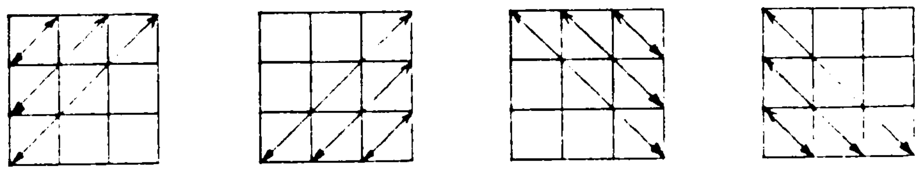
Figure II-8. The pictures of the rotated checkerboard image and the noisy image. Where the dark checks have gray tone intensity 75 and the light checks have gray tone intensity 175; the added noise has $\sigma_n = 35.0$ and $\mu_n = 0.0$.

the performance of both edge detectors on the rotated checkerboard are worse than the normally oriented checkerboard. The conditional probability $P(E'|E^*)$ is raised by the context dependent edge operator from 63.45 percent to 87.41 percent. *Figure II-10* shows the edge images of both context free and context dependent operators. The superior performance of the context dependent operator can be easily verified by visual evaluation.

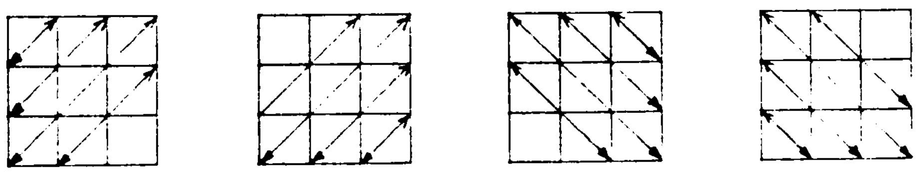
In order to see the performance of context operator and context free operator under different noise level, we add zero mean Gaussian noise of standard deviation 10, 20, 30, 45, 55, and 70 on the perfect checkerboard image (see *Figure II-4*), respectively. We then apply both context and context free second derivative zero crossing operator on these images. The adjustable parameters of each operator are chosen to equalize the two conditional probabilities as possible. The result edge images of both operstors are shown in *Figure II-11.* And the edge probabilities of each edge image are listed in *Table II-3, and II-4.* Also the average edge probabilities are ploted as curves and shown in *Figure II-12.*

It is interesting to see that according to the edge probability measure when the noise has standard deviation 10, the context edge operator performs worse than the context free operator. As the noise increases the context operator has higher probability than the context

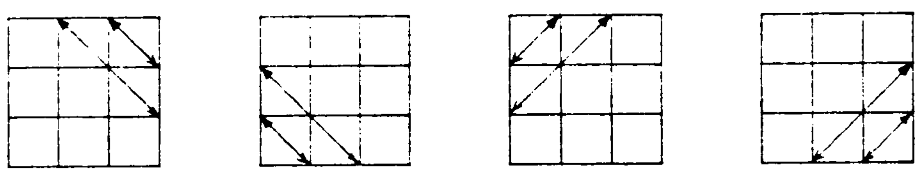
Edge pattern (5,1) - (5,4)



Edge pattern (6,1) - (6,4)



No-edge pattern (4,1) - (4,4)



No-edge pattern (5,1) - (5,4)

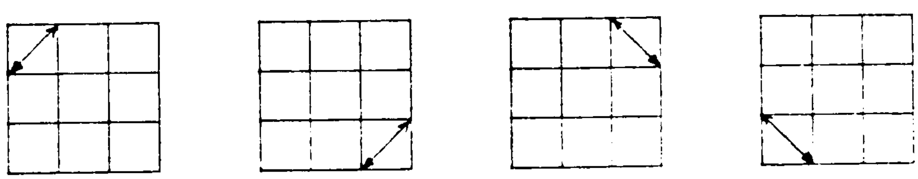


Figure II-9. The additional edge and non-edge context patterns used in the rotated checkerboard edge detection.

Table II-2. $P(E'|E^*)$ and $P(E^*|E')$ values of context free second directional derivative edge operator and context operators on a noisy rotated checkerboard image. The cubic facet used has 5X5 window size.

prob \ operator	*context free operator*	*context operator*
$P(E'\|E^*)$	0.6345	0.8741
$P(E^*\|E')$	0.6354	0.8674

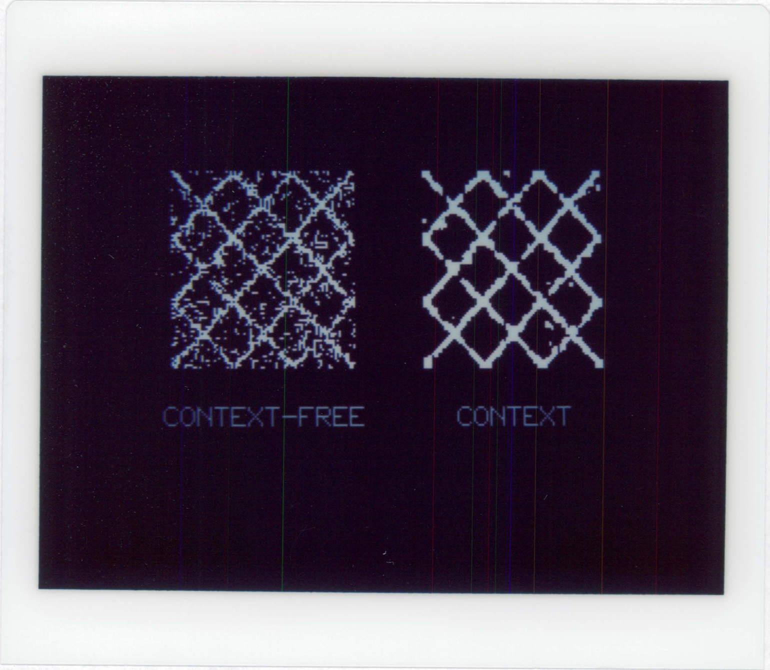


Figure II-10. The edge images of the context free second directional derivative zero crossing edge operator and the context dependent edge operator. The selected context edge patterns are edge pattern sets 7, 8 and non-edge pattern sets 1, 4, 5.

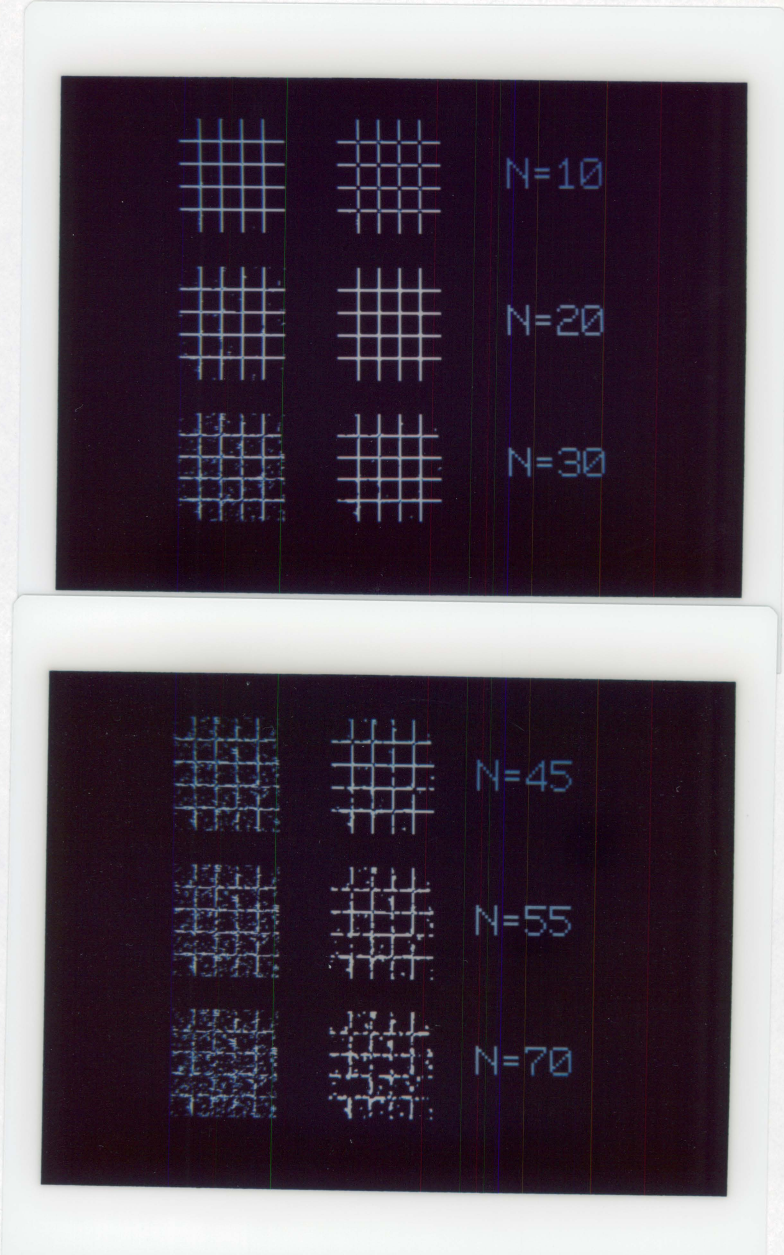


Figure II-11. The edge images of the context operator and context free operator on the checkerboard image with signal to noise ratio (SNR) 10, 5, 3.3, 2.2, 1.8, and 1.43, respectively. The signal to noise ratio (SNR) is defined as $\frac{edge\ contrast}{RMS\ noise}$. The window size for cubic polynomial fitting are 5 X 5. The selected context edge patterns are Edge pattern set 6 and Non-edge pattern sets 1 and 2.

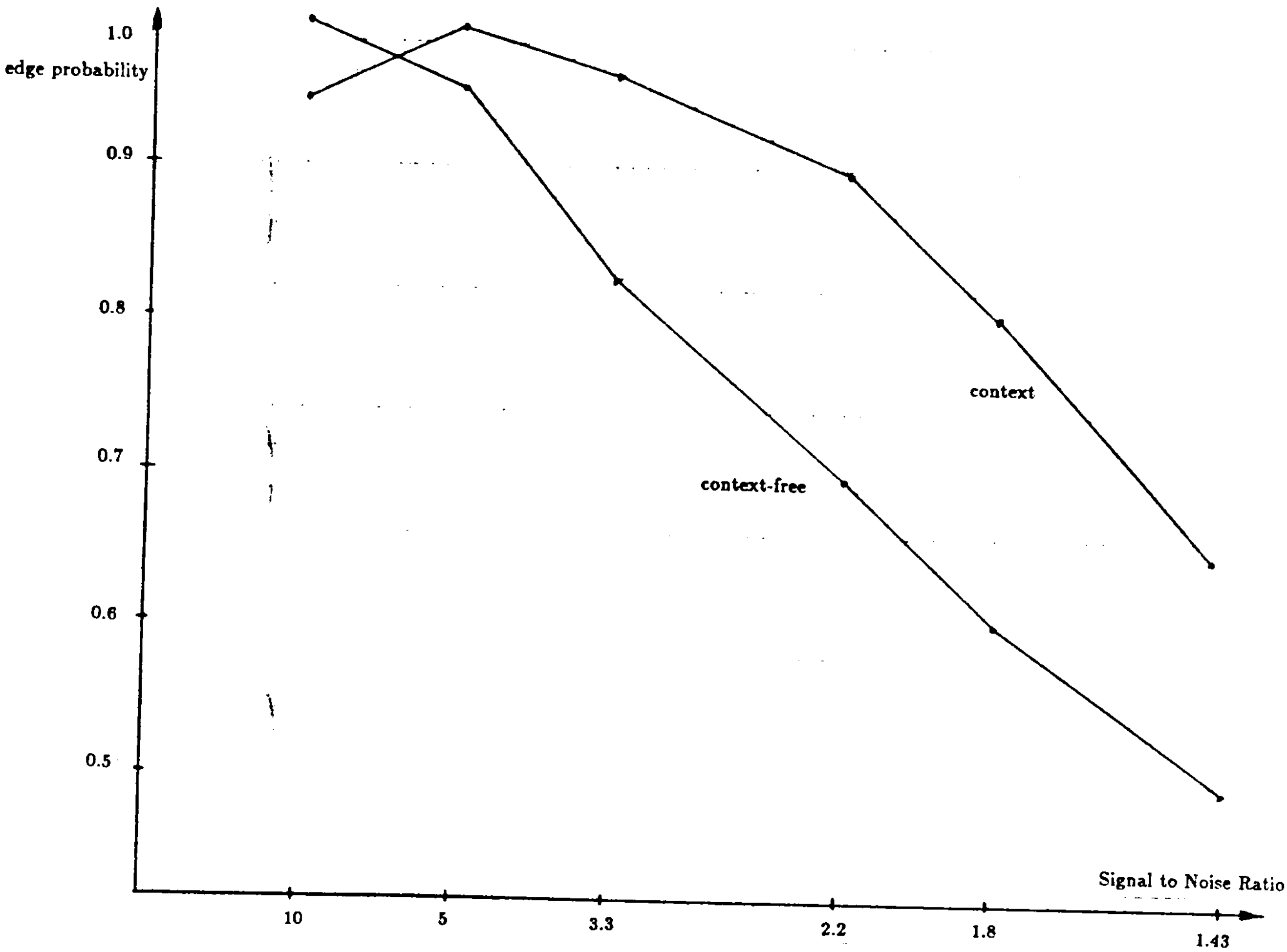


Figure II-12. The average probability $(\frac{P(E'|E^*)+P(E^*|E')}{2})$ curves of the context operator and context free operator on the checkerboard image with signal to noise ratio of 10, 5, 3.3, 2.2, 1.8, and 1.43, respectively.

Table II-3. $P(E'|E^*)$ and $P(E^*|E')$ values of context free second directional derivative edge operator on noisy checkerboard images with different noise levels.

noise \ *prob*	$P(E^*\|E')$	$P(E'\|E^*)$
$SNR = 10$	0.9935	0.9935
$SNR = 5$	0.9487	0.9673
$SNR = 3.3$	0.8246	0.8246
$SNR = 2.2$	0.7078	0.7078
$SNR = 1.8$	0.6234	0.6234
$SNR = 1.43$	0.4967	0.4935

Table II-4. $P(E'|E^*)$ and $P(E^*|E')$ values of context edge operator on noisy checkerboard images with different noise levels.

noise \ *prob*	$P(E^*\|E')$	$P(E'\|E^*)$
$SNR = 10$	1.0000	0.8888
$SNR = 5$	0.9807	1.0000
$SNR = 3.3$	0.9610	0.9610
$SNR = 2.2$	0.9019	0.8961
$SNR = 1.8$	0.8092	0.8039
$SNR = 1.43$	0.6493	0.6493

free operator. The difference between the edge probability increases as $\sigma_n \leq 45$ and then the difference decreases as $\sigma_n > 45$. We can also explain the reason why the context operator has less edge probability than the context free operator when $\sigma_n = 10$. The ideal edge image we used for the probability measurement includes all the saddle points of the checkerboard as edge points and the allowable edge context for the context operator does not include the pattern of two edge lines cross the center edge pixel. Therefore, the context operator only detect the edges other than the saddle points. Although the context operator can perfectly detect all the edge points other than the saddle points(see *Figure II-11 and Table II-3*), the edge probability measure is bad. However, this result is caused by the inconsistency between ideal edge image and the selected edge context. Nothing is wrong with the context operator.

To see the performance of the general context edge operator and compare its performance with the local context edge operator and the context free second derivative zero-crossing edge operator, we first apply it on the noisy checkerboard image of Signal to Noise Ratio 2.22. The edge probability results are listed in *Table II-5*.

It is noted that the general context edge operator has better performance than the context free edge operator. It rises the edge probability of the context free edge operator from 70 percent to 81

Table II-5. $P(E'|E^*)$ and $P(E^*|E')$ values of the general context edge operator, the local context edge operator, and the context free second derivative zero-crossing edge operator apply on a noisy checkerboard images of SNR 2.22.

operator \ prob	$P(E^*\|E')$	$P(E'\|E^*)$
General Context	0.8116	0.8116
Local Context	0.9019	0.8961
Context Free	0.7078	0.7078

percent. The general context edge operator has worse performance compared with the local context edge operator. This is because that the general context edge operator uses context in a general way while the local context edge operator picks up edge context which specifically fits the image under processed. However, the local context edge operator depends heavily on the prior information about the local edge context which is not generally available.

Finally, we apply the general context edge detector on three noisy object images. The object images are simulated range images with image size 128 by 128 pixels on which we add zero mean Gaussian noise of standard deviation 30. The images are shown in *Figures II-13(a), II-14(a) and II-15(a).* On these images we first apply a cubic polynominal fitting with neighborhood size 5 X 5 and then apply both context free second directional derivative zero crossing edge operator and the dynamic programming based context dependent edge detector. The window for context pattern is also 5 X 5. The result of the context free edge operator on these images are shown in *Figure II-13(b), II-14(b) and II-15(b).* And the result of the general context edge operator on these images are shown in *Figure II-13(c), II-14(c) and II-15(c).* It can be easily verified that on all of these images the context edge operator better. The edge images of the context edge operator have better connectivity and much less noisy than the context free edge operator.

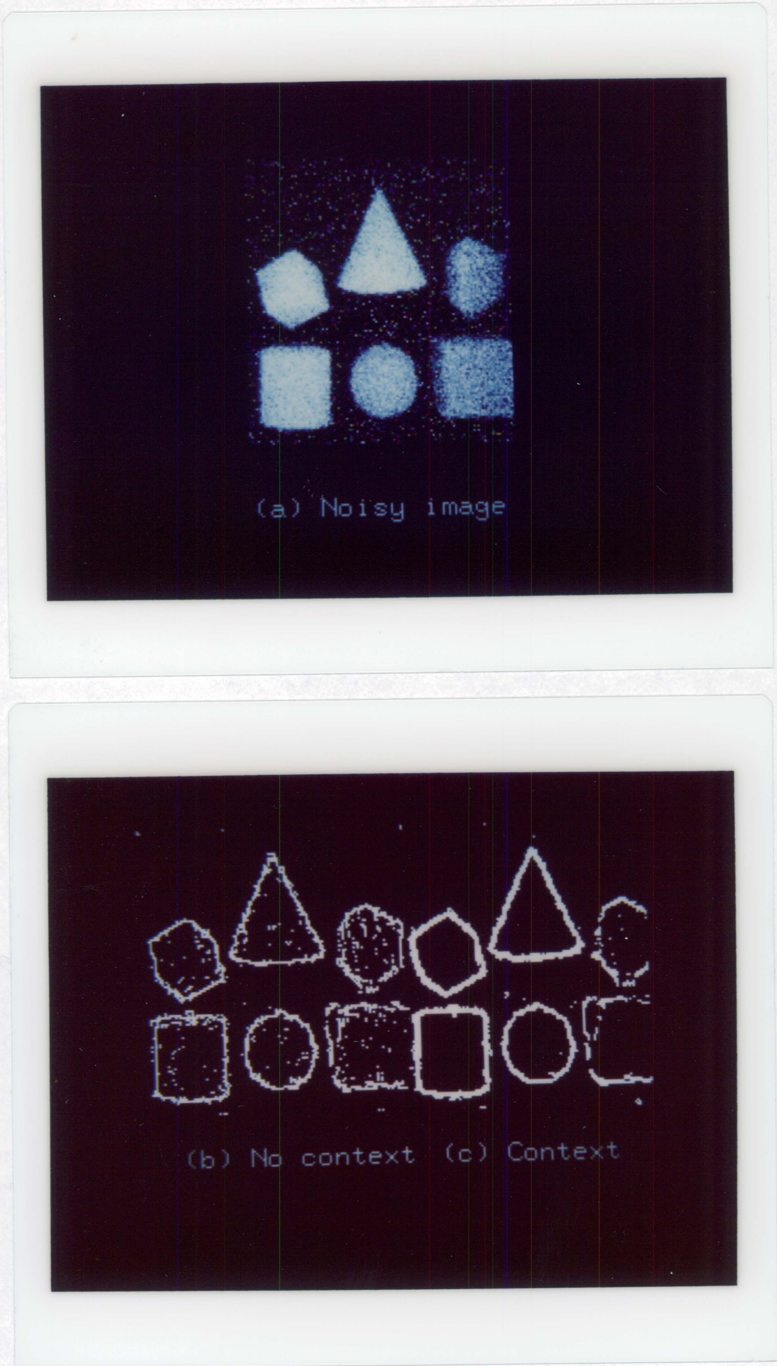


Figure II-13. The noisy object image 1 (a) and its context free edge image (b) and context edge image(c). The window size for cubic polynomial fitting and context pattern are both 5 X 5.

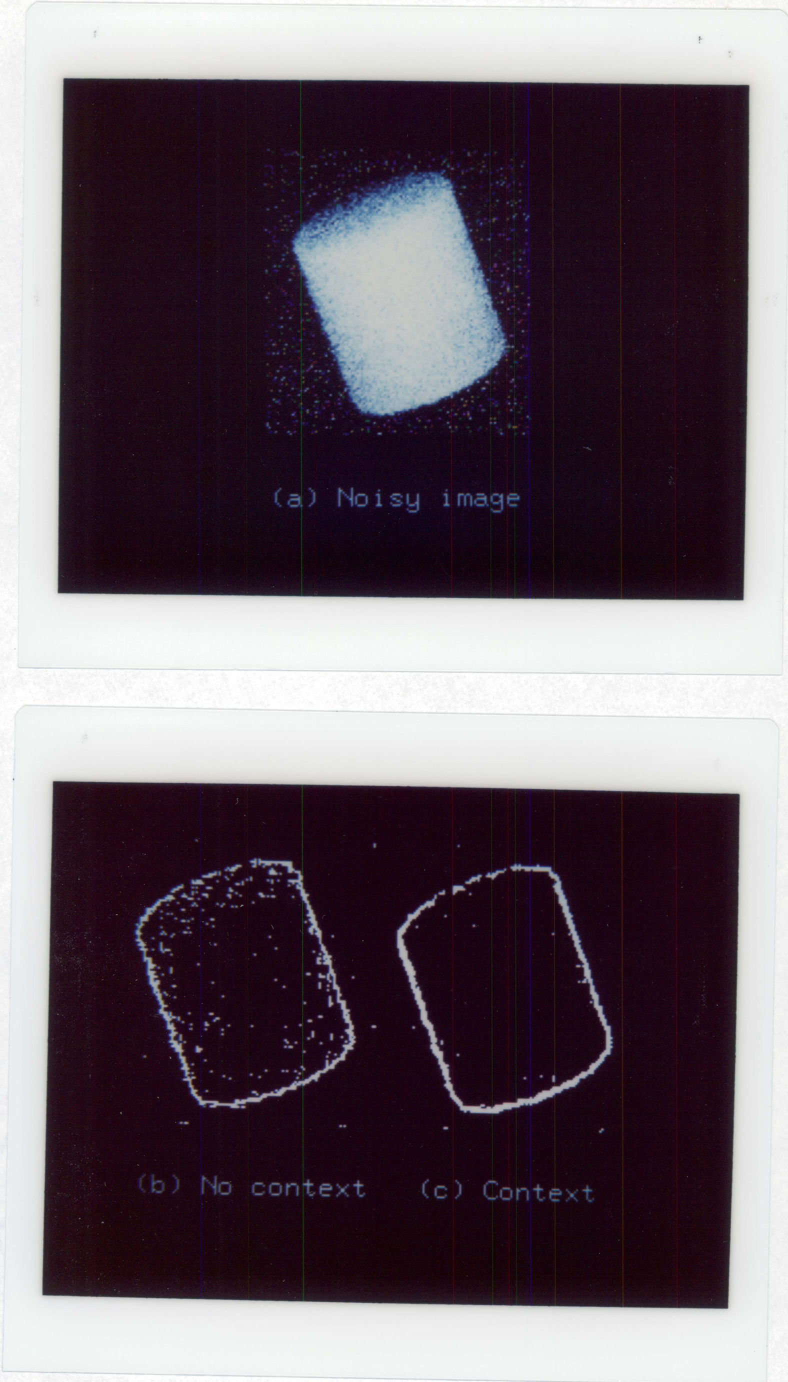


Figure II-14. The noisy object image 2 (a) and its context free edge image (b) and context edge image(c). The window size for cubic polynomial fitting and context pattern are both 5 X 5.

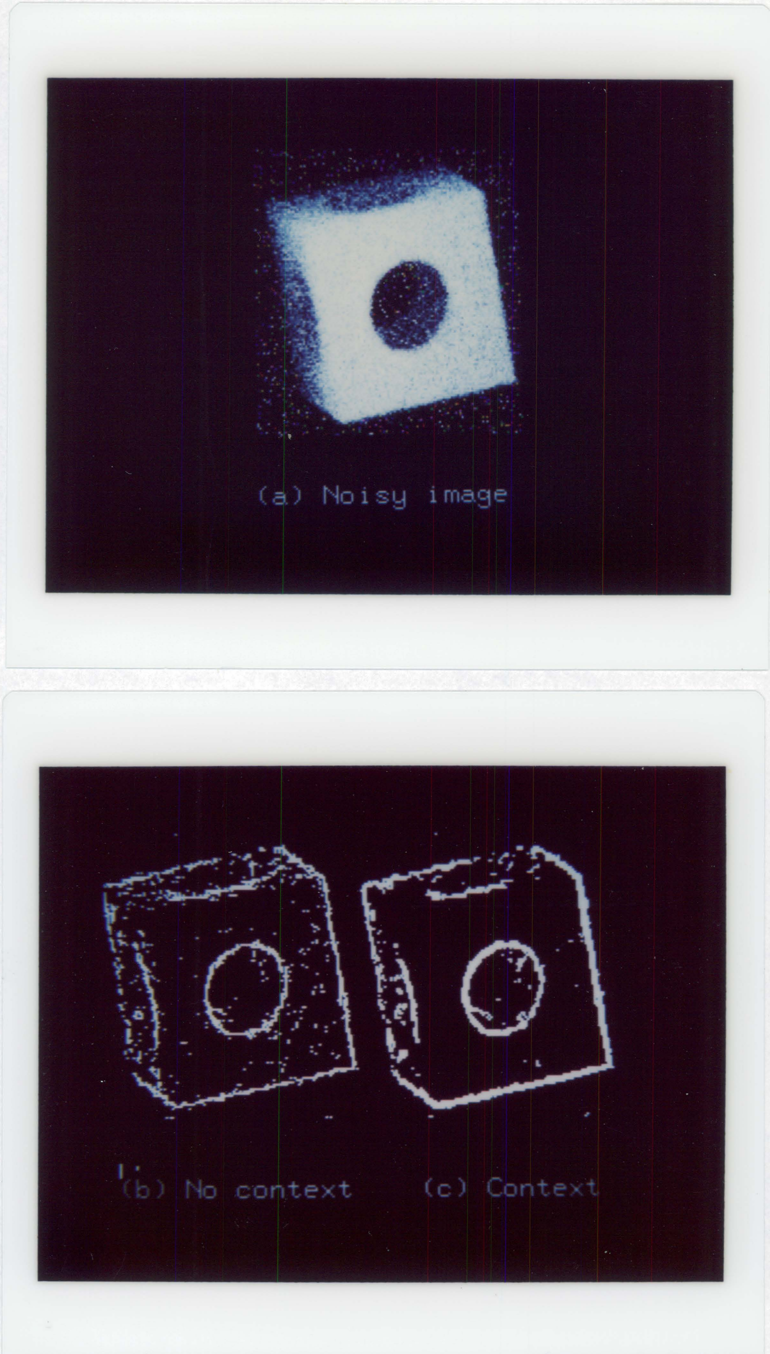


Figure II-15. The noisy object image 3 (a) and its context free edge image (b) and context edge image(c). The window size for cubic polynomial fitting and context pattern are both 5 X 5.

III. FULL CONTEXT EDGE DETECTION

III-1. Algorithms

In chapter II, we introduced a context dependent edge detection scheme. The context we used in chapter II is basically the edge data in the neighborhood of the pixel under consideration. Any pixels outside this neighborhood have nothing to do with the edge detection of the current pixel. In this chapter we introduce a context dependent edge detection scheme which uses all the edge data in the image as the context to help the edge detection process of any pixel.

According to chapter II, the edge detection problem can be formulated as a Baysian decision problem. The solution to this problem under the zero-one gain function is: for each pixel position (r,c) of the image independently assign the edge state ε_{rc} as 'edge' if

$$P(\varepsilon^*_{rc} =' edge'|K) > P(\varepsilon^*_{rc} =' no - edge'|K) \qquad (III-1)$$

and assign the edge state ε^*_{rc} as 'no-edge',otherwise.

The context information appearing in *equation (III-1)* is K which is the cubic facet parameters of the whole image. Thus, we have the most desirable kind of edge labeling process which gives each pixel the highest probability edge state label given the entire context of the image.

To explain the way we organize for the entire context of the image, let's select any pixel in the image. Now consider all the row monotonically increasing paths which begin at any border pixel of the image above the selected pixel, go through the selected pixel, and end at some border pixel of the image below the selected pixel. Each such path represents a context for the pixel. Corresponding to each path and the observed pixel data on the path, there is an associated probability of edge state for the given pixel. Among all the paths there is some best 'edge' path whose associated edge state='edge' probability is higher than the probability of every other path. In general it is not necessary that the edge state of the pixels in an edge path should be 'edge'. In the following derivations we allow pixels which have edge state='no-edge' in a edge path. However, the derivations are general enough so that by a minor modification, we can require all the pixels in an edge path have edge state='edge'. Similarly, for each pixel there is some best 'no-edge' path passing through it whose associated edge state='no-edge' probability is higher than the probability of every other path.

In considering the difference between the edge and no-edge context, we do not use the best 'no-edge' path alone for the non-edge context. For a given pixel (r,c), a row monotonically increasing path has to pass through one of the pixels (r-1,c-1),(r-1,c),(r-1,c+1),and (r,c-1) before entering the pixel (r,c) and it has to go through one of the pixels among (r+1,c-1),(r+1,c),(r+1,c+1), and (r,c+1) when leaving (r,c). Thus, for each entering and leaving pixel pair there exists a best non-edge path. The non-edge context we use for the edge detection is the average no-edge probability of these 16 best paths. Hence, *Equation (III-1)* implies that we will assign a pixel edge state 'edge' if the edge probability of the best 'edge' path is higher than the average no-edge probability of the best 'no-edge' paths. And we will assign a pixel 'no-edge' otherwise.

We begin our description by reviewing some of the definitions defined in Haralick(1985). For a path, we mean any connected sequence of pixels, each pixel neighboring its successor, in which the path does not intersect itself. A row monotonically increasing path is a path in which each successor pixel is on the same row or one row below its predecessor. The set U_{rc} designates the set of all row monotonically increasing paths which begin at some border pixel of the image above or to the left of pixel (r,c) and terminate at pixel (r,c). The set L_{rc} designates the set of all row monotonically increasing paths which begin at (r,c) and terminate at some border pixel below or to the right of pixel (r,c). These are illustrated in *Figure (III-1)*. Let $N_1(r,c) = \{(r-1,c-1),(r-1,c),(r-1,c+1),(r,c-1)\}$. and $N_2(r,c) = \{(r+1,c-1),(r+1,c),(r+1,c+1),(r,c+1)\}$. The set $U_{rc(pq)}$ where $(p,q) \in N_1(r,c)$ is defined as

$$U_{rc(pq)} = \{T : T \in U_{rc} \; and \; (p,q) \in T\}$$

Similarly, we can define

$$L_{rc(ij)} = \{T : T \in U_{rc} \; and \; (i,j) \in T\}$$

where $(i,j) \in N_2(r,c)$.

The set Z_{rc} designates the set of all row monotonically increasing paths begining from a border of the image passing through pixel (r,c) and continuing to another border pixel of the image. The relationship between Z_{rc} and L_{rc} and U_{rc} should be obvious. Z_{rc} is just the join of all paths in U_{rc} with the paths in L_{rc}. Similarly, we can define $Z_{rc(pq,ij)}$ as the join of all paths in $U_{rc(pq)}$ with the paths in $L_{rc(ij)}$.

The set U^*_{rc} designates the set of all row monotonically increasing paths which begin at some border pixel of the image at the same row or above pixel (r,c) and terminate at pixel (r,c). The set L^*_{rc} designates the set of all row monotonically increasing paths which begin at pixel (r,c) and terminate at some border pixel at the same row or below the pixel (r,c). This is illustrated in *Figure (III-2).*

For pixel (i,j), we let $\underline{k}_{ij}$ designate the cubic facet parameters of the pixel (i,j). We define $f_{Z_{rc}}(\varepsilon^*_{rc})$ to be the probability that pixel

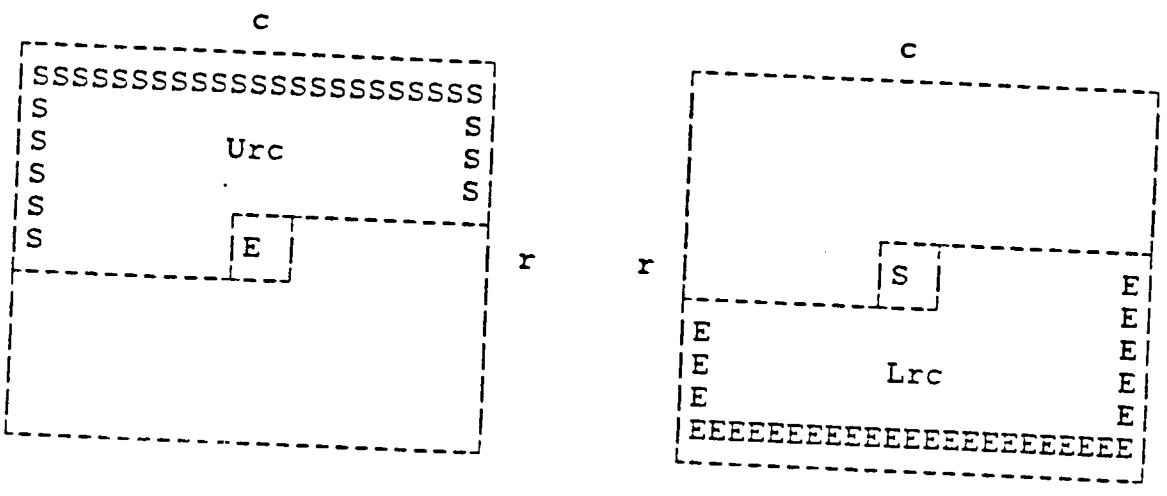


Figure III-1. illustrates the set U_{rc} and the set L_{rc}. U_{rc} is the set of all row and column monotonically increasing paths beginning on a border of the image above or to the left of the pixel (r,c) and terminating at pixel (r,c). L_{rc} is the set of all row and column monotonically increasing paths beginning at the pixel (r,c) and terminating on the border of the image below or to the right of the pixel.

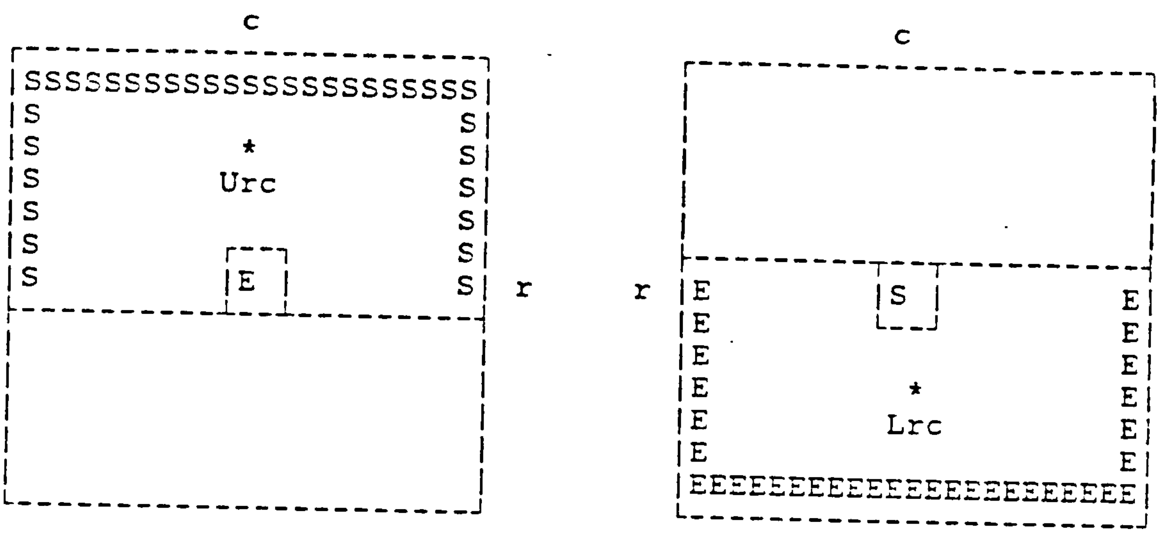


Figure III-2. illustrates the set U^*_{rc} and the set L^*_{rc}. U^*_{rc} is the set of all row monotonically increasing paths beginning on a border of the image at the same row or above pixel (r,c) and terminating at pixel (r,c). L^*_{rc} is the set of all row and column monotonically increasing paths beginning at the pixel (r,c) and terminating on the border of the image below or to the right of the pixel.

(r,c) has true edge state ε^*_{rc} given the cubic facet parameters of the best row monotonically increasing path T taking the direction θ^*_{rc} through (r,c) and the direction θ^*_{rc} is the direction which maximizes $P(\theta^*_{rc}|\underline{k}_{rc})$. Thus,

$$f_{Z_{rc}}(\varepsilon^*_{rc}) = q_{Z_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc0})$$

where θ^*_{rc0} is a fixed value which is determined by

$$P(\theta^*_{rc0}|\underline{k}_{rc}) > P(\theta_{rc}|\underline{k}_{rc}) \; \forall \; \theta_{rc} \neq \theta^*_{rc0} \qquad (III-2)$$

and

$$q_{Z_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max_{T \in Z_{rc}} P(\varepsilon^*_{rc}, \theta^*_{rc}|\underline{k}_{ij} : (i,j) \in T) \qquad (III-3)$$

Similarly, we can define $f_{Z_{rc(pq,ij)}}(\varepsilon^*_{rc})$. We also define

$$\overline{f}_{Z_{rc}}(\varepsilon^*_{rc}) = \frac{1}{16} \sum_{\substack{(pq)\in N_1(r,c) \\ (ij)\in N_2(r,c)}} f_{Z_{rc(pq,ij)}}(\varepsilon^*_{rc})$$

Therefore, the Baysian decision theory based edge detection scheme becomes

$$\varepsilon^*_{rc} = \begin{array}{ll} edge & if \; f_{Z_{rc}}(\varepsilon^*_{rc} =' edge') > \overline{f}_{Z_{rc}}(\varepsilon^*_{rc} =' no-edge') \\ no-edge & \qquad otherwise \end{array} \qquad (II-4)$$

To perform edge detection we have to compute $f_{Z_{rc}}$. Thus, we have to compute $q_{Z_{rc}}$ first.

Analogous to the definition of $q_{Z_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$ we can now define $g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$ and $g_{L_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$ as follows:

$$g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max_{T \in U_{rc}} P(\varepsilon^*_{rc}, \theta^*_{rc} | \underline{k}_{ij} : (i,j) \in T) \qquad (III-5)$$

and

$$g_{L_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max_{T \in L_{rc}} P(\varepsilon^*_{rc}, \theta^*_{rc} | \underline{k}_{ij} : (i,j) \in T) \qquad (III-6)$$

In a similar way, $g_{U_{rc(pq)}}(\varepsilon^*_{rc}, \theta^*_{rc})$ and $g_{L_{rc(ij)}}(\varepsilon^*_{rc}, \theta^*_{rc})$ can be defined. Since

$$q_{Z_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max_{T \in Z_{rc}} P(\varepsilon^*_{rc}, \theta^*_{rc} | \underline{k}_{ij} : (i,j) \in T)$$

$$= \max_{T \in Z_{rc}} \sum_{\substack{\theta^*_{rc}, \varepsilon^*_{rc} \\ (i,j) \in T^-}} P(\varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T | \underline{k}_{ij} : (i,j) \in T)$$

$$= \max_{T \in Z_{rc}} \sum_{\substack{\theta^*_{rc}, \varepsilon^*_{rc} \\ (i,j) \in T^-}} P(\underline{k}_{ij} : (i,j) \in T | \varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T)$$

$$* \frac{P(\varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T)}{P(\underline{k}_{ij} : (i,j) \in T)} \qquad (III-7)$$

where T^- designates the set of all pixels in T but pixel (r,c).

