DALHOUSIE UNIVERSITY

To comply with the Canadian Privacy Act the National Library of Canada has requested that the following pages be removed from this copy of the thesis:

Preliminary Pages
- Examiners Signature Page
- Dalhousie Library Copyright Agreement

Appendices
- Copyright Releases (if applicable)

IDEMPOTENT, DIRECTION-CONSISTENT ANISOTROPIC DIFFUSION

by

Hongwen Yi

Submitted

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

Major Subject: Electrical and Computer Engineering

at

DALHOUSIE UNIVERSITY

Halifax, Nova Scotia

March, 2004

Your file Votre référence
ISBN: 0-494-02124-1
Our file Notre référence
ISBN: 0-494-02124-1

TO MY FAMILY

who have been supporting and encouraging me

throughout my academic years

Contents

List of Tables viii

List of Figures ix

List of Symbols and Abbreviations xii

Acknowledgements xiv

Abstract xv

1 Introduction 1

2 Background 7
2.1 The Basic Concepts of AD 7
2.1.1 The Idea of AD 7
2.1.2 Formulation of AD 7
2.1.3 The Discrete Version of AD 11
2.1.4 Expected Merits of AD 12
2.1.5 Criteria of AD 13
2.2 State of Art 14
2.2.1 Research on AD theory 14
2.2.2 Applications of AD 21
2.2.3 Research on AD Implementation in Hardware 26
2.3 The Major Issues of AD Research 31
2.3.1 Problem 1: Automatically Stopping Diffusion 32
2.3.2 Problem 2: Finding a Suitable Value For Thresholds 41
2.4 Summary of Current AD Research 50

3 Idempotent Anisotropic Diffusion 54
3.1 Analysis of Traditional Anisotropic Diffusion 55
3.1.1 TAD Techniques: A Form of Averaging Filter 55
3.1.2 Review of Current Ways 57
3.2 Behavioral Analysis of Anisotropic Diffusion 60

3.2.1 AD Behavioral Analysis in Continuous Domain 60
3.2.2 A Criterion For The Conduction Function 70
3.3 A Discrete Form of Anisotropic Diffusion 71
3.3.1 Continuous Version of Anisotropic Diffusion 71
3.3.2 AD Behavioral Analysis in Discrete Domain 73
3.3.3 Discussion 79
3.4 Idempotent Anisotropic Diffusion 80
3.4.1 Idempotent Anisotropic Diffusion Algorithm 80
3.4.2 Assessment of The IAD Scheme 82
3.5 Experiments 86
3.5.1 Source of Test Images 86
3.5.2 Simulation Functions 87
3.5.3 Experiments on 1D signals 90
3.5.4 Experiments with 2D images 95

4 Idempotent, Direction-Consistent AD **139**
4.1 Analysis of Gradient-Magnitude-Based Anisotropic Diffusion 140
4.1.1 Why Most AD Methods Detect Edge by Gradient 140
4.1.2 The Difficulty of Choosing K 141
4.1.3 Review of Ways of Choosing K 142
4.2 Itempotent, Direction-Consistent Anisotropic Diffusion 144
4.2.1 Gradient's Attributes 144
4.2.2 Direction-Consistency-Based Anisotropic Diffusion 145
4.3 Experiments 151
4.3.1 Source of Test images 151
4.3.2 Simulation Functions 151
4.3.3 Analysis of Experimental Results 152
4.3.4 Experiment Results 154

5 Conclusion and Discussion **165**
5.1 Conclusion 165
5.2 Future Work 170
5.2.1 Improving the performance of IDCAD 170
5.2.2 Implementing AD in VLSI - The Next Hot Topic 171

bibliography **174**

Appendices **198**

A Programs for TAD **198**

B Program for AD by CFs of "Mono" flux **209**

C Programs for IAD 213

D Programs for GFAB 224

E Program for IDCAD 233

List of Tables

3.1 Iterations for "Mono" DCs to give idempotent results 90
3.2 Iterations for "Mixed" DCs to give idempotent results 92
3.3 Gradient re-distribution produced by IAD in 1D experiment 94
3.4 Gradient re-distribution produced by TAD in 1D experiments 94
3.5 Gradient re-distribution produced by IAD in 2D images 96
3.6 Gradient re-distribution produced by TAD in 2D images 97

List of Figures

2.1 A typical diffusion coefficient function 8
2.2 Heat diffusion 9
2.3 A typical flux function 10
2.4 A convential real-time image processing system 27
2.5 A two-dimensional grid structure implementing AD in hardware . . . 29
2.6 Recommended flux $\phi(\cdot)$ for anisotropic diffusion 30
2.7 Detailed structure of a pixel and its neighbors in AD hardware 31
2.8 Problem of over-bluring for current AD techniques 33
2.9 Automatic stopping time defined in the research of Lin *et al.* 37
2.10 Problem of detecting edge based on gradient magnitude. 43
2.11 Principle of edge detection by gradient 44

3.1 Derivative of flux 61
3.2 CF curve of the "Gilboa" FAB filter 82
3.3 Conduction functions in this experiment 87
3.4 The "Mono" fluxes produced by each CF 88
3.5 The "Mixed" fluxes generated by the CFs 89
3.6 Results produced by CFs generating "Mono" fluxes. 99
3.7 Results produced by CFs generating "Mono" fluxes. 100
3.8 Results of "Mixed" flux plus a negative CF of different widths. 101
3.9 Results produced by the "Gilboa" FAB filter. "$*$" are the results after a fidelity term is included with $\lambda = 0.05$ [83]. 102
3.10 Results of "fidel" of different λ. 103
3.11 Results produced by CFs generating "Mixed" fluxes. 104
3.12 Results produced by CFs generating "Mixed" fluxes. 105
3.13 Comparision between results of "exponential" of different flux. 106
3.14 1D results from the IAD filter. 107
3.15 1D results from the TAD filter. 108
3.16 Histogram results from the IAD filter on 1D signal. 109
3.17 Histogram results from the TAD filter on 1D signal. 110
3.18 Image results produced by the "Mono" flux. 111
3.19 Gradient results produced by the "Mono" flux. 112
3.20 Image results by the "Gilboa" FAB filter with a fidelity term λ added. 113

3.21 Gradient results by the "Gilboa" FAB filter with a fidelity term λ added. . . . 114
3.22 Image results produced by the traditional AD techniques. . . . 115
3.23 Gradient results produced by the traditional AD techniques. . . . 116
3.24 Image results produced by the "Mixed" flux. . . . 117
3.25 Gradient results produced by the "Mixed" flux. . . . 118
3.26 Results produced by traditional AD techniques. . . . 119
3.27 Gradient results produced by traditional AD techniques. . . . 120
3.28 Image results by the "Gilboa" FAB filter with a fidelity term λ added. 121
3.29 Gradient results by the "Gilboa" FAB filter with a fidelity term λ added. . . . 122
3.30 Results produced by "Mono" flux. . . . 123
3.31 Gradient results produced by "Mono" flux. . . . 124
3.32 Results produced by the "Mixed" flux. . . . 125
3.33 Gradient results produced by "Mixed" flux. . . . 126
3.34 Image results by the "Gilboa" FAB filter with a fidelity term λ added. 127
3.35 Gradeint results by the "Gilboa" FAB filter with a fidelity term λ added. . . . 128
3.36 Results of TAD on 2D images. . . . 129
3.37 Gradients resulting from the TAD on 2D images. . . . 130
3.38 Results of IAD on 2D images. . . . 131
3.39 Gradients resulting from the IAD on 2D images. . . . 132
3.40 Intensity Histogram of Image Results from TAD. . . . 133
3.41 Gradient Histogram of Image Results from TAD. . . . 134
3.42 Intensity Histogram of Image Results from IAD. . . . 135
3.43 Gradient Histogram of Image Results from IAD 136
3.44 Three-dimensional view of resulted intensities by TAD. . . . 137
3.45 Three-dimensional view of resulted intensities by IAD. . . . 138

4.1 Two squares used for showing their differency in edge directions . . . 146
4.2 Gradient vectors 147
4.3 Sum of the gradient vectors 149
4.4 3x3 Sobel operators. . . . 152
4.5 Original images for the experiments 154
4.6 Noisy images for the experiments 155
4.7 DC maps estimated by DCEE 156
4.8 Experiment results obtained by GMAD on original image "lena". . . . 157
4.9 Image and gradient results obtained by IDCAD on original image "lena". . . . 158
4.10 Experiment results obtained by GMAD on original image "cameraman". 159
4.11 Experiment results obtained by IDCAD on original image "cameraman". . . . 160

4.12 Experiment results obtained by GMAD on noisy image "lena". 161
4.13 Experiment results obtained by IDCAD on noisy image "lena". 162
4.14 Experiment results obtained by GMAD on noisy image "cameraman". 163
4.15 Experiment results obtained by IDCAD on noisy image "cameraman". 164

5.1 Summary of contributions of this thesis. 169

List of Symbols and Abbreviations

∇	gradient operator
Δ	Laplace operator
ρ	consistency measure
div	divergence operator
AD	anisotropic diffusion
A/D	analog to digital
CAD	continuous anisotropic diffusion
CCD	charge coupled device
CF	conduction function
$CMOS$	complementary metal oxide semiconductor
DAD	discrete form of anisotropic diffusion
DC	diffusion coefficient
DF	derivative of flux
$DCAD$	diffusion-consistency-based anisotropic diffusion
$DCEE$	diffusion-consistency-based edge estimator
DSP	digital signal processor
FAB	forward and backward diffusion
$GFAB$	forward-and-backward diffusion proposed by Gilboa.
$GMAD$	gradient-magnitude-based anisotropic diffusion
$GMEE$	gradient-magnitude-based edge estimator

IAD	idempotent anisotropic diffusion
$IDCAD$	idempotent, direction-consistent anisotropic diffusion
M_1	magnitude of the sum of gradient vectors
M_2	sum of gradient vectors
MRI	magnetic resonance image
PDE	partial derivative equation
$P-M$	Perona and Malik
$RTIP$	real time image processing
SNR	signal-to-noise ratio
TAD	traditional anisotropic diffusion
$VLSI$	very large scale integration

Acknowledgements

I would like to give my sincere appreciation to my supervisor, Dr. Gregson, for his inspiration, support and helpful suggestions. He introduced me into a fascinating field of research, and has been inspiring me with his extensive knowledge throughout my research. It is my privilege to have him as my supervisor.

Thanks to Dr. E. El-Masry and Dr. M. Heywood for their kindly agreeing to be my thesis committee members. Thank you Dr. J. Little for your participation as external examiner. I also want to acknowledge all members in the iDLab.

Finally, I want to express my eternal gratitude to my family for their constant help and support, and extend my deepest appreciation to my wife Guidan and my daughter Tong for their support and understanding during my academic years.

Abstract

This thesis focuses on two of the most critical problems in the field of anisotropic diffusion (AD), *viz.* automatically stopping the diffusion process and selecting a suitable threshold for edges, that remained unsolved since AD theory was introduced in 1987 [171].

Over-smoothing of semantically meaningful features occurs very easily with Traditional AD (TAD) filters if the number of iterations is not carefully selected. Our research explains why TAD approaches do not act as might be expected from current AD theory. Idempotent AD (IAD), a new interpretation of AD using the non-negative part of the derivative of flux (DF) to control the smoothing strength, is proposed. A behavioral analysis is presented in detail of the AD process along the whole gradient magnitude due to the conduction function (CF). The analysis shows that a mathematically well-posed, a mathematically ill-posed or an idempotent diffusion process can be produced by the same CF. A criterion for selecting the CF is created. A threshold is proposed for true edges. We show the form of the discrete version of AD (DAD) whose solution converges to that of its continuous counterpart stably and consistently. Our proposed IAD keeps meaningful edges throughout the diffusion process with noise being smoothed, thereby making the experimental results agree with AD theory for the first time since 1987.

Choosing a suitable threshold is very important for AD techniques because it controls which edges are preserved. Determining this threshold with gradient-magnitude-based edge estimator (GMEE) is image dependent and becomes very complex with the appearance of noise and changes in illumination. This stubborn problem is avoided by the proposed idempotent, direction-consistent AD technique (IDCAD). This new technique uses a new criterion for implementing AD, combining the merits of a direction-consistency-based edge estimator (DCEE) and those of IAD. DCEE has low sensitivity to noise because regions containing edges show much more consistent edge directions as compared to regions of noise. Our algorithm implements IAD on noise regions located by DCEE.

Experiments carried out on 1D and 2D images with both artificial and real images validate the effectiveness of the proposed IAD and IDCAD techniques.

Chapter 1

Introduction

The concept of AD, first introduced by Perona and Malik (P-M) in 1987 [171], has led to a considerable amount of research into its theory and practical applications [243]. The AD technique is distinguished from other image processing techniques in three significant and desirable ways. It implements noise-smoothing and edge-preserving simultaneously, can be implemented in hardware, and is suitable for unsupervised computer vision tasks. If combined with very large scale integration (VLSI) techniques, AD filters avoid the speed restrictions of serial operations, thereby making feasible the implementation of real time image processing at low cost.

The two of the most critical problems of 1) automatically stopping the diffusion process and 2) selecting a suitable value for the threshold for estimating edges remain unsolved since the introduction of AD theory in 1987 [171]. The first problem, over-smoothing resulting in meaningless results, occurs very easily with current AD techniques if the number of iterations or the stopping criterion is not preset carefully. The selection of the stopping criterion or iteration count is usually image dependent. The second problem, choosing a suitable threshold, makes the implementation of AD very image sensitive. Research into this problem has mostly been based on gradient-magnitude-based edge estimators (GMEE) that are not good at

distinguishing edges from noise with high intensity gradients. Both problems make implementing AD for practical computer vision very problematic. Our research has concentrated on exploring solutions for both of them.

Among the ways of dealing with the first problem, simply restricting the number of iterations is popular [2, 23, 25, 33, 47, 134, 136, 185, 214, 224, 213, 230]. The smoothing is stopped manually when "good" results, in which noise seems to be removed while edges have not been blurred obviously, appear. The number of iterations is both image-dependent and person-dependent. Believing that there is an "optimal" state in the AD process is the basis for considerable research [131, 132, 206, 207]. These approaches regard AD as a process of improvement and then deterioration of the resultant image. Finding out the conditions under which degradation is going to take place is the goal of the work. However, there is no guarantee that the AD process acts in this way. A fidelity term may be used to restrict the over-blurring [47, 48, 83, 84, 180, 231]. The fidelity term controls the similarity between the results and the original images. This method is problematic because of the empirical parameters, and noise still remained in results. Considerable work has been carried out on the conduction function (CF) that controls the diffusion process. The CF has been modified and a number of new CFs have been proposed [30, 31, 83, 84, 85, 132, 201, 220, 240, 242, 249, 254, 256, 257]. The most recent significant progress was made by Gilboa *et al.* [83] who showed the behavior of AD within a partial range of gradient magnitude. Research on the behavior over the entire range of gradient magnitude is one of the topics of this thesis.

The discrete version of AD (DAD) has not been discussed enough since the introduction of AD theory in 1987 [171]. Among the large number of published papers on AD research, very few of them deal with the numerical aspects of AD [213, 231, 233, 235]. The behaviour of DAD proposed by Perona and Malik (P-M) has

been explained under certain conditions [80, 109, 171, 182]. Whether the solution of this discrete version converges to that of its continuous counterpart stably and consistently, however, has not been shown. The continuous formulation of AD is ill-posed [249, 254] while this discrete version acts as a well-posed system except for "staircasing" that is the only practically observable instability [233, 234]. Thus there is a very interesting consequence: that DAD does not agree with its continuous counterpart is accepted as a paradox in AD research.

Our first proposal, idempotent anisotropic diffusion (IAD), creates a new criterion for the selection of the CF with over-smoothing being prevented. IAD uses the non-negative part of the derivative of flux (DF) as the CF for the diffusion process, IAD is one result of our research on AD in both the continuous domain and the discrete domain. We studied the effect of the flux produced by various CFs on AD behaviour unlike previous work that concentrates on the CF only. Based on this, we provide a detailed behavioral analysis of the AD process throughout the whole gradient range. We show that a mathematically well-posed, a mathematically ill-posed or an idempotent diffusion process can be produced by the same CF. These three results correspond to when the flux is increasing, decreasing or constant, respectively. In the first case, smoothing is carried out and gradient magnitudes are consistently reduced to zero. The second case is a divergent diffusion process that eventually disappears with sufficient iterations. For edge preservation and enhancement, this unstable diffusion process should be exploited. The "staircasing" effect, regarded as either a useful feature or a drawback of AD techniques, occurs mainly in this range. The third case keeps the result image with no smoothing being applied. We show that two thresholds, edge threshold T_e and noise threshold T_n, are essential to the success of AD. Gradient magnitudes less than T_n will be smoothed while those larger than T_e will be kept. We show that the traditional edge threshold K_e is actually the noise threshold T_n. For Gradient magnitudes

between T_n and T_e, both smoothing and enhancement are possible. We explain why there is no guarantee that edges are kept by traditional AD (TAD) techniques. We show that smoothing is applied to varying degrees over the whole image and the whole gradient magnitude range by TAD techniques because threshold T_e is not incorporated in previous approaches. Since oversmoothing can occur from the first iteration, methods of seeking the "optimal" conditions to stop the diffusion process become moot.

Negative CFs that "move intensity uphill" are also explored. We show that they produce an unstable, divergent diffusion process and make the results unpredictable. Distortion cannot be avoided even with a fidelity term to control the similarity between the results and the original images.

Little research has been conducted on the discrete version of AD (DAD) since it was proposed by P-M in 1987 [171]. We address AD in the discrete domain because previous DAD approaches do not act as AD theory expects. According to AD theory [169, 171, 249], noise is smoothed by forward diffusion while edges are preserved or enhanced by less, zero or even backward smoothing. Under such selective smoothing, edges should survive the diffusion process. This is far from being the case with current DAD. We show a form of DAD that converges to its continuous counterpart stably and consistently. A new numerical scheme using the non-negative part of the DF is proposed to control the smoothing strength. This DF shows significant advantages in implementing the desired AD operation, providing a stable diffusion process guaranteed by the "Maximum Principle", and avoiding the distortion resulting from negative diffusion coefficients (DCs). The proposed IAD technique using this DF acts as AD theory expects, removing noise and trivial details and enhancing/preserving edges at the same time.

In our second proposal, we indicate that the problem of selecting a suitable value for the threshold is nearly impossible within the space of methods based on gradient magnitude because it is a problem intrinsic to gradient-magnitude-based edge estimators (GMEEs). We avoid this stubborn problem by creating a new criterion to conduct AD, employing the edge direction. We propose a new algorithm, idempotent, direction-consistent anisotropic diffusion (IDCAD) that combines the merits of IAD and those of direction-consistency-based edge estimators (DCEEs).

DCEE is much better than edge estimators based on gradient magnitude since DCEE reflects directly the spatial characteristics of real edges. These estimators are based on the fact that noisy regions exhibit high variation in edge direction while regions containing edges show high consistency in edge direction. They are immune to changes of intensity and have low sensitivity to noise. In contrast to traditional implementations of AD, in which region boundaries are located by gradient magnitude, DCEE locates the regions for conducting AD. Smoothing is implemented on regions of noise while regions deemed to contain edges are preserved.

The effectiveness of our proposed IAD and IDCAD techniques is illustrated by experiments carried out on 1D and 2D images on both artificial and real images. With our IAD technique, AD behaves for the first time as AD theory predicts, with semantically meaningful features being kept and noise smoothed throughout the diffusion process. The challenging problem of automatically stopping the diffusion process is solved because the results produced by IAD are idempotent. In contrast, diffusion never ceases with TAD techniques if the number of iterations is not preset. AD performance is improved significantly with the proposed IDCAD because of its low sensitivity to noise. Gradient-magnitude-based anisotropic diffusion (GMAD), however, is unable to smooth certain high-gradient-magnitude noise. One significant point is that our proposed algorithm uses the same set of parameters to process

all images, while GMAD thresholds must be adjusted for every image for good results.

This thesis addresses two of the most critical problems in AD research. In the chapter "Background", firstly we introduce the basic concepts of AD. Then a comprehensive overview of field of AD is presented. This overview includes research on AD theory, current AD techniques, AD's relationship with other image processing techniques, implementing AD in hardware, fields of application, types of images, image dimensions and image color, etc. Previous research related to the two top issues is studied in detail. For convenience, a summary of current AD research into these two problems is presented.

In chapter "Idempotent Anisotropic Diffusion", we show the reason why oversmoothing of semantically meaningful features occurs very easily with TAD techniques if the number of iterations has not been carefully selected. Then research on AD behavior in both the continuous domain and the discrete domain is conducted, and IAD is proposed with the non-negative part of DF to control the smoothing strength. Experiments are conducted with a number of CFs to test the proposed IAD techniques.

In chapter "Idempotent, Direction-Consistent Anisotropic Diffusion" we explain why the threshold K is difficult to determine when using GMEE. We show that combining the merits of IAD and those of DCEE permits unsupervised applications of AD to a large variety of image processing tasks.

A summary of this thesis, with discussion of future work, is presented in chapter "Conclusion and Discussion".

Chapter 2

Background

2.1 The Basic Concepts of AD

2.1.1 The Idea of AD

Anisotropic diffusion (**AD**) is an efficient, nonlinear and selective image smoothing technique. It should smooth noise and trivial parts of images and preserve their semantically meaningful features at the same time. Its basic idea is to encourage smoothing within homogeneous regions while discouraging smoothing across boundaries or edges. Because of its significant merits of noise removal and edge preservation, considerable research [243] has been devoted to the theoretical understanding and practical applications of this method since it was first proposed by Perona and Malik (P-M) in 1987 [171].

2.1.2 Formulation of AD

In their pioneer research, Perona and Malik proposed the AD equation as [169, 170]:

$$\frac{\partial I}{\partial t} = div(c(x,y,t)\nabla I) = c(x,y,t)\Delta I + \nabla c \nabla I \tag{2.1}$$

where:

I	=	intensity value of a pixel,
div	=	divergence operator,
∇	=	gradient operator,
Δ	=	Laplacian operator,
$c(x, y, t)$	=	conduction function (**CF**) that computes the spatiotemporally localized diffusion coefficient (**DC**).

