IMAGE PROCESSING USING THE HOUGH TECHNIQUE

by

XINMENG LIAO

A thesis
presented to the University of Manitoba
in fulfillment of the
thesis requirement for the degree of
M.Sc.
in
Department of Electrical Engineering

Winnipeg, Manitoba

ISBN 0-315-47885-3

IMAGE PROCESSING USING THE HOUGH TECHNIQUE

BY

XINMENG LIAO

A thesis submitted to the Faculty of Graduate Studies of
the University of Manitoba in partial fulfillment of the requirements
of the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

XINMENG LIAO

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

XINMENG LIAO

The University of Manitoba requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

ABSTRACT

A recurring problem in image processing is the detection of known shapes in digitized pictures. In theory, the problem can be solved by scanning the whole image to find all subsets of pixels that are close to particular features. However, the computation required actually prohibits this method. A new procedure with greatly improved efficiency in this area was presented by P.V.C.Hough and titled the Hough transform. This thesis has developed some new techniques to apply the Hough transform as a filtering procedure to extract interesting features from noisy binary images and outdoor scenes.

Quantitative comparisons between the performance of the Hough transform approach and that of the Median filter are made in this thesis. The comparison results strongly support the opinion that the Hough transform procedure is able to be employed as a specific feature filter to search and extract desired shapes from noisy images.

ACKNOWLEDGEMENTS

The author wishes to express his sincere appreciation to his academic advisor, Dr. M.Pawlak, for his patient supervision, invaluable suggestions, generous guidance, and encouragement throughout this research, and for his endless efforts to be available for the many educational discussions.

The author is also thankful for the numerous assistance, excellent suggestions he received from his colleagues and friends.

The author would like also to thank his wife, Yang Ming, for her moral support and understanding.

Special gratitudes are due to the author's parents for their encouragement and understanding over the years.

Financial support from the China Government and NSERC Grant A8131 is appreciated gratefully.

CONTENTS

Chapter I

INTRODUCTION

There are many industrial, medical, remote sensing and other applications where it is necessary to extract objects of known shape from digital images. An interesting and completely new approach in this area was introduced by P.V.C.Hough[1] in 1962, named the Hough transform. Hough's algorithm was based on a slope-intercept parameterization of lines and was initially developed for the detection of straight lines. In 1972, the technique was further improved by Duda and Hart[2] through the use of the angle-radius parameterization. Duda and Hart suggested that their Hough transform line detection approach could be extended to detect other curves. As an example, they considered detecting the family of all circles having centers in the same retina by choosing a parametric represention a-b-c, where a and b are the Cartesian coordinates of the center point of a circle and c the radius. Since then, considerable research on the Hough transform has been conducted. Kimme et al.[27] increased its efficiency and used it for the detection of chest tumors in 1975. In 1978, Tsuji and Matsumoto[28] adapted it for the detection of ellipses. Ballard[6] presented how to use it for the detection of general shapes

with developed computational efficiency in 1981. More recently, Krishnapuram and Casasent[5] suggested a new approach. This approach, by using the basic Hough transform defined for straight lines to estimate the scale, translation and rotation distortion parameters of an input test object, is capable of multi-class object discrimination and multiple-distortion object recognition.

However, the research mostly has been based on man-made noise free images and few attempts were made to images degraded with noise. This is partly because of the reasons that Duda and Hart have pointed out. Several problems have limited the utility of the Hough transform. First, finer quantization in the Hough space gives better resolution, but increases the computation time and exposes the problem of clustering entries corresponding to nearly collinear points. Secondly, the Hough transform procedure finds collinear points without regard to contiguity. Related problems are that the position of a best-fit line can be distorted by the presence of unrelated figure points in another part of the picture, while meaningless groups of collinear points are detected.

In this thesis, an attempt is made to solve the above mentioned problems. Also, as an extension of the Hough transform technique, the thesis has made an effort to apply the Hough transform as a filtering procedure to extract specific known features from noisy images.

There are three major sections in the thesis. The first section is composed of Chapter II and Chapter III. In Chapter II, the edge detection problem, which is a prerequisite for the Hough transform technique, is reviewed. Chapter III gives a fundamental description of the Hough transformation.

The second section contains one chapter (Chapter IV). As mentioned above, it is a new endeavor to employ the Hough transform for extracting known shape from images degraded with noise. To accomplish this, some problems have to be solved before the Hough transform technique can be applied as the filtering procedure. Chapter IV will discuss these problems and present some new results.

The last section of the thesis, which consists of Chapter V and Chapter VI, presents the results concerning the Hough transform as the filtering procedure. In this section, the Hough transform filter is applied to artificial images involving different types of noise. Some outcomes relating to straight line extraction of outdoor scenes are also presented. In Chapter V, a comparison is made between the performance of the Hough transform filter and that of the Median filter. The Median filter is commonly considered as one of the best filters to remove impulse noise from images[17].

Concluding remarks and suggestions on possible improvements for further research in this field are presented in Chapter VII.

In this thesis, a new displaying package was developed based on the Xerox 8700 printer. The new routine displays each image pixel by employing 64 small dots rather than striking characters several times. Therefore, it offers higher resolution and better quality in image displaying. The details and programs about this package are presented in Appendix A. Appendix B includes all the computer programs for the research.

Chapter II

EDGE DETECTION

Edges are primitive features of an image that are widely used in image understanding and pattern recognition systems to outline the boundaries of objects. An edge may be defined as the boundary between two adjacent regions in an image, with each region homogeneous within itself, but different from the other with respect to some given local property.

Edge detection is an important operation in a number of image processing applications. It is hard to over emphasize the importance of edge detection in image understanding. Most modules in a conceivable vision system directly or indirectly depend on the performance of the edge detector. The usual aim of edge detection is to locate the boundaries of objects of interest.

There are two basic image edge detection approaches: the enhancement-thresholding method and the edge fitting method. This thesis will base on the former strategy. In the enhancement-thresholding method, discontinuities in an image characteristic are enhanced or accentuated by some spatial operators, if the enhanced discontinuities are suf-

ficiently large — greater than a given threshold level — the edges are deemed to be present.

The edge enhancement-thresholding edge detection approach is described in Fig.2-1. In this method, the discrete image array $F(j,k)$ is spatially processed by a set of N linear operators or masks $H_i(j,k)$ to produce a set of gradient functions

$$G_i(j,k) = F(j,k)*H_i(j,k)$$

where * represents a two-dimensional spatial convolution. Then, at each pixel, the gradient functions are combined by a linear or nonlinear point operator « » to create an edge enhanced array

$$A(j,k) = « G_i(j,k) »$$

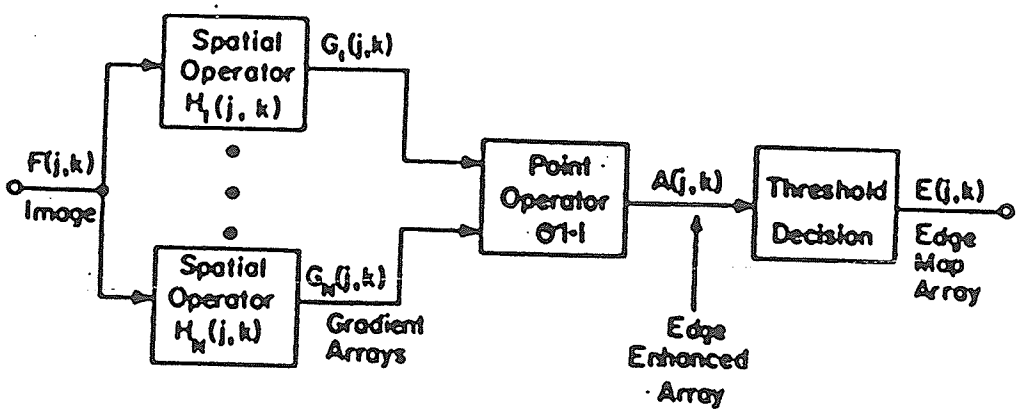


Fig.2-1. Edge enhancement-thresholding edge detection system.

An edge decision is made on the basis of the amplitude of A(j,k) with respect to a threshold(t). If

$$A(j,k) \geq t$$

an edge is assumed present, and if

$$A(j,k) < t$$

no edge is indicated.

The differential operators perform differentiation of an image array to produce a gradient field. This group includes the Roberts[14], Prewitt[15], and Sobel[16] operators. Among them, the 3X3 Sobel edge detector is considered one of the best edge detection operators in the 3X3 pixel differential class[13], and is chosen for this research.

Consider the 3X3 image region:

a	b	c
d	e	f
g	h	i

Define Gx as $Gx = (g+2h+i)-(a+2b+c)$

and Gy as $Gy = (c+2f+i)-(a+2d+g)$

where Gx represents an estimate of the derivative in the x direction. Similarly, Gy is an estimate of the derivative in the y direction.

For computational simplicity, a magnitude point nonlinearity is employed to produce an edge enhanced array

$$A(j,k) = |Gx(j,k)|+|Gy(j,k)|$$

The final result of detection is a binary picture, pixels deemed to be on an edge having the value one, all others having the value zero.

The selection of the threshold greatly affects the nature of the edge images. In this thesis, the optimum thresholds were found experimentally for each processed images.

Some of the difficulties in edge detection are caused by noise in the image. For improving the results of edge detection in noisy images, a substantial effort has been made to find filters that could both remove noise and preserve edges in the meantime.

In this thesis, an endeavor is made to utilize the Hough transformation[1] as a post filtering technique to improve edge detection results by removing impulse noise and keeping edges in the edge detected images.

Chapter III

HOUGH TRANSFORM

3.1 INTRODUCTION

Suppose we are given n points in an image and we are asked to find subsets of these points lying on straight lines. One practical approach is to find all subsets of points that are close to particular lines. However, the computation required for n points is approximately proportional to n^2, and may be prohibitive for large n.

This problem could be viewed in a different direction with an approach proposed by P.V.C.Hough, which is commonly referred to as the Hough transformation. The Hough transform technique[1,2], as suggested originally, is a method for detecting straight line segments in an input digitized image. In the simplest case, the picture contains binary gray levels, black image pixels lying on a white background or vice versa. Over the past decade, the Hough transform technique has been efficiently extended to the detection of parametric curves in images other than that of straight lines. With the appearance of more powerful computers, the technique hopefully will play a more important role in the search and detection of specific features from images.

3.2 PARAMETER SPACE AND IMAGE SPACE

Consider a point (x_i, y_i) and the general equation of a straight line in slope-intercept form, $y_i = ax_i + b$. There is an infinite number of lines that pass through (x_i, y_i), but all of them satisfy the equation $y_i = ax_i + b$ for varying values of **a** and **b**. However, if the equation $y_i = ax_i + b$ is rewritten as $b = -x_i a + y_i$, and considered in the **ab** plane(also called the parameter space), then the equation of a single line for a fixed pair (x_i, y_i) can be obtained. Furthermore, a second point (x_j, y_j) will also have a line in the parameter space associated with it, and this line will intersect the line associated with (x_i, y_i) at (a',b'), where **a'** is the slope and **b'** the intercept of the line containing both (x_i, y_i) and (x_j, y_j) in the **xy** plane. In fact, all points on this line will have lines in the parameter space that intersect at (a',b'). These concepts are illustrated in Fig.3-1.

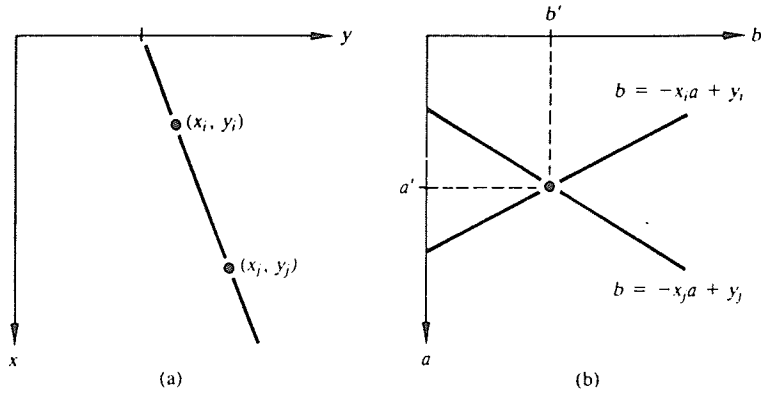


Fig.3-1.

A problem with using the equation $y = ax + b$ to represent a line is that both the slope and intercept approach infinity as the line approaches a vertical position. This difficulty can be avoided by employing the normal representation of a line, which may be parametrically demonstrated in Fig.3-2, as

$$x\cos\theta + y\sin\theta = \rho \qquad (3\text{-}1)$$

where ρ is the normal distance of the line from the origin and θ is the angle of the origin with respect to the x axis.

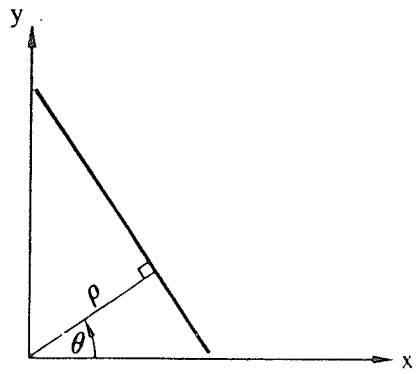


Fig.3-2.

With (3-1), however, the points on a line correspond to an infinite number of sinusoidal curves in the $\theta - \rho$ plane rather than straight lines.

The range of angle θ is from $0°$ to $180°$, measured with respect to the x axis. Thus with reference to Fig.3-2, a horizontal line has $\theta = 90°$, with ρ being equal to the positive y intercept. Similarly, a vertical line has $\theta = 0°$, with

ρ being equal to the positive x intercept, or $\theta = 180°$, with ρ being equal to the negative x intercept.

Some interesting properties of the point-to-curve transformation are summarized as follows[2]:

Property 1. A point in the picture plane corresponds to a sinusoidal curve in the parameter plane.

Property 2. A point in the parameter plane corresponds to a straight line in the picture plane.

Properties 1 and 2 are obvious and no proofs are presented here.

Property 3. Points lying on the same straight line in the picture plane correspond to curves through a common point in the parameter plane.

Proof:

Suppose there are two arbitrary points (x_i, y_i) and (x_j, y_j) lying on the straight line $y = ax + b$.

So, we have

$$y_i = ax_i + b \tag{1}$$

$$y_j = ax_j + b \tag{2}$$

Solve (1) and (2), a and b can be described by

$$a = \frac{y_i - y_j}{x_i - x_j} \tag{3}$$

and

$$b = \frac{x_i y_j - x_j y_i}{x_i - x_j} \tag{4}$$

In the Hough space, (x_i, y_i) and (x_j, y_j) correspond to two curves by applying $\rho = xcos\theta + ysin\theta$

$$\rho = x_i cos\theta + ysin_i\theta \tag{5}$$

$$\rho = x_j cos\theta + ysin_j\theta \tag{6}$$

Solve (5) and (6) to find the common point of two curves in the Hough space.

From (5) and (6),

$$x_i cos\theta + y_i sin\theta = x_j cos\theta + y_j sin\theta$$

$$tan\theta = \frac{x_i - x_j}{y_j - y_i}$$

So we have

$$tan\theta = -\frac{1}{a} \tag{7}$$

and

$$\theta = tan^{-1}(-1/a) \tag{8}$$

By using (7), we have

$$sin^2\theta = \frac{tan^2\theta}{1 + tan^2\theta}$$

$$sin^2\theta = \frac{1}{a^2 + 1}$$

Since $0 \leq \theta \leq \pi, \quad sin\theta \geq 0,$

So, $sin\theta$ can be represented by

$$sin\theta = \frac{1}{\sqrt{a^2 + 1}} \tag{9}$$

On the other hand, from (5) and (6), we have

$$x_j\rho = x_i x_j cos\theta + x_j y_i sin\theta \tag{10}$$

$$x_i\rho = x_i x_j cos\theta + x_i y_i sin\theta \tag{11}$$

or

$$\rho = \frac{x_i y_j - x_j y_i}{x_i - x_j} sin\theta$$

using the result of (4)

$$\rho = b sin\theta$$

So, ρ is described by

$$\rho = \frac{b}{\sqrt{a^2+1}} \tag{12}$$

Hence, we have proved that the common point of two curves in the parameter plane, which correspond to points (x_i, y_i) and (x_j, y_j) in the picture plane, is $(b/\sqrt{a^2+1}, tan^{-1}(-1/a))$. Because all points lying on the straight line $y = ax + b$ have the same a and b, so, the corresponding curves in the Hough space pass through **a common point**.

Property 4. Points lying on the same curve in the parameter plane correspond to lines through the same point in the picture plane.

Proof:

Fix x_0 and y_0, then two arbitrary points on curve $\rho = x_0 cos\theta + y_0 sin\theta$, say (ρ_i, θ_i) and (ρ_j, θ_j), present the following equations:

$$\rho_i = x_0 cos\theta_i + y_0 sin\theta_i \tag{13}$$

$$\rho_j = x_0 cos\theta_j + y_0 sin\theta_j \tag{14}$$

Next we consider two straight lines in the picture plane which correspond to two points (ρ_i, θ_i) and (ρ_j, θ_j) in the parameter space.

$$\rho_i = x cos\theta_i + y sin\theta_i \tag{15}$$

$$\rho_j = x cos\theta_j + y sin\theta_j \tag{16}$$

Because eqns (15) and (16) are linear equations, only one point of intersection exists (when $\theta_i \neq \theta_j$), therefore point (x_0, y_0) is the only common point of the two straight lines in the picture plane. So, all points on $\rho = x_0 cos\theta + y_0 sin\theta$ in the parameter space correspond to the straight lines which pass through the point (x_0, y_0) in the picture plane.

Applying above results, the problem of detecting collinear figure points in the image space may be carried out with significantly improved computational efficiency in the parameter space.

The computational attractiveness of the Hough transform emerges from quantizing the $\theta - \rho$ domain into **accumulator cells**, as illustrated in Fig.3-3.

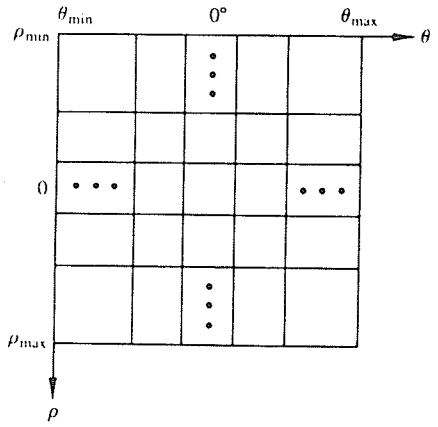
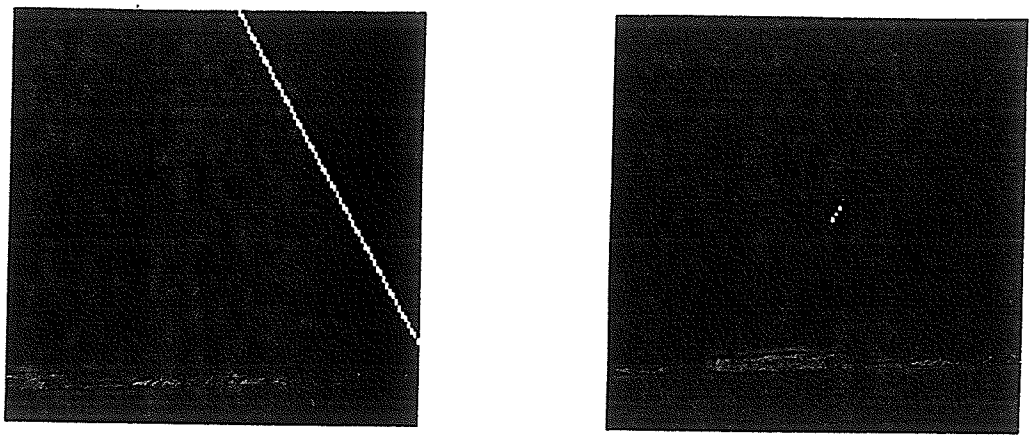


Fig.3-3

where $(\theta_{max},\theta_{min})$ and (ρ_{max},ρ_{min}) are the expected ranges of θ and ρ values. The cell at coordinates(i,j), with accumulator value A(i,j), corresponds to the square associated with parameter space coordinates(θ_i,ρ_j). Initially, these cells are set to zero. Then, for every point (x_k,y_k) in the image plane, the parameter θ is set at equal to each of the allowed subdivision values on the θ axis and ρ is solved for with the equation $\rho = x_k cos\theta + y_k sin\theta$. Then the resulting ρ's are rounded off to the nearest allowed quantized value in the ρ axis. If a choice of θ_p results in solution ρ_q, the value in accumulator A(p,q) is increased by one, A(p,q)=A(p,q)+1. At the end of this procedure, a value of M in A(i,j) corresponds to M points which lie on the line $\rho_j = xcos\theta_i + ysin\theta_i$ in the xy plane. The larger cell counts coincide with the greater possibility that collinear figure points may be fitted by a straight line with the (θ,ρ) parameters. The accuracy of the collinearity of these points is established by the number of subdivisions in the $\theta-\rho$ plane.

Fig.3-4 shows the transformation of a straight line between the image and the parameter space. Fig.3-4(a) shows the line in the image space. Fig.3-4(b) presents the loci of the parameter lines, which were generated by edge pixels in Fig.3-4(a), in the Hough space. Fig.3-4(c) is parts of corresponding accumulator cells. In Fig.3-4(c), accumulator A(60,120) has the maximum value 118. It indicates that there are 118 edge pixels lying on the line $xcos60^\circ + ysin60^\circ = 120$.



(a) (b)

ρ

0	0	0	0	11	30	19
0	0	0	0	58	26	11
0	0	0	0	46	9	0
0	0	4	118	3	0	0
1	9	49	0	0	0	0
13	29	52	0	0	0	0
21	29	13	0	0	0	0

(57,117) θ

(c)

Fig.3-4 (a)The original image, (b)Mapping image in the Hough space, (c)Parts of accumulator cells.

It is clear that the general transform approach can be extended to curvers other than straight lines. For example, the attention is focused on circle detection in images. In Cartesian coordinates, a circle is described by

$$(x-a)^2 + (y-b)^2 = r^2 \qquad (3-2)$$

For each edge pixel, if it is to lie on a circle, the locus for the parameters of the circle will be a right circular cone which is shown in Fig.3-5. This can be seen from equation (3-2) by treating x and y as fixed and letting a, b, and r vary.

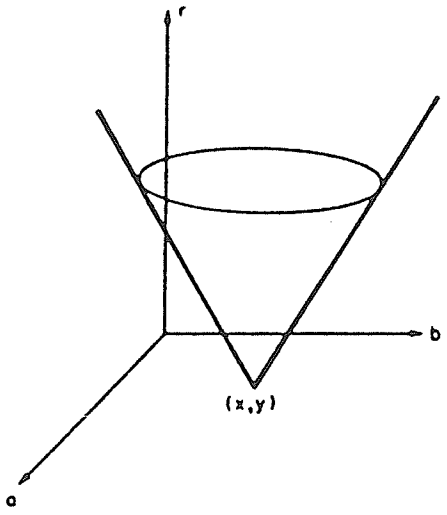
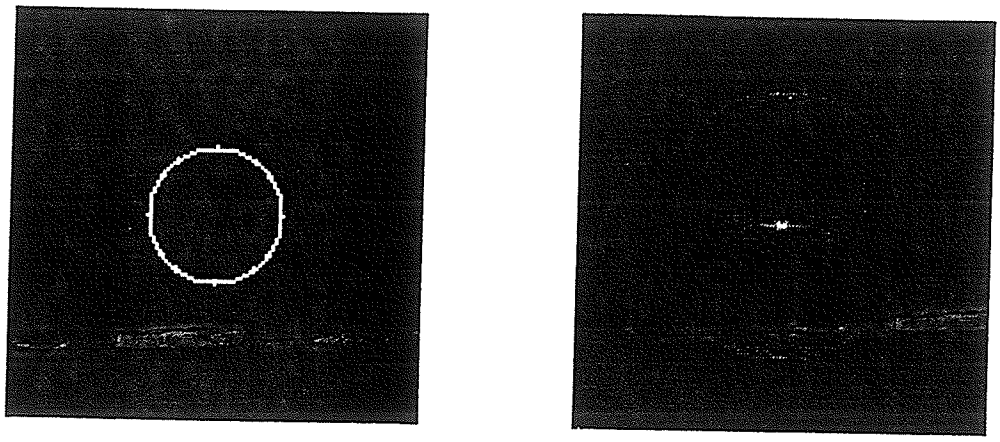


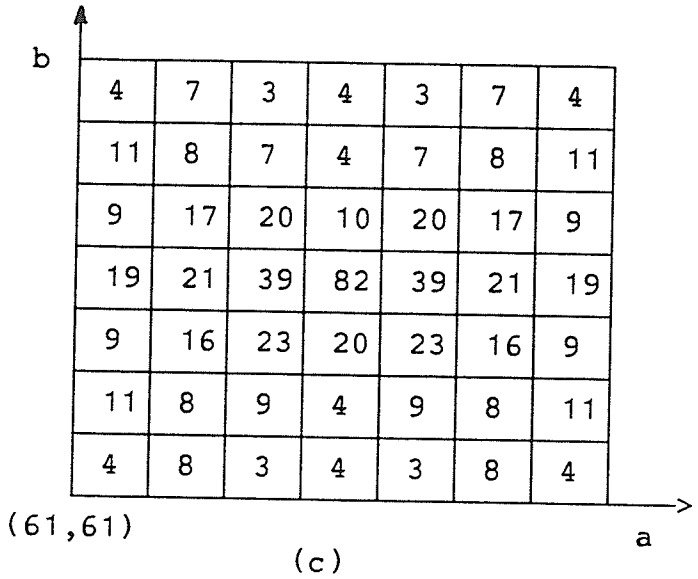
Fig.3-5

In the image space, if a set of edge pixels in an image are arranged on a circle with parameters a_0, b_0 and r_0, the resultant loci of parameters for each such point will pass through the same point (a_0,b_0,r_0) in the parameter space.