We assume that when the pixel (r,c) is being examined, no observed characteristics from any other pixel but pixel (r,c) affects the observed data of position (r,c). Hence

$$P(\underline{k}_{ij} : (i,j) \in T | \varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T)$$

$$= \prod_{(i,j)\in T} P(\underline{k}_{ij}|\varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T) \qquad (III-8)$$

and

$$P(\underline{k}_{ij} : (i.j) \in T) = \prod_{(i,j)\in T} P(\underline{k}_{ij}) \qquad (III-9)$$

The second assumption states that the observed facet measurements $\underline{k}_{ij}$ at each pixel (i,j) depends only upon the true facet parameters and edge data associated with pixel (i,j) and does not depend upon any true edge data of the other pixels. Hence

$$P(\underline{k}_{ij}|\underline{k}^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl} : (k,l) \in T)$$

$$= P(\underline{k}_{ij}|\underline{k}^*_{ij}, \varepsilon^*_{ij}, \theta^*_{ij}) \qquad (III-10)$$

Following the same derivations as illustrated in *Appendix 3* we have

$$P(\underline{k}_{ij} : (i,j) \in T|\varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T)$$

$$= \prod_{(i,j)\in T} P(\underline{k}_{ij}|\varepsilon^*_{ij}, \theta^*_{ij}) \qquad (III-11)$$

The probability $P(\varepsilon^*_{ij}, \theta^*_{ij}, (i,j) \in T)$ is the joint prior probability of having the edge data for each pixel (i,j) on the path T be $\varepsilon^*_{ij}, \theta^*_{ij}$. This probability encodes all the information we have about context. The simplest assumption of higher order than independence is a Markov like assumption in which the joint prior probability becomes a function expressible as the product of functions whose arguments are the label pairs for successive pixels in the path T. This is a second order generalized conditional independence assumption. Letting $R(T)$ designate the set of all pairs of successive pixels in the path T, we have

$$P(\varepsilon^*_{ij}.\theta^*_{ij} : (i,j) \in T) =$$

$$\prod_{((i,j),(k,l))\in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) \qquad (III-12)$$

Using assumptions (III-9),(III-11),and (III-12) we are ready to decompose (III-7) into

$$q_{Z_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max_{T\in Z_{rc}} \sum_{\substack{\theta^*_{ij},\varepsilon^*_{ij} \\ (i,j)\in T^-}} \prod_{(i,j)\in T} \frac{P(\underline{k}_{ij}|\varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l))\in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) \qquad (III-13)$$

Since Z_{rc} can be decomposed as the join of U_{rc} and L_{rc}, this results in

$$q_{Z_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max_{T_1\in U_{rc}} \max_{T_2\in L_{rc}} \sum_{\substack{\theta^*_{ij},\varepsilon^*_{ij} \\ (i,j)\in T_1^-}} \sum_{\substack{\theta^*_{ij},\varepsilon^*_{ij} \\ (i,j)\in T_2^-}}$$

$$\prod_{(i,j)\in T_1} \frac{P(\underline{k}_{ij}|\varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})} \prod_{(i,j)\in T_2} \frac{P(\underline{k}_{ij}|\varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})} \frac{P(\underline{k}_{rc})}{P(\underline{k}_{rc}|\varepsilon^*_{rc}, \theta^*_{rc})}$$

$$* \prod_{((i,j),(k,l))\in R(T_1)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl})$$

$$* \prod_{((i,j),(k,l))\in R(T_2)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) \qquad (III-14)$$

Rearranging (III-14) we can group all expressions involving T_1 together and all expressions involving T_2 together and we obtain

$$q_{Z_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) =$$

$$\frac{P(\underline{k}_{rc})}{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})} g_{U_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) * g_{L_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) \qquad (III-15)$$

Similarly, we have

$$q_{Z_{rc(pq,ij)}}(\varepsilon^*_{rc},\theta^*_{rc}) =$$

$$\frac{P(\underline{k}_{rc})}{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})} g_{U_{rc(pq)}}(\varepsilon^*_{rc},\theta^*_{rc}) * g_{L_{rc(ij)}}(\varepsilon^*_{rc},\theta^*_{rc})$$

The decomposition of $g_{U_{rc}}$ will be in terms of the neighboring $g_{U_{rc-1}}$ and $h_{U^*_{r-1,c-1}}, h_{U^*_{r-1,c}}$ *and* $h_{U^*_{r-1,c+1}}$, all of which need definition. By definition

$$g_{U_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) = \max_{T\in U_{rc}} P(\varepsilon^*_{rc},\theta^*_{rc}|\underline{k}_{ij}:(i,j)\in T)$$

$$= \max_{T\in U_{rc}} \sum_{\substack{\theta^*_{rc},\varepsilon^*_{rc}\\(i,j)\in T^-}} P(\varepsilon^*_{ij},\theta^*_{ij},(i,j)\in T|\underline{k}_{ij}:(i,j)\in T)$$

$$= \max_{T\in U_{rc}} \sum_{\substack{\theta^*_{ij},\varepsilon^*_{ij}\\(i,j)\in T^-}} \prod_{(i,j)\in T} \frac{P(\underline{k}_{ij}|\varepsilon^*_{ij},\theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l))\in R(T)} A(\varepsilon^*_{ij},\theta^*_{ij},\varepsilon^*_{kl},\theta^*_{kl}) \qquad (III-16)$$

Also, we define

$$h_{U^*_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) = \max_{T\in U^*_{rc}} P(\varepsilon^*_{rc},\theta^*_{rc}|\underline{k}_{ij}:(i,j)\in T)$$

$$= \max_{T \in U^*_{rc}} \sum_{\substack{\theta^*_{ij}, \varepsilon^*_{ij} \\ (i,j) \in T^-}} \prod_{(i,j) \in T} \frac{P(\underline{k}_{ij} | \varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l)) \in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) \qquad (III-17)$$

To do the decomposition for $g_{U_{rc}}$ we need to recognize that whatever the best path is, the best path to (r,c) must have come from one of the pixel locations: (r,c-1), (r-1,c-1),(r-1,c), or (r-1,c+1). Because the best paths cannot cross itself, if the best path came from (r,c-1), then the path must be in $U_{r,c-1}$. However, there is no danger of the path crossing itself if the best path comes from (r-1,c-1). Hence, such a path must be in $U^*_{r-1,c-1}$. Likewise, a best path coming from (r-1,c) must be in $U^*_{r-1,c}$ and a best path coming from (r-1,c+1) must be in $U^*_{r-1,c+1}$. Using this idea, we can write (III-16) as

$$g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \frac{P(\underline{k}_{rc} | \varepsilon^*_{rc}, \theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\max\{ \max_{T \in U_{r,c-1}} \sum_{\substack{\theta^*_{ij}, \varepsilon^*_{ij} \\ (i,j) \in T}} \prod_{(i,j) \in T} \frac{P(\underline{k}_{ij} | \varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l)) \in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) * A(\varepsilon^*_{r,c-1}, \theta^*_{r,c-1}, \varepsilon^*_{rc}, \theta^*_{rc}),$$

$$\max_{T \in U^*_{r-1,c-1}} \sum_{\substack{\theta^*_{ij}, \varepsilon^*_{ij} \\ (i,j) \in T}} \prod_{(i,j) \in T} \frac{P(\underline{k}_{ij} | \varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l)) \in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) * A(\varepsilon^*_{r-1,c-1}, \theta^*_{r-1,c-1}, \varepsilon^*_{rc}, \theta^*_{rc}),$$

$$\max_{T \in U^*_{r-1,c}} \sum_{\substack{\theta^*_{ij}, \varepsilon^*_{ij} \\ (i,j) \in T}} \prod_{(i,j) \in T} \frac{P(\underline{k}_{ij} | \varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l)) \in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl}) * A(\varepsilon^*_{r-1,c}, \theta^*_{r-1,c}, \varepsilon^*_{rc}, \theta^*_{rc}),$$

$$\max_{T \in U^*_{r-1,c+1}} \sum_{\substack{\theta^*_{ij}, \varepsilon^*_{ij} \\ (i,j) \in T}} \prod_{(i,j) \in T} \frac{P(\underline{k}_{ij} | \varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l)) \in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl})$$

$$*A(\varepsilon^*_{r-1,c+1}, \theta^*_{r-1,c+1}, \varepsilon^*_{rc}, \theta^*_{rc})\} \qquad (III-18)$$

Examining the first term in the maximization and comparing it to $g_{U_{r,c-1}}(\varepsilon^*_{r,c-1}, \theta^*_{r,c-1})$ as defined by (III-16) we discover that the expressions are almost identical. The only difference is that the expression for $g_{U_{r,c-1}}(\varepsilon^*_{r,c-1}, \theta^*_{r,c-1})$ involves a summation iterated over all $(i,j) \in T^-$ while the first expression in the maximization involves a summation iterated over all $(i,j) \in T$. Since the maximization done in the first term is over all paths terminating in (r,c-1), the summation over $\varepsilon^*_{r,c-1}$ and $\theta^*_{r,c-1}$ can be interchanged with the maximization over all $T \in U_{r,c-1}$. A similar reorganization can be done with the second, third, and fourth terms of the summation after comparison of them with $h_{U^*_{r-1,c-1}}, h_{U^*_{r-1,c}}, h_{U^*_{r-1,c+1}}$. This results in

$$g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \frac{P(\underline{k}_{rc} | \varepsilon^*_{rc}, \theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\max\{\sum_{\theta^*_{r,c-1},\varepsilon^*_{r,c-1}} g_{U_{r,c-1}}(\varepsilon^*_{r,c-1},\theta^*_{r,c-1}) * A(\varepsilon^*_{r,c-1},\theta^*_{r,c-1},\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\sum_{\theta^*_{r-1,c-1},\varepsilon^*_{r-1,c-1}} h_{U^*_{r-1,c-1}}(\varepsilon^*_{r-1,c-1},\theta^*_{r-1,c-1})$$

$$*A(\varepsilon^*_{r-1,c-1},\theta^*_{r-1,c-1},\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\sum_{\theta^*_{r-1,c},\varepsilon^*_{r-1,c}} h_{U^*_{r-1,c}}(\varepsilon^*_{r-1,c},\theta^*_{r-1,c}) * A(\varepsilon^*_{r-1,c},\theta^*_{r-1,c},\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\sum_{\theta^*_{r-1,c+1},\varepsilon^*_{r-1,c+1}} h_{U^*_{r-1,c+1}}(\varepsilon^*_{r-1,c+1},\theta^*_{r-1,c+1})$$

$$*A(\varepsilon^*_{r-1,c+1},\theta^*_{r-1,c+1},\varepsilon^*_{rc},\theta^*_{rc})\} \qquad (III-19)$$

Equation (III-19) says that for each edge label $\varepsilon^*_{rc},\theta^*_{rc}$, the conditional probability of $\varepsilon^*_{rc},\theta^*_{rc}$ given $\{\underline{k}_{ij} : (i,j) \in T\}$ where T is the path giving the highest probability can be obtained on the basis of the previously computed $g_{U_{r,c-1}}$, and on the previously computed $h_{U^*_{r-1,c-1}}, h_{U^*_{r-1,c}}, h_{U^*_{r-1,c+1}}$ coming from the row above the current row. So providing we can demonstrate a way to compute $h_{U^*_{rc}}$, *equation (III-19)* specifies a recursive neighborhood operator which scans the image in a top down left right scan to produce for each pixel (r,c) and for each edge label $\varepsilon^*_{rc},\theta^*_{rc}$ the probability $g_{U_{rc}}(\varepsilon^*_{rc},\theta^*_{rc})$.

It is easy to find that we also have

$$g_{U_{rc(r,c-1)}}(\varepsilon^*_{rc},\theta^*_{rc}) = \frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\sum_{\theta^*_{r,c-1},\varepsilon^*_{r,c-1}} g_{U_{r,c-1}}(\varepsilon^*_{r,c-1},\theta^*_{r,c-1}) * A(\varepsilon^*_{r,c-1},\theta^*_{r,c-1},\varepsilon^*_{rc},\theta^*_{rc})$$

and

$$g_{U_{rc(ij)}}(\varepsilon^*_{rc},\theta^*_{rc}) = \frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\sum_{\theta^*_{i,j},\varepsilon^*_{i,j}} h_{U^*_{i,j}}(\varepsilon^*_{i,j},\theta^*_{i,j}) * A(\varepsilon^*_{i,j},\theta^*_{i,j},\varepsilon^*_{rc},\theta^*_{rc})$$

where $(i,j) \in N_1(r,c) - (r,c-1)$.

The algorithm for computing $h_{U^*_{rc}}$ is similar to that of $g_{U_{rc}}$. For a path from U^*_{rc} to reach (r,c) it must first have gone through one of the pixel location: (r,c-1),(r,c+1),(r-1,c-1),(r-1,c), or (r-1,c+1). Furthermore, if it went through (r,c-1), since the path cannot cross itself, it must be a path in $U_{r,c-1}$. From our development of *equation (III-19)* , we know that the maximization over the probability of the paths go through (r,c-1),(r-1,c-1),(r-1,c), and (r-1,c+1) yields $g_{U_{rc}}(\varepsilon^*_{rc},\theta^*_{rc})$. Thus we have

$$h_{U^*_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) = \max\{g_{U_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} * \max_{T\in U^*_{r,c+1}} \sum_{\substack{\theta^*_{ij},\varepsilon^*_{ij}\\(i,j)\in T}} \prod_{(i,j)\in T} \frac{P(\underline{k}_{ij}|\varepsilon^*_{ij},\theta^*_{ij})}{P(\underline{k}_{ij})}$$

$$* \prod_{((i,j),(k,l))\in R(T)} A(\varepsilon^*_{ij},\theta^*_{ij},\varepsilon^*_{kl},\theta^*_{kl})$$

$$*A(\varepsilon^*_{r,c+1},\theta^*_{r,c+1},\varepsilon^*_{rc},\theta^*_{rc})\} \qquad (III-20)$$

In a similar manner as in the development of (III-19), we interchange the order of the summation over $\varepsilon^*_{r,c+1},\theta^*_{r,c+1}$ with the maximization over all path T in $U^*_{r,c+1}$. Now (III-20) becomes

$$h_{U^*_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max\{g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}),$$

$$\frac{P(\underline{k}_{rc}|\varepsilon^*_{rc}, \theta^*_{rc})}{P(\underline{k}_{rc})} * \sum_{\theta^*_{r,c+1}, \varepsilon^*_{r,c+1}} \max_{T \in U^*_{r,c+1}} \sum_{\substack{\theta^*_{ij}, \varepsilon^*_{ij} \\ (i,j) \in T^-}} \prod_{(i,j) \in T}$$

$$\frac{P(\underline{k}_{ij}|\varepsilon^*_{ij}, \theta^*_{ij})}{P(\underline{k}_{ij})} * \prod_{((i,j),(k,l)) \in R(T)} A(\varepsilon^*_{ij}, \theta^*_{ij}, \varepsilon^*_{kl}, \theta^*_{kl})$$

$$*A(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1}, \varepsilon^*_{rc}, \theta^*_{rc})\} \qquad (III-21)$$

By definition of (III-17), the bracketed expression of (III-21) is precisely $h_{U^*_{r,c+1}}(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1})$. Now this results in

$$h_{U^*_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = \max\{g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}),$$

$$\frac{P(\underline{k}_{rc}|\varepsilon^*_{rc}, \theta^*_{rc})}{P(\underline{k}_{rc})} * \sum_{\theta^*_{r,c+1}, \varepsilon^*_{r,c+1}} h_{U^*_{r,c+1}}(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1})$$

$$*A(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1}, \varepsilon^*_{rc}, \theta^*_{rc})\} \qquad (III-22)$$

Equation (III-22) states that $h_{U^*_{rc}}$ can be recursively computed from $g_{U^*_{rc}}$ and the previous $h_{U^*_{r,c+1}}$ in a right left scan of a row done after $g_{U^*_{rc}}$ has been computed. To start the recursive calculation (III-22), we take $h_{U^*_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}) = g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$ for that column position c which is the rightmost position.

In summary, *equation (III-19)* and *(III-22)* gives the following algorithm for the computation of $g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$. From (III-19), we perform a top down left right scan of the image recursively computing $g_{U_{rc}}$ from the previous $g_{U_{r,c-1}}$ and the $h_{U^*_{r-1,c-1}}$, $h_{U^*_{r-1,c}}$, and $h_{U^*_{r-1,c+1}}$ which had been computed on the previous row. Following the computation of $g_{U_{rc}}$ for all pixels on row r, we perform a right left scan of row r using *equation (III-22)* to compute $h_{U^*_{rc}}$.

An absolutely mirror image derivation applies to $g_{L_{rc}}$. Thus, we have

$$g_{L_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) = \frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\max\{ \sum_{\theta^*_{r,c+1},\varepsilon^*_{r,c+1}} g_{L_{r,c+1}}(\varepsilon^*_{r,c+1},\theta^*_{r,c+1}) * A(\varepsilon^*_{r,c+1},\theta^*_{r,c+1},\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\sum_{\theta^*_{r+1,c+1},\varepsilon^*_{r+1,c+1}} h_{L^*_{r+1,c+1}}(\varepsilon^*_{r+1,c+1},\theta^*_{r+1,c+1})$$

$$*A(\varepsilon^*_{r+1,c+1},\theta^*_{r+1,c+1},\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\sum_{\theta^*_{r+1,c},\varepsilon^*_{r+1,c}} h_{L^*_{r+1,c}}(\varepsilon^*_{r+1,c},\theta^*_{r+1,c}) * A(\varepsilon^*_{r+1,c},\theta^*_{r+1,c},\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\sum_{\theta^*_{r+1,c-1},\varepsilon^*_{r+1,c-1}} h_{L^*_{r+1,c-1}}(\varepsilon^*_{r+1,c-1},\theta^*_{r+1,c-1})$$

$$*A(\varepsilon^*_{r+1,c-1},\theta^*_{r+1,c-1},\varepsilon^*_{rc},\theta^*_{rc})\} \qquad (III-23)$$

and

$$h_{L^*_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}) = \max\{g_{L_{rc}}(\varepsilon^*_{rc},\theta^*_{rc}),$$

$$\frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} * \sum_{\theta^*_{r,c-1},\varepsilon^*_{r,c-1}} h_{L^*_{r,c-1}}(\varepsilon^*_{r,c-1},\theta^*_{r,c-1})$$

$$*A(\varepsilon^*_{r,c-1},\theta^*_{r,c-1},\varepsilon^*_{rc},\theta^*_{rc})\} \qquad (III-24)$$

It is also easily to find that we have

$$g_{L_{rc(r,c+1)}}(\varepsilon^*_{rc},\theta^*_{rc}) = \frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\sum_{\theta^*_{r,c+1},\varepsilon^*_{r,c+1}} g_{L_{r,c+1}}(\varepsilon^*_{r,c+1},\theta^*_{r,c+1}) * A(\varepsilon^*_{r,c+1},\theta^*_{r,c+1},\varepsilon^*_{rc},\theta^*_{rc})$$

and

$$g_{L_{rc(ij)}}(\varepsilon^*_{rc},\theta^*_{rc}) = \frac{P(\underline{k}_{rc}|\varepsilon^*_{rc},\theta^*_{rc})}{P(\underline{k}_{rc})} *$$

$$\sum_{\theta^*_{i,j},\varepsilon^*_{i,j}} h_{L^*_{i,j}}(\varepsilon^*_{i,j},\theta^*_{i,j}) * A(\varepsilon^*_{i,j},\theta^*_{i,j},\varepsilon^*_{rc},\theta^*_{rc})$$

where $(i,j) \in N_2(r,c) - (r,c+1)$.

To compute them, we perform a bottom up right left scan of the image recursively computing $g_{L_{rc}}$ from the previous $g_{L_{r,c+1}}$ and the $h_{L^*_{r+1,c-1}}$, $h_{L^*_{r+1,c}}$, and $h_{L^*_{r+1,c+1}}$ which had been computed on the previous row from the bottom up scan. Following the computation of $g_{L_{rc}}$ for all pixels on row r, we perform a left right scan of row r and compute $h_{L^*_{rc}}$.

As soon as $g_{L_{rc}}(\varepsilon^*_{rc},\theta^*_{rc})$ has been computed, it can be combined with $g_{U_{rc}}(\varepsilon^*_{rc},\theta^*_{rc})$ as given in *equation (III-15)* to compute $q_{Z_{rc}}(\varepsilon^*_{rc},\theta^*_{rc})$. In practice since the only θ^*_{rc} which is useful for the purpose of edge detection is the angle θ^*_{0rc} which maximizes $P(\theta^*_{rc}|\underline{k}_{rc})$. Thus, we can determine θ^*_{0rc} first and then compute $q_{rc}(\varepsilon^*_{rc},\theta^*_{0rc})$ for both θ^*_{rc}="edge" and "no-edge". Now by the definition of (III-2), we can compute the two probability terms by

$$f_{Z_{rc}}(\varepsilon^*_{rc}) = q_{rc}(\varepsilon^*_{rc}, \theta^*_{0rc})$$

and

$$\overline{f}_{Z_{rc}}(\varepsilon^*_{rc}) = \frac{1}{16} \sum_{\substack{(pq)\in N_1(r,c) \\ (ij)\in N_2(r,c)}} f_{Z_{rc(pq,ij)}}(\varepsilon^*_{rc}).$$

Thus, we can label edge state of pixel (r,c) by means of the rule provided by *equation (III-4)*.

III-2. Probabilities

To perform the recursive algorithms (III-19),(III-22),(III-23),(III-24), we need the probability ratio $\frac{P(\underline{k}_0|\varepsilon^*_0,\theta^*_0)}{P(\underline{k}_0)}$ and the edge consistent function $A(\varepsilon^*_0, \theta^*_0, \varepsilon^*_1, \theta^*_1)$ where 0 and 1 designate any two adjacent pixel positions in the image.

In *Appendix 2* we have derived the conditional probability as

$$P(k_2, \ldots, k_{10}|\varepsilon^*, \theta^*) = P(k^*_2, k^*_3|\varepsilon^*, \theta^*)P(A, B|\varepsilon^*, \theta^*)$$

$$*P(k_4, \ldots, k_{10}|A, B) \qquad (III-25)$$

Let λ be the ratio of standard deviation of k_2 and k_3 values between edge and non-edge pixels. Then $P(k_2, k_3|\varepsilon^* =' edge', \theta^*)$ is a normal distribution

$$N\left(H\begin{pmatrix} 0 \\ \sqrt{\frac{\pi}{2}}\lambda\sigma^*_{g1}\end{pmatrix},\ H\begin{pmatrix} \sigma^2 q & 0 \\ 0 & 2\lambda^2\sigma^{*2}_{g1} + \sigma^2_q\end{pmatrix}H'\right) \qquad (III-26)$$

and $P(k_2, k_3|\varepsilon^* =' no - edge', \theta^*)$ is a normal distribution

$$N\left(H\begin{pmatrix}0\\0\end{pmatrix}, H\begin{pmatrix}\sigma^2 q & 0\\0 & 2\sigma_{g1}^{*2}+\sigma_q^2\end{pmatrix}H'\right) \qquad (III-27)$$

where

$$H = \begin{pmatrix}\cos\theta^* & \sin\theta^*\\ -\sin\theta^* & \cos\theta^*\end{pmatrix} \qquad (III-28)$$

The conditional probability $P(A, B|\varepsilon^* =' edge', \theta^*)$ is

$$N\left(\begin{pmatrix}\mu_A^*\\0\end{pmatrix}, \; T\begin{pmatrix}0 & 0\\0 & \sigma_s^{2*}\end{pmatrix}T' + \begin{pmatrix}\sigma_{\eta A}^2 & 0\\0 & \sigma_{\eta B}^2\end{pmatrix}\right) \qquad (III-29)$$

and the conditional probability $P(A, B|\varepsilon^* =' no - edge', \theta^*)$ is

$$N\left(\begin{pmatrix}0\\0\end{pmatrix}, \; T\begin{pmatrix}\sigma_r^{2*} & 0\\0 & \sigma_s^{2*}\end{pmatrix}T' + \begin{pmatrix}\sigma_{\eta A}^2 & 0\\0 & \sigma_{\eta B}^2\end{pmatrix}\right) \qquad (III-30)$$

where all the notations $T, A, B, \sigma^2, \ldots$,are defined in *Appendix2.*

Since

$$P(k_2, \ldots, k_{10}) =$$

$$\int_{\theta^*} P(k_2^*, k_3^*|\varepsilon^* =' no - edge', \theta^*)P(A, B|\varepsilon^*, \theta^*)$$

$$*P(k_4, \ldots, k_{10}|A, B) * \frac{(1 - P(edge))}{2\pi}d\theta^*$$

$$+ \int_{\theta^*} P(k_2^*, k_3^*|\varepsilon^* =' edge',, \theta^*)P(A, B|\varepsilon^*, \theta^*)$$

$$*P(k_4, \ldots, k_{10}|A, B) * \frac{P(edge)}{2\pi}d\theta^* \qquad (III-31)$$

From (III-19), (III-21), (III-22), (III-23), and (III-24) we know that the term we used for the local edge probability is $\frac{P(\underline{k}_0|\varepsilon_0^*, \theta_0^*)}{P(\underline{k}_0)}$. And, when we compute this local probability, the probability $P(k_4, \ldots, k_{10}|A, B)$ is cancelled out by dividing (III-31) by (III-25) and need not be computed. The only probability need to be given is the prior probability of edge state is 'edge'. This is, the probability $P(\varepsilon^* =' edge')$ which we denote by $P(edge)$.

Let "0" be the center pixel and "1" is an adjacent pixel of "0". The edge consistent function $A(\varepsilon_0^*, \theta_0^*, \varepsilon_1^*, \theta_1^*)$ is defined in three different cases:

(1). ε_0^*='edge' and ε_1^*='edge':

We follow the function defined in chapter II. Due to the low cumulative curvature requirement of an edge line in a small neighborhood, it is reasonable to assume that $A(\varepsilon_0^*, \theta_0^*, \varepsilon_1^*, \theta_1^*)$ has maximal value when the edge direction at the immediate adjacent neighbor agrees with that at the center. And the expected neighbor direction based on this relative position agrees with the neighbor direction. To satisfy these requirements, we define

$$A(\varepsilon_0^* =' edge', \theta_0^*, \varepsilon_1^* =' edge', \theta_1^*) = \frac{d(\theta_0^*, \theta_1^*) + d_1(\theta_0^*, \frac{M\pi}{4})}{2} \qquad (III-32)$$

where d and d_1 are defined as

$$d(\alpha, \beta) = \frac{\cos(\alpha - \beta) + 1}{2}$$

$$d_1(\alpha, \beta) = \frac{\cos 2(\alpha - \beta) + 1}{2} \qquad (III-33)$$

and M is the adjacent edge position index with respect to the center pixel according to the following order

$$\begin{matrix} 5 & 4 & 3 \\ 6 & 0 & 2 \\ 7 & 8 & 1 \end{matrix}$$

The center position is the position of pixel ε_1^*. Different positions of pixel ε_0^* correspond to different M values.

The range of function d and d_1 is the closed interval [0, 1]. The reason to select this nonlinear function for edge consistent function is that as α approaches β the function has less penality (higher value) than the absolute difference function which computes absolute difference between two angles. Conversely, it gives more penality when the angle difference is large. Thus the angle quantization effect due to the rectangular grid layout of the pixels will tend to be minimized.

(2). ε_0^*='edge' and ε_1^*='no-edge':

The edge direction consistency constraint on the 'edge' to 'no-edge' case is in general much weaker than the constraint on the 'edge' to 'edge' case. The best way to obtain the consistent function for a specific set of images is to perform a measurement from a training set of image data. And fit the measurement data to an appropriate function. However, for the purpose of completeness, we suggest a general function for this case. We define

$$A(\varepsilon_0^* =' edge', \theta_0^*, \varepsilon_1^* =' no - edge', \theta_1^*) =$$

$$\frac{d(\theta_0^*, \theta_1^*) + d_1(\theta_0^*, \frac{N\pi}{4})}{2} \qquad (III - 34)$$

where N is the adjacent edge position index with respect to the center pixel according to the following order

$$\begin{matrix} 3 & 2 & 1 \\ 4 & 0 & 8 \\ 5 & 6 & 7 \end{matrix}$$

The center position is the position of pixel ε_1^*. Different positions of pixel ε_0^* correspond to different N values.

(3). ε_0^*='no-edge' and ε_1^*='no-edge':

For the purpose of completeness, we define

$$A(\varepsilon_0^* =' no - edge', \theta_0^*, \varepsilon_1^* =' no - edge', \theta_1^*) =$$

$$\max\left\{(1 - A(\varepsilon_0^* =' edge', \theta_0^*, \varepsilon_1^* =' edge', \theta_1^*)), \frac{1}{2\pi}\right\} \qquad (III - 35)$$

so that if the edge direction is consistent we will have a low value and vice versa. However, it is possible for a no-edge pixel pair to also have edge direction consistency in some cases. Therefore, we set a lower bound for the function; the lower bound is set to $\frac{1}{2\pi}$.

III-3. Implementation Considerations

In this section we describe the way of using gereral context most efficiently for edge detection. To implement the recursive algorithm for $g_{U_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$ we append one more image column to the left of the image (we name it column 0) and initialize both $g_{U_{r0}}(\varepsilon^*_{r0} =' edge', \theta^*_{r0})$ and $g_{U_{r0}}(\varepsilon^*_{r0} =' no - edge', \theta^*_{r0})$ to 1. Similarly, we append one more row on the top of the image (r=0) and initialize both $g_{U_{0c}}(\varepsilon^*_{0c} =' edge', \theta^*_{0c})$ and $g_{U_{0c}}(\varepsilon^*_{0c} =' no - edge', \theta^*_{0c})$ to 1. We also define the consistent function A in a way that whenever one of its arguments includes the appended boundary pixels it always has value 1. Thus, starting from the pixel position (1,1) we have

$$g_{U_{11}}(\varepsilon^*_{11}, \theta^*_{11}) = \frac{P(\underline{k}_{11}|\varepsilon^*_{11}, \theta^*_{11})}{P(\underline{k}_{11})} \qquad (III-36)$$

and

$$\begin{aligned} g_{U_{12}}(\varepsilon^*_{12}, \theta^*_{12}) &= \frac{P(\underline{k}_{12}|\varepsilon^*_{12}, \theta^*_{12})}{P(\underline{k}_{12})} \\ &\max\{\sum_{\varepsilon^*_{11}, \theta^*_{11}} \frac{P(\underline{k}_{11}|\varepsilon^*_{11}, \theta^*_{11})}{P(\underline{k}_{11})} \\ &*A(\varepsilon^*_{11}, \theta^*_{11}, \varepsilon^*_{12}, \theta^*_{12}), 1\} \\ &\vdots \quad \ddots \quad \vdots \end{aligned} \qquad (III-37)$$

We now interpret the meaning of the probability ratio $\frac{P(\underline{k}|\varepsilon^*,\theta^*)}{P(\underline{k})}$. There are two ratios one for edge state ='edge', one for edge state ='no-edge'. Since

$$P(\underline{k}) = \int_{\theta^*} P(\underline{k}|\varepsilon^* =' edge', \theta^*)P(\theta^*|\varepsilon^* =' edge')P(edge)d\theta^*$$

$$+\int_{\theta^*} P(\underline{k}|\varepsilon^* = no -' edge', \theta^*)P(\theta^*|\varepsilon^* =' no - edge')P(no - edge)d\theta^*$$

$$= \frac{P(edge)}{2\pi}\int_{\theta^*} P(\underline{k}|\varepsilon^* =' edge', \theta^*)d\theta^*$$

$$+\frac{(1 - P(edge))}{2\pi}\int_{\theta^*} P(\underline{k}|\varepsilon^* =' no - edge', \theta^*)d\theta^* \qquad (III - 38)$$

Let θ_0^* be the θ^* which maximizes $P(\theta^*\varepsilon^*|\underline{k})$. It is the observed gradient direction of a pixel. By definition the probability $P(\theta^*\varepsilon^*|\underline{k})$ for all $\theta^* \neq \theta_0^*$ is less than the probability for θ_0^*. To simplify the computation, we scale $P(\underline{k}|\varepsilon^* =' edge', \theta^*)$ and $P(\underline{k}|\varepsilon^* =' no - edge', \theta^*)$ such that when summed over all possible edge directions the integral is 2π times the probability when $\theta^* = \theta_0^*$. Thus,

$$\int_{\theta^*} P(\underline{k}|\varepsilon^* =' edge', \theta^*)d\theta^* = 2\pi P(\underline{k}|\varepsilon^* =' edge', \theta_0^*)$$

and

$$\int_{\theta^*} P(\underline{k}|\varepsilon^* =' no - edge', \theta^*)d\theta^* = 2\pi P(\underline{k}|\varepsilon^* =' no - edge', \theta_0^*)$$

Hence

$$p(\underline{k}) = P(edge)P(\underline{k}|\varepsilon^* =' edge', \theta_0^*)$$

$$+(1 - P(edge))P(\underline{k}|\varepsilon^* =' no - edge', \theta^*) \qquad (III-39)$$

In the case of $\varepsilon^* =' edge'$, the local edge probability becomes

$$\frac{P(\underline{k}|\varepsilon^* =' edge', \theta_0^*)}{P(\underline{k})} =$$

$$= \frac{P(\underline{k}|\varepsilon^* =' edge', \theta_0^*)}{P(E)P(\underline{k}|\varepsilon^* =' E', \theta_0^*) + (1 - P(E))P(\underline{k}|\varepsilon^* =' N', \theta^*)} \qquad (III-40)$$

Thus,

$$\frac{P(\underline{k}|\varepsilon^* =' edge', \theta_0^*)}{P(\underline{k})}$$

$$\begin{aligned} &= 1 \;\; if \; P(\underline{k}|\varepsilon^* =' E', \theta_0^*) = P(\underline{k}|\varepsilon^* =' N', \theta_0^*) \\ &> 1 \;\; if \; P(\underline{k}|\varepsilon^* =' E', \theta_0^*) > P(\underline{k}|\varepsilon^* =' N', \theta_0^*) \\ &< 1 \;\; if \; P(\underline{k}|\varepsilon^* =' E', \theta_0^*) < P(\underline{k}|\varepsilon^* =' N', \theta_0^*) \end{aligned}$$

This can explain the meaning of this local probability term which is used in determining the optimal edge or no-edge paths. If the observed cubic facet parameters of a pixel support the edge state of the pixel ='edge' over the edge state of the pixel ='no-edge', then this local probability will be greater than one. Otherwise, it will be less than one. If the observed cubic facet data do not support any edge state, the probability will be one. This is consistent with the initial term on each path having the value one. In the case of $\varepsilon^* =' no - edge'$, the probability becomes

$$\frac{P(\underline{k}|\varepsilon^* =' no - edge', \theta_0^*)}{P(\underline{k})} =$$

$$\frac{P(\underline{k}|\varepsilon^* =' no-edge', \theta_0^*)}{P(N)P(\underline{k}|\varepsilon^* =' E', \theta_0^*) + (1-P(N))P(\underline{k}|\varepsilon^* =' N', \theta^*)} \quad (III-41)$$

Thus,

$$\frac{P(\underline{k}|\varepsilon^* =' no-edge', \theta_0^*)}{P(\underline{k})}$$

$$\begin{aligned}
&= 1 \;\; if \; P(\underline{k}|\varepsilon^* =' E', \theta_0^*) = P(\underline{k}|\varepsilon^* =' N', \theta_0^*) \\
&> 1 \;\; if \; P(\underline{k}|\varepsilon^* =' E', \theta_0^*) < P(\underline{k}|\varepsilon^* =' N', \theta_0^*) \\
&< 1 \;\; if \; P(\underline{k}|\varepsilon^* =' E', \theta_0^*) > P(\underline{k}|\varepsilon^* =' N', \theta_0^*)
\end{aligned}$$

The way of implementing the recursive algorithm for $g_{L_{rc}}(\varepsilon_{rc}^*, \theta_{rc}^*)$ is similar to what we describe above. We append one row at the bottom of the image (r=(Image row size) +1) and append one column to the right of the image (c=(Image Column size) + 1). Then initialize $g_{L_{rc}}(\varepsilon_{rc}^*, \theta_{rc}^*)$ of the appended portion to the value one.

In the derivations we treat context of 'edge to edge', 'edge to no-edge', and 'no-edge to no-edge' equally. However, in the context of edge lines. 'edge to edge' consistency is much stronger than 'no-edge to edge' and 'no-edge to no-edge' consistency. We endeavor to remedy this unequal strength of context by emphsizing the 'edge to edge' context. We will evaluate an edge path assuming all its pixels belong to the edge state='edge' and on the other hand, evaluate a no-edge path to have all its pixels belong to the path only edge state='no-edge'.

In order to have a balance between local information and context information we set limitations on the context influence. Thus, we set a lower bound of influence ϵ_{min} and an upper bound of influence ϵ_{max}, where $\epsilon_{min} < 1 < \epsilon_{max}$. The maximum possible value of any g and h functions are limited to ϵ_{max}. Similarly, the minimum possible value of any g and h functions are limited to ϵ_{min}. Thus, (III-19) becomes

$$g_{U_{rc}}(\varepsilon^*, \theta^*_{0r,c}) = \frac{P(\underline{k}_{rc}|\varepsilon^*, \theta^*_{rc})}{P(\underline{k}_{rc})} * \max\{$$

$$\min\left\{g_{U_{r,c-1}}(\varepsilon^*, \theta^*_{0r,c-1}) * A(\varepsilon^*, \theta^*_{0r,c-1}, \varepsilon^*, \theta^*_{0r,c}), \epsilon_{max}\right\},$$

$$\min\left\{h_{U^*_{r-1,c-1}}(\varepsilon^*, \theta^*_{0r-1,c-1}) * A(\varepsilon^*, \theta^*_{0r-1,c-1}, \varepsilon^*, \theta^*_{0r,c}), \epsilon_{max}\right\},$$

$$\min\left\{h_{U^*_{r-1,c}}(\varepsilon^*, \theta^*_{0r-1,c}) * A(\varepsilon^*, \theta^*_{0r-1,c}, \varepsilon^*, \theta^*_{0r,c}), \epsilon_{max}\right\},$$

$$\min\left\{h_{U^*_{r-1,c+1}}(\varepsilon^*, \theta^*_{0r-1,c+1}) * A(\varepsilon^*, \theta^*_{0r-1,c+1}, \varepsilon^*, \theta^*_{0r,c}), \epsilon_{max}\right\},$$

$$\epsilon_{min}\} \qquad (III-42)$$

where ε^* can be either 'edge' or 'no-edge'. Similar equations hold for $g_{L_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc}), h_{U^*_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$, and $h_{L^*_{rc}}(\varepsilon^*_{rc}, \theta^*_{rc})$.

III-4. Experimental results

To understand the performance of the full context edge detector we examine the behavior of the full context edge detector on one well structured simulated image and two real images and compare the results with the context free second derivative zero-crossing edge operator to see how and in what degree the full context information can improve the operator.

The simulated test image is a noisy bar image. The image size is 100 X 50 pixels. The dark bars have pixel intensity 0 and the white bars have pixel intensity 175. We apply a 2 X 2 box filter on this image to simulate ideal single pixel width edge lines. We then add a zero mean Gaussian noise with standard deviation 40 to this image. The image is the same as what we used in chapter III and is shown in *Figure III-4(a)*.

To show the motivation of treating 'edge' context and 'no-edge' context differently, we try two schemes on the simulated image. The first scheme uses best non-edge path for non-edge context. Thus, it uses $f_{Z_{rc}}(\varepsilon^*_{rc} =' no - edge')$ as non-edge probability in the edge detection process. The second scheme uses all the best non-edge path for non-edge context by using the average probability $\overline{f}_{Z_{rc}}(\varepsilon^*_{rc} =' no - edge')$ as non-edge probability in the edge detection process.

We test the image by applying the scheme which uses single optimal non-edge path passing through each pixel for non-edge context against the scheme which uses the average of the non-edge paths from different directions passing through each pixel for non-edge context. On this image we fit each 5 X 5 neighborhood by a cubic polynominal. and then apply the two full context operators on it. The resulting edge images of the first scheme and the second scheme are shown in *Figure III-3(a) and (b)*.