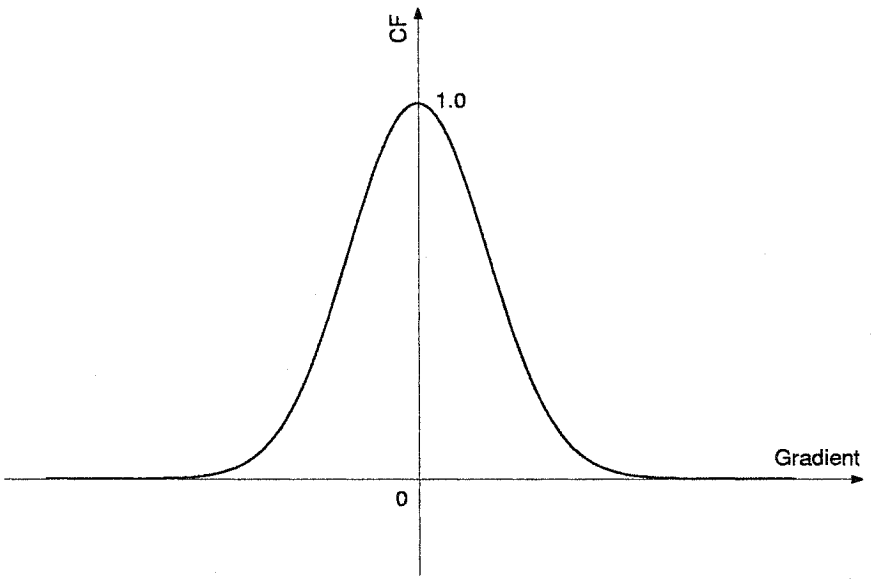


Figure 2.1: A typical diffusion coefficient function

The function $c(x, y, t)$ is chosen to be a function of the gradient magnitude:

$$c(x, y, t) = g(\|\nabla I(x, y, t)\|) \tag{2.2}$$

where $g(\cdot)$ is used to make a first estimate of the location of edges. Function $c(x, y, t)$ is a nonnegative, monotonically decreasing function of gradient magnitude, of the form of the curve shown in Figure 2.1. This function should satisfy $g(0) = 1$ and $\lim_{x \to \infty} g(x) = 0$ so that more diffusion is carried out within normally homogeneous regions of low gradient magnitudes while less or even no diffusion takes place in

regions normally containing edges and having high gradient magnitudes.

The following two functions were proposed for function $g(\cdot)$ [169]:

$$g(\|\nabla I\|) = e^{-(\frac{\|\nabla I\|}{K})^2} \tag{2.3}$$

and

$$g(\|\nabla I\|) = \frac{1}{1+(\frac{\|\nabla I\|}{K})^2} \tag{2.4}$$

where:

$\|\nabla I\|$ = gradient magnitude,

K = threshold.

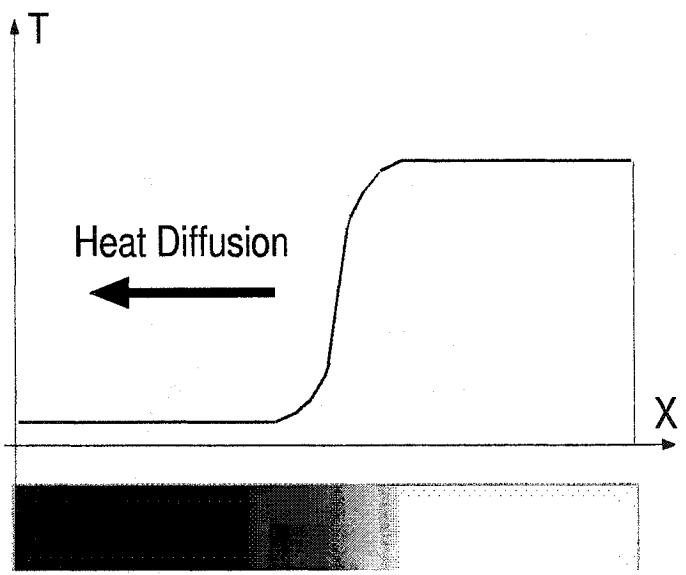


Figure 2.2: Heat diffusion. T: Temperature; X: Position

K should be chosen to reflect the range of gradients in the image. We denote flux as $\phi(\nabla I) = c(\nabla I)\nabla I$. AD theory is derived from heat equation [238]. For simplicity, a one-dimensional example of heat diffusion, as in Figure 2.2, is used for explanation of the term "flux". Assume the bar in the lower part of the figure is insulated from its environment, and its original temperature distribution is given

by the curve in the upper part of the figure. Temperature will be re-distributed with time. Heat diffuses from high-temperature areas of the bar to parts of low temperature. The flow of heat is dependent on the temperature difference between these parts. A good explanation of the physics can be found in [233].

The diffusion process in image processing is somewhat similar to heat diffusion. Flux indicates the "flow" of intensity between pixels. This flow of intensity is due to the local gradient of the image [141]. There exists a threshold K as Figure 2.3 such that flux increases monotonically for $\|\nabla I\| < K$ and decreases monotonically for $\|\nabla I\| > K$. Thus, selection of K should reflect the "strength" of edges. Ideally, the AD filter should apply smoothing in regions whose gradients are less than K, reduce smoothing strength in regions near edges and finally stop smoothing at edges.

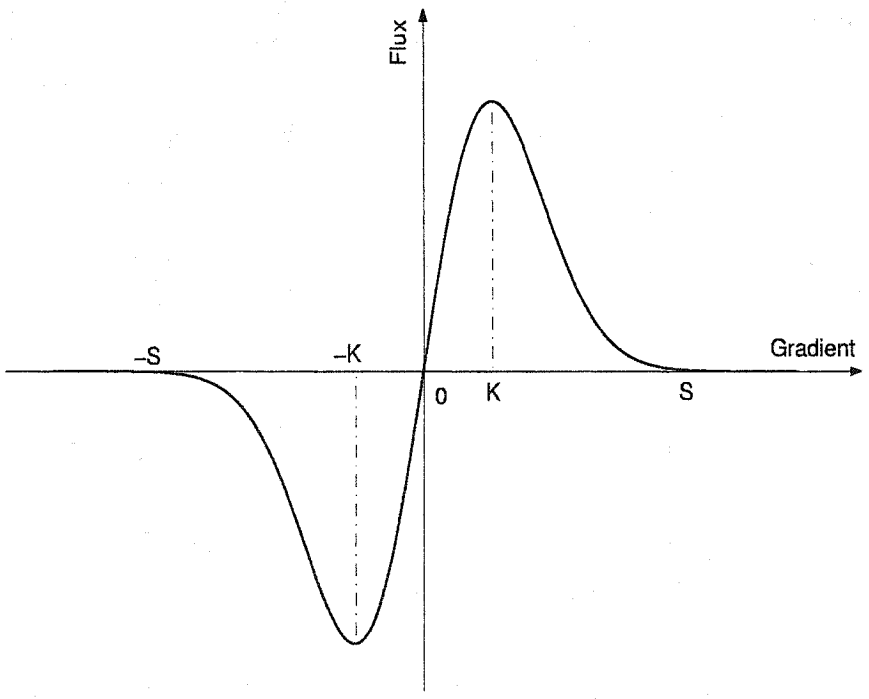


Figure 2.3: A typical flux function

2.1.3 The Discrete Version of AD

For a 1D signal, the discrete approximation of the AD operator is:

$$I_i^{t+1} = I_i^t + \lambda[c_L \cdot \nabla_L I + c_R \cdot \nabla_R I]_i^t \tag{2.5}$$

where:

i	=	pixel position,
t	=	discrete time step,
λ	=	a scalar within the range of $0 \leq \lambda \leq \frac{1}{2}$ for controlling the rate of diffusion,
L, R	=	directions, to the left and right respectively,
∇	=	difference between the nearest neighbors as given by:

$$\begin{aligned} \nabla_L I_i &= I_{i-1} - I_i \\ \nabla_R I_i &= I_{i+1} - I_i \end{aligned} \tag{2.6}$$

The diffusion coefficient is calculated by:

$$\begin{aligned} c_{R_i}^t &= g(||\nabla_R I_i^t||) \\ c_{L_i}^t &= g(||\nabla_L I_i^t||) \end{aligned} \tag{2.7}$$

where:

$g(\cdot)$	=	the conduction function (equations (2.2) to (2.4)).

The 4-nearest-neighbors discrete approximation of the AD operator as suggested by Perona and Malik is:

$$I_{i,j}^{t+1} = I_{i,j}^t + \lambda[c_N \cdot \nabla_N I + c_S \cdot \nabla_S I + c_E \cdot \nabla_E I + c_W \cdot \nabla_W I]_{i,j}^t \tag{2.8}$$

where:

i, j	=	pixel position,
t	=	discrete time step,
λ	=	a scalar within the range of $0 \leq \lambda \leq \frac{1}{4}$ for controlling the rate of diffusion,
N, S, E, W	=	directions in North, South, East and West respectively,
∇	=	difference between the nearest neighbors given by:

$$\begin{aligned} \nabla_N I_{i,j} &= I_{i,j+1} - I_{i,j} \\ \nabla_S I_{i,j} &= I_{i,j-1} - I_{i,j} \\ \nabla_E I_{i,j} &= I_{i+1,j} - I_{i,j} \\ \nabla_W I_{i,j} &= I_{i-1,j} - I_{i,j} \end{aligned} \tag{2.9}$$

and the diffusion coefficient is approximated by:

$$\begin{aligned} c^t_{N_{i,j}} &= g(||\nabla_N I^t_{i,j}||) \\ c^t_{S_{i,j}} &= g(||\nabla_S I^t_{i,j}||) \\ c^t_{E_{i,j}} &= g(||\nabla_E I^t_{i,j}||) \\ c^t_{W_{i,j}} &= g(||\nabla_W I^t_{i,j}||) \end{aligned} \tag{2.10}$$

2.1.4 Expected Merits of AD

AD is distinct from other image processing techniques in three significant, desirable characteristics.

1. It is a nonlinear, selective smoothing technique. When it is used in image processing, it should provide noise smoothing while simultaneously preserving edges, which is nearly impossible for most image processing techniques. With

the AD technique, smoothing is encouraged within homogeneous regions while discouraged across boundaries or edges.

2. Its algorithm structure makes possible local implementation in parallel hardware. If implemented in VLSI, AD filters avoid the speed restrictions of serial operations in software, thereby making feasible the implementation of real-time image processing (RTIP) at low cost.

3. Its ability of keeping semantically meaningful image feature throughout the diffusion process makes suitable for unsupervised computer vision systems.

2.1.5 Criteria of AD

The basic requirement for AD filters is that they possess the ability to treat noise and edges differently. The criteria used for evaluating AD techniques are [169]:

"1. **Causality**: Any feature at a coarse level of resolution is required to possess a (not necessarily unique) "cause" at a finer level of resolution although the reverse need not be true. In other words, no spurious detail should be generated when the resolution is diminished.

2. **Immediate Localization**: At each resolution, the region boundaries should be sharp and coincide with the semantically meaningful boundaries at that resolution.

3. **Piecewise Smoothing**: At all scales, intraregion smoothing should occur preferentially over interregion smoothing."

2.2 State of Art

2.2.1 Research on AD theory

Since the concept of AD was first introduced by P-M in 1987 [171], a considerable amount of research has been devoted to the theory and practical applications of this method for processing images. Up to now, about two hundred relevant papers have been published. They have concentrated on:

- understanding the mathematical properties of AD and its related variational formulations,
- exploiting its relationships with other image processing techniques,
- extending or modifying AD for specific applications or for improvement, and
- combining AD with other image processing methods for image enhancement.

An overview of the research in AD is presented in the following two sections. Work in this field has been categorized into two groups: theoretical understanding of AD, and the applications of AD. The former is presented in this section while the latter will be discussed in section 2.2.2.

Because research on AD has been conducted in such diverse directions and areas, we need to sub-divide both groups again. This section presents an overview of research on the theory of the algorithm, modified or extended AD, AD in combination with other image processing techniques, AD's relationship to other image processing techniques, and implementing AD in hardware. Although previous research into AD theory addressed certain kinds of problems, the theoretical algorithm of AD is much more closely related to the major issues of AD research that we selected for our research. Thus it is to be discussed together with the top problems of AD in section 2.3.

Modified AD or Extended AD

AD has been used in a wide range of applications because of its capability to combat noise while preserving semantically meaningful features [4, 102, 140, 198, 221, 245].

Lee *et al.* [125, 126] extended 2-D AD to 3-D AD with time sequences as the third dimension to remove noise in video sequences. The flickering effect is reduced because the correlation between frames has been considered. However, their method has the problem of finding a good trade-off between removing noise and preserving edges.

Maeda *et al.* [140] use region boundaries instead of gradients to control the diffusion process. In their scheme, boundaries are found by detection and linking of edges before AD is applied. Their method is problematic to implement because there is no guarantee that edges are detected correctly.

Besides only using 4-nearest neighbors in vertical and horizontal directions, Wong *et al.* [240] also consider the other 4-nearest neighbors in the diagonal directions. They declared that it improves the directionality properties of AD. Sensitivity to the effect of noise and the need for a stopping criterion are still problems. Further, however, their approach is more costly in calculation.

Instead of using the input images directly, Acton *et al.* [4, 198] use the results of morphologically filtering the image to remove noise within the image. This approach is discussed in detail in "Problem2" in section 2.3

A modified AD algorithm proposed by Huang *et al.* is interesting [99]. Instead of smoothing original images directly, their scheme implements AD on an energy space which is the same size as the original image and is created by the intensity

difference between pixels in the original image. The parameters, such as thresholds for lowest energy, highest energy and initial energy, are determined empirically and depend upon the size of image. Also, since the diffusion process is conducted by a gradient-magnitude-based method, it is still sensitive to the effects of noise and scene illumination.

Other modified AD approaches are proposed by Neoh *et al.* [153], Teo *et al.* [216, 217] and Pardo *et al.* [163, 164], who used AD to diffuse probability space instead of filtering the image. Normally this method is applied for a special reason. For example, Teo *et al.* state that because these cortial magnetic resonance images (MRI) should be segmented into only a limited number of classes, smoothing the raw image data directly is not as good as diffusing the probability space of pixel classes. They declared that their method resulted in a significant speed improvement.

Torkamani-Azar *et al.* [220] proposed a modified AD algorithm that adjusts some parameters during the diffusion process. They set λ in equation (2.8) to a large value for the first several iterations so as to achieve high processing speed. In later iterations, λ is decreased according to the noise level in the image so as to keep desirable information. Their method sounds plausible but is difficult to carry out. One problem is how to determine suitable values for the large λ and small λ, both of which are strongly dependent upon the content of images to be processed. The other problem is that they have not explained how and under what conditions the large value of λ should be reduced.

Monteil *et al.* [148] proposed a modified method, adaptive nonlinear AD, in which they select the value of K automatically and adaptively. Their main contribution is a strategy to control the "pinhole effect", an undesirable smoothing effect at edges caused by a pixel or group of adjacent pixels with gray-levels of intermediate value

near a sharp transition region. However, several parameters in their functions must be determined empirically. Further, the algorithm exhibits blocking artifacts since the image is divided into non-overlapping blocks.

AD in Combination with other Techniques

AD has been combined with other image processing techniques [9, 11, 16, 45, 57, 69, 71, 103, 109, 166, 199, 218]. Acton *et al.* [9, 11] combined AD with the resolution pyramid. In their scheme, the low resolution image representation is used to guide boundary detection at high resolution. It is similar to P-M AD except that an averaging filter is applied first. By their method, moreover, there is a strict restriction on the size of images so that pyramid calculations can be conducted. Jin *et al.* [109] proposed using the central limit theorem to select the threshold K. Their approach is discussed in "Problem2" of the major issues of AD research in section 2.3.

Instead of using a defined function of the gradient to determine the DC directly, Aja *et al.* [13] calculate the parameter through a rule table set up by fuzzy reasoning. The construction of the rule table for the fuzzy DC, however, is dependent upon the application domains in a non-obvious way. The method for setting the parameters has not been given. Though it enhances the capability for resisting speckle and additive Gaussian noise, the algorithm is more complicated and it still has the same problem of overblurring as P-M AD.

You *et al.* [244, 247] proposed a class of fourth-order partial differential equations to reduce the "blocky" effects appearing in most AD-processed images. However, their method tends to produce speckle artifacts that need to be removed by another algorithm.

Wavelets have also been incorporated with AD [166, 70]. Fontaine *et al.* [70] declared that wavelet-based multiresolution expansions give compact representations of images with piecewise smooth intensity distributions, and the use of wavelet improves the estimation of edge threshold K by assigning a small K to high contrast regions and a large K to low contrast regions. Their algorithm, however, is complex because the computation had to be carried out in both the spatial domain and wavelet domain iteratively.

Relationship of AD to Other Image Processing Techniques

A group of papers studied the relationship of AD to other image-processing techniques [8, 39, 167, 168, 188, 191, 202]. The AD technique belongs to the class of algorithms based on partial derivative equations (PDEs), which include AD, mean curvature motion [45], the min/max flow technique, active contours or snakes, etc. A detailed description of PDE-based image processing techniques can be found in [5]. Sapiro [189] tried to explore the mathematical and qualitative relationship between AD and other PDE-based operators while Kornprobst *et al.* [116] compared a group of nonlinear image operators including AD on their mathematical foundations.

The relationship between AD and local monotonicity has been investigated by Acton who used 1-D signals [6]. Fischl *et al.* [67] compared AD with the median filter and a Green's function approximation. Ramponi [179] presents certain relationships between AD and rational operators while Barash [24] shows the similarity between AD, adaptive smoothing and bilateral filtering.

After analyzing information loss in images undergoing fine-to-coarse transformation, Ferraro *et al.* [65] show that in the case of applying AD, information loss is less than in the case of isotropic diffusion because AD shows more respect for edges.

A group of image processing methods work in a similar manner to AD. Saint-Marc *et al.* [182, 183] proposed using a very small averaging mask, whose coefficients reflect the degree of discontinuity, to smooth images. They declared that their method acted similarly to AD but was more stable. Their technique is a kind of modification of P-M AD by averaging the diffusion coefficients. Thus it cannot avoid the same problems as those of P-M AD: the termination of the diffusion process and selection of a suitable edge threshold.

The electric retina is a kind of realization of AD in hardware [21]. Its object is to incorporate AD into low level vision processing for a low computational cost implemented in a parallel architecture. It is fast but poor at removing noise. The simulation results are presented within 35 iterations because there was no stopping criterion for the diffusion process.

Izquierdo *et al.* [105, 106, 107] used weighted Gaussian filter kernels to selectively smooth images. They treat an image differently by dividing it into three types of regions based on disparity variations and intensity variations. They declared that their algorithm shows some AD properties but is not as complex as P-M AD. One problem of their method is how to determine the parameters for the three types of regions. As the size of the image to be processed becomes large, requirements for memory to store relevant information for pixels become considerable. Also, their approach is not easy to implement in parallel hardware.

Implementing AD in Hardware

One of the most desirable aspects of the AD method is that it can be carried out in a parallel implementation. In implementing AD in software, however, the serial

nature of any algorithm repeated at each pixel makes the computation extremely costly. Time is required for regularizing and preserving the intermediate image for every diffusion step. This makes it unsuitable for applications in real-time systems. Up to now, very few papers have discussed implementing AD in hardware.

Andersen [21] uses AD in his proposed silicon retina model and Gijbels *et al.* [82] in their board-level structure for non-linear diffusion of images. However, neither provided a description of their implementation in hardware. Gijbels' model works in the digital domain. It needs memory to store intermediate pixel values. For simplification, their algorithms are approximated by piecewise linear functions.

Certain CMOS structures implement nonlinear diffusion and have similar characteristics to those of AD [258, 193]. Sawaji *et al.* [193] suggested using a resistive fuse implemented by a floating gate MOS transistor. This is very attractive for it can implement nonlinear diffusion with a small number of transistors. In their design, only two pairs of floating gate MOS transistors were used for each resistive fuse. However, they did not provide a means for determining the essential parameters (such as K) in their strategy.

Gulino [91] suggested an analog circuit topology for implementing AD by using the consistency of direction of the gradient as an indicator of regions containing edges. He simulated his proposed circuit with a 10 by 10 image grid and states that the results showed that it could eliminate image noise while preserving image edges. In his design approximately 1000 transistors were required for each pixel because he used a collection of available circuits to implement the various functions. Such a large number of transistors is a problem for practical applications in VLSI because of low fill-factor and large chip size, leading to high-cost devices.

2.2.2 Applications of AD

In this section our main object is to present an overview of the range of applications for AD in the field of image processing. Individual comments about their effectiveness are not given as much as in the discussion in section 2.2.1. The applications of AD is summarized by application, source of the image, image dimensions and the use of colour .

A wide range of applications for AD have been developed since its introduction. The technique has been applied, extended or modified in these applications. In most of the published papers, AD is used as an image pre-processing step or is combined with other image processing techniques. Although certain improvements are claimed in these papers, their common drawback is their strong dependence upon operator interaction. Results are unpredictable without this intervention, because:

1. The diffusion process will not stop without operator intervention or operator-defined limits.

2. Smoothing is applied along the whole range of gradient magnitudes including noise and valid edges. Thus, edges are continuously smoothed from the first iterations. This oversmoothing will become both obvious and unacceptable without operator-defined limits on the number of iterations.

In this case, results are actually a product of carefully selected parameters and *a priori* knowledge of the content of the particular images to be processed.

Field of Application

We discuss AD applications in two groups: applications in processing medical images and applications in processing other images. This classification was chosen because medical image processing is a major application area for AD [3, 20, 12, 21,

22, 28, 32, 35, 36, 37, 53, 60, 63, 74, 75, 80, 94, 96, 109, 117, 119, 120, 121, 122, 128, 133, 145, 149, 153, 156, 165, 186, 192, 196, 217, 216, 219, 227, 228, 229, 241].

Application in medical images A number of applications are concerned with the study of the brain [22, 75, 145, 217, 216, 228, 229]. Teo *et al.* [217] proposed a system to segment gray matter and create a connected representation of the gray matter for functional visualization, while Vinitski *et al.* [228, 229] used AD in studying the distribution of specific abnormal tissues in the brain.

Another significant application of AD in medical image processing is the diagnosis of cardiac pathologies [28, 63, 120, 121, 122, 186]. Research is focused on the detection of the heart contour, determining motion of the ventricular wall, and estimating heart wall thickness. In these approaches, AD is mainly used as an image preprocessing step.

Filtering mammographic images with AD has been studied [37, 38, 128]. Li *et al.* [128] applied AD twice in their scheme. The first AD step is used to remove the background from mammographic images, and the second AD step enhances microcalcifications in the resultant images.

Segmenting lesions in dermatoscopic images has also been done [53, 196]. Besides the employment of AD, another algorithm is needed for removing hairs if lesions are covered by hairs.

Vazquez *et al.* [227] used AD to extract neuron boundaries while Neoh *et al.* [153] applied AD to classify pixels based on neurons' activities. Applying AD to enhance the contrast of medical images has been studied by Boccignone *et al.* [32, 35, 36].