With a procedure similar to the case of straight lines, a three-dimensional array of accumulators is employed to represent the three-dimensional parameter space. Each edge pixel in the image space is transformed to a right circular



(a) (b)



(c)

Fig.3-6 (a)A circle with r=19 in the image space. (b)Mapping image in the Hough space. (c)Parts of accumulator cells.

cone in the a-b-r space and the corresponding accumulator, say $A(a_0,b_0,r_0)$, is increased. After all pixels are transformed, a value of M in $A(a_0,b_0,r_0)$ corresponds to M points in the xy plane lying on the circle

$$(x-a_0)^2 + (y-b_0)^2 = r_0^2 \qquad (3-3)$$

An example is given in Fig.3-6 to explain the above conception. Fig.3-6(a) shows a circle in the image space with its centre at a=64, b=64 and r=19. Fig.3-6(b) illustrates the mapping of the circle in the Hough space with fixed r=19, say H(a,b,19). In Fig.3-6(b), it can be seen that the loci of parameters for each edge pixel passed through the same point (64,64,19) in the Hough space. Parts of accumulator cells are displayed in Fig.3-6(c). The value of accumulator A(64,64,19) is 82, it means that there are 82 edge pixels lying on the circle $(x-64)^2+(y-64)^2=19^2$ in the image space.

In principle, the Hough transform method can be extended to arbitrary curves. It only asks to pick a convenient parameterization for the family of curves of interest and proceed in the obvious way.

Chapter IV

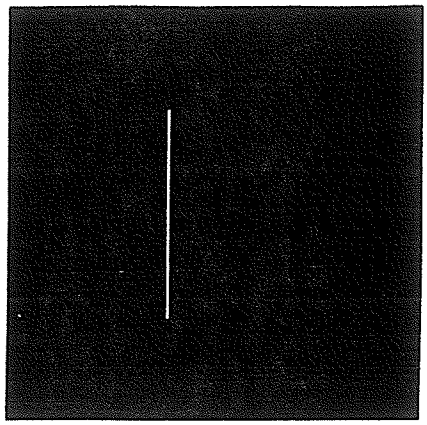
PRELIMINARIES AND SUPPORTING RESULTS

4.1 LOGIC MATCHING

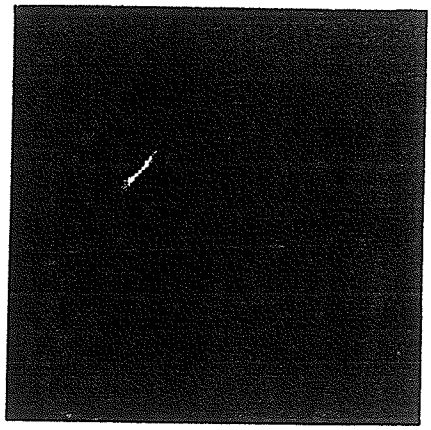
One of the problems associated with the Hough transform technique is that a uniform point (θ,ρ) of the parameter space results in nonuniform inverse curves in the image space. The connected problem is that the Hough transform approach finds the collinear pixels without regard to their continuity. Consider a straight line in the image space. The Hough transform maps each single edge pixel on the line of the image space to a point (θ,ρ), say $H(\theta,\rho)$, in the Hough space by applying (3-1). The edge pixels lying on the same straight line will be mapped to the same point $H(\theta,\rho)$. At the end of the procedure, the value of accumulator $A(\theta_0,\rho_0)$ equates to M means that M edge pixels lie on the straight line

$$\rho_0 = x\cos\theta_0 + y\sin\theta_0 \qquad (4-1)$$

in the image space. However, (4-1) does not give any information about which parts of the line the M edge pixels lie on. Three cases are shown in Fig.4-1 to explain the above conception. Fig.4-1(a), (d) and (g) are three different



(a)



(b)

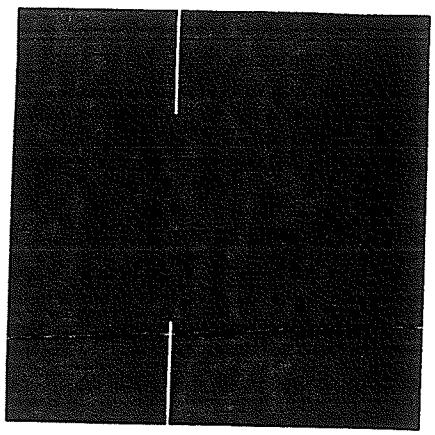
ρ

0	0	0	0	0	26	19
0	0	0	0	11	28	14
0	0	0	0	53	10	0
0	0	0	64	0	0	0
0	11	54	0	0	0	0
17	29	10	0	0	0	0
19	24	0	0	0	0	0

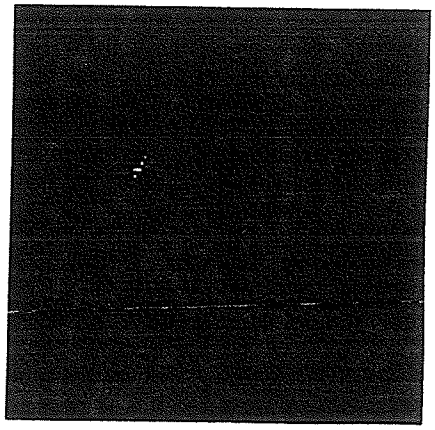
(87,47)

θ

(c)



(d)



(e)

Fig.4-1. (a), (d) and (g) are three different 128X128 edge images. (b), (e) and (h) are the mapping images of (a), (d), (g) in the Hough space respective. (c), (f) and (i) are parts of accumulator cells of (b), (e) and (h).

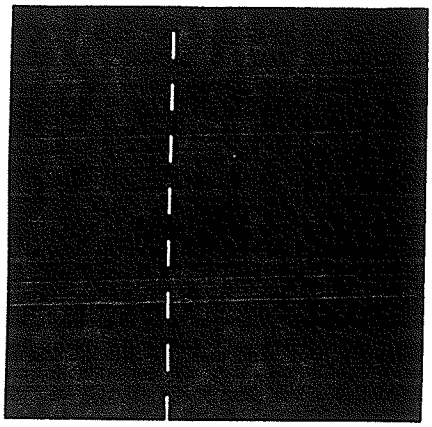
0	0	0	0	0	3	0
0	0	0	0	32	0	5
0	0	0	0	4	19	19
10	15	29	64	28	13	8
19	17	3	0	0	0	0
3	0	32	0	0	0	0
0	5	0	0	0	0	0

ρ

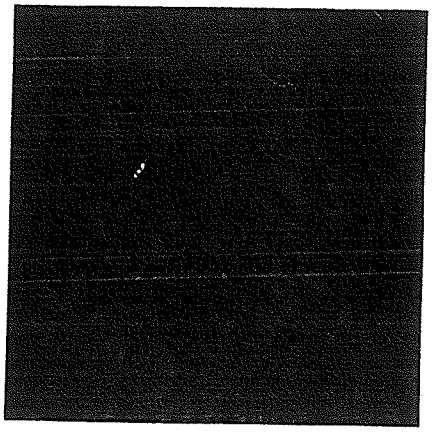
(87,47)

θ

(f)



(g)



(h)

0	0	0	0	0	16	10
0	0	0	0	24	14	11
0	0	0	0	28	13	11
2	7	13	64	12	5	0
11	12	27	0	0	0	0
11	13	24	0	0	0	0
8	16	0	0	0	0	0

ρ

(87,47)

θ

(i)

original images in the image space. Fig.4-1(b), (e) and (h) are the mapped images of (a), (d) and (h) in the Hough space, respectively. Fig.4-1(c), (f) and (i) illustrate parts of accumulator cells of (b), (e) and (h). To study the accumulator cells presented in Fig.4-1(c), (e) and (h), a meaningful reality is that each case has the same maximum value 64 in the accumulator $A(90^{0},50)$. It indicates that in each image there are same number of edge pixels on the line $50=x\cos 90^{0}+y\sin 90^{0}$.

The question is that from $A(90^{0},50)=64$, none of the three original images can be obtained directly from the inverse of the Hough space. The inverse image of $H(90^{0},50)$ is displayed in Fig.4-2, which, obviously, can not replace any case in Fig.4-1(a), (d) and (g). All three original images of Fig.4-1 are subsets of Fig.4-2.

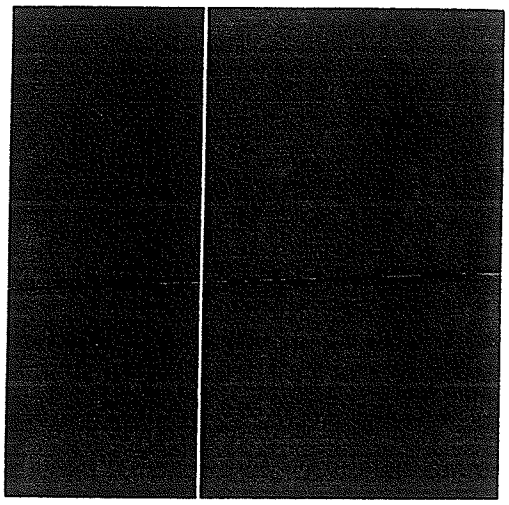


Fig.4-2 The inverse image of $H(90^{0},50)$.

As Duda and Hart point out, the remarkable problem above is related to the limitations of the Hough transform approach[2]. To apply the Hough transform as a filtering procedure, this problem must be solved. Otherwise, the Hough transform can not perform well as a filter to extract specific features from a noisy edge detected image.

Suppose the background pixels in the image space have value zero. The inverse of $H(\theta_0,\rho_0)$ determines a straight line, which is filled with the pixels having value one, in the image space by equation (4-1).

An important but easily neglected fact is: in the original edge image and the image obtained from the inverse Hough transform, the candidate edge element has value one in both images. On the other hand, if a pixel is not a real edge pixel, at least one of its positions in both images has the value zero. Therefore, employing the logic **AND** procedure to match two images, the true edge element points will be kept while others be removed.

$$g(i,j) = f_0(i,j) \cdot f_I(i,j) \qquad (4-2)$$

where $f_0(i,j)$ is original edge detected image and $f_I(i,j)$ is the inverse Hough transform image.

Applying (4-2) to Fig.4-1(a),(d) and (h) with the inverse image of $H(90^0,50)$ which is shown in Fig.4-2, as expected, the results are **exactly the** same images as the original Fig.4-1(a), (d) and (h).

Evidently, the logic AND matching procedure can be extended to arbitrary curves with the utilization of the Hough transform. For example, it may be employed to extract candidate edge pixels from the inverse image of H(a,b,r) in circles searching, where H(a,b,r) is a point in the 3-D(a-b-r) Hough space.

In this research, the logic matching approach will be applied as an important part in the Hough transform filtering procedure.

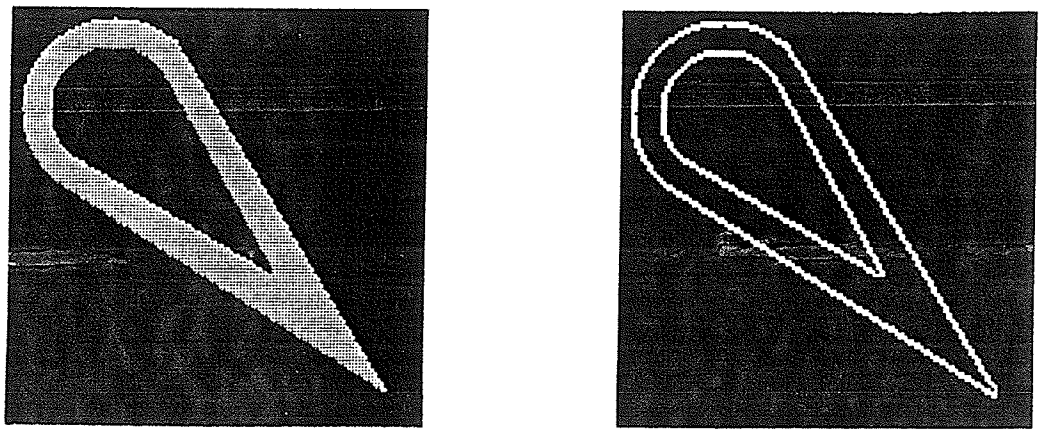
4.2 TEST IMAGE

Straight lines and circles are two most important geometric features in the image structure. A large class of man-made objects and natural scenes can be modelled with these primitives. The recognition of these features is therefore basic to many image analysis tasks. This thesis will base mainly on these two features to present the Hough transform technique as an effective method to remove noise from edge detected images.

Though the program package developed for this research is available for both 256X256 pixels, 65 gray-level and 128X128 pixels, 33 gray-level images, for the purpose of avoiding unnecessary computational expense, the 128X128 pix-

els with 33 gray-level images are chosen in most cases. However, any result produced from 128X128 images can be extended to the case of 256X256 images automatically. The only difference between 128X128 and 256X256 system is that for any process, the computation time cost is approximately quadrupled for the latter.

A 128X128 pixels, 33 gray levels image, consisting of straight lines and parts of circles, is generated, which will be employed as the test image in the next chapter. The test image and its noise free edge version are shown in Fig.4-3.



(a) (b)

Fig.4-3. (a) The 128X128 test image displayed in 33 gray levels. (b) The original image detected by the Sobel edge detector with t(threshold)=11.

4.3 FIGURE OF MERIT COMPARISON

How well does the Hough transform filtering approach improve edge detected images? Compared with the same noise-free ideal edge image, a new **figure** of **gain factor**(FGF) is defined to assess the performance of the Hough transform filtering

$$FGF = \frac{N_{O1} + \gamma N_{O2}}{N_{H1} + \gamma N_{H2}} \qquad (4-3)$$

where, N_{O1} and N_{O2} are the number of background pixels added and the number of target pixels removed from the original edge detected image, respectively. N_{H1} and N_{H2} have the same meaning as N_{O1} and N_{O2} in the image produced by the Hough transform technique. γ is a constant chosen to be 0.7 in this research. If γ is less than 1.0, it provides a relative penalty to isolated pixels in background; if γ is greater than 1.0, the penalty will be supplied to the pixels removed from edges.

The equation (4-3) will proffer a merit dealing with approximately how many times the processed image has been improved. If nothing has been changed, the value of FGF will be 1.0. An improved image will correspond to a greater than 1.0 FGF result. The better an image is improved, the higher the FGF value will be.

4.4 HOUGH TRANSFORM PROGRAMMING

A Fortran version of the Hough transform program package was implemented on an IBM compatible Amdahl 5870 computer under the MVS operating system.

For more accurate results, a higher resolution in the $\theta-\rho$ domain[2,12] is employed. For straight line searching, the $H(\theta,\rho)$ transform space was computed by equation (3-1) with $\Delta\theta=1^{\circ}$ and $\Delta\rho=1$(pixel). The origin has been allocated in the left-up corner (1,1) of the 128X128 image frame. The Hough accumulator array (θ,ρ) is thus of size (181,361), where the range of θ goes from 0° to 180° and ρ from -180 to 180. For circle searching, the 3-D H(a,b,r) transform space was determined by equation (3-2) with $\Delta a=1$, $\Delta b=1$ and $\Delta r=1$, where the unit is pixel. Considering the limit of computer's memory, the size of the Hough accumulator array (a,b,r) is bounded by (128,128,30). If the range of r is over 30, the circle searching subroutine may be called more than once to fit the requirements.

In the Hough transform program package, the inverse $H(\theta_0,\rho_0)$ transform is obtained by equation (4-1) with fixed θ_0 and ρ_0. Similarly, the inverse $H(a_0,b_0,r_0)$ transform is secured from equation (3-3) with fixed a_0, b_0 and r_0.

When the Hough transform procedure is applied to detect the straight lines, an important concept has to be pointed

out: the value M in an accumulator cell $A(\theta_0, \rho_0)$ does not reflect the actual **length** of the line defined by (4-1), even when all pixels are connected. Fig.4-4 will help to show **the concept.**

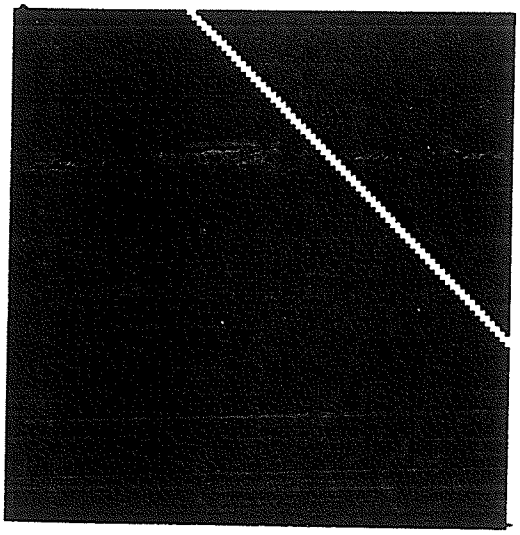


Fig.4-4

There is a straight line defined by

$$x\cos 135^0 + y\sin 135^0 = 45 \qquad (4\text{-}4)$$

in the Fig.4-4. After the Hough transform is applied, the value in the accumulator $A(135^0, 45)$ will be 129. However, this just means that there are 129 pixels lying on the line (4-4), but does not reflect the actual **length** of the line which approximates to $65 \cdot \sqrt{2}$. In this thesis, the Hough transform program package is based on **pixels** other than **length** in the thresholding process, though the two can be integrated by introducing a coefficient.

A primary subject in the Hough transform programming is the coincidence between the Hough transform and the inverse Hough transform. In the Hough transform program package, this key point is well guaranteed.

A listing of the Fortran programs is available in Appendix B.

4.5 COMPUTATION TIME REQUIREMENTS

The main disadvantage of the Hough transform is that it is a time consuming procedure. However, this research shows that the computation time based on the Amdahl 5870 computer is still acceptable.

In the Hough transform filtering procedure, the computing time is mainly based on three factors: the number of searched edge element points, the resolution in the parameter space, and, in the circle searching, the range of candidate circle's radius. For more accurate results, the minimized step(one pixel) is chosen for $\Delta\rho$, Δa, Δb, Δr, and 1^{0} is selected for $\Delta\theta$ in the Hough transform programming package. Based on them, the time-consuming curves for straight lines and circles searching procedures are illustrated in Fig.4-5.

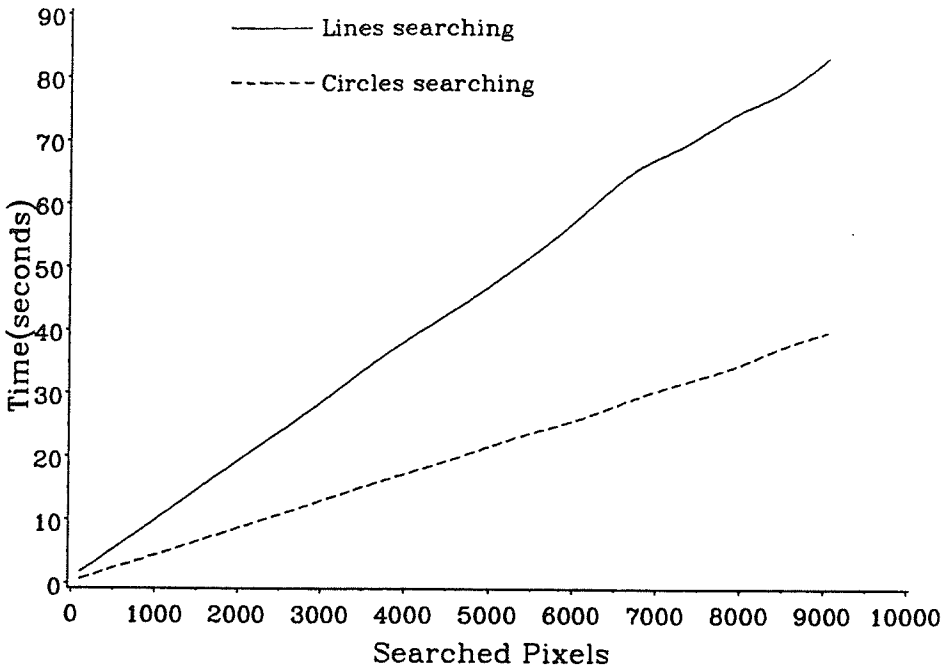


Fig.4-5

In Fig.4-5, the curve for circles searching is based on per radius. For example, if the range of the radius for circles searching is 3, say from r=20 to r=22, the actual computation time will triple that illustrated in Fig.4-5.

The Hough transform filtering procedure is a somewhat expensive searching approach, but is still acceptable on the Amdahl 5870 computer.

Chapter V

NOISE PERFORMANCE

Images are subject to many different types of noise. Some of these are independent of the image signals, while others are not. In this chapter, some physical noise models that are frequently encountered in practical imaging systems are used to examine the efficacy of the Hough transform filtering approach.

Furthermore, the results of the Hough transform procedure are compared with those of the Median filter technique, which is considered one of the most powerful filters dealing with impulse noise[17].

5.1 INDEPENDENT NOISE

5.1.1 Additive Gaussian Noise

Independent random noise u(i,j) with a Gaussian distribution and of zero mean and different variance was added to each pixel in the test image by

$$g(i,j) = f(i,j) + u(i,j) \qquad (5\text{-}1)$$

where f(i,j) is the original image and g(i,j) is the image degraded by the additive Gaussian noise.

According to Eq.(5-1), several noisy images with different variance σ^2 are displayed in Fig.5-1. Applying the Sobel edge detection operator to each noisy image in Fig.5-1, the edge detected images are shown in Fig.5-2. Since the purpose of the testing is to examine the best performance of the Hough transform filtering, the parameters in the inverse Hough transform are tuned to their best capabilities. Fig.5-3 presents the inverse Hough transform images. Then, the logic **AND** matching procedure is employed, the final images obtained are shown in Fig.5-4. For comparison, the 3X3 Median filter was applied to each image in Fig.5-2. The results are displayed in Fig.5-5.

The difference between Fig.5-2 and Fig.5-4 is obvious, the images are enormously refined by the Hough transform filtering process. The attractive fact is that the results from the Hough transform procedure are quite stable with the rise of the noise level.

Compare Fig.5-4 with Fig.5-5, though the Median filter gives a strong improvement to the noisy edge images, the Hough transform, however, offers an even better result.