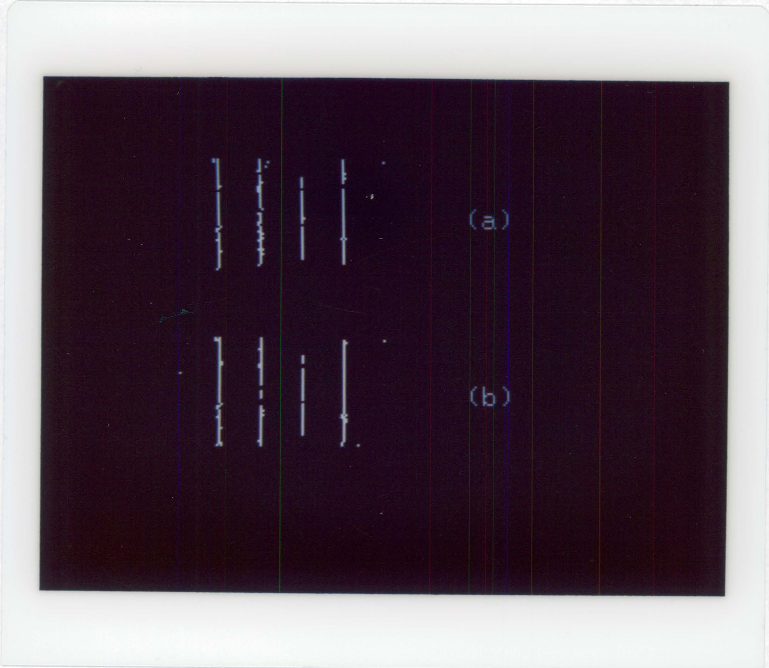


Figure III-3. (a) shows the edge image of the first non-edge context scheme. (b) shows the edge image of the second non-edge context scheme.

In order to quantitively see the difference in performance of these two operators, we use the conditional probability of assigned edge state 'edge' given the true edge states 'edge' of an image, $P(E'|E^*)$, and the conditional probability of true edge states 'edge' given assigned edge states 'edge' of an image $P(E^*|E')$. The adjustable parameters of each edge operator are chosen to equalize these two conditional probabilities. The quality of the edge operator is determined by the value of $P(E'|E^*) = P(E^*|E')$. The performance in terms of $P(E'|E^*)$ and $P(E^*|E')$ are shown in *Table III-1.* It lists the test results of the edge operator for the two different non-edge context schemes. Both context operators have much better performance than the context free edge operator which has probability of correct assignment of around 0.69 (see table III-1). The results show that the second scheme performs better than the first. From here on we will use the second scheme for the rest of the experiments.

We then apply the full context edge detector on two noisy object images. The object images are simulated range images with image size 128 by 128 pixels on which we add zero mean Gaussian noise of standard deviation 30. The images are shown in *Figures III-4(a), and III-5(a)* . On these images we first apply a cubic polynominal fitting with neighborhood size 5 X 5 and then apply both context free second derivative zero-crossing edge operator and the full context

Table III-1. $P(E'|E^*)$ and $P(E^*|E')$ values of the full context edge operator with different schemes for non-edge context and the context free edge operator

operator \ prob	$P(E'\|E^*)$	$P(E^*\|E')$
average path	0.8400	0.8316
single path	0.8600	0.8600
context free	0.6950	0.6814

edge operator on these images. The context bounds we used for both images are ϵ_{min}= 0.1 and ϵ_{max}=10. The result of the context free edge operator on these images are shown in *Figure III-4(b), and III-5(b)*. The results of the full context edge operator on these images are shown in *Figure III-4(c), and III-5(c)*. It can be easily verified that on both of these images the full context edge operator performs much better. The edge images of the context edge operator have better connectivity and much less noisy than the context free edge operator.

The edge consistent function we used is a linear combination of the functions $d(\alpha, \beta)$ and $d1(\alpha, \beta)$ which are defined by (III-33). The cosine based function we used has the characteristic that as α approaches β the function has less penality (higher value) than the absolute difference function which computes absolute difference between two angles. Conversely, it gives more penality when the angle difference is large. Thus, the angle quantization effect due to the rectangular grid layout of the pixels will tend to be minimized. In order to see the performance of the context edge operator when employing different consistent functions. we use d' and $d1'$ to replace d and $d1$ for the edge consistent function. d' and $d1'$ are defined as a power of d and $d1$, respectively. Thus,

$$d'(\alpha, \beta) = (d(\alpha, \beta))^{power}$$

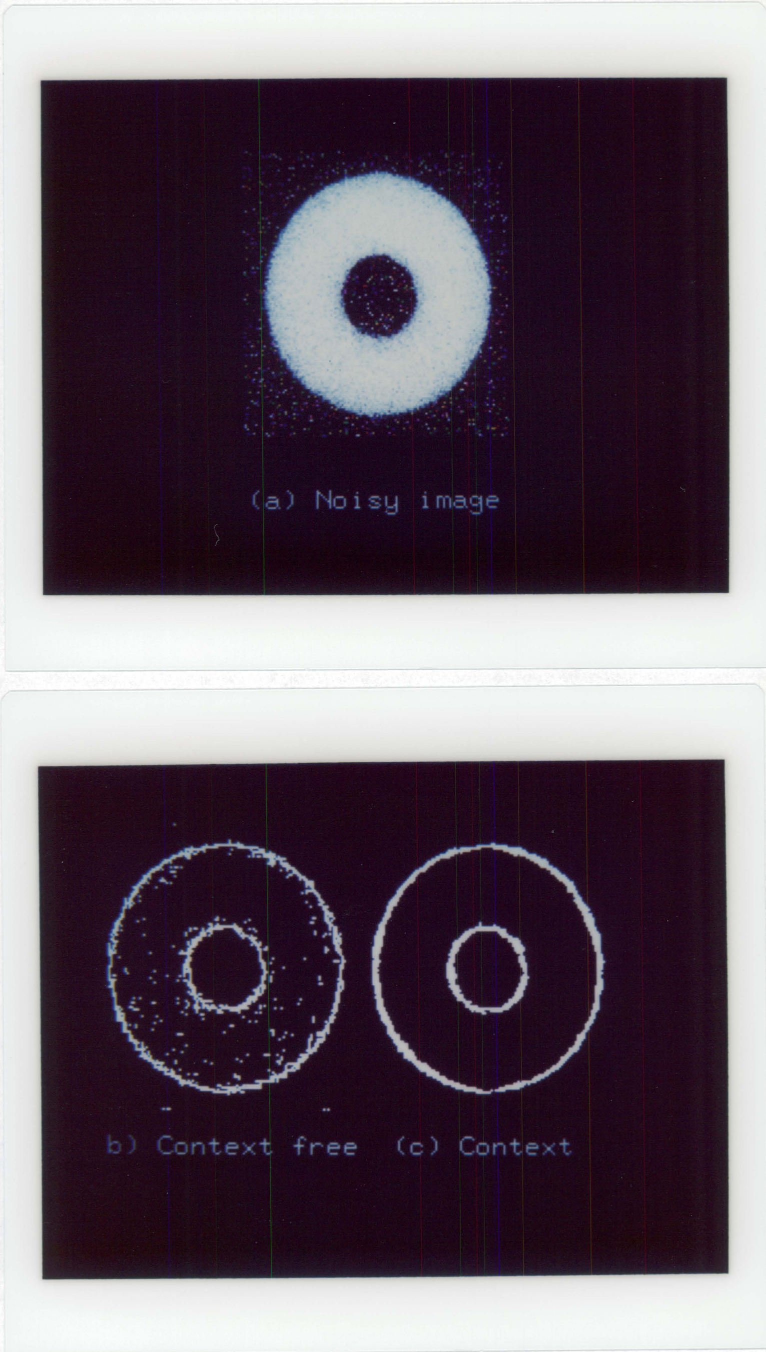


Figure III-4. The noisy object image 1 (a) and its context free edge image (b) and full context edge image(c). The window size for cubic polynomial fitting is 5 X 5.

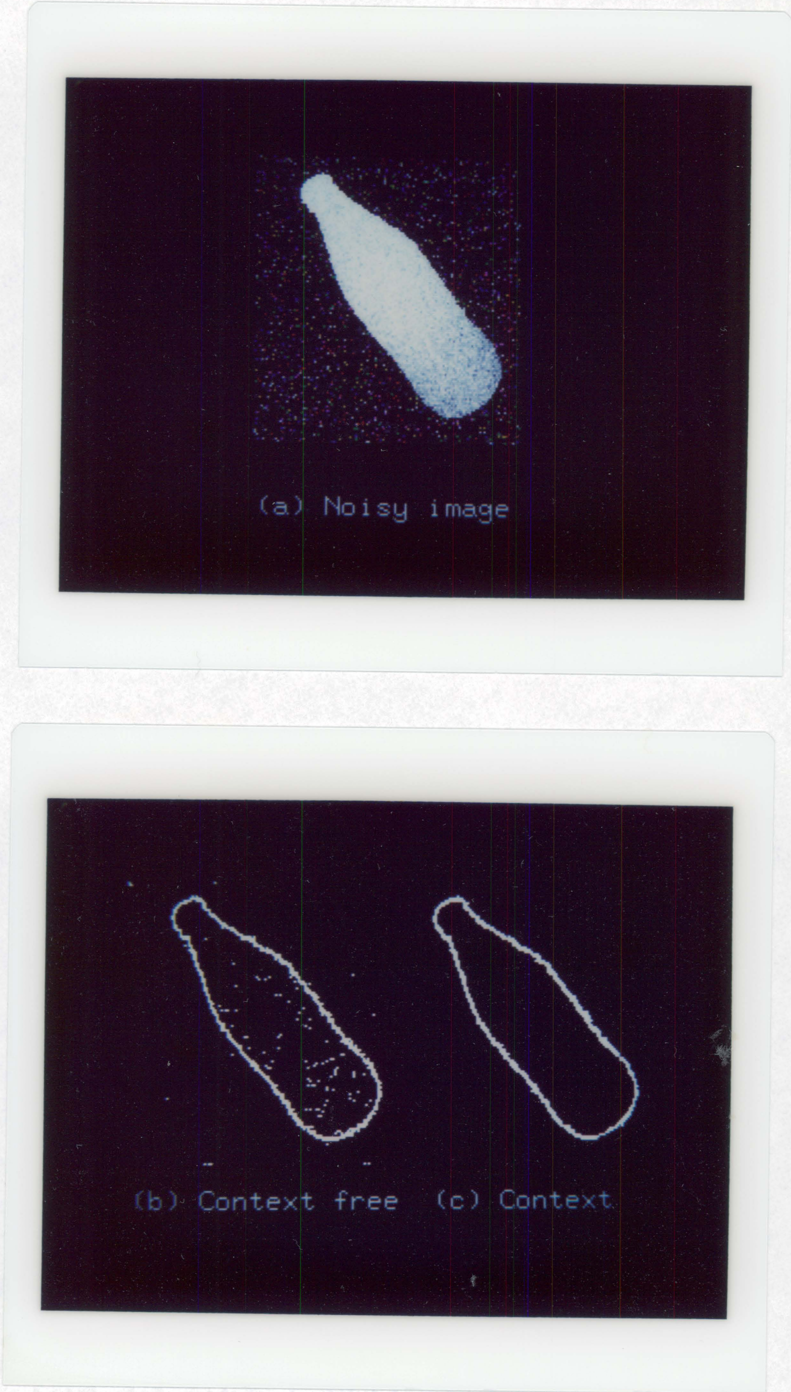


Figure III-5. The noisy object image 2 (a) and its context free edge image (b) and full context edge image(c). The window size for cubic polynomial fitting is 5 X 5.

and

$$dl'(\alpha, \beta) = (dl(\alpha, \beta))^{power}$$

We apply the context operator on the image shown in *Figure III-4(*a*)* by employing consistent functions with powers 0.3, 0.5, 1.0, 1.5, 2.0, 3.0, 4.0, and 5.0, respectively. All the parameter settings except the power of the consistent function are keep at the same values as we used to produce *Figure III-4(c)*. The edge results are shown in *Figure III-6*.

It is found that when the power is small the detected edge lines appear very thick and the connectivity of the edge lines is good. On the other hand, as the power of the consistent function becomes large, the detected edge lines appear thin and there are broken edge lines. The best edge image among them is the output of the context edge operator with power one. It is not difficult to explain this phenomenon. When the power of the consistent function is small the consistent function has high value in a wide range of angle difference. Thus, the influence of the context information in the edge detection process increases. Hence, the pixels adjacent to an edge pixel tend to be labeled as an edge pixel. On the other hand, when the power of the consistent function is large, the influence of the context information decreases. Thus, the context edge operator can not fill some of the gaps in the middle of edge lines.

Figure III-6. The context operator applies on the noisy object image 1 with different power of the consistent function. The powers are from left to right top to bottom: 0.3, 0.5, 1.0, 1.5, 2.0, 3.0, 4.0, and 5.0.

Finally, we apply the full context edge operator on an image which is corrupted by a signal dependent noise. The signal dependent noise image is generated by the function

$$SDN(f(r,c)) = N(f(r,c), \frac{f(r,c)}{5} + 10)$$

where $N(a,b)$ returns a normal random number with mean a and standard deviation b. We apply the signal dependent noise on object image 1. The noisy image is shown in *Figure III-7(a).* We fit each 5 X 5 neighborhood of the noisy image by a bivariate cubic polynomial. And then apply both the context-free second derivative zero-crossing edge operator and the full context operator on the cubic polynomial fitted image. The edge results are shown in *Figure III-7(b) and (c).* It is easy to verify that the edge image produced by the full context operator is better than the edge image produced by the context-free operator.

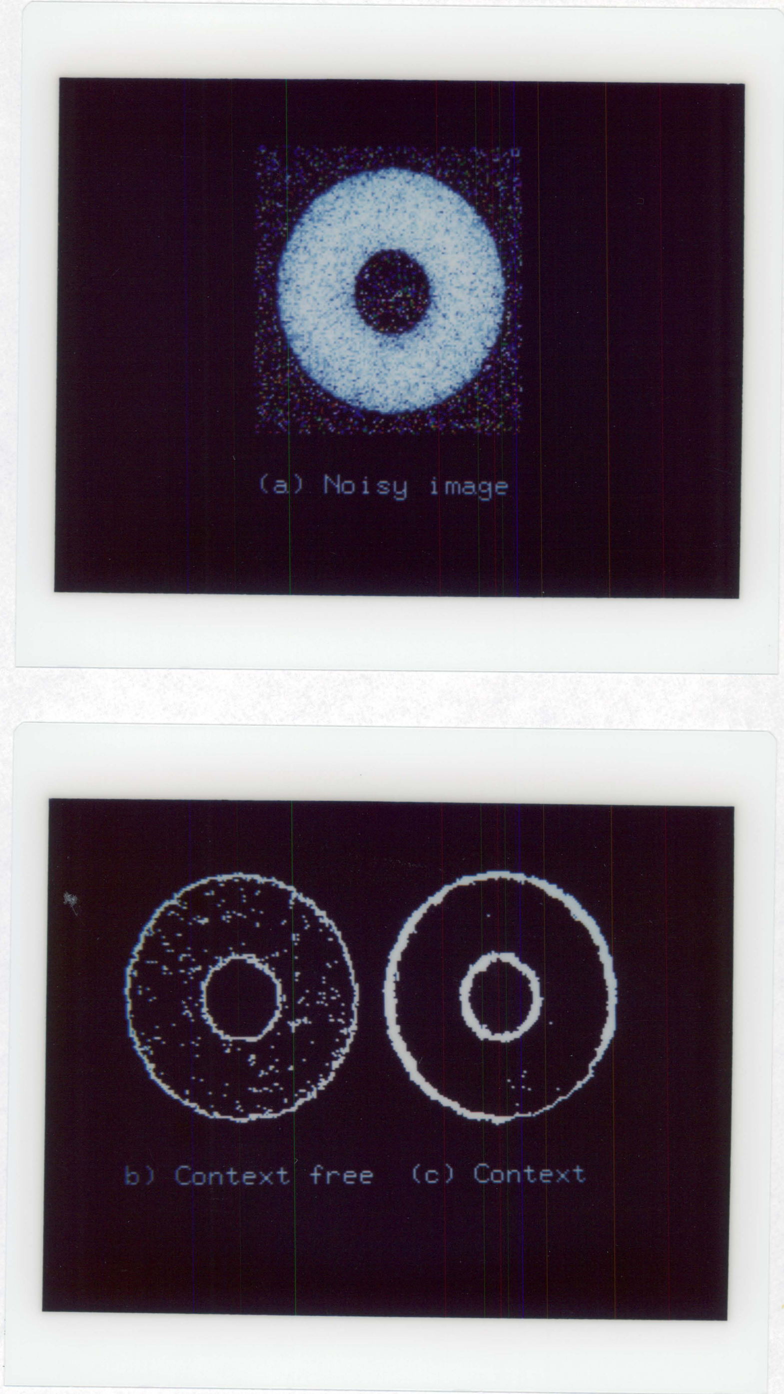


Figure III-7. The object image 1 corrupted by a signal dependent noise (a) and its context free edge image (b) and full context edge image(c). The window size for cubic polynomial fitting is 5 X 5.

IV. EDGE DETECTION USING WORLD CONSTRAINTS

Although it is possible to derive an edge detecting algorithm and argue that it is mathmatically optimal under certain ideal image models, the edge detecting problem for real scenes has not yet been completely solved. The reason is that there is still no acceptable complete model of the image intensity surface which corresponds to edges in the real world scenes. People only select some particular image intensity surface patterns (for example, step or ramp function) and look for an edge operator which can optimally detect these particular image intensity surface patterns. Thus, there is no guarantee that the operator derived from these incomplete patterns can optimally identify all kinds of image intensity changes corresponding to edges of real scenes. However, the edges in the real scene domain are not difficult to define and they are less ambiguous than the edges defined in the image domain. The ambiguity with image edges is mainly caused by the unknown factors involved in the many to one physics gowerning image acquisition.

Edges in a scene are the consequence of changes in some physical properties of surfaces of the scene, such as illumination (shadows, for example), geometry (orientation or depth) and reflectance. In general, due to noise, occlusion or susceptibility to lighting changes the images are degraded models of real world scenes. In order to handle these degraded data it is better if we use not only information extracted from the image but also world constraints from scenes. Thus, if an image intensity change around a pixel satisfies all the edge requirements we set on the shape of the greytone intensity surface we will not identify it as an edge pixel unless it also satisfies the requirements imposed by the world constraints. We have discussed in chapter II a context dependent scheme which uses the world constraint that real scene edges should be formed as continuous arcs instead of as independent points. In this chapter, we introduce some more world constraints which reflect the characteristics of real scene edges and when used in the edge detection process can further improve the performance of edge detection on degraded real images.

IV-1. Compensating for non-uniform lighting

In general, the illumination intensity on a scene is hardly uniform. Thus some areas of an image are illuminated more or less than other areas. In a brighter area there may be some relatively high contrast local neighborhoods which do not correspond to edges of three dimensional surfaces. While, in a darker area there may be some relatively low contrast local neighborhoods which correspond to edges of surfaces, This implies that any gradient based edge detection process which has no lighting compensation could not be expected in general to have good results.

In this section we introduce a dynamic compensation scheme which can take care of the non-uniform lighting in an image and have uniformly good results. The scheme is similar to the scheme described by Barrow and Tenenbaum (1981). The scheme of Barrow et.al. is employed to recover shape from shading and recover surface albedo from shading while the scheme we disscuss in this section is employed to compensate for the non-uniform lighting of edges. Besides, the former only allows a change of either lighting or albedo while the latter simultaneously allows for lighting, albedo, and orientation changes. *Figure IV-1* shows a simple model of image generation(see Pentland(1984)). A distant point source illuminant at direction $\vec{L}$, a patch of surface with surface normal $\vec{N}$, and a viewer in direction $\vec{V}$. Where $\vec{L}, \vec{N}, \vec{V}$ are unit three dimensional vectors.

The image irradiance I can be generally given by

$$I = \rho\lambda(\vec{N} \cdot \vec{L}) R(\vec{N}, \vec{L}, \vec{V}) (\vec{N} \cdot \vec{V})^{-1} \qquad (IV-1)$$

where ρ, called albedo, is the percentage of incident flux which is reflected, λ is the amount of radiant flux incident upon the surface and $R(\vec{N}, \vec{L}, \vec{V})$ is the reflectance function which describes how much of the reflected light leaves in each direction. The term $(\vec{N} \cdot \vec{L})$ describes the amount of light incident on the surface, while the term $(\vec{N} \cdot \vec{V})^{-1}$ describes the foreshortening that occurs during projection

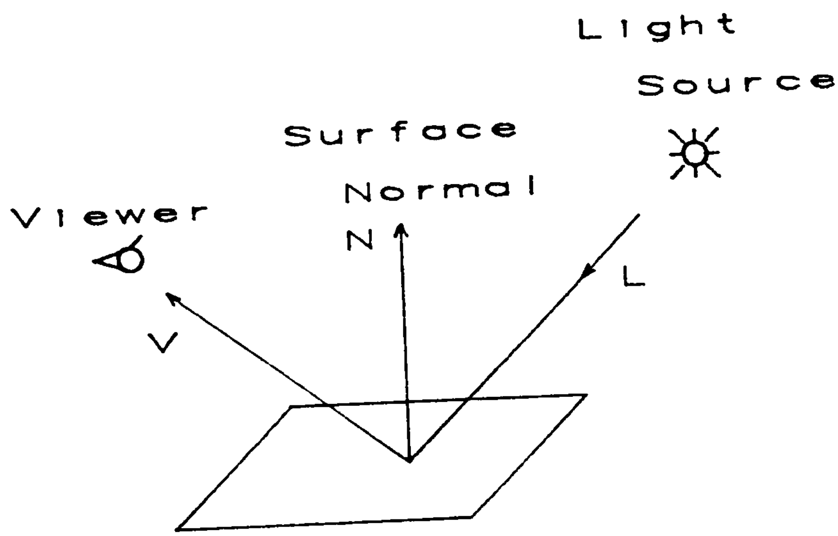


Figure IV-1. A simple model of image generation.

into the image. A Lambertian reflectance function, an idealization of a rough, matte surface, is defined as

$$R(\vec{N}, \vec{L}, \vec{V}) = \vec{N} \cdot \vec{V}$$

A surface with a Lambertian reflectance function scatters incident light isotropically. We shall assume a Lambertian reflectance function. Under these assumptions, the image irradiance I becomes

$$I = \rho\lambda(\vec{N} \cdot \vec{L}) \qquad (IV - 2)$$

The assumption of a Lambertian reflectance function is not as restrictive as it might seem. Even very specular reflectance functions are nearly Lambertian outside of the specular areas. For a Lambertian surface any constant distribution of illumination is equivalent to a single distant point source illuminant. This follows from application of the mean value theorem. Thus, in the following we use the single point source model for the derivations of the lighting compensation scheme.

Edges in images arise from several very different kinds of scene events: material reflectance changes, discontinuities of surface orientation, occluding contours and cast shadows, etc. In the following, we shall discuss a compensation technique which can take care of most of the cases.

A. Edges caused by material reflectance changes

Under the Lambertian reflectance model, Equation (IV-2) implies

$$dI = d(\rho\lambda(\vec{N} \cdot \vec{L})) \qquad (IV - 3)$$

If we are examining a small edge region of an image, it is reasonable to assume that the illumination and surface normals change very little, and so we may treat $\lambda, \vec{N}$ and $\vec{L}$ as constants. Then (IV-3) becomes

$$dI = \lambda(\vec{N} \cdot \vec{L})d\rho$$

Assume we have lighting changes in this image. Thus, the radiant flux is not uniform around the whole image. If the same kind of edge region in a different position of the image is illuminated by a different radiant flux λ'. Then it has a different irradiance change dI' and

$$dI' = \lambda'(\vec{N} \cdot \vec{L})d\rho$$

Let ρ_h be the albedo of the surface on that side of edge which has higher image irradiance I_h or $I_{h'}$. Then

$$\frac{dI}{dI'} = \frac{\lambda}{\lambda'} = \frac{\lambda\rho_h(\vec{N} \cdot \vec{L})}{\lambda'\rho_h(\vec{N} \cdot \vec{L})} = \frac{I_h}{I_{h'}}$$

$$or\ dI = \frac{I_h}{I_{h'}} * dI' \qquad (IV - 4)$$

Since the observed image intensity change is dI', to recover the effect of lighting change, we can apply *equation(IV-4)* to obtain dI. And then use dI instead of dI' in the edge detection process.

B. Edges caused by discontinuity of surface orientations

In a small edge region of an image, the intensity change caused by discontinuity of surface orientation can be described by the following equation:

$$dI = d(\rho\lambda(\vec{N} \cdot \vec{L})) = \rho\lambda(d\vec{N} \cdot \vec{L})$$

Assume that the lighting is non-uniform in this image and the same edge region is illuminated by a different radiant flux λ'.

$$dI' = \rho\lambda'(d\vec{N} \cdot \vec{L}) \quad and$$

$$\frac{dI}{dI'} = \frac{\lambda}{\lambda'} = \frac{\lambda\rho(\vec{N}_h \cdot \vec{L})}{\lambda'\rho(\vec{N}_h \cdot \vec{L})} = \frac{I_h}{I_{h'}}$$

$$or \; dI = \frac{I_h}{I_{h'}} * dI'$$

where N_h is the surface normal of the surface which corresponds to higher image irradiance I_h or $I_{h'}$. in one side of edge.

The same as in case (A) to compensate for the effect of lighting change, we can apply equation (IV-4) to recover dI and use dI in the edge detection process.

C. Edges caused by occluding contours

The intensity change caused by occluding contours is given by

$$dI = d(\rho\lambda(\vec{N}\cdot\vec{L}))$$

$$= \frac{d(\rho\lambda(\vec{N}\cdot\vec{L}))}{d\rho} * d\rho + \frac{d(\rho\lambda(\vec{N}\cdot\vec{L}))}{d\lambda} * d\lambda + \frac{d(\rho\lambda(\vec{N}\cdot\vec{L}))}{d\vec{N}} * d\vec{N}$$

$$+ \frac{d(\rho\lambda(\vec{N}\cdot\vec{L}))}{d\vec{L}} * d\vec{L}$$

$$= \lambda(\vec{N}\cdot\vec{L})d\rho + \rho(\vec{N}\cdot\vec{L})\frac{d\lambda}{dr} * dr + \rho\lambda(d\vec{N}\cdot\vec{L}) + \rho\lambda(\vec{N}\cdot d\vec{L}) \quad (IV-5)$$

Where r is the distance between the point source and the illuminanted surface and λ is a function of r which can be written as $\lambda = \frac{\lambda_0}{r^2}$. Thus $\frac{d\lambda}{dr} = \frac{-2\lambda_0}{r^3}$ and (IV-5) becomes

$$dI = \frac{\lambda_0}{r^2}[(\vec{N}\cdot\vec{L})d\rho - 2\rho(\vec{N}\cdot\vec{L})\frac{dr}{r} + \rho(d\vec{N}\cdot\vec{L}) + \rho(\vec{N}\cdot d\vec{L})] \quad (IV-6)$$

Assume the image has non-uniform lighting and the same sort of edge points are illuminated by a different radiant flux λ'. Where λ can be expressed as $\frac{\lambda_0'}{r^2}$. Then

$$dI' = \frac{\lambda_0'}{r^2}[(\vec{N}\cdot\vec{L})d\rho - 2\rho(\vec{N}\cdot\vec{L})\frac{dr}{r} + \rho(d\vec{N}\cdot\vec{L}) + \rho(\vec{N}\cdot d\vec{L})]$$

and

$$\frac{dI}{dI'} = \frac{\lambda}{\lambda'} = \frac{\lambda\rho_h(\vec{N}_h\cdot\vec{L}_h)}{\lambda'\rho_h(\vec{N}_h\cdot\vec{L}_h)} = \frac{I_h}{I_{h'}}$$

$$or\ dI = \frac{I_h}{I_{h'}} * dI'$$

where $\rho'_h, \vec{N}_h and \vec{L}_h$ are the $\rho, \vec{N}$ and $\vec{L}$ values of the surface corresponds to one side of the edge region which has higher I_h or $I_{h'}$ value. To compensate for the effect of lighting change, equation (IV-4) can be applied to recover dI.

In summary, to compensate for the effect of lighting change, for a given image, we first estimate the expected I_h value from all the image data and then estimate $I_{h'}$ for each local neighborhood. We multiply the intensity change of the local neighborhood dI' by the factor $\frac{I_h}{I_{h'}}$ to compensate for the effect of lighting changes. Since the term $\frac{I_h}{I_{h'}}$ is different from neighborhood to neighborhood, we can also call it dynamic compensation.

We next show how to estimate $\frac{I_h}{I_{h'}}$ by means of the cubic facet model on a local edge neighborhood. Let the gradient direction of the center pixel of the neighborhood be θ We wish to only consider points (r,c) on the line in direction θ. Hence $r = \rho \sin\theta \ and \ c = \rho\cos\theta$. Then we have

$$f_\theta(\rho) = \frac{A}{6} * \rho^3 + \frac{B}{2} * \rho^2 + D\rho + k_1$$

$$f'_\theta(\rho) = \frac{A}{2} * \rho^2 + B\rho + D$$

$$f''_\theta(\rho) = A\rho + B$$

$$and \ f'''_\theta(\rho) = A \qquad (IV-7)$$

where ρ is the distance along the direction θ between (r,c) and (0,0),$A = 6[k_7 \sin\theta^3 + k_8 \sin^2\theta\cos\theta + k_9 \sin\theta\cos^2\theta + k_{10}\cos^3\theta]$, $B = 2[k_4 \sin^2\theta + k_5 \sin\theta\cos\theta + k_6\cos^2\theta]$ and $D = k_2\sin\theta + k_3\cos\theta$.

Let $\rho_0 = \frac{-B}{A}$, then $f_\theta''(\rho_0) = 0$. Therefore ρ_0 is the estimated edge position if ρ_0 is inside the edge neighborhood we have an edge candidate. Let ρ_1 be the boundary of the neighborhood in the θ direction so that the high intensity part of the surface profile starts from the position ρ_0 along the direction θ ends at ρ_1. Without loss of generality we assume $\rho_1 > \rho_0$. Then we can estimate I_h by the following rule

$$I_h = \frac{\int_{\rho_0}^{\rho_1} f_\theta(\rho)d\rho}{\rho_1 - \rho_0} =$$

$$\frac{\frac{A}{24} * (\rho_1^4 - \rho_0^4) + \frac{B}{6} * (\rho_1^3 - \rho_0^3) + \frac{D}{2} * (\rho_1^2 - \rho_0^2) + k_1(\rho_1 - \rho_0)}{\rho_1 - \rho_0}$$

$$= \frac{A}{24} * (\rho_1 + \rho_0) * (\rho_1^2 + \rho_0^2) +$$

$$\frac{B}{6} * (\rho_1^2 + \rho_1\rho_0 + \rho_0^2) + \frac{D}{2} * (\rho_1 + \rho_0) + k_1 \qquad (IV-8)$$

IV-2. Curvature constraint on the edge detection

In view of the edges in the real scenes, it is reasonable to expect an edge to be on the boundary between two large enough scene surfaces which correspond to two significantly different image regions. The small regions in an image are most probably caused by noise or correspond to minor features in a scene. Thus, the boundaries of these small regions should not be identified as edges. This world information can be formulated as a curvature constraint on the image intensity profile taking in the gradient direction of the pixel under consideration. The constraint is that the maximum curvature of the profile in the neighborhood of an edge point should not be too high. Because, if the curvature is high, it will most probably corresponding to a noise point.

In order to incorporate this constraint into the edge detection process, we introduce an image feature called normalized gradient which is defined for each pixel position as the gradient value of the pixel divided by the absolute value of the maximum curvature in the neighborhood of this pixel. If the graytone intensity profile in the gradient dirction of an image position has small normalized gradient, it is hardly possible that this image position adjacents two large enough image regions. Thus, this image position should not be an edge pixel. Hence, the normalized gradient of image graytone intensity surface around an edge point must not be too small and in this way the curvature information can aid the edge detecting process.

We now show the way to obtain the maximum curvature of an image position taking in the gradient direction by means of cubic facet image model. Let f(r,c) be the surface function underlying the neighborhood of pixel centered at the image position (0,0). Let $f_\theta(\rho)$ be the surface function along the gradient direction θ and ρ be the distance between (r,c) and (0,0). It is well known that the curvature at any point of this one dimensional profile can be described by

$$k_\theta(\rho) = \frac{|f''_\theta(\rho)|}{\left(1 + (f'_\theta(\rho))^2\right)^{1.5}} \qquad (IV-9)$$

Based on the cubic facet model we have

$$k_\theta(\rho) = \frac{|A\rho + B|}{\left(1 + (\frac{A}{2}\rho^2 + B\rho + D)^2\right)^{1.5}} \qquad (IV-10)$$

where A, B, and D are defined in *equation (IV-7).* The maximal curvature along the gradient direction appears at position ρ_0, such that $f'_\theta(\rho_0) = 0$. We compute the curvature $k_\theta(\rho)$ at position ρ_0. That is

$$k_\theta(\rho_0) = |A\rho_0 + B| \qquad (IV-11)$$

Since $f'_\theta(\rho) = \frac{A}{2} * \rho^2 + B\rho + D = 0$ we can solve for ρ_0 which is the smaller root. Thus,

$$\rho_0 = \frac{-B + \sqrt{B^2 - 2AD}}{A}$$

and

$$k_\theta(\rho_0) = \sqrt{B^2 - 2AD}$$

The gradient of the point at $\rho = 0$ is D. Thus the normalized gradient is

$$G_n = \frac{D}{\sqrt{B^2 - 2AD}} \qquad (IV-12)$$

We now show how the normalized curvature constraint can aid in the edge detection process by means of a simple one-dimensional example. Let 5 points in 1-dimensional space be symmetric indexed by the set $R = \{-2, -1, 0, 1, 2\}$. From Haralick(1981) we know that the discrete orthogonal polynominal set up to third order for the index set R is $\{1, r, r^2 - 2, r^3 - \frac{17}{5}r\}$.

The least square error fitting of the discrete orthogonal polynominal set on the intensity function I(r) which has domain R yields

$$f(r) = a_0 + a_1 r + a_2(r^2 - 2) + a_3(r^3 - \frac{17}{5}r) \qquad (IV - 13)$$

and the coefficients $a_0, \ldots, a_3$ can be obtained by summing over I(r) by the weights of the following masks:

(1) a_0 mask

$\frac{1}{5}$ * [1 1 1 1 1]

(2) a_1 mask

$\frac{1}{10}$ * [-2 -1 0 1 2]

(3) a_2 mask

$\frac{1}{14}$ * [2 -1 -2 -1 2]

(4) a_3 mask

$\frac{1}{72}$ * [-6 12 0 -12 6]

Since

$$f(r) = a_0 + a_1 r + a_2(r^2 - 2) + a_3(r^3 - \frac{17}{5} r)$$

we have

$$f'(r) = (a_1 - \frac{17}{5} a_3) + 2a_2 r + 3a_3 r^2 \qquad (IV-14)$$

$$f''(r) = 2a_2 + 6a_3 r \qquad (IV-15)$$

$$f'''(r) = 6a_3 \qquad (IV-16)$$

and according to (IV-11) the maximum curvature is

$$k(r_{max}) = \sqrt{4a_2^2 - 12a_3(a_1 - \frac{17}{5} a_3)} \qquad (IV-17)$$

By solving $f''(r_0) = 0$ we have the position of second derivative zero crossing as

$$r_0 = -\frac{1}{3} * \frac{a_2}{a_3} \qquad (IV-18)$$

An ideal step edge in 1-dimensional discrete space can be represented as

$$I(r) = \begin{cases} a & \forall\, r \in \{-2, -1, 0\} \\ b & \forall\, r \in \{1, 2\} \end{cases}$$

where $b > a$ and the value b-a is the contrast of the step edge.

By applying the masks to I(r) we can obtain the least square error discrete orthogonal polynomial fitting coefficients for the ideal step edge as follows

$$a_0 = \frac{1}{5}(3a + 2b)$$

$$a_1 = \frac{3}{10}(b - a)$$

$$a_2 = \frac{1}{14}(b - a)$$

$$a_3 = \frac{1}{12}(a - b)$$

Thus, the gradient at the center position is

$$f'(0) = a_1 - \frac{17}{5}a_3 = \frac{7}{12}(b - a)$$

and the maximum curvature is

$$k(r_{max}) = \sqrt{\frac{1}{49} + \frac{7}{12}}(b - a)$$

Besides, the position of zero crossing of second derivative is

$$r_0 = \frac{2}{7}$$

Now let's carry out the same procedure on a noise point specified by

$$J(r) = \{ \begin{matrix} a & if\ r \neq 1 \\ b & if\ r = 1 \end{matrix}$$

where $b > a$ and the value b-a is the height of the noise point. By applying the masks to J(r) we can obtain the least square error discrete orthogonal polynominal fitting coefficients for ideal single noise point as follows

$$a_0 = \frac{1}{5}(4a + b)$$

$$a_1 = \frac{1}{10}(b - a)$$

$$a_2 = \frac{1}{14}(a - b)$$

$$a_3 = \frac{1}{6}(a - b)$$

Thus, the gradient at the center position is

$$f'(0) = a_1 - \frac{17}{5}a_3 = \frac{2}{3}(b - a)$$

and the maximum curvature is

$$k(r_{max}) = \sqrt{\frac{1}{49} + \frac{4}{3}}(b - a)$$

Besides, the position of the zero crossing of the second derivative is

$$r_0 = \frac{1}{7}$$

It is easy to find that the noise point has higher estimated gradient value than that of the edge point. It is even worse that the noise point has estimated second derivative zero crossing point closer to the center when compared with that of the edge point. Therefore, we will definitely identify the noise point as an edge when we try to identify the edge point based on the context free criteria described in section

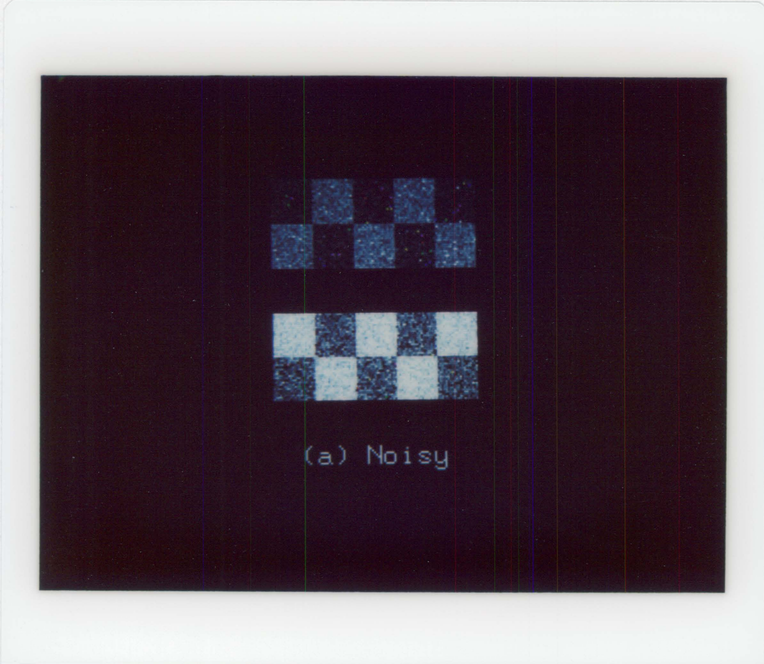
II. Now we show how the curvature constraint can aid in this example. By dividing the gradient by the curvature we can get a normalized gradient. Since the curvature of a noise point is larger than the edge point, the normalized gradient of the edge point (0.75) is now larger than that of the noise point (0.57). Furthermore, since both gradient and curvature values depend on the edge contrast (or noise height) b-a. Thus, b-a will be cancelled out in the normalized gradient. Hence, the normalized gradient can have uniform performance for different edge contrasts.