Several other fields have also been researched. Pathak *et al.* [165] used AD to detect prostate boundaries for diagnosing prostate disease. Haris *et al.* [94] employed AD in the extraction of vessel regions and boundaries in angoigraphic images for assessing the severity of arterial stenoses. Paplinski [162] applied AD in processing posterior capsular opacification (PCO) images that are used to present the back surface of the lens implanted during cataract operation. Using morphological AD to track leukocytes in *in vivo* video microscopy has been carried out by Acton *et al.* [4] for studying inflammatory disease.

Other Applications In this group, most papers aim for an improvement in image processing technique, and are not aimed at a specific application [16, 41, 42, 56, 62, 69, 71, 103, 111, 112, 114, 127, 130, 156, 212, 225, 236, 246, 252, 251, 253, 259]. AD has been applied, extended or modified in these applications. In the segmentation of images modeled by Markov random fields, Zhang *et al.* [259] declared that they removed the necessity for selecting the optical scale and simplified the computation by applying AD to derive a multi-scale representation of images. Burkle *et al.* [41] extended AD in flow visualization for time-dependent vector fields. Izquierdo *et al.* [103] detect the main image contours at places where the second derivative of the AD preprocessed image crosses zero in object segmentation. Ford *et al.* [71] showed that AD and directional interpolation could be combined as complementary strategies with AD smoothing noise and directional interpolation removing alias effects in structured areas.

There are a group of authors who clearly indicated their application. In the field of topography, Inglada *et al.* [101] used AD as an image pre-processing step to estimate the depth and underwater bottom surface while Sohh *et al.* [205] applied AD

to monitor and extract ice-sheet margin. AD was also applied for labeling remotely sensed images by Fernandes *et al.* [64] and for estimating the velocities of vehicles moving in poor weather conditions [184].

The application of AD also has been extended to deal with video sequences [103, 104, 126, 178]. Lee *et al.* [126] applied AD to remove noise and flickering effects in video sequences while Izquierdo *et al.* [104] extract moving objects from AD-processed image sequences. Qian *et al.* [178] perform background replacement in video images without using a physical screen. In their proposal each pixel is classified as foreground or background in a probability map. AD is used to process the probability map to reduce classification errors.

Image compression should benefit much from the applications of AD. Some papers [115, 211, 250] suggest that a high compression rate and better image quality could be achieved if AD is applied as an image preprocessing step.

Wang *et al.* [230] used AD to smooth calligraphy in their processing of Chinese characters. Interesting applications of AD are given by Giakoumis *et al.* [81] and Yang *et al.* [242]. Giakoumis *et al.* used AD to fill cracks in images of paintings. Yang *et al.* added a bias term to the standard AD to remove blocking effects and conceal lost blocks in degraded images.

Source of Images

The AD technique has been applied to various types of images. Besides digital cameras, numerous other image sources have been used. They include synthetic aperture radar images (SAR) [77, 101, 166, 205], remotely sensed images [64], magnetic resonance images (MRI) [25, 1, 92, 75, 145, 186, 191, 217, 216, 228, 229], ultra-

sound images [3, 52, 136, 165, 209, 256, 257], images obtained by electron microscopy [227], echo-cardiographic images [28, 63, 121, 122, 123], X-ray images [109, 119, 128], infrared images [10], Raman microscopic images [133, 134], images collected by laser-scanning confocal microscopes [20], images produced by computer tomography (CT) [74], and images produced by Positron emission tomography (PET) [22, 60].

Certain kinds of images show poor signal-to-noise rate (SNR). Vazquez *et al.* [227] said that images obtained from electron microscopy are extremely noisy. Images captured by a charge coupled device (CCD) [125] suffer from a kind of noise artifact that can be classified as quantum non-stationary noise having Poisson statistics. Steen *et al.* [209] declared that AD is very important for the success of volume rendering in medical ultrasound images due to the substantial noise found in them.

Raman microscopic images [133, 134] are produced by a chemical imaging technology that provides information about both the spatial distribution of structures and their chemical construction. The SNR of Raman images is very low due to weak Raman signals.

Image Dimensions

Applications of AD in 3 dimensions have been explored [20, 28, 40, 49, 60, 63, 101, 103, 186, 228, 229]. They show that AD theory also holds in the 3-dimensional case. Santarelli *et al.* [186] proposed an algorithm which segments MRIs prefiltered by AD for 3D detection and tracking of cardiac wall motion. Vinitske *et al.* [228] used AD to correct for partial volume effects in 3D segmentations of brain tumors in MRI. AD was also applied to 3D topographical reconstruction of underwater bottom [101].

Image Color

Most of the images processed by AD techniques are monochrome. Recently, several authors [26, 50, 137, 138, 187, 196, 194, 195, 201, 215] extended the range of AD application from processing monochrome images to processing colour images. Chambolle [50] proposed using AD to individually process colour images as three separate independent components. All three components must be taken into account because certain colour components may not give large gradients [196]. Guillermo *et al.* [187] take the directions and magnitudes of the maximal and minimal rate of change in the vector-image into account for controlling the diffusion process. Lucchese *et al.* [137, 138] and Bei *et al.* [215, 26] split colour images into chromaticity and brightness components and implement AD and segmentation on them individually. They produce the final processed colour images by combining the two resulting separate images.

2.2.3 Research on AD Implementation in Hardware

This section is worth a detailed description, not only because implementing AD in hardware for RTIP at low cost is our final goal, but also the benefit it will bring to the field of image acquisition, image processing and image transmission. Up to now, however, this field remains largely untouched.

The Role of Edges in Image Processing

Edges of an image contain some of the most essential information about the image. Edge detection is a crucial initial step for many image processing techniques [27, 51, 154, 226], such as feature extraction, object recognition, segmentation and compression as applied in robot and machine vision. Another significant attribute

of edges is that calculations on edges is a lower computational burden than calculations on the whole image.

An image filter makes edge detectors and image processing much more efficient if the filter is able to remove noise while preserve the meaningful features of the images. Possessing these two properties simultaneously is nearly impossible for most image filters [43]. The AD technique not only reaches these two goals at the same time, but also possesses another significant characteristics of being able to be implemented in parallel hardware. An AD filter operation can be carried out very quickly if the algorithm is implemented in VLSI.

Problem of Conventional Methods for RTIP

A typical RTIP system is shown in Figure 2.4. Scenes are captured digitally or converted to digital signals through an A/D converter, then transmitted to the image processing system [19, 144]. The image processing system is normally composed of digital signal processors (DSP), application-specific integrated circuits (ASIC) and/or computers. Such an architecture has several problems for real-time image

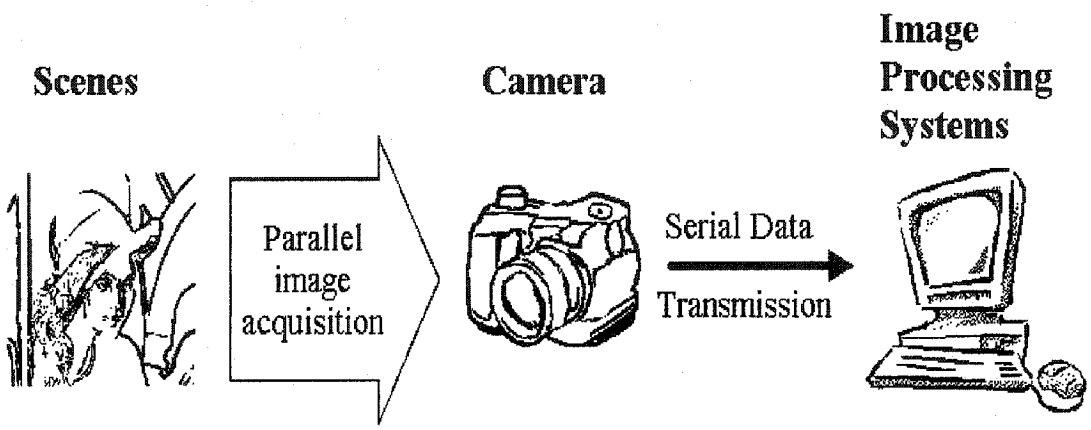


Figure 2.4: A convential real-time image processing system

processing:

Speed: As Figure 2.4 shows, scenes are acquired in a parallel way but transmitted to the digital image processing system serially. The requirements for communication, storage and processing generated by the huge amount of input image data are difficult to meet. Consider a low-resolution, image-processing system for fruit inspection at a rate of ten fruits per second [14], for example. Six color pictures are required from different angles for each fruit, each at 256x256 pixels at 8 bits. The input image data is 94 Mbyte/s flowing from the camera to the image processing system! This data volume increases rapidly if higher performance is required, such as for an industrial production line inspection system [200] where input images may be 512x512 with 24-bit color or greater.

Complexity Many sophisticated chips (memory modules, analog to digital (A/D) converter, DSP and relevant control circuitry) are needed to deal with the huge data volume. The higher the processing rate that is required, the larger the number of components that is needed, and the more complex the system becomes. For example, the number of processors required for the fruit inspection system is determined by the time available and the time one processor takes to perform the image processing task. However, increasing the number of processors does not mean a proportional increase in performance due to bus contention between these processors. When the system reaches the point of bus saturation, calculation rate cannot be improved even if additional processors are added. Obviously, the huge input image data flow is a bottleneck for the application of current RTIP systems.

Cost These chips (A/D, DSP, microprocessor, etc.) are normally expensive. This means that conventional RTIP systems are costly for real-time image processing.

AD Implemented in Hardware

AD can be implemented in hardware as Figure 2.5 shows. A capacitor C on each pixel "o" holds the charge corresponding to the brightness of that pixel. "$\varphi(\cdot)$" on every arm acts as a kind of variable resistor that controls the smoothing rate. It has

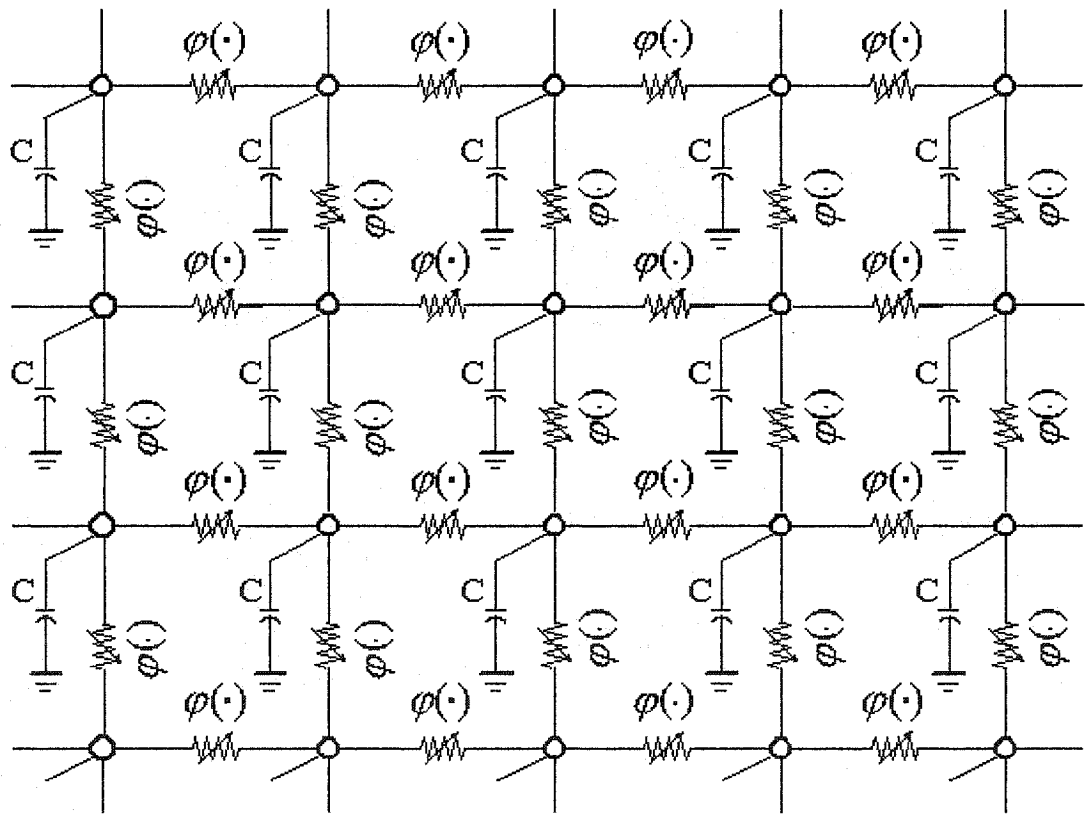


Figure 2.5: A two-dimensional grid structure implementing AD in hardware

an $I-V$ characteristic as Figure 2.6 proposed in Chapter 3 "Idempotent Anisotropic Diffusion ". A brief overview of such a system is given here. The proposed $\phi(\cdot)$

divides the gradient range into two intervals: a positive diffusion function within the range of gradient magnitudes less than K, and a negative diffusion function in the range of gradient magnitudes greater than K. The positive DC implements noise-

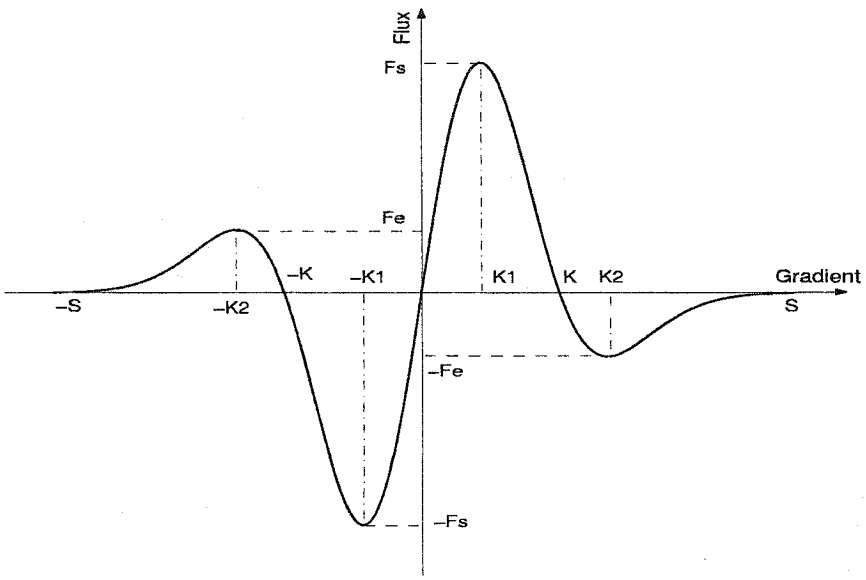


Figure 2.6: Recommended flux $\phi(\cdot)$ for anisotropic diffusion

removal within the range $[0 \ K_1)$, and smoothing and edge-enhancement within the range $(K_1 \ \mathrm{K})$. It will be shown that a negative DC produces an unstable diffusion process and should be restricted. These two ranges are treated differently because the proposed $\phi(\cdot)$ function is bimodal, with noise being smoothed while edges are preserved and enhanced.

Let us take a close look at the process of AD. For every pixel of an image except for those on the margins, its 4 nearest neighbors are as shown in Figure 2.7. $I_{i,j}$ stands for the intensity value of the center pixel. Its value at the next step is dependent upon its relationships with its four neighbors in directions E, W, N and S respectively. For example, if the difference between $I_{i,j}$ and $I_{i,j+1}$ (intensity of its north neighbor) is within K_1, smoothing is to be implemented between the center pixel and its north neighbor. If the difference is within the range $(K_1 \ \mathrm{K})$, smooth-

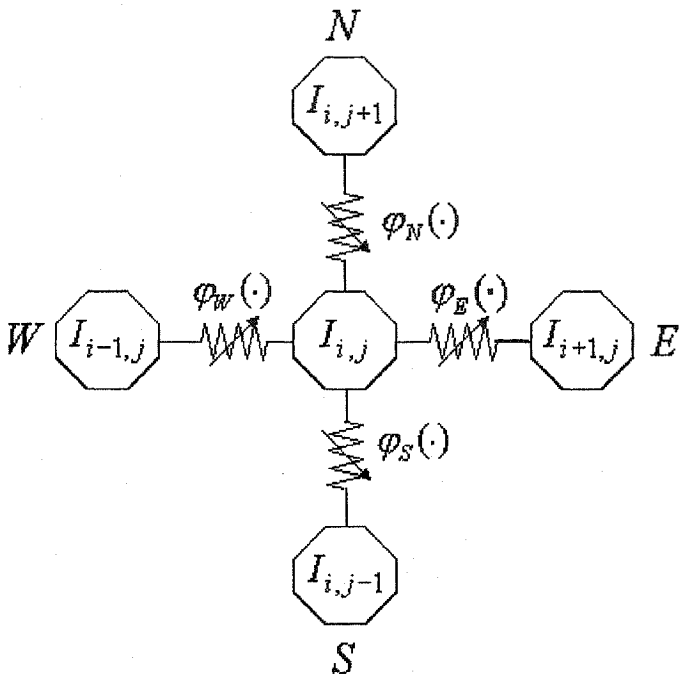


Figure 2.7: Detailed structure of a pixel and its neighbors in AD hardware

ing or enhancing is to be carried out. On the other hand, if the difference between $I_{i,j}$ and $I_{i,j-1}$ (intensity of its south neighbor) is larger than K, enhancement will be implemented. In this way, anisotropic diffusion is carried out between the center pixel and its neighbors. AD can be implemented for every pixel in the image by using this regular structure. Thus, a high image processing speed can be achieved with this pixel-parallel implementation.

2.3 The Major Issues of AD Research

Though considerable research has been devoted to AD, it is still a long way from practical application to image processing, especially for computer vision tasks in which the machine makes decisions. Two major issues remain unsolved since the

concept of the AD theory was proposed in 1987: 1)automatically stopping the diffusion process and 2)choosing a suitable value for the threshold for edges.

2.3.1 Problem 1: Automatically Stopping Diffusion

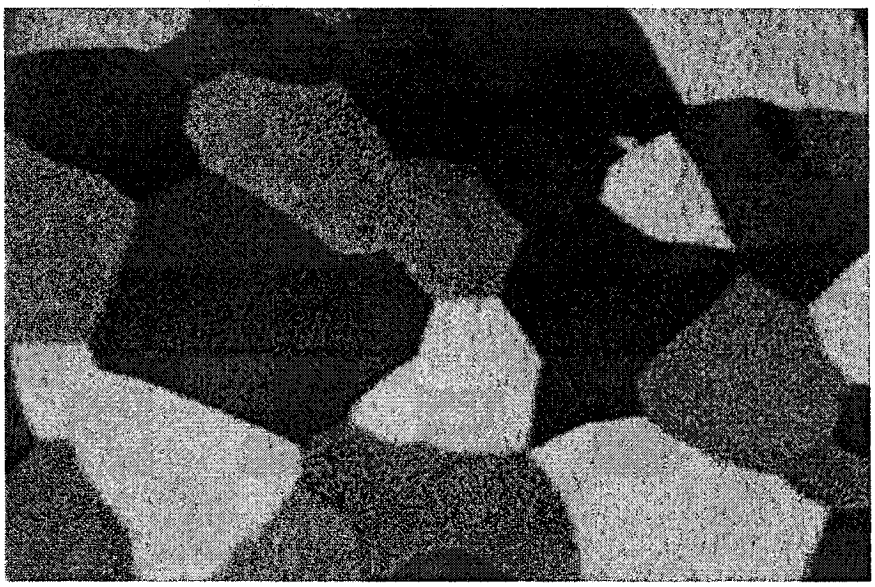
The Problem: Over-Blurring

With current AD techniques, over-blurring of semantically meaningful features occurs very easily if the number of iterations is not selected carefully. Without a preset number of iterations, diffusion never ceases. This finally produces an uniform grey result with no edges at all! This problem is illustrated in Figure 2.8[1]. Figure 2.8(b) shows a result obtained by a current AD technique after 5000 iterations. It shows that the image is smeared and edges are badly damaged. Poorer results can be expected with more iterations since the smoothing does not end until an uniform grey result is produced.

Current Approaches

The most common way to treat this problem is very simple: restrict the number of iterations before the effect of over-blurring becomes apparent [7, 33, 34, 47, 79]. In this way, smoothing is stopped manually if "good" results have been produced. There is no standard for such "good" results. Images in which noise seems to be smoothed while edges have not been blurred obviously would be accepted as "good" results. The number of iterations depends on the content of images and people's sense of "goodness". Generally, the number of iterations is not larger than 100 in the papers published.

[1] "alumgms", a test image in the Matlab image processing toolbox, is used as a sample image added with Gaussian noise ($mean = 15$ and $\sigma = 15$).



(a) Original image

(b) Result of Over-blurred Image

Figure 2.8: Problem of over-bluring for current AD techniques

A plausible solution to this problem seems to concern the diffusion coefficient (DC) produced by the CF because it controls the diffusion process. Considerable research into the effects of the CF function on convergence and on selection of the CF has been carried out [248, 109, 124, 175, 246, 249, 254, 255]. You *et al.* [249, 254, 255] regard AD as a process of dissipating energy with time. They treat it as a process that minimizes an energy function based on the shape of the energy surface. They demonstrated that a unique global minimum of the energy functional will lead to well-posed AD while a number of global minima distributed densely on the image space can cause ill-posed AD. Their research has been complemented by Hollig *et al.*'s work on the diffusion equation [98] which states that an 1-D AD process is well-posed if and only if:

$$(x \cdot c(x))' \geq 0 \tag{2.11}$$

where in image processing:

x = gradient magnitude defined as equation (2.9)

$c(x)$ = CF function as in equation (2.3) or (2.4)

You *et al.* developed an additional condition for anisotropic diffusion for the case of 2D images:

$$\lim_{x \to \infty} x \cdot c(x) \neq 0 \tag{2.12}$$

where x and $c(x)$ have the same meanings as in equation (2.11).

You *et al.* believe a CF function, $c(x) = \frac{1}{x}$, guarantees a well posed AD. Since this CF cannot deal with the case of $x = 0$, they recommend a CF as follows:

$$c(x) = \begin{cases} \frac{1}{T} & \text{if} \quad x < T, \\ \frac{1}{x} & \text{if} \quad x \geq T \end{cases} \tag{2.13}$$

where:

T = a parameter specified for different applications.

This function seems problematic in practical application since it produces constant (except in the range [-T T]) flux along the whole gradient magnitude.

Black *et al.* [30, 31] adopted three functions (the Lorentzian function, Tukey's biweight function and Huber's minimax function) as the CF. Tukey's biweight function is good because it forces the diffusion to zero for gradient magnitudes larger than a certain range.

$$c(x) = \begin{cases} \left(1 - (\frac{x}{K_e})^2\right)^2 & , \ |x| \leq K_e \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

where:

K_e = the threshold.