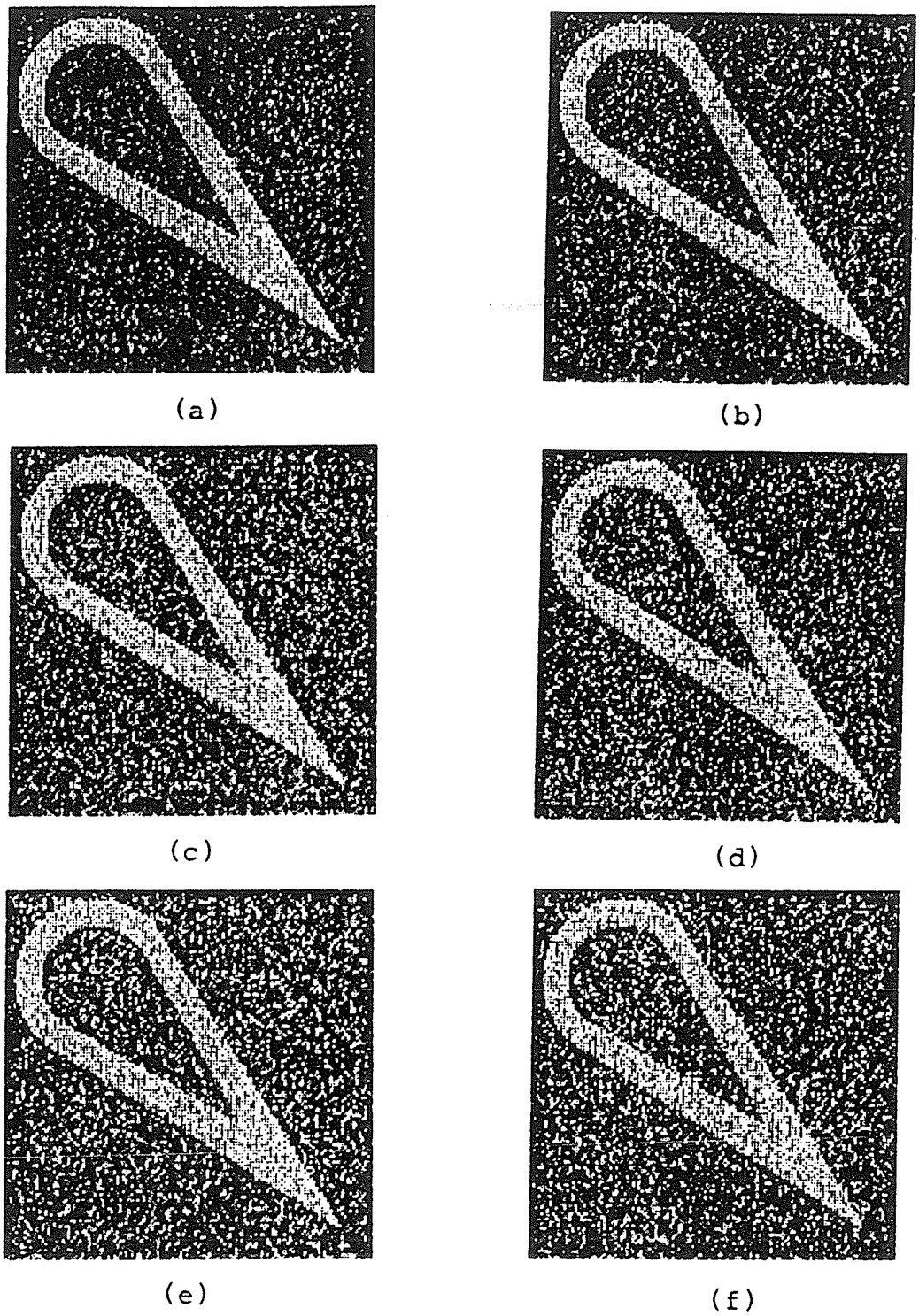
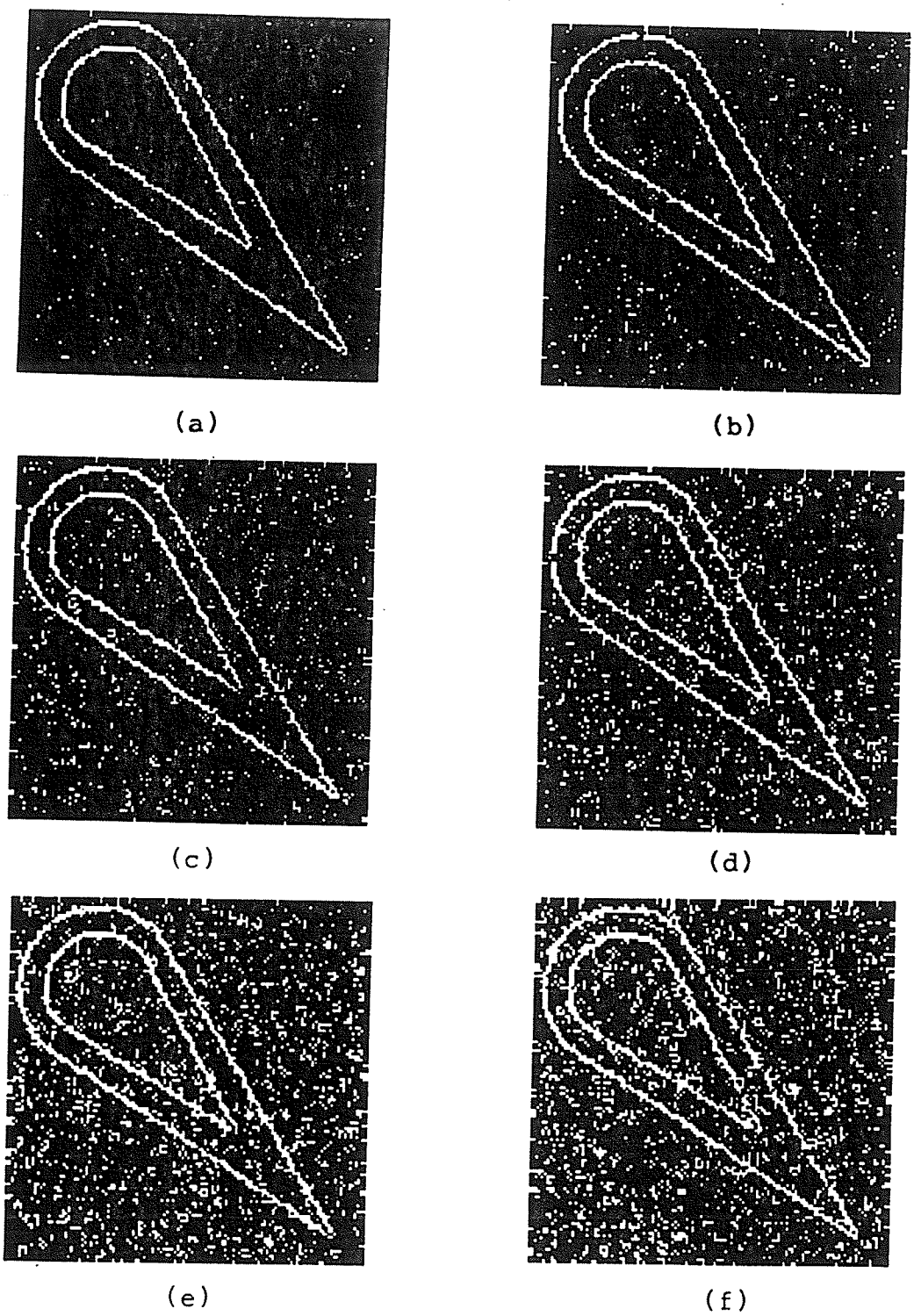


Fig.5-1 Original images degraded by Gaussian noise. From (a) to (f), σ^2 = 10,14,18,22,26,30.

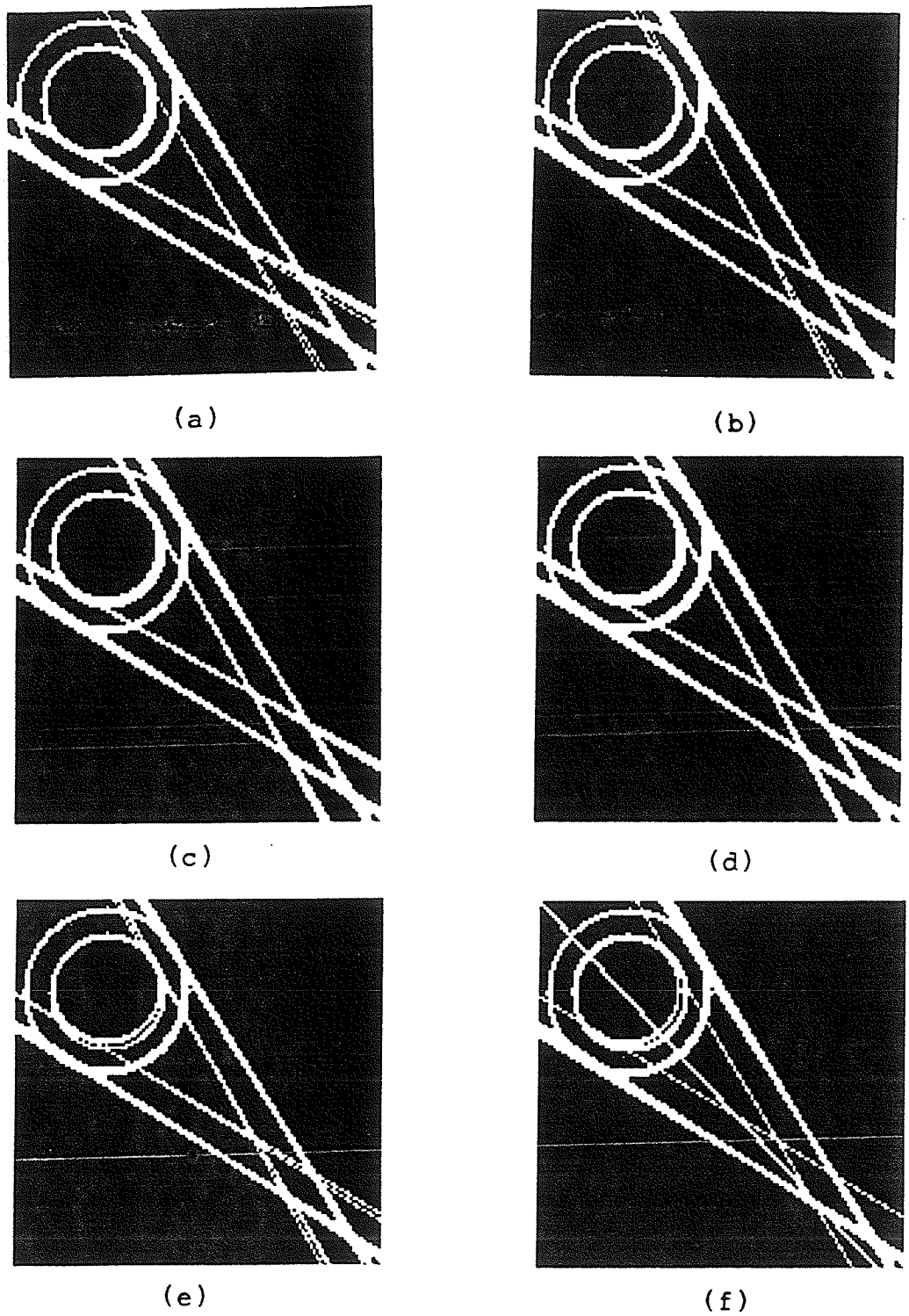


(a) (b)

(c) (d)

(e) (f)

Fig.5-2 The edge detected images of Fig.5-1. By applying the Sobel edge detector with thresholding value 11.



(a) (b)

(c) (d)

(e) (f)

Fig.5-3 The inverse Hough transform images.

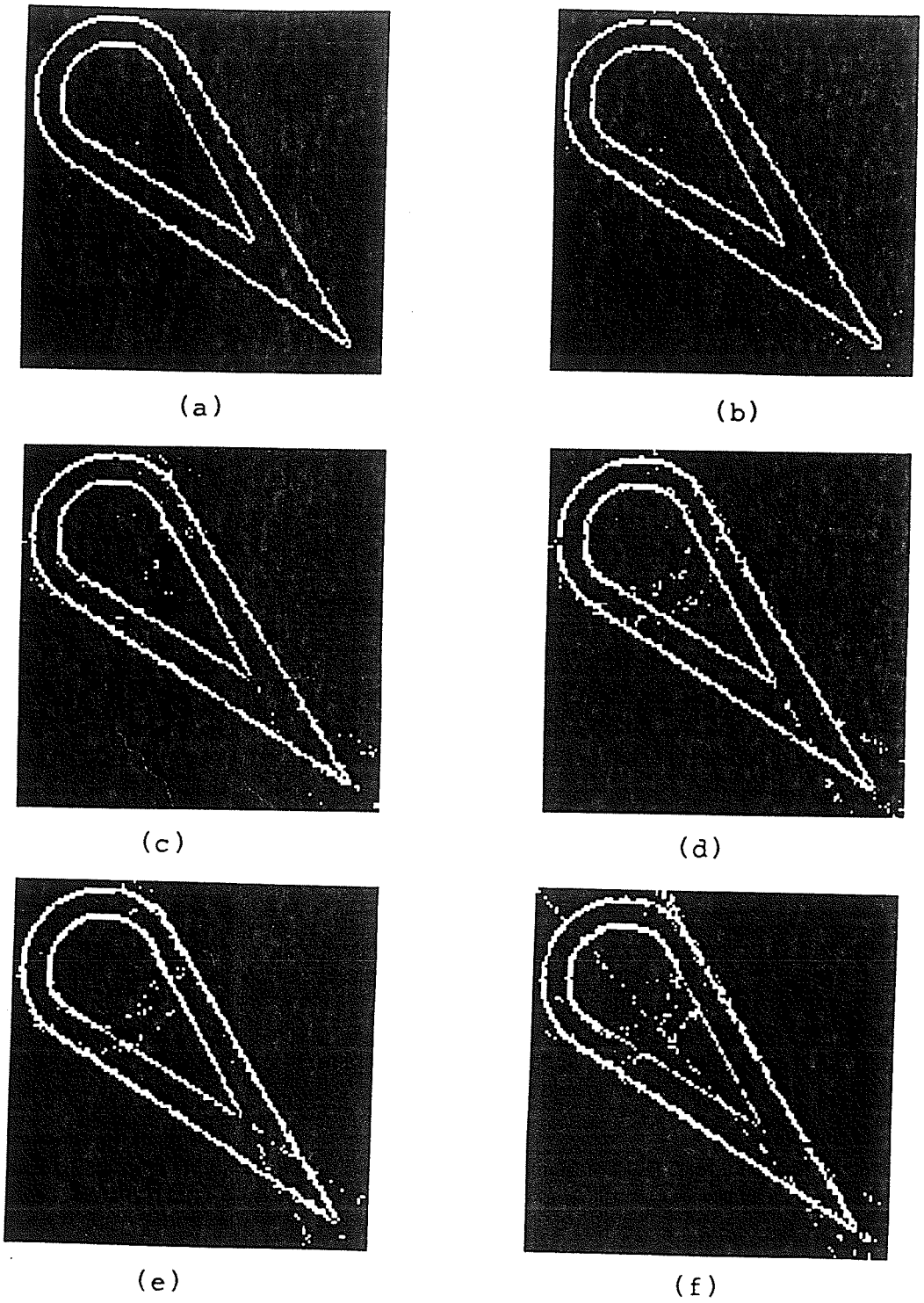


Fig.5-4 The final images by employing the Hough transform filtering procedure to the images in Fig.5-2.

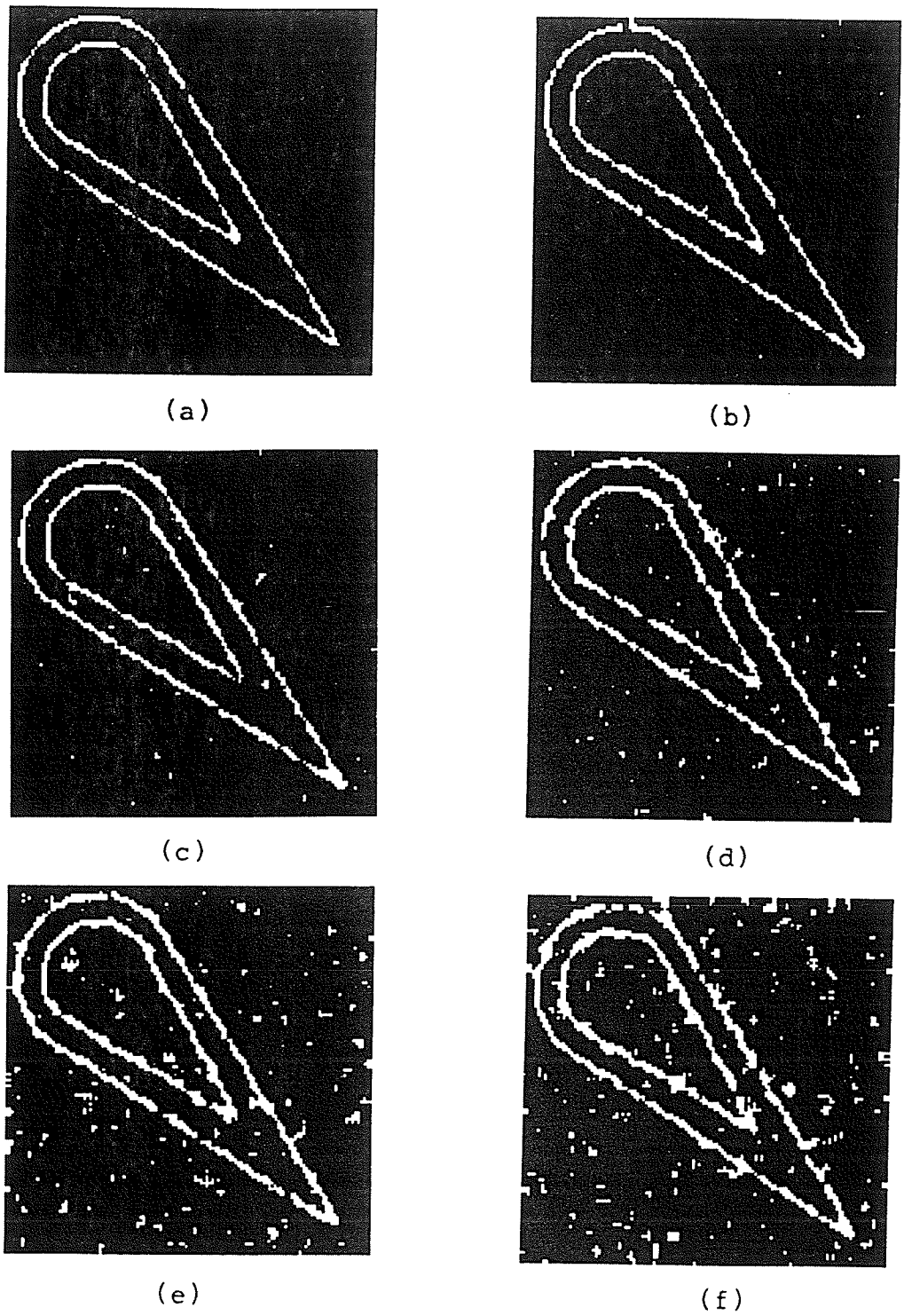


Fig.5-5 The results of applying the 3X3 Median filter to the images in Fig.5-2.

5.1.2 Salt-and-Pepper Noise

An important type of independent noise corroding digital images is salt-and-pepper noise. Generally, the salt-and-pepper noise is a kind of noise in which scattered pixels of an image are markedly darker or lighter than their immediate surroundings. Fig.5-6 is an example. In this research, the maximum gray level(33) is chosen as the white and the minimum gray level(1) as the black. In Fig.5-6, the gray level of each dark pixel has been incremented while that of the light pixel has been decremented, by a random amount z in the range $0 \le z \le \theta(\mathbf{l}-\mathbf{d})$, where $\mathbf{l}$ and $\mathbf{d}$ are the gray levels of the light and dark region in the test image respectively, and $0.0 \le \theta \le 1.0$. The values of θ chosen in Fig.5-6 are 0.40, 0.45, 0.50, 0.55, 0.60, 0.65 in order.

Fig.5-7 and Fig.5-8 display the noisy edge detected images and the final results obtained by applying the Hough transform filtering procedure to Fig.5-5, respectively. The 3X3 Median filter is employed to refine the degraded images in Fig.5-5 as well, the results are presented in Fig.5-9.

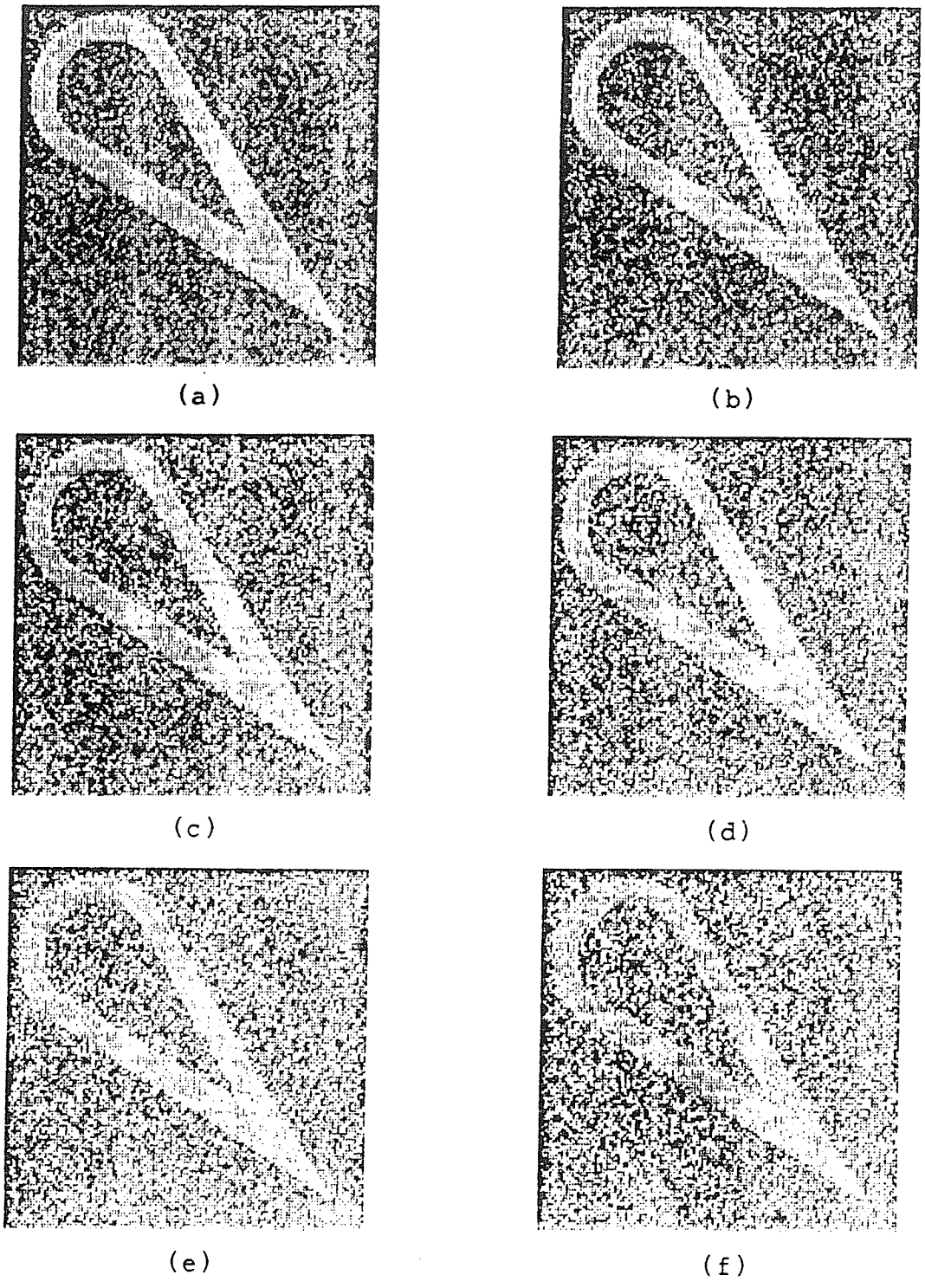


Fig.5-6 The original images with salt-and-pepper noise. From (a) to (f), θ = 0.40, 0.45, 0.50, 0.55, 0.60, 0.65.