IV-3. Experiment results

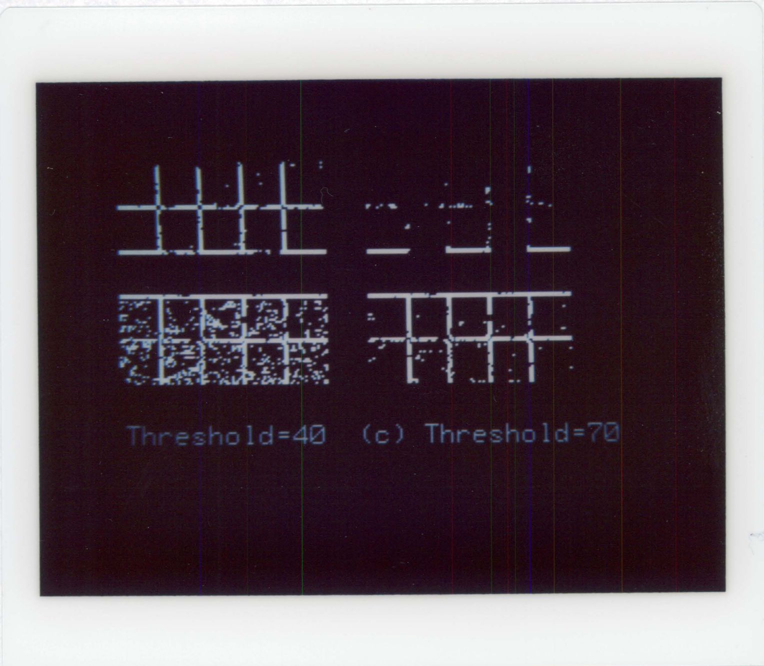
To demonstrate the performance of an edge detectors incorporating world constraints, We use the lighting cpmpensated cubic facet based second derivative zero-crossing edge operator and examine its behavior on one well structured simulated image and two real images. We then apply the curvature constrained operator on two simulated images. In all the experiments we compare the performance of the new operator with the old operator to see how and in what degree the world constraints can improve the operator.

The simulated image for the lighting compensation experiment is a checkerboard of size 100 x 100 pixels with a check size of 20 x 20 pixels. In order to simulate an extreme of the non-uniform lighting condition. We have all the checks in the middle row of the checkerboard greyscale intensity zero(darkest) and thus separate the checkerboard into top portion and bottom portion. The dark checks of the top portion have gray tone intensity 38 and the light checks have gray tone intensity 88. The dark checks of the bottom portion have gray tone intensity 75 and the light checks have gray tone intensity 175. Thus, the bottom portion doubles the lighting of the top portion. To this checkerboard, we add independent Gaussian noise having mean zero and standard deviation 15 on the top portion of the checkerboard and add zero mean independent Gaussian noise of standard deviation 30 on the bottom portion of the checkerboard. The noisy non-uniform lighting checkerboard is shown in *Figure IV-2(a)*.

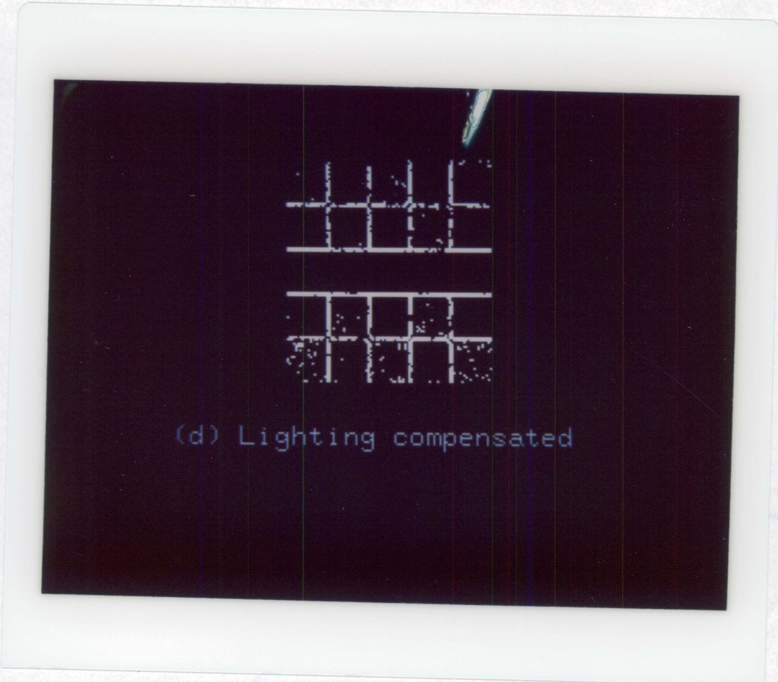
We first fit each 5 X 5 neighborhood of this image to a cubic polynominal and then apply the second derivative zero-crossing edge operator on this image. It is found that if the selected threshold value for edge detection is low , the bottom portion of the edge image is very noisy(see *Figure IV-2(b)*). On the other hand, if the threshold value is high, the top portion of the edge image becomes invisible (see *Figure IV-2(c)*). There is no appropriate gradient threshold value which can produce good edges on both top and bottom portions of the checkerboard. We then incorporate the lighting compensation



(a)



(b)



(c)

Figure IV-2. (a) shows the simulated noisy non-uniform lighting image. (b) and (c) show edge images of the second derivative zero crossing edge operator with different thershold settings.(d) shows the edge image of the same edge operator with lighting compensation.

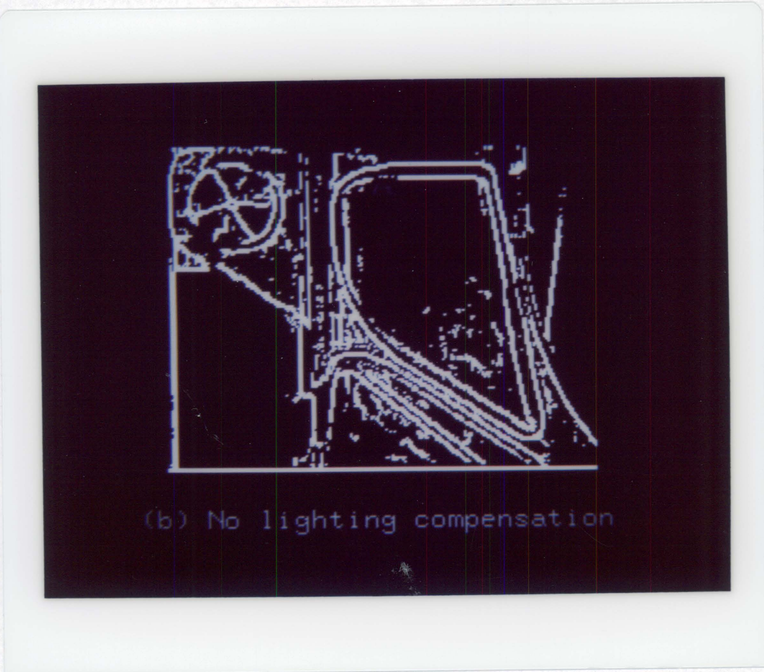
scheme with the same edge operator. The result (see *Figure IV-2(d)*) shows that this new operator can detect reasonably good edges on both top and bottom portions of this image.

In order to see how this scheme is also useful on real images, we apply both second derivative zero-crossing operator without and with lighting compensation on a pair of stereo bulkhead images. (see *Figure IV-3(a) and Figure IV-4(a)*). The image sizes are both 205 by 154 pixels.

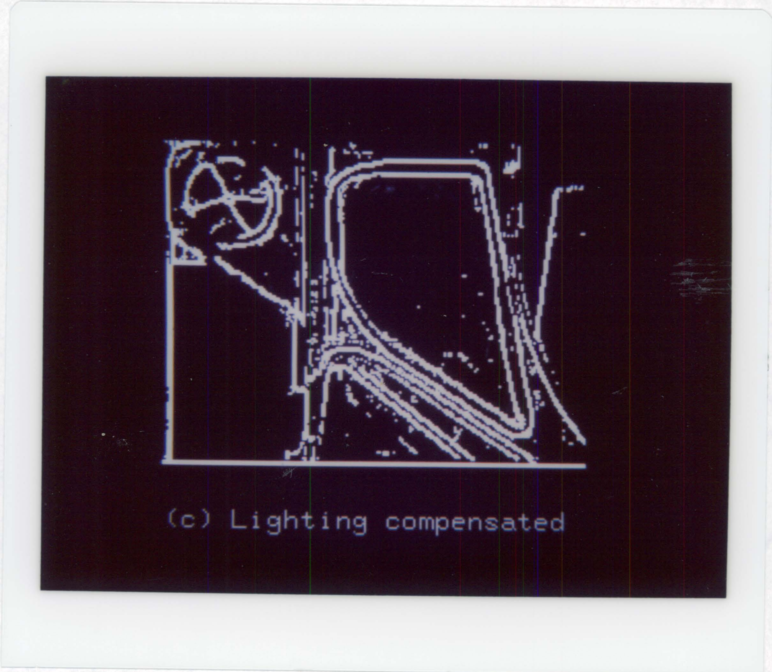
It is easy to see that due to the specular reflection of the bulkhead metal surface the lighting is non-uniform on both images. We first fit each 7 X 7 neighborhood of both images by a cubic polynomial and then apply the edge detectors on them. The edge detection result of the without compensation operator is shown in *Figure IV-3(b)* and *Figure IV-4(b)*. It is found that for both images they are pretty noisy in bright regions and on the other hand, have bad connectivity in the dark regions. the result of the same edge operator with lighting compensation are shown in *Figure IV-3(c) and Figure IV-4(c)*,respectively. It is easy to verify that on both images the new operator produces less noise and better connectivity edge images compare with the old one.

We are now ready to test the curvature constrained edge detection scheme. The first test image is a noisy bar image. The image

(a)

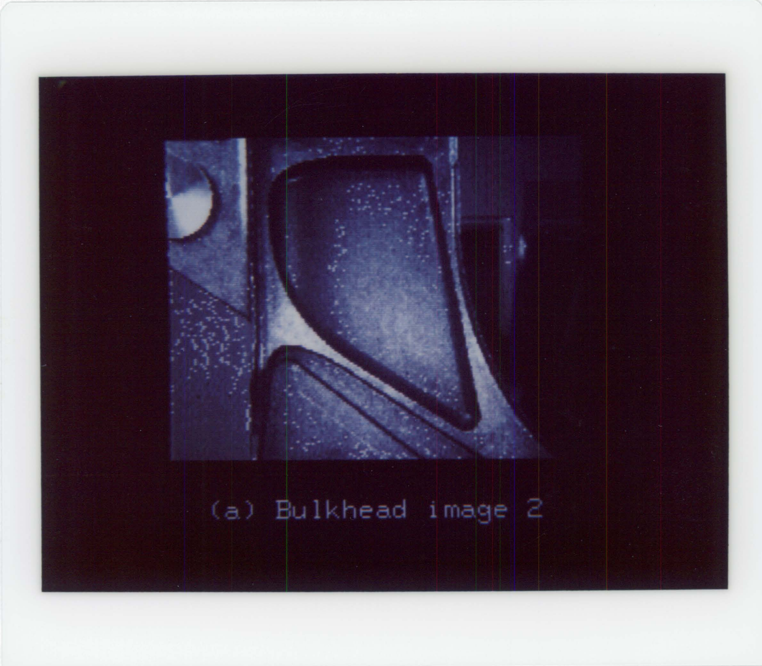


(b)

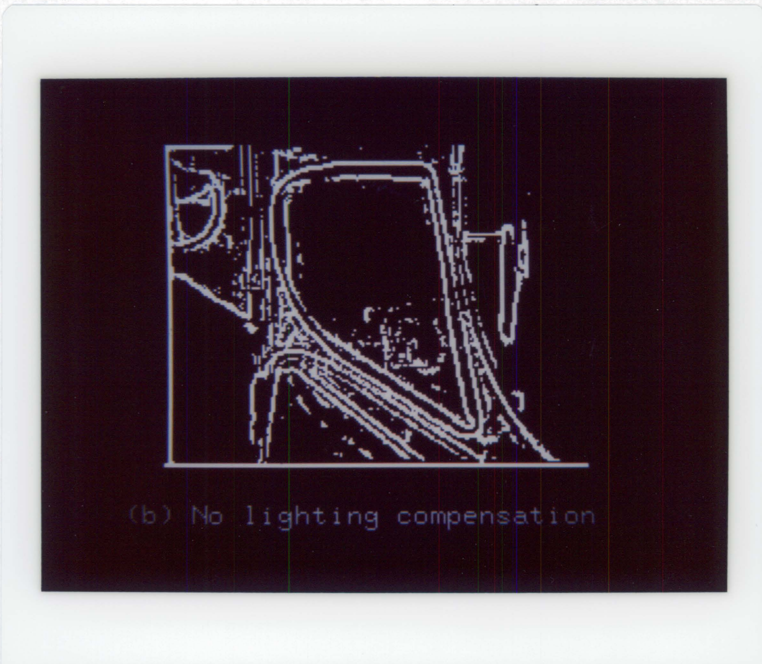


(c)

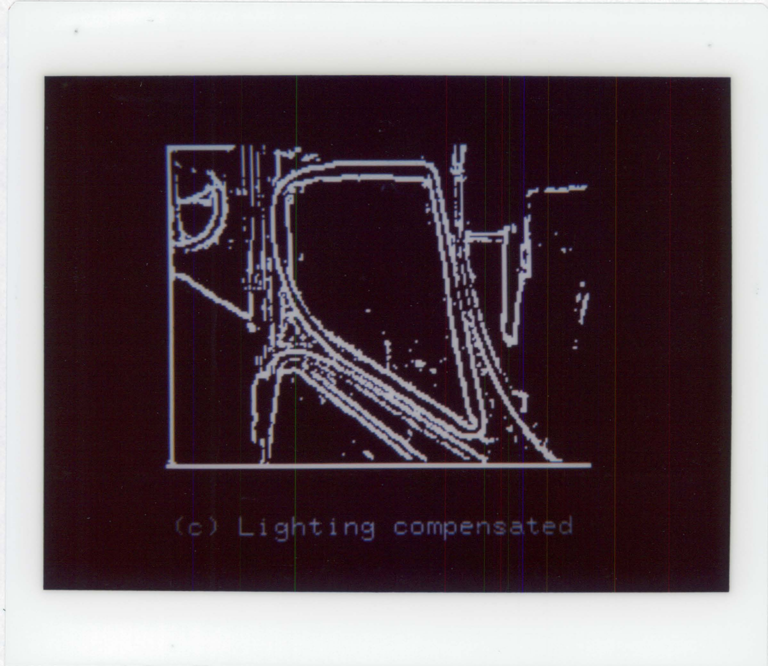
Figure IV-3. (a) shows the bulkhead image 1. (b) shows edge images of the second derivative zero crossing edge operator.(c) shows the edge image of the same edge operator with lighting compensation.



(a)



(b)



(c)

Figure IV-4. (a) shows the bulkhead image 2. (b) shows edge images of the second derivative zero crossing edge operator.(c) shows the edge image of the same edge operator with lighting compensation.

size is 100 X 50 pixels. The dark bars have pixel intensity 0 and the white bars have pixel intensity 175. We apply a 2 X 2 averaging on this image to simulate ideal single pixel width edge lines. We then add a zero mean Gaussian noise with standard deviation 40 on this image. The noisy image is shown in *Figure IV-5(a)*.

On this image we fit each 5 X 5 neighborhood by a cubic polynominal. and then apply second derivative zero crossing operators without and with curvature constraint. The result edge images are shown in *Figure IV-5(b) and (c)*. In order to quantitively see the difference in performance of these two operators. we use the conditional probability of assigned edge state 'edge' given the true edge states 'edge' of an image, $P(E'|E^*)$, and the conditional probability of true edge states 'edge' given assigned edge states 'edge' of an image $P(E^*|E')$. The adjustable parameters of each edge operator are chosen to equalize these two conditional probabilities. The quality of the edge operator is determined by the value of $P(E'|E^*) = P(E^*|E')$. Although this performance measure is not in general applicable on all kinds of images, it is well suitful for this simulated bar image. the performance in terms of $P(E'|E^*)$ and $P(E^*|E')$ are shown in *Table IV-1*. It lists the test results of the edge operator with and without curvature constraint. The results show that the conditional probabilities are improved from 68 percent to 75 percent by the curvature

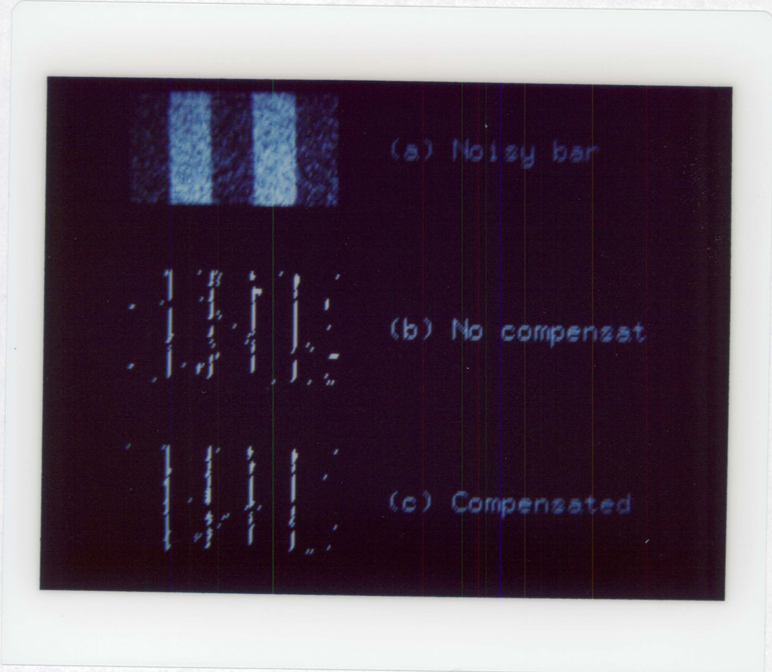


Figure IV-5. (a) shows the noisy bar image. (b) shows edge images of the second derivative zero crossing edge operator without curvature constraint.(c) shows the edge image of the same edge operator with curvature constraint.

constraint scheme.

Finally, we apply the with and without curvature constrained edge operators on a second image consisting of concentric light rings(grey level 140) on a dark background(grey level 115). This image is generted as 512 by 512 image, with a central dark circle of radius 64, surrounded by three bright rings of width 32, these being separated by two dark rings of the same width, with a dark surround. The decision as to whether a pixel should be light or dark was based on its Euclidean distance from the center of the image. Then this image was reduced to size 128 by 128, by replacing each 4 by 4 block with a single pixel having the average grey level of the block. The image is the same as the test image used by Kitchen and Rosenfeld(1981).(see *Figure IV-6* (a))

We add a zero mean Gaussian noise with standard deviation 10 on this image (Thus, following Pratt(1978)'s convention the signal-to-noise ratio is 6.25). Then we fit this image by a 5 X 5 cubic facet model. The fitted image is applied by the edge operators with and without curvature constraint. The resulting edge images are shown in *Figure IV-6(b) and (c)*. It can be easily verified by a visual evaluation that the one with curvature constraint has better result.

Table IV-1. $P(E'|E^*)$ and $P(E^*|E')$ values of the second directional derivative edge operator with and without curvature constraint.

prob \ operator	*without constraint*	*with constraint*
$P(E'\|E^*)$	0.6950	0.7500
$P(E^*\|E')$	0.6814	0.7425

Figure IV-6. (a) shows the noisy ring image. (b) shows edge images of the second derivative zero crossing edge operator without curvature constraint.(c) shows the edge image of the same edge operator with curvature constraint

V. A GENERAL EDGE EVALUATOR

V-1. Introduction

It is of interest to evaluate the quality of an edge detector, both to compare one detector scheme with another, and also to study the behavior of a given detector under different conditions and parameter settings. Several authors have proposed techniques for edge evaluation (Fram and Deutsch, 1974; ,Abdou and Pratt, 1979; Peli and Malah, 1982; and Kitchen and Rosenfeld ,1981).

Three basic failings in all of the other above edge evaluation schemes were noted by Kitchen and Rosenfeld. With the exception of Shaw, they all required prior knowledge of the true edge position. While this provides the opportunity to make definite statements concerning spatial precision, similar techniques are not applicable to real world images where the true edge positions are unknown. Another failing, as noted by Peli and Malah, is the general lack of a continuity measure. Edges that are fragmented but consistently displaced from the true edge, receive similar scores from Abdou and Pratt as perfectly continuous but similarly displaced edges. Finally, none of these schemes use consistency in the direction of the detected edges in their evaluation scores. Only Shaw notes edge directions, but only to compare the changes in direction of edge segments between noisy and noise-free images. Ideally the edge gradient direction should be everywhere perpendicular to the edge and in a manner consistent with adjacent edge pixels.

To address these criticisms and the undesirability of qualitative human intervention, Kitchen and Rosenfeld developed a fully automatic edge evaluation technique based on the idea of local edge coherence. The idea of local edge coherence is founded on the premise that ideal edge features should be locally line like. Edge coherence has, therefore, two components: edge pixels should be adjacent and connected. They should be thin like a line and ideally one pixel wide. Kitchen and Rosenfeld chose to incorporate these two components into one edge evaluation scheme. The first component, continuation (C), measures the degree to which adjacent pixels agree in their determination of the local edge direction. The thinness component (T) simply measures the local edge density. No knowledge of 'true' edge position is ever required. Both components can range in value 0 (poor) to 1 (perfect).

C and T are then combined into a convex combination evaluation measure for the center pixel of every 3X3 pixel neighborhood in the threshold edge-filtered image:

$$E = \gamma C + (1 - \gamma)T$$

where $\gamma(< 1)$ is a weighting coefficient adjusted to give E a suitable balance between thinness and coherence. The details of calculating C and T are clearly presented in Rosenfeld (1981).

One drawback of the Kitchen-Rosenfeld approach is that it has an inherent bias against curved edges. This is a result of the premise that ideal edge features should be locally line like. Besides, it can only take care of 3X3 neighborhood.

The thinness requirement proposed by Kitchen and Rosenfeld only allow edge lines of a single pixel width. This is apparently not true for the ideal step edges. An ideal step edge may have pixels on both sides of the edge. This is the reason why in their paper they chose $\gamma = 0.8$ (giving small weight to the thinness measure) to perform the experiments. And all the test images they used have only ideal ramp edges and no ideal step edges.

Furthermore, as mentioned by the authors, Kitchen & Rosenfeld's measure disregards the correct location of the edges. Thus, an edge detector that systematically mislocated edged will receive an evaluation measure equal to that of a detector which perfectly located edges. Besides, the approach is basically ad hoc. No underlying theories had been described.

In this chapter, we formulate the edge evaluation problem as a Bayesian decision problem and show that the edge evaluation of Kitchen and Rosenfeld is just a special solution to this problem. Finally, a general evaluator which can deal with any size of neighborhood is proposed. The edge detector can also measure the correctness of edge position. This is the first time a general edge position correctness measure has been proposed.

V-2. Edge coherence measure

In order to make decision about which edge operator performs best, the expected score of each edge operator must be evaluated. Let ε_i and θ_i be the observed edge state and edge direction at pixel i detected by a given edge operator; I is the set of the entire image; $A = \{j | j \in I \ and \ \varepsilon_j =' E'\}$; $\overline{A} = I - A$; E and θ are the observed edge state and edge direction of the whole image. And ε_i^* and θ_i^* are the true edge state and true edge direction at pixel i. $S(\varepsilon_i^*, \theta_i^*, \varepsilon_i = {}^{'E'}_{'N'}, \theta_i : i \in {}^{A}_{\overline{A}})$ is the score obtained when $\varepsilon_i = {}^{'edge'}_{'no-edge'}$ and θ_i are observed edge direction and the true edge data are ε_i^* and θ_i^*, for all $i \in {}^{A}_{\overline{A}}$. The expected score is formulated as

$$\frac{1}{|I|} \sum_{\substack{\varepsilon_i^*, \theta_i^* \\ i \in I}} S(\varepsilon_i^*, \theta_i^*, \varepsilon_i, \theta_i : i \in I) P(\varepsilon_i^*, \theta_i^* : i \in I | E, \theta)$$

$$= \frac{1}{|I|} [\sum_{\substack{\varepsilon_i^*, \theta_i^* \\ i \in A}} S(\varepsilon_i^*, \theta_i^*, \varepsilon_i =' E', \theta_i : i \in A) P(\varepsilon_i^*, \theta_i^* : i \in A | E, \theta)$$

$$+ \sum_{\substack{\varepsilon_i^*, \theta_i^* \\ i \in \overline{A}}} S(\varepsilon_i^*, \theta_i^*, \varepsilon_i =' N', \theta_i : i \in \overline{A}) P(\varepsilon_i^*, \theta_i^* : i \in \overline{A} | E, \theta)] \quad (V-1)$$

An edge operator which has highest expected score for a given image is considered as the best operator for this image.

The score function we use here computes the score realized when the observed edge data is ε_i, θ_i and the true edge data is $\varepsilon_i^*, \theta_i^*$ as the sum of the scores realized on a pixel by pixel basis. That is,

$$S(\varepsilon_i^*, \theta_i^*, \varepsilon_i, \theta_i : i \in I)$$

$$= \sum_{\substack{\varepsilon_i^*, \theta_i^* \\ i \in I}} S(\varepsilon_i^*, \theta_i^*, \varepsilon_i, \theta_i) \quad (V-2)$$

In the case that the score function is defined by (V-2), (V-1) becomes

$$\frac{1}{|I|} [\sum_{\substack{\varepsilon_i^*, \theta_i^* \\ i \in A}} S(\varepsilon_i^*, \theta_i^*, \varepsilon_i =' E', \theta_i) P(\varepsilon_i^*, \theta_i^* | E, \theta)$$

$$+ \sum_{\substack{\varepsilon_i^*, \theta_i^* \\ i \in \overline{A}}} S(\varepsilon_i^*, \theta_i^*, \varepsilon_i =' N', \theta_i) P(\varepsilon_i^*, \theta_i^* | E, \theta)] \quad (V-3)$$

For edge detection it is reasonable to use a zero-one score function for each pixel in (V-3). When the pixel's observed edge data and true edge data are identical, the score is one. If they are different, the score is zero. Then (V-3) becomes

$$\frac{1}{|I|} [\sum_{i \in A} P(\varepsilon_i^* =' E', \theta_i^* = \theta_i | E, \theta)$$

$$+\sum_{j\in\overline{A}} P(\varepsilon_j^* =' N', \theta_j^* = \theta_j|E,\theta)] \qquad (V-4)$$

Two conditional independence assumptions are applied to make the above probability $P(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i|E,\theta)$ only depend on the local neighborhood of i. The first assumption amounts to limited influence. It says that the true edge data at a pixel i only influences the observed edge data for pixels in the neighborhood of i and do not influences the observed edge data or pixels outside the neighborhood of i. Hence

$$P(\varepsilon_k, \theta_k : k \notin N(i)|\varepsilon_i^*, \theta_i^*) = P(\varepsilon_k, \theta_k : k \notin N(i)) \qquad (V-5)$$

The second conditional independance assumption amounts to noise independence. It states that given the true edge data at pixel i, the probability of jointly observing edge data for the neighborhood of i, $\varepsilon_j, \theta_j : j \in N(i)$ and for the mutually exclusive image of N(i), $\varepsilon_k, \theta_k : k \notin N(i)$ is equal to the conditional probability of $\varepsilon_j, \theta_j : j \in N(i)$ given $\varepsilon_i^*, \theta_i^*$ times the conditional probability of $\varepsilon_k, \theta_k : k \notin N(i)$ given $\varepsilon_i^*, \theta_i^*$:

$$P(\varepsilon_j, \theta_j : j \in N(i); \varepsilon_k, \theta_k : k \notin N(i)|\varepsilon_i^*, \theta_i^*) =$$

$$P(\varepsilon_j, \theta_j : j \in N(i)|\varepsilon_i^*, \theta_i^*)P(\varepsilon_k, \theta_k : k \notin N(i)|\varepsilon_i^*, \theta_i^*) \qquad (V-6)$$

Putting together the two conditional independence assumptions and after some mathemtical manipulations we have

$$P(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i|E,\theta) =$$

$$P(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i | \varepsilon_j, \theta_j : j \in N(i)) \qquad (V-7)$$

Thus (V-4) becomes

$$\frac{1}{|I|}[\sum_{i \in A} P(\varepsilon_i^* =' E', \theta_i^* = \theta_i | \varepsilon_j, \theta_j : j \in N(i))$$

$$+ \sum_{j \in \overline{A}} P(\varepsilon_j^* =' N', \theta_j^* = \theta_j | \varepsilon_k, \theta_k : k \in N(j))] \qquad (V-8)$$

The probability $P(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i | \varepsilon_j, \theta_j : j \in N(i))$ of (V-8) is the conditional probability of $\varepsilon_i^* = \varepsilon_i$ and the true edge direction is the observed edge direction given all the edge context of the neighborhood of i. We organize the neighborhood context of i in a way similar to what we defined in the previous chapter. The definitions of Z_i, U_i, L_i, U_i^* and L_i^* are similar to their definations in that chapter. The only difference is that we are now only concerning with the local neighborhood of i not the entire image. We define

$$q_{Z_i}(\varepsilon_i^*, \theta_i^*) =$$

$$\max_{T \in Z_i} P(\varepsilon_i^*, \theta_i^* | \varepsilon_j, \theta_j : j \in T) \qquad (V-9)$$

where T is the best local row monotonically increasing path in the local neighborhood taking the direction θ_i^* through i. Then (V-8) becomes

$$\frac{1}{|I|} \sum_{i \in I} q_{Z_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i)$$

$$= \frac{1}{|I|} \sum_{i \in I} \max_{T \in Z_i} P(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i | \varepsilon_j, \theta_j : j \in T) \qquad (V-10)$$

By the definition of the conditional probability

$$q_{Z_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) =$$

$$\max_{T \in Z_i} P(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i | \varepsilon_j, \theta_j : j \in T) =$$

$$\max_{T \in Z_i} \sum_{\substack{\varepsilon_j^*, \theta_j^* \\ j \in T^-}} P(\varepsilon_k^*, \theta_k^* : k \in T | \varepsilon_l, \theta_l : l \in T) =$$

$$\max_{T \in Z_i} \sum_{\substack{\varepsilon_j^*, \theta_j^* \\ j \in T^-}} P(\varepsilon_l, \theta_l : l \in T | \varepsilon_k^*, \theta_k^* : k \in T)$$

$$* \frac{P(\varepsilon_k^*, \theta_k^* : k \in T)}{P(\varepsilon_l, \theta_l : l \in T)} \qquad (V-11)$$

where T^- designates the set of all pixels in T but pixel i. By making similar assumptions as we made on chapter II and from *Appendix 3*, (V-11) can be decomposed into

$$q_{Z_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) = \max_{T \in Z_i} \sum_{\substack{\theta_j^*, \varepsilon_j^* \\ j \in T^-}} \prod_{j \in T} \frac{P(\varepsilon_j, \theta_j | \varepsilon_j^*, \theta_j^*)}{P(\varepsilon_j, \theta_j)}$$

$$* \prod_{(i,j) \in R(T)} a(\varepsilon_i^*, \theta_i^*, \varepsilon_j^*, \theta_j^*) \qquad (V-12)$$

Since Z_i can be decomposed as the joint of U_i and L_i, this results in

$$q_{Z_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) = \max_{T_1 \in U_i} \max_{T_2 \in L_i} \sum_{\substack{\theta_j^*, \varepsilon_j^* \\ j \in T_1^-}} \sum_{\substack{\theta_j^*, \varepsilon_j^* \\ j \in T_2^-}}$$

$$\prod_{j\in T_1}\frac{P(\varepsilon_j,\theta_j|\varepsilon_j^*,\theta_j^*)}{P(\varepsilon_j,\theta_j)}\prod_{j\in T_2}\frac{P(\varepsilon_j,\theta_j|\varepsilon_j^*,\theta_j^*)}{P(\varepsilon_j,\theta_j)}\frac{P(\varepsilon_i,\theta_i)}{P(\varepsilon_i,\theta_i|\varepsilon_i^*,\theta_i^*)}$$

$$* \prod_{(i,j)\in R(T_1)} a(\varepsilon_i^*,\theta_i^*,\varepsilon_j^*,\theta_j^*)$$

$$* \prod_{(i,j)\in R(T_2)} a(\varepsilon_i^*,\theta_i^*,\varepsilon_j^*,\theta_j^*) \qquad (V-13)$$

Rearranging (V-13) we can group all expressions involving T_1 together and all expressions involving T_2 together and we obtain

$$q_{Z_i}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i)=$$

$$\frac{P(\varepsilon_i,\theta_i)}{P(\varepsilon_i,\theta_i|\varepsilon_i^*,\theta_i^*)}g_{U_i}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i)*g_{L_i}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i) \quad (V-14)$$

where

$$g_{U_i}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i)=\max_{T\in U_i}P(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i|\varepsilon_j,\theta_j:j\in T)$$

$$=\max_{T\in U_i}\sum_{\substack{\theta_j^*,\varepsilon_j^*\\ j\in T^-}}P(\varepsilon_j^*,\theta_j^*:j\in T|\varepsilon_j,\theta_j:j\in T)$$

$$=\max_{T\in U_i}\sum_{\substack{\theta_j^*,\varepsilon_j^*\\ j\in T^-}}\prod_{j\in T}\frac{P(\varepsilon_j,\theta_j|\varepsilon_j^*,\theta_j^*)}{P(\varepsilon_j,\theta_j)}$$

$$* \prod_{(i,j)\in R(T)} a(\varepsilon_i^*,\theta_i^*,\varepsilon_j^*,\theta_j^*) \qquad (V-15)$$

and the definition of $g_{L_i}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i)$, $h_{U_i^*}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i)$ and $h_{L_i^*}(\varepsilon_i^*=\varepsilon_i,\theta_i^*=\theta_i)$ are similar.

The edge probability $P(\varepsilon_j, \theta_j)$ can be computed as

$$P(\varepsilon_j, \theta_j) = \int_{\theta_j^*} \sum_{\varepsilon_j^*} P(\varepsilon_j, \theta_j, \varepsilon_j^*, \theta_j^*) d\theta_j^*$$

$$= \int_{\theta_j^*} \sum_{\varepsilon_j^*} P(\varepsilon_j, \theta_j | \varepsilon_j^*, \theta_j^*) P(\theta_j^* | \varepsilon_j^*) P(\varepsilon_j^*) d\theta_j^*$$

$$= \int_{\theta_j^*} P(\varepsilon_j, \theta_j | \varepsilon_j^* =' E', \theta_j^*) P(\theta_j^* | \varepsilon_j^* =' E') P(\varepsilon_j^* =' E') d\theta_j^*$$

$$+ \int_{\theta_j^*} P(\varepsilon_j, \theta_j | \varepsilon_j^* =' N', \theta_j^*) P(\theta_j^* | \varepsilon_j^* =' N') P(\varepsilon_j^* =' N') d\theta_j^* \quad (V-16)$$

We assume that there is no favorite edge direction for either edge or no-edge pixel. Thus

$$P(\theta_j^* | \varepsilon_j^* =' E') = P(\theta_j^* | \varepsilon_j^* =' N') = \frac{1}{2\pi} \quad (V-17)$$

and (V-16) becomes

$$\frac{1}{2\pi} \int_{\theta_j^*} P(\varepsilon_j, \theta_j | \varepsilon_j^* =' E', \theta_j^*) P(\varepsilon_j^* =' E') d\theta_j^*$$

$$+ \frac{1}{2\pi} \int_{\theta_j^*} P(\varepsilon_j, \theta_j | \varepsilon_j^* =' N', \theta_j^*) P(\varepsilon_j^* =' N') d\theta_j^* \quad (V-18)$$

An implicit assumption has been made in Kitchen and Rosenfeld's approcah. The assumption is that for the pixels which have observed edge state 'edge', the probability of the true edge state 'edge' is approximately 1. Hence (V-18) becomes

$$P(\varepsilon_j =' E', \theta_j) =$$

$$\frac{1}{2\pi}\int_{\theta_j^*} P(\varepsilon_j =' E', \theta_j | \varepsilon_j^* =' E', \theta_j^*) d\theta_j^* \qquad (V-19)$$

Although the assumption of observed edge implying true edge is not exactly true, this is the best value we can give for edge detector evaluation. Because it is not feasible to give some prior value of how good certain edge detector is and then evaluate how good the operator is based on the prior value. Otherwise, the evaluation will be biased. Since the edge angle probability is a function of the true edge angle θ_j^* and it has maximum value when $\theta_j^* = \theta_j$. Thus, it is convenient to pick up a function for the probability. such that

$$\int_{\theta_j^*} P(\varepsilon_j =' E', \theta_j | \varepsilon_j^* =' E', \theta_j^*) d\theta_j^*$$

$$= 2\pi P(\varepsilon_j =' E', \theta_j | \varepsilon_j^* =' E', \theta_j^* = \theta_j)$$

Hence,(V-19) becomes

$$P(\varepsilon_j =' E', \theta_j) = P(\varepsilon_j =' E', \theta_j | \varepsilon_j^* =' E', \theta_j^* = \theta_j)$$

Thus

$$\frac{P(\varepsilon_j =' E', \theta_j | \varepsilon_j^* =' E', \theta_j^* = \theta_j)}{P(\varepsilon_j =' E', \theta_j)} = 1$$

Similarly, we can have

$$\frac{P(\varepsilon_j =' N', \theta_j | \varepsilon_j^* =' N', \theta_j^* = \theta_j)}{P(\varepsilon_j =' N', \theta_j)} = 1 \qquad (V-20)$$

Therefore

$$g_{U_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) = \max_{T \in U_i}$$

$$\prod_{(i,j)\in R(T)} a(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i, \varepsilon_j^* = \varepsilon_j, \theta_j^* = \theta_j) \qquad (V-22)$$

Thus, we can use the observed edge data of both the center pixel and its neighborhood to estimate the local edge coherence instead of using the true edge data which are not in general available.

It is noted that Kitchen and Rosenfeld directly used observed edge data for edge detector evaluation. While, they did not mention in their paper about the assumptions behind it. It is also noted that Kitchen and Rosenfeld's scheme considers only local edge coherence of the pixels which have observed edge state = 'edge' and ignore the pixels which have observed edge state = 'no-edge'. This is the reason why they need the thinness measure to have a balance between the continuation measure and the number of edge pixels. Since if they used the continuation measure alone ($\gamma = 1.0$) the edge score will reach a maximum value on an edge image which selects quite a good set of edge pixels and these edges are several pixels thick. However, when they used the thinness measure it will bias against ideal step edges. Thus, some improvement need to be done by either considering the coherence of the non-edge pixels and ignore the thinness measure or using an improved version of the thinness measure which does not bias against ideal step edge.

Since the constraints set on the coherence between local non-edge pixels is much weaker than the coherence constraints set on the edge direction relationship between local edge pixels. Thus, unless for a particular set of images the non-edge coherence is very strong and we know this coherence very well, we can not expect to have a fair edge operator evaluation based on the local non-edge coherence. In the following derivations we consider only the local coherence of edge pixels. However, it is easy to extend the derivations to include all the non-edge coherence in the edge evaluation for certain edge images. The thinness measure we used which will be discribed in the next section is much more robust than what Kitchen and Rosenfeld used.