Torkamani-Azar *et al.* [220] tried to make improvements by working on both the CF and the parameter λ in the discrete version of AD (equation (2.8)) simultaneously. In their proposal, the CF is:

$$c(x, y, t) = \frac{k_1}{2} e^{-k_1 \cdot (|G_x| + |G_y|)} \tag{2.15}$$

where:

k_1 = parameter controlling the DC function,

G_x, G_y = gradient magnitude components in x and y direction respectively.

They state that both λ and k_1 should be specific to each image and adjusted during the diffusion process. A small k_1 is more effective at removing noise and so it is chosen for images with low SNR, while a large k_1 is more reasonable for images with high SNR because it does not smooth as much. Because SNR increases with smoothing, at the beginning of smoothing a large λ should be employed for a high speed, and in the later iterations it should be reduced to weaken the smoothing effects for keeping the best results.

Both of the parameters are image-sensitive. As well as the selection of the threshold, k_1 and the function that describes λ as a function of iteration number must be determined. Their results were restricted to only 20 iterations.

Creating an automatic stopping criterion is another way to stop AD that is attractive to many researchers [131, 132, 207, 206]. They noticed that as diffusion proceeds, images are improved. It seems that versions of resultant images could be optimal after a certain number of iterations, and then degrade afterwards. Their basic idea is to find the conditions that indicate this optimality. Knowing these conditions, the diffusion process could be stopped when they occur to keep the best results from over-blurring.

Lin *et al.* [132] proposed a function to measure local smooth levels. For pixel (i, j), their smooth-level-measure function is:

$$S = \frac{N_s}{N_t} \tag{2.16}$$

where:

S = smooth level measure function,

N_s = number of pixels deemed as smoothed within the neighborhood of pixel (i, j),

N_t = total number of pixels within the neighborhood of pixel (i, j).

They regard a pixel as smoothed if it satisfies:

1. Its gradient magnitude is less than a threshold defined for edges, such as the "K" in equations (2.3) and (2.4). K is determined by a selected percentage of the whole range of gradient magnitudes in the image.

2. Its second order derivative is less than another threshold K_1.

They state that the value of S in equation (2.16) increases with successive iterations as Figure 2.9 shows. They created an automatic diffusion stopping criteria

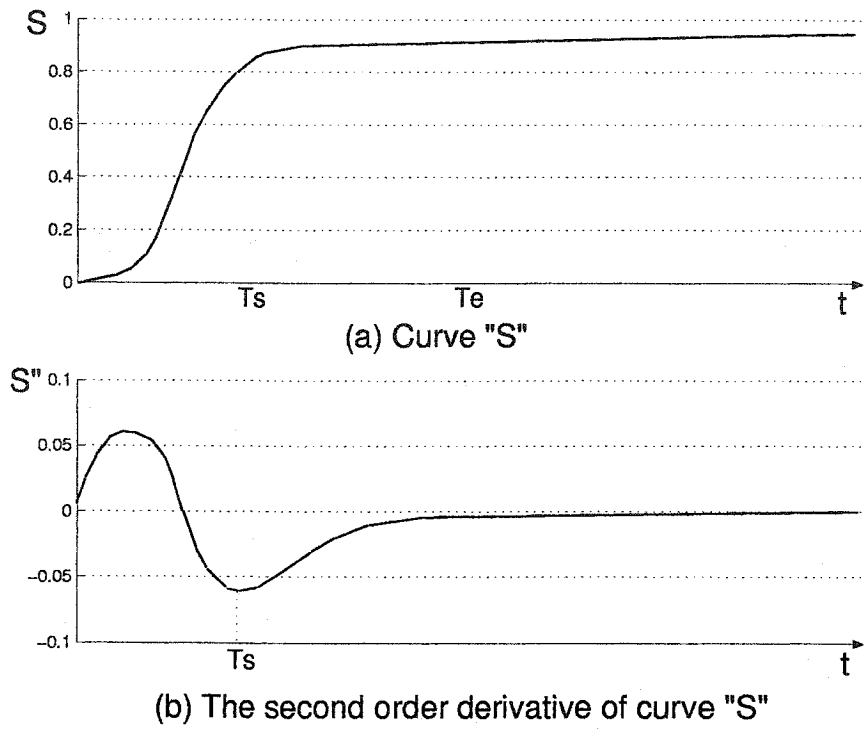


Figure 2.9: Automatic stopping time defined in the research of Lin *et al.*

by choosing:

$$T_e = K_2 * T_s \tag{2.17}$$

where:

T_e = number of iterations when diffusion ends,

T_s = number of iterations when noise is believed to be smoothed,

K_2 = a scale controlling the number of iterations.

All the parameters (K, K_1, K_2), however, are image sensitive and are determined heuristically.

Liang *et al.* [131] proposed a method to automatically stop the diffusion process by incorporating a noise estimator. For pixel (i, j), their noise estimator is:

$$n(\alpha, \beta, \gamma) = \beta \frac{f_z(\alpha, \beta, \gamma)}{f_g(\alpha, \gamma)} \tag{2.18}$$

where:

$n(\cdot)$ = noise estimator function,

α = direction of pixel (i, j),

β = parameter controlling the strength of noise estimator,

γ = distance from pixel (i, j) to a neighbor pixel,

$f_z(\cdot)$ = function of zero-crossing of second order derivative,

$f_g(\cdot)$ = function of gradient magnitude.

Their noise estimator is mainly based on both the gradient and the second order derivative, to help make a judgment about the location of edges. Smoothing is stopped on pixels either with significant gradient magnitude or having a significant zero-crossing of a second-order derivative. Parameters in equation (2.18) need to be selected carefully. For example, a large β will remove weak edges and small structures while a small β can enhance trivial details and produce "staircase" effects. Experimental results of 60 iterations with noise still remaining in them, however, are presented for their proposed method.

Researchers realize that when to stop is crucial to AD techniques. Solo [206, 207] developed a complex optimal stopping-time estimator for a 1-D signal as follows:

$$R_h = \frac{1}{N}\sum_{i=1}^{N} e_i^2 - \sigma^2 + \frac{2\sigma^2}{N}\omega_h \tag{2.19}$$

with

$$\omega_h = trace\left(\frac{\partial f^{(t)T}}{\partial y}\right) \tag{2.20}$$

$$e_i = f(\frac{i}{N}) - f^{(t)}(\frac{i}{N}) + \varepsilon_i \tag{2.21}$$

where:

R_h = estimator of stopping time,

N = number of pixels,

e_i = white Gaussian noise of zero mean and variance σ^2,

$f(\cdot)$ = input 1-D signal.

He declares that his proposal is the first to create an automatic stopping criterion for AD. He states that a minimum number of iterations can be found on the curve of R_h obtained by calculating equation (2.21) step by step. His research has been deeply influenced by the work of others on the effect of the CF. He treats their conclusions as theorems. His proposed estimator is empirical and requires a set of parameters that are chosen or known in advance.

The discrete version of AD (DAD), however, has not been sufficiently researched since AD theory was introduced in 1987 [171]. Among the large number of published papers on AD research, very few of them deal with the numerical aspects of AD [213, 231, 233, 235]. The 4-nearest-neighbors discrete AD operator proposed by P-M, equations (2.8) to (2.10), has been widely employed in AD's applications and research up to today [4, 5, 8, 6, 9, 10, 11, 13, 21, 23, 24, 30, 42, 60, 81, 63, 91, 92, 109, 127, 132, 133, 134, 140, 147, 163, 169, 171, 175, 178, 179, 184, 197, 198, 199, 213, 220, 224, 256, 240].

The performance of the discrete version of AD, however, does not agree with its continuous counterpart. According to AD theory [171], both forward diffusion and backward diffusion should be carried out in the image so that it can implement noise-removing and edge-preserving/enhancing at the same time. It should be able to enhance and keep edges throughout the diffusion process since no or only inverse diffusion is supposed to apply on them. No edge, however, is able to survive the diffusion process carried out by P-M DAD if the number of iterations were not limited in advance. This results in a very interesting fact in AD research. The continuous

formulation of AD is ill-posed [249, 254] while this discrete version acts as a well-posed system except for "staircasing" as the observable instability [233, 234, 235]. The disagreement between the discrete version of AD and the continuous version of AD has been accepted as a paradox. There has not been much research carried out on DAD but wide application of P-M DAD.

Modifications have been proposed for P-M DAD. A fidelity term to deal with the problem of oversmoothing is used by many researchers in simulating their proposals [47, 48, 83, 84, 180, 231].

$$I_t = div(c(x, y, t)\nabla I) - \lambda(I - I_0) \tag{2.22}$$

where:

I_0 = the original image,

λ = the fidelity term.

The fidelity term controls the difference between the result image and the original image. A small fidelity term gives more freedom for the result image while a large one requires greater similarity between the result image and the original image. However, choosing a value for λ is image dependent.

Several authors sought to make the parameters, such as the scalar λ or the Gaussian G_σ, time dependent [129, 201, 220, 237]. Their goal is high processing speed in the first iterations and gradually reduced smoothing strength with time to keep desirable information. Determining the values of the parameters and how they change are problematic.

P-M DAD has been explained under various ways [80, 109, 171, 182]. It has been inferred from the continuous version of AD with restrictions that the DC coefficients

are non-negative and they must sum to unity [91, 182]:

$$c^t(i-1) + c^t(i) + c^t(i+1) \equiv 1 \tag{2.23}$$

with

$$0 \leq c^t(i-1), c^t(i), c^t(i+1) \leq 1 \tag{2.24}$$

where:

$c(\cdot)$ = conduct coefficient,

t = simulation time step,

i = pixels' position.

or under the requirement that the discretized space steps are one exactly [80, 109, 171]:

$$\begin{cases} \Delta x \equiv 1 \\ \Delta y \equiv 1 \end{cases} \tag{2.25}$$

A critical question that whether the solution of this discrete version converges to that of its continuous counterpart stably and consistently, however, has not been given. We believe that the DAD, an area remaining nearly untouched since it was proposed for AD research, should be addressed for its importance in validly implementing proposed AD schemes.

2.3.2 Problem 2: Finding a Suitable Value For Thresholds

The Problem: No Distinction Between Noise And Edges

Making filters more robust to noise is not a problem specific to AD techniques. It is a problem common to methods dependent upon the gradient because the gradient is not reliable. In some case, when noise is substantial or speckle noise is present, noise and edges cannot be distinguished based on gradient magnitude. In such a

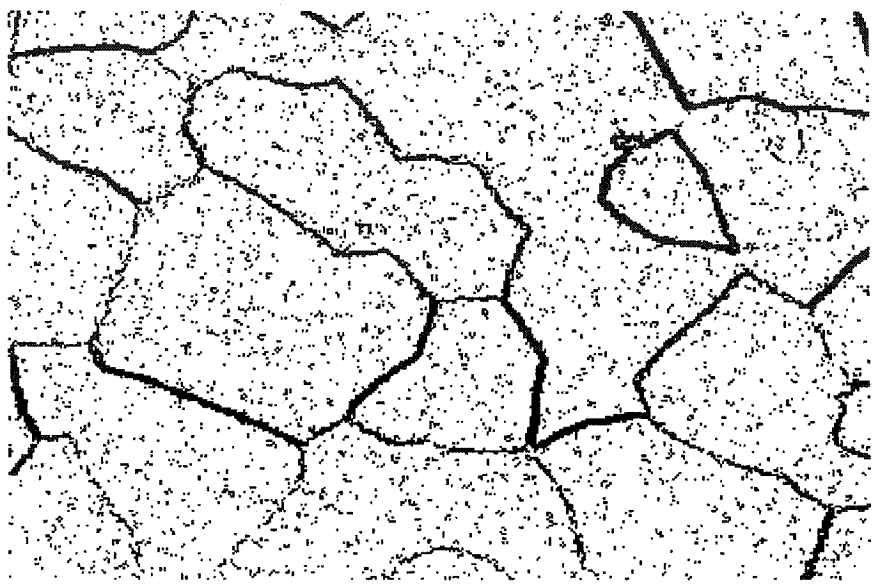
case noise will be enhanced instead of being removed. In order to reduce the uncertainty caused by noise, contextual information from neighboring pixels should be incorporated into the AD algorithm.

Ideally, threshold K should be chosen on the basis of the "strength" of edges. Choosing a global parameter for K does not work well with the diverse, spatially varying conditions in typical images. For current AD techniques, determination of a suitable value for K is a problem because it is image dependent and becomes very complex with the appearance of noise and variable intensity levels.

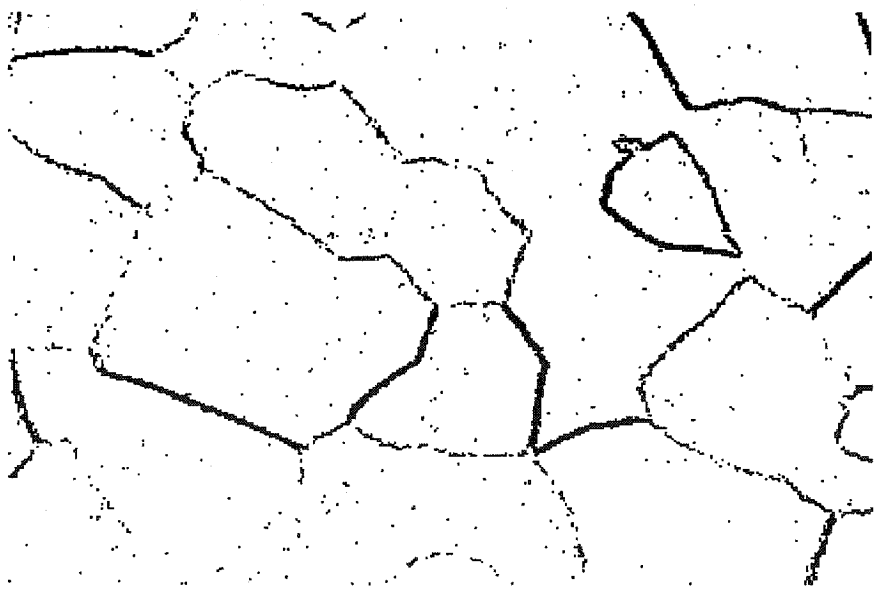
This problem is shown clearly in Figure 2.10. It shows the edges detected from Figure 2.8(a) by a noise estimator recommended by Perona and Malik. This estimator has been employed widely in AD research for the determination of the threshold K. It assigns K a value based on a percentage of total pixels in an image. In Figure 2.10(a), for example, 90% of pixels are regarded as noise, and so K is selected to retain 10% of the pixels as edge pixels. If K is selected so that there are many edge pixels, most edges are detected but noise is unavoidably detected as edges at the same time, as shown in Figure 2.10(a). Setting K to reflect a lower percentage of edge pixels (Figure 2.10(b)), less noise is detected but parts of edges are also missed. Thus, gradient-magnitude-based edge estimators (GMEE) are sensitive to the effect of noise. A difficult, usually impossible, trade-off between detecting edges and not including noise must be made for all image processing techniques based on GMEE.

Current Approaches

Research on edge detection has been conducted widely in the field of image processing. In this review the focus is on methods applied in the area of AD research.



(a) Edges detected by choosing K based on 90%



(b) Edges detected by choosing K based on 96%

Figure 2.10: Problem of detecting edge based on gradient magnitude.

GMEE detects local changes in an image. The principle is illustrated in Figure 2.11, which uses a 1-D signal for explanation. For the edge in signal $f(x)$, there is a local peak in its first derivative $f'(x)$, and a zero-crossing in its second derivative $f''(x)$. GMEE locates the edges of an image by detecting the maximum of gradient magnitude and/or zero-crossing in the second derivatives. Instead of detecting the exact point where the gradient magnitude reaches a peak or the second derivative crosses zero, in practice an interval defined by a threshold is used to determine edges because of illumination fluctuation. This threshold should be selected carefully so that only pixels with strong edge information are regarded as edges. As Figure 2.11 shows, signals within the interval $[a\ b]$ will be all regarded as edges because their gradient magnitudes are larger than the preset threshold K.

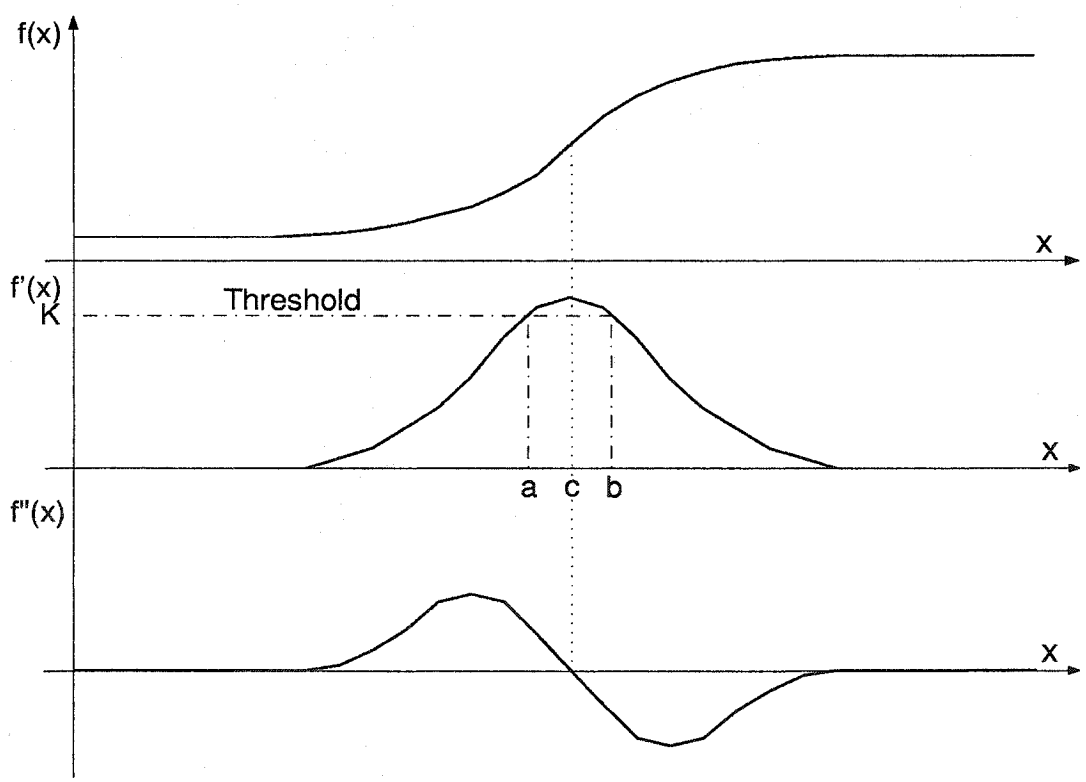


Figure 2.11: Principle of edge detection by gradient

The way of choosing the threshold based on a percentage, used widely in the field of AD research, has already been illustrated in Figure 2.10. It consists of the following steps:

1. Calculate the absolute values of the gradient magnitudes throughout the image.
2. Assume a certain percentage of the pixels in the image to be edges.
3. Get the smallest gradient value from the pixels regarded as edges.
4. Set K equal to the value of the smallest gradient got in step 3.
5. Repeat step 2~4 until an acceptable result is produced.

Determining K in this way is very image-dependent process.

Using a histogram to determine a value for K is another popular method [29, 87, 139, 223]. Jin *et al.* [109] proposed an adaptive nonlinear diffusion scheme by using the central limit theorem to choose the threshold. They declared that methods based on the histogram are not good because these methods require a bi-peak histogram for the determination of K, and this assumption is not met for most images. To deal with single-peak histograms, Jin *et al.* assume that the background has either a Gaussian or a Rayleigh distribution. When the number of background pixels is large enough, a Gaussian distribution is used:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (\sigma > 0) \tag{2.26}$$

with

$$\mu = \max_{i \in S}(s_i) \tag{2.27}$$

$$\sigma = \sqrt{\sum_{i \in S} 2(s_i - \mu)^2} \tag{2.28}$$

where:

s_i = the sampled data

$i \in S$ = pixels on the same side of the histogram against the mean value.

A Rayleigh distribution is employed if the number of background pixels is not large enough.

$$p(x) = \begin{cases} \frac{x}{\mu^2} e^{-\frac{x^2}{\mu^2}} & \text{if} \quad x \geq 0 \\ 0 & \text{otherwise} \end{cases} \qquad (\mu > 0) \tag{2.29}$$

with

$$\begin{cases} \mu = \sqrt{\frac{\pi}{2}} \max\limits_{i \in S}(s_i) \\ \sigma = \sqrt{\frac{4-\pi}{2}\mu^2} \end{cases} \tag{2.30}$$

where s_i and $i \in S$ have the same meaning as in equation (2.28).

In implementing their proposed algorithm, one has to determine:

1. the number of background pixels that is more suitable for employing the Gaussian model instead of Rayleigh model, or *vice versa*, and

2. the size of mask optimal for processing the image.

Their scheme is designed for handling single-peak histograms with the assumption that there is only one object and background in the processed image window. No consideration is given to the case if the histogram contains more than two peaks.

Some authors sought to make the diffusion process less susceptible to noise [89, 90, 120, 237]. Catte *et al.* [46] modified the AD equation (2.1) to:

$$\frac{\partial I}{\partial t} = div[g(\|\nabla(G_\sigma * I)\|)\nabla I] \tag{2.31}$$

where:

G_σ = Gaussian smoothing operator of scale σ.

This equation smooths the image with a Gaussian operator to reduce the effects of noise before estimating the location of edges. It yields more consistent performance by estimating the gradient of the smoothed image $G_\sigma * I$. Though it shows an improvement in dealing with ramp edges in comparison with Perona and Malik's operator, it removes true detail edges and causes distortion, especially if noise is considerable or if G_σ is large.

That using Gaussian operators of the same scale throughout the diffusion process is not reasonable has been noticed [129, 237]. With iteration, the image becomes more smoothed, thus edge confidence in subsequent versions of the image has been increased. Whitaker *et al.* [237] modified the algorithm of equation (2.31) by decreasing the scale parameter with time to create $G_{\sigma(t)}$. However, determining the decrease rate of $G_{\sigma(t)}$ is problematic. If $G_{\sigma(t)}$ decreases too slowly, it does not have much effect and may still let important edges be smoothed. On the other hand, if it decreases too quickly, unwanted luminance fluctuations may not be smoothed. They did not suggest a method for choosing $\sigma(t)$ to get optimal results. Also, they noticed that even without noise, "staircase" effects can arise in areas with low gradient.

Other image processing techniques have been borrowed for dealing with this problem [4, 57, 109, 175, 177, 198], such as the method of using wavelets for choosing K [166, 70], discussed in section 2.2.1. In the morphological AD proposed by Acton *et al.* to preserve certain shapes at certain scales [4, 198], the CF (2.3) is modified to be:

$$g(\|\nabla I\|) = e^{-(\frac{\|\nabla\{[(I \circ B) \bullet B]\}I\|}{K})^2} \tag{2.32}$$

where:

B	=	structuring element,
$I \circ B$	=	the morphological opening of I by B,
$I \bullet B$	=	the morphological closing of I by B.