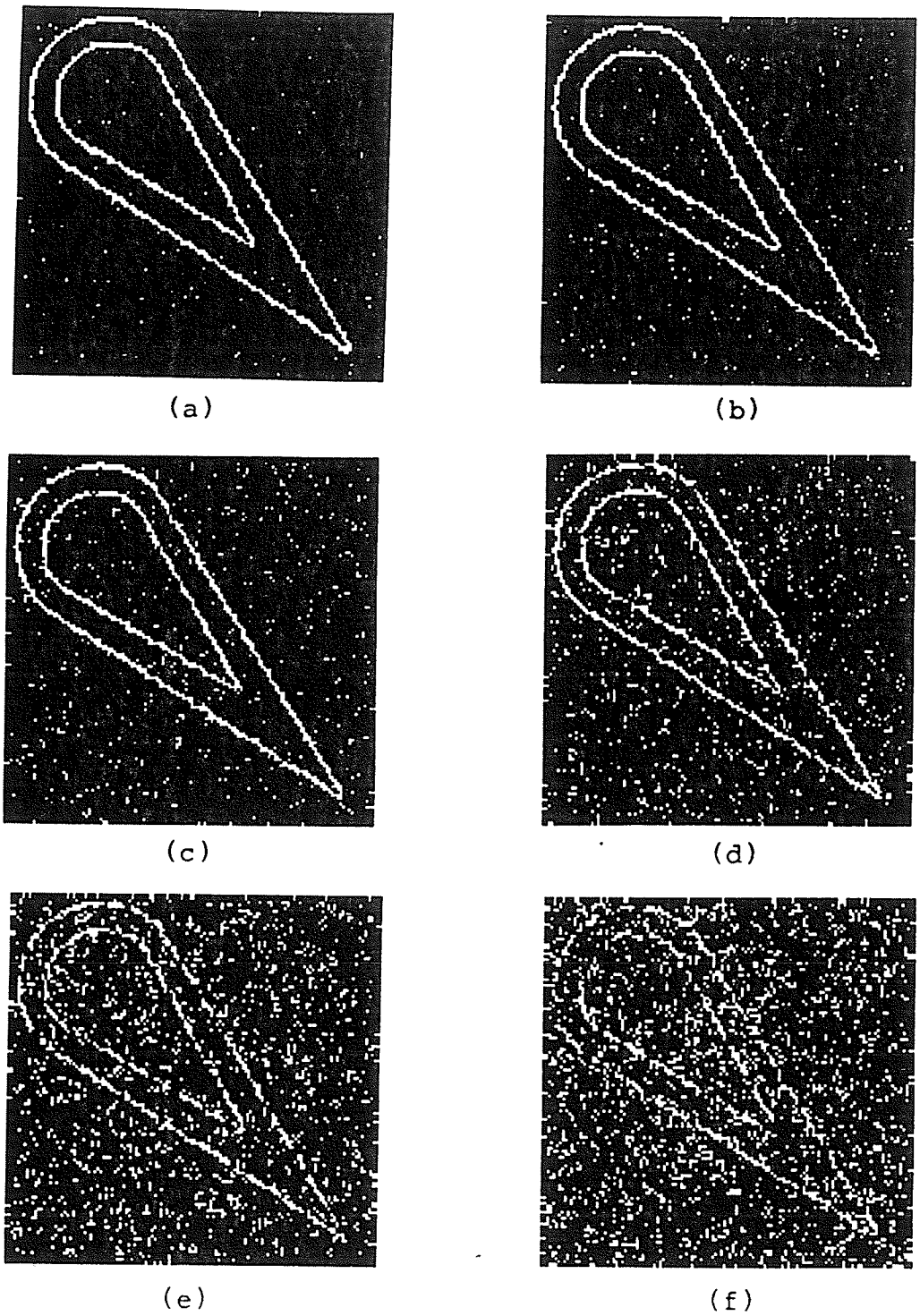


Fig.5-7 The edge detected images by applying the Sobel edge detector with t=6.

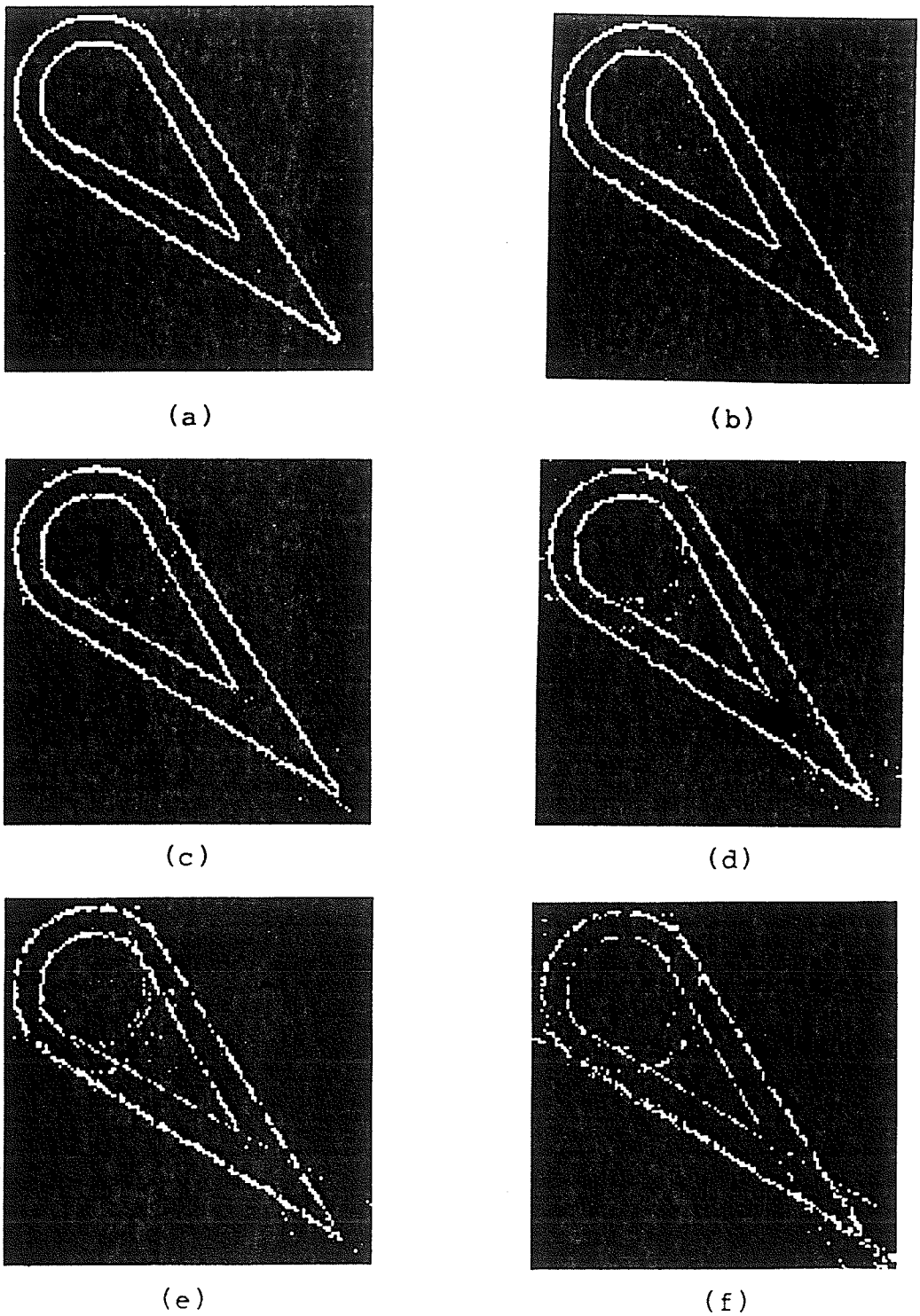


Fig.5-8 The final results by applying the Hough transform filtering procedure to the images in Fig.5-6.

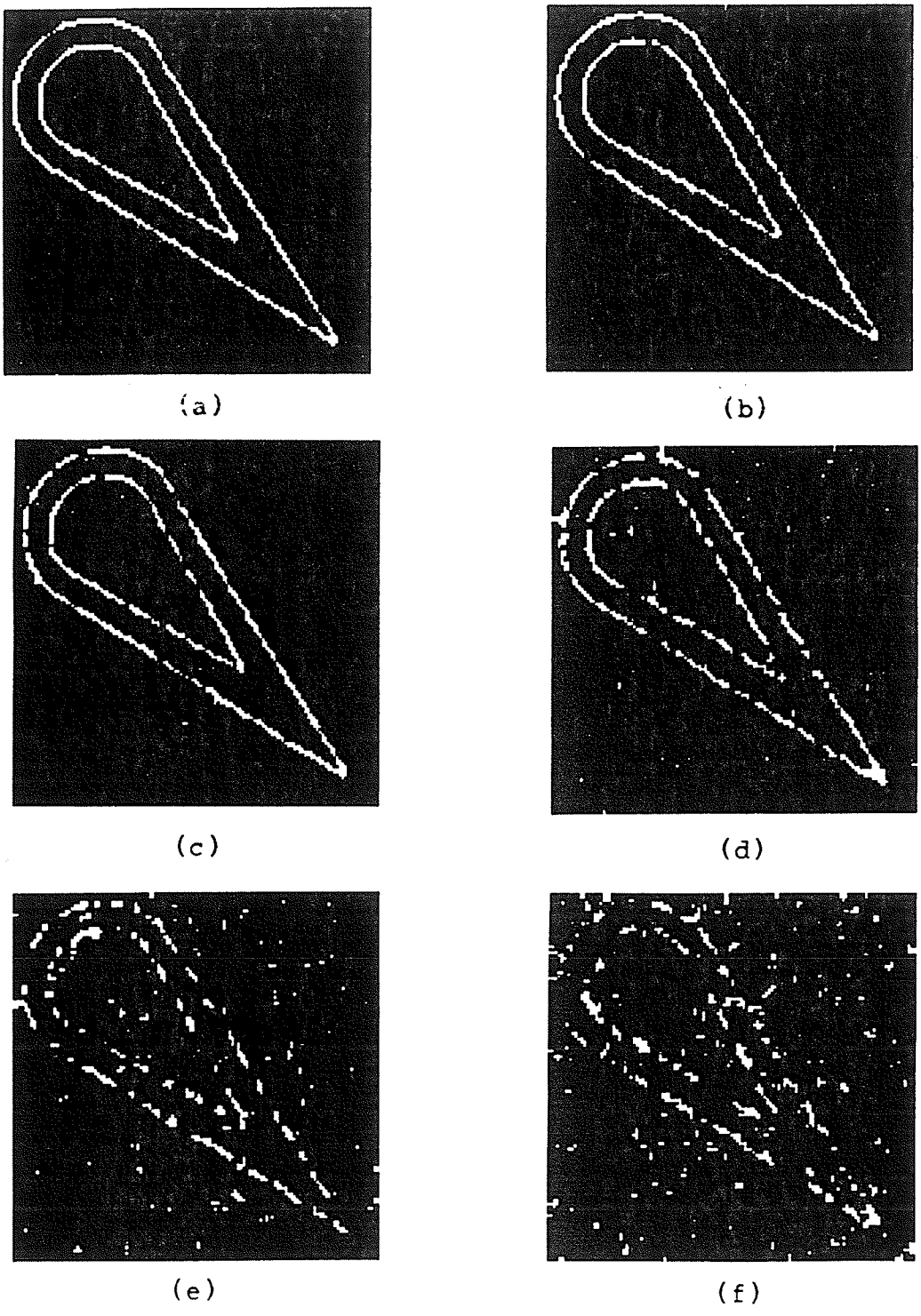


Fig.5-9 The result images by employing the 3X3 Median filter to the images in Fig.5-6.

5.2 DEPENDENT NOISE

5.2.1 Film-Grain Noise

Film-grain noise is a kind of dependent noise which exists in the process of photographic recording and reproduction. If the film is processed in the linear region of the D-log E curve and the blurring effect of the model[19] is ignored, the degraded image is presented as

$$g(i,j) = f(i,j) + kf^{1/3}(i,j)u(i,j) \qquad (5\text{-}2)$$

where $u(i,j)$ is a signal-independent noise and k is a constant. In this thesis, Gaussian noise with zero mean and different variance is employed as $u(i,j)$ and constant k equates to 1.0.

The images degraded by the film-grain noise are illustrated in Fig.5-10. The edge detected versions of Fig.5-10 are presented in Fig.5-11. Fig.5-12 and Fig.5-13 are the inverse Hough transform images and the final results of the Hough transform filtering procedure, respectively.

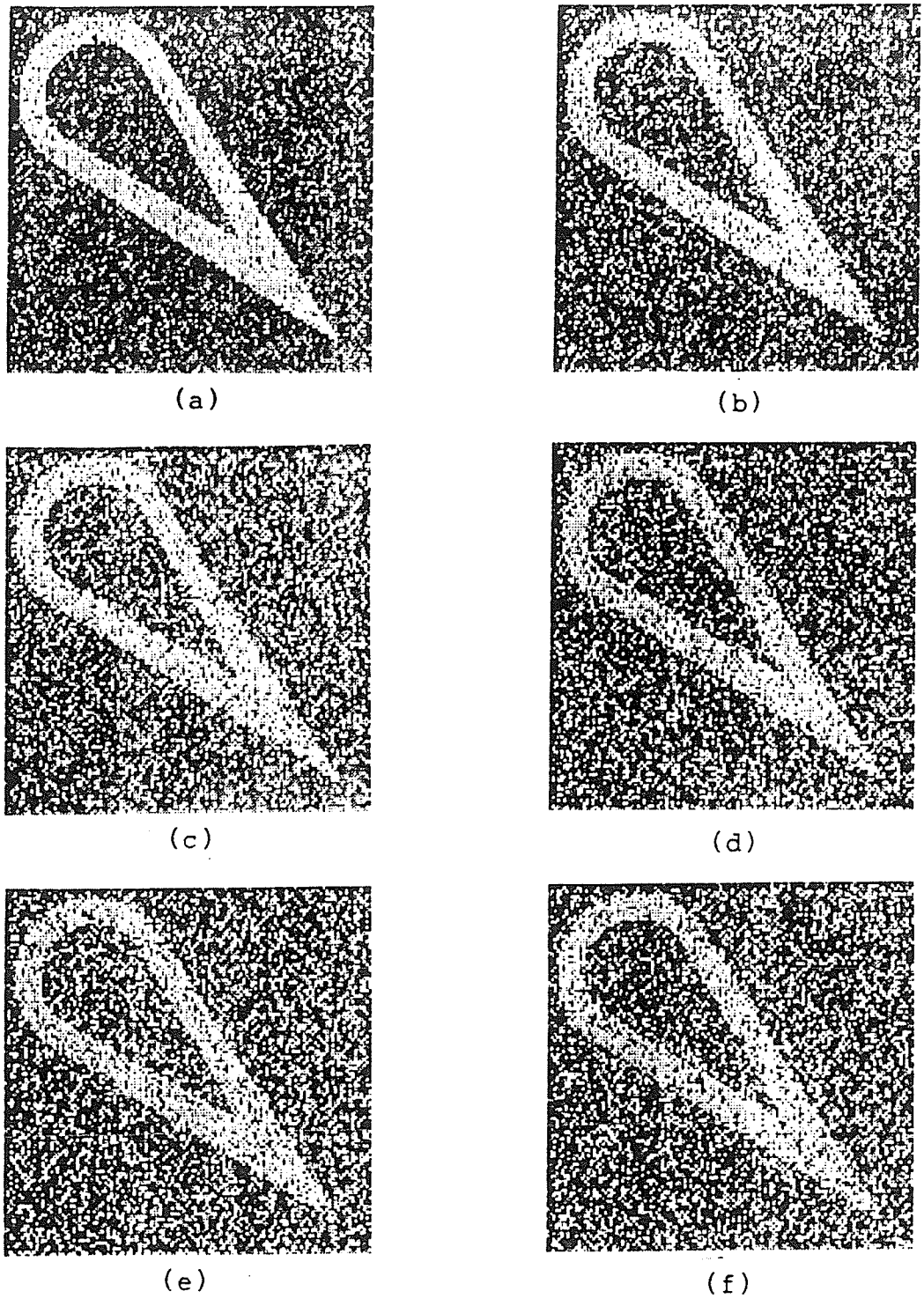


Fig.5-10 The original images degraded by film-grain noise. From (a) to (f), σ^2 = 6.0, 10.0, 14.0, 16.0, 18.0, 20.0.

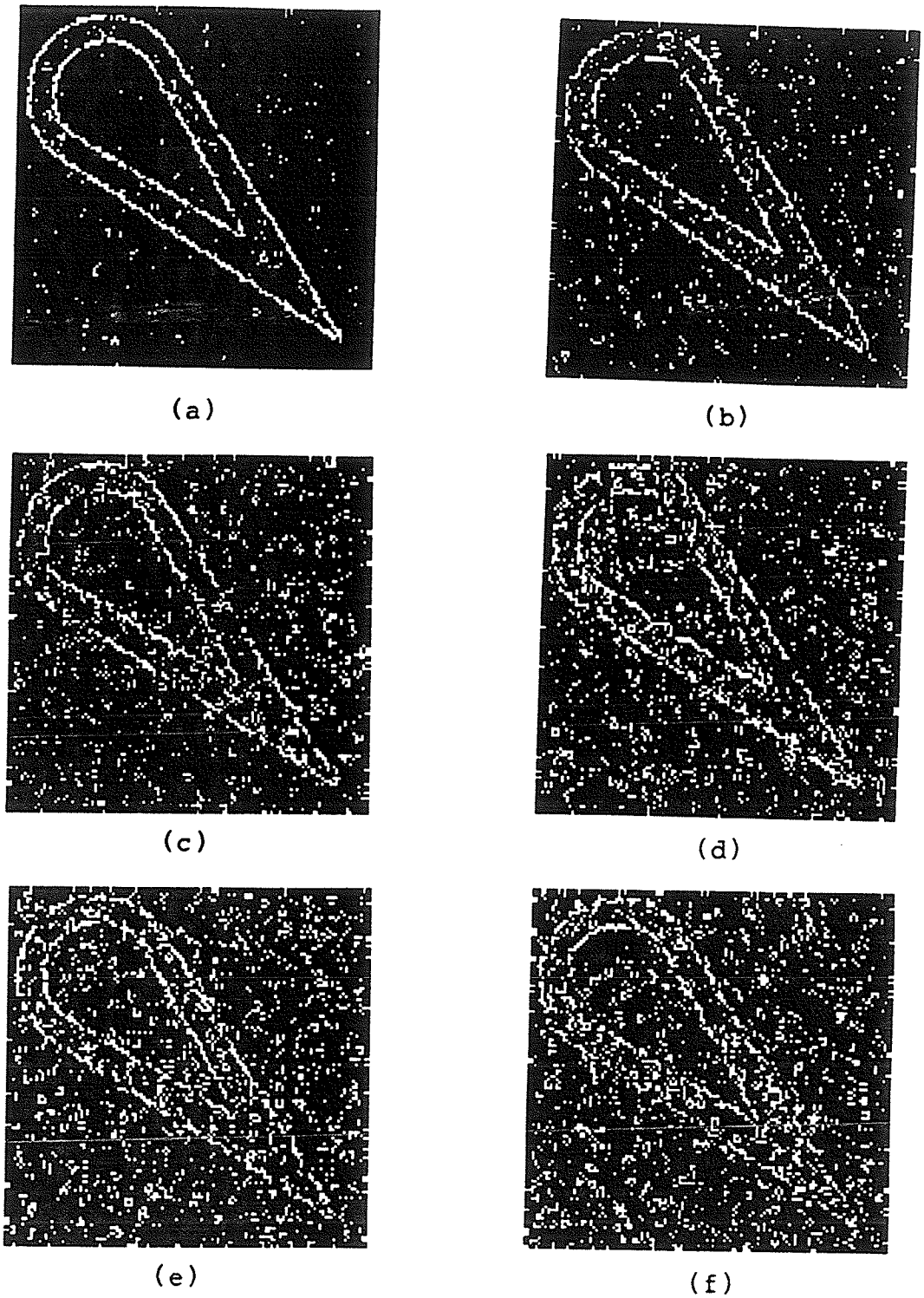


Fig.5-11 The edge detected images with the Sobel edge detector, t=11.

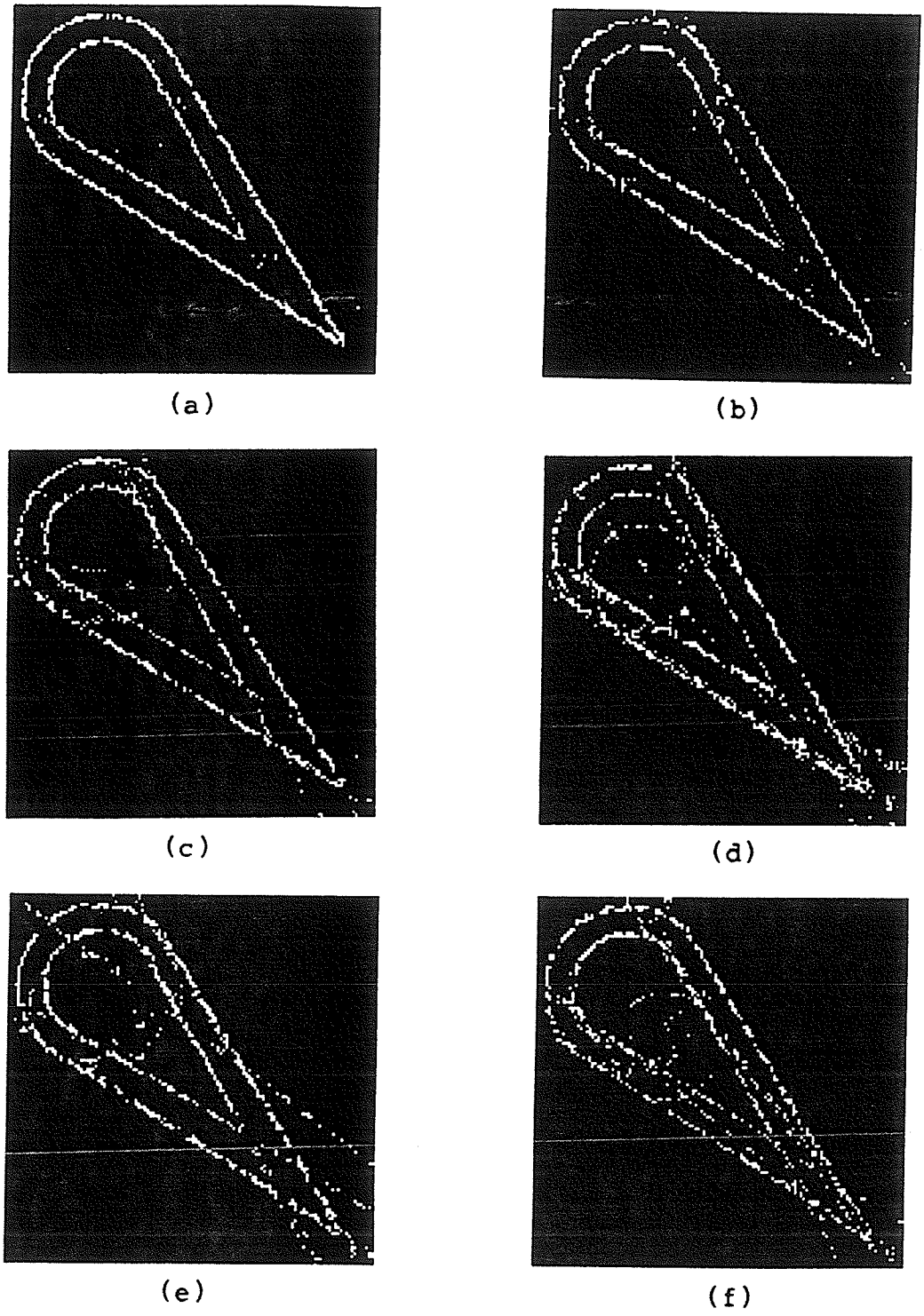


Fig.5-12 The final images by applying the Hough transform filtering procedure.