Putting (V-20) and (V-21) into (V-14) we have

$$q_{Z_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) =$$

$$g_{U_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) * g_{L_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i)$$

$$= \max_{T_1 \in U_i} \prod_{(i,j) \in R(T_1)} a(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i, \varepsilon_j^* = \varepsilon_j, \theta_j^* = \theta_j)$$

$$* \max_{T_2 \in L_i} \prod_{(i,j) \in R(T_2)} a(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i, \varepsilon_j^* = \varepsilon_j, \theta_j^* = \theta_j) \qquad (V-22)$$

By a similar derivation as in *chapter IV*. Both $g_{U_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i)$ and $g_{L_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i)$ can be obtained by a dynamic programming technique carried out in the local neighborhood around pixel i. Assume the row column coordinate of i is (r,c), then

$$g_{U_{rc}}(\varepsilon_r^* c, \theta_r^* c) = \max\{$$

$$g_{U_{r,c-1}}(\varepsilon^*_{r,c-1} = \varepsilon_{r,c-1}, \theta^*_{r,c-1} = \theta_{r,c-1}) * a(\varepsilon^*_{r,c-1}, \theta^*_{r,c-1}, \varepsilon^*_r c, \theta^*_r c),$$

$$h_{U^*_{r-1,c-1}}(\varepsilon^*_{r-1,c-1} = \varepsilon_{r-1,c-1}, \theta^*_{r-1,c-1} = \theta_{r-1,c-1})$$

$$*a(\varepsilon^*_{r-1,c-1}, \theta^*_{r-1,c-1}, \varepsilon^*_r c, \theta^*_r c),$$

$$h_{U^*_{r-1,c}}(\varepsilon^*_{r-1,c} = \varepsilon_{r-1,c}, \theta^*_{r-1,c} = \theta_{r-1,c}) * a(\varepsilon^*_{r-1,c}, \theta^*_{r-1,c}, \varepsilon^*_r c, \theta^*_r c),$$

$$h_{U^*_{r-1,c+1}}(\varepsilon^*_{r-1,c+1} = \varepsilon_{r-1,c+1}, \theta^*_{r-1,c+1} = \theta_{r-1,c+1})$$

$$*a(\varepsilon^*_{r-1,c+1}, \theta^*_{r-1,c+1}, \varepsilon^*_r c, \theta^*_r c)\} \qquad (V-23)$$

and

$$h_{U^*_{rc}}(\varepsilon^*_r c, \theta^*_r c) = \max\{$$

$$g_{U_{rc}}(\varepsilon^*_r c, \theta^*_r c), h_{U^*_{r,c+1}}(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1}) *$$

$$a(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1}, \varepsilon^*_r c, \theta^*_r c)\} \qquad (V-24)$$

Similarly, we have

$$g_{L_{rc}}(\varepsilon^*_r c, \theta^*_r c) = \max\{$$

$$g_{L_{r,c+1}}(\varepsilon^*_{r,c+1} = \varepsilon_{r,c+1}, \theta^*_{r,c+1} = \theta_{r,c+1}) * a(\varepsilon^*_{r,c+1}, \theta^*_{r,c+1}, \varepsilon^*_r c, \theta^*_r c),$$

$$h_{L^*_{r+1,c+1}}(\varepsilon^*_{r+1,c+1} = \varepsilon_{r+1,c+1}, \theta^*_{r+1,c+1} = \theta_{r+1,c+1})$$

$$*a(\varepsilon^*_{r+1,c+1}, \theta^*_{r+1,c+1}, \varepsilon^*_r c, \theta^*_r c),$$

$$h_{L^*_{r+1,c}}(\varepsilon^*_{r+1,c} = \varepsilon_{r+1,c}, \theta^*_{r+1,c} = \theta_{r+1,c}) * a(\varepsilon^*_{r+1,c}, \theta^*_{r+1,c}, \varepsilon^*_r c, \theta^*_r c),$$

$$h_{L^*_{r+1,c-1}}(\varepsilon^*_{r+1,c-1} = \varepsilon_{r+1,c-1}, \theta^*_{r+1,c-1} = \theta_{r+1,c-1})$$

$$*a(\varepsilon^*_{r+1,c-1}, \theta^*_{r+1,c-1}, \varepsilon^*_r c, \theta^*_r c)\} \qquad (V-25)$$

and

$$h_{L^*_{rc}}(\varepsilon^*_r c, \theta^*_r c) = \max\{$$

$$g_{L_{rc}}(\varepsilon^*_r c, \theta^*_r c), h_{L^*_{r,c-1}}(\varepsilon^*_{r,c-1}, \theta^*_{r,c-1})*$$

$$a(\varepsilon^*_{r,c-1}, \theta^*_{r,c-1}, \varepsilon^*_r c, \theta^*_r c)\} \qquad (V-26)$$

The scheme of Kitchen and Rosenfeld is just a special case of this scheme when the neighborhood size is selected as 3 X 3 and $a(\varepsilon^*_i =' E', \theta^*_i = \theta_i, \varepsilon^*_j =' E', \theta^*_j = \theta_j)$ is defined as

$$b(\theta_i, \theta_j) * b(\frac{\pi K}{4}, \theta^*_i + \frac{\pi}{2})$$

where

$$b(\alpha, \beta) = \frac{\pi - |\alpha - \beta|}{\pi}$$

and K is the adjacent edge position index with respect to the center pixel according to the following order:

3	2	1
4	0	0
5	6	7

We follow the function defined in chapter II. Due to the low cumulative curvature requirement of an edge line in a small neighborhood, it is reasonable to assume that $a(\varepsilon^*_0, \theta^*_0, \varepsilon^*_1, \theta^*_1)$ has maximal value when the edge direction at the immediate adjacent neighbor agrees with that at the center. And the expected neighbor direction based on this relative position agrees with the neighbor direction. To satisfy these requirements, we define

$$a(\varepsilon_0^* =' edge', \theta_0^*, \varepsilon_1^* =' edge', \theta_1^*) =$$

$$\frac{d(\theta_0^*, \theta_1^*) + d_1(\theta_0^*, \frac{M\pi}{4})}{2} \qquad (V-27)$$

where d and d_1 are defined as

$$d(\alpha, \beta) = \frac{\cos(\alpha - \beta) + 1}{2}$$

$$d_1(\alpha, \beta) = \frac{\cos 2(\alpha - \beta) + 1}{2} \qquad (V-28)$$

and M is the adjacent edge position index with respect to the center pixel according to the following order

5	4	3
6	0	2
7	8	1

The center position is the position of pixel 1, different position of pixel 0 corresponds to different M values.

The range of function d and d_1 is the closed interval [0, 1]. The reason to select this nonlinear function for edge consistent function is that as α approaches β the function has less penality (higher value) than the absolute difference function which computes absolute difference between two angles. Conversely, it gives more penality when the angle difference is large. Thus the angle quantization effect due to the rectangular grid layout of the pixels will tend to be minimized.

V-3. Edge correctness and thinning measures

As we mentioned before Kitchen and Rosenfeld's edge evaluator disregards the correct location of the edges. An edge detector that systematically mislocated edges will receive an evaluation measure equal to that of a detector which perfectly locates edges. To resolve this problem, a new scheme is proposed in this section. This scheme considers both the observe edge data and the original greyscale image. The idea is that an edge point is considered as having been correctly detected if the mean of the image region to the left (with respect to edge direction) of the point is quite different from the mean of the image region to the right of the point. And the left and right regions are by themselves homogeneous. We obtained the means and variances from the greyscale image which has been processed by the given edge detector. Suppose the left region and right region of an edge pixel have greyscale means and variances (μ_1, σ_1^2) and (μ_2, σ_2^2), respectively. Then the features μ and σ should satisfy the following criterion

$$\max(\sigma_1^2, \sigma_2^2) < \frac{(\mu_1 - \mu_2)^2}{k} \qquad (V-29)$$

where K is a selected constant represents the prior knowledge about the minimum signal to noise ratio of any homogeneous regions of the given image. Thus, an edge point should have the difference between its left region mean and right region mean significantly greater than the maximum possible difference caused by noise. As we define in (V-9) and (V-22)

$$q_{Z_i}(\varepsilon_i^*, \theta_i^*) =$$

$$\max_{T \in Z_i} P(\varepsilon_i^*, \theta_i^* | \varepsilon_j, \theta_j : j \in T)$$

$$= g_{U_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i) * g_{L_i}(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i)$$

$$= \max_{T_1 \in U_i} \prod_{(i,j) \in R(T_1)} a(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i, \varepsilon_j^* = \varepsilon_j, \theta_j^* = \theta_j)$$

$$* \max_{T_2 \in L_i} \prod_{(i,j) \in R(T_2)} a(\varepsilon_i^* = \varepsilon_i, \theta_i^* = \theta_i, \varepsilon_j^* = \varepsilon_j, \theta_j^* = \theta_j)$$

Let T be the best row monotonically increasing path starting from an upper boundary and ending at a lower boundary of the neighborhood. Let T_u, *and* T_l be the best row monotonically increasing paths ending at pixel i and starting at pixel i, respectively. Then $T = T_u \cup T_l$ Let

$$r_{umin} = \min\{r \ ; \ \forall \ (r, c) \in T_u\}$$

and

$$c_{umin} = \min\{c \ ; \ \forall \ (r_{umin}, c) \in T_u\}$$

then $b_u = (r_{umin}, c_{umin}) \in T_u$ which is the starting point of the best path T_u. Similarly, we can define

$$r_{lmax} = \max\{r \ ; \ \forall \ (r, c) \in T_l\}$$

$$c_{lmax} = \max\{c \; ; \; \forall \; (r_{lmax}, c) \in T_l\}$$

and $b_l = (r_{lmax}, c_{lmax}) \in T_l$ which is the ending point of the best path T_l.

Let the row column coordinate of the pixel i be (r_i, c_i). We define vectors $\vec{u}$ and $\vec{l}$ as

$$\vec{u} = b_u - i = (r_{umin} - r_i, c_{umin} - c_i)$$

$$\vec{l} = b_l - i = (r_{lmax} - r_i, c_{lmax} - c_i) \qquad (V-30)$$

Let θ_u be the angle between $\vec{u}$ and the column axis $\vec{C}$. We define the set T_u^1 as

$$T_u^1 = \begin{array}{ll} \{(r+1, c) | \forall (r, c) \in T_u\} & if \; 0^\circ \leq \theta_u \leq 22.5^\circ \\ \{(r+1, c+1) | \forall (r, c) \in T_u\} & if \; 22.5^\circ < \theta_u \leq 67.5^\circ \\ \{(r, c+1) | \forall (r, c) \in T_u\} & if \; 67.5^\circ < \theta_u \leq 112.5^\circ \\ \{(r-1, c+1) | \forall (r, c) \in T_u\} & if \; 112.5^\circ < \theta_u \leq 157.5^\circ \\ \{(r-1, c) | \forall (r, c) \in T_u\} & if \; 157.5^\circ < \theta_u \leq 180^\circ \end{array} \qquad (V-31)$$

as shown in *Figure V-1.* And

$$T_u^2 = \begin{array}{ll} \{(r+1, c) | \forall (r, c) \in T_u^1\} & if \; 0^\circ \leq \theta_u \leq 22.5^\circ \\ \{(r+1, c+1) | \forall (r, c) \in T_u^1\} & if \; 22.5^\circ < \theta_u \leq 67.5^\circ \\ \{(r, c+1) | \forall (r, c) \in T_u^1\} & if \; 67.5^\circ < \theta_u \leq 112.5^\circ \\ \{(r-1, c+1) | \forall (r, c) \in T_u^1\} & if \; 112.5^\circ < \theta_u \leq 157.5^\circ \\ \{(r-1, c) | \forall (r, c) \in T_u^1\} & if \; 157.5^\circ < \theta_u \leq 180^\circ \end{array}$$

The definitions of $T_u^3, T_u^4, \ldots$ are similar.

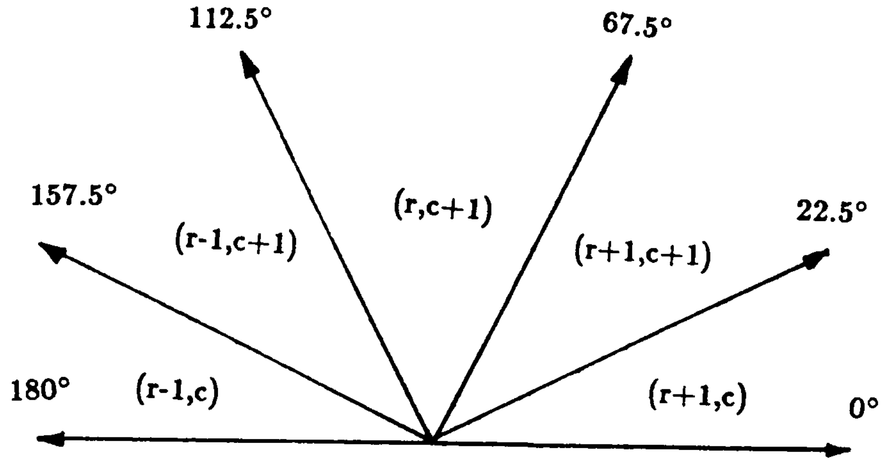


Figure V-1. Shows how the set T_u^{k+1} is constructed based on the angle θ_u.

In summary, let $T_u^0 = T_u$. then

$$T_u^{k+1} = \begin{array}{ll} \{(r+1,c)|\forall(r,c)\in T_u^k\} & if\ 0^\circ \le \theta_u \le 22.5^\circ \\ \{(r+1,c+1)|\forall(r,c)\in T_u^k\} & if\ 22.5^\circ < \theta_u \le 67.5^\circ \\ \{(r,c+1)|\forall(r,c)\in T_u^k\} & if\ 67.5^\circ < \theta_u \le 112.5^\circ \\ \{(r-1,c+1)|\forall(r,c)\in T_u^k\} & if\ 112.5^\circ < \theta_u \le 157.5^\circ \\ \{(r-1,c)|\forall(r,c)\in T_u^k\} & if\ 157.5^\circ < \theta_u \le 180^\circ \end{array} \qquad (V-32)$$

where $K \in$ integer I. Let θ_l be the angle between $\vec{l}$ and the column axis $\vec{c}$ and let $T_l^0 = T_l$. Then we can define T_l^{k+1} as

$$T_l^{k+1} = \begin{array}{ll} \{(r-1,c)|\forall(r,c)\in T_l^k\} & if\ 0^\circ \ge \theta_l \ge -22.5^\circ \\ \{(r-1,c+1)|\forall(r,c)\in T_l^k\} & if\ -22.5^\circ > \theta_l \ge -67.5^\circ \\ \{(r,c+1)|\forall(r,c)\in T_l^k\} & if\ -67.5^\circ > \theta_l \ge -112.5^\circ \\ \{(r+1,c+1)|\forall(r,c)\in T_l^k\} & if\ -112.5^\circ > \theta_l \ge -157.5^\circ \\ \{(r+1,c)|\forall(r,c)\in T_l^k\} & if\ -157.5^\circ > \theta_l \ge -180^\circ \end{array} \qquad (V-33)$$

where $K \in$ integer I. *Figure V-2* shows how the set T_l^{k+1} is constructed.

From (V-32), we have

$$T_u^{k-1} = \begin{array}{ll} \{(r-1,c)|\forall(r,c)\in T_u^k\} & if\ 0^\circ \le \theta_u \le 22.5^\circ \\ \{(r-1,c-1)|\forall(r,c)\in T_u^k\} & if\ 22.5^\circ < \theta_u \le 67.5^\circ \\ \{(r,c-1)|\forall(r,c)\in T_u^k\} & if\ 67.5^\circ < \theta_u \le 112.5^\circ \\ \{(r+1,c-1)|\forall(r,c)\in T_u^k\} & if\ 112.5^\circ < \theta_u \le 157.5^\circ \\ \{(r+1,c)|\forall(r,c)\in T_u^k\} & if\ 157.5^\circ < \theta_u \le 180^\circ \end{array} \qquad (V-34)$$

Similarly, from (V-33), we have

$$T_l^{k-1} = \begin{array}{ll} \{(r+1,c)|\forall(r,c)\in T_l^k\} & if\ 0^\circ \ge \theta_l \ge -22.5^\circ \\ \{(r+1,c-1)|\forall(r,c)\in T_l^k\} & if\ -22.5^\circ > \theta_l \ge -67.5^\circ \\ \{(r,c-1)|\forall(r,c)\in T_l^k\} & if\ -67.5^\circ > \theta_l \ge -112.5^\circ \\ \{(r-1,c-1)|\forall(r,c)\in T_l^k\} & if\ -112.5^\circ > \theta_l \ge -157.5^\circ \\ \{(r-1,c)|\forall(r,c)\in T_l^k\} & if\ -157.5^\circ > \theta_l \ge -180^\circ \end{array} \qquad (V-35)$$

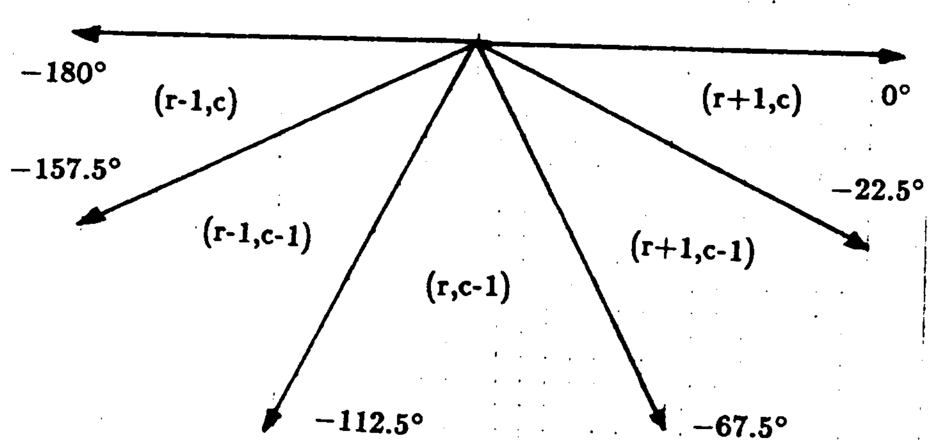


Figure V-2. Shows how the set T_l^{k+1} is constructed based on the angle θ_l.

Let $T^k = T_u^k \cup T_L^k,\ k \in I$. For a selected width d which is smaller than or equal to the minimum width of any meaniful region of the given image, the pixels belong to the set $\bigcup_{i=1}^{d} T^i$ and $\bigcup_{i=-1}^{-d} T^i$ construct the right and the left regions of the pixel i. If the observed edge state of i is 'edge', we will compute the means μ_1, μ_2 and variances σ_1^2, σ_1^2 of the right and left regions, respectively. The location accuracy measure *la* of the observed edge point can then be defined as

$$la = \begin{cases} 1 & if\ \max(\sigma_1^2, \sigma_2^2) \leq \frac{(\mu_1-\mu_2)^2}{k} \\ \frac{(\mu_1-\mu_2)^2}{k*\max(\sigma_1^2,\sigma_2^2)} & otherwise \end{cases} \qquad (V-36)$$

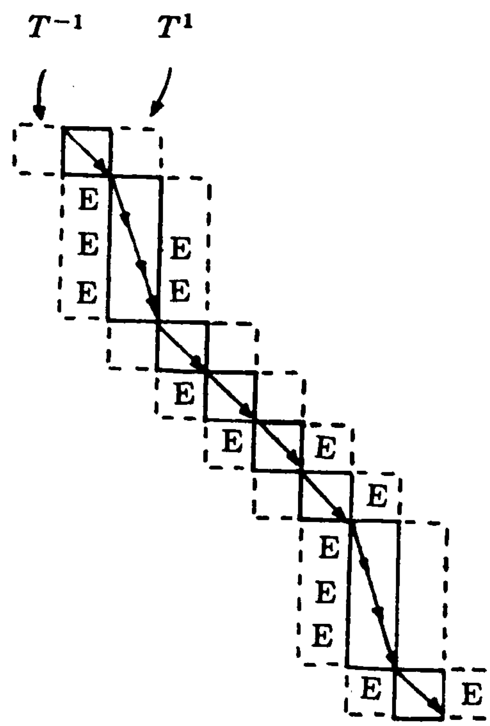
Such that if the square difference of the region means are larger than the given signal to noise ratio times the maximum of the standard deviation of the regions the *la* value will be one. Otherwise it will be less than one.

Let $|N_l|$ and $|N_r|$ designate the number of pixels immediately adjacent to the pixels belonging to the local optimal edge path T and which have observed edge state 'edge' on the left side and the right side of the optimal path T. Thus,

$$N_l = \{i | i \in T^{-1}\ and\ \varepsilon_i =' edge'\}$$

$$N_r = \{i | i \in T^{1}\ and\ \varepsilon_i =' edge'\}$$

and $|N_l|, |N_r|$ represents the number of elements belong to the set N_l and N_r. Let $|T^1|, |T^{-1}|$ represents the number of elements belong to the set T^1 and T^{-1}. *Figure V-3* shows the sets N_l, N_r, T^1, and T^{-1}.



$|N_l| = 9 \qquad |T^1| = 12$

$|N_r| = 5 \qquad |T^{-1}| = 12$

Figure V-3. shows the sets N_l, N_r, T^1, and T^{-1}.

A generalize thinness measure T which is applicable for any size of neighborhood is defined as:

$$T = 1 - \frac{|N_l| + |N_r|}{|T^1| + |T^{-1}|} \qquad (V-37)$$

or

$$T = 1 - \frac{2 * \min\{|N_l| + |N_r|\}}{|T^1| + |T^{-1}|} \qquad (V-38)$$

depends on whether we allow the existence of ideal step edge (two pixel width) or not (only allow single pixel width). We use *equation (V-38)* if we allow the existence of the ideal step edge. Otherwise, we use *equation (V-37)*.

The edge score based on local edge coherence (not include edge position correctness) can be a combination of the continuation measure C, obtained from *equation (V-10)* and the thinness measure T. Kitchen and Rosenfeld used a linear combination of the two measures and had the continuation be weight four times as the thinness weight. We propose a more robust way for the combination between C and T. The combination we defined is the same as what they used. Thus

$$E = \gamma(T)C + (1 - \gamma(T))T$$

However, the γ we used is a function of T. And it is defined as:

$$\gamma(T) = \begin{matrix} 0.2 & if \;\; 1 \geq T \geq 0.85 \\ 0.25 & if \;\; 0.85 > T \geq 0.75 \\ 0.3 & if \;\; 0.75 > T \geq 0.65 \end{matrix} \qquad (V-39)$$

Hence,we do not allow any edge image to have $T < 0.65$. We claim that for those edge images which have $T < 0.65$ the evaluation score is meaningless and the only comment we can make is they are bad edge images because once T is small the image has quite a lot redundant edges. And the edge evaluation scheme is not reliable for this kind of image. In Kitchen and Rosenfeld's experiment for edge detection they did not set the lower limit on T. Therefore, it happens that for the low SNR test images the edge evaluation scheme they used will either have little difference between the images of different SNR values or the edge score may be higher for some noisy images and have lower edge score for the images which have higher SNR.

V-4. Experimental results

We present some experiments to understand the performance of the general edge evaluator. To permit a comparison, we have tried to make our experimental setup as similar as possible to that of Kitchen and Rosenfeld. We used similar edge detection schemes and the same noise model and test images. However we use a 5 X 5 window for the edge evaluation instead of a 3 X 3 window.

Three well known edge detectors are used in the experiments. They are the Kirsch operator, the 3 X 3 Sobel operator and Neviatia's compass operator (see Neviatia and Babu (1980)). Two test images were used. The first one is an image of 64 X 64 pixels. It consisted of a left panel with grey level 115, a right panel with grey level 140, and a single central column of intermediate grey level 128. We call this image the "vertical edge" image. It is the same as the one used by both Abdou & Pratt and Kitchen & Rosenfeld. The second image consisting of concentric light rings (grey level 140) on a dark background (grey level 115). This image was originally generated as a 512 by 512 image with a central dark circle of radius 64, surrounded by three bright rings of width 32, these being separated by two dark rings of the same width, with a dark surround. The decision as to whether a pixel should be light or dark is based on its Euclidean distance from the center of the image. Then this image was reduced to size 128 by 128, by replacing each 4 by 4 block with a single pixel having the average grey level of the block. This image is the same as one of the test image used by Kitchen and Rosenfeld. From now on we call this image the "rings" image.

To study the effects of noise, independent zero-mean Gaussian noise was added to each of the test images at seven different signal to noise ratios: 1, 2, 5, 10, 20, 50, and 100. The signal-to-noise ratio (SNR) is defined as

$$SNR = \left(\frac{h}{\sigma}\right)^2$$

where h is the edge constrast (in this case 25), and σ is the standard deviation of the noise, adjusted to give the selected values of SNR. *Figure V-4 and V-5* shows these two test images and their noisy images with different SNR.

In the experiment, we applied each operator to the test images at the seven different signal-to-noise ratios, and at each noise level the threshold was adjusted to maximize the edge evaluation score E. The local neighborhood size for edge evaluation is selected as 5 X 5. The size is larger than Kitchen and Rosenfeld's scheme which can only deal with 3 X 3 neighborhood. *Figures V-6 and V-7* show the evaluation result for different edge operators.

For the vertical edge test image we allow only single pixel width edge and the edge evaluator is set up in a way that it performs the evaluation based on this single edge assumption. The results based on the local edge coherence measure give compass edge operator high score over the other two operators when the noise level is not too high. However, once noise is very high the compass operator becomes the worst operator on this image. This is due to the fact that the Nevitia's compass edge detection scheme performs not only edge detection but also edge line linking. Thus, when the image is not too noisy the edge linking process can improve the edge image. However, if the image is too noisy, the edge linking process tends to link wrong edge lines and the edge image becomes even worse than the raw edge

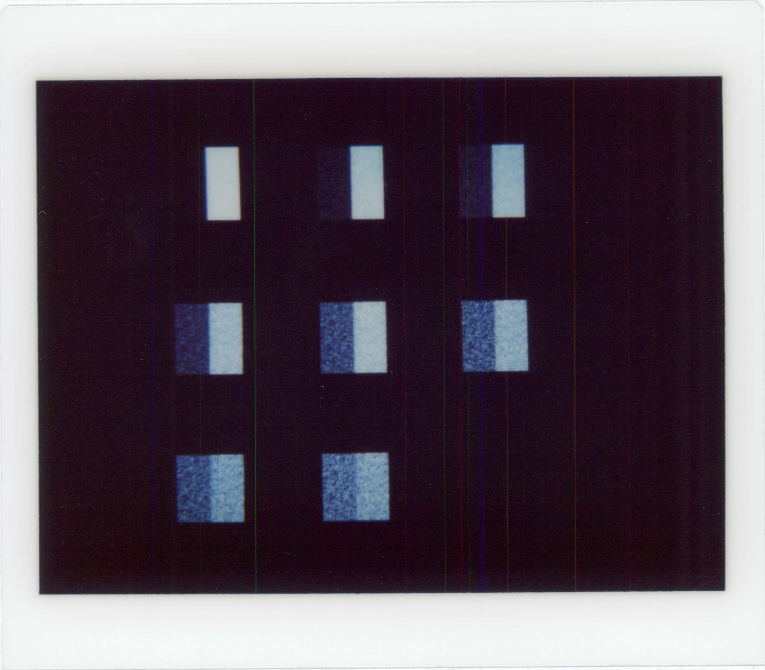


Figure V-4. Vertical edge test image, with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

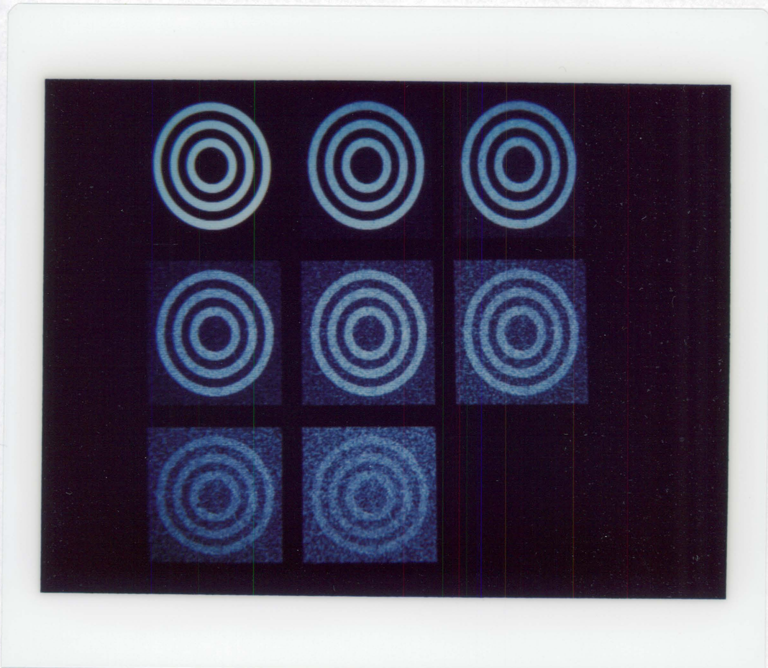


Figure V-5. Rings test image, with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

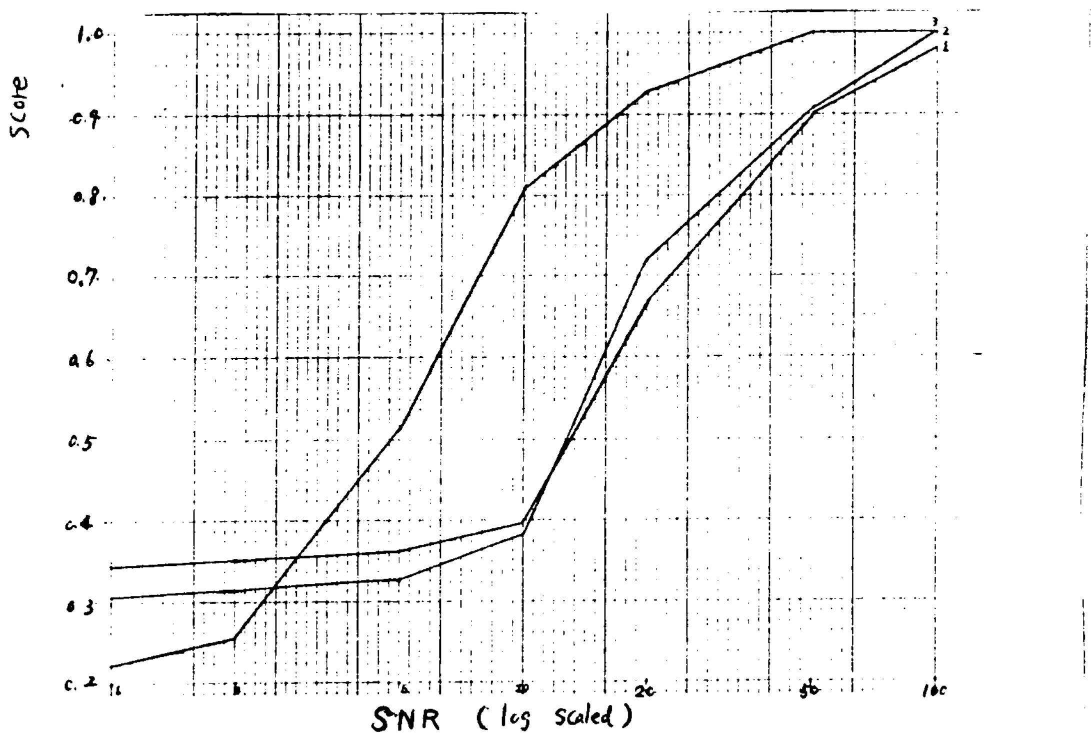


Figure V-6. The edge coherence score for different edge operators. The test image is the vertical edge image. where 1: for Sobel operator 2: for Kirsch operator and 3: for Nevitia's compass operator.

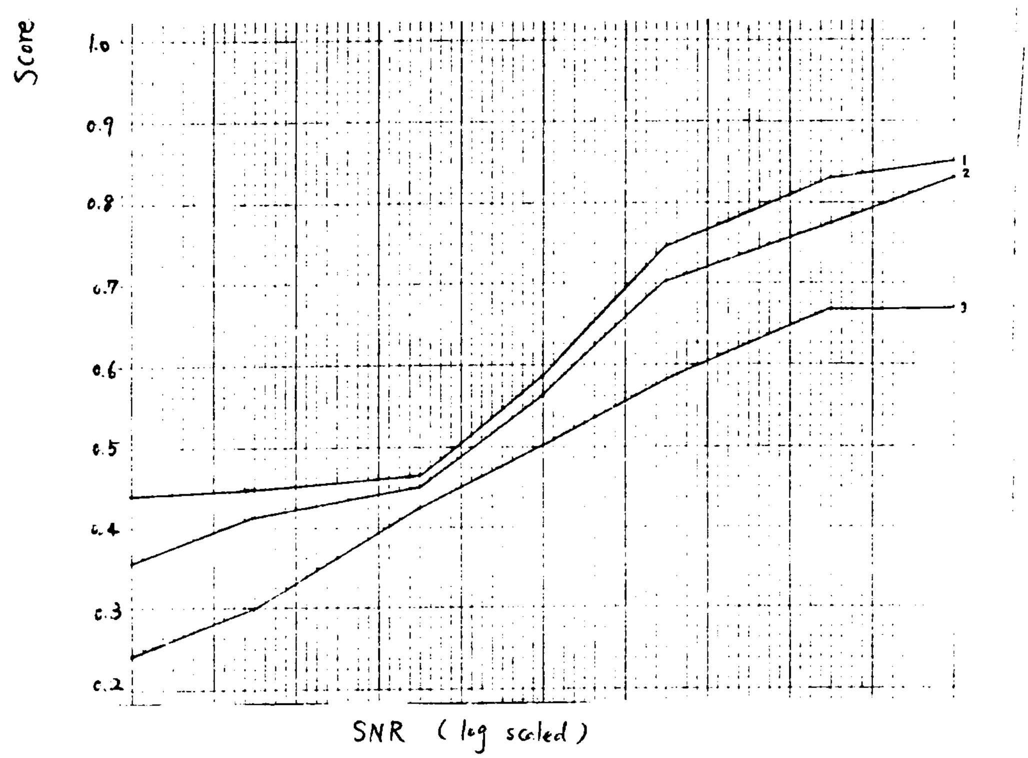


Figure V-7. The edge coherence score for different edge operators. The test image is the ring image. where 1: for Sobel operator 2: for Kirsch operator and 3: for Nevitia's compass operator.

image. The Kirsch operator has better performance compared with the Sobel operator when the SNR is higher than 10. Its performance becomes worse than the Sobel operator when the SNR is greater than 10. It is noted that the performance score curves of these operators for the low SNR images are not as flat as the curves of Kitchen and Rosenfeld's evaluator(see *Figure V-6*). This is because that we have a lower bound for the thinness measure T. The edge images of these operators on the vertical edge image are shown in *Figures V-8, V-9 and V-10*,respectively. It is noted that a visual evaluation of these edge images is consistent with what we obtained quantitively based on the evaluator.

The result of the edge position correctness measure for different operators are listed in *Tables V-1 to V-3*. The table shows that for this particular image the edge correctness measure is consistent with the edge coherence measure. Any combination of them will yield almost equivalent evaluation score of each operator.

For the rings test image we allow the existence of double pixel width edge and the edge evaluator is set up to accept double width edge lines when performing the evaluation . The results based on the local edge coherence measure on this image give different evaluations for each operator compared with the evaluation based on the vertical edge images. On this image the Sobel operator has uniformly best

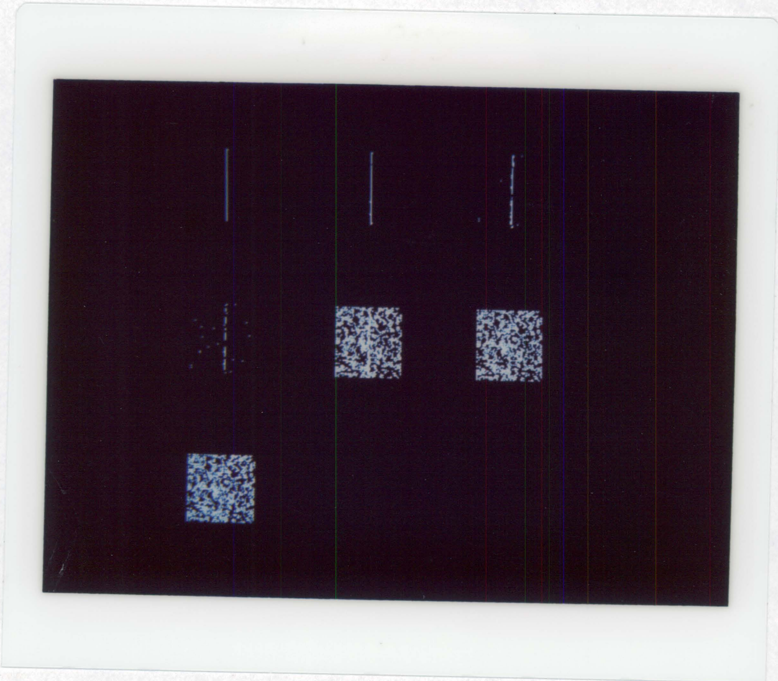


Figure V-8. edge results from Sobel operator for the vertical edge test image with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

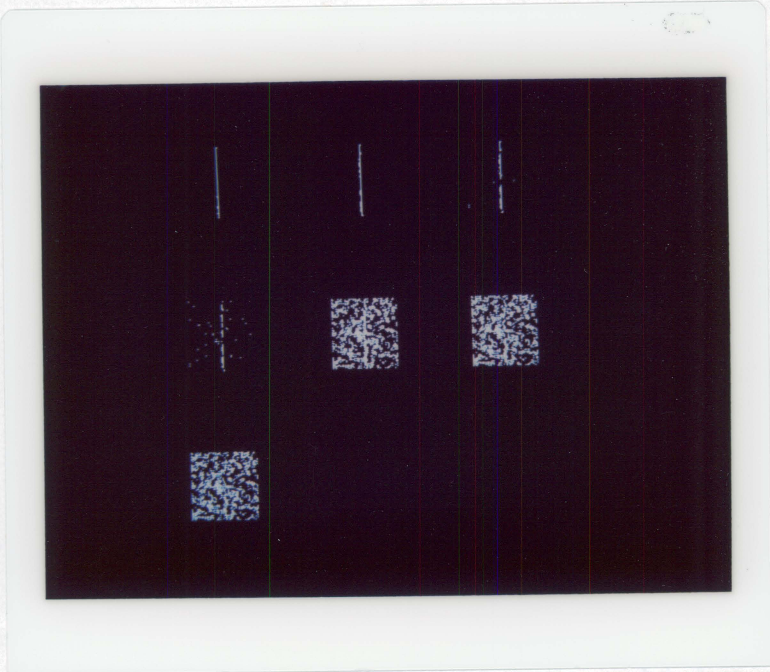


Figure V-9. edge results from Kirsch operator for the vertical edge test image with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

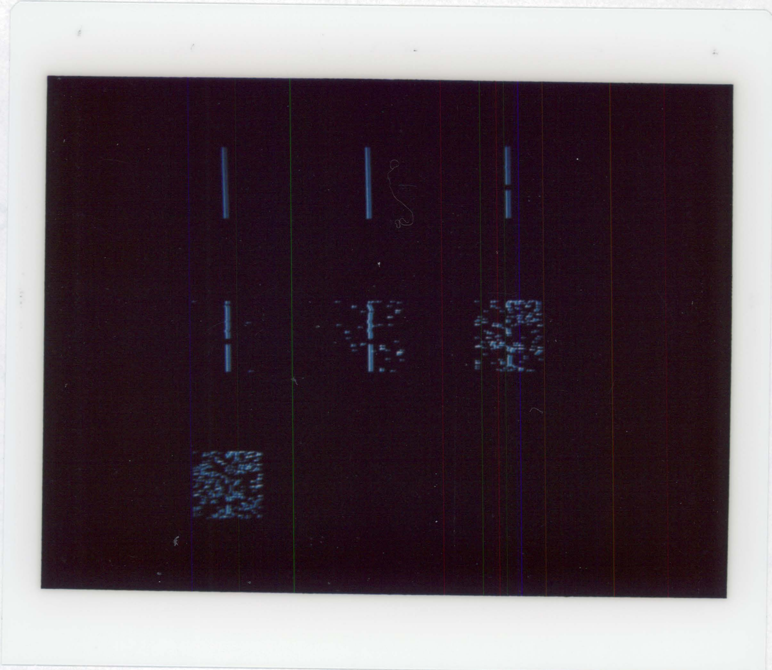


Figure V-10. edge results from Nevitia's compass operator for the vertical edge test image with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

Table V-1. The edge accuracy measure for the Sobel edge operator on the vertical edge image.

SNR \ result	*k input*	*accuracy*
100	5	0.9771
50	5	0.9243
20	3	0.7612
10	2	0.4897
05	1	0.4157
02	1	0.3761
01	1	0.3550

SNR \result	*k input*	*accuracy*
100	5	0.9754
50	5	0.8993
20	3	0.7913
10	2	0.4532
05	1	0.4027
02	1	0.3562
01	1	0.3378

Table V-3. The edge accuracy measure for the compass edge operator on the vertical edge image.