Morphological operators are used before detecting edge locations. The morphological filters are used to eliminate noise and its effects while distorting regions as little as possible. This method shows some improvement in the performance of estimating edge locations if the structuring element is set appropriately. Choosing B and K is image dependent.

Dang *et al.* [57] combines the symmetric nearest neighbor (SNN) filter, an image enhancement technique, with the AD filter. The SNN filter uses both spatial and intensity information within the neighborhood of the pixel of interest. It deals only with pairs of pixels that are symmetrically opposite about the pixel of interest and have the closest intensity values to the pixel of interest. They state that the SNN filter is good at removing noise while preserving edges, thus reducing the AD filters' sensitivity to noise. In their proposed scheme, several other thresholds need to be determined as well as K.

Gregson [89] suggested a way to estimate the location of edges by using the consistency of gradient direction as an indicator of edge region.

$$E\{\alpha^2\} = 2\left[1 - E\left\{\frac{|\sum G_m|}{\sum |G_m|}\right\}\right] \tag{2.33}$$

where:

$E\{\alpha^2\}$	=	the expected angular variation,
G_m	=	gradient magnitudes of pixels within the pixel of interest.

Gradient direction is one of a pixel's attributes and has been largely ignored in edge detection. The proposed algorithm (equation (2.33)) is less sensitive to noise

and is independent of image intensity. This approach is much better than using the gradient magnitude as the indicator of an edge region as it reflects directly the spatial characteristics of real edges. Its computational complexity and the selection of block size, however, are of concern for real-time applications.

Weickert proposed a coherence-enhancing anisotropic diffusion technique (CEAD) that employs orientation instead of direction to implement smoothings [232]. He defined that "gradients with opposite sign share the same orientation, but point in opposite directions.". He suggested a structure tensor for calculating the average orientation as follows.

$$J_\rho(\nabla u_\sigma) = K_\rho * (\nabla u_\sigma \otimes \nabla u_\sigma) \tag{2.34}$$

where:

J_ρ	=	the structure tensor, also called as interest operator or second-moment matrix,
$*$	=	convolution operator,
K_ρ	=	a Gaussian filter with a scale ρ that should reflect the characteristic of the texture,
$\otimes$	=	product operator,
∇u_σ	=	gradient computed from a version of the image smoothed by a Gaussian filter with scale σ.

The common smoothed gradient ∇u_σ is replaced by equation (2.34) that experiences Gaussian filtering twice. The production after the first smoothing, $\nabla u_\sigma \otimes \nabla u_\sigma$, is used to prevent the gradients of opposite sign from cancelling each other when the average gradient is calculated within the neighbourhood of a pixel. This result is smoothed again by the second smoothing factor K_ρ. The eigenvalues of the structure tensor, $\mu_1, \cdots, \mu_n$, are used as a measure for a coherence κ defined as:

$$\kappa = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (\mu_i - \mu_j)^2 \tag{2.35}$$

Weickert states that the κ is used to control the smoothing strength and his CEAD may be carried out by the following way:

$$\partial_t u = div(D\nabla u) \tag{2.36}$$

where:

D = the diffusion tensor that controls the smoothing strength.

The eigenvectors of D are assigned in the same way as J_ρ and are constructed as follows:

$$\lambda_i = \alpha \quad \text{for} \quad i = 1, \cdots, n-1, \tag{2.37}$$

and for $i = n$:

$$\lambda_i = \begin{cases} \alpha & \text{for} \quad \kappa = 0, \\ \alpha + (1-\alpha)e^{-\frac{C}{\kappa}} & \text{otherwise} \end{cases} \tag{2.38}$$

where:

C = a threshold for measuring the coherence,

α = a positive parameter $\in$ (0, 1) for keeping the smoothness of the structure diffusion tensor.

Weickert recommends using a look-up table to implement the algorithm. His method is based on a processed version of the image smoothed by Gaussian filters that blur signals though it reduces the sensitivity to noise. Parameters (the Gaussian filter scale ρ and σ, the positive parameter α and the threshold C) are image-dependent. Implementing this algorithm in hardware would be difficult due to its complexity.

2.4 Summary of Current AD Research

Considerable research on AD has been conducted since it was first introduced. It is encouraging to see that AD has already been used in such a wide range of appli-

cations and applied to such a large variety of images.

Research in automatically stopping the diffusion process is very challenging and remains open. In a typical AD process, at first AD removes noise and undesirable details. If it could be stopped at this point, an improved image would result. However, normally diffusion continues and the image is blurred and degraded. Also, the number of iterations for diffusing the image is different from one region to another. In regions of small fluctuation of gradient, only a few iterations are enough. On the other hand, regions of large gradient need more iterations due to their low DCs. Obviously, knowing when to stop the diffusion process is critical to the success of implementing AD. Three ways have been used or exploited to deal with this problem in the literature:

1. simply stopping the diffusion process with a preset number of iterations, thus the effects of over-blurring are not easy to observe,

2. studying the characteristics of the CF, in the hope of finding a good DC function that controls the smoothing strength as AD theory expects, and

3. seeking a criterion for automatically stopping the diffusion process when the result are "optimal".

The discrete implementation of AD first proposed by P-M has been used continuously since it was proposed in 1987. It is known that this DAD acts differently from its continuous counterpart. However, this is accepted as a paradox with very little further research undertaken on it.

Making edge detectors more robust to noise is an old problem for all image processing techniques based on gradient magnitude. It is also very important to the success of AD filters. A problem common to edge detectors for current AD filters is to select a suitable value for the threshold K. This threshold divides an image

into two groups: noise and edges. The performance of AD filters is dependent upon the selection of K because it determines in what way a pixel will be treated: being smoothed or being preserved/enhanced. For current AD filters, the determination of K is still empirical. It is image sensitive, and so the lack of a "goodness" measure prevents application of AD to computer-vision tasks where it is the machine that makes decisions.

In the field of AD research, methods for determining the threshold are mainly as follows.

1. Select K to detect a desired percentage of the total pixels in the image. This is the dominant approach due to its easy implementation.

2. Filter images with other image processing techniques before estimating edge locations for implementing AD smoothing. In this way performance of edge estimators could be improved to some degree. Such methods mitigate the difficulty of selecting the threshold, but at the cost of implementing these pre-processing techniques and their side effects.

3. Employ other techniques, such as methods based on histogram, wavelet or edge direction etc. Difficulty in implementing these schemes and the complexity of their algorithms are the main concerns.

Another very attractive AD research field is its application to real-time systems. AD meets the requirement of real-time applications since one of the most desirable aspects of the AD algorithm is that it can be carried out in a parallel implementation. Among the published papers concerning AD, however, very few of them are concerned with implementation in hardware. In a software implementation, its intrinsic iterative nature at each pixel makes the computation extremely costly. As a result, software implementations are not suitable for real-time systems, because

software implementation does not exploit the inherent parallelism of the algorithm. Research into theory and implementation for hardware realization of AD is necessary for implementations of AD that provide real-time performance.

In conclusion, research on the problem of automatically stopping the diffusion process, and on finding an edge detector that is robust to noise, are prerequisites to the application of AD techniques for computer vision systems, especially for RTIP systems.

Chapter 3

Idempotent Anisotropic Diffusion

The basic idea of AD is to encourage smoothing within regions while discouraging smoothing across boundaries or edges. If an image is processed under such an AD principle, features finally left in the image after AD processing should be meaningful edges. Edges should survive the AD smoothing process because weak or no smoothing should be applied to them. However, traditional anisotropic diffusion (TAD) techniques do not behave in the manner predicted by AD theory. Under TAD, meaningful image features cannot be preserved if the number of diffusion iterations is not carefully selected [2, 13, 23, 80, 109, 147, 179, 185, 246, 255, 261]. In a typical TAD process, over-smoothing occurs very easily, blurring semantically meaningful features. Instead of preserving or enhancing edges, TAD filters continually smooth all parts of an image, although to varying degrees. Without a preset number of iterations, diffusion never ceases. Determining the stopping criterion for the AD technique is extremely problematic [132, 207, 206].

This section addresses overblurring, one of the most critical problems. The discussion is in two parts: a review of TAD and our proposed IAD. In the first part, the behavior of TAD techniques is discussed, and why current approaches do not act as desired is analyzed. The proposed IAD covers three topics: analysis of AD behavior

along the whole range of gradient magnitude, a discrete version of AD (DAD) that agrees with its continuous version, and finally, our proposed IAD technique that solves the problem of overblurring.

3.1 Analysis of Traditional Anisotropic Diffusion

3.1.1 TAD Techniques: A Form of Averaging Filter

Equations (2.8)–(2.10) do not permit AD to behave in the desired manner. According to AD theory, if an image is processed by the AD technique, more smoothing should occur in areas with gradients smaller than the threshold K shown in Figure 2.3 while less or no diffusion is implemented for edges with gradients larger than the threshold. Because the flux curve in Figure 2.3 is positive over the whole gradient range, it should be noted that as a result, semantically meaningful edges could be smoothed at each iteration even if their gradient magnitudes are much greater than K. This means that with each iteration, they will be smoothed by increasingly greater amounts. Edges disappear if the number of iterations is not carefully chosen. Also, it is hard to tell when the point of optimum diffusion is attained, where noise has been removed while edges are preserved. Selection of the number of iterations is empirical and image-dependent. Finding a stopping criterion is a challenge for TAD research.

It may be that such an optimal diffusion result can never be attained with TAD. As equation (2.8) and Figure 2.3 show, the total change for a pixel in each iteration is the sum of changes in four directions. How TAD acts is clear if we consider those changes individually. The positive DC function produces smoothing at every pixel as Figure 2.3 shows. No matter how slight the smoothing in high-gradient regions, the gradients in those regions are reduced. At the first iteration, smoothing is slight

in regions deemed to contain edges. With successive iterations, however, smoothing becomes ever greater, finally causing edges to disappear.

Let us take a close look at the theory for TAD. Here we denote I^t_{max} as the maximum intensity value within the neighborhood of a pixel at position (i, j) at iteration t:

$$I^t_{max} = max\{I^t_{i,j}, I^t_{i+1,j}, I^t_{i-1,j}, I^t_{i,j+1}, I^t_{i,j-1}\} \tag{3.1}$$

where:

$I^t_{i,j}$ = intensity value of the center pixel,

$I_{i+1,j}$ = neighbor pixel in the east,

$I_{i-1,j}$ = neighbor pixel in the west,

$I_{i,j+1}$ = neighbor pixel in the north,

$I_{i,j-1}$ = neighbor pixel in the south.

I^t_{min} is the minimum intensity value within that neighborhood:

$$I^t_{min} = min\{I^t_{i,j}, I^t_{i+1,j}, I^t_{i-1,j}, I^t_{i,j+1}, I^t_{i,j-1}\} \tag{3.2}$$

where factors bear the same meaning as in equation (3.1).

$I^{t+1}_{i,j}$ is the new intensity value of the pixel at position (i, j) after the next iteration. We show that $I^{t+1}_{i,j}$ satisfies:

$$I^t_{min} \leq I^{t+1}_{i,j} \leq I^t_{max} \tag{3.3}$$

From equation 2.8 and the restrictions for λ and the DC function, the right inequality of equation (3.3) can be reached by:

$$\begin{aligned} I^{t+1}_{i,j} &= I^t_{i,j} + \lambda c^t_E \cdot \nabla_E I + \lambda c^t_W \cdot \nabla_W I + \lambda c^t_S \cdot \nabla_S I + \lambda c^t_N \cdot \nabla_N I \\ &= I^t_{i,j}(1 - \lambda(c^t_E + c^t_W + c^t_S + c^t_N)) \\ &\quad +\lambda(c^t_E \cdot I^t_{i,j+1} + c^t_W \cdot I^t_{i,j-1} + c^t_S \cdot I^t_{i-1,j} + c^t_N \cdot I^t_{i+1,j}) \end{aligned}$$

$$\begin{aligned} &\leq I^t_{max}(1-\lambda(c^t_E+c^t_W+c^t_S+c^t_N))+\lambda I^t_{max}(c^t_E+c^t_W+c^t_S+c^t_N) \\ &= I^t_{max} \end{aligned} \tag{3.4}$$

In a similar way, we can get the left inequality of formula (3.3).

Thus, edges are not enhanced because pixel intensity is always "moved" toward a lower intensity neighbor. This is clearly a form of averaging filter.

3.1.2 Review of Current Ways

Now let us look back on the methods for dealing with the problem of over-blurring summarized in section 2.4. The most popular way, restricting the number of iterations, is a kind of "ostrichism". Here "ostrichism" does not bear any negative meaning. It simply indicates the fact that "over-blurring does not exist because it is set not to be seen".

That TAD does not agree with AD theory has been ignored since the introduction of the AD concept. Edges cannot be preserved or enhanced if smoothing is applied to them unless no smoothing or smoothing is run backwards. Research on the effect of the DC function does not seem to recognize that a positive DC function over the whole range of gradient magnitude can never stop the diffusion process before everything has been smoothed.

You *et al.*'s work on the convergence of the CF [249] has influenced much research into seeking a DC function. You's conclusions are treated as theorems. Here for convenience equations (2.11) and (2.12), one of their conclusions about conditions

for a well-posed AD for 2D image, are rewritten as follows.

$$(\nabla I \cdot c(\nabla I))' \geq 0 \tag{3.5}$$

$$\lim_{\nabla I \to \infty} \nabla I \cdot c(\nabla I) \neq 0 \tag{3.6}$$

where:

∇I = gradient magnitude defined as equation (2.9),

$c(\cdot)$ = DC function as equation (2.3) or (2.4).

The second condition tells us that the flux should never be zero while the first one indicates that the flux increases monotonically, or at least that there is a constant flow of flux corresponding to the case of $(\nabla I \cdot c(\nabla I))' = 0$. Under this theorem, they proposed a CF (equation (2.13)) and declared this CF guarantees a well posed AD process. However, this CF does nothing to an image because it produces constant flux over the whole range of gradient magnitude except for $x \in [-T, T]$.

Finding an automatic stopping criterion sounds more plausible if there exists an optimal result where noise has been removed while edges have not been over blurred. The AD process is assumed to evolve in such a way. Whether there is an "optimal" point in the diffusion process is not shown. The essential question, does AD act in the assumed way, has never been answered but is taken for granted. Let us use the smooth-level-measure function S in equation (2.16) in section 2.3 for example. The essential part of the idea is to find the time "T_s" from the second derivative of S shown in Figure 2.9, when noise is believed to have been smoothed. The problem is that there is no guarantee that the curve of the second derivative of the smooth-level-measure function is always similar to the curve of Figure 2.9(b), and so there may not be point T_s. A simple check can be carried out by assuming a smooth-level-measure function as $S = 1 - e^{-t}$ for $t \geq 0$, a monotonically increasing function starting from $S = 0$ and going to $S = 1$. Its second derivative is $S'' = -e^{-t}$. No such T_s exists for it.

In the end, we would like to compare TAD with the median filter, which is employed widely as a pre-processing filter to remove noise and speckles from raw images for real time image processing systems [15, 59, 95, 97, 78, 151]. Median filtering is also a nonlinear image smoothing technique. It is good at removing spike noise and not blurring edges too much. It replaces the current pixel by the median of a set of ordered values consisting of the intensity values within its neighborhood. It satisfies inequality (3.3) as well. Clearly TAD acts in a manner similar to the median filter. The difference is that the TAD techniques extend the range of intensity values for replacing the central pixel, without the restriction of only using the median within that pixel's neighborhood. As with the median filter, there is no guarantee that edges are not smoothed by TAD.

3.2 Behavioral Analysis of Anisotropic Diffusion

Although nearly two hundred papers relevant to AD techniques have been published, only a few of them discuss how the CF affects the AD process [169, 171, 83, 84, 85, 249, 254]. Research into AD theory cannot be considered to be complete until an understanding of the effects of the CF is gained.

Research on the effects of CF is very important to AD theory because the AD process is dependent upon the selection of the CF that controls the "smoothing strength" over the whole image. Ideally, the CF should be able to produce more diffusion in noisy areas and areas containing trivial detail and cause no smoothing or even run diffusion backward at boundaries so as to preserve or enhance edges. Clearly, researching the effect of the CF is especially important for a mathematical understanding of the AD process and will lead to an understanding of how to choose a suitable CF.

The most recent progress in researching the effects of CF is reached by Gilboa *et al.* [83] in their research for a forward-and-backward diffusion (FAB) technique. They succeeded in showing that, as Figure 2.6 shows, a gradient magnitude remains trapped in the range between the two extrema $[-K1\ \ K1]$ if it is in this interval initially. Where this gradient magnitude will go, however, was not discussed. Also, explanations for other parts of the gradient range are not given.

3.2.1 AD Behavioral Analysis in Continuous Domain

The basis of our proposed approach is to use the derivative of flux directly to determine the diffusion coefficient (DC). Denote flux as $\phi(I_x) = c(I_x, t)I_x$ and treat it as

one function. Thus in the 1D case the AD equation (2.1) can be rewritten as:

$$\begin{aligned} \frac{\partial I}{\partial t} &= \frac{\partial}{\partial x}(c(I_x, t)I_x) \\ &= \frac{\partial}{\partial x}(\phi(I_x)) \\ &= \phi_x(I_x)I_{xx} \end{aligned} \tag{3.7}$$

where $\phi_x(\cdot)$ is the derivative of flux (DF).

In Figure 3.1, showing a typical DF curve, "S" stands for the point where the flux is small enough to be ignored. This agrees with most practical applications that have a limit to resolution. As seen in Figure 3.1, the possible values of the DF can

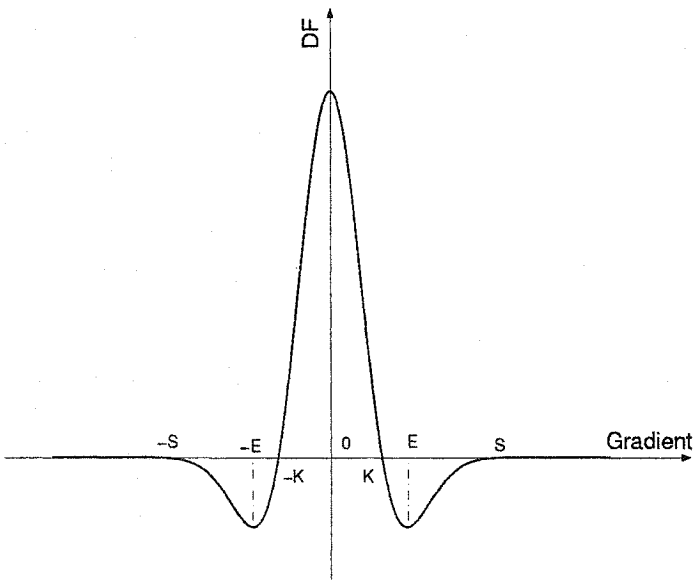


Figure 3.1: Derivative of flux

be grouped into three ranges:

$$DF \begin{cases} > 0 & \text{for} \quad |I_x| \in (-K \;\; K) \quad \text{and} \quad |I_x| \neq 0, \\ < 0 & \text{for} \quad |I_x| \in (K \;\; S) \quad \text{and} \quad |I_x| \in (-S \;\; -K), \\ = 0 & \text{otherwise} \end{cases} \tag{3.8}$$

To understand the performance of our approach, an analysis of the effects in each range is needed.

Case: $DF > 0$

Assume that $I(x,t)$ is continuous in a bounded square Γ: $\{\alpha \leq x \leq \beta, 0 \leq t \leq T\}$. For simplicity, we assume that the DF is a constant, a^2, thus the AD equation is rewritten as:

$$\frac{\partial I}{\partial t} = a^2 \frac{\partial^2 I}{\partial x^2} \tag{3.9}$$

and:

$$I(x,0) = u(x) \tag{3.10}$$

where $u(x)$ is the original image.

Denote the Fourier transforms of $I(x,t)$ and $u(x)$ with respect to x as follows.

$$\widetilde{I}(\omega,t) = \mathcal{F}[I(x,t)] \tag{3.11}$$

$$\widetilde{u}(\omega) = \mathcal{F}[u(x)] \tag{3.12}$$

Apply the Fourier transform to equation (3.9) to get:

$$\frac{\partial \widetilde{I}}{\partial t} = -a^2\omega^2 \widetilde{I} \tag{3.13}$$

which meets the initial condition:

$$\widetilde{I}(\omega,0) = \widetilde{u}(\omega) \tag{3.14}$$

The solution to equation (3.13) is:

$$\widetilde{I}(\omega,t) = \widetilde{u}(\omega) e^{-a^2\omega^2 t} \tag{3.15}$$

so the solution to equation (3.9) can be found from the inverse Fourier transform of equation (3.15):

$$I(x,t) = \mathcal{F}^{-1}[\widetilde{I}(\omega,t)]$$

$$\begin{aligned} &= \mathcal{F}^{-1}[\tilde{u}(\omega)e^{-a^2\omega^2 t}] \\ &= \mathcal{F}^{-1}[\tilde{u}(\omega)] * \mathcal{F}^{-1}[e^{-a^2\omega^2 t}] \\ &= u(x) * \mathcal{F}^{-1}[e^{-a^2\omega^2 t}] \end{aligned} \tag{3.16}$$

The second factor on the right side of equation (3.16) is:

$$\begin{aligned} \mathcal{F}^{-1}[e^{-a^2\omega^2 t}] &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-a^2\omega^2 t}e^{j\omega x}d\omega \\ &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-a^2\omega^2 t}(cos(\omega x) + jsin(\omega x))d\omega. \end{aligned} \tag{3.17}$$

Since $e^{-a^2\omega^2 t}sin\omega x$ is an odd function of ω,

$$\begin{aligned} \mathcal{F}^{-1}[e^{-a^2\omega^2 t}] &= \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-a^2\omega^2 t}cos(\omega x)d\omega \\ &= \frac{1}{2a\sqrt{\pi t}}e^{-\frac{x^2}{4a^2 t}} \end{aligned} \tag{3.18}$$

From equations (3.16) - (3.18) we have:

$$I(x,t) = \frac{1}{2a\sqrt{\pi t}}\int_{-\infty}^{\infty} u(\lambda)e^{-\frac{(x-\lambda)^2}{4a^2 t}}d\lambda \tag{3.19}$$

This solution has been derived with the assumption that $I(x,t)$ satisfies the conditions of the Fourier transform. Thus we need to prove that if it is a valid solution, it must satisfy equations (3.9) and (3.10).