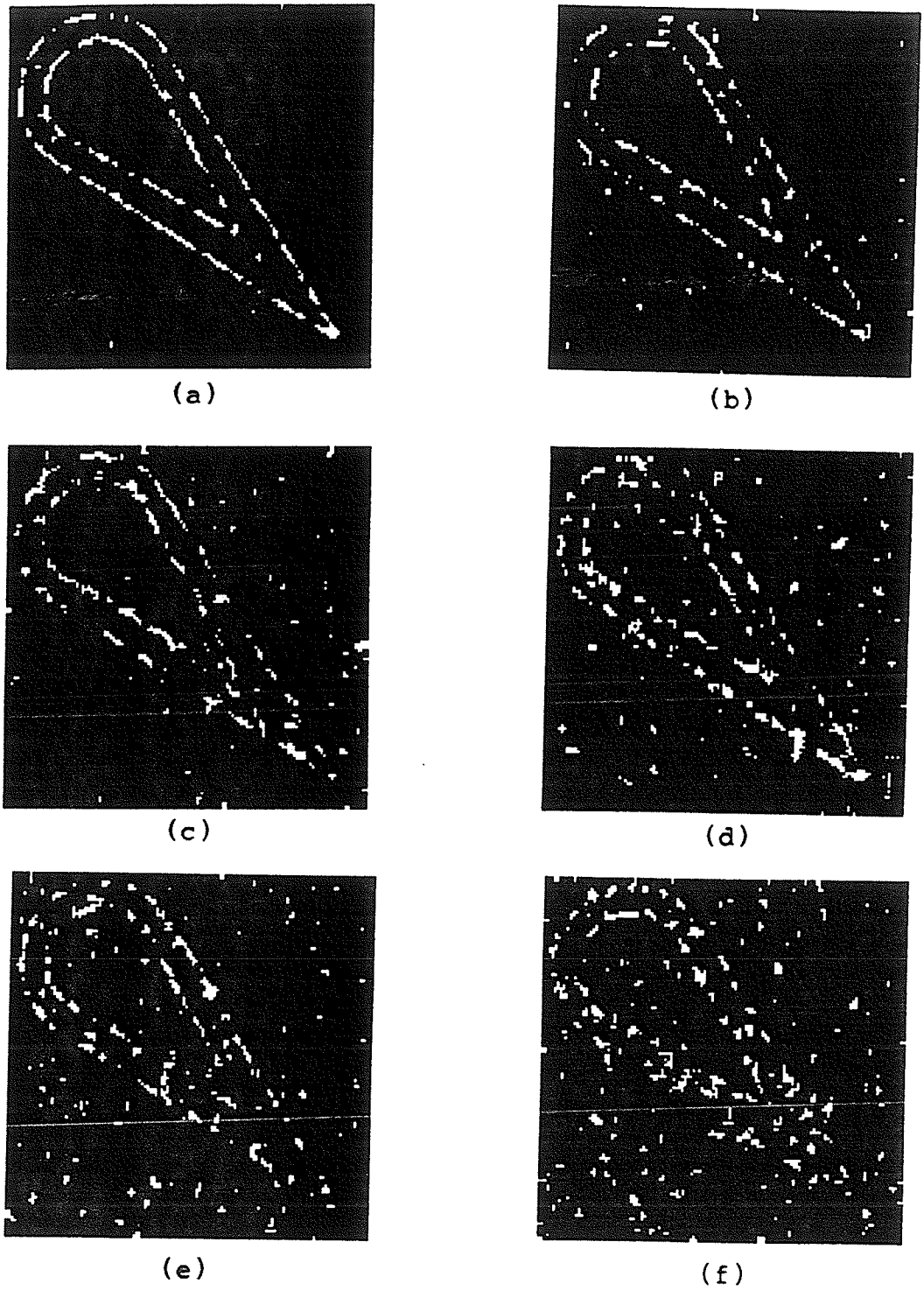


Fig.5-13 The results of applying the 3X3 Median filter to the images shown in Fig.5-11.

5.2.2 Poisson Noise

In nuclear medical imaging, the emission of gamma-rays is described by a Poisson process, and the resulting measurement uncertainty, the statistical quantum fluctuations of the counting rate, is called Poisson noise[22].

The degradation model of Poisson noise is given by

$$g(i,j) = \text{Poisson}_{\lambda}(f(i,j)) \qquad (5\text{-}3)$$

where $\text{Poisson}_{\lambda}(.)$ is a Poisson random number generator, and λ is a proportionality factor. In the thesis, λ=1.0 is chosen.

The picture degraded by Poisson noise is used here as the test image to examine the Hough transform filtering procedure. Some results are illustrated in Fig.5-14.

Fig.5-14(a) shows the original noise free image. Fig.5-14(b) displays the image degraded by Poisson noise with λ=1.0. The edge detected version of Fig.4-14(b) is presented in Fig.5-14(c). Fig.5-14(d) and Fig.5-14(e) are the inverse Hough transform image and the final result of applying the Hough transform filtering procedure to (c), respectively. Also, the 3X3 Median filter is employed to improve the degraded edge image, Fig.5-14(f) is the result.

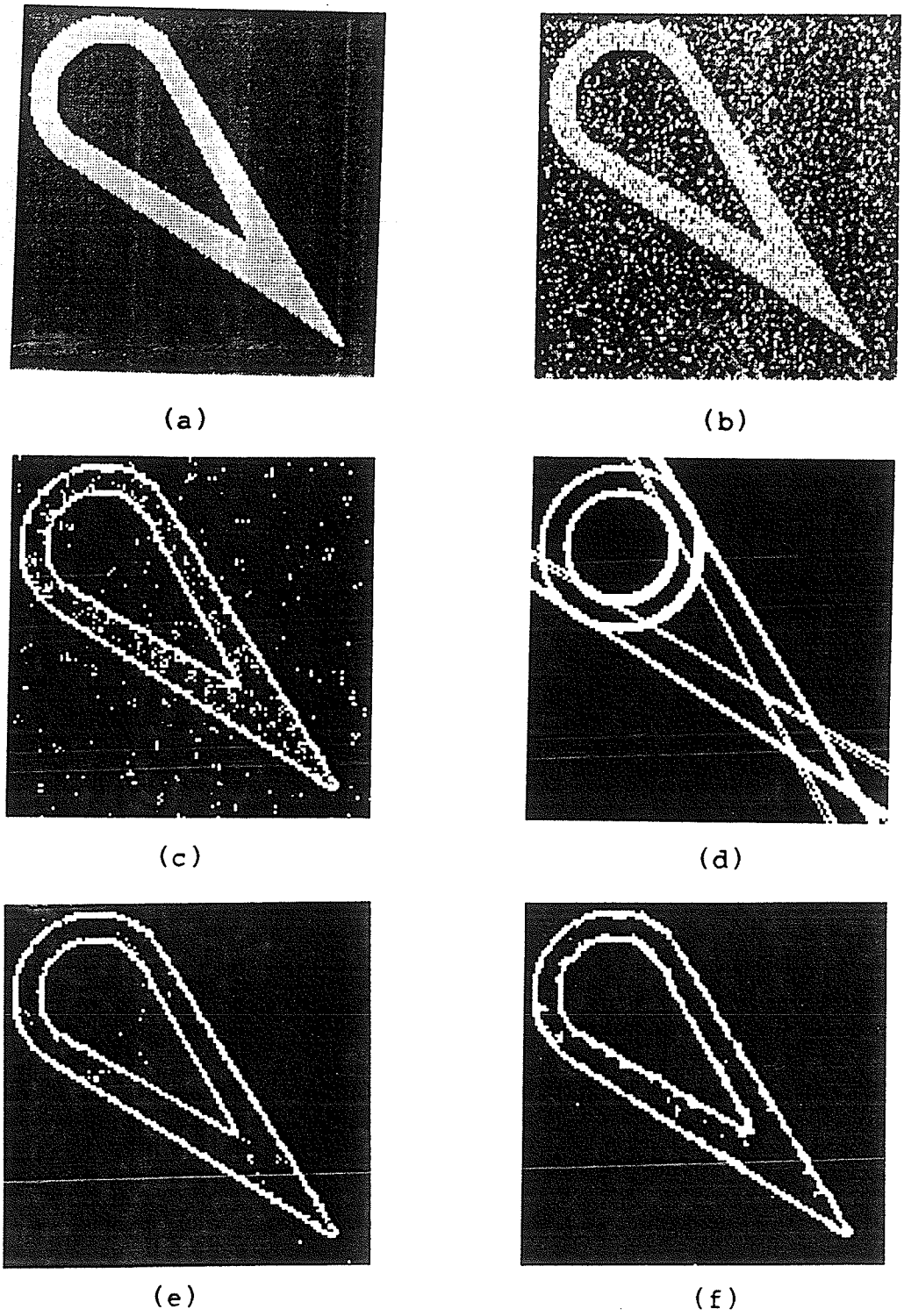


Fig.5-14 (a)The original noise free image. (b)Degraded by Poisson noise with λ=1.0. (c)Edge detected version. (d)The inverse Hough transform image. (e)The final result image by applying the Hough transform filtering procedure. (f)The result by employing the 3X3 Median filter to (c).

5.3 COMPARISON AND ANALYSIS

The Median filter has been considered as an "ad hoc tool" for removing impulse noise[3], so, it will be meaningful to present a comparison between the Hough transform filtering procedure and the Median filter technique. Employing the **figure of gain factor**(FGF), which was defined in section 4.3, the performances of the Hough transform filtering and the Median filtering technique are illustrated as the functions of different noise level.

Fig.5-15 and Fig.5-16 present the curves of FGF for the independent additive Gaussian noise and salt-and-pepper noise, respectively. Fig.5-17 shows the curves of the FGF as a function of variance σ^2 for the dependent film-grain noise.

Since all three types of noise were generated in a random way, the procedures were repeated ten times and the average values were employed for more reasonable results.

No matter which kind of noise, independent or dependent, the curves in Fig.5-15, Fig.5-16 and Fig.5-17 indicate that the Hough transform filtering procedure provides a substantially greater advantage than the Median filter does. The results from the Hough transform filtering procedure are still quite dramatic in the increased noise cases while those of the Median filtering are going down.

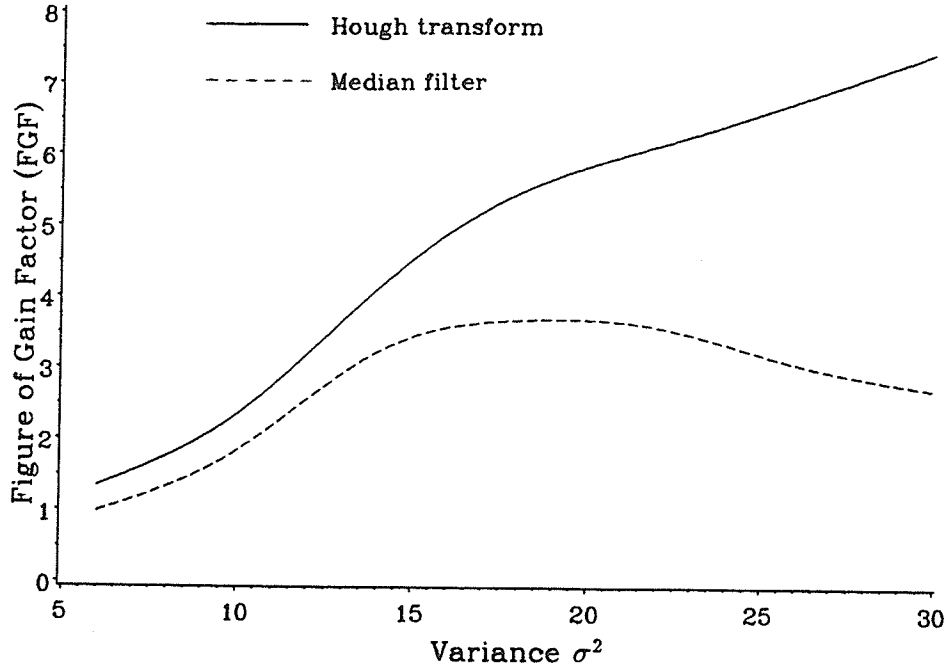


Fig.5-15

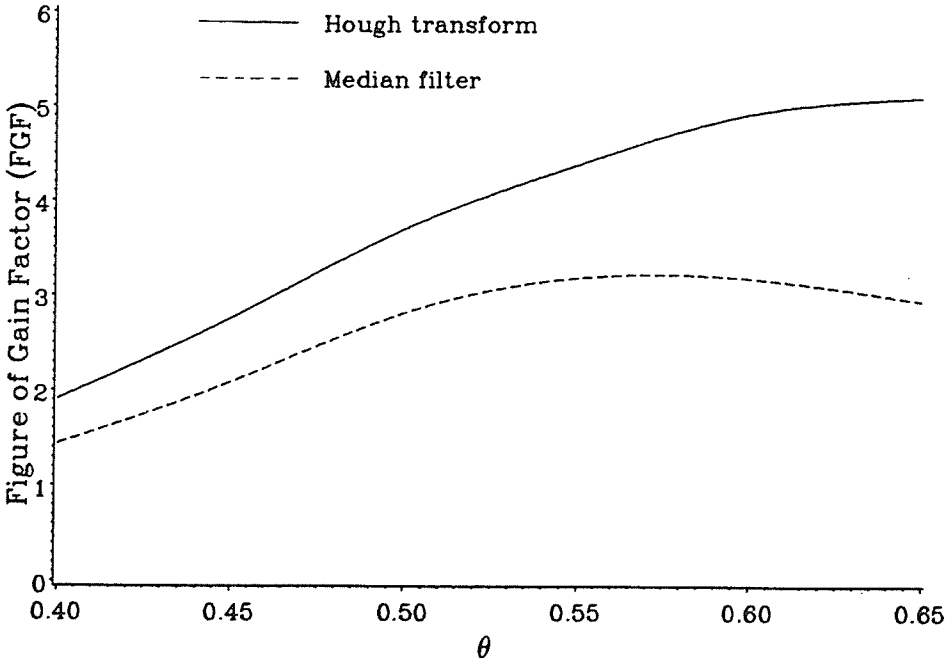


Fig.5-16

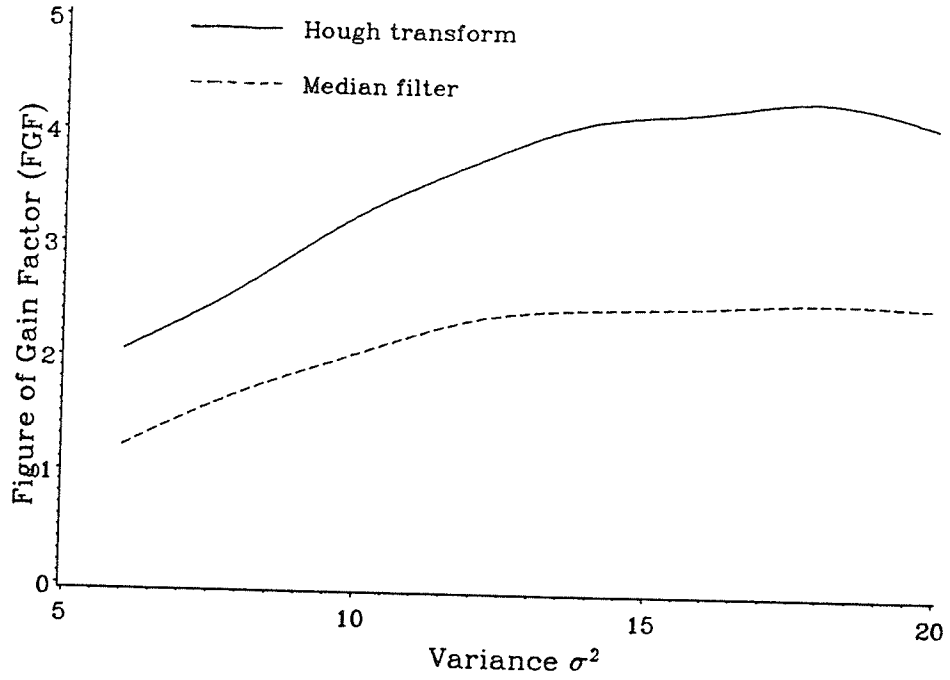


Fig.5-17

As seen from Fig.5-15 through Fig.5-17, there is a considerable amount of improvement in the FGF under different type of noise. This signifies that the Hough transform filtering procedure is not sensitive to the state of noise, so it can be applied as a general filtering technique to extract specific geometric features from the noisy binary image.

One of the weak points of the Hough transform filtering procedure is that while the procedure can remove added noisy pixels from images effectively, it can not make up the pixels which were taken from edges by noise, though these pix-

els may be controlled in a small number by selecting the edge detection threshold carefully.

From Fig.5-15 through Fig.5-17, it is concluded that the Hough transform procedure is able to offer a better improvement when the input images getting worse. In other words, the approach has great potential dealing with poor signal to noise ratio(SNR) input, while the Median filtering technique keeps an approximate constant improvement to different SNR input.

The signal to noise ratio(SNR) is described as[5]

$$SNR = \frac{N}{N_1 + N_2} \qquad (5\text{-}4)$$

where N is the number of boundary pixels on the noise-free target, N_1 is the number of background pixels added and N_2 is the number of target pixels removed.

For the images with the additive Gaussian noise, when $\sigma^2 = 30$, the SNR of Fig.5-2(f) is 0.3305. Thus, the observed performance of the Hough transform filtering procedure is excellent in the case of poor input SNR.

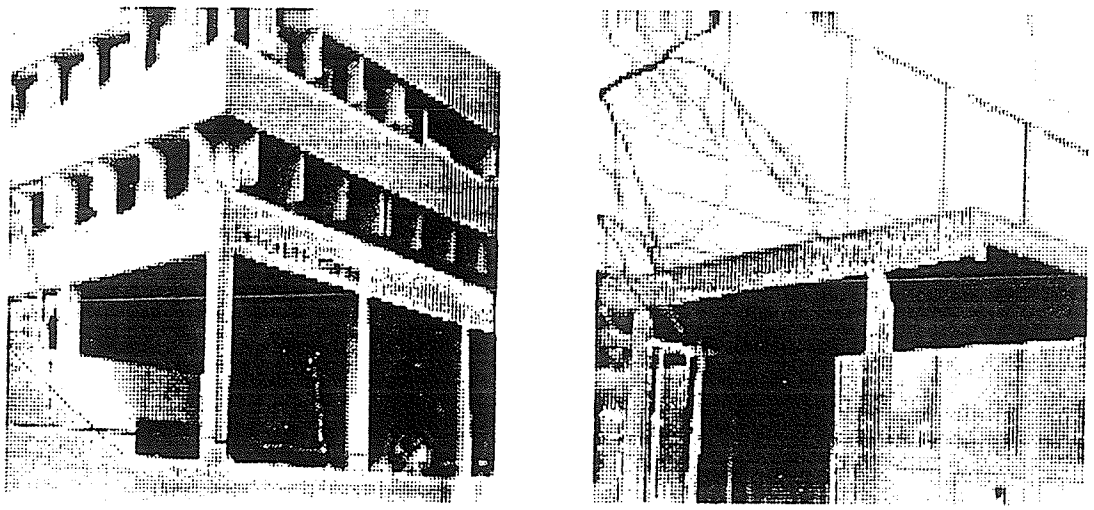
A main advantage of the Hough transform filtering procedure is that the procedure does not require any **a priori** information of the original image. All the parameters needed are obtained from the required features in the noisy binary image instead of the original image.

Chapter VI

APPLYING HOUGH TRANSFORM ON OUTDOOR SCENES

In the previous chapter, the Hough transform approach was proved an effective filtering procedure for extracting specific edge segments from the edge detected noisy man-made images. In this chapter, the application of the Hough transform filtering procedure to outdoor scenes will be discussed.

Straight lines are often used as key features in most building scene models, and they also consist of the main structures of the images as shown in Fig.6-1. The original pictures in Fig.6-1 were taken on the campus of the University of Manitoba.



(a) (b)

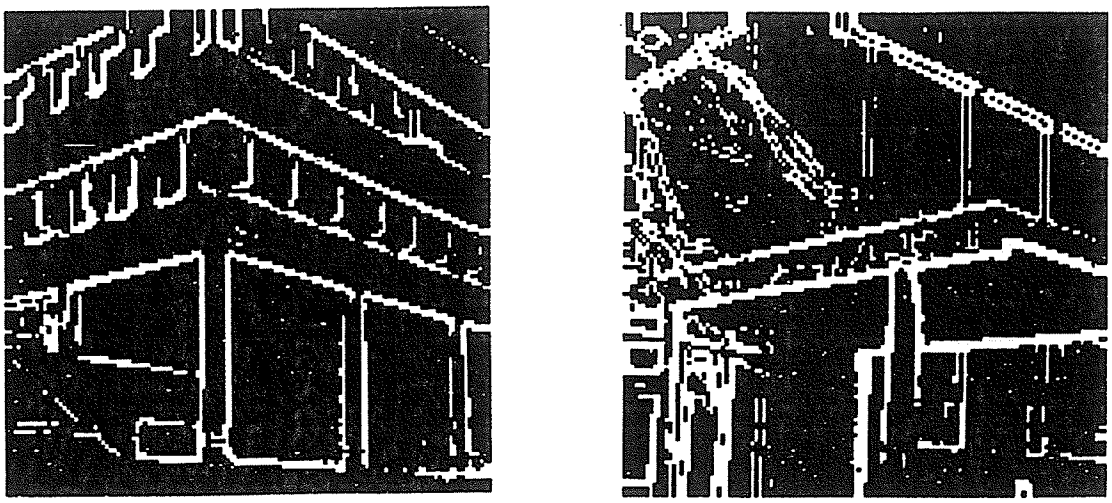
Fig.6-1 Original images.

6.1 NOISE-FREE CONSTRUCTION

It is more difficult to extract straight lines from outdoor scenes than from man-made or indoor images. This is primarily because more lines and elements are detected from outdoor scenes. Some results are presented in this section for the straight line searching of the noise free outdoor scenes.

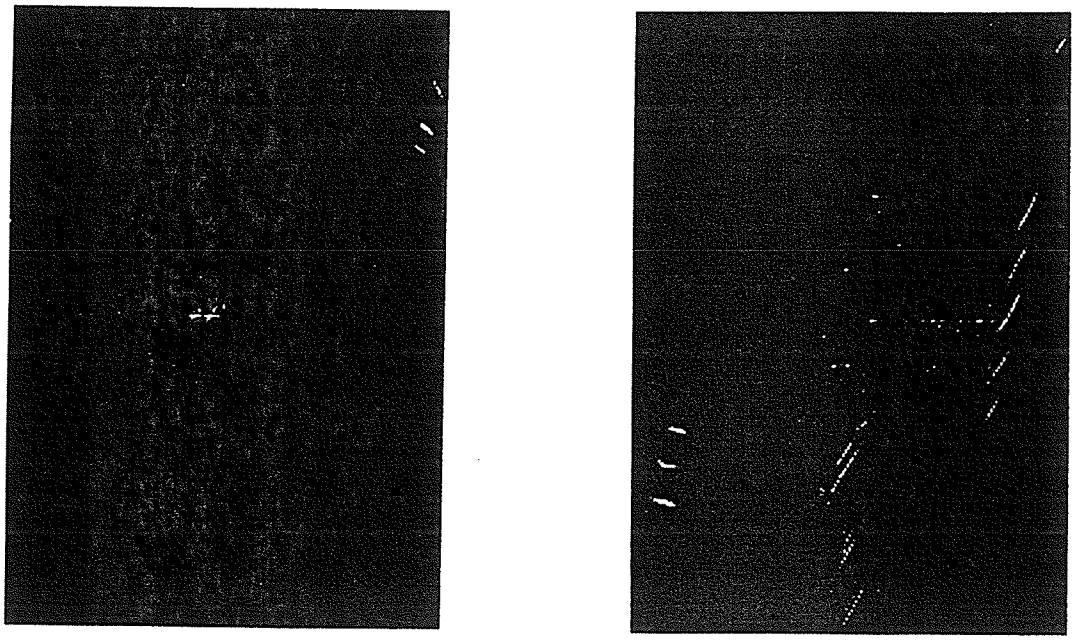
The original scenes shown in Fig.6-1 contain man-made structures and natural background, such as trees. As in the previous chapters, the Sobel edge detection operator is employed here. The edge versions of Fig.6-1 are shown in Fig.6-2. Fig.6-3 illustrates the mapped images of Fig.6-2 in the Hough space. The mapped images presented in Fig.6-3 correspond to the fact that the structure of Fig.6-2(b) is more complicated than that of Fig.6-2(a), or say, there are more potential straight lines in Fig.6-2(b). Fig.6-4 are the inverse Hough transform images. After applying the logic Matching procedure, the final results are shown in Fig.6-5.

Some line segments were missed during the process, however, the main features are still contained in the final results. Compared with Fig.6-5(b), Fig.6-5(a) presents a better result because of a simpler construction.