SNR \result	*k input*	*accuracy*
100	5	1.0000
50	5	0.9859
20	3	0.9310
10	2	0.8586
05	1	0.7032
02	1	0.4837
01	1	0.4247

score. The Nevatia's compass operator has worst score and the Kirsch operator has performance in between. The reason that the compass operator has low evaluation score on this image is partially because that the edge linking process provided by the compass operator always try to detect single width edge lines although they are not there. Besides, the edge direction between any adjacent edge points are not exactly equivalent. Thus, the edge linking process on the rings image can not have as good performance as it is applied on the vertical edge image. It is also noted that the performance score curves of these operators for the low SNR images are not as flat as the curves of Kitchen and Rosenfeld's evaluator. This is because that we have set the lower bound on the thinness measure T. The edge images of these operators on the rings image are shown in *Figures V-11, V-12 and V-13.*

The result of edge position correctness measure for different operators are listed in *Tables V-4 to V-6.* The tables show that if we were to combine the edge correctness measure with the edge corherence measure, we will increase the score of the compass operator. This is because that the performance of the compass operator is not really as bad as it appears in the edge coherence score. The reason why it appears bad in the edge coherence measure is because this operator tries to detect single edge line while double edge lines are acceptable

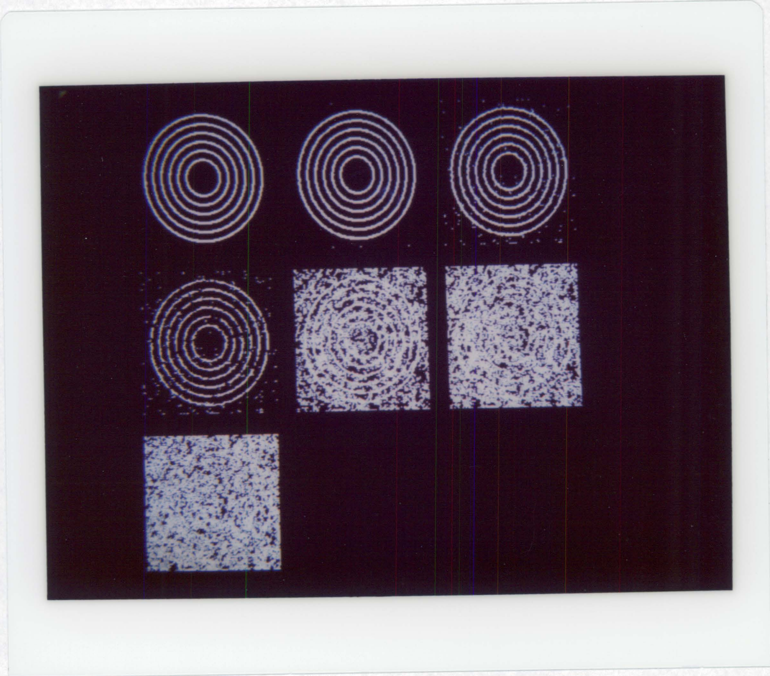


Figure V-11. edge results from Sobel operator for the rings test image with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

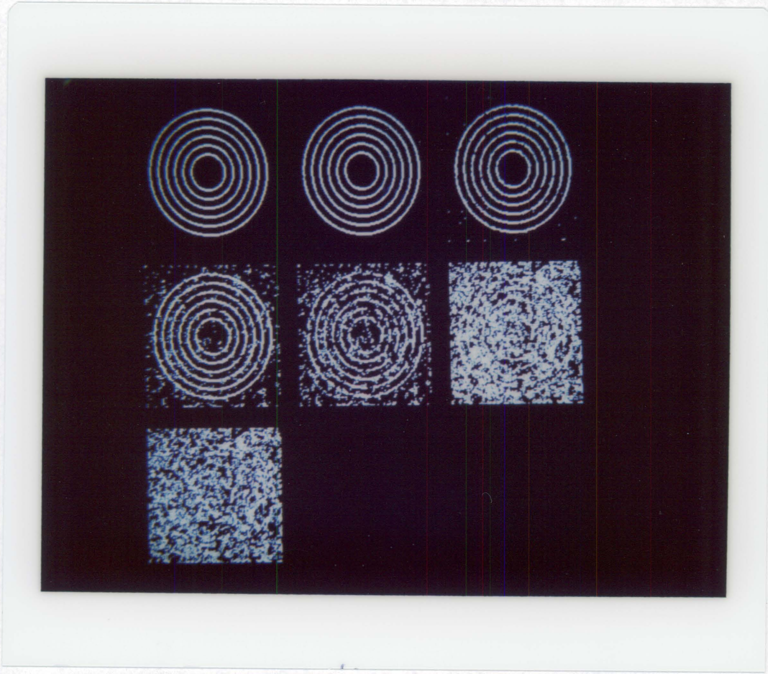


Figure V-12. edge results from Kirsch operator for the rings test image with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

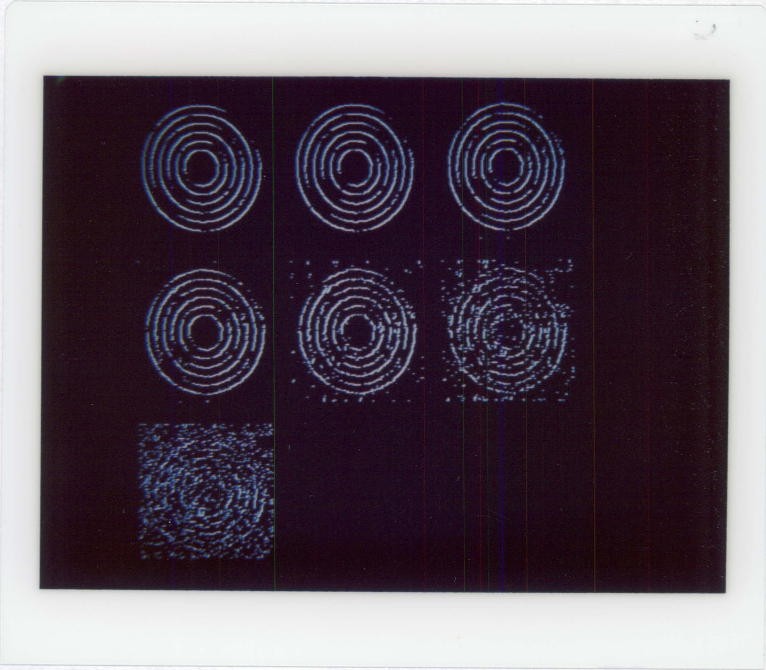


Figure V-13. edge results from Nevitia's compass operator for the rings test image with various levels of noise. From left to right, top to bottom: no noise, SNR 100, 50, 20, 10, 5, 2, and 1.

Table V-4. The edge accuracy measure for the Sobel edge operator on the rings image.

SNR \ result	*k input*	*accuracy*
100	5	0.9073
50	5	0.8833
20	3	0.8507
10	2	0.7438
05	1	0.5704
02	1	0.4589
01	1	0.3957

Table V-5. The edge accuracy measure for the Kirsch edge operator on the rings image.

SNR \result	*k input*	*accuracy*
100	5	0.9199
50	5	0.8964
20	3	0.8648
10	2	0.7082
05	1	0.6016
02	1	0.4608
01	1	0.3958

Table V-6. The edge accuracy measure for the compass edge operator on the rings image.

SNR \result	*k input*	*accuracy*
100	5	0.8049
50	5	0.7908
20	3	0.7674
10	2	0.7433
05	1	0.7541
02	1	0.5924
01	1	0.4746

to the coherence measure. Thus, a combination of the two measures can really have more reliable edge score.

In order to see this edge evaluator applies on the context edge operator and the context free second derivative zero-crossing edge operator. We apply the full context edge operator which is described in chapter III, the local context edge oparetor with 5 X 5 context neighborhood size(see chapter II), and the context free second derivative zero-crossing operators (Haralick, 1984) on a set of noisy images with added noise have standard deviation 10, 20, 30, 40, and 50, respectively. Thus, we can not only see the performance of the edge evaluator but also see the performance of the context edge operators against context free edge operator under different noise level. The adjustable parameters of each operator are adjusted so that they can have highest edge score for a given image.

The edge results which maximize the edge score are shown in *Figure V-14.* It is easy to verify that by a visual evaluation both context edge operators have better performance over the context free edge operator. The full context edge operator and the local context edge operator have similar performance when the noise is small. When the noise increases, the full context edge operator tends to produce edge images of better edge connectivity and less noisy while the edge lines are thicker than the edge lines produced by the local context edge


10
20
30

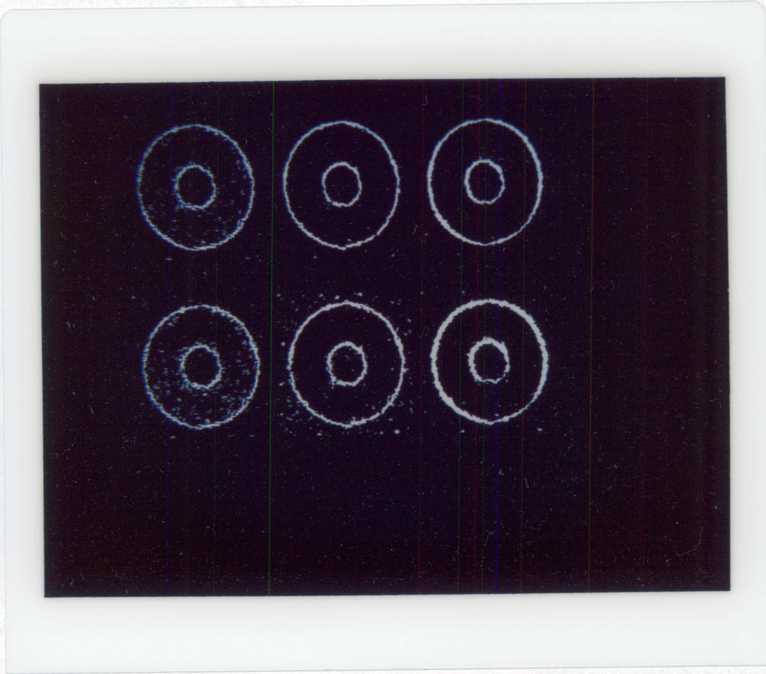


Figure V-14. Shows the edge images of the full context edge operator, local context edge operator with 5 X 5 context neighborhood size, and the context free edge operator applied on a set of noisy images with noise standard deviations 10, 20, 30, 40, and 50, respectively.

operator. The local context edge operator tends to produce broken edge lines and has more noise points than the full context edge images while it produces thin edge lines. These results consistent with what we expect on context. The full context edge operator has stronger context influence on the image. Thus, when the image is noisy the effect of context tends to fill the gaps between edge lines and at the same time merge the adjacent pixels of edge lines. Besides, most of the noise points are clean out by context. The local context edge operator has weaker context influence on the noisy image. Thus, when the image is noisy, the context effect can not fill large gaps between edge lines and it does not unselectively merge pixels adjacent to an edge line. While, it can not clean out as many noise points as full context edge operator does.

The edge score curves for different noise level are shown in *Figure V-15*. It is found that the edge scores is consistent with the visual evaluations. It can reflect very well what we have seen from the image. From the edge score curves we can find that the context operators always have better performance on all five noise levels. As the noise increases, the difference between the edge score increases and then when noise has standard deviation greater than 40 it decreases. The maximum of the score difference occurs when the noise standard deviation equals to 40. When the noise is small, the full context edge operator and local context edge operator have similar edge scores. As the noisy level is greater than 30, the edge score difference is significant and the difference keeps steadily at this value. It is noted that the full context edge operator has higher edge score than the local context edge operator in all the noise levels.

By a visual evaluation, we find that as the noise level increases. The full context operator tends to detect thick edge lines which have good connectivity and less noisy. The thinness measure alone is shown in *Figure V-16.* It is consistent with the visual evaluations. Finally, the location accuracy measure of these operators on the noisy images are shown in *Figure V-17.* It is found that just as we expected the context operators have better score over the context free operator. And the local context edge operator has better location accuracy when the noise level is greater than 30.

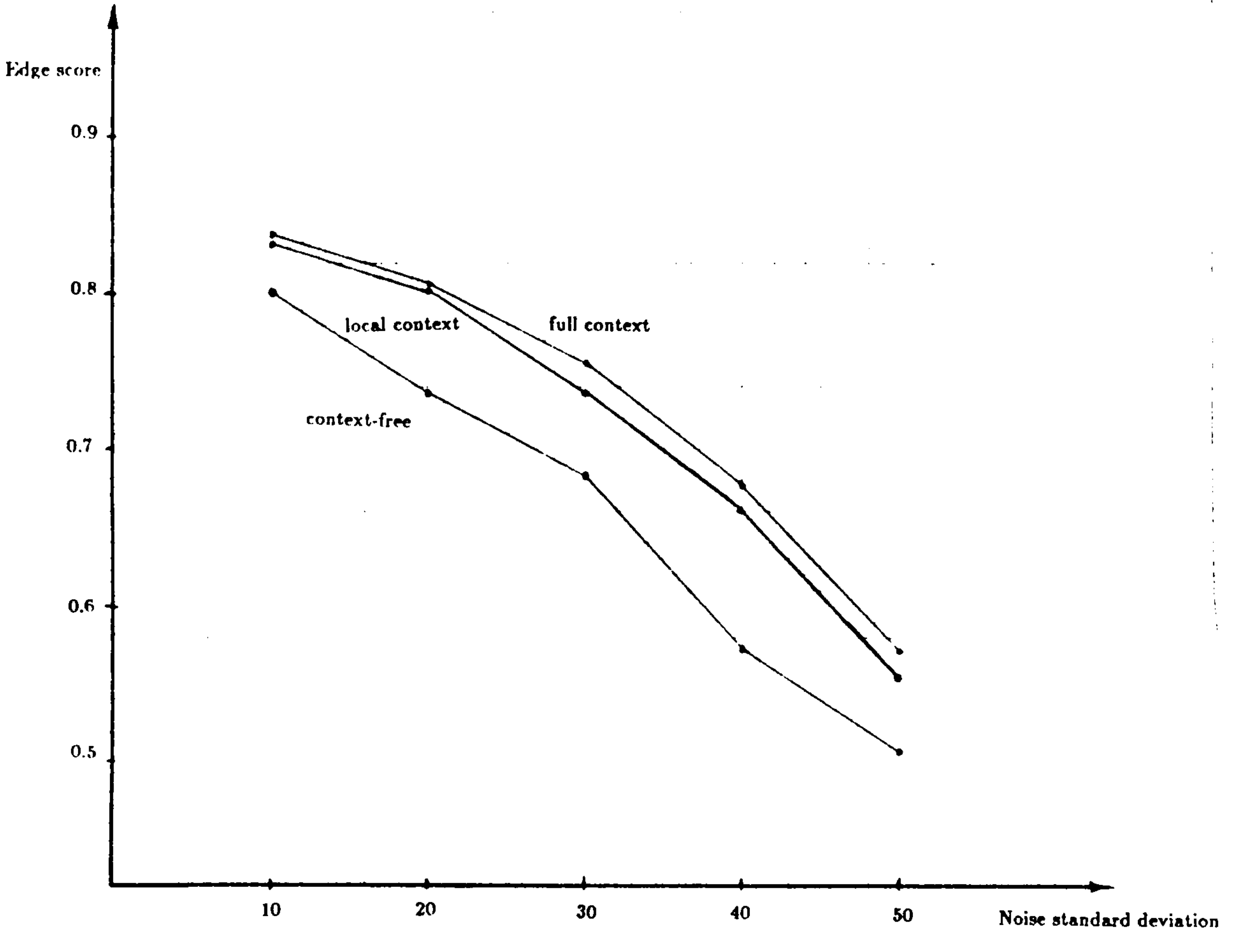


Figure V-15. Shows the edge scores of the context edge operators and the context free edge operator applied on a set of noisy images with noise standard deviations 10, 20, 30, 40, and 50, respectively.

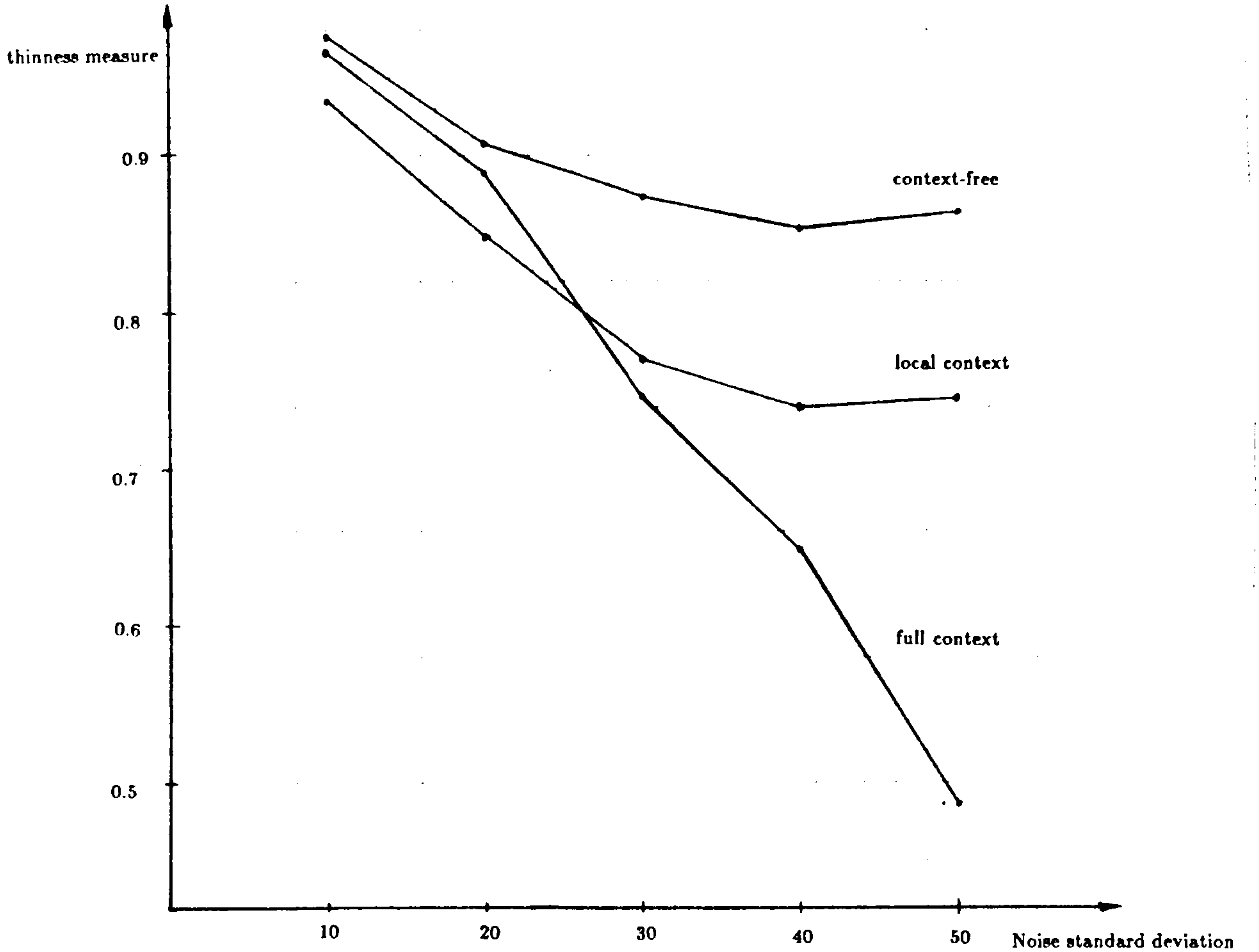


Figure V-16. Shows the edge thinness measures of the context edge operators and the context free edge operator applied on a set of noisy images with noise standard deviations 10, 20, 30, 40, and 50, respectively.

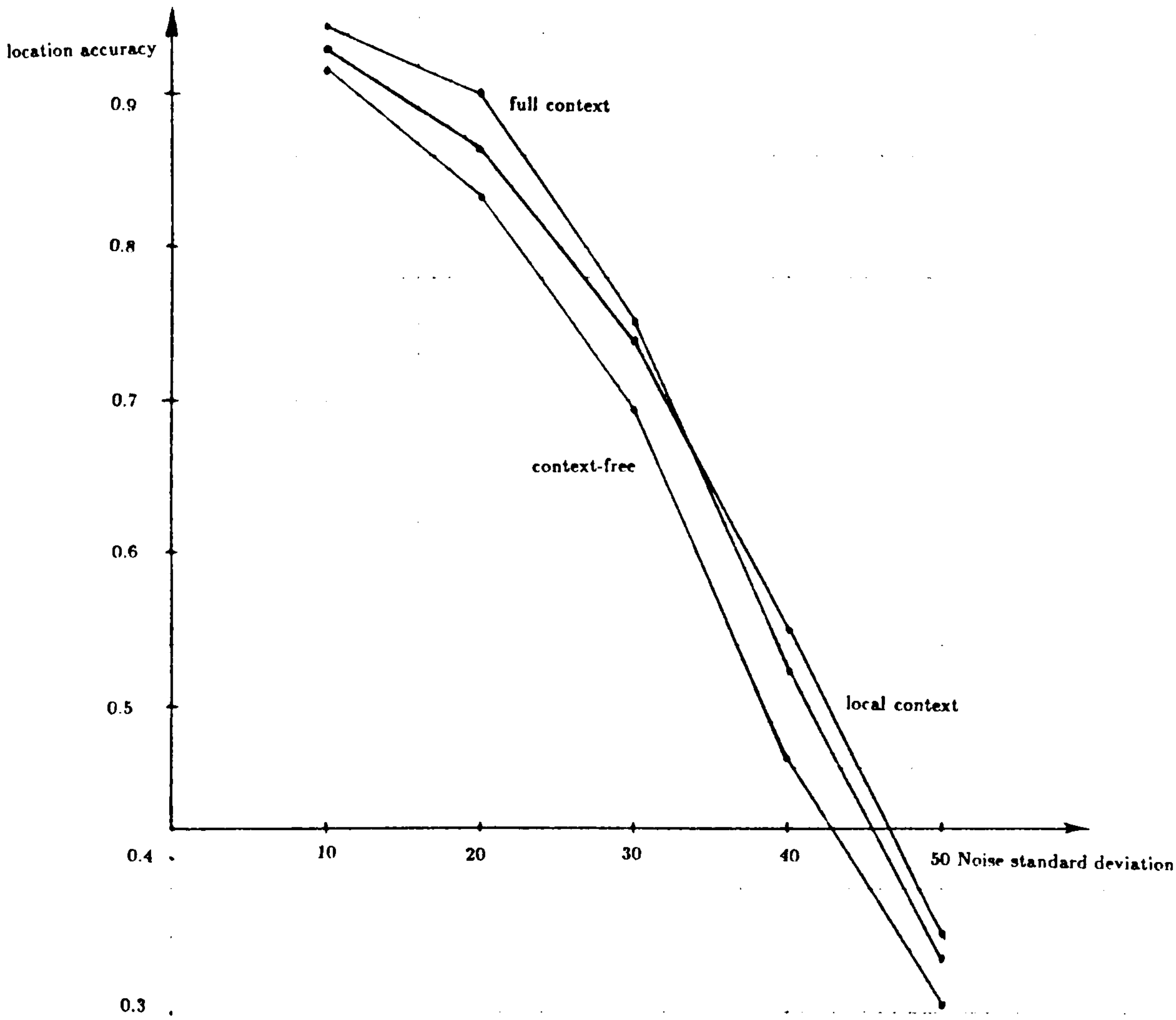


Figure V-17. shows the edge location accuracy of the context edge operators and the context free edge operator applied on a set of noisy images with noise standard deviations 10, 20, 30, 40, and 50, respectively.

VI. MORPHOLOGIC EDGE DETECTION

VI-1. INTRODUCTION

Mathematical morphology is an approach to image processing based on set theoretic concepts of shape. The early works include Dineen (1955), Kirsch (1957), Preston (1961, 1971, 1983), Landsmon (1965), Moore (1966,1968), and Golay (1969,1984). It was formalized at the Ećole de Mine in Paris in the mid 1970's by G. Matheron (1975), and expended by J. Serra (1982) and S. Sternburg (1980). It has grown to envelop a variety of applications and hardwares.

Several companies manufacturing machine vision hardware successfully use mathematical morphology to solve industrial machine vision problems (MVI, 1984). The machines they produce can deal with morphological operations more efficiently than other operations. Judging by the scientific archival literature, the techniques of mathematical morphology seem to be less used and less explored by the academic research communities.

Mathematical morphology can take full advantage of the specific shapes of the target objects. It always assumes that we have certain prior knowledge about the shapes of the target objects and can select in advance the most appropriate structuring elements to fit these shapes. In these applications it does not necessarily have to depend on the local properties of each small neighborhood of the image. However, as the problems we confront become more and more complicated because different views of the same object may have to be processed, mathematical morphology alone cannot fully handle these problems efficiently. A combination of mathematical morphology with other approaches such as statistical pattern recognition, structural pattern recognition, two-dimensional signal processing and methods of conventional image processing will undoubtedly be appropriate.

One class of approaches to computer vision depends heavily on good image segmentations determined by region growing and/or edge operators. A general edge detection scheme which can extract good enough arc and region information for the matching process of structural pattern recognition or the connectivity and feature analysis used by statistical pattern recognition is the basic requirement of a good conventional vision system.

This section explores the capability of morphology to perform edge detection. As far as we can determine, the image processing literature does not discuss any morphological edge detector.

VI-2. Basic Morphologic Operators

An image can be represented by a set of pixels. The morphologic operators can be thought to work with two images: the original data to be analyzed and a structuring element image which analogous to the kernel in convolution operation. Each structuring element has a shape which can be thought of as a parameter to the operators.

First we consider the case of binary images. Let A be the set of points representing the original binary image and B be the set of points representing a stucturing element. The *dilation* of A by B, denoted $A \oplus B$, is defined by

$$A \oplus B = \bigcup_{a \in A} \{b + a | b \in B\}$$

The *erosion* of A by B, denoted $A \ominus B$, is defined by

$$A \ominus B \ = \{p | B + p \subseteq A\}$$

The extensions of the morphologic transformations from binary into greyscale processing by S. Sternberg (1980,1983,1984) in the mid 1980's introduced a natural morphologic generalization of the dilation and erosion operations.

The dilation of a greyscale image f by a greyscale structuring element b is denoted by d, and is defined by

$$d(r, c) = \max_{i,j}(f(r - i, c - j) + b(i, j))$$

where the maximum is taken over all (i, j) in the domain of b. The erosion of a greyscale image f by a structuring element b is denoted by e and is defined by

$$e(r,c) = \min_{i,j}(f(r+i,c+j) - b(i,j))$$

where the minimum is taken over all (i,j) in the domain of b.

The algorithms of mathematical morphology combine sequences of dilations and erosions, and the residues of them in a way which often produces useful and pleasantly surprising results.

VI-3. The Simple Morphologic Edge Detectors

The purpose of this section is to explore some simple morphologic edge detectors which do not work in order that we provide some understanding of morphologic operators and provide the basis for those discussed in section IV which do work. A simple method of performing gray scale edge detection in a morphology based vision system is to take the difference between an image and its erosion by a small structuring element. The difference image is the image of edge strength. We can then select an appropriate threshold value to threshold the edge strength image into a binary edge image.

Let's denote the center of a local neighborhood by (0,0) and a point which is δr apart from the center in row direction and δc apart from the center in column direction by $(\delta r, \delta c)$. We define rod as a greyscale structuring element which is flat on top and has disk like domain. As an example, the domain of the structuring element rod radius 1 is defined by the set

$$D_{rod1} = \{(0,-1),(0,1),(-1,0),(1,0)\}$$

Let $b\colon D_{rod1} \rightarrow \{0,\ldots,255\}$ be the rod1 structuring element. Since a rod is flat on top, the grayscale value of all the $b(r,c)$, $(r,c) \in D_{rod1}$ is 0.

The erosion of a grayscale image f(r,c) by the structuring element rod 1 can be carried out by the rule:

$$e(r,c) = \min_{(i,j)\in D_{rod1}} [f(r+i,c+j) - b(r,c)]$$

which in the case of a zero height structuring element becomes

$$e(r,c) = \min_{(i,j)\in D_{rod1}} [f(r+i,c+j)] \qquad (VI-1)$$

The erosion residue edge detectior produce the edge strength image G_e defined by

$$\begin{aligned} G_e(r,c) &= f(r,c) - e(r,c) \\ &= f(r,c) - \min_{(i,j)\in D_{rod1}} f(r+i,c+j) \\ &= \max_{(i,j)\in D_{rod1}} [f(r,c) - f(r+i,c+j)] \qquad (VI-2) \end{aligned}$$

Since D_{rod1} includes exactly the four connected neighbors of position (0,0), the edge strength image we obtain is

$$G_e(r,c) = \max_{(i,j)\in N_4(r,c)} [f(r,c) - f(i,j)] \qquad (VI-3)$$

where $N_4(r,c)$ is the set of four connected neighbors of position (r,c).

We can now interpret the morphologic edge operator as a local neighborhood nonlinear operator which takes the maximum among the four first differences in directions $0°, 90°, 180°$, and $270°$.

A natural non-morphological variation of this operator takes the summation instead of maximization. This is the familar linear digital Laplacian operator $\nabla^2 f(r,c)$ (Rosenfeld et. al.,1982) which is the digital convolution of $f(r,c)$ with the kernel

$$\begin{matrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{matrix}$$

In order to compare the performance of the nonlinear morphological edge operator with the linear Laplacian operator, we apply them to four perfect digital step edge patterns of edge contrast E running in directions $0°, 90°, 45°$, and $135°$, respectively, i.e.

$$\begin{matrix} E & E & E \\ E & E & E \\ 0 & 0 & 0 \end{matrix} \quad \begin{matrix} E & E & 0 \\ E & E & 0 \\ E & E & 0 \end{matrix} \quad \begin{matrix} E & E & E \\ E & E & 0 \\ E & 0 & 0 \end{matrix} \quad \begin{matrix} E & E & E \\ 0 & E & E \\ 0 & 0 & E \end{matrix}$$
$$E_{0°} \qquad\qquad E_{90°} \qquad\qquad E_{45°} \qquad\qquad E_{135°}$$

The magnitude of the responses of G and $\nabla^2 f$ are as follows:

	$E_{0°}$	$E_{90°}$	$E_{45°}$	$E_{135°}$
G_e	E	E	E	E
$\nabla^2 f$	E	E	$2E$	$2E$

Thus the responses of $\nabla^2 f$ to these edges are E, E, 2E and 2E, a bias of 2 in favor of the diagonal edges. These biases are eliminated if we use the morphological operator G_e instead of $\nabla^2 f$. Besides G_e yields values in the same range as the original grayscale, which is most convenient on any computer vision system which has frame buffer limitations on the range of greyscale values.

Next we try both operators on a single noise point pattern with noise height h. i.e.

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

The responses of G_e and $\nabla^2 f$ are h and 4h respectively. Thus although both G_e and $\nabla^2 f$ are noise sensitive, the noise response of $\nabla^2 f$ is four times the response of G_e and hence four times the response of $\nabla^2 f$ on a vertical or horizontal ideal step edge with edge contrast h.

It is also possible to increase the neighborhood size of the morphologic edge operator by increasing the size of the structuring element used on the erosion operation. For example, we can have an 8-connected neighborhood edge operator by changing the structuring element to be flat on top and have domain

$$D_{8-connected} = \{(-1,-1),(0,-1),(1,-1),$$

$$(-1,0),(0,0),(1,0),(-1,1),(0,1),(1,1)\} \qquad (VI-4)$$

The edge strength image we obtain is then

$$G_{e\ 8-connected}(r,c) = \max_{(i,j)\in N_8(r,c)} [f(r,c) - f(i,j)] \qquad (VI-5)$$

where $N_8(r,c)$ is the set of eight connected neighbors of image position (r,c).

The corresponding linear Laplacian operator which has eight connected neighborhood support can be implemented as the digital convolution of $f(r,c)$ with

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

We now apply both operators in the four perfect digital step edge patterns E_{0°,E_{90°, E_{45°, and E_{135°. The magnitude of the responses of G_e and$\nabla^2 f$ are as follows:

	E_{0°	E_{90°	E_{45°	E_{135°
$G_{e\ 8-connected}$	E	E	E	E
$\nabla^2 f_{8-connected}$	$3E$	$3E$	$3E$	$3E$

It is found that by incresing the neighborhood size, the $\nabla^2 f$ operator now has uniform performance on these edges. The response of both operators on the single noise point pattern are h and 8h respectively. Thus, both operators are noise sensitive, the Laplacian being more noise sensitive. This explains why the raw Laplacian operator is not a good edge detector in noisy images. Laplacian edge operator generally low pass filter noisy images by a Gaussian filter and then apply a Laplacian operator. Edges are located at zero- crossings of the Laplacian (Marr et. al. 1980). However the Gaussian filter can shift the positions of most of the edges in real images.

The erosion residue morphological edge detector is a nonlinear Laplacian-like operator which is also noise sensitive, it cannot be a good edge detector for noisy images. The rule that increasing the neighborhood size of the operator will reduce the amount of noise fails with the erosion residue morphological edge detector. Consider, for example, the erosion residue morphological edge detector on the following image pattern

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

The pattern shown above is a flat area with pixel intensity F and a noise spike at the center of this area with pixel intensity 0. The response of the morphological edge operator is

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

which has the same value F for all the 8-connected neighbors of the center point. If we increase the size of the neighborhood support of the morphological operator, the number of pixels which are given a value F will also increase accordingly. As a matter of fact all the neighborhood support but the center point will be given the value F. Thus, the larger the operator's support, the noisier the operator's response can possibly be.

The erosion residue morphological edge detector is position biased. It only gives edge strength to border pixels on that side of the edge which have the higher value. For example, the operator only gives the inside boundary of the higher valued checkers of a perfect checkerboard image its corresponding edge strength and gives the outside boundary of the higher valued checkers edge strength zero.

To resolve this bias and give both inside and outside boundaries of the checkers their corresponding edge strengths, the dilation residue morphological edge detector can be used in conjunction with erosion residue operator. The dilation residue operator takes the difference between a dilated image and its original image. For example, if the structuring element for the dilation is a rod of radius 1, then the dilation of the grayscale image $f(r,c)$ is

$$d(r,c) = \max_{(i,j)\in D_{rod1}} [f(r-i,c-j)] \qquad (VI-6)$$

and the edge strength image is

$$G_d(r,c) = d(r,c) - f(r,c) =$$

$$\max_{(i,j)\in N_4(x,y)} [f(i,j) - f(r,c)] \qquad (VI-7)$$

It is obvious that this operator only gives edge strength to that side of the edge which has the lower value.

A position unbiased edge operator can be obtained by a combination of the operators $G_e(r,c)$ and $G_d(r,c)$ using the pixel-wise minimum, maximum, or sum. Let us consider the maximum first. Define

$$E(r,c) = \max(G_e(r,c), G_d(r,c))$$

$$= \max_{(i,j)\in N(r,c)} |f(r,c) - f(i,j)| \qquad (VI-8)$$

where $N(r,c)$ is the neighborhood support of the structuring element for both dilation and erosion operations.

To understand the performance of this operator, we apply it on four perfect digital step edge patterns of edge contrast E one ideal ramp edge pattern of edge contrast E and one single noise pattern of noise height N, i.e.,

$$\begin{array}{ccccc} E & E & E & E & E \\ E & E & E & E & E \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & E_{0^\circ} & & \end{array} \qquad \begin{array}{ccccc} E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ & & E_{90^\circ} & & \end{array}$$

$$\begin{array}{ccccc} E & E & E & E & E \\ E & E & E & E & 0 \\ E & E & E & 0 & 0 \\ E & E & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \\ & & E_{45^\circ} & & \end{array} \qquad \begin{array}{ccccc} E & E & E & E & E \\ 0 & E & E & E & E \\ 0 & 0 & E & E & E \\ 0 & 0 & 0 & E & E \\ 0 & 0 & 0 & 0 & E \\ & & E_{135^\circ} & & \end{array}$$

$$\begin{array}{ccccc} 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ & & E_{r0^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & noise & & \end{array}$$

The results of this edge operator when the neighborhood support is $N_4(r, c)$ are

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ & & E_{0^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ & & E_{90^\circ} & & \end{array}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & E \\ 0 & 0 & E & E & 0 \\ 0 & E & E & 0 & 0 \\ E & E & 0 & 0 & 0 \\ & & E_{45^\circ} & & \end{array} \qquad \begin{array}{ccccc} E & 0 & 0 & 0 & 0 \\ E & E & 0 & 0 & 0 \\ 0 & E & E & 0 & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & 0 & E & E \\ & & E_{135^\circ} & & \end{array}$$

$$\begin{array}{ccccc} 0 & \frac{E}{2} & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & \frac{E}{2} & 0 \\ & & E_{r0^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & N & N & N & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & noise & & \end{array}$$

Now the performance of the edge operator can be easily evaluated. This operator performs perfectly on ideal step edge patterns. However, it is noise sensitive. It responds with five noise edge points on a single noise pattern. It is even worse than that for an ideal ramp edge pattern. Ideally, the detected edge pattern for the ramp should have single pixel width edge line and its edge strength should be equal to the edge contrast. It detects edge lines giving them only half their edge contrast and the detected edge is wide, three pixels in width.

We now consider the edge operator which is defined as summation of $G_e(r,c)$ and $G_d(r,c)$. Thus

$$E(r,c) = G_e(r,c) + G_d(r,c) \qquad (VI-9)$$

To see the performance of this operator, we apply it on the edge patterns we used to test the maximum version edge operator. The results of this edge operator when the neighborhood support for dilation and erosion is $N_4(r,c)$ are

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \end{array}$$

$$E_{0^\circ} \qquad\qquad\qquad E_{90^\circ}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & E \\ 0 & 0 & E & E & 0 \\ 0 & E & E & 0 & 0 \\ E & E & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} E & 0 & 0 & 0 & 0 \\ E & E & 0 & 0 & 0 \\ 0 & E & E & 0 & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & 0 & E & E \end{array}$$

$$E_{45^\circ} \qquad\qquad\qquad E_{135^\circ}$$

$$\begin{array}{ccccc} 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ & & E_{r0^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & N & N & N & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & noise & & \end{array}$$

It is easy to see that just as the maximum version of this edge operator, the summation version performs perfectly on ideal step edge patterns. And it is also noise sensitive. It responds with five noise edge points on a single noise pattern. However, for the ideal ramp edge pattern it detects an edge line whose edge strength equals edge contrast and two lines on both sides of the edge line whose edge strength equals half edge contrast. Thus, by thresholding with a value greater than half edge contrast it is possible to have perfect performance on the ideal ramp edge pattern in the sense that it detects a single width edge line.

Finally we consider the edge operator which is defined as minimum of $G_e(r,c)$ and $G_d(r,c)$. Hence,

$$E(r,c) = \min(G_e(r,c), G_d(r,c)) \qquad (VI-10)$$

To understand the performance of this operator, we again apply it on the edge patterns we used to test the maximum and summation version edge operators. The result of this edge operator when the neighborhood support for dilation and erosion is $N_4(r,c)$ are

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & E_{0^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & E_{90^\circ} & & \end{array}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & E_{45^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & E_{135^\circ} & & \end{array}$$

$$\begin{array}{ccccc} 0 & 0 & \frac{E}{2} & 0 & 0 \\ 0 & 0 & \frac{E}{2} & 0 & 0 \\ 0 & 0 & \frac{E}{2} & 0 & 0 \\ 0 & 0 & \frac{E}{2} & 0 & 0 \\ 0 & 0 & \frac{E}{2} & 0 & 0 \\ & & E_{r0^\circ} & & \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ & & noise & & \end{array}$$

The results of this operator on these edge patterns are interesting. The performance on the ideal ramp edge pattern is promising. It can detect a single edge line with edge strength $\frac{E}{2}$. And it is noise insensitive. It has no response when applied to the single noise point. Unfortunately, it is not able to detect ideal step edge patterns. This motivates a new edge operator which first performs a blur operation to convert all the ideal step edges into ideal ramp edges and then applies the minimum version of edge operator on them. In the next section, we will analyze this blur-minimum operator in greater detail as well as some others.

VI-4. Morphologic Operators Which Work

As mentioned in the previous section, the simple morphological edge operators based on erosion or dilation residues are either sensitive to noise or can not detect ideal step edges. In this section we discuss a few more morphological edge operators which can detect ideal step edges and are not noise sensitive. They are the improved dilation and erosion residue operators and the blur-minimum operator.