Since $u(x)$ is continuous and bounded in Γ, there must be a number $M > 0$ such that:

$$|u(x)| \leq M \tag{3.20}$$

For the solution, both its first derivative with respect to x:

$$\frac{1}{2a\sqrt{\pi}}\int_{-\infty}^{\infty}\left(\frac{\lambda - x}{2a^2 t^{\frac{3}{2}}}\right)u(\lambda)e^{-\frac{(x-\lambda)^2}{4a^2 t}}d\lambda \tag{3.21}$$

and its second derivative with respect to x:

$$\frac{1}{2a\sqrt{\pi}}\int_{-\infty}^{\infty}\left(\frac{(\lambda - x)^2}{4a^4 t^{\frac{5}{2}}} - \frac{1}{2a^2 t^{\frac{3}{2}}}\right)u(\lambda)e^{-\frac{(x-\lambda)^2}{4a^2 t}}d\lambda \tag{3.22}$$

are uniformly convergent for $t > 0$. Thus the order of integral operator and derivative operator can be exchanged. Placing the solution into equation (3.9) we get:

$$\frac{\partial I}{\partial t} = \frac{1}{2a\sqrt{\pi}} \int_{-\infty}^{\infty} \left(\frac{(x-\lambda)^2}{4a^2 t^{\frac{5}{2}}} - \frac{1}{2t^{\frac{3}{2}}} \right) u(\lambda) e^{-\frac{(x-\lambda)^2}{4a^2 t}} d\lambda = a^2 \frac{\partial^2 I}{\partial x^2} \tag{3.23}$$

This shows that the solution satisfies equation (3.9).

If the solution meets the initial condition (3.10), then for an arbitrary x_0 in the original image $u(x)$, there should be $I(x,t) \to u(x_0)$ when $x \to x_0$ and $t \to 0$. Thus, we need to show that for an arbitrary $\epsilon > 0$, there should be a $\delta_1 > 0$ and a $\delta_2 > 0$, which makes:

$$|I(x,t) - u(x_0)| < \epsilon \tag{3.24}$$

when $|x - x_0| < \delta_1$ and $t < \delta_2$.

With the variable replacement:

$$\mu = \frac{\lambda - x}{2a\sqrt{t}} \tag{3.25}$$

equation (3.19) can be rewritten as:

$$I(x,t) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} u(x + 2a\sqrt{t}\mu) e^{-\mu^2} d\mu \tag{3.26}$$

and so, with the boundary condition of equation (3.10), $u(x_0)$ can be expressed as:

$$u(x_0) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} u(x_0) e^{-\mu^2} d\mu \tag{3.27}$$

Thus:

$$I(x,t) - u(x_0) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \{u(x + 2a\sqrt{t}\mu) - u(x_0)\} e^{-\mu^2} d\mu \tag{3.28}$$

For a given $\epsilon > 0$, choose a N large enough to make:

$$\int_{N}^{\infty} e^{-\mu^2} d\mu < \frac{\epsilon\sqrt{\pi}}{6M} \tag{3.29}$$

$$\int_{-\infty}^{-N} e^{-\mu^2} d\mu < \frac{\epsilon\sqrt{\pi}}{6M} \tag{3.30}$$

Because $u(x)$ is continuous, then for a fixed N there must be a $\delta_1 > 0$ and a $\delta_2 > 0$ that, when $|x - x_0| < \delta_1$ and $t < \delta_2$, makes:

$$|u(x + 2a\sqrt{t}\mu) - u(x_0)| < \frac{\epsilon}{3} \qquad (-N < \mu < N) \tag{3.31}$$

So:

$$\begin{aligned}
|I(x,t) - u(x_0)| &= |\frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \{u(x + 2a\sqrt{t}\mu) - u(x_0)\} e^{-\mu^2} d\mu| \\
&\leq \frac{1}{\sqrt{\pi}} \int_{-\infty}^{-N} \{|u(x + 2a\sqrt{t}\mu)| + |u(x_0)|\} e^{-\mu^2} d\mu \\
&\quad + \frac{1}{\sqrt{\pi}} \int_{-N}^{N} \{|u(x + 2a\sqrt{t}\mu) - u(x_0)|\} e^{-\mu^2} d\mu \\
&\quad + \frac{1}{\sqrt{\pi}} \int_{N}^{\infty} \{|u(x + 2a\sqrt{t}\mu)| + |u(x_0)|\} e^{-\mu^2} d\mu \\
&\leq 2M \frac{\epsilon}{6M} + \frac{\epsilon}{3\sqrt{\pi}} \int_{-N}^{N} e^{-\mu^2} d\mu + 2M \frac{\epsilon}{6M} \\
&\leq \frac{\epsilon}{3} + \frac{\epsilon}{3} + \frac{\epsilon}{3} = \epsilon
\end{aligned} \tag{3.32}$$

Equations (3.23) - (3.32) show that equation (3.19) is a valid solution for equation (3.9) with the initial condition of equation (3.10).

Now, we can consider the behavior characteristics of equation (3.19). From equations (3.19), (3.20) and (3.25) we get:

$$\begin{aligned}
|I(x,t)| &\leq \frac{1}{2a\sqrt{\pi t}} \int_{-\infty}^{\infty} M e^{-\frac{(x-\lambda)^2}{4a^2 t}} d\lambda \\
&\leq \frac{M}{\sqrt{\pi}} \int_{-\infty}^{\infty} e^{-\mu^2} d\mu \\
&\leq M
\end{aligned} \tag{3.33}$$

This shows that the solution is convergent and bounded for a positive DF.

Finally, let us evaluate the behavior characteristics of the gradient function in equation (3.19). From equations (3.19) to (3.21) and (3.25), we have:

$$\lim_{t \to \infty} \left| \frac{\partial I}{\partial x} \right| = \lim_{t \to \infty} \left| \frac{1}{2a\sqrt{\pi}} \int_{-\infty}^{\infty} \left(\frac{\lambda - x}{2a^2 t^{\frac{3}{2}}} \right) u(\lambda) e^{-\frac{(x-\lambda)^2}{4a^2 t}} d\lambda \right|$$

$$= \lim_{t\to\infty} \frac{1}{a\sqrt{\pi t}} \int_{-\infty}^{\infty} |u(x + 2a\sqrt{t}\mu)\mu e^{-\mu^2}| d\mu$$
$$= 0 \tag{3.34}$$

This shows that, with a positive diffusion coefficient, smoothing is carried out continuously with time until no non-zero gradients exist.

Case: $DF < 0$

For a negative DF, equation (3.9) can be rewritten as:

$$\frac{\partial I}{\partial t} = -a^2 \frac{\partial^2 I}{\partial x^2} \tag{3.35}$$

Following the same procedure as for the case of $DF > 0$, we get:

$$\frac{\partial \tilde{I}}{\partial t} = a^2\omega^2 \tilde{I} \tag{3.36}$$

and

$$\tilde{I}(\omega, t) = \tilde{u}(\omega) e^{a^2\omega^2 t} \tag{3.37}$$

Clearly equation (3.37) is divergent with t and therefore $I(x, t)$ must be as well. This shows that AD is an unstable diffusion process for $DF < 0$.

Case: $DF = 0$

When the diffusion coefficient is zero, equation (3.9) can be rewritten as:

$$\frac{\partial I}{\partial t} = 0 \tag{3.38}$$

and its solution is:

$$I = u_1(x) \tag{3.39}$$

where $u_1(x)$ is the image when the DF is zero. In this case, the AD filter does nothing.

Based on these results, we get:

Conclusion: There are three kinds of diffusion process: mathematically well-posed, mathematically ill-posed and no diffusion. They correspond to flux increasing with gradient magnitude, flux decreasing with gradient magnitude, and flux of zero respectively.

Now, we explore the change of gradient magnitudes during AD. A typical DF shown in Figure 3.1 is employed for the discussion. We examine the interval $[0, \infty)$ because the flux function has odd symmetry. S stands for "Stop", indicating the region where the flux is small enough to assume that flux is zero in $[S, \infty)$. Thus we explore the interval $[0\ S]$.

Differentiating both sides of equation (3.7) with respect to x yields:

$$\frac{\partial I_t}{\partial x} = \phi_{xx} I_{xx}^2 + \phi_x I_{xxx}. \tag{3.40}$$

With the assumption that $I(x,t)$ is continuous and bounded in $\Gamma : \{x \in (-\infty\ \infty),\ t > 0\}$, the order of differentiation can be exchanged. To evaluate the behavior of the gradient with respect to time, equation (3.40) is rewritten as:

$$\begin{aligned} I_{xt} &= \phi_{xx} I_{xx}^2 + \phi_x I_{xxx} \\ &= \frac{\partial I_x}{\partial t}. \end{aligned} \tag{3.41}$$

As Figure 3.1 shows, the derivative of flux is:

$$\phi_x \begin{cases} > 0 & \text{for} \quad x \in (0,\ K), \\ < 0 & \text{for} \quad x \in (K,\ S), \\ = 0 & \text{otherwise} \end{cases} \tag{3.42}$$

and the second derivative of flux is:

$$\phi_{xx} \begin{cases} < 0 & \text{for} \quad x \in (0,\ E), \\ > 0 & \text{for} \quad x \in (E,\ S), \\ = 0 & \text{otherwise} \end{cases} \tag{3.43}$$

So:

$$\phi_{xx} I_{xx}^2 \begin{cases} < 0 & \text{for} \quad x \in (0, \ E), \\ > 0 & \text{for} \quad x \in (E, \ S), \\ = 0 & \text{otherwise} \end{cases} \tag{3.44}$$

Thus, the first term on the right side of equation (3.41) affects the AD process in two different ways in the range $(0, \ S)$. Smoothing is carried out by moving the gradient toward zero in the first interval $(0, \ E)$ and edge enhancement is implemented in interval $(E, \ S)$.

Now, we consider the effect of the second term on the right side of equation (3.41) by itself to simplify our discussion. Ignoring the first term, we get:

$$\frac{\partial I_x}{\partial t} = \phi_x \frac{\partial^2 I_x}{\partial x^2} \tag{3.45}$$

This equation has the same form as equation (3.9). Thus, I_x is bounded and convergent in the range $(0, \ K)$ but divergent in $(K, \ S)$. Note that the same conclusion could be reached for the interval $[-S, \ 0]$. Since $I_x \leq |I_x(x, 0)|$ and equation (3.44) produces nothing but smoothing in $(0, \ K]$, we come to:

Conclusion: Smoothing is carried out in the interval $(-K, \ K)$. *Any non-zero gradient magnitude within this interval at the beginning of AD will eventually reach a value of zero.*

Gradient behavior in $(K, \ S)$ is more complex, because E is in the interval $(K, \ S)$. Since the second term on the right hand side of equation (3.41) produces a divergent solution for I_{xt}, but equation (3.44) implements both smoothing and enhancement in this interval, there is no guarantee on the direction of changes to these gradient magnitudes. Balance between the effects produced by equations (3.44) and (3.45) may move a gradient toward zero as noise, or may enhance it. So we come to:

Conclusion: In the intervals $(K,\ S)$ *and* $(-S,\ -K)$*, the diffusion process is unstable. Any gradient magnitude within these intervals at the start of AD may be smoothed to zero as noise or enhanced as an edge.*

This shows that the interval $(K,\ E)$ generates undesirable uncertainties in the result images. Clearly this interval should be restricted if it is exploited for edge enhancement. Observed "staircasing" effects [47, 237], the creation of edges in smooth intensity "ramps", takes place in $(K,\ S)$. With the assumption that the flux is zero in $[S,\ \infty)$, the image remains unchanged after the diffusion process finishes in $(-S,\ S)$. Thus, we come to:

Conclusion: The diffusion process stops modifying the image when the only gradients within the interval $(-S,\ S)$ *have zero magnitude.*

Clearly, the threshold "S" is very important for applications of AD because it defines the value of edge gradient magnitudes to be preserved. Edges, however, are not preserved if S is made too large. This is because the AD process is circumscribed by the "Maximum Principle" [155] which states that all of the maxima belong to the initial conditions. S should be selected to be the minimum value for semantically meaningful edges. Without a correct setting for S, attempts to create a general-purpose criterion for AD will fail. S should be called the "edge threshold" T_e, while "K", the traditional threshold for edges, should be called the "noise threshold" T_n. T_n is important for applications in which a certain range of gradient magnitudes should be treated as noise.

A new interpretation of AD behavior in the whole range of gradient magnitudes is developed in the previous with the aid of the Fourier transform. We showed that AD is composed of three different processes: a mathematically well-posed smoothing

process, an ill-posed unstable process that could generate edge enhancement, and a process that does not change the image.

The conditions for generating the "staircasing" effects are discussed. We showed that the prerequisite for creating a stopping criterion for the challenging problem of oversmoothing does not exist, because there is no gurantee that edges do not suffer from being smoothed with the first iteration. Negative CFs should be used carefully for they can produce unpredictable results.

A threshold that results in zero flux is found to be very important to the quality of resultant images and is proposed for the first time. The edge threshold defined in TAD techniques is renamed the "noise threshold", because it actually denotes the range in which smoothing is carried out.

3.2.2 A Criterion For The Conduction Function

Finally, we propose a criterion for selecting the CF to implement AD:
Any convenient function can be used as the CF if it produces the desired flux, that increases monotonically in the range [0, T_n] and decreases monotonically in the interval (T_n, T_e]. T_e and T_n are the thresholds for edges and noise respectively.

3.3 A Discrete Form of Anisotropic Diffusion

The P-M AD is paradoxical. While their discrete form does not agree with its continuous counterpart, nonetheless it has been widely used to the present. The problem is that the discrete form performs smoothing on edges not predicted by the continuous AD theory.

This section focuses on connecting the continuous and discrete forms of AD.

3.3.1 Continuous Version of Anisotropic Diffusion

We first address the P-M discrete version of AD (DAD). For simplicity, we consider the 1D case along the x axis. Assuming that $c_L = c_R$, from equation (2.5) we get the P-M 1-D discrete equation:

$$\begin{aligned} I_i^{t+1} &= I_i^t + \lambda c_L (I_{i+1}^t - 2I_i^t + I_{i-1}^t) \\ &= I_i^t + \dot{\lambda}(I_{i+1}^t - 2I_i^t + I_{i-1}^t) \\ &= I_i^t + \dot{\lambda}\Delta I_i^t \end{aligned} \tag{3.46}$$

where: $\dot{\lambda} = \lambda c_L$ for later convenience.

Equation (3.46) is significantly different from equation (2.1). Equation (3.46) agrees with the first term on the right hand side of equation (2.1), but does not contain the second term.

As discussed in Section 3.2.1, the derivative of flux should be used to control the diffusion process. Assume an image of a step edge at arbitrary orientation. Without loss of generality, we assume a neighborhood coordinate frame such that the edge is parallel to the y axis so that we may treat this as a 1D problem. Denoting flux by

$\phi(I_x) = c(I_x, t)I_x$, equation (2.1) can be rewritten as:

$$\begin{aligned} \frac{\partial I}{\partial t} &= \frac{\partial}{\partial x}(c(I_x, t)I_x) \\ &= \frac{\partial}{\partial x}(\phi(I_x)) \\ &= \phi_x(I_x)I_{xx} \end{aligned} \tag{3.47}$$

The DF is considered to be one function and is used for the DC. Now let us consider the DF's behavior.

$$\begin{aligned} \phi_x &= \frac{\partial}{\partial x}(x \cdot c(x)) \\ &= c(x) + x \cdot c_x(x) \end{aligned} \tag{3.48}$$

Here x, instead of I_x, is used for convenience. The first summand $c(x)$ is a non-negative, monotonously decreasing function starting at $c(0) = 1$, with $\lim_{x \to \infty} c(x) = 0$. Assume both $c(x)$ and its derivative $c_x(x)$ are continuous functions. For the second term, the boundary values are:

$$x \cdot c_x(x)|_{x=0} = 0 \tag{3.49}$$

and

$$\begin{aligned} \lim_{x \to \infty} x \cdot c_x(x) &= \lim_{x \to \infty} \frac{c(x)}{lnx} \\ &= 0 \end{aligned} \tag{3.50}$$

$c_x(x)$ is negative. With the assumption of continuity, there is a number $M > 0$ such that:

$$|x \cdot c_x(x)| \leq M \tag{3.51}$$

with:

$$M = |min\{x \cdot c_x(x)\}| \tag{3.52}$$

M is dependent upon $c(x)$. For example, M equals $|-\frac{2}{e}|$ and $|-\frac{1}{2}|$ for the P-M CFs of equations (2.3) and (2.4) respectively.

With equations (3.49) to (3.51), we find that the second term is negative except at its boundaries. The DF is a function starting at $\phi_x(0) = 1$ and ending at $\lim_{x \to \infty} \phi_x(x) = 0$, typically in the form as shown in Figure 3.1. It implements the desired operations of noise removal and edge preservation and enhancement. Negative values of DF are not employed due to their side-effects.

3.3.2 AD Behavioral Analysis in Discrete Domain

Now let us look for a discrete form of AD that corresponds to equation (3.47). Since a negative DC creates an unstable diffusion process, we concentrate on the numerical formulation of equation (3.47) for positive derivative of flux. For simplicity, we consider a 1-D signal and assume that the DF is a constant a^2. With these assumptions, equation (3.47) can be rewritten as:

$$\frac{\partial I(x,t)}{\partial t} = a^2 \frac{\partial^2 I(x,t)}{\partial x^2} \tag{3.53}$$

with

$$I(x,t)|_{t=0} = u(x) \tag{3.54}$$

where:

$u(\cdot)$ = the original image.

To form the difference equation of a continuous image in $\mathcal{R} : \{x \in [\ 0\ \ l\ ], t \in [\ 0\ \ T\ ]\}$, divide the continuous image evenly in space into N parts with step $\Delta x = \frac{l}{N}$, and numerically approximate the continuous diffusion process by time step Δt. In this way we create a discrete representation of the image. The value of $I(x,t)$ at position i and time k is represented in the discrete domain as:

$$I_i^k = I(x,t)|_{x=i\Delta x, t=k\Delta t} \tag{3.55}$$

where:

i = pixel position,

k = time, based on unit Δt.

and the values of its partial differentials are denoted as:

$$\begin{cases} (\frac{\partial I}{\partial t})_i^k = \frac{\partial I(x,t)}{\partial t}\Big|_{x=i\Delta x, t=k\Delta t} \\ (\frac{\partial^2 I}{\partial x^2})_i^k = \frac{\partial^2 I(x,t)}{\partial x^2}\Big|_{x=i\Delta x, t=k\Delta t} \end{cases} \tag{3.56}$$

Assume $I(x,t)$ is sufficiently smooth in $\mathcal{R}$. Using Taylor's series, a pixel at position i and step k is:

$$\frac{I_i^{k+1} - I_i^k}{\Delta t} - \left(\frac{\partial I}{\partial t}\right)_i^k = \frac{\Delta t}{2}\frac{\partial^2 I(x_i, \hat{t})}{\partial t^2} \qquad t_k \leq \hat{t} \leq t_{k+1} \tag{3.57}$$

$$\frac{I_{i+1}^k - 2I_i^k + I_{i-1}^k}{(\Delta x)^2} - \left(\frac{\partial^2 I}{\partial x^2}\right)_i^k = \frac{(\Delta x)^2}{12}\frac{\partial^4 I(\hat{x}, t_k)}{\partial x^4} \qquad x_i - \Delta x \leq \hat{x} \leq x_i + \Delta x \tag{3.58}$$

Placing the partial differentials at $(i\Delta x, k\Delta t)$ from equations (3.57) and (3.58) into equation (3.53), we get:

$$\frac{I_i^{k+1} - I_i^k}{\Delta t} - a^2\frac{I_{i+1}^k - 2I_i^k + I_{i-1}^k}{(\Delta x)^2} = \frac{\Delta t}{2}\frac{\partial^2 I(x_i, \hat{t})}{\partial t^2} - a^2\frac{(\Delta x)^2}{12}\frac{\partial^4 I(\hat{x}, t_k)}{\partial x^4} \tag{3.59}$$

which meets the original conditions:

$$I_i^0 = u(i\Delta x) \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.60}$$

Assume that the right hand side of equation (3.59) is zero. With this assumption, AD equation (3.53) is approximated by:

$$\frac{\dot{I}_i^{k+1} - \dot{I}_i^k}{\Delta t} - a^2\frac{\dot{I}_{i+1}^k - 2\dot{I}_i^k + \dot{I}_{i-1}^k}{(\Delta x)^2} = 0 \tag{3.61}$$

with

$$\dot{I}_i^0 = u(i\Delta x) \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.62}$$

Equation (3.61) is based on the assumption that the right hand side of equation (3.59) is zero. Clearly the solutions of equations (3.59) and (3.61) are different. We use symbol $\dot{I}(x,t)$ to distinguish between them. When equation (3.61) is used to approximate AD equation (3.53), the error due to discretization, also called the cut-off error, is:

$$\epsilon_i^k = \frac{\Delta t}{2}\frac{\partial^2 I(x_i, \widehat{t})}{\partial t^2} - a^2\frac{(\Delta x)^2}{12}\frac{\partial^4 I(\widehat{x}, t_k)}{\partial x^4} \tag{3.63}$$

With:

$$\lambda = a^2\frac{\Delta t}{(\Delta x)^2} \tag{3.64}$$

the numerical approximation of AD is:

$$\dot{I}_i^{k+1} = \lambda\dot{I}_{i+1}^k + (1-2\lambda)\dot{I}_i^k + \lambda\dot{I}_{i-1}^k \tag{3.65}$$

with

$$\dot{I}_i^0 = u(i\Delta x) \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.66}$$

Is this a suitable approximation to the continuous equation (3.53)? First, let us consider the convergence of this discrete approximation algorithm. Its solution should converge to the solution of equation (3.53) as $\Delta x \to 0$ and $\Delta t \to 0$.