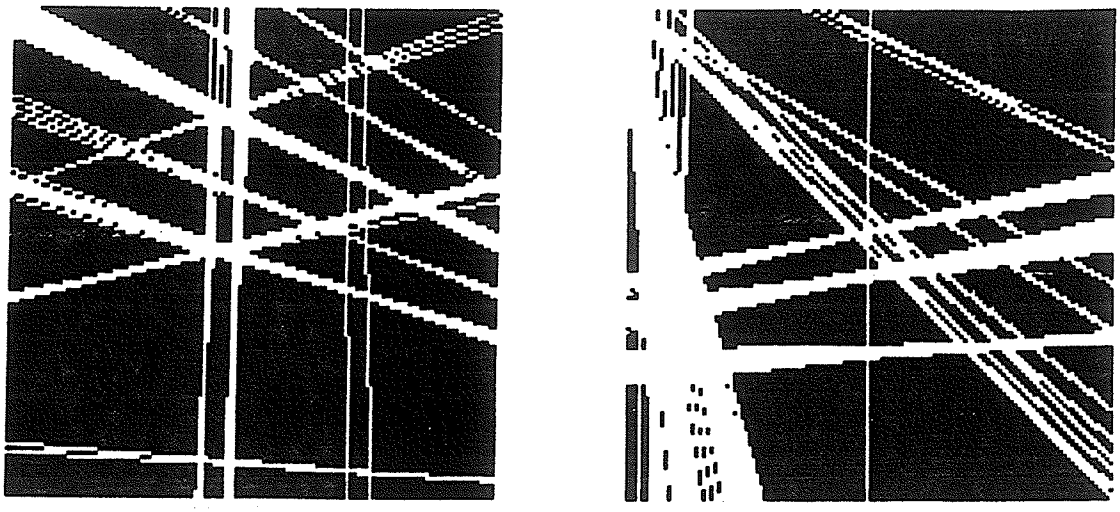
(a) (b)

Fig.6-2 The original images Fig.6-1(a) and Fig.6-1(b) detected by the Sobel edge detection operator with threshold values equate to 10 and 5, respectively.



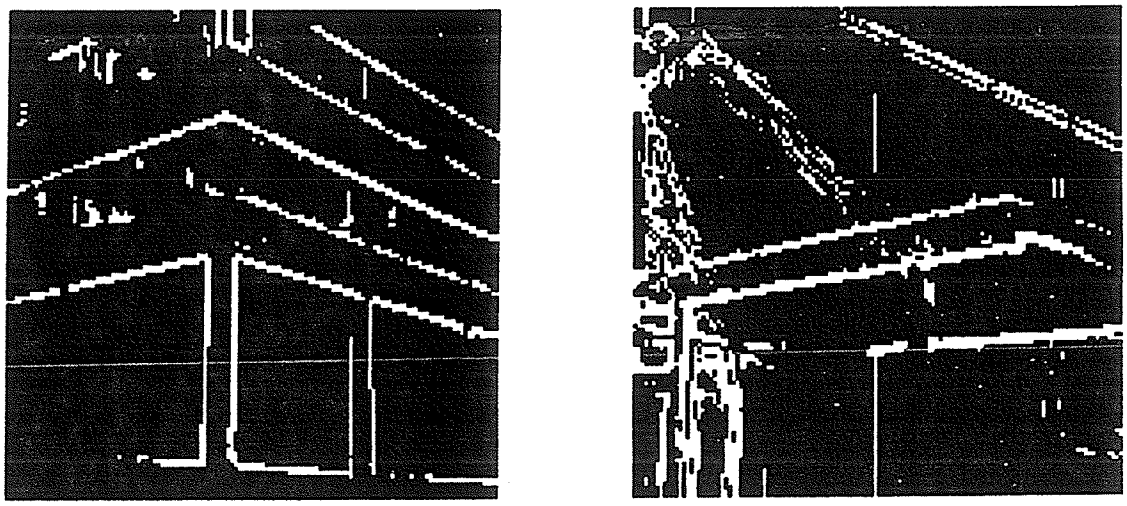
(a) (b)

Fig.6-3 The mapping images in the Hough space.



(a) (b)

Fig.6-4 The inverse Hough transform images of Fig.6-2.



(a) (b)

Fig.6-5 The final images by applying the Hough transform filtering procedure to the images in Fig.6-2.

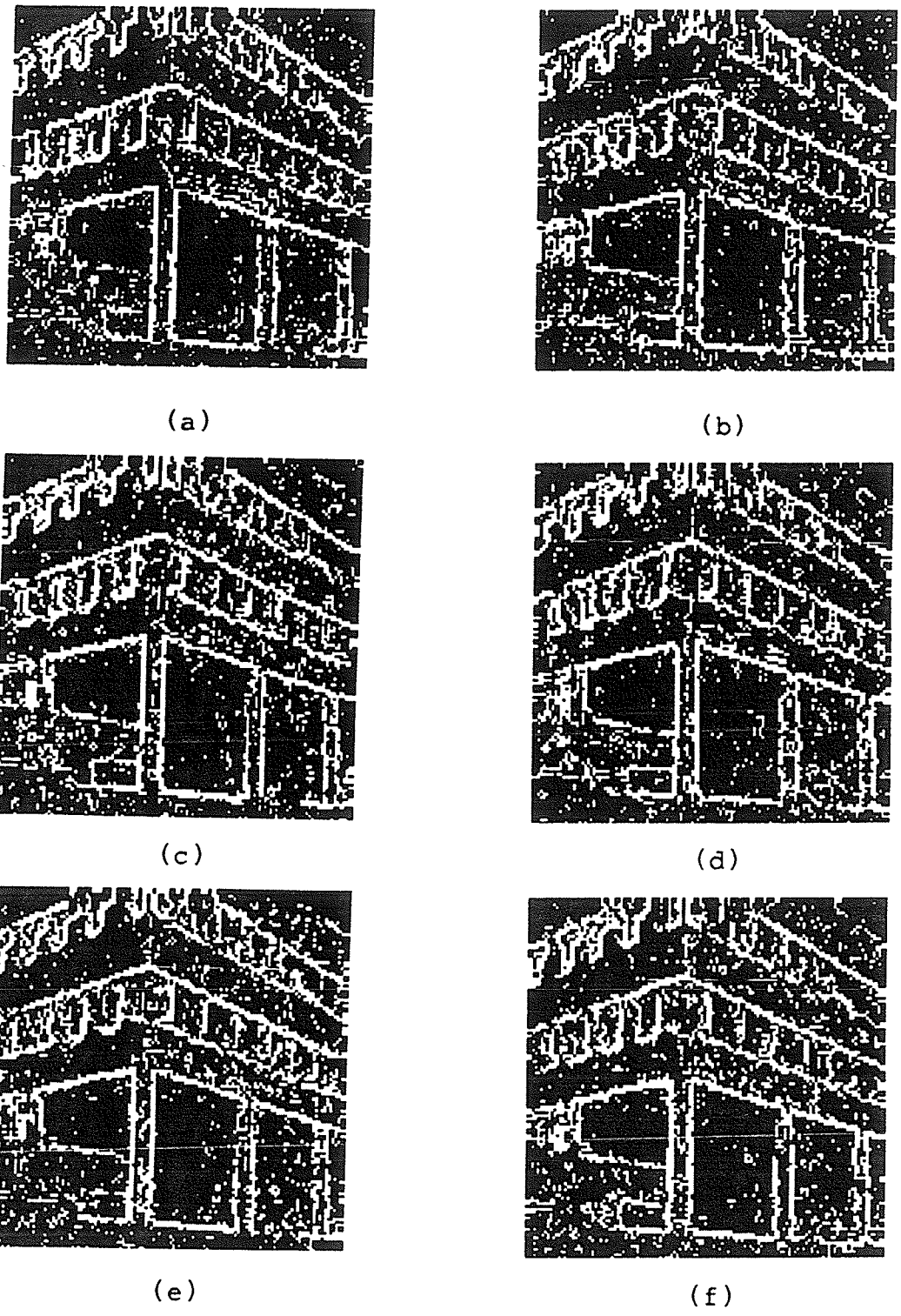
6.2 NOISE PERFORMANCE

To examine the ability of the Hough transform filtering procedure to extract special features from noisy outdoor scenes, different types of noise were used to degrade Fig.6-1(a) and Fig.6-1(b), respectively.

Fig.6-1(a) is degraded by additive Gaussian noise with different variances, and the edge detected versions are presented in Fig.6-6. The Sobel edge detection operator is employed here. Fig.6-7 displays the inverse Hough transform images of Fig.6-6. Fig.6-8 illustrates the final results of the complete Hough transform filtering procedure.

Degraded by Poisson noise with $\lambda=1.0$, the version of noised Fig.6-1(b) is shown in Fig.6-9(a). Fig.6-9(b) displays the edge detected version. The inverse Hough transform image and the final result of the Hough transform filtering procedure are presented in Fig.6-9(c) and Fig.6-9(d).

Comparing Fig.6-8 with Fig.6-6, it is concluded that the noisy images are satisfactorily improved by applying the Hough transform filtering procedure. Nevertheless, the result shown in Fig.6-9(d) is not very satisfactory. It is chiefly because the construction of Fig.6-1(b) is more complex and quite a few lines contained in the scene are very short. Therefore, it is very difficult to choose a proper threshold value for obtaining a suitable inverse Hough transform image.

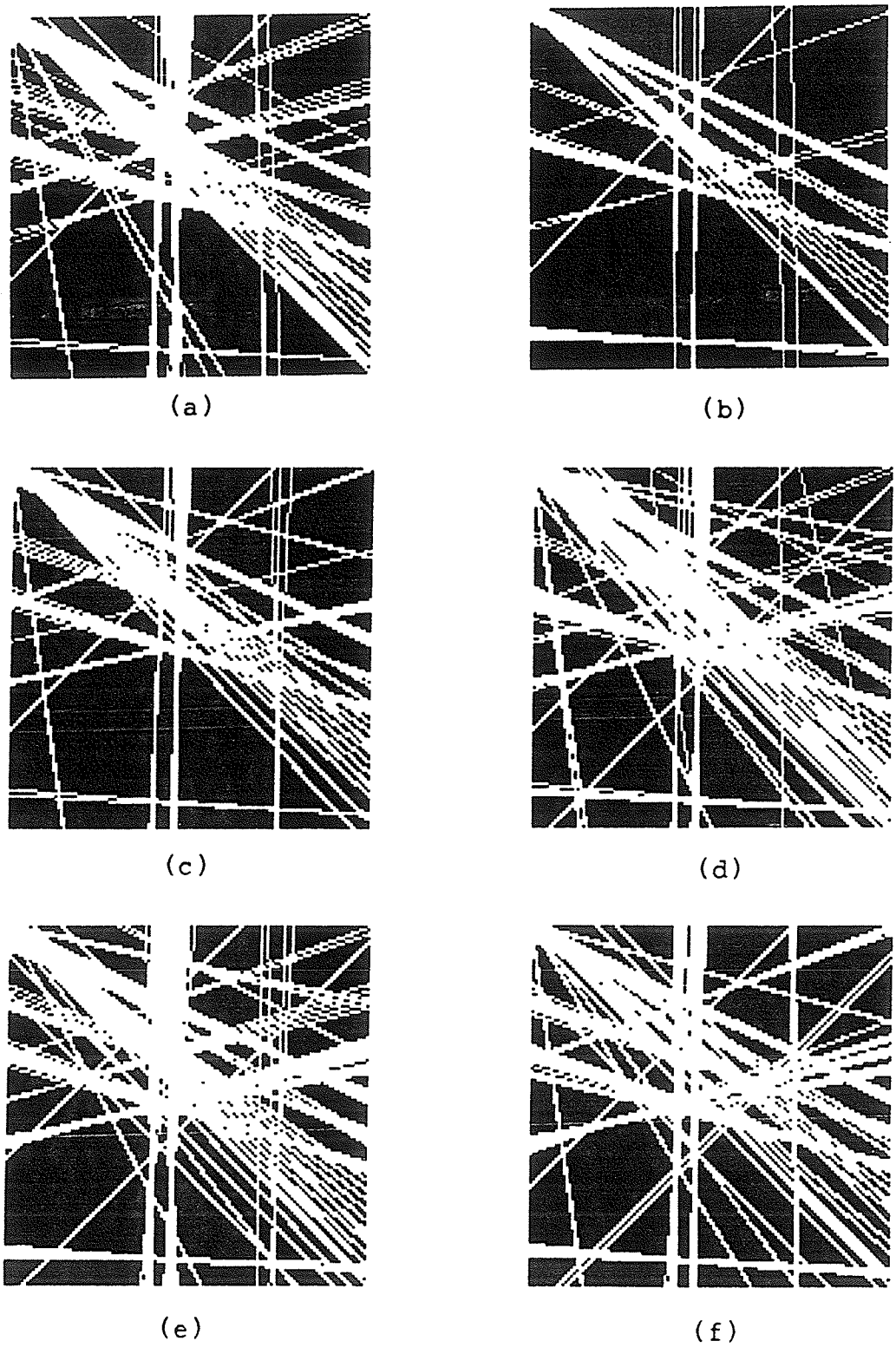


(a) (b)

(c) (d)

(e) (f)

Fig.6-6 The edge detected versions which obtained by employing the Sobel operator to Gaussian noise degraded Fig.6-1(a). σ^2=8, 10,12,14,16,18, respectively.

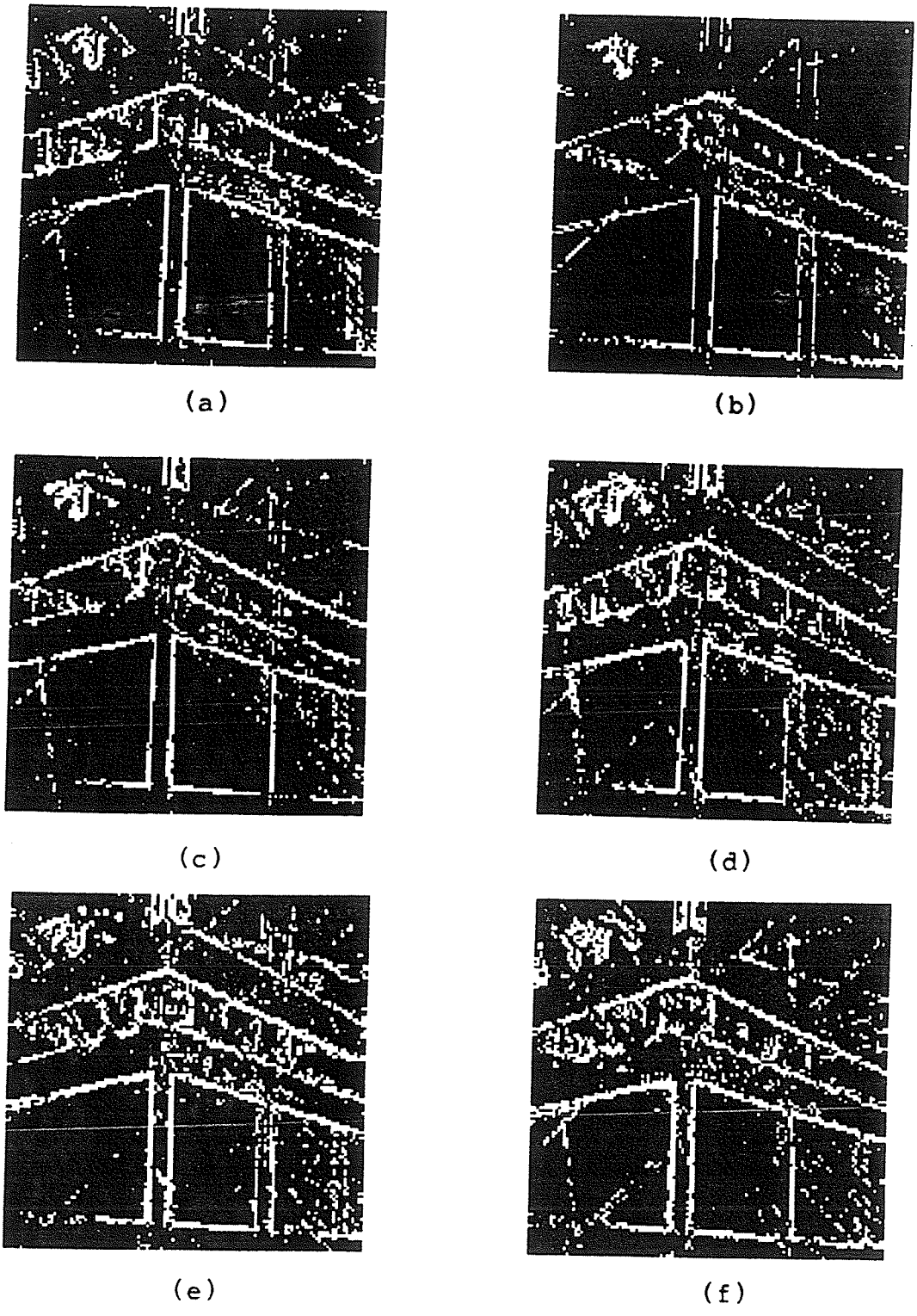


(a) (b)

(c) (d)

(e) (f)

Fig.6-7 The inverse Hough transform images of Fig.6-6.

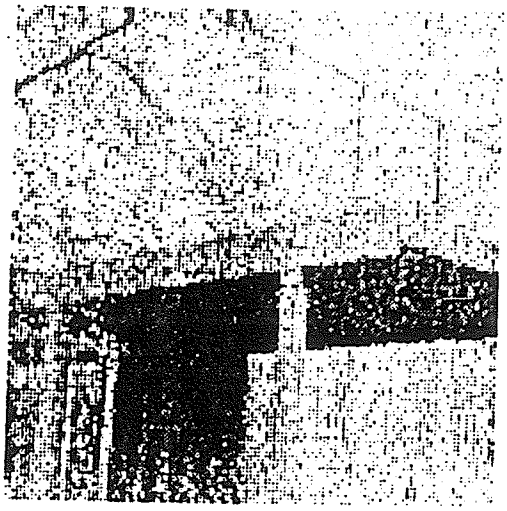


(a) (b)

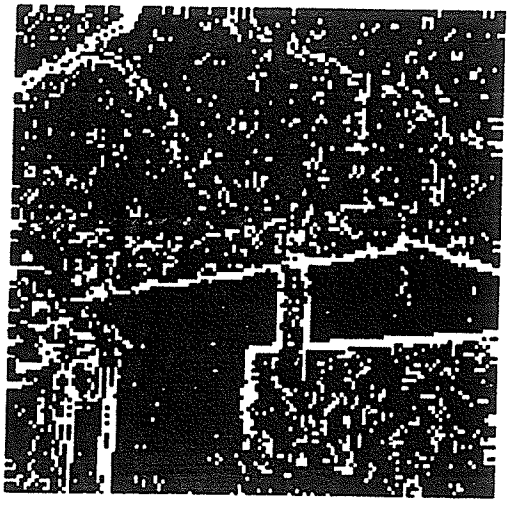
(c) (d)

(e) (f)

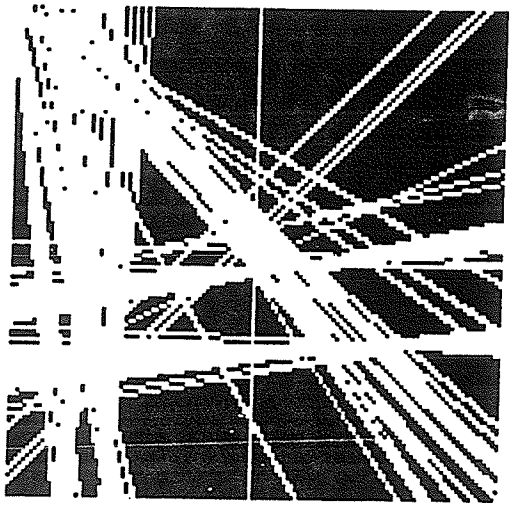
Fig.6-8 The final images of employing the Hough transform filtering procedure to Fig.6-6.



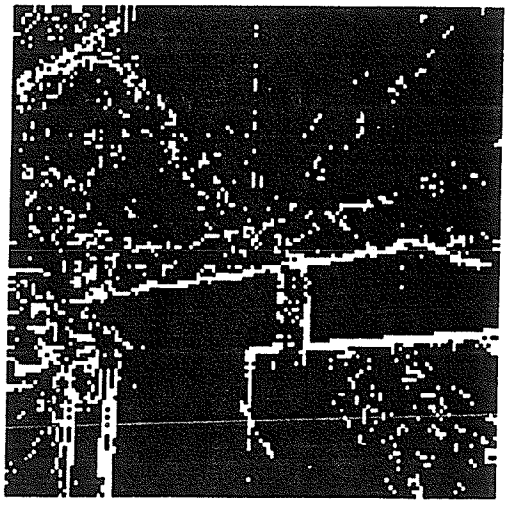
(a)



(b)



(c)



(d)

Fig.6-9 (a)Poisson noise degraded Fig.6-1(b) with λ=1.0. (b)The edge detected image. (c)The inverse Hough transform image. (d)The final result of complete Hough transform filtering procedure.

Chapter VII

CONCLUSIONS AND RECOMMENDATIONS

In this thesis, a new filtering procedure for the detection of specific features in noisy images has been suggested as an attempt to extend the use of the Hough transformation. The filtering procedure is composed of edge detection, Hough transform, threshold selection, inverse Hough transform and logic **AND** matching process.

An artificial image, which consists of circles and straight lines, was used as the test image with different types of noise. The research shows that, no matter what kind of noise, the Hough transform filtering procedure presented a stable performance when it extracted required features and ignored noisy pixels. Furthermore, the filtering procedure was applied to some noisy natural outdoor scenes and the improved results were obtained as expected.

Based on the behavior of the approach, the Hough transform filtering procedure can be suggested as a specific feature filter to extract known shapes from noisy image. A comparison is made between the performance of the Hough transform filtering procedure and that of the Median filter, which gives a strong support to the above opinion.

After reviewing the results of this research, some possible improvements in this area are presented for further research.

One of the weaknesses of the Hough transform procedure is that it is a time consuming approach. With the appearance of faster computers, the problem will be resolved gradually. Nevertheless, even without more powerful computers, higher computational speed can still be obtained. Since the Hough transform filtering procedure works parallelly, it lends itself to a potentially high speed implementation by dividing the time consuming search work to an array of computers, or, even to an array of microprocessors. Also, VLSI systolic array processors could be a possible powerful implement to offer a faster processing procedure[31]. If the time consuming problem is further improved, the Hough transform filtering procedure will be a more feasible approach.

In this research, the noisy image was treated as a direct input for the Hough transform filtering procedure, in which, edge detected image was handed to the Hough transform approach immediately. If a pre-filter of some kind is employed before applying the Hough transform filtering procedure, or between the edge detection and the Hough transform procedure in the processing, that is, if the Hough transform procedure is combined with other filters, higher qualitative results can be expected.

BIBLIOGRAPHY

1. P.V.C.Hough, Method and Means for Recognizing Complex Patterns, U.S, Patent 3069654, 1962.

2. R.O.Duda and P.E.Hart, "Use of the Hough transform to detect lines and curves in pictures", Comm. ACM. 15,1972,11-15.

3. W.K.Pratt, Digital Image Processing. New York, Wiley-Inter-Science, 1978.

4. R.C.Gonzalez and P.Wintz, Digital Image Processing. Addison-Wesley, 1987.

5. R.Krishnapuram and D.Casasent, "Hough Space Transformations for Discrimination and Distortion Estimation", Computer Vision, Graphics,and Image Processing 38,299-316(1987).

6. D.H.Ballard, "Generalizing The Hough Transform to Detect Arbitrary Shapes", Pattern Recognition, Vol.13, No.2, pp.111-122, 1981.

7. D.Casasent and R.Krishnapuram, "Curved Object Location by Hough Transformations and Inversions", Pattern Recognition, Vol.20, No.2, pp.181-188,1987.

8. J.Sklansky, "On the Hough Technique for Curve Detection", IEEE Transactions on Computers, Vol.c-27, No.10, October 1978.

9. A.Rosenfeld and A.C.Kak, Digital Picture Processing. 1982.

10. J.Illingworth and J.Kittler, "The Adaptive Hough Transform", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.RAMI-9, No.5, September 1987.

11. V.S.Frost, J.A.Stiles, K.S.Shanmugan and J.C.Holtzman, "A Model for Radar Images and Its Application to Adaptive Digital Filtering of Multiplicative Noise", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.RAMI-4, No.2, March 1982.

12. S.A.Dudani and A.L.Luk, "Locating Straight-Line Edge Segments on Outdoor Scenes", Pattern Recognition, Vol.10, pp.145-157, 1978.

13. I.E.Abdou and W.K.Pratt, "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors", Proceedings of The IEEE, Vol.67, No.5, May 1979.