VI-4-1. Improved residue operators

A. Improved dilation residue operator

In this section we introduce an improved version of the dilation residue operator $G_d(r,c)$. Let a_1, a_2, a_3, and a_4 be the structuring elements which are flat on top and have domains

$$D_{a1} = \{(-1,0),(0,1)\}$$

$$D_{a2} = \{(0,-1),(1,0)\}$$

$$D_{a3} = \{(-1,0),(0,-1)\}$$

$$D_{a4} = \{\ (0,1),(1,0)\ \}$$

and D_2 be a structuring element which is flat on top and has domain

$$\{(-1,-1),(-1,1),(1,-1),(1,1)\}$$

The operator is defined by

$$G'_d(r,c) = \min\{Dilation_{D_{rod1}}(r,c) - f(r,c),$$

$$Dilation_{D_2}(r,c) - f(r,c), G''_d(r,c)\} \qquad (VI-11)$$

where $G''_d(r,c)$ is defined by

$$G''_d(r,c) = \max\{$$

$$|(Dilation_{a1}(r,c) - f(r,c)) - (Dilation_{a2}(r,c) - f(r,c))|,$$

$$|((Dilation_{a3}(r,c) - f(r,c)) - (Dilation_{a4}(r,c) - f(r,c))|\}$$

To see the performance of this operator, we apply it on four perfect digital step edge patterns of edge contrast E, one ideal ramp edge pattern of edge contrast E, and two single noise patterns of noise height N, i.e.,

$$\begin{matrix} E & E & E & E & E \\ E & E & E & E & E \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \\ E & E & E & 0 & 0 \end{matrix}$$

$$E_{0^\circ} \qquad\qquad E_{90^\circ}$$

$$\begin{matrix} E & E & E & E & E \\ E & E & E & E & 0 \\ E & E & E & 0 & 0 \\ E & E & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} E & E & E & E & E \\ 0 & E & E & E & E \\ 0 & 0 & E & E & E \\ 0 & 0 & 0 & E & E \\ 0 & 0 & 0 & 0 & E \end{matrix}$$

$$E_{45^\circ} \qquad\qquad E_{135^\circ}$$

$$\begin{array}{ccccc} 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \\ 0 & 0 & \frac{E}{2} & E & E \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$

$$E_{r0^\circ} \qquad\qquad noise1$$

$$\begin{array}{ccccc} N & N & N & N & N \\ N & N & N & N & N \\ N & N & 0 & N & N \\ N & N & N & N & N \\ N & N & N & N & N \end{array}$$

$$noise2$$

The $Dilation_{D_{rod1}}(r,c) - f(r,c)$ operation results

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$

$$E_{0^\circ} \qquad\qquad E_{90^\circ}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$

$$E_{45^\circ} \qquad\qquad E_{135^\circ}$$

$$\begin{array}{ccccc} 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & N & 0 & N & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$

$$E_{r0^\circ} \qquad\qquad noise1$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$
$$noise2$$

And the results of $Dilation_{D_2}(r,c) - f(r,c)$ operation are

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$
$$E_{0^\circ} \qquad\qquad E_{90^\circ}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$
$$E_{45^\circ} \qquad\qquad E_{135^\circ}$$

$$\begin{array}{ccccc} 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & N & 0 & N & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & N & 0 & N & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$
$$E_{r0^\circ} \qquad\qquad noise1$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$
$$noise2$$

The results of $G_d''(r,c)$ will be

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$
$$E_{0^\circ} \qquad\qquad\qquad E_{90^\circ}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$
$$E_{45^\circ} \qquad\qquad\qquad E_{135^\circ}$$

$$\begin{array}{ccccc} 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & N & 0 & N & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$
$$E_{r0^\circ} \qquad\qquad\qquad noise1$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$
$$noise2$$

Finally, the results of $G_d'(r,c)$ will be

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$
$$E_{0^\circ} \qquad\qquad\qquad E_{90^\circ}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \end{array}$$

$$E_{45^\circ} \qquad\qquad\qquad E_{135^\circ}$$

$$\begin{array}{ccccc} 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \\ 0 & \frac{E}{2} & \frac{E}{2} & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$

$$E_{r0^\circ} \qquad\qquad\qquad noise1$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}$$

$$noise2$$

Thus, just as the simple dilation residue operator, it assigns edge strength E to all the edge pixels which are on the low value side of the ideal step edge and it detects two edge lines of half edge contrast from ideal ramp edge patterns. However, unlike the simple dilation residue operator it will not pick up a noise pixel. Hence, this is an improved version of the $G_d(r,c)$ operator. It is also noted that this operator does not need any blurring as a preprocessor.

B. Improved erosion residue operator

In this section we introduce an improved version of the erosion residue operator $G_e(r,c)$. Let a_1, a_2, a_3, a_4 and D_2 be the structuring elements which are defined in the previous section. Then the operator is defined by

$$G'_e(r,c) = \min\{f(r,c) - Erosion_{D_{rod1}}(r,c),$$

$$f(r,c) - Erosion_{D_2}(r,c), G''_e(r,c)\} \qquad (VI-12)$$

where $G''_e(r,c)$ is defined as

$$G''_e(r,c) = \max\{$$

$$|(f(r,c) - Erosion_{a1}(r,c)) - (f(r,c) - Erosion_{a2}(r,c))|,$$

$$|(f(r,c) - Erosion_{a3}(r,c)) - (f(r,c) - Erosion_{a4}(r,c))|\}$$

We apply this operator on the same edge and noise patterns as we used in the previous section. The results are

$$|(f(r,c) - Erosion_{a1}(r,c)) - (f(r,c) - Erosion_{a2}(r,c))| :$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \end{array}$$

$$E_{0°} \qquad\qquad E_{90°}$$

$$\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array} \qquad \begin{array}{ccccc} E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & 0 & E \end{array}$$

$$E_{45°} \qquad\qquad E_{135°}$$

$$\begin{matrix} 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

$$E_{r0^\circ} \qquad noise1$$

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

$$noise2$$

$$|(f(r,c) - Erosion_{a3}(r,c)) - (f(r,c) - Erosion_{a4}(r,c))| :$$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \end{matrix}$$

$$E_{0^\circ} \qquad E_{90^\circ}$$

$$\begin{matrix} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

$$E_{45^\circ} \qquad E_{135^\circ}$$

$$\begin{matrix} 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

$$E_{r0^\circ} \qquad noise1$$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & N & 0 & N & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$
$$noise2$$

The results of $\min\{f - Erosion_{D_{rod1}}, f - Erosion_{D_2}\}$ are

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \end{matrix}$$
$$E_{0^\circ} \qquad\qquad E_{90^\circ}$$

$$\begin{matrix} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & 0 & E \end{matrix}$$
$$E_{45^\circ} \qquad\qquad E_{135^\circ}$$

$$\begin{matrix} 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & N & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$
$$E_{r0^\circ} \qquad\qquad noise1$$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$
$$noise2$$

And finally the $G'_e(r,c)$ results are:

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & E & 0 & 0 \end{matrix}$$
$$E_{0^\circ} \qquad\qquad E_{90^\circ}$$

$$\begin{matrix} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ E & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} E & 0 & 0 & 0 & 0 \\ 0 & E & 0 & 0 & 0 \\ 0 & 0 & E & 0 & 0 \\ 0 & 0 & 0 & E & 0 \\ 0 & 0 & 0 & 0 & E \end{matrix}$$
$$E_{45^\circ} \qquad\qquad E_{135^\circ}$$

$$\begin{matrix} 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \\ 0 & 0 & \frac{E}{2} & \frac{E}{2} & 0 \end{matrix} \qquad \begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$
$$E_{r0^\circ} \qquad\qquad noise1$$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$
$$noise2$$

It is found that just as $G_e(r,c)$ operator, it assigns edge strength E to all the edge points which are on the high value side of the ideal step edge and it detects two edge lines of half edge contrast from ideal ramp edge patterns. However, unlike simple erosion residue operator, it will not pick up the noise pixel. Hence this is an improved version of the $G_e(r,c)$ operator.

C. Putting G'_e and G'_d together

It is natural that an improved version of the morphologic edge operator which has no position bias and is noise insensitive will be a combination of the improved dilation residue and erosion residue operator. Consider the sum of these two.

$$E'(r,c) = G'_e(r,c) + G'_d(r,c). \qquad (VI-13)$$

The results of this operator applied to the same test patterns as we used before are:

$$\underset{E_{0^\circ}}{\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ E & E & E & E & E \\ E & E & E & E & E \\ 0 & 0 & 0 & 0 & 0 \end{array}} \qquad \underset{E_{90^\circ}}{\begin{array}{ccccc} 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & E & E & 0 \end{array}}$$

$$\underset{E_{45^\circ}}{\begin{array}{ccccc} 0 & 0 & 0 & 0 & E \\ 0 & 0 & 0 & E & E \\ 0 & 0 & E & E & 0 \\ 0 & E & E & 0 & 0 \\ E & E & 0 & 0 & 0 \end{array}} \qquad \underset{E_{135^\circ}}{\begin{array}{ccccc} E & 0 & 0 & 0 & 0 \\ E & E & 0 & 0 & 0 \\ 0 & E & E & 0 & 0 \\ 0 & 0 & E & E & 0 \\ 0 & 0 & 0 & E & E \end{array}}$$

$$\underset{E_{r0^\circ}}{\begin{array}{ccccc} 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \\ 0 & \frac{E}{2} & E & \frac{E}{2} & 0 \end{array}} \qquad \underset{noise1}{\begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}}$$

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

noise2

The operator has perfect performance on ideal step edges and single noise patterns. By a thresholding with a threshold value greater than $\frac{E}{2}$ it can perfectly detect ideal ramp edges. The shortcoming of this operator is that it works only on 3 X 3 local neighborhood. Thus, its capability of reduce noise effect is limited.

VI-4-2. Blur and Minimum operator

This morphological edge operator is defined by

$$I_{Edge-strength} = \min\{I_1 - Erosion(I_1), Dilation(I_1) - I_1\} \qquad (VI-14)$$

where $I_1 = Blur\{\ I_{input}\ \}$ and $Blur\{\ I_{input}\ \}$ is the input image with a blurring operation. We use the same neighborhood size for the kernal of both blur and the structuring element of the dilation and erosion.

Consider the following one-dimensional step edge sequence as a motivation for this definition: The blur uses a neighborhood of width three and the erosion and dilation use a flat structuring element of domain $\{-1,0,1\}$.

original	0	0	0	0	E	E	E
blur	0	0	0	$\frac{E}{3}$	$\frac{2E}{3}$	E	E
erosion of blur	0	0	0	0	$\frac{E}{3}$	$\frac{2E}{3}$	E
dilation of blur	0	0	$\frac{E}{3}$	$\frac{2E}{3}$	E	E	E
blur − erosion	0	0	0	$\frac{E}{3}$	$\frac{E}{3}$	$\frac{E}{3}$	0
dilation − blur	0	0	$\frac{E}{3}$	$\frac{E}{3}$	$\frac{E}{3}$	0	0
edge strength	0	0	0	$\frac{E}{3}$	$\frac{E}{3}$	0	0

The advantage of this operator, as illustrated below, is that it will not detect single noise point.

original	0	0	0	N	0	0	0
blur	0	0	$\frac{N}{3}$	$\frac{N}{3}$	$\frac{N}{3}$	0	0
erosion of blur	0	0	0	$\frac{N}{3}$	$\frac{N}{3}$	0	0
dilation of blur	0	$\frac{N}{3}$	$\frac{N}{3}$	$\frac{N}{3}$	$\frac{N}{3}$	0	0
blur − erosion	0	0	$\frac{N}{3}$	0	0	0	0
dilation − blur	0	$\frac{N}{3}$	0	0	0	0	0
edge strength	0	0	0	0	0	0	0

Now let us examine the performance of this operator on ideal ramp edge sequences. Let a_i, be an one-dimensioal sequence which has $a_i = 0$ for all $i \leq 0$ and $a_i = E$ for all $i > 5$. Besides, $a_i \geq a_j$, for all $i > j$ and $i, j \in \{1, \ldots, 5\}$. Let a sequence e_i be the sequence a_i after a blur-minimum operation of three point support. Then

$$e_i = 0 \; for \; all \; i < 0$$

$$e_0 = \frac{a_1}{3}; \; e_1 = \frac{a_2}{3}$$

$$e_2 = \frac{\min(a_3, a_4 - a_1)}{3}$$

$$e_3 = \frac{\min(a_4 - a_1, a_5 - a_2)}{3}$$

$$e_4 = \frac{\min(a_5 - a_2, E - a_3)}{3}$$

$$e_5 = \frac{E - a_4}{3};\ e_6 = \frac{E - a_5}{3}$$

$$e_i = 0 \ for\ all\ i > 6$$

Thus, for an ideal ramp edge of three pixels $a_1 = a_2 = 0, a_3 = \frac{E}{2}, a_4 = a_5 = E$ the edge strength will be

e_0	e_1	e_2	e_3	e_4	e_5	e_6
0	0	$\frac{E}{6}$	$\frac{E}{3}$	$\frac{E}{6}$	0	0

For an ideal ramp edge of four pixels $a_1 = a_2 = 0, a_3 = \frac{E}{3}, a_4 = \frac{2E}{3}, a_5 = E$ the edge strength will be

e_0	e_1	e_2	e_3	e_4	e_5	e_6
0	0	$\frac{E}{9}$	$\frac{2E}{9}$	$\frac{2E}{9}$	$\frac{E}{9}$	0

For an ideal ramp edge of five pixels $a_1 = 0, a_2 = \frac{E}{4}, a_3 = \frac{E}{2}, a_4 = \frac{3E}{4}, a_5 = E$ the edge strength will be

e_0	e_1	e_2	e_3	e_4	e_5	e_6
0	$\frac{E}{12}$	$\frac{E}{6}$	$\frac{E}{4}$	$\frac{E}{6}$	$\frac{E}{12}$	0

For an ideal ramp edge of six pixels $a_1 = \frac{E}{5}, a_2 = \frac{2E}{5}, a_3 = \frac{3E}{5}, a_4 = \frac{4E}{5}, a_5 = E$ the edge strength will be

e_0	e_1	e_2	e_3	e_4	e_5	e_6
$\frac{E}{15}$	$\frac{2E}{15}$	$\frac{E}{5}$	$\frac{E}{5}$	$\frac{2E}{15}$	$\frac{E}{15}$	0

For an ideal ramp edge of seven pixels $a_1 = \frac{E}{6}, a_2 = \frac{E}{3}, a_3 = \frac{E}{2}, a_4 = \frac{2E}{3}, a_5 = \frac{5E}{6}$ the edge strength will be

$$\begin{array}{ccccccc} e_0 & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \\ \frac{E}{18} & \frac{E}{9} & \frac{E}{6} & \frac{E}{6} & \frac{E}{6} & \frac{E}{9} & \frac{E}{18} \end{array}$$

It is noted that this operator is noise insensitive. For the ideal step edge it produces a result which has non-zero edge strength on both the two edge pixels. This is consistent with the fact that an ideal step edge line should be two pixels in width. However, due to the effect of blurring, The edge strength assigned to the edge pixels is one third the edge contrast. For ideal ramp edges of different spatial extent, it will assign a non-zero edge strength to more than one pixels. However, the true edge pixel is mostly given higher edge strength than its neighbors. Thus, by thresholding the edge strength image with a suitful threshold value we can extract the ideal ramp edges. It is also noted that as the spatial extent of the ideal ramp edge increases, the edge contrast of the detected edge point decreases. For the case of a seven pixel ramp edge, the detected edge strength of the edge point is the same as the edge strength of its immediate adjacent pixels.

The reason why we need the small amount of blurring is that this operator only assigns a pixel to be an edge pixel if it has a value in the middle between two greyscale extremes of the neighborhood centered at the given pixel. Thus, there must be significant differences in greyscale value between the pixel and both its nearby greyscale maximum and nearby greyscale minimum pixel. The edge pixels for are the two pixels on either side of the jump. For the ideal step edge, these pixels are a local maximum and minimum. Hence, this operator can not detect the ideal step edges unless we blur the ideal step edge before applying this operator.

In order to have a better understanding of this edge operator, we give a derivation which explains this operator as an easily understandable local neighborhood non-linear operator. Let K be the neighborhood size of the kernel of the blur and the domain of the structuring element. Without loss of generality we assume that K is an odd number. Let $L = \frac{K-1}{2}$, and $b_i = Blur(a_i)$, $e_i = b_i - Erosion(a_i)$, $d_i = Dilation(a_i) - b_i$, then

$$b_i = \frac{\sum_{p=i-L}^{i+L} a_p}{K}$$

$$e_i = \max_{q\in\{0,\ldots,K-1\}} \{b_i - b_{i+L-q}\}$$

$$= \frac{1}{K} \max_{q\in\{1,\ldots,L\}} \{(\sum_{p=1}^{q} a_{i+L+p} - \sum_{p=1}^{q} a_{i-L-1+p}),$$

$$(\sum_{p=1}^{q} a_{i+L+1-p} - \sum_{p=1}^{q} a_{i-L-p}), 0\}$$

and

$$d_i = \max_{q\in\{0,\ldots,K-1\}} \{b_{i+L-q} - b_i\}$$

$$= \frac{1}{K} \max_{q \in \{1,\ldots,L\}} \{(\sum_{p=1}^{q} a_{i-L-1+p} - \sum_{p=1}^{q} a_{i+L+p}),$$

$$(\sum_{p=1}^{q} a_{i-L-p} - \sum_{p=1}^{q} a_{i+L+1-p}), 0\}$$

For example, let K = 3, we define one-dimensional five point masks

$$\begin{array}{l} A_1 = \frac{1}{3} * \quad [\begin{array}{ccccc} -1 & 0 & 0 & 1 & 0 \end{array}] \\ A_2 = \frac{1}{3} * \quad [\begin{array}{ccccc} 0 & 1 & 0 & 0 & -1 \end{array}] \\ A_3 = \frac{1}{3} * \quad [\begin{array}{ccccc} 1 & 0 & 0 & -1 & 0 \end{array}] \\ A_4 = \frac{1}{3} * \quad [\begin{array}{ccccc} 0 & -1 & 0 & 0 & 1 \end{array}] \end{array}$$

Then, the edge detector becomes

$$\min\{\max\{A_1 * f, A_2 * f\}, \max\{A_3 * f, A_4 * f\}\}$$

where f is the input data and "*" is the convolution operation.

In the case that K = 5, we define one-dimensional nine point masks

$$\begin{array}{l} A_1 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array}] \\ A_2 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} -1 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}] \\ A_3 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \end{array}] \\ A_4 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 0 & 0 & 1 & 1 & 0 & 0 & 0 & -1 & -1 \end{array}] \\ A_5 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 0 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{array}] \\ A_6 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 1 & 1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \end{array}] \\ A_7 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}] \\ A_8 = \frac{1}{3} * \quad [\begin{array}{ccccccccc} 0 & 0 & -1 & -1 & 0 & 0 & 0 & 1 & 1 \end{array}] \end{array}$$

Then, the edge detector becomes

$$\min\{\max\{A_1 * f, A_2 * f, A_3 * f, A_4 * f\}, \max\{A_5 * f, A_6 * f, A_7 * f, A_8 * f\}\}$$

It is found that this operator finds the difference between each side of a given point. Instead of considering only the difference of the average pixel intensity on both sides of a pixel it considers differences of varieties of local structures and combine the result of each difference by maximizations and a minimization operation. Thus, by increasing the neighborhood size of the blur operator and the neighborhood size of the morphologic operation this operator can reduce the noise effect and yet not blur too much the edges.

VI-5. Experimental Results

To understand the performance of the morphologic edge operators, we examine the behavior of the morphologic edge operators on two simulated images and compare the results of the morphologic edge operators with the cubic facet second derivative edge operator (Haralick, 1984).

The first simulated image is an image which has ten distorted square boxes. In order to simulate edges of different directions, the left side of the square boxes are tilted such that their correspoding edge lines have directions ranging from 0° to 45°. The edge contrast of these edges are 50 and the box size is 50 X 50 pixels. To this image, we add independent Gaussian noise having mean zero and standard deviation 15. The original and noisy images are shown in *Figure VI-1.*

In order to compare the performance of different edge operators, we use the conditional probability of assigned edge on state 'edge' given the true edge states 'edge' of an image, $P(E'|E^*)$, and the conditional probability of true edge states 'edge' given assigned edge states 'edge' of an image $P(E^*|E')$. The adjustable parameters of each edge operator are chosen to equalize these two conditional probabilities. The quality of the edge operator is determined by the value of $P(E'|E^*) = P(E^*|E')$. Although this performance measure is not in general applicable on all kinds of images, it is well suited for this simulated image.

We apply the maximum version (see equation (VI-7)), the improved summation version (see equation (VI-13)), the blur-minimum version (see equation (VI-14)) of the morphologic edge detectors and the cubic facet second derivative zero-crossing edge operator on the noisy square boxes and compare the performance in terms of $P(E'|E^*)$ and $P(E^*|E')$. The equivalent neighborhood supports for the blur-minimum operators used are 5 X 5 (use 3 X 3 neighborhood for the operators), 9 X 9 (use 5 X 5 neighborhood for the operators), and 13 X 13 (use 7 X 7 neighborhood for the operators), respectively. The neighborhood sizes for the cubic facet edge detector are 5 X 5 and 9 X 9, respectively. Table 1 lists the test results of these edge operators. The conditional probability of the maximum version mor-

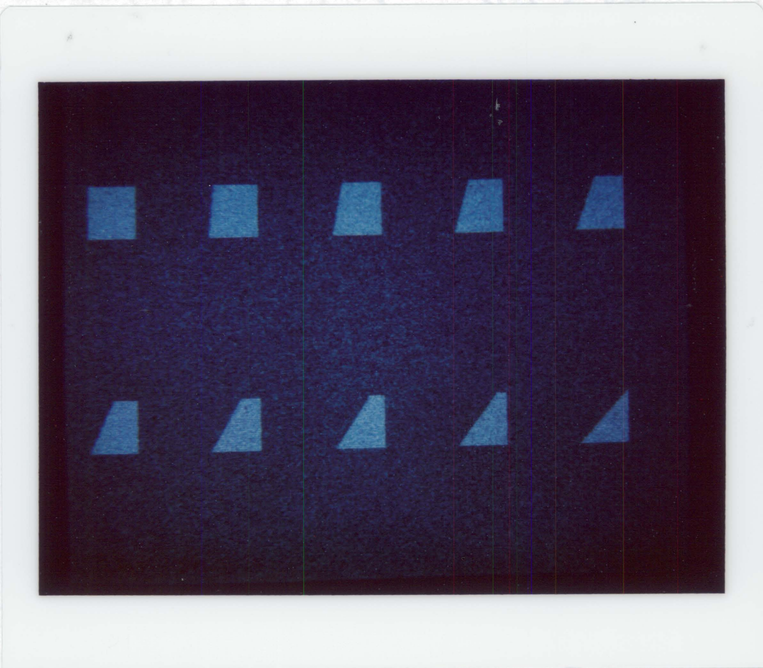


Figure VI-1. Shows the image containing ten boxes of different edge directions and the noisy image with $\sigma_n = 15.0$. The edge contrast is 50.

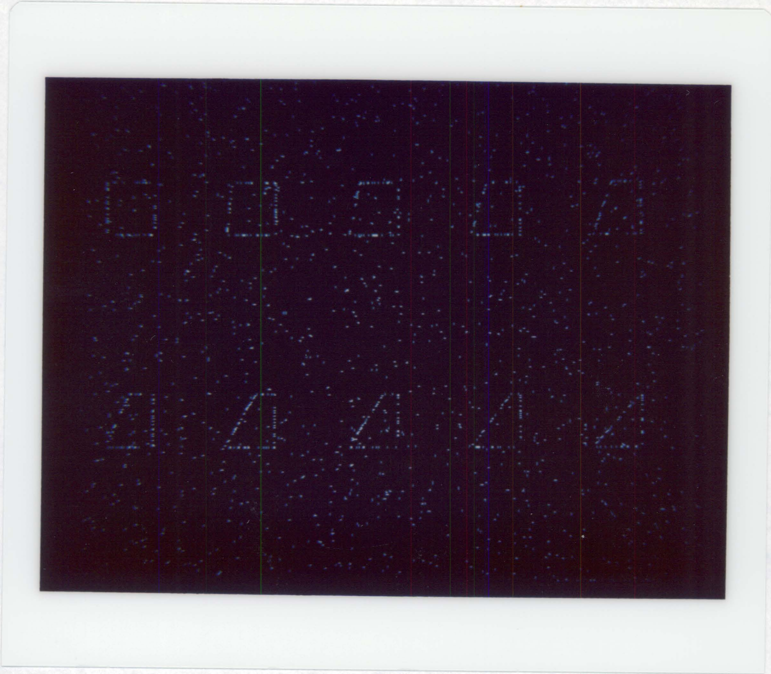
phologic edge operator is only 29 percent. The improved summation version of the morphologic edge operator increases the probability to 45 percent. The blur-minimum version of the morphologic edge operator has superior performance. It increases the probability to 78, 94, and 92 percents for 5 X 5, 9 X 9, and 13 X 13 neighborhood support, respectively. This is because the blur-minimum operators have a larger number of pixels involved in the edge detection than just a 3 X 3 neighborhood. The equivalent neighborhoods involved in edge detection process for the blur-minimum operator are 5 X 5, 9 X 9, and 13 X 13, respectively. Similarly, the cubic facet second derivative zero-crossing edge operator also makes use of larger than 3 X 3 neighborhood support and it has good performance. Its performance probability is about 82 and 92 percents for the 5 X 5 and 9 X 9 neighborhoods, respectively.

Figure VI-2(a),(b),(c), and (d) show the results of the maximum version, improved summation version, the blur-minimum (9 X 9) version of the morphologic edge operators, and the cubic facet edge operator (9 X 9).

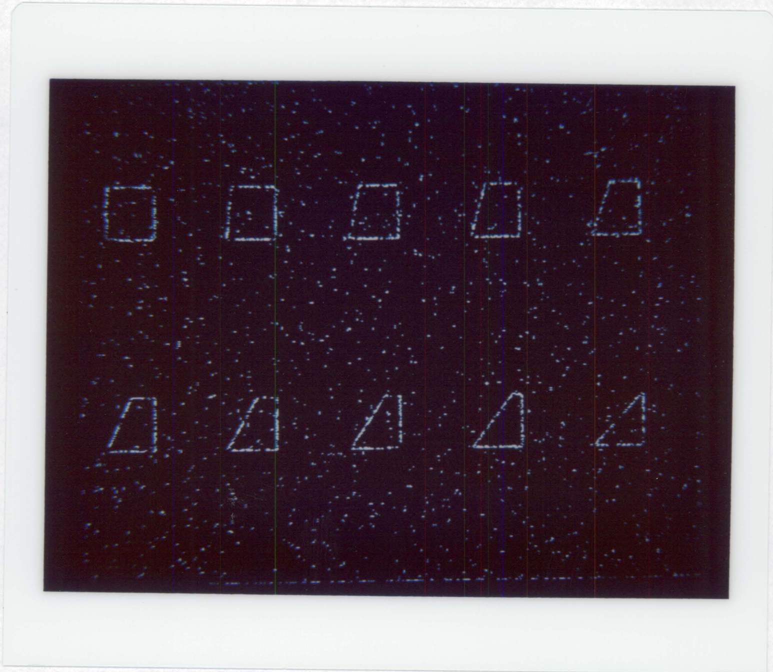
A visual evaluation also leaves the impression that the blur-minimum edge detector and the cubic facet second derivative zero-crossing edge operator produces much better edge continuity and has less noise than the other edge detectors.

Table 1. $P(E'|E^*)$ and $P(E^*|E')$ values of morphologic edge operators and cubic facet edge operators.

operator \ prob	$P(E'\|E^*)$	$P(E^*\|E')$
maximum	0.2911	0.2953
improved sum	0.4565	0.4380
blur-minimum(5X5)	0.7870	0.7870
blur-minimum(9X9)	0.9587	0.9350
blur-minimum(13X13)	0.9061	0.9369
facet edge (5X5)	0.8387	0.8256
facet edge (9X9)	0.9355	0.9063



(a)



(b)

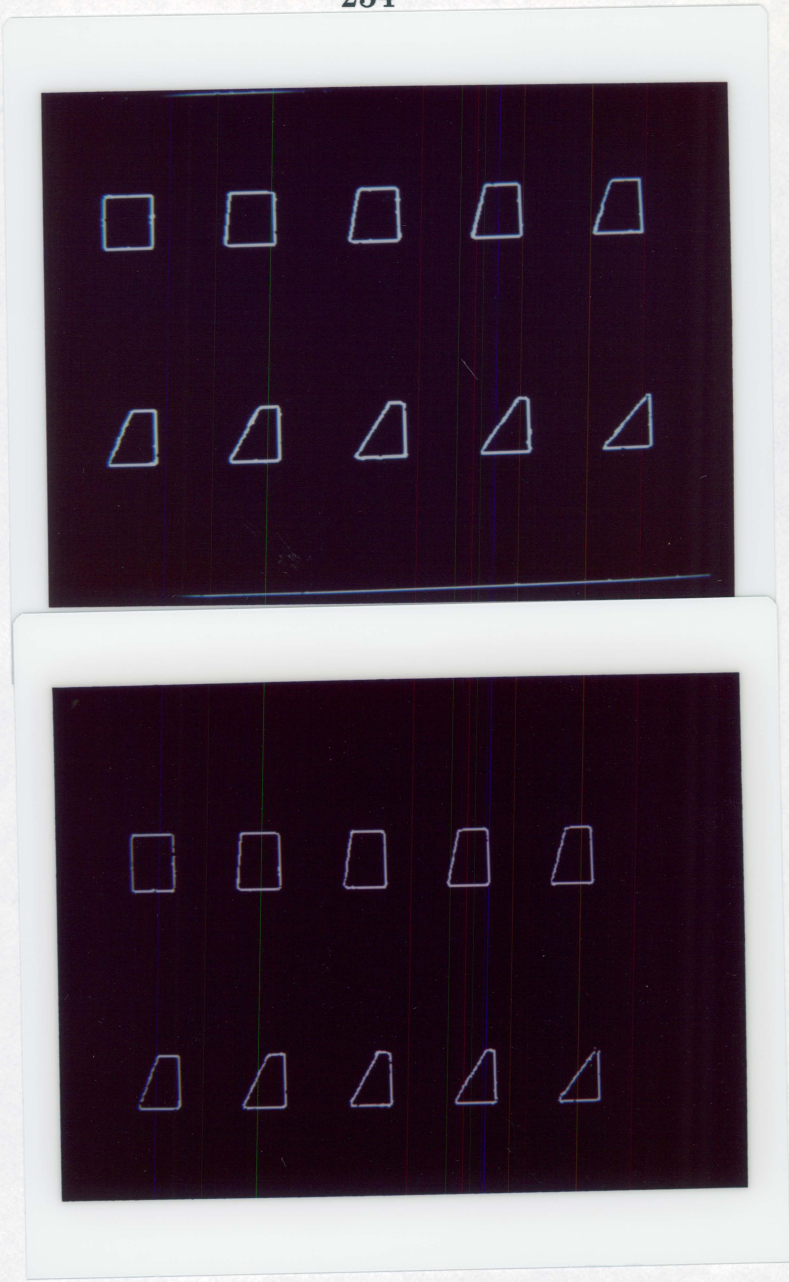


Figure VI-2. The edge results of different edge operators apply on the square box image. (a) Maximum version morphologic edge operator; (b) Improved summation version morphologic edge operator; (c) Blur-minimum version (9 X 9) morphologic edge operator. (d) Cubic facet second derivative zero-crossing edge operator of 9 X 9 neighborhood size.

The second simulated image is a checkerboard of size 100 x 100 pixels with a check size of 20 x 20 pixels. The dark checks have gray tone intensity 50 and the light checks have gray tone intensity 100. To this perfect checkerboard, we add independent Gaussian noise having mean zero and standard deviation 7.5, 15.0, and 30.0, respectively. Thus, the Signal to Noise Ratio (SNR $= \frac{edgecontrast}{noisedeviation}$) of these images are 6.67, 3.33, and 1.67, respectively. The perfect and noisy checkerboards are shown in *Figure VI-3.*

To compare the performance of the morphologic based edge operators with non-morphologic based edge operator. We apply the cubic facet based second directional zero-crossing edge operator and the maximum, improved summation and blur-minimum versions of the morphologc edge operator on the noisy checkboard images and compare the performance in terms of $P(E'|E^*)$ and $P(E^*|E')$. The structuring element for the maximum and improved summation morphologic edge operators is a rod of radius 1 which has four connected pixels as its neighborhood support. The neighborhood supports for the blur-minimum operator are 5 by 5 and 9 by 9, respectively. The window sizes of the cubic polynominal fitting for the second derivative zero-crossing operator are 5 by 5 and 9 by 9, respectively.

Figure VI-4 plots the probability results of these edge operators. The results show that the second derivative zero-crossing edge op-

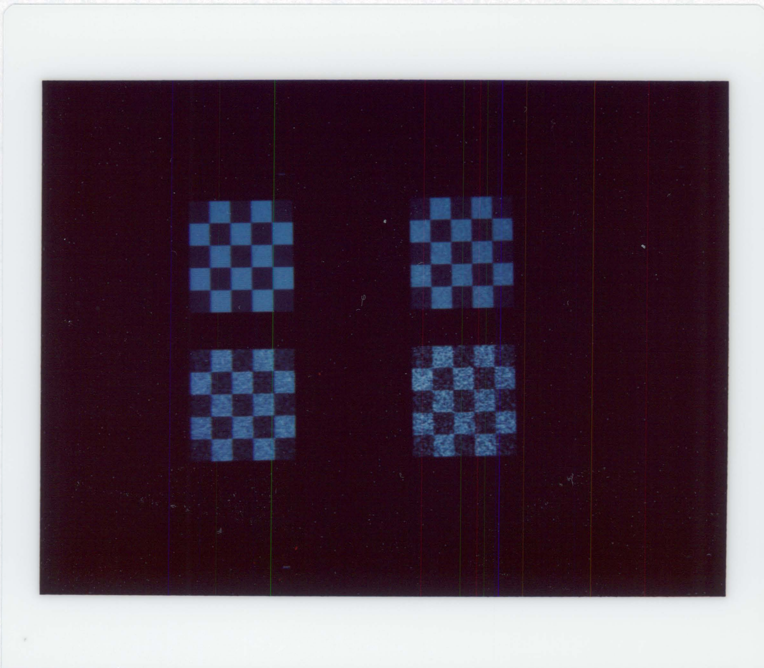


Figure VI-3. The perfect checkboard image and its noisy images. From left to right top to bottom are perfect, SNR = 6.67, SNR = 3.33, and SNR = 1.67, respectively. The image size is 100 X 100 pixels. The check size is 20 X 20 pixels. The edge contrast is 50 and the added noise is zero mean Gaussian noise.

erator and the blur-minimum morphologic edge operator have much better performance compare with the other two operators. When the SNR is large, the blur-minimum edge operator of 5 X 5 neighborhood size and the cubic facet zero-crossing edge operator perform best. As the SNR becomes small, the blur-minimum edge operator having a 9 X 9 neighborhood size and the cubic facet zero-crossing edge operator perform best. The improved summation morphologic edge operator has good performance as the SNR is large. As the SNR decreases its performance probability decreases dramatically and soon becomes much worse than the zero-crossing and blur-minimum operators. The maximum version morphologic edge operator has worst performance among all these operators.

The performance of the edge operators can be explained in terms of the neighborhood size we used for each operator. Both the blur-minimum edge operator of 9 X 9 equivalent kernel support and the 9 X 9 cubic facet second derivative zero-crossing edge operator involve 81 pixels in the edge detection process to assign the edge state of a single pixel. Due to the fact of large neighborhood size, they have best performance as the noise increases. The 5 by 5 blur-minimum morphologic edge operator and the 5 by 5 cubic facet second derivative zero-crossing edge operator involve 25 pixels in the edge detection process. Thus, they have best performance as the noise is small.

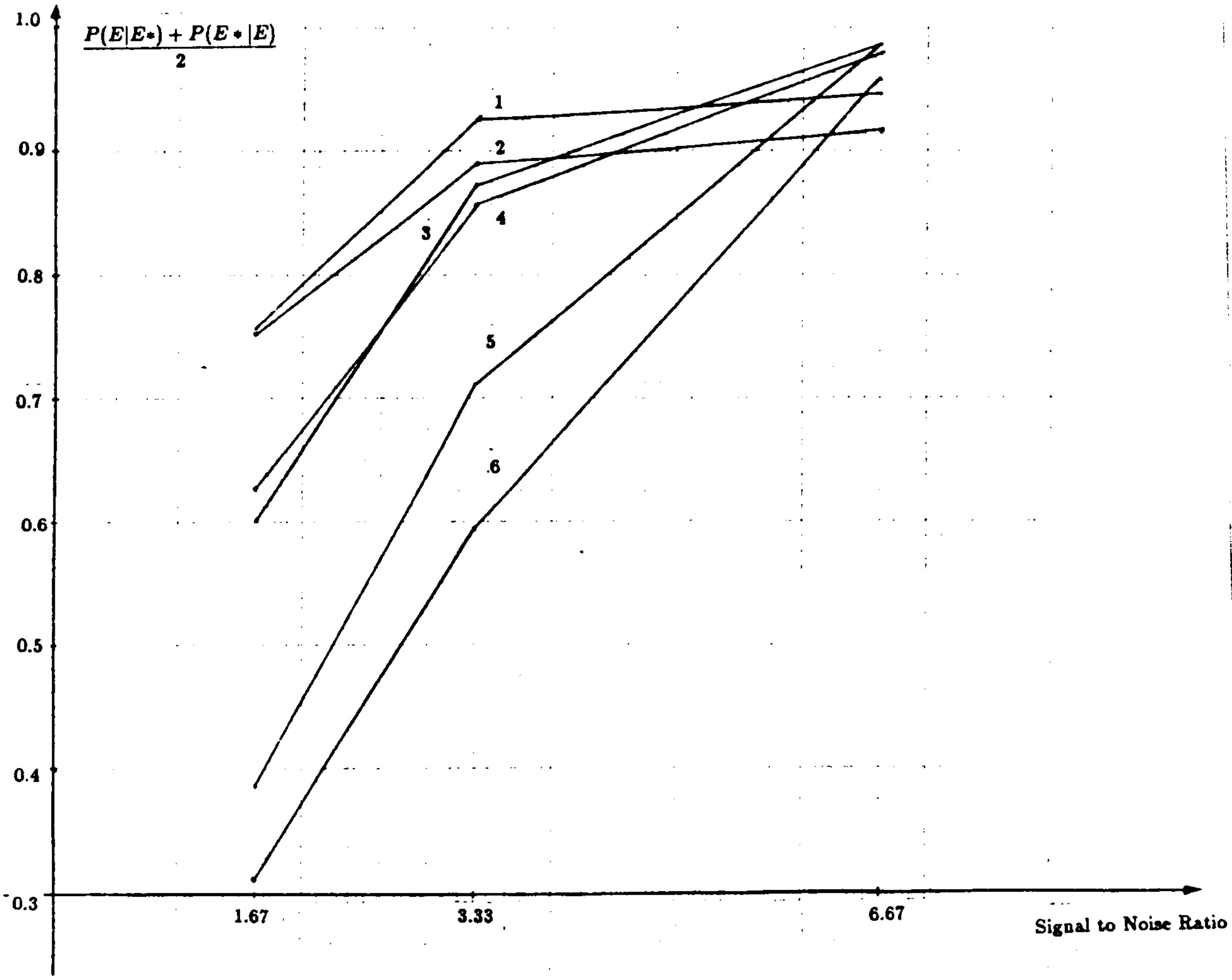


Figure VI-4. The performance probabilities of different edge operators applied on the noisy checkboard images. curve 1: blur-minimum morphologic edge operator of 9 by 9 equivalent support; curve 2: second derivative zero-crossing edge operator of 9 by 9 support; curve 3: blur-minimum morphologic edge operator of 5 by 5 equivalent support; curve 4: second derivative zero-crossing edge operator of 5 X 5 support; curve 5: improved summation morphologic edge operator; curve 6: maximum morphologic edge operator.

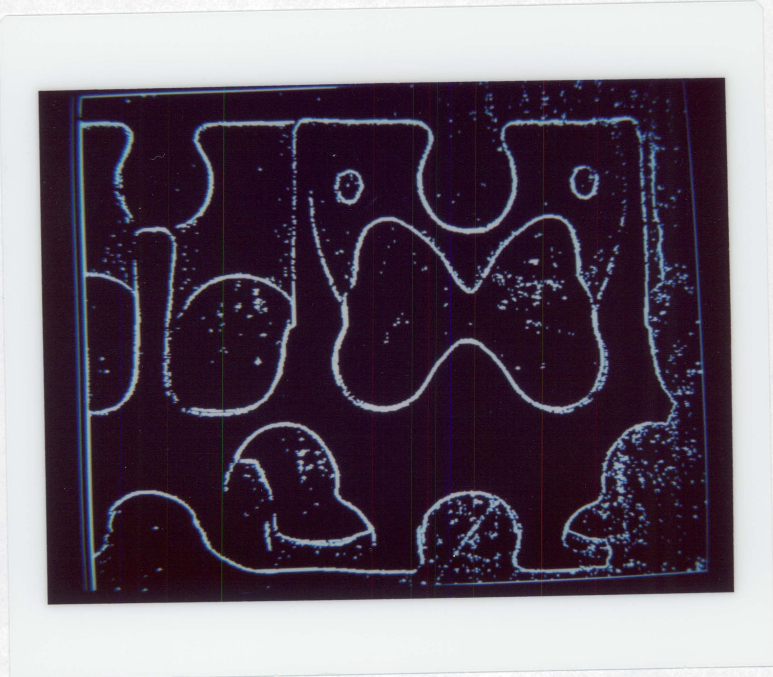
because they are good enough to deal with small noise and they will not blur too much the true edges. As the noise increases, their performance is worse than those with a larger neighborhood size. However, their performance is much better than the performance of the improved summation morphologic edge operator which involves only 9 pixels and the maximum version morphologic edge operator which involves only 5 pixels in the edge detection process.