Theorem: Assume that a continuous solution of equation (3.53) exists in $R : \{x \in [\ 0\ l\ ],\ t \in [0\ T]\}$, *and that continuous partial derivatives* $\frac{\partial^2 I}{\partial t^2}, \frac{\partial^4 I}{\partial x^4}$ *exist, then the solution approximated by equation (3.65) converges to that of equation (3.53) if*

$$\lambda \leq \frac{1}{2}. \tag{3.67}$$

Denote by I_i^k and $\dot{I}_i^k$ the solutions of the CAD and DAD at point $(i\Delta x, k\Delta t)$ respectively. The difference of these two solutions due to discretization is:

$$V_i^k = I_i^k - \dot{I}_i^k \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.68}$$

From equations (3.59) - (3.66),

$$\frac{V_i^{k+1} - V_i^k}{\Delta t} - a^2 \frac{V_{i+1}^k - 2V_i^k + V_{i-1}^k}{(\Delta x)^2} = \epsilon_i^k \tag{3.69}$$

which meets

$$V_i^0 = 0 \qquad \text{for} \quad i = 0, 1, 2, \cdots, N-1. \tag{3.70}$$

Introduce two constants for our discussion:

$$M_1 = max \left| \frac{1}{2} \frac{\partial^2 I(x,t)}{\partial t^2} \right| \tag{3.71}$$

$$M_2 = max \left| \frac{a^2}{12} \frac{\partial^4 I(x,t)}{\partial x^4} \right| \tag{3.72}$$

and let:

$$M = M_1 \Delta t + M_2 (\Delta x)^2, \tag{3.73}$$

It is clear that:

$$|\epsilon_i^k| \leq M \qquad \text{for} \quad i = 0, 1, 2, \cdots, N-1. \tag{3.74}$$

Rewriting equation (3.69) as:

$$V_i^{k+1} = \lambda V_{i+1}^k + (1 - 2\lambda) V_i^k + \lambda V_{i-1}^k + \Delta t \epsilon_i^k \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.75}$$

and denote its maximum value at time $k\Delta t$ as:

$$V^k = max|V_i^k| \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.76}$$

From equations (3.75) - (3.76) and the restriction for λ, we get:

$$\begin{aligned} |V_i^{k+1}| &\leq |\lambda V_{i+1}^k + (1-2\lambda)V_i^k + \lambda V_{i-1}^k + M\Delta t| \\ &\leq |\lambda V_{i+1}^k| + |(1-2\lambda)V_i^k| + |\lambda V_{i-1}^k| + |M\Delta t| \\ &\leq |\lambda V^k| + |(1-2\lambda)V^k| + |\lambda V^k| + |M\Delta t| \\ &\leq V^k + M\Delta t \qquad (i = 0, 1, 2, \cdots, N-1) \end{aligned} \tag{3.77}$$

Thus we get:

$$V^{k+1} \leq V^k + M\Delta t \tag{3.78}$$

This is a recursion equation and its solution is given by:

$$\begin{aligned} V^{k+1} &\leq V^k + M\Delta t \\ &\leq V^{k-1} + 2M\Delta t \\ &\leq \cdots \\ &\leq V^0 + (k+1)M\Delta t \end{aligned} \tag{3.79}$$

Since $V_i = 0$ and

$$(k+1)\Delta t \leq T \tag{3.80}$$

we can say:

$$V^{k+1} \leq MT = \left(M_1 + \frac{M_2 a^2}{\lambda}\right) T\Delta t. \tag{3.81}$$

Let:

$$L = \left(M_1 + \frac{M_2 a^2}{\lambda}\right) T \tag{3.82}$$

and note that it is independent of k, thus:

$$V^{k+1} \leq L\Delta t \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.83}$$

This shows that the difference between the solution of the DAD and that of the CAD converges to zero as $\Delta t \to 0$. Thus, the solution of discrete AD converges to

the solution of the continuous AD.

Because there is a cut-off error at every step that also affects the solution of the next step, this solution is actually an approximation of the solution for discrete AD. Though the deviation between them may be very small at one step, its accumulation could result in a poor solution or even make the calculation process uncontrollable. Now let us address the stability and consistency of the discrete version of AD.

The Lax equivalence theorem [58] provides a relationship between the concepts of convergence, stability and consistency. It states that a scheme converges if and only if it is consistent and stable. The discussion above shows that the proposed discrete version of AD is convergent, and so its solution converges to the solution of CAD stably and consistently.

Now, we can rewrite the discrete version to correspond to its continuous counterpart, equation (3.53), as follows:

$$I_i^{k+1} = I_i^k + \lambda(I_{i+1}^k - 2I_i^k + I_{i-1}^k) \tag{3.84}$$

with

$$I_i^0 = u(i\Delta x) \qquad (i = 0, 1, 2, \cdots, N-1) \tag{3.85}$$

This shows, although for a simplified case, that the DF should be used to form the DC for control of the diffusion process.

3.3.3 Discussion

P-M DAD does not use the derivative of flux. Still consider the 1D case. Equations (3.84) and (3.46) share the same form but with a significant difference. In equation (3.84), λ is a function of the DF, Δx and Δt. When equation (3.84) is used to approximate its continuous counterpart, it gives convergent, stable and consistent results if $\lambda \leq \frac{1}{2}$. In equation (3.46), $\dot{\lambda}$ is a product of a constant and the CF itself. With this positive CF controlling the smoothing strength over the whole gradient range, it is difficult, if not impossible, for edges to survive the diffusion process.

The advantage of P-M DAD should be emphasized even though it does not agree with the continuous case. First, when it was proposed, it was an improvement to smoothing techniques in comparison with linear filters with the introduction of spatially variable smoothing strength. It implements selective, nonlinear smoothing, with different smoothing strength for different gradient magnitudes. We would rather call it a semi-AD because smoothing is also carried out on large gradient magnitudes that should be treated as edges, though the smoothing strength for large gradient magnitude is much smaller than that applied on the noise area if a suitable threshold is selected. Second, it exhibits good stability due to the positive diffusion coefficient, although with a sacrifice in its performance. It is guaranteed by the "Maximum Principle" [169, 155] that using a positive diffusion coefficient produces no new maxima or minima in the diffusion process. Finally, good results may result if the iterations are stopped in time.

3.4 Idempotent Anisotropic Diffusion

3.4.1 Idempotent Anisotropic Diffusion Algorithm

Based on the discussion above, in the 1-D case the AD equation is:

$$\begin{aligned} I_t &= \frac{\partial}{\partial x}(c(I_x,t)I_x) \\ &= \frac{\partial}{\partial x}\phi(I_x) \\ &= \phi_x(I_x)\cdot I_{xx} \end{aligned} \tag{3.86}$$

Its corresponding spatiotemporally discrete version is proposed as:

$$\begin{aligned} I_i^{t+1} &= I_i^t + \lambda\phi_x(I_x^t)\cdot I_{xx}^t \\ &= I_i^t + \lambda\phi_x(I_x^t)\{(I_{i+1}^t - I_i^t) - (I_i^t - I_{i-1}^t)\} \\ &= I_i^t + \lambda\phi_x(I_R^t)(I_{i+1}^t - I_i^t) + \lambda\phi_x(I_L^t)(I_{i-1}^t - I_i^t) \end{aligned} \tag{3.87}$$

where:

i	=	the pixel position,
λ	=	a scalar $\in R^+$ for controlling the rate of diffusion,
t	=	discrete time step,
R,L	=	directions, to the right and left respectively.

For 2D digital images, the AD equation can be interpreted as follows:

$$\begin{aligned} I_t &= div(c(I_x,I_y,t)\nabla I) \\ &= div(c(I_x,I_y,t)(I_x\vec{i} + I_y\vec{j})) \\ &= \nabla\cdot c(I_x,I_y,t)(I_x\vec{i} + I_y\vec{j}) \\ &= (\frac{\partial}{\partial x}\vec{i} + \frac{\partial}{\partial y}\vec{j})\cdot(c(I_x,I_y,t)I_x\vec{i} + c(I_x,I_y,t)I_y\vec{j}) \\ &= (\frac{\partial}{\partial x}\vec{i} + \frac{\partial}{\partial y}\vec{j})\cdot(\phi(I_x,I_y,t)\vec{i} + \phi(I_x,I_y,t)\vec{j}) \\ &= \phi'(I_x)I_{xx} + \phi'(I_y)I_{yy} \end{aligned} \tag{3.88}$$

where:

$\vec{i}, \vec{j}$ = unit vectors in the directions of x and y respectively.

A 4-nearest-neighbor discrete version is proposed for it as:

$$I_{i,j}^{t+1} = I_{i,j}^t + \lambda\{\phi'(\nabla_E I_{i,j}^t)\nabla_E I_{i,j}^t + \phi'(\nabla_W I_{i,j}^t)\nabla_W I_{i,j}^t + \phi'(\nabla_N I_{i,j}^t)\nabla_N I_{i,j}^t + \phi'(\nabla_S I_{i,j}^t)\nabla_S I_{i,j}^t\} \quad (3.89)$$

where all subscripts, superscripts and parameters bear the same meaning as in equation (2.8).

To verify this discrete formula, assume that $\phi'(\nabla_E I_{i,j}^t) = \phi'(\nabla_W I_{i,j}^t) = \phi'_{EW}(\nabla I_{i,j}^t)$ and $\phi'(\nabla_N I_{i,j}^t) = \phi'(\nabla_S I_{i,j}^t) = \phi'_{NS}(\nabla I_{i,j}^t)$ for simplicity. Then:

$$\begin{aligned} \phi'(\nabla_E I_{i,j}^t)\nabla_E I_{i,j}^t + \phi'(\nabla_W I_{i,j}^t)\nabla_W I_{i,j}^t &= \phi'_{EW}(\nabla I_{i,j}^t)(\nabla_E I_{i,j}^t + \nabla_W I_{i,j}^t) \\ &= \phi'_{EW}(\nabla I_{i,j}^t)(I_{i+1,j}^t - I_{i,j}^t + I_{i-1,j}^t - I_{i,j}^t) \\ &= \phi'_{EW}(\nabla I_{i,j}^t)(I_{i+1,j}^t - I_{i,j}^t - (I_{i,j}^t - I_{i-1,j}^t)) \\ &= \phi'_{EW}(\nabla I_{i,j}^t) I_{xx}^t. \end{aligned} \quad (3.90)$$

and

$$\begin{aligned} \phi'(\nabla_N I_{i,j}^t)\nabla_N I_{i,j}^t + \phi'(\nabla_S I_{i,j}^t)\nabla_S I_{i,j}^t &= \phi'_{NS}(\nabla I_{i,j}^t)(\nabla_N I_{i,j}^t + \nabla_S I_{i,j}^t) \\ &= \phi'_{NS}(\nabla I_{i,j}^t)(I_{i,j+1}^t - I_{i,j}^t + I_{i,j-1}^t - I_{i,j}^t) \\ &= \phi'_{NS}(\nabla I_{i,j}^t)(I_{i,j+1}^t - I_{i,j}^t - (I_{i,j}^t - I_{i,j-1}^t)) \\ &= \phi'_{NS}(\nabla I_{i,j}^t) I_{yy}^t. \end{aligned} \quad (3.91)$$

So equation (3.89) is in a form as follows that agrees with equation (3.88).

$$I_{i,j}^{t+1} = I_{i,j}^t + \lambda\{\phi'_{EW}(\nabla I_{i,j}^t) I_{xx}^t + \phi'_{NS}(\nabla I_{i,j}^t) I_{yy}^t\} \quad (3.92)$$

3.4.2 Assessment of The IAD Scheme

The determination of the DC function is very important to AD techniques. The proposed IAD technique is evaluated with respect to the criteria described in Section 2.1.5 based on the effects of DC function.

1. Causality, the first criterion for AD filters, requires that no fictitious edge is produced when the resolution reduces. This is met by the positive part of the flux's derivative, except for the "staircasing" effect for which more research is needed [54, 237]. This is because the AD process is circumscribed by the "Maximum Principle" [155] which states that all of the maxima must only belong to the initial conditions. There is a problem, however, for the negative part of the flux's derivative to meet the criterion of causality. It does not obey the "Maximum Principle" and generates an unstable diffusion process by running backwards smoothing. Obviously, this part should be removed or restricted.

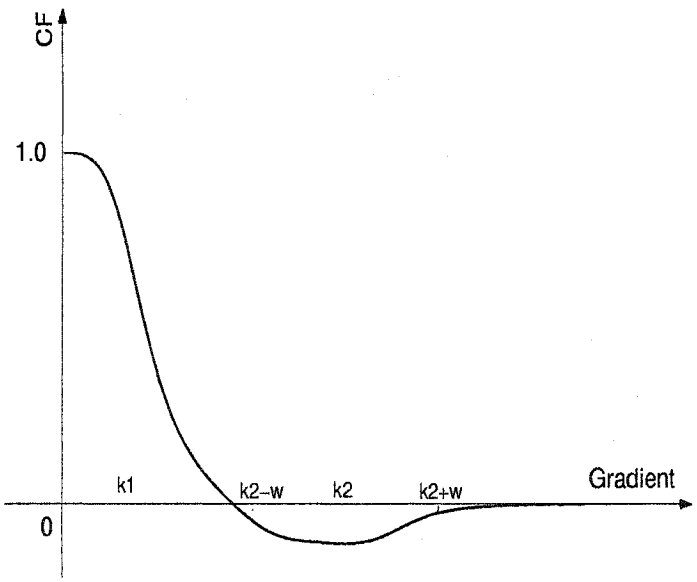


Figure 3.2: CF curve of the "Gilboa" FAB filter

Backwards diffusion has been studied by some researchers [76, 83, 84, 85, 173, 174, 201, 203, 204]. Gilboa *et al.* [83, 84, 85] proposed a CF as in Figure 3.2 that runs forward-and-backward diffusion (FAB) and produces a flux similar in form to Figure 2.6:

$$c(x) = \frac{1}{1+(\frac{x}{K_1})^n} - \frac{\alpha}{1+(\frac{x-K_2}{w})^{2m}} \tag{3.93}$$

where:

K_1 = a parameter controls the forward "force",

K_2 = a parameter controls the backward "force",

α = for a balance between the forward and backward "force",

w = width of the negative CF,

n, m = parameters control the shape of forward and backward factors.

They recommend a formula to calculate these parameters [83]:

$$\begin{cases} [K_1\ K_2\ w] = [2\ 4\ 1] * MAG \\ \alpha \leq \frac{K_1}{2(K_2+w)} \qquad \text{for any } 0 < w < K_2 - K_1 \\ n = 4,\ \ m = 2. \end{cases} \tag{3.94}$$

where:

MAG = the mean absolute gradient.

In their experimental results, this rule was not followed and the determination of these parameters was image dependent.

Another problem is that significant distortion occurs due to the wide region of negative values with their CF. Adding a fidelity term is helpful for mitigating the distortion [83, 231]:

$$I_t = \frac{\partial}{\partial x}\phi(I_x) - \lambda(I - I_0) \tag{3.95}$$

where:

λ = the fidelity term.

The fidelity term restricts the difference between the result and the original image. A small λ gives more freedom for the result while a large λ requires more similarity

between the result and the original image.

Thus, the benefit produced by the negative part of the flux's derivative should be considered carefully with respect to both its advantages and disadvantages. The behavior analysis in Section 3.2.1 shows that gradient magnitudes within the negative range keep moving intensities "uphill" until they go out of this range. The problem is that sometimes it is hard for a gradient magnitude to go out of the range due to the interactions with its neighbors. This may lead to undesired results even with certain restrictions added. Based on the above discussion, we recommend considerable caution, restricting the negative part of the flux's derivative to a small range, and only if there is a clear benefit. For example, the range of negative value is set within 10% of the edge threshold.

2. Improvement of the second criterion, immediate localization, is shown clearly by comparing Figure 3.1 with Figure 2.1. The TAD equation treats the range [0, K] as noise and applies smoothing. The IAD method divides the range into two parts with a noise threshold T_e. Gradient magnitudes in the part less than T_e are smoothed as noise, while the gradient magnitudes in the other part may be smoothed or enhanced. Figure 3.1 shows that diffusion diminishes as gradient magnitudes approach the threshold, and stop exactly at the threshold. As the gradient magnitude increases in the range (K, S), the IAD filter runs diffusion "backwards", enhancing edges. With our recommendation of restricting the negative part to a small range, large gradient magnitudes are kept throughout the diffusion process as edges. On the other hand, as Figure 2.1 shows, the TAD equations continue to cause smoothing in the range (K, ∞). This causes edges to be blurred, and finally to disappear.

3. With respect to the final criterion, piecewise smoothing, the IAD equations

implement the desired selective diffusion. IAD can produce a mathematically well-posed, a mathematically ill-posed or an idempotent diffusion process. These three results correspond to when the generated flux is increasing, decreasing or constant respectively. In the first case, smoothing is carried out and gradient magnitudes are consistently reduced to zero. The second case is a divergent diffusion process that eventually disappears with sufficient iterations. As for the third case, semantically meaningful edges are kept by the AD filter throughout the diffusion process. By controlling the unstable diffusion process, IAD filter implements piecewise smoothing and realizes the goal of noise removal and edge preservation/enhancement.

The TAD equations do not stop smoothing over the whole image. They implement the strongest smoothing within a close range of the threshold defined for edges. Only edges with gradients much larger than the threshold may remain nearly "untouched" at the beginning. Here we use the word "nearly untouched" because they are actually smoothed, although very slightly. However, even such a state is not able to last for a long time because no edge can withstand continual smoothing.

3.5 Experiments

Experiments have been carried out on both 1D signals and 2D images, with a number of CFs employed for different purposes. The CFs were scaled so that the fluxes generated by them are similar. For example, the ranges of the monotonically increasing fluxes produced by the two P-M CFs are different if the same threshold K_e is used.

3.5.1 Source of Test Images

Both real images and artificial images were used in our experiments. The artificial image is simple but allows the experiment be controlled easily, while real images are employed for the following reasons:

- Diverse, spatially varying conditions in real images cannot be designed. That a technique works with an artificial signal does not means it is able to work the complexity of real images.

- Our final goal is to apply the AD technique to real images, and so real images should be used for testing and evaluation.

Artificial Image

Art: A synthetic image containing curves, lines, ramps and corners.

Real image

Lab: A picture of 256x240 pixels taken of the former CVIP Lab.

Lab1d: An 1D image consisting of a horizontal line in the "Lab" image.

Lena: A standard test image (256x256 pixels) in the field of image processing.

3.5.2 Simulation Functions

Experiments are designed to study AD behaviors with different CFs for different types of fluxes and compare the performance of idempotent anisotropic diffusion (IAD) filters and traditional anisotropic diffusion (TAD) filters. Floating-point simulation of a number of functions was carried out.

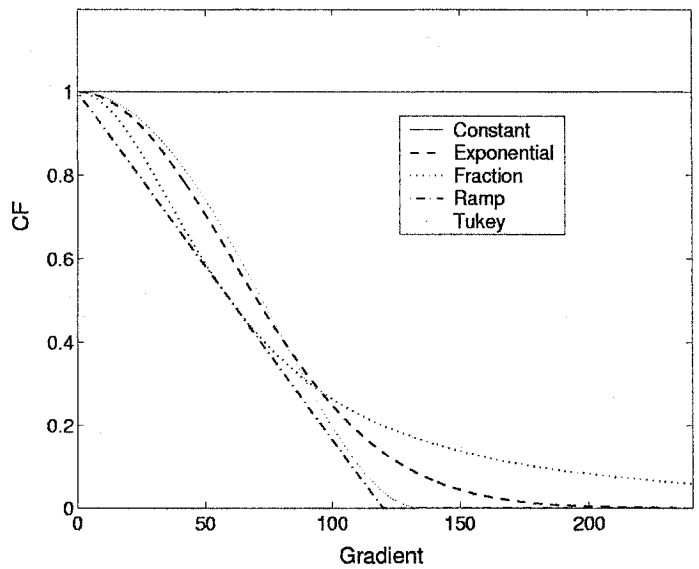


Figure 3.3: Conduction functions in this experiment

We explored the two P-M CFs (equations (2.3) and (2.4)), Tukey's biweight function (2.14), FAB diffusion (equations (3.93) and (3.94)) proposed by Gilboa *et al.* [83], their scheme of adding a fidelity term to restrict the distortion (equation (3.95)), and two simple "straight line" CFs. The definitions of the "straight line" CFs in the interval $[0\ \infty)$ are as follows:

$$c(x) = \begin{cases} 1 \quad \text{for} & 0 \leq x \leq K_e \\ 0 \quad \text{otherwise} & \end{cases} \tag{3.96}$$

and

$$c(x) = \begin{cases} 1 - \frac{x}{K_e} & \text{for} \quad 0 \leq x \leq K_e \\ 0 & \text{otherwise} \end{cases} \tag{3.97}$$

For convenience the CFs used, given by equations (2.3), (2.4), (2.14), (3.93), (3.95), (3.96) and (3.97), are called as "exp", "frac", "Tukey", "GFAB", "fidel", "const" and "ramp" respectively. They are summarized in Figures 3.3 and 3.2.

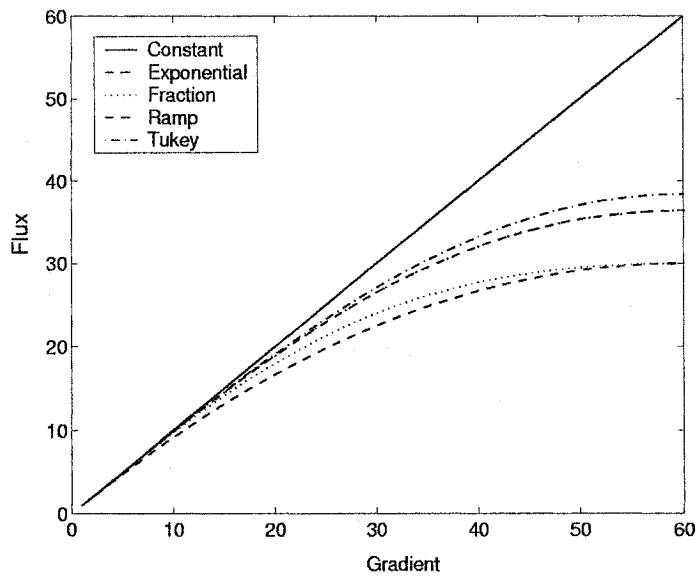


Figure 3.4: The "Mono" fluxes produced by each CF

Experiments are designed based on the desired flux shown in Figure 2.6. The following results were sought:

1. AD behavior under a monotonically increasing flux as shown in Figure 3.4.
2. AD behavior using the flux's derivative for the CF.
3. AD behavior under a mixed flux shown in Figure 3.5 containing both a monotonically increasing flux and a monotonically decreasing flux.

4. Comparison of IAD and TAD techniques. IAD is realized by our proposed discrete version for AD with the range of negative DC set to zero. Its flux is in the shape as in Figure 3.5 that shows the desired feature of ill-posed diffusion process in constrast to the interest only on well-posed process [249]. TAD is realized by the P-M numerical approximation algorithm, the most commonly used discrete version for AD (equations (2.5) to (2.10)).