14. L.G.Roberts, "Machine perception of three-dimensional solids," in Optical and Electro-Optical

Information Processing, J.T.Tippett et al., Eds. Cambridge, MA: M.I.T. Press, 1965, pp.159-197.

15. J.M.S.Prewitt, "Object enhancement and extraction," in Picture Processing and Psychopictorics, B.S.Lipkin and A.Rosenfeld, Eds. New York:Academic Press, 1970.

16. R.O.Duda and P.E.Hart, Pattern Classification and Scene Analysis. New York: Wiley, 1973.

17. A.C. Bovik, T.S.Huang and D.C.Munson, "The Effect of Median Filtering on Edge Estimation and Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-9,No.2,March 1987.

18. D.T.Kuan, A.A.Sawchuk, T.C.Strand and P.Chavel, "Adaptive Noise Smoothing Filter for Images with Signal-Dependent Noise", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol.PAMI-7, No.2,March 1985.

19. T.S.Huang, "Some notes on film-grain noise", Appendix 14, in Restoration of Atmospherically Degraded Images, NSF Summer Study Rep., Woods Hole, MA, 1966,pp.105-109.

20. T.S.Huang, G.J.Yang and G.Y.Tang, "A Fast Two-Dimensional Median Filtering Algorithm", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-27, No.1, February, 1979.

21. J.Maeda and K.Murata, "Digital Restoration of Scinitigraphic Image By a Two-Step Procedure", IEEE Transactions on Medical Imaging, Vol.MI-6, No.4, December 1987.

22. C.M.Lo and A.A.Sawchuk, "Nonlinear restoration of filtered images with Poisson noise", Proc.SPIE, Vol.207, pp.84-95, 1979.

23. C.C.Li, J.F.Mancuso, D.B.Shu, Y.N..Sun and L.D.Roth, "A Preliminary Study of Automated Inspection of VLSI Resist Patterns", IEEE International Conference on Robotics and Automation, St.Louis, Missouri, March 25-28,1985.

24. S.D.Shapiro and A.Iannino, "Geometric Constructions for Predicting Hough Transform Performance", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-1, No.3, July 1979.

25. E.R.Davies, "A New Framework for Analyzing The Properties of The Generalised Hough Transform", Pattern Recognition Letters 6(1987) 1-7, North-Holland.

26. E.R.Davies, "A New Parametrisation of The Straight Line and Its Application for The Optimal Detection of Objects with Straight Edges", Pattern Recognition Letters 6(1987) 9-14, North-Holland.

27. C.Kimme, D.Ballard and J.Sklansky, "Finding Circles by An Array of Accumulators", Comm.Assoc.Comput.Mach.18(2), 120-122.

28. S.Tsuji and F.Matsumoto, "Detection of Ellipses by A Modified Hough Transform. IEEE Trans. Computers 27(8), 777-781, 1978.

29. M.Cohen and G.T.Toussaint, "On The Detection of Structures In Noisy Pictures", Pattern Recognition Pergamon Press 1977, Vol.9, pp.95-98.

30. V.S.Nalwa and T.O.Binford, "On Detecting Edges", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.PAMI-8, No.6, November 1986.

31. S.Y.Kung, "VLSI Array Processors", 1988.

Appendix A

IMAGE DISPLAYING SUBROUTINES

Printing image display devices are required primarily for low resolution image processing work. In this thesis, a new displaying package, based on the Xerox 8700 printer, has been developed. The new routine employs 64 small dots as an image pixel, rather than overprints characters at the same point, to obtain different gray levels. Compared with overstrike approach[4], the new displaying package offers higher resolution, presents better quality and needs less computational time.

The new package is able to display 256 by 256 images with 65 gray-level, 128 by 128 images with 33 or 65 gray-level, respectively.

Fig.A-1 illustrates images with 256 rows, 256 columns and 65 gray-level. Fig.A-2 and Fig.A-3 present 128 by 128 images with 33 and 65 gray-level in order. From the images shown in Fig.A-3, it is concluded that if the size of image is limited in 128 by 128, 65 gray-level does not give significant improvement in the image quality. Therefore, the choices of 256X256, 65 gray-level and 128X128, 33 gray-level are suggested.

The FORTRAN subroutines and output examples are presented as follows:

```
      COMMON IA(256,256)
C
C     IA -- Integer array containing values ranging from
C           1 to 65 presenting the different graylevels.
C
      INTEGER*2 IA
      CALL READIM(128,1)
C
C     The subroutine reads in data to IA.
C
      CALL DSP(128,128,2)
C
C     This subroutine is used to display images in Xerox
C     8700 printer.
C
      STOP
      END
C
C
      SUBROUTINE READIM(N,ISC)
C
C     The arguments:
C         N -- The size of image(128 or 256).
C       ISC -- Image size choice
C              SC=1: 128 by 128 images,
C              SC=2: 256 by 256 images.
C
      COMMON IA(256,256)
      INTEGER*2 IA
      GO TO (10,20),ISC
10    READ 100,((IA(I,J),J=1,128),I=1,128)
100   FORMAT(1X,32I2/1X,32I2/1X,32I2/1X,32I2)
      GO TO 999
20    READ 101,((IA(I,J),J=1,256),I=1,256)
101   FORMAT(1X,32I2/1X,32I2/1X,32I2/1X,32I2/
     &       1X,32I2/1X,32I2/1X,32I2/1X,32I2)
999   RETURN
      END
C
C
      SUBROUTINE DSP(NX,NY,IDT)
C
C     The arguments:
C         NX -- Number of rows of IA to be printed;
C               the maximum NX is 256.
C         NY -- Number of columns of IA to be printed;
C               the maximum NY is 256.
C        IDT -- Different image types variable.
C               IDT=1:128 by 128 images with 33 graylevels.
```

```
C               IDT=2:128 by 128 images with 65 graylevels.
C               IDT=3:256 by 256 images with 65 graylevels.
C
      INTEGER SAVE/0/
      COMMON IA(256,256)
      INTEGER*2 IA
      INTEGER XMAX,YMAX,XSPACE
      LOGICAL*1 IMA256(256,8),GLEV65(65,8)
      LOGICAL*1 IMA128(128,8),GLEV33(33,8)
      LOGICAL*1 PADNUL(332)/332*Z00/
      DATA GLEV65/19*Z00,4*Z20,6*Z22,2*Z2A,7*ZAA,
     *     ZBA,9*ZBB,11*ZBA,2*ZBE,3*ZFE,ZFF,
     *     7*Z00,8*Z42,3*Z52,8*Z54,19*Z55,5*Z75,
     *     3*Z7D,4*Z7F,8*ZFF,
     *     2*Z00,Z20,4*Z24,Z00,3*Z08,7*Z28,3*Z2A,
     *     13*ZAA,3*ZEA,5*ZEE,ZEF,22*ZFF,
     *     Z00,Z10,4*Z00,3*Z10,4*Z14,15*Z54,5*Z55,
     *     Z5D,Z55,2*Z5D,10*Z55,5*Z5D,2*Z7D,2*Z7E,9*ZFF,
     *     5*Z00,Z10,4*Z08,4*Z28,13*Z2A,9*ZAA,
     *     2*ZBA,2*ZAB,ZBB,3*ZFB,21*ZFF,
     *     2*Z00,2*Z04,3*Z24,5*Z10,5*Z14,5*Z54,
     *     24*Z55,2*Z57,ZD7,2*ZDF,14*ZFF,
     *     7*Z00,9*Z42,Z4A,8*Z2A,9*ZAA,3*ZAE,
     *     11*ZEE,10*ZFE,7*ZFF,
     *     20*Z00,4*Z04,6*Z44,2*Z54,23*Z55,5*Z5D,
     *     2*Z7D,Z7F,2*ZFF/
      GO TO (100,200,300),IDT
100   XMAX=1772
      YMAX=2144
      XSPACE=748
      DO 105 I=1,33
      L=2*I-1
      DO 106 K=1,8
      GLEV33(I,K)=GLEV65(L,K)
106   CONTINUE
105   CONTINUE
      WRITE(8,112) XMAX,YMAX,(PADNUL(I),I=1,260)
112   FORMAT(2A4,156A1,104A1)
      DO 199 J=1,XSPACE
      WRITE(8,111) (PADNUL(I),I=1,268)
199    CONTINUE
      DO 101 I=1,NX
      DO 102 J=1,NY
      IF (IA(I,J).GT.33) IA(I,J)=33
      IF (IA(I,J).LT.1) IA(I,J)=1
      KLT=34-IA(I,J)
         DO 108 K=1,8
         IMA128(J,K)=GLEV33(KLT,K)
108      CONTINUE
102   CONTINUE
      DO 109 K=1,8
      WRITE(8,111) (PADNUL(L),L=1,140),(IMA128(L,K),L=1,128)
109   CONTINUE
101   CONTINUE
```

```
111   FORMAT(140A1,128A1)
      GO TO 999
200   XMAX=1772
      YMAX=2144
      XSPACE=748
      WRITE(8,212) XMAX,YMAX,(PADNUL(I),I=1,260)
212   FORMAT(2A4,156A1,104A1)
      DO 299 J=1,XSPACE
      WRITE(8,111) (PADNUL(I),I=1,268)
299   CONTINUE
      DO 201 I=1,NX
      DO 202 J=1,NY
      IF (IA(I,J).GT.65) IA(I,J)=65
      IF (IA(I,J).LT.1) IA(I,J)=1
      KLT=66-IA(I,J)
         DO 208 K=1,8
         IMA128(J,K)=GLEV65(KLT,K)
208      CONTINUE
202   CONTINUE
      DO 209 K=1,8
      WRITE(8,211) (PADNUL(L),L=1,140),(IMA128(L,K),L=1,128)
209   CONTINUE
201   CONTINUE
211   FORMAT(140A1,128A1)
      GO TO 999
300   XMAX=2284
      YMAX=2656
      XSPACE=236
      WRITE(8,312) XMAX,YMAX,(PADNUL(I),I=1,324)
312   FORMAT(2A4,248A1,76A1)
      DO 399 J=1,XSPACE
      WRITE(8,311) (PADNUL(I),I=1,332)
399   CONTINUE
      DO 301 I=1,NX
      DO 302 J=1,NY
      KLT=66-IA(I,J)
         DO 308 K=1,8
         IMA256(J,K)=GLEV65(KLT,K)
308      CONTINUE
302   CONTINUE
      DO 309 K=1,8
      WRITE(8,311) (PADNUL(L),L=1,76),(IMA256(L,K),L=1,256)
309   CONTINUE
301   CONTINUE
311   FORMAT(76A1,128A1,128A1)
999   END FILE 8
      CALL XIC300(SAVE)
      CALL PSCLHX(SAVE)
      RETURN
      END
```

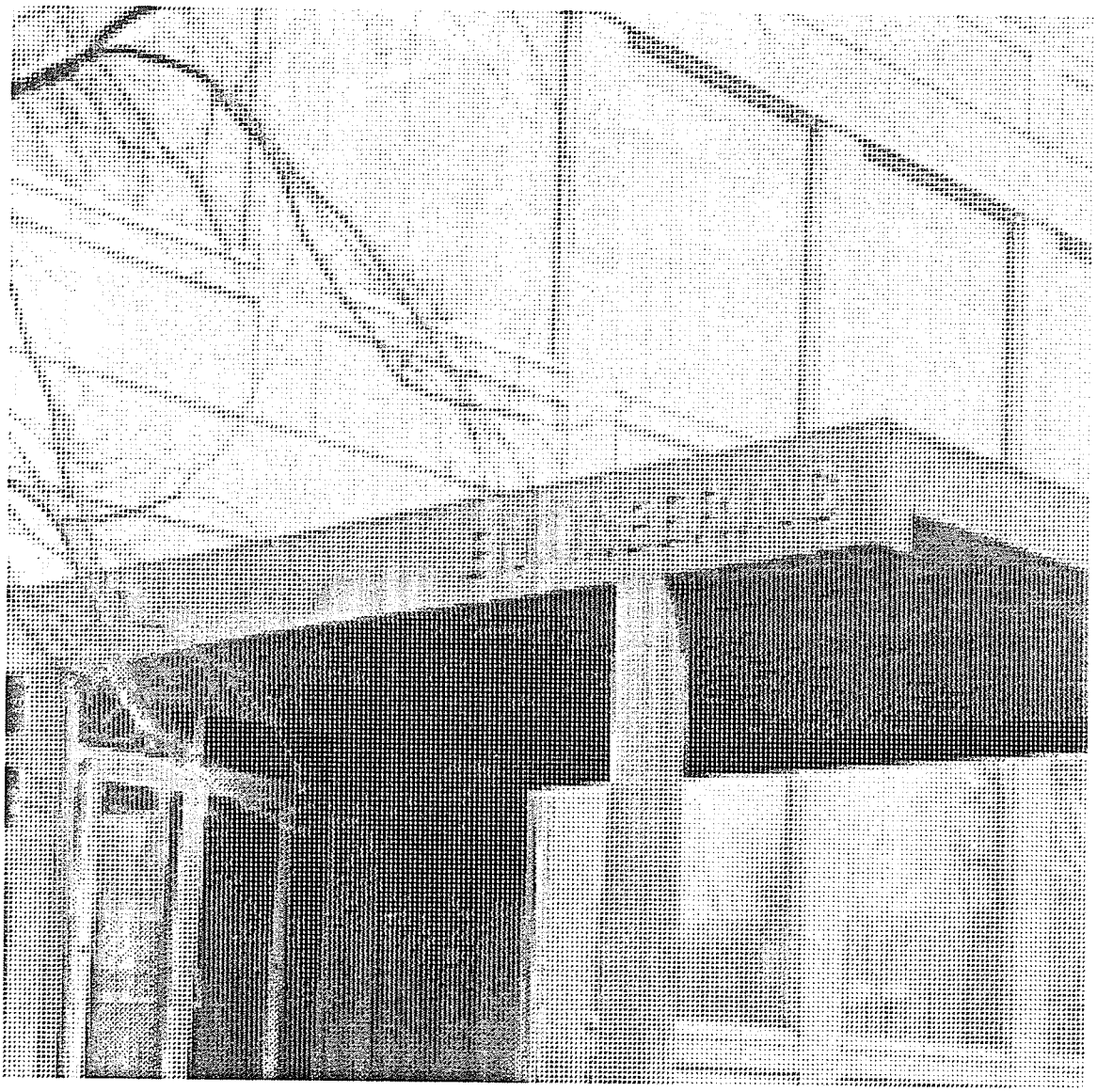


Fig.A-1(a)

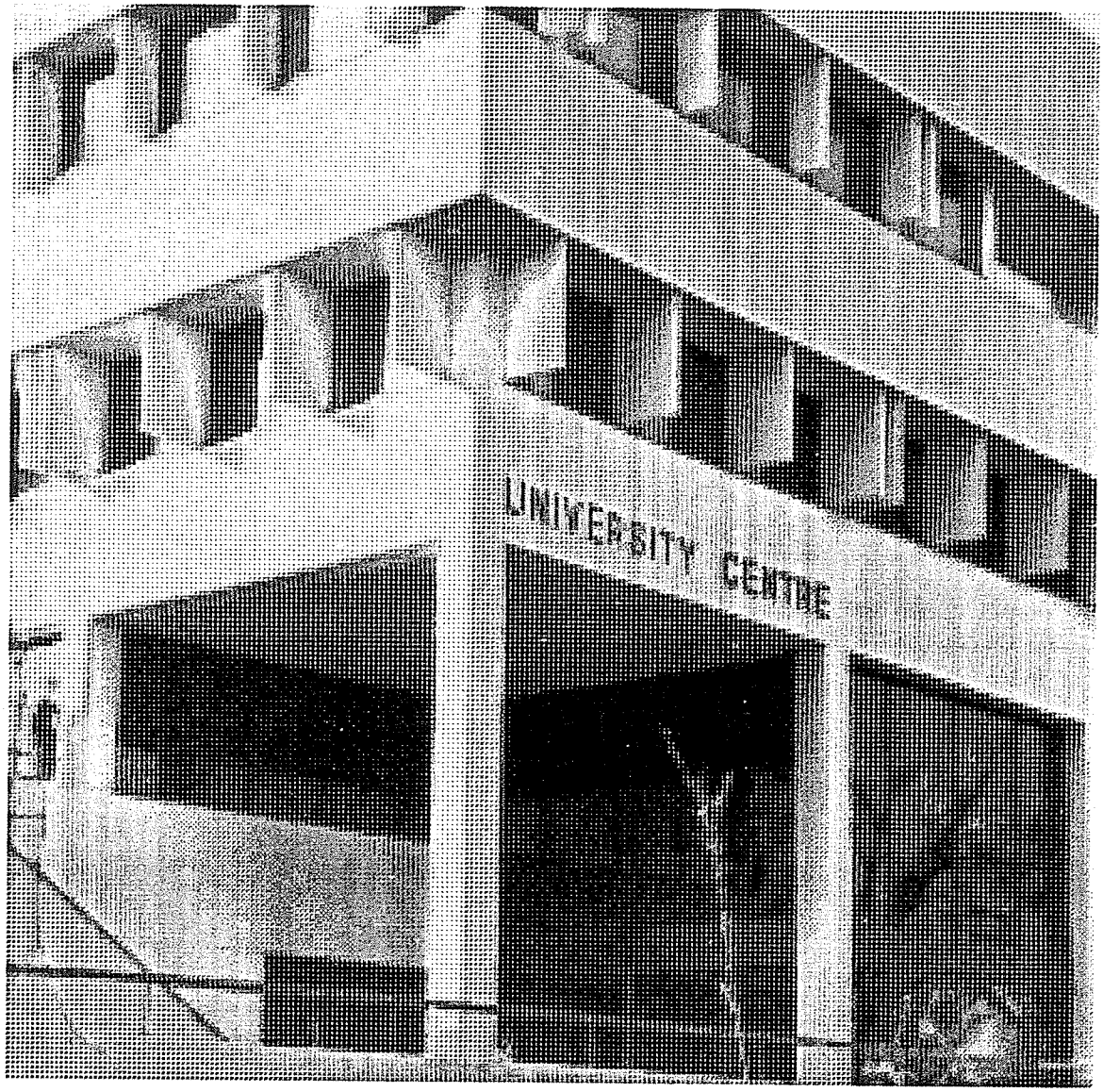


Fig.A-1(b)

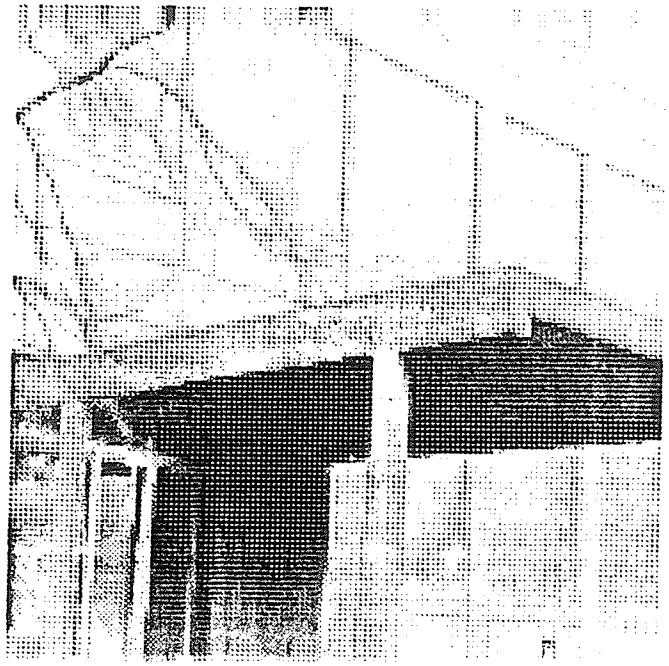


Fig.A-2(a) 128 by 128 pixels, 33 gray levels.

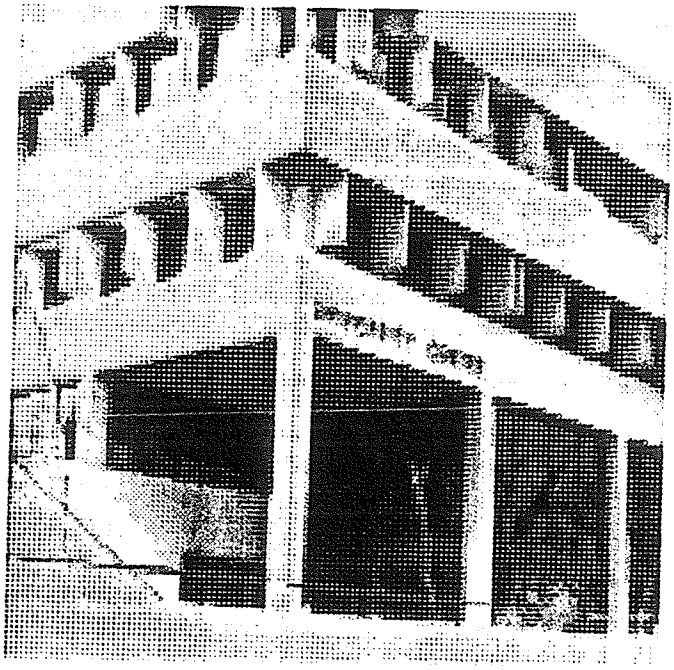


Fig.A-2(b) 128 by 128 pixels, 33 gray levels.

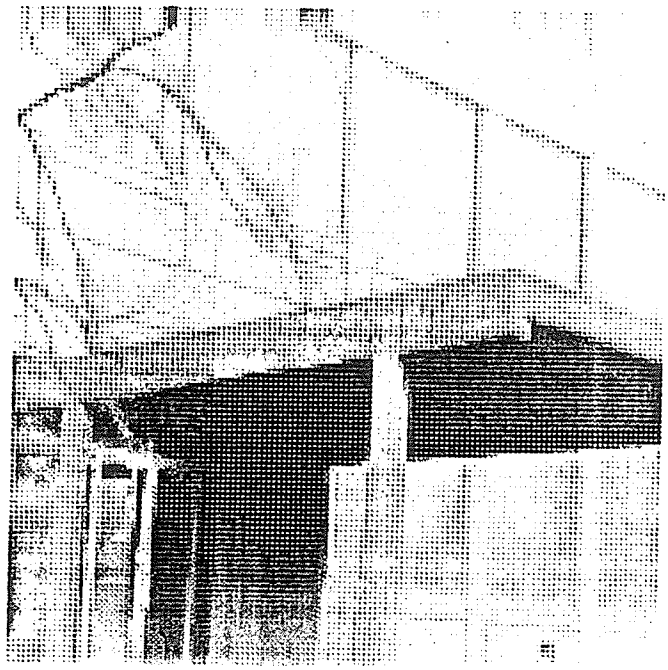


Fig.A-3(a) 128 by 128 pixels, 65 gray levels.

Fig.A-3(b) 128 by 128 pixels, 65 gray levels.