It is noted that as the noise is small, an edge operator of small neighborhood support such as the improved summation operator is good enough. However, as the SNR becomes small, we have to use edge operators of larger neighborhood support. Since the performance of the blur-minimum morphologic edge operator is comparable to the zero-crossing edge operator, it will be very useful in those applications which can not afford the higher computation cost of the facet edge operator.

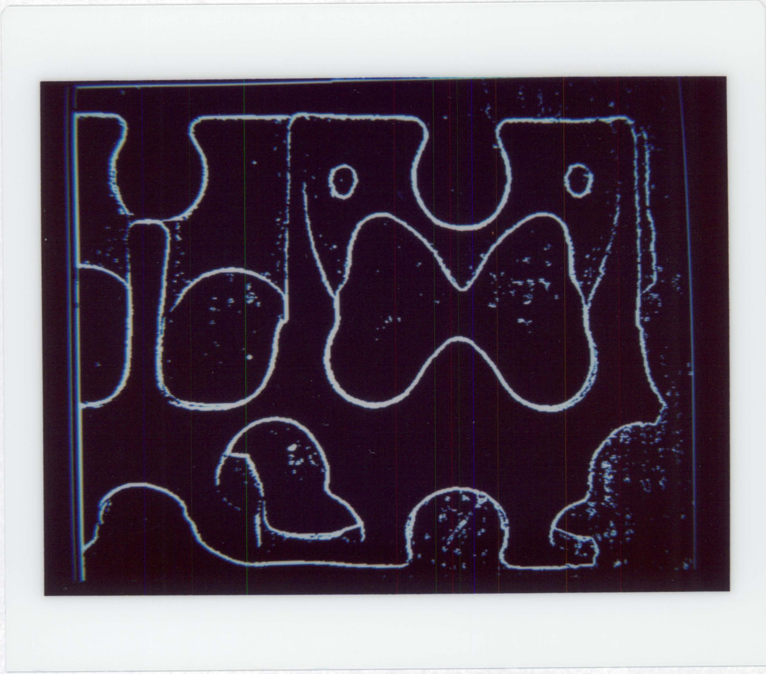
Finally, we illustrate an example of the morphologic edge detectors applied on a real image. The image is a mold for sand casting (see *Figure* VI-5(a)). We apply the maximum, the improved summation, the blur minimum morphologic edge detector with 9 by 9 support, a difference of Gaussian operator with circular support of diameter 40 and 24 pixels, and cubic facet second derivative zero-crossing of window size 9 by 9 on this image. The resulting edge images are shown in

Figure VI-5(b)-(f). A visual evaluation leaves the impression that the cubic facet edge operator and the blur-minimum morphologic edge detector have best performance. The difference of Gaussian operator produces thick edge lines and the edge connectivity is not as good as the blur-minimum edge operator. The improved summation edge operator is better than the maximum version edge operator. While, both of them are more noisy than the cubic facet and the blur-minimum edge operator.

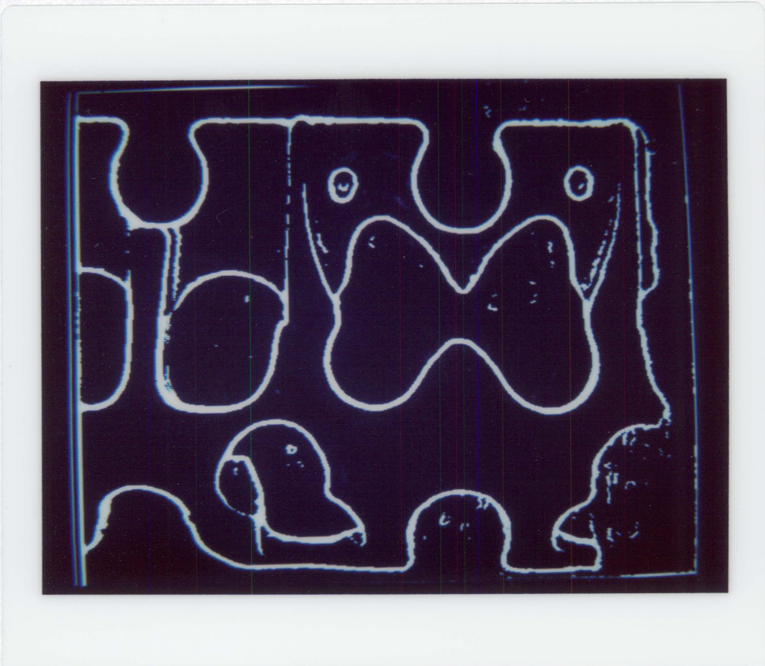
(a)



(b)



(c)



(d)

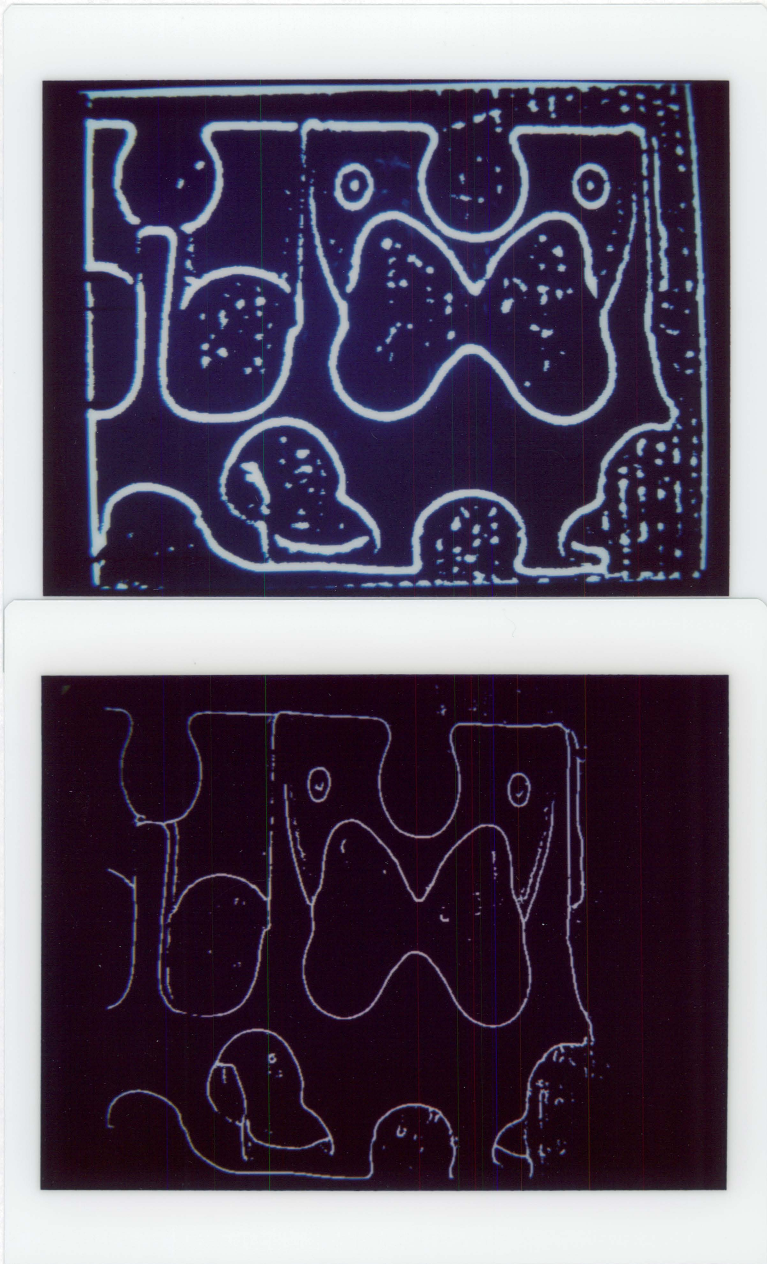


Figure VI-5. The sand casting mold image (a) and its edge images; (b) Edge image of maximum version morphologic edge operator; (c) Edge image of improved summation morphologic edge operator; (d) Edge image of 9 by 9 blur-minimum morphologic edge operator; (e) Edge image of difference of Gaussian edge operator; (f) Edge image of the 9 X 9 cubic facet edge operator.

VII. CONCLUSIONS

The problem of detecting edges in an image is addressed in this dissertation. We argued that image edges are not necessary scene edges and vice versa. To simulate the edge perception ability of human eyes and detect scene edges from a image, context information and world constraint must be employed in the edge detection process.

We have described one way of developing an edge detection scheme from a Bayesian theoretic framework. By assuming that only certain patterns of edge state and edge direction combinations are likely to occur in a 3 x 3 neighborhood of positions and giving all allowable patterns equal probability, a dictionary-based context dependent edge detection algorithm was formulated. We then discussed a general approach based on the dynamic programming method which can be used in the context edge detecting process in place of the dictionary scheme.

We also reviewed the facet model and the directional derivative concept. Based on a cubic polynomial facet model and the statistical analysis, in *Appendix 2* we showed how to obtain edge probability from local measurement.

We then demonatrated how lighting compensation and curvature constraint can aid the edge detection process. We provided some experimental results of these scheme applied on both simulated images and real images to support our arguments and concluded that world constraint can really improve any edge detection scheme which detects only image edges.

We derived an edge detection process which uses the context of the whole image for edge detection. Thus, we have the most desirable kind of edge labeling process which uses the entire context of the image. The edge detection operation is then a global process and not a local neighborhood operation.

We also illustrated how in the implementation of this operator we can achieve the balance between local and global context information. Besides, we show a good way of efficively using context. In the implementation, We make fully use of the strongest context information and avoid the weak and ambiguous context constraints.

The context information and world constraint can be used to evaluate the performance of edge detectors. We formulated a general edge evaluator based on local edge context and world constraint. A general edge coherence measure and a robust edge thinness measure are introduced. Besides, a general edge position correctness measure is also described. This is the first time a general edge correctness measure is proposed.

Finally, a simple edge detection scheme based on morphologic operators was discussed. In order to have a noise insensitive version of this operator some improved morphologic edge operator are also suggested.

The context information we used in the dissertation to aid the edge detction process is the general context information (for example, edge direction consistency). Thus, the context operators based on this general context are good for all the real images and they can extract more accurate scene edges. However, the algorithms we derived for the context edge detection mechanism are capable of incorporating much more context information to the edge detection process. Thus, we have a tool which can accept prior information about edge context (not necessary to be general) and apply this information most effectively to achieve good edge detection for the images which consistent with the given prior context information. This capability is very useful for the real applications. Because, in the real applications each specific set of images will most probably have some prior context information which are a lot stronger than the general context. By employing this specific information in the edge detection process through the context mechanism developed by this dissertation, we can expect to have a superior result. The concept is similar to the concept of applying morphologic operations. Although, it is almost impossible to have a general applicable morphologic operator for all the images, the morphologic operators are very useful in the real applications. In most of the real applications there exists some useful prior information about the shape of the objects to be processed. The morphologic operators are nothing but a set of tools which can make use of prior shape information most effectively. Since we have a tool which can use context information most effectively, we have reason to believe that it will also be very useful in real applications.

The scheme of using context and world constraint to help scene feature extraction can be applied on more than just edge detection. A modification of this scheme is directly applicable on the field of line detection.

REFERENCES

1. I. E. Abdou and W. K. Pratt, 'Qualitative design and evaluation of enhancement/thresholding edge detector', **Proceedings of IEEE**, Vol. 67, No.5, pp. 753-763, Mar. 1979.

2. Harry G. Barrow and Jay M. Tenenbaum,'Computational Vision', **Proceedings of IEEE**, Vol. 69, No.5, May 1981, pp. 572-595 .

3. M. J. Brooks, 'Rationalizing Edge Detectors', **C V G I P** Vol. 8, 1978, pp. 277-85.

4. F. W. Campbell and J. G. Robson, 'Applications of fourier analysis to the visibility of gratings **J. Physiol.** Lond. 197, 551-556, 1968.

5. J. F. Canny, 'Finding edges and lines in image', Master thesis, Department of E.E. and C.S. M.I.T., May 12, 1983.

6. F. M. Dickey and K. S. Shanmugam, 'Optimum edge detection filter,' **Applied Optics**, Vol. 16, No. 1, pp. 145-148, Jan. 1977.

7. G. P. Dineen,'Programming Pattern Recognition', **Proceeding of the Western Joint Computer Conference**, 1955, pp. 94-100.

8. N. R. Draper and H. Smith, **Applied Regression Analysis**, John Wiley and Son, Inc., New York, 1981.

9. R. O. Duda and P. E. Hart, 'Pattern classification and scene analysis', **Wiley**, New York, 1973.

10. J. R. Fram and E. S. Deutsch, 'On the quantitative evaluation of edge detection schemes and their comparison with human performance', **IEEE Trans. Computer**, Vol. C-24, No.6, pp.616-627, June 1975.

11. J. Fglein, 'On edge gradient approximations', **Pattern Recognition Letters**, 1983, pp.429-434

12. M. Golay,'Hexagonal Parallel Pattern Transformations',**IEEE Trans. on Computer** C-18, 1969, pp. 733-740.

13. Robert Haralick, 'Edge and Region Analysis for Digital Image Data', **Computer Graphics and Image Processing**, Vol. 12, 1980, pp. 60-73.

14. Robert Haralick, 'The Digital Edge', **Proceedings IEEE 1981 Conference Pattern Recognition Image Processing**, New York: IEEE Computer Society, pp. 285-94, 1981.

15. Robert Haralick, 'Zero-Crossing of Second Directional Derivative Edge Operator', **Society of Photogrammetric Instrumenta-**

tion Engineers Symposium on Robot Vision, Washington, D.C., May, 1982.

16. Robert Haralick, 'Ridge and Valleys on Digital Images', **Computer Vision Computer Graphics and Image Processing**, Vol. 22, 1983, pp. 28-38.

17. Robert Haralick, 'Digital Step Edges from Zero Crossing of Second Directional Derivatives', **IEEE Trans. Pattern Analysis and Machine Intelligence**, Vol. PAMI-6, No. 1, January, 1984, pp. 58-68.

18. Robert Haralick, L. T. Watson, T. J. Laffey, 'The Topographic Primal sketch', **The International Journal of Robotics Research**, Vol. 2, No. 1, Spring 1983, pp. 50-72.

19. Robert Haralick, 'Decision Making in Context', **IEEE Trans. Pattern Analysis and Machine Intelligence**, Vol. PAMI-5, No. 4, July 1983, pp. 417-28.

20. Robert Haralick and L. T. Watson, 'A Facet Model for Image Data', **Computer Graphics and Image Processing**, Vol. 15, 1981, pp. 113-29.

21. M. Hashimoto and J. Sklansky, 'Edge detection by estimation of multiple-order derivatives', *Computer Vision Graphics and Image Processing* 1985.

22. M. Hueckel, 'An Operator which Locates Edges in Digitized Pictures', **J. Assoc. Computing Machinery**, Vol. 18, 1971, pp. 113-25.

23. M. Hueckel, 'A Local Visual Operator Which Recognizes Edges and Lines', **J. Assoc. Computing Machinery**, Vol. 20, 1973, pp. 634-47.

24. R. A. Hummel, 'Feature detection using basis functions', **Computer graphics and image processing**, Vol. 9, pp.40-55, 1979.

25. A. Iannino and S. D. Shapiro, 'An interactive generalization of the Sobel edge detection', **IEEE 1979 Pattern Recognition and Image Processing**, pp. 130-137

26. R. Kirsch, 'Computer determination of the constituent structure of biological images', **Computer. Biomed.**, REs. 4, pp.315-328, 1971.

27. R. A. Kirsch,'Experiments in Processing lifemotion with a Digital Computer',**Proc. Eastern Joint Computer Conference**, 1957, pp. 221-229.

28. L. Kitchen and A. Rosenfeld, 'Edge evaluation using local edge coherence', **IEEE Trans. Sys. Man and Cybern.**, Vol. SMC-11, No. 9, pp.597-605, Sept. 1981.

29. R. M. Landsmon, L. B. Scott, M. J. E. Golay.,'Apparatus for Counting Bi-nucleate Lymplocytes in Blood',Patent 3214574 filed Oct 8, 1959, issued Oct 26, 1965.

30. W. H. H. J. Lunscher, 'The asymptotic optimal frequency domain filter for edge detection', **IEEE Trans. Pattern. Anal. Mach. Intell.**, Vol. PAMI-5, No.6, Nov. 1983, pp.678-680

31. Machine Vision International Corp.'Genesis 2000 BLIX language Manual' Version 1.0, 1984.

32. David Marr and Ellen Hildreth, 'Theory of Edge Detection', **Proceedings Royal Society of London**, B, Vol. 207, 1980, pp. 187-217.

33. A. Martelli, 'An Application of Heuristic Search Methods to Edge and Contour Detection', **Commun. ACM** 19, 2 February 1976, pp. 73-83.

34. G. Matheron,'Random Sets and Integral', **Geometry**, Wiley, New York ,1975.

35. J. W. Modestino and R. W. Fries, 'Edge detection in noisy images using recursive digital filtering', **Computer Graphics and Image Processing**, Vol. 6, 1977, pp. 409-433.

36. U. Montanari, 'On the Optimal Detection of Curves in Noisy Pictures', **Communication of the ACM** 14, 5 May 1971, pp. 335-45.

37. G. A. Moore,'Applications of Computers to Quantitative Analysis of Microstructure',U.S.W.B.S. Rept. No. 9428, 1966.

38. G. A. Moore,'Automatic Sensing and Computer Process for The Quantitative Analysis of Micrographs and Equivalued Subjects', in **Picturial Pattern Recognition**, Thompson Book Co. 1968, pp. 275-326.

39. David Morgenthaler, 'A New Hybrid Edge Detector', **Computer Graphics and Image Processing**, Vol. 16, 1981, pp. 166-176.

40. David Morgenthaler and Azriel Rosenfeld, 'Multidimensional Edge Section by Hypersurface Fitting', **IEEE Trans. on Pattern Analysis and Machine Intelligence**, Vol. PAMI-3, No. 4, July 1981, pp. 482-86.

41. R. Nevatia and K.R. Babu, 'Linear Feature Extraction and Description,'**Computer Graphics and Image Processing**, 1980, pp. 257-269.

42. T. Peli and D. Malah, 'A study of edge detection algorithms', **Computer Graphics and Image Processing**, Vol. 20, Sept. 1982, pp.1-21.

43. Alex P. Pentland, 'Local shading analysis', **IEEE Trans. Patt. Analy. Machi. Intelli**, Vol. PAMI-6, No.2, March 1984, pp.170-187.

44. K. T. Preston,'The cellscon System-A Lencocyte Pattern Analyzer', **Proceedings Western Joint Computer Conference**, 1961, pp 175-178.

45. K. T. Preston,'Feature Extraction by Golay Hexagonal Pattern Transforms',**IEEE Trans. Computer**, c-20. No. 9,1971, pp. 1007-1014.

46. K. T. Preston,'Multidimensional Logical Transforms', **IEEE Trans. PAMI**, Vol. PAMI-5, No. 5, Sept. 1983, pp. 539-554.

47. Judith Prewitt, 'Object Enhancement and Extraction', **Picture Processing and Psychopictorics** (B. Lipkin and A. Rosenfeld Eds.), Academic Press, New York, 1970, pp. 75-149.

48. G. S. Robinson, 'Edge detection by compass gradient masks,' **Computer Graphics and Image Processing**, vol. 6, 1977, pp.492-501.

49. A. Rosenfeld and J. L. Pfaltz, 'Sequential operatons in digital picture processing', **J. Assoc. Computer.**, March, 12(4), 1966, pp.471-494.

50. Azriel Rosenfeld and A. Kak, **Digital Picture Processing**, Vol. 2, Academic Press, New York, 1982.

51. J. Serra.,'Image analysis and Mathematical Morphology,' **Academic Press (London)**, 1982.

52. K. S. Shanmugam and F. M. Dickey and J. A. Green. 'An optimal frequency domain filter for edge detection in digital pictures', **IEEE Trans. P.A.M.I.**, Vol. PAMI-1, No.1 1979, pp.37-49.

53. S. R. Sternberg.,'Cellular Computers and Biomedical Image Processing', in **Lecture Notes in Medical Informations**, vol. 17: Biomedical images and computers. Proceedings 1980. Edited by J. Sklansky and J. C. Bisconte. Springer-Verlag, Berlin, 1980, PP.294-319.

54. S. R. Sternberg.,'Languages and Architectures of parallel Image Processing', Proceesings of the **Conference on Pattern Recognition in Practice**, Amsterdam, May 21-23, 1980. Kanal, L.N. and Gelsema, E.S.,eds.,North-Holland, Netherlands, 1980.

55. S. R. Sternberg.,'Biomedical Image Processing,' **IEEE Computer Magazine**, Volume 16, Number 1, January 1983.

56. S. R. Sternberg.,'Grayscale Morphology,' Submitted for publication in **Computer Graphics and Image Processing**.

57. Steve W. Zucker and Robert A. Hummel and Azriel Rosenfeld, 'An application of relaxation labeling to line and curve enhancement', **IEEE Trans. on Computers**, Vol. C-26, No.4, April 1977, PP. 394-403

58. Steve Zucker & Robert Hummel, 'An Optimal Three-Dimensional Edge Operator', **IEEE Pattern Recognition and Image Processing Conference**, Chicago, August 1979, pp. 162-68.

APPENDIX 1.

Putting the two conditional independence assumptions (II-13) and II-(14) together, we have

$$P(\varepsilon^*_{rc}|\underline{k}_{ij} : (i,j) \in RXC) =$$

$$\sum_{\underline{k}^*_{rc}} \frac{P(\underline{k}_{ij} : (i,j) \in N(r,c), \underline{k}_{lm} : (l,m) \notin N(r,c), \underline{k}^*_{rc}, \varepsilon^*_{rc})}{P(K)}$$

$$= \sum_{\underline{k}^*_{rc}} \frac{P(\underline{k}_{ij} : (i,j) \in N(r,c), \underline{k}^*_{rc}, \varepsilon^*_{rc})P(\underline{k}_{ij} : (i,j) \notin N(r,c))}{P(K)}$$

$$\frac{P(\varepsilon^*_{rc}|\underline{k}_{ij} : (i,j) \in N(r,c))}{P(K)}$$

$$*P(\underline{k}_{ij} : (i,j) \in N(r,c))P(\underline{k}_{ij} : (i,j) \notin N(r,c))$$

Now when the left hand side is summed over the possible values ε^*_{rc} can take, the sum must be unity. Summing the right hand side over the possible values ε^*_{rc} can take, the sum is

$$\frac{P(\underline{k}_{ij} : (i,j) \in N(r,c))P(\underline{k}_{ij} : (i,j) \notin N(r,c))}{P(K)}$$

which also, therefore, must be unity. So, the two conditional independence assumptions imply *equation (II-15)*.

APPENDIX 2.

Let's examine the facet model which produces the coefficients $k_1, \ldots, k_{10}$. Suppose that the observed facet f(r,c) given the true facet coefficients $k_1^*, \ldots, k_{10}^*$ can be written as

$$f(r,c) = k_1^* + k_2^* r + k_3^* c + k_4^* r^2 + k_5^* rc + k_6^* c^2$$
$$+ k_7^* r^3 + k_8^* r^2 c + k_9^* rc^2 + k_{10}^* c^3 + \eta(r,c)$$

(A2.1)

where $\eta(r,c)$ is independent Gaussian noise having mean 0 and standard deviation σ. We assume that $k_2^*, \ldots, k_{10}^*$ are independent Gaussian random variables with mean 0 and standard deviation $\sigma_2^*, \ldots, \sigma_{10}^*$, respectively if there is no edge. If there is an edge, the means remain 0, but the standard deviation may change. Also, $k_4^*, \ldots, k_{10}^*$ will not be independent but will have one linear dependency.

Unfortunately, $k_2^*, \ldots, k_{10}^*$ are not observed. They are estimated by $k_2, \ldots, k_{10}$. Each coefficient k_i is determined by

$$k_i = \sum_{(r,c)} f(r,c) a_i(r,c)$$

where the weighting mask a_i has the property that

$$\sum_{(r,c)} f(r,c) a_i(r,c) = k_i^* + \sum_{(r,c)} \eta(r,c) a_i(r,c)$$

and $\sum_{(r,c)} a_i(r,c)a_j(r,c) = 0$

$$for\ i \neq j, i,j \in \{1,\ldots,6\}\ or\ i,j \in \{4,\ldots,10\}$$

Hence $E[k_i|k_i^*] = k_i^*$ and $V[k_i|k_i^*] = \sigma^2 \sum_{(r,c)} a_i^2(r,c)$. To simplify notation, we let $q_i = \sum_{(r,c)} a_i^2(r,c)$ Then we have $V[k_i] = \sigma q_i + \sigma_i^2$. Finally, because the masks for the first order coefficients are orthogonal to the masks of the second order coefficients and the masks of the second order coefficients are orthogonal to the third order coefficients and because the noise is independent normal, $k_2\ and\ k_3$ are independent of k_4, k_5, k_6 and k_4, k_5, k_6 are independent of $k_7, \ldots, k_{10}$. Coffecients $k_2\ and\ k_3$ are not independent of $k_7, \ldots, k_{10}$ but in the following derivation we will assume that their dependence is small so that they can be treated as independent. Finally the existence or non-existence of an edge has no influence on the coefficient k_1, the constant term, and the value of the constant term has no influence on weather or not an edge exists.

Thus,

$$P(\varepsilon^*, \theta^*|k_1, \ldots, k_{10}) = P(\varepsilon^*, \theta^*|k_2, \ldots, k_{10})$$

$$= P(k_2, k_3|\varepsilon^*, \theta^*)P(k_4, \ldots, k_{10}|\varepsilon^*, \theta^*)|P(k_2, \ldots, k_{10}) * \frac{P(\varepsilon^*)}{2\pi}$$

since edge pixels have no a priori favored edge directions. The important probabilities in this decomposition for the edge and no edge case are $P(k_2, k_3|\varepsilon^*, \theta^*) and P(k_4, \ldots, k_{10}|\varepsilon^*, \theta^*)$. First we consider the conditional probability for k_2 and k_3 given edge or no edge and then $k_4, \ldots, k_{10}$.

We do the analysis for k_2 and k_3 in terms of two orthogonal linear combinations of k_2 *and* k_3. Whether or not there is an edge, the direction θ^* is the gradient direction. The gradient direction θ^* is well defined when $k_2^{*2} + k_2^{*2} > 0$ and satisfies

$$k_2^* \cos\theta^* - k_3^* \sin\theta^* = 0$$

Thus, given the direction θ^* the random variable $k_2^* \cos\theta^* - k_3^* \sin\theta^*$ has a normal distribution with mean 0 and variance $\sigma^2(q_2 \cos^2\theta^* + q_3 \sin^2\theta^*)$. Since $q_2 = q_3$ the variance becomes $\sigma^2 q_2$.

Next we determine the distribution of the random variable $r = k_2 \sin\theta^* - k_3 \cos\theta^*$. The true value of the first directional derivative taken in the direction of the gradient is given by $r^* = k_2^* \sin\theta^* - k_3^* \cos\theta^*$.

By definition of the gradient direction, θ^* is a random variable and satisfies $r^* = \sqrt{k_2^{*2} + k_3^{*2}}$ which is the magnitude of the gradient. Let $\sigma_2 = \sigma_3 = \sigma_g$, then $\frac{k_2^{*2}+k_3^{*2}}{\sigma_g^2}$ has a chi-squared distribution with 2 degrees of freedom. Let $\eta_i = \sum_{(r,c)} \eta(r,c)a_i(r,c)$. Then each η_i has an independent normal distribution with mean 0 and variance $q_i\sigma^2$.

Since $q_2 = q_3, q_4 = q_6, q_7 = q_{10}, and\ q_8 = q_9$.

$$r = k_2 \sin\theta^* + k_3 \cos\theta^* = (k_2^* + \eta_2)\sin\theta^* + (k_3^* + \eta_3)\cos\theta^*$$

$$= r^* + \eta_2 \sin\theta^* + \eta_3 \cos\theta^* \tag{A2.2}$$

From (A2.2) the expected value μ_g of r is $\mu_g = \sqrt{\frac{\pi}{2}}\sigma_g^*$ and the variance of r is $2\sigma_g^{*2} + q\sigma^2$, where $q = q_2 = q_3$. If $\sigma_g^{*2} \gg q\sigma^2$, the density function for r will be approximately the density function for $\sqrt{k_2^{*2} + k_3^{*2}}$ which is the Rayleigh density function:

$$f(r) = \frac{r}{\sigma_g^{*2}} * \exp\left(\frac{-r^2}{2\sigma_g^{*2}}\right)$$

In order for there to be an edge the value of the first directional derivative must be non-zero. Thus the random variable r will have a significantly non-zero mean μ_g and a variance of $2\sigma_g^{*2} + q\sigma^2$. Let

$$V[k_2^* | \varepsilon^* =' no - edge'] = \sigma_{g1}^2$$

then

$$V[k_2^* | \varepsilon^* =' edge'] = \lambda^2 \sigma_{g1}^2$$

where λ is greater than 1. Therefore, In the no edge case we have mean $\mu_g = 0$ and variance $2\sigma_{g1}^{*2} + q\sigma^2$. In the edge case, we have a mean $\mu_g = \sqrt{\frac{\pi}{2}}\lambda\sigma_g^*$ and a larger variance $2\lambda^2\sigma_g^{*2} + q\sigma^2$.

From the above analysis we have determined that k_2 and k_3 are normally distributed. The expected value of $\begin{pmatrix} k_2 \\ k_3 \end{pmatrix}$ is given by

$$\begin{pmatrix} \mu_2 \\ \mu_3 \end{pmatrix} = \begin{pmatrix} \cos\theta^* & \sin\theta^* \\ -\sin\theta^* & \cos\theta^* \end{pmatrix} \begin{pmatrix} 0 \\ \mu_g \end{pmatrix} \tag{A2.3}$$

and its covariance is

$$\begin{pmatrix} \cos\theta^* & \sin\theta^* \\ -\sin\theta^* & \cos\theta^* \end{pmatrix} \begin{pmatrix} \sigma^2 q & 0 \\ 0 & 2\sigma_g^{*2} + \sigma_q^2 \end{pmatrix} \begin{pmatrix} \cos\theta^* & -\sin\theta^* \\ \sin\theta^* & \cos\theta^* \end{pmatrix} \tag{A2.4}$$

Next we consider the conditional distribution of $k_4, \ldots, k_{10}$. If the pixel is a non-edge pixel, there is no constraint on the second directional derivative. However, if the pixel is an edge, there must be a point in the pixel's area such that the second directional derivative taken in the gradient direction must have a negatively sloped zero crossing. This represents a linear constraint on $k_4^*, \ldots . k_{10}^*$. From *equation (II-8)* we know that

$$f''_{\theta^*}(\rho) = A^* \rho + B^* \tag{A2.5}$$

where

$$A^* = 6[k_7^* S^3 + k_8^* S^2 C + k_9^* S C^2 + k_{10}^* C^3]$$

$$B^* = 2[k_4^* S^2 + k_5^* S C + k_6^* C^2]$$

$$S = \sin\theta^*;\ C = \cos\theta^*$$

Thus the constraint on $k_4^*, \ldots, k_{10}^*$ is that an edge exists if for some $\rho, 0 < |\rho| < \rho_0$ where ρ_0 is just smaller than the length of a side of a pixel, we have $A^*\rho + B^* = 0$ and $A* < 0$. This constraint causes the difference in the distribution for the observed $(k_4, \ldots, k_{10})$ in the case of edge and non-edge pixels.

The true value of coefficients A^* and B^* are not known. Given the true gradient direction θ^*, the computed coefficients A and B satisfy

$$A = 6[k_7S^3 + k_8S^2C + k_9SC^2 + k_{10}C^3]$$

$$= A^* + 6[\eta_7^3 + \eta_8S^2C + \eta_9SC^2 + \eta_{10}C^3] = A^* + \eta_A$$

$$B = 2[k_4S^2 + k_5SC + k_6C^2]$$

$$= B^* + 2[\eta_4S^2 + \eta_5SC + \eta_6C^2] = B^* + \eta_B$$

We want to determine the joint distribution of A and B under the conditions of edge and no edge. If there is no edge there is no constraint of A^* and B^*. If there is an edge, there is the constraint

$$r^* = \frac{A^*\rho + B^*}{\sqrt{1+\rho^2}} = 0$$

Our analysis of the joint distribution of A and B will be relative to the orthonormal linear combinations

$$r^* = \frac{(A^*\rho + B^*)}{\sqrt{1+\rho^2}} \ and$$

$$s^* = \frac{(-A^* + \rho B^*)}{\sqrt{1+\rho^2}}$$

Let the orthonormal matrix T be given by

$$T = \begin{pmatrix} \frac{\rho}{\sqrt{1+\rho^2}} & \frac{-1}{\sqrt{1+\rho^2}} \\ \frac{1}{\sqrt{1+\rho^2}} & \frac{\rho}{\sqrt{1+\rho^2}} \end{pmatrix} \tag{A2.6}$$

Then,

$$\begin{pmatrix} A^* \\ B^* \end{pmatrix} = T \begin{pmatrix} r^* \\ s^* \end{pmatrix}$$

If there is no edge all the random variables A^*, B^*, η_A, η_B are independent normals with mean 0 and variances

$$\sigma^2_{A^*} = 36(S^6\sigma_7^2 + S^4C^2\sigma_8^2 + S^2C^4\sigma_9^2 + C^6\sigma_{10}^2)$$

$$\sigma^2_{B^*} = 4(S^4\sigma_4^2 + S^2C^2\sigma_5^2 + C^4\sigma_6^2)$$

$$\sigma^2_{\eta_A} = 36(S^6\sigma^2_{q_7} + S^4C^2\sigma^2_{q_8} + S^2C^4\sigma^2_{q_9} + C^6\sigma^2_{q_{10}})$$

$$\sigma^2_{\eta_B} = 4(S^4\sigma^2_{q_4} + S^2C^2\sigma^2_{q_5} + C^4\sigma^2_{q_6})$$

Thus

$$\begin{pmatrix} A \\ B \end{pmatrix} = T \begin{pmatrix} r^* \\ s^* \end{pmatrix} + \begin{pmatrix} \eta_A \\ \eta_B \end{pmatrix}$$

from which it follows that $\begin{pmatrix} A \\ B \end{pmatrix}$ has a normal distribution with mean 0 and covariance

$$T \begin{pmatrix} \sigma^2_{r^*} & 0 \\ 0 & \sigma^2_{s^*} \end{pmatrix} T' + \begin{pmatrix} \sigma^2_{\eta_A} & 0 \\ 0 & \sigma^2_{\eta_B} \end{pmatrix}$$

(A2.7)

where

$$\sigma_{r^*}^2 = \frac{\rho^2\sigma_{A^*}^2 + \sigma_{B^*}^2}{1+\rho^2} \; and \; \sigma_{s^*}^2 = \frac{\sigma_{A^*}^2 + \rho^2\sigma_{B^*}^2}{1+\rho^2}$$

If there is an edge, we have $r^* = 0$ and s^* has normal distribution with positive mean $-\mu_{A^*}$ (since $A^* < 0$) and variance $\sigma_{s^*}^2$. In this case,

$$\begin{pmatrix} A \\ B \end{pmatrix} = T \begin{pmatrix} 0 \\ r_5^* \end{pmatrix} + \begin{pmatrix} \eta_A \\ \eta_B \end{pmatrix}$$

(A2.8)

from which it follows that $\begin{pmatrix} A \\ B \end{pmatrix}$ has a normal distribution with mean $\begin{pmatrix} \mu_{A^*} \\ 0 \end{pmatrix}$ and covariance

$$T \begin{pmatrix} 0 & 0 \\ 0 & \sigma_{r^*}^2 \end{pmatrix} T' + \begin{pmatrix} \sigma_{\eta_A}^2 & 0 \\ 0 & \sigma_{\eta_B}^2 \end{pmatrix}$$

(A2.9)

Since A and B are function of $k_4, \ldots, k_{10}$, The probability

$$P(k_4, \ldots, k_{10} | \varepsilon^*, \theta^*) = P(k_4, \ldots, k_{10}, A, B | \varepsilon^*, \theta^*)$$

$$= \frac{P(\varepsilon^*, \theta^* | k_4, \ldots, k_{10}, A, B)}{P(\varepsilon^*, \theta^*)} * P(k_4, \ldots, k_{10}, A, B)$$

(A2.10)

Now θ^* is independent of $k_4, \ldots, k_{10}$ and the true edge state constrains $k_4, \ldots, k_{10}$ only through A and B. That is, once the values of A, B are known, the values $k_4, \ldots, k_{10}$ do not contain any further information about ε^*. Thus

$$P(\varepsilon^*, \theta^* | k_4, \ldots, k_{10}, A, B) = P(\varepsilon^*, \theta^* | A, B)$$

(A2.11)

Putting (A2.11) into (A2.10) we have

$$P(k_4, \ldots, k_{10} | \varepsilon^*, \theta^*) = P(A, B | \varepsilon^*, \theta^*) P(k_4, \ldots, k_{10} | A, B)$$

We assume that for each value of (A,B), $P(k_4, \ldots, k_{10} | A, B)$ is nearly constant. In this case

$$P(k_4, \ldots, k_{10} | \varepsilon^*, \theta^*) \sim P(A, B | \varepsilon^*, \theta^*)$$

Therefore, instead of using $P(k_4, \ldots, k_{10} | \varepsilon^*, \theta^*)$ we can use

$P(A, B | \varepsilon^*, \theta^*)$ in the edge detecting process.

APPENDIX 3.

Under assumptions of *equations(II-27) and (II-28)* we have

$$P(\underline{k}_n|\varepsilon_N^*,\theta_N^*) = \sum_{\underline{k}_n^*} P(\underline{k}_n,\underline{k}_n^*|\varepsilon_N^*,\theta_N^*)$$

$$= \frac{\sum_{\underline{k}_n^*} P(\underline{k}_n^*,\varepsilon_N^*,\theta_N^*)P(\underline{k}_n^*,\varepsilon_N^*,\theta_N)}{P(\varepsilon_N^*,\theta_N^*)}$$

$$= \frac{\sum_{\underline{k}_n^*} P(\underline{k}_n^*,\varepsilon_n^*,\theta_n^*)P(\underline{k}_n^*,\varepsilon_N^*,\theta_N)}{P(\varepsilon_N^*,\theta_N^*)}$$

$$= \frac{\sum_{\underline{k}_n^*} P(\underline{k}_n^*,\varepsilon_n^*,\theta_n^*)P(\underline{k}_n^*,\varepsilon_n^*,\theta_n)P(\varepsilon_{N-n}^*,\theta_{N-n}^*|\underline{k}_n^*,\varepsilon_n^*,\theta_n^*)}{P(\varepsilon_N^*,\theta_N^*)}$$

where

$$\varepsilon_{N-n}^* = \{\varepsilon_k^* : k \in \{0,\ldots,8\} \; and k \neq n\}$$

$$\theta_{N-n}^* = \{\theta_k^* : k \in \{0,\ldots,8\} \; and k \neq n\}$$

Since the dependence between $\varepsilon_{N-n}^*,\theta_{N-n}^*$ and $\underline{k}_n^*,\varepsilon_n^*,\theta_n^*$ is only through edge data ε_n^*, θ_n^*, we have

$$P(\varepsilon_{N-n}^*,\theta_{N-n}^*|\underline{k}_n^*,\varepsilon_n^*,\theta_n^*) = P(\varepsilon_{N-n}^*,\theta_{N-n}^*|\varepsilon_n^*,\theta_n^*) \tag{A3.1}$$

Now we can show that under *equations (A3.1) and (II-28)* and some mathematical manipulations , the observed facet parameters for each pixel depend only on the true edge state and direction for the pixel and does not depend on the true edge state and direction for any other pixels in the neighborhood. Thus

$$P(\underline{k}_n | \varepsilon_N^*, \theta_N^*) = P(\underline{k}_n | \varepsilon_n^*, \theta_n^*) \tag{A3.2}$$

From *equations (A3.2) and (II-27), equation(II-29)* follows directly.

The vita has been removed from the scanned document