Appendix B

FORTRAN SUBROUTINES

```
      SUBROUTINE READIM(N,IL,IH)
C
C     Subroutine reads digitized image from file into
C     array IA.
C
C     Arguments:
C         N: The size of image to be read.
C        IH: Maximum gray level in the image.
C        IL: Minimum gray level in the image.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      READ 100,((IA(I,J),J=1,N),I=1,N)
  100 FORMAT(1X,32I2/1X,32I2/1X,32I2/1X,32I2)
      IH=IA(1,1)
      IL=IH
      DO 1 I=1,N
      DO 1 J=1,N
        IF (IA(I,J) .GT. IH) IH=IA(I,J)
        IF (IA(I,J) .LT. IL) IL=IA(I,J)
    1 CONTINUE
      RETURN
      END
C


      SUBROUTINE DSPLAY(NX,NY,IL,IH,NEG,LG)
C
C     Subroutine displays a 128X128 digital image with
C     32 gray levels on a standard line printer.
C     This subroutine was written refer to Bibliography [4].
C
C     Arguments:
C          NX: Number of rows of IA to be printed;
C              maximum NX is 128.
C          NY: Number of columns of IA to be printed;
C              maximum NY is 128.
C          IL: Minimum gray level in IA, calculated
C              in the subroutine READIM.
C          IH: Maximum gray level in IA, calculated
C              in the subroutine READIM.
```

```
C          NEG: A value equal to 1 gives the normal image;
C               A value equal to 0 gives the negative of
C               the image.
C           LG: Logical unit number for the line printer.
C
      COMMON IA(128,128)
      INTEGER*2 IA,IB(128,128)
      INTEGER*2 LEV(32),BLANK(5)
      LOGICAL*1 LINE(128,5),GRAY(32,5)
      DATA GRAY /6*'M',5*'H','X','H','X','O',
     1 'Z','W','M','N','O','S','=','I','*','+',
     2 '+','=',':','-','.','-',' ',6*'W',3*'#',
     3 '*','+','+',4*'-',5*' ','=',' ',' ',' ','-',
     4 3*' ','-',3*' ',4*'#','0','O','+','-',
     5 24*' ','O','0','O',29*' ','+',31*' '/
      GN=32.0
      FL=IL
      FH=IH
      IF ((FH-FL) .GT. 0.0) GO TO 100
      T=FL
      FL=FH
      FH=T
  100 RANGE=(FH-FL+1)/GN
      AA=(SQRT(FH)-SQRT(FL))/GN
      EE=(FH-FL)/ALOG(GN+1.0)
      T=AMAX1(FL,1.0)
      SS=-(1.0/GN)*ALOG(FH/T)
      DO 160 I=1,32
      FLEV=FL+(I-1)*RANGE+0.5
      LEV(I)=FLEV
  160 CONTINUE
      IF (NX .GT. 128) NX=128
      IF (NY .GT. 128) NY=128
      DO 180 I=1,NX
      DO 180 J=1,NY
        KLT=1
        DO 170 K=1,32
         IF (IA(I,J) .GE. LEV(K)) KLT=K
  170   CONTINUE
        IB(I,J)=KLT
  180 CONTINUE
      WRITE(LG,1)
      IX=NX
      IY=NY
      DO 210 I=1,IX
       DO 190 J=1,5
        BLANK(J)=0
  190  CONTINUE
       DO 200 K=1,IY
        J=K
        NG=IB(I,J)
        IF (NEG .EQ. 0) NG=33-NG
        DO 200 L=1,5
         LINE(K,L)=GRAY(NG,L)
```

```
          IF (NG .NE. 32) BLANK(L)=1
  200     CONTINUE
          WRITE(LG,2)
          DO 210 L=1,5
           IF (BLANK(L) .EQ. 0) GO TO 210
           WRITE(LG,3) (LINE(M,L),M=1,IY)
  210     CONTINUE
        1 FORMAT(1H1)
        2 FORMAT(1H )
        3 FORMAT(1H+,3X,128A1)
          RETURN
          END
C



          SUBROUTINE CUTOFF(N,IL,IH)
C
C         Subroutine cuts off the values which are
C         over range[IL,IH).
C
C         Arguments:
C              N: The size of IA (NXN).
C             IL: Minimum gray level in IA.
C             IH: Maximum gray level in IA.
C
          COMMON IA(128,128)
          INTEGER*2 IA
          DO 1 I=1,N
          DO 1 J=1,N
          IF (IA(I,J) .GT. IH) IA(I,J)=IH
          IF (IA(I,J) .LT. IL) IA(I,J)=IL
    1     CONTINUE
          RETURN
          END
C



          SUBROUTINE SCALE(N,IL,IH,NIL,NIH)
C
C         Subroutine scales IA to the range[NIL,NIH]
C
C         Arguments:
C              N: Size of image which stored in IA.
C             IL: Minimum gray level in IA, calculated
C                 in calling subroutine READIM.
C             IH: Maximum gray level in IA, calculated
C                 in calling subroutine READIM.
C            NIL: Required new Minimum gray level in IA.
C            NIH: Required new Maximum gray level in IA.
C
          COMMON IA(128,128)
          INTEGER*2 IA
          DO 1 I=1,N
```

```
      DO 1 J=1,N
       IA(I,J)=(NIH-NIL)*(IA(I,J)-IL)/(IH-IL)+1.5
    1 CONTINUE
      RETURN
      END
C
      SUBROUTINE PRNTIA(N,LG)
C
C     Write contents of IA into unit number LG where LG
C     represents an output file.
C
C     Arguments:
C          N: Size of IA (N X N).
C         LG: Nuit number of output file.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      WRITE(LG,100)((IA(I,J),J=1,N),I=1,N)
  100 FORMAT(1X,32I2/1X,32I2/1X,32I2/1X,32I2)
      RETURN
      END
C
      SUBROUTINE DUP(N,A,B)
C
C     Subroutine duplicates array A to array B.
C     N: Size of arrays (N X N).
C
      INTEGER*2 A(128,128),B(128,128)
      DO 1 I=1,N
      DO 1 J=1,N
1     B(I,J)=A(I,J)
      RETURN
      END
C
      SUBROUTINE GAUSS(N,SIGMA)
C
C     Subroutine adds Gaussian noise to the image in IA.
C
C     Argument:
C          N:   The size of image (NXN).
C      SIGMA: Variance.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      REAL SIGMA
```

```
      PI=3.141593
      SSIGMA=SQRT(8.0)
      M=10000
      DO 2 I=1,N
      DO 2 J=1,N
      A1=IRANDN(M)/(M*1.0)
      A2=IRANDN(M)/(M*1.0)
      R=SSIGMA*SQRT(-2*ALOG(A1))*COS(2*PI*A2)
      IF (R . GE . 0.0)  THEN DO
         IB(I,J)=INT(R+0.5)
      ELSE DO
         IB(I,J)=-1*INT(-1*R+0.5)
      END IF
2     CONTINUE
      DO 10 I=1,N
      DO 10 J=1,N
10    IA(I,J)=IA(I,J)+IB(I,J)
      RETURN
      END
C



      SUBROUTINE SALTPE(N,SITA,IL,IH)
C
C     Subroutine adds salt-peper noise to the image
C     stored in IA.
C
C     Arguments:
C          N: Size of IA (N X N).
C       SITA: 0.0≤SITA≤1.0.
C         IL: Minimum gray level in IA.
C         IH: Maximum gray level in IA.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      LD=IH-IL-1
      M=LD*1000
      DO 1 I=1,N
      DO 1 J=1,N
       Y=IRANDN(M)/(1000.0)
       IZ=IFIX((Y+1)*SITA+0.5)
       IF (IA(I,J).EQ.IL) IA(I,J)=IA(I,J)+IZ
       IF (IA(I,J).EQ.IH) IA(I,J)=IA(I,J)-IZ
    1 CONTINUE
       RETURN
      END
C
```

```
      SUBROUTINE POISS(N,LANMAD)
C
C     Subroutine degrades the image stored in IA with
C     Poissian noise.
C
C     Arguments:
C          N: The size of IA (NXN).
C     LANMAD: Proportionality factor.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      INTEGER COUNT
      REAL SIGMA,LANMAD,EINV,SUM
      M=10000
      DO 1 I=1,N
      DO 1 J=1,N
      EINV=EXP(-LANMAD*IA(I,J))
      COUNT=0
      SUM=1.0
10    A=IRANDN(M)/(M*1.0)
      SUM=SUM*A
      IF (SUM.GE.EINV) THEN DO
      COUNT=COUNT+1
      GOTO 10
      ENDIF
      IA(I,J)=COUNT+1
1     CONTINUE
      RETURN
      END
C


      SUBROUTINE FILMGR(N,ALGHA,SIGMA,IL,IH)
C
C     Subroutine adds film-grain noise to the image
C     Stored in IA.
C
C     Arguments:
C          N: The size of IA.
C      ALGHA: A constant. In this thesis, ALGHA=1.0 is chosen.
C      SIGMA: Variance.
C         IL: Minimum gray level in IA.
C         IH: Maximum gray level in IA.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      REAL AIB(128,128)
      REAL SIGMA,ALGHA
      PI=3.141593
      SSIGMA=SQRT(SIGMA)
      M=10000
      DO 2 I=1,N
      DO 2 J=1,N
```

```
      A1=IRANDN(M)/(M*1.0)
      A2=IRANDN(M)/(M*1.0)
      AIB(I,J)=SSIGMA*SQRT(-2*ALOG(A1))*COS(2*PI*A2)
2     CONTINUE
      DO 10 I=1,N
      DO 10 J=1,N
      IA(I,J)=IA(I,J)+INT(ALGHA*((IA(I,J)*1.0)
     $        **(1./3.)*AIB(I,J))+0.5)
      IF (IA(I,J).LT.IL) THEN DO
      IA(I,J)=IL
      END IF
      IF (IA(I,J).GT.IH) THEN DO
      IA(I,J)=IH
      END IF
10    CONTINUE
      RETURN
      END
C



      SUBROUTINE SOBEL(N,IL,IH,T)
C
C     Subroutine applies the Sobel edge detector to the
C     image stored in IA.
C
C     Arguments:
C          N: Size of IA (N X N).
C         IL: Minimum gray level in IA.
C         IH: Maximum gray level in IA.
C          T: Nonnegative threshold.
C
      COMMON IA(128,128)
      INTEGER T
      INTEGER*2 IA
      M=N-1
      DO 1 I=2,M
      DO 1 J=2,M
      IB(I,J)=IA(I-1,J-1)+2*IA(I-1,J)+IA(I-1,J+1)-IA(I+1,J-1)
     @        -2*IA(I+1,J)-IA(I+1,J+1)
      IC(I,J)=IA(I-1,J-1)+2*IA(I,J-1)+IA(I+1,J-1)
     @        -IA(I-1,J+1)-2*IA(I,J+1)-IA(I+1,J+1)
1     CONTINUE
      DO 2 I=2,M
      DO 2 J=2,M
      IA(I,J)=(IABS(IB(I,J)*1)+IABS(IC(I,J)*1))/4
2     CONTINUE
      DO 9 I=2,M
      DO 9 J=2,M
      IF (IA(I,J) .GE. T) IA(I,J)=IH
      IF (IA(I,J) .LT. T) IA(I,J)=IL
9     CONTINUE
      IA(1,1)=IA(2,2)
      IA(1,N)=IA(2,M)
```

```
      IA(N,1)=IA(M,2)
      IA(N,N)=IA(M,M)
      DO 3 I=2,M
      IA(1,I)=IA(2,I)
      IA(N,I)=IA(M,I)
      IA(I,1)=IA(I,2)
      IA(I,N)=IA(I,M)
3     CONTINUE
      RETURN
      END
C



      SUBROUTINE MEDIAN(N)
C
C     Subroutine applies the 3X3 square Median filter to
C     the image stored in IA.
C
C     N: The size of IA.
C
      COMMON IA(128,128),IB(128,128),IC(128,128)
      INTEGER*2 IA,IB,IC
      INTEGER C(9),K,L,M,BOTTOM,EXCH,CURRNT,TEMP
      K=2
      M=N-K+1
      DO 1 I=K,M
      DO 1 J=K,M
      C(1)=IA(I,J)
      C(2)=IA(I,J+1)
      C(3)=IA(I-1,J+1)
      C(4)=IA(I-1,J)
      C(5)=IA(I-1,J-1)
      C(6)=IA(I,J-1)
      C(7)=IA(I+1,J-1)
      C(8)=IA(I+1,J)
      C(9)=IA(I+1,J+1)
C     **********************
C     *   Start sorting    *
C     **********************
      BOTTOM=9
      WHILE (BOTTOM .GT. 1 ) DO
         EXCH=0
         CURRNT=1
         WHILE (CURRNT .LT. BOTTOM) DO
           IF (C(CURRNT) .GT. C(CURRNT+1)) THEN DO
             TEMP=C(CURRNT)
             C(CURRNT)=C(CURRNT+1)
             C(CURRNT+1)=TEMP
             EXCH=CURRNT
           END IF
           CURRNT=CURRNT+1
         END WHILE
         BOTTOM=EXCH
```

```
        END WHILE
C     ********************
C     *   End sorting    *
C     ********************
        IB(I,J)=C(5)
1       CONTINUE
        DO 2 I=K,M
        DO 2 J=K,M
        IA(I,J)=IB(I,J)
2       CONTINUE
        DO 3 I=K,M
        IA(I,1)=IA(I,2)
        IA(I,N)=IA(I,M)
3       CONTINUE
        DO 4 J=1,N
        IA(1,J)=IA(2,J)
        IA(N,J)=IA(M,J)
4       CONTINUE
        RETURN
        END
C



        SUBROUTINE MATCH(N,MA,MB,MC,IL,IH)
C
C       Subroutine matches the images in MA and MB,
C       then stores the final image in MC.
C
C       Arguments:
C            N: The size of array A, B and C.
C        A,B,C: Integer arrays.
C           IL: Minimum gray level in A and B.
C           IH: Maximum gray level in A and B.
C
        INTEGER*2 MA(128,128),MB(128,128),MC(128,128)
        DO 1 I=1,N
        DO 1 J=1,N
        IF (MA(I,J).EQ.IH .AND. MB(I,J).EQ.IH) THEN DO
        MC(I,J)=IH
        ELSE DO
        MC(I,J)=IL
        END IF
1       CONTINUE
        RETURN
        END
C
```

```
      SUBROUTINE CLEAR(N,A,IL)
C
C     Subroutine clears the array A with IL.
C
C     Arguments:
C          N: The size of A.
C          A: The array to be cleared.
C         IL: Minimum gray level asked in main program.
C
      COMMON IA(128,128)
      INTEGER*2 IA,A(128,128)
      DO 1 I=1,N
      DO 1 J=1,N
1     A(I,J)=IL
      RETURN
      END
C



      SUBROUTINE FGF(N,B,C,GAMMA,IL,IH)
C
C     Subroutine computes the  FGF's of original  noisy edge
C     detected image and final result of the Hough transform
C     filtering procedure.
C
C     Arguments:
C          N: The size of IA.
C      GAMMA: Penalty constant.
C         IL: The lower gray level in the images.
C         IH: The higher gray level in the images.
C
      COMMON IA(128,128)
      INTEGER*2 IA,B(128,128),C(128,128)
      REAL GAMMA
      INTEGER NB,NN1,NN2,MM1,MM2
C
C     NB:  Number of boundary pixels on the noise-free target.
C     NN1: Number of background pixels added to the original
C          noisy edge detected image.
C     NN2: Number of target pixels removed from the original
C          noisy edge detected image.
C     MM1: Number of background pixels added to the final result
C          of Hough transform filtering procedure.
C     NN2: Number of target pixels removed from the final result
C          of Hough transform filtering procedure.
C
      NB=NN1=NN2=MM1=MM2=0
      CALL READIM(N,IL,IH)
      CALL DUP(N,IA,B)
C
C     Read in original noisy edge detected image
C     and store the image in array B.
```

```
C
      CALL READIM(N,IL,IH)
      CALL DUP(N,IA,C)
C
C     Read in final result of Hough transform filtering
C     procedure and store the image in array C.
C
      CALL READIM(N,IL,IH)
C
C     Read in noise free target.
C
      DO 1 I=1,N
      DO 1 J=1,N
      IF (IA(I,J).EQ.IH) NB=NB+1
      IF ((IA(I,J).EQ.IL).AND.(B(I,J).EQ.IH)) NN1=NN1+1
      IF ((IA(I,J).EQ.IH).AND.(B(I,J).EQ.IL)) NN2=NN2+1
      IF ((IA(I,J).EQ.IL).AND.(C(I,J).EQ.IH)) MM1=MM1+1
      IF ((IA(I,J).EQ.IH).AND.(C(I,J).EQ.IL)) MM2=MM2+1
1     CONTINUE
      R=NB*1.0/((NN1+NN2*GAMMA)*1.0)
      Q=NB*1.0/((MM1+MM2*GAMMA)*1.0)
10    FORMAT('1')
      PRINT 10
      PRINT 20,Q/R
20    FORMAT('/','   FGF   = ',F7.4)
      PRINT 10
      RETURN
      END
C



      SUBROUTINE RELINE(N,IRHO,ITHETA,IH,RB)
C
C     Subroutine reconstructs the line with given
C     IRHO,ITHETA in array RB.
C
C     Arguments:
C         N: The size of IA
C      IRHO:
C    ITHETA:
C        IH: Maximum gray level in IA.
C        RB: An integer array to store result.
      COMMON IA(128,128)
      INTEGER*2 IA,RB(128,128)
      INTEGER X,Y
      REAL THETA
      PI=3.141593
      THETA=(PI/180.0)*ITHETA
      DO 1 X=1,N
      DO 1 Y=1,N
      RHO=X*COS(THETA)+Y*SIN(THETA)
      IF (RHO.GT.0.0) IR=IFIX(RHO+0.5)
      IF (RHO.LE.0.0) IR=IFIX(RHO)
```

```
      IF (IR.EQ.IRHO) RB(X,Y)=IH
1     CONTINUE
      RETURN
      END
C



      SUBROUTINE HLINES(N,NUMTHE,THOLD,IH,HB)
C
C     Subroutine searches straight lines in
C     the image stored in IA.
C
C     Arguments:
C         N: The size of IA.
C    NUMTHE: The steps between 0 and PI.
C     THOLD: Threshold value.
C        IH: Maximum gray level.
C        HB: The array to store the final result.
C
      COMMON IA(128,128)
      INTEGER*2 IA,HB(128,128),COUNT(362,181)
      INTEGER THOLD
      PI=3.141593
      M=NUMTHE-1
      DETA=PI/NUMTHE
      DO 100 I=1,362
      DO 100 J=1,181
100   COUNT(I,J)=0
      DO 1 I=1,N
      DO 1 J=1,N
      IF (IA(I,J).EQ.IH) THEN DO
      DO 2 K=1,M
      RHO=I*COS(K*DETA)+J*SIN(K*DETA)
      IF (RHO.GT.0.0) IX=IFIX(RHO+0.5)
      IF (RHO.LE.0.0) IX=IFIX(RHO)
      COUNT(IX+181,K)=COUNT(IX+181,K)+1
2     CONTINUE
      END IF
1     CONTINUE
      DO 10 I=1,361
      DO 10 J=1,181
      IF (COUNT(I,J).GE.THOLD) THEN DO
      IRHO=I-181
      CALL RELINE(N,IRHO,J,IH,HB)
C      PRINT 11,I-181,J,COUNT(I,J)
C11    FORMAT('/','RHO=',I4,'  THETA=',I3,'  THE NUMBER=',I3)
      END IF
10    CONTINUE
      RETURN
      END
C
```

```
      SUBROUTINE RECIRC(N,X0,Y0,R0,IL,IH,RC)
C
C     Subroutine reconstructs circle with given
C     X0,Y0,R0 in array RC.
C
C     Arguments:
C          N: The size of IA.
C         X0: Horizontal coordinate of the centre.
C         Y0: Vertical coordinate of the centre.
C         R0: Radius of the circle.
C         IL: Minimum gray level in IA.
C         IH: Maximum grat level in IA.
C         RC: An integer array to store result.
C
      COMMON IA(128,128)
      INTEGER*2 IA,RC(128,128)
      INTEGER X0,Y0,R0
      DO 1 I=1,N
      DO 1 J=1,N
      RP=SQRT(((I-X0)*(I-X0)+(J-Y0)*(J-Y0))*1.0)
      R=RP-R0
      IF ((R.LE.1.0).AND.(R.GE.0.0)) RC(I,J)=IH
1     CONTINUE
      RETURN
      END
C



      SUBROUTINE HCIRCL(N,MINR,MAXR,THOLD,IL,IH,HC)
C
C     Subroutine searches circles in the image stored in IA.
C
C     Arguments:
C         N: The size of IA.
C      MINR,MAXR: The searching range starts from MINR and
C                 stop in MAXR.
C     THOLD: Threshold value.
C        IL: Minimum gray level in IA.
C        IH: Maximum gray level in IA.
C        HC: The array to store result.
C
      COMMON IA(128,128)
      INTEGER*2 IA,HC(128,128)
      INTEGER COUNT(30,128,128)
      INTEGER MAXR,MINR,X,Y,R,THOLD,X0,Y0,RR
      M=MAXR-MINR+1
      RR=MINR-1
      DO 11 K=1,M
      DO 11 I=1,N
      DO 11 J=1,N
11    COUNT(K,I,J)=0
      DO 01 X=1,N
```

```
      DO 01 Y=1,N
      IF (IA(X,Y).EQ.IH) THEN DO
      DO 10 K=1,M
      R=K+RR
      DO 10 Y0=1,N
      C=(R*R-(Y-Y0)*(Y-Y0))*1.0
      IF (C.LT.0.0) GOTO 10
      X0=IFIX(X+SQRT(C)+1.0)
      IF (X0.LE.N) THEN DO
      COUNT(K,X0,Y0)=COUNT(K,X0,Y0)+1
      END IF
      X0=IFIX(X-SQRT(C))
      IF (X0.GT.0) THEN DO
      COUNT(K,X0,Y0)=COUNT(K,X0,Y0)+1
      END IF
10    CONTINUE
      END IF
01    CONTINUE
      DO 12 K=1,M
      R=K+RR
      DO 12 X0=1,N
      DO 12 Y0=1,N
      IF (COUNT(K,X0,Y0).GE.THOLD) THEN DO
      CALL RECIRC(N,X0,Y0,R,IL,IH,HC)
      END IF
12    CONTINUE
      RETURN
      END
C



      SUBROUTINE PREHL(N,NUMTHE,THOLD,IH)
C
C     Subroutine obtains information about the image
C     before employing subroutine HLINES.
C
C     Arguments:
C         N: The size of IA.
C    NUMTHE: The steps between 0 and PI.
C     THOLD: Threshold value.
C        IH: Maximum gray level.
C
      COMMON IA(128,128)
      INTEGER*2 IA,COUNT(362,181)
      INTEGER THOLD
      PI=3.141593
      M=NUMTHE-1
      DETA=PI/NUMTHE
      DO 100 I=1,362
      DO 100 J=1,181
100   COUNT(I,J)=0
      DO 1 I=1,N
      DO 1 J=1,N
```

```
      IF (IA(I,J).EQ.IH) THEN DO
      DO 2 K=1,M
      RHO=I*COS(K*DETA)+J*SIN(K*DETA)
      IF (RHO.GT.0.0) IX=IFIX(RHO+0.5)
      IF (RHO.LE.0.0) IX=IFIX(RHO)
      COUNT(IX+181,K)=COUNT(IX+181,K)+1
2     CONTINUE
      END IF
1     CONTINUE
      DO 10 I=1,361
      DO 10 J=1,181
      IF (COUNT(I,J).GE.THOLD) THEN DO
      PRINT 11,I-181,J,COUNT(I,J)
11    FORMAT('/','RHO=',I4,'  THETA=',I3,'  THE NUMBER=',I3)
      END IF
10    CONTINUE
      RETURN
      END
C



      SUBROUTINE PREHC(N,MINR,MAXR,THOLD,IL,IH)
C
C     Subroutine obtains information about the image
C     before employing subroutine HCIRCL.
C
C     Arguments:
C         N: The size of IA.
C      MINR,MAXR: The searching range starts from MINR and
C                 stop in MAXR.
C     THOLD: Threshold value.
C        IL: Minimum gray level in IA.
C        IH: Maximum gray level in IA.
C
      COMMON IA(128,128)
      INTEGER*2 IA
      INTEGER COUNT(30,128,128)
      INTEGER MAXR,MINR,X,Y,R,THOLD,X0,Y0,RR
      M=MAXR-MINR+1
      RR=MINR-1
      DO 11 K=1,M
      DO 11 I=1,N
      DO 11 J=1,N
11    COUNT(K,I,J)=0
      DO 01 X=1,N
      DO 01 Y=1,N
      IF (IA(X,Y).EQ.IH) THEN DO
      DO 10 K=1,M
      R=K+RR
      DO 10 Y0=1,N
      C=(R*R-(Y-Y0)*(Y-Y0))*1.0
      IF (C.LT.0.0) GOTO 10
      X0=IFIX(X+SQRT(C)+1.0)
```

```
      IF (X0.LE.N) THEN DO
      COUNT(K,X0,Y0)=COUNT(K,X0,Y0)+1
      END IF
      X0=IFIX(X-SQRT(C))
      IF (X0.GT.0) THEN DO
      COUNT(K,X0,Y0)=COUNT(K,X0,Y0)+1
      END IF
10    CONTINUE
      END IF
01    CONTINUE
      DO 12 K=1,M
      R=K+RR
      DO 12 X0=1,N
      DO 12 Y0=1,N
      IF (COUNT(K,X0,Y0).GE.THOLD) THEN DO
      PRINT 19,X0,Y0,R,R,X0,Y0,COUNT(K,X0,Y0)
      END IF
12    CONTINUE
19    FORMAT('/','   X0=',I3,'  Y0=',I3,
     $       ' R=',I3,' COUNT(',I3,',',I3,',',I3,')=',I4)
      RETURN
      END
```