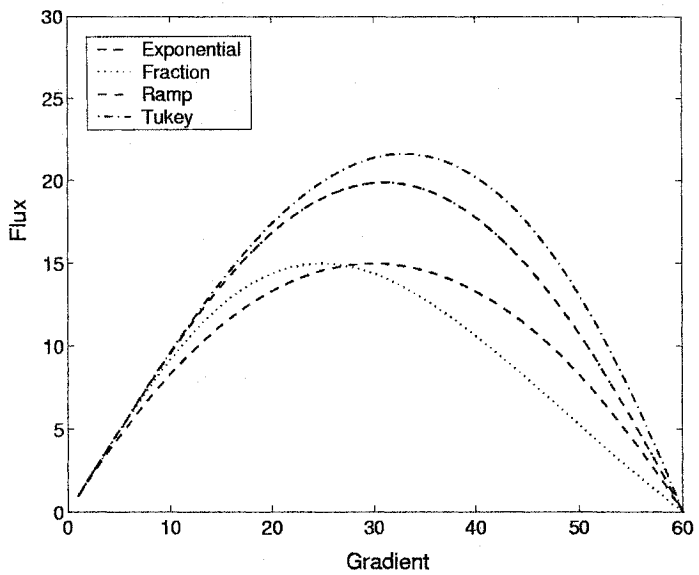


Figure 3.5: The "Mixed" fluxes generated by the CFs

The curve of the flux's derivative (Figure 2.6) has the same basic shape as that of FAB diffusion (Figure 3.2). Thus they are considered together. For convenience, these types of flux's curve in Figures 3.4, 2.6, 3.5 are called "Mono", "FAB" and "Mixed" respectively. "Mixed" flux is produced by the non-negative part of DF while "Mono" flux contains only the monotonically increasing part of "Mixed" flux. CFs are scaled so that their fluxes can be compared with each other. For example, the intervals of monotonically increasing fluxes produced by equations (2.3) and (2.4) are $[0, \frac{K_1}{\sqrt{2}}]$ and $[0, K_2]$ respectively. To make them produce the comparable fluxes, set $K_2 = \frac{K_1}{\sqrt{2}}$.

3.5.3 Experiments on 1D signals

Firstly, the CFs are scaled to produce the "Mono" fluxes within the same range as shown in Figure 3.4, and are set to zero outside this range. T_e is selected under the assumption that a given percentage of the pixels in the image are edge pixels [171]. In this experiment we chose 20%. Figures 3.6 and 3.7 show the results. Note that all five "Mono" CFs produced identical results when AD became idempotent. In our experiment, if the total change for a resultant image remains unchanged within a preset resolution ϵ for at least one hundred consecutive iterations, we consider it to be idempotent on this solution. For 1-D experiments $\epsilon = 10^{-5}$. Note also that the number of iterations for each CF to result in AD idempotency is different.

Table 3.1: Iterations for "Mono" DCs to give idempotent results

CF	Exponential	Fraction	Tukey	Constant	Ramp
Iterations	1989	1993	1989	2023	1983

After ten iterations, there are clear differences in the results with the five CFs. As the results show, the edges are clearly defined by 100 iterations, as can be seen if we compare the results at 100 iterations with the final results. In other words, typically the work for edge preservation is performed early in processing 1D signals. Further processing makes the image increasingly smoother by implementing smoothing in the areas of noise and trivial details.

Secondly, we looked at AD behavior with the flux as in Figure 2.6, when the CF incorporates negative values. CF equations "exp" and "GFAB" shown in Figure 3.2 were used. The "exp" is derived so that it generates flux as shown in Figure 2.6. The maximum smoothing force F_s must be larger than the maximum enhancing force $|-F_e|$ to maintain the stability of smooth regions [83].

We considered the behaviors of the AD process with respect to the interval $(K,\ S)$, the width of negative flux range. To emphasize this, we call it "Mixed" flux plus a negative range of CF of different widths. The threshold T_e for "exp" is set by assuming 20% of pixels in the image are edge-pixels. Parameters for the "GFAB" CF were calculated by equation (3.94) proposed in [83]. Also, the performance of "fidel" is considered so that the effectiveness of FAB diffusion could be addressed.

Results in Figure 3.8 show that selection of the negative CF region is important to the quality of resulting images. A narrow range may help to preserve edges. A wider negative region, though it sharpens edges, creates distortion. This can be seen by comparing the results for different widths. As Figures 3.8(g) and (h) show, the distortion can occur in the first few iterations. The "GFAB" filter exhibits similar behavior as shown in Figure 3.9, but with significant distortion due to its wide range of negative CF.

Adding a fidelity term as in equation (3.95) is helpful for mitigating the distortion. The fidelity term restricts the difference between the result and the original image. Results in Figure 3.10 show, however, that noise or trivial detail is also kept after the fidelity term is included. It shows that the results are also dependent upon the selection of λ. Tests for a small range of values for λ shows that the determination of this fidelity term is problematic. The distortion and the number of iterations to make the diffusion process reach idempotency are dependent on both λ and the width of the region of negative CF.

Results on the effect of negative CF show that it is damaging to the quality of resultant images, especially when the CF is negative over a wide range of gradient magnitudes. Thus the range of negative CF should be restricted.

We evaluated AD using CFs that produce the "Mixed" fluxes shown in Figure 3.5. Based on the experimental results when the CF incorporates negative values, in this experiment the range of negative CF is set to zero width. Figures 3.11 and 3.12 show the results at 10 and 100 iterations and the final results when AD became idempotent. All CFs produce the same final results except for the "frac" CF that has one additional edge at the top right corner. Again, edges are identified early in the diffusion process. Table 3.2 shows the number of iterations required to reach idempotency.

Table 3.2: Iterations for "Mixed" DCs to give idempotent results

CF	Exponential	Fraction	Tukey	Ramp
Iterations	1682	1688	1697	1680

The effect of edge enhancement by the "Mixed" flux is clear by comparing its results with those produced by the "Mono" fluxes (Figures 3.6, 3.7, 3.11 and 3.12). More edges were enhanced due to the monotonically decreasing flux in region $(K_1,\ K)$ of the "Mixed" flux shown in Figure 2.6. The comparison of the "Mixed" DC filter and "Mono" flux filter is shown in Figure 3.13. It shows that the "Mixed" flux keeps not only the meaningful edges but also avoids the distortion as produced by FAB filters. This "Mixed" flux, probably with a small range of negative CF, is now called the idempotent anisotropic diffusion (IAD) filter.

The proposed IAD technique was compared with the TAD techniques in detail as shown in Figures 3.14 and 3.15. The advantage shown in IAD diffusion, however, is not gained if T_e is ignored as in TAD techniques. That is equivalent to setting $T_e \to \infty$. With only the traditional edge threshold T_n for the edges to be kept, the

AD technique produces meaningless results with no edges left (Figure 3.15).

Comparing the ramps in these two figures (Figures 3.14 and 3.15) marked with "*" we note that the IAD filter indeed preserves and enhances edges. A profile of the resultant image has already been roughly created within the first ten iterations with semantically meaningful edges being kept. Further iterations continue to smooth regions and enhance edges. In contrast, the TAD filter continually smooths edges until they disappear. No matter how steep an edge, it will eventually disappear.

Histograms (Figures 3.16 and 3.17) show the corresponding intensity distributions and gradient distributions for the 1D signals processed by IAD and TAD filters respectively. "Frequency" indicates the number of pixels with the same intensity value or gradient value as a percentage of the total number of pixels in the image at the indicated number of iterations. Figure 3.16(a) shows the redistribution of intensity produced by IAD. The subfigure "Original" in Figure 3.16(a) presents the original intensity distribution of this 1D image. The process of re-distributing intensity continues until IAD reaches the idempotent state. As the subfigure "Idempotent" in Figure 3.16(a) shows, intensity has been re-grouped, and is still widely distributed after IAD filter reaches the stable state.

Figure 3.16(b) and Table 3.3 show the corresponding gradient histogram. Limited by the width of the paper, Table 3.3 records gradient in groups with more detailed attention paid for the noise range. In the experiments, the threshold was $K = 4$ that is emphasized in Table 3.3. At the beginning, as Table 3.3 shows, pixels whose gradient is zero constitute about 16% of the image. After IAD reaches the idempotent state, the zero-gradient pixels are about 80% of the image. In other words, IAD has removed most of the noise and trivial details in the image. The remaining 20% of pixels are shared by edges with different gradients.

Table 3.3: Gradient re-distribution produced by IAD in 1D experiment

	Gradient magnitude										
	0	1	2	3	4	5	6	7	$8 \sim 10$	$11 \sim 15$	≥ 16
Original	15.8	26.8	17.1	13.6	8.8	5.7	2.6	0.4	3.5	3.5	2.1
T=10	51.3	24.1	1.3	0.0	3.1	5.7	3.9	2.2	3.1	4.0	2.2
$T=10^2$	75.0	3.5	0.0	0.0	0.6	4.4	3.9	2.2	3.5	4.4	2.4
Idempotent	78.1	0.4	0.0	0.0	0.6	4.8	3.5	2.2	3.1	4.4	2.8

Table 3.4: Gradient re-distribution produced by TAD in 1D experiments

	Gradient magnitude										
	0	1	2	3	4	5	6	7	$8 \sim 10$	$11 \sim 15$	≥ 16
Original	15.8	27.2	17.5	13.6	8.8	5.7	2.6	0.4	3.5	3.5	2.1
T=10	34.2	39.5	14.5	5.3	2.2	0.0	0.0	0.4	1.3	0.0	2.4
$T=10^2$	54.4	37.3	6.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8
$T=10^3$	76.3	22.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0
$T=10^4$	98.1	1.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Idempotent	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figure 3.17(a) shows the convergence process of intensity values of pixels with TAD. Finally, as its subfigure "Idempotent" shows, every pixel has the same intensity value, the average of the total intensity of the image. Figure 3.17(b) and Table 3.4 illustrate the corresponding process showing the loss of gradients and hence edges.

3.5.4 Experiments with 2D images

Experiments on 2D images are more complex than those on 1D signals. Many more iterations are required for the AD process to reach the idempotent state. We explored AD behaviors with CFs "Mono", "Mixed", FAB and TAD techniques. Figures 3.18 to 3.25 show results with the synthetic image of Figure 3.18(b). This image is obtained by adding Gaussian noise of zero mean and σ of 10% of the maximum gradient magnitude to the noise-free image Figure 3.18(a). Figures 3.26 to 3.33 are the results for image "lena" and Figures 3.34 to 3.39 are the results with image "Lab". Different resolutions are tested with $\epsilon = 10^{-3}$ for "Art" and "Lena" while $\epsilon = 10^{-5}$ for "Lab".

All the results show that for 2D images, AD behavior is similar to 1D behavior, but apparently the processing is much more complicated. In 2D image processing, edges collapse more easily than in the 1D case because every pixel is affected by its neighbours. The "Mixed" flux shows much better performance in keeping edges than "Mono" flux, although some of edges, such as the edges on lena's hat, may be regarded as undesirable "staircases". The "Mono" flux is not good at keeping edges because it applies the strongest smoothing for gradient magnitudes near T_e while the "Mixed" flux implements edge enhancement and stops smoothing at these values. This is clearly shown by the results of Figures 3.18, 3.19, 3.24, 3.25 and Figures 3.30 to 3.33. Results with TAD techniques in Figures 3.22, 3.23, 3.26, 3.27, 3.36 and 3.37 exhibit the problem that occurs when a zero-flux threshold for true edges is omitted. Figures 3.20, 3.21, 3.28, 3.29, 3.34 and 3.35 show the unpredictable processes produced by negative CF employed in FAB techniques with noise still remaining. The diffusion process is uncontrollable even with a fidelity term. Determining the number of iterations is strongly image-dependent.

As for the 1D case, detailed analysis with statistical data is also given to com-

Table 3.5: Gradient re-distribution produced by IAD in 2D images

	Gradient magnitude									
	0	1	2	3	4	$5 \sim 8$	**9** ~ 19	$20 \sim 29$	$30 \sim 39$	≥ 40
Original	78	232	200	131	81	141	91	27	13	1
T=10	152	346	157	88	49	78	79	28	14	1
$T=10^2$	326	402	72	23	14	30	82	29	13	3
$T=10^3$	513	289	21	5	10	24	78	33	15	5
$T=10^4$	667	155	5	3	9	24	71	36	21	7
$T=10^5$	761	63	4	2	9	22	65	37	20	11

pare IAD with TAD on 2D images. Results from "Lab" are used for calculation. Figure 3.39 and Table 3.5 show the gradient results from our IAD filter. Gradient magnitude of 9 is the threshold for edges and is emphasized in bold font. Because the number of pixels in 2D image is much larger than in 1D experiments, "Frequency" for recording the histogram is normalized by 1000. As in the 1D experiment, the most semantically meaningful features have been kept in the diffusion process by IAD. Though more time is needed to process the 2D image, this clearly shows how our IAD behaves in the expected way, preserving and enhancing edges, removing noise and trivial details.

The 2D gradient results of the TAD filter are shown in Figure 3.37 and Table 3.6. From subfigures 3.37(b) to 3.37(f), edges are continually removed. Though there are still some pixels with high gradient magnitudes kept in subfigure 3.37(f), the image is meaningless.

Intensity histograms and gradient histograms for the 2D image results from IAD and TAD are calculated in the same way as for 1D results, with some modifications.

Table 3.6: Gradient re-distribution produced by TAD in 2D images

	Gradient magnitude									
	0	1	2	3	4	$5 \sim 8$	$9 \sim 19$	$20 \sim 29$	$30 \sim 39$	≥ 40
Original	78	232	200	131	81	141	91	27	13	1
T=10	150	349	168	100	56	82	57	20	11	0
$T=10^2$	275	405	128	72	41	47	11	9	9	0
$T=10^3$	405	417	116	35	9	5	0	0	0	0
$T=10^4$	519	413	54	7	2	1	0	0	0	0
$T=10^5$	829	170	0	0	0	0	0	0	0	0

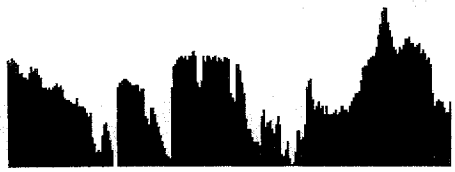
An extra subfigure showing the detail for non-zero gradient is added. The number of iterations needed for both AD approaches to reach the idempotent state is much larger than for 1D signals. Though both approaches did not reach the idempotent state in 10^5 iterations, both intensity histograms and gradient histograms (Figures 3.40 to 3.43) show the same conclusion as for 1D results. The subfigure "$t = 10^5$" in Figure 3.40 shows that TAD has packed all pixels into a small intensity range. Its corresponding gradient histogram (two subfigures marked "$t = 10^5$" in Figure 3.41) shows that all gradients have magnitudes of zero or one. On the other hand, subfigures "$t = 10^5$" in Figure 3.42 shows that our IAD approach causes pixels to be grouped at a wide range of intensity values. Table 3.5 tells us that more than ten percent of the pixels are kept as edges.

The advantage of the IAD filter over the TAD filter is illustrated clearly if the result images are shown in 3-D view. As Figure 3.45 shows, the IAD filter generates terraced fields on the image with edges kept or enhanced. These fields get more and more flattened with successive iterations. On the other hand, edges are kept being broken down by the TAD filter as shown in Figure 3.44. The final point of high

intensity left on Figure 3.44 (f) will collapse too if more iterations are implemented.

We have evaluated the effects of CF along the whole gradient magnitude range. Results show that:

- Two thresholds, T_n and T_e, are essential for implementing AD. With only the traditional threshold K_e, there is no guarantee that edges do not suffer from smoothing from the first iteration.
- A CF can implement smoothing, null diffusion, bounded unstable diffusion, and unbounded unstable diffusion. The CF can be selected based on the desired flux, thereby extending the range of functions used for CFs and making the selection of CF easy.
- Negative CF is dangerous to the quality of results unless its width is controlled cautiously.
- The flux's derivative makes the discrete version produce the desired IAD when the range of its negative value is set to zero or restricted to a small interval; while the commonly used P-M discrete formula of AD cannot avoid smoothing edges.
- Generally, significant diffusion occurs in the early stage.



(a) Original image

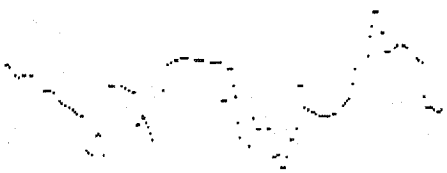
(b) By "exp" at t=10

(c) By "frac" at t=10

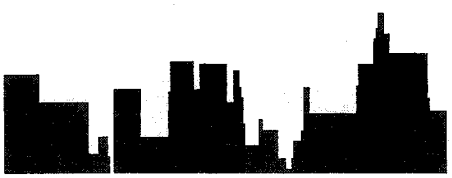
(d) By "Tukey" at t=10

(e) By "ramp" at t=10

(f) By "const" at t=10

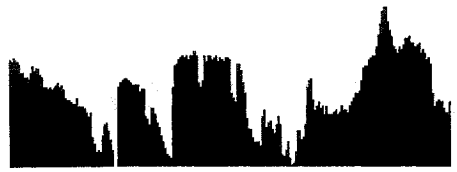


(g) Difference between (d) and (f)



(h) Idempotent result

Figure 3.6: Results produced by CFs generating "Mono" fluxes.



(a) Original image

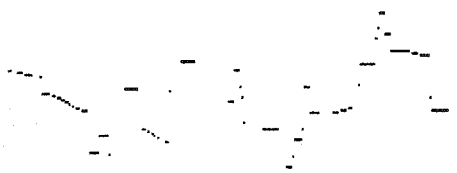
(b) By "exp" at t=10^2

(c) By "frac" at t=10^2

(d) By "Tukey" at t=10^2

(e) By "ramp" at t=10^2

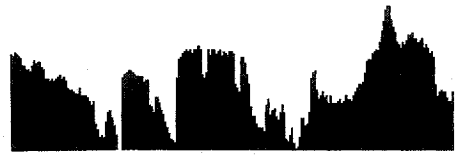
(f) By "const" at t=10^2



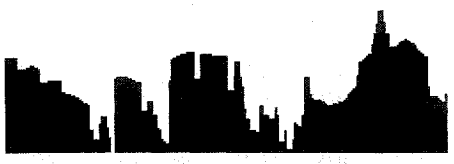
(g) Difference between (d) and (f)

(h) Idempotent result

Figure 3.7: Results produced by CFs generating "Mono" fluxes.



(a) Original image



(b) Width=1 and t=10

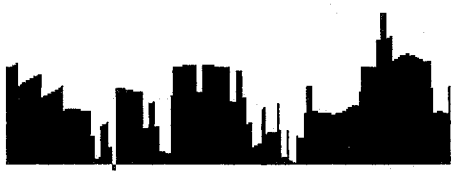
(c) Width=1 and t=50

(d) Width=1 (idempotent at 1681)

(e) Width=5 and t=10

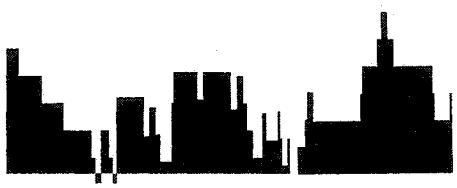
(f) Width=10 and t=10

(g) Width=5 and t=50



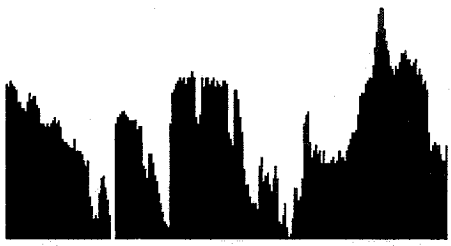
(h) Width=10 and t=50

(i) Width=5 (idempotent at 4232)



(j) Width=10 (idempotent at 6843)

Figure 3.8: Results of "Mixed" flux plus a negative CF of different widths.

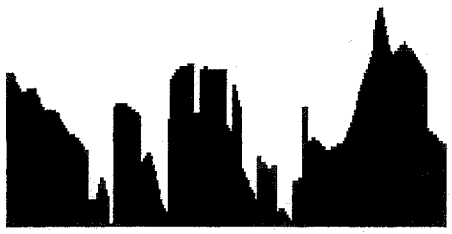


(a) Original image

(b) *Result at* $t = 10$

(c) *Result at* $t = 10^2$

(d) *Result at* $t = 10^3$



(e) *Result* at* $t = 10$

(f) *Idempotent* at* $t = 228$

Figure 3.9: Results produced by the "Gilboa" FAB filter. "$*$" are the results after a fidelity term is included with $\lambda = 0.05$ [83].

(a) Original image

(b) $\lambda = 0.01$ *at* $t = 10$

(c) $\lambda = 0.01$ *at* $t = 10^2$

(d) $\lambda = 0.01$ *at idempotent*($t = 1114$)

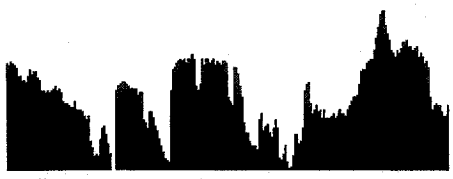
(e) $\lambda = 0.05$ *at* $t = 10$

(f) $\lambda = 0.05$ *at idempotent*($t = 228$)

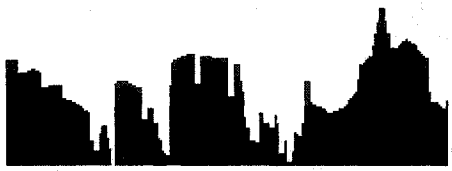
(g) $\lambda = 0.1$ *at* $t = 10$

(h) $\lambda = 0.1$ *at idempotent*($t = 116$)

Figure 3.10: Results of "fidel" of different λ.



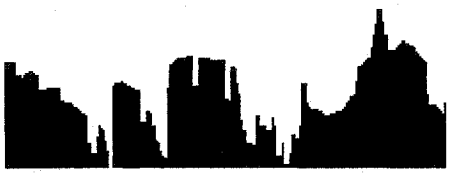
(a) Original image



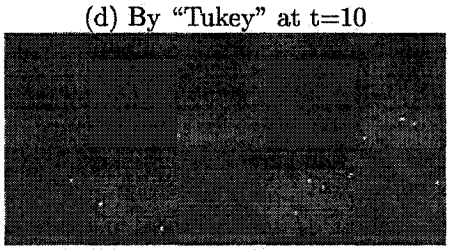
(b) By "exp" at t=10

(c) By "frac" at t=10

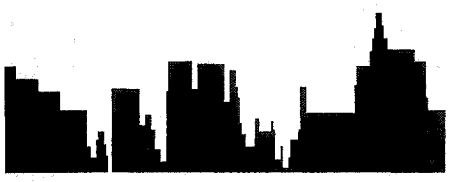
(d) By "Tukey" at t=10



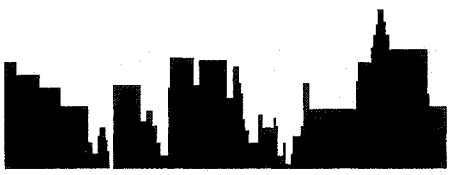
(e) By "ramp" at t=10



(f) Difference between (b) and (c)

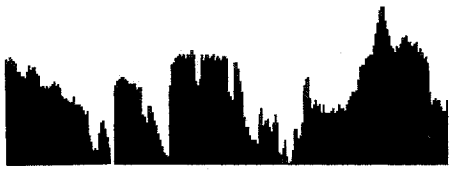


(g) Idempotent result by "frac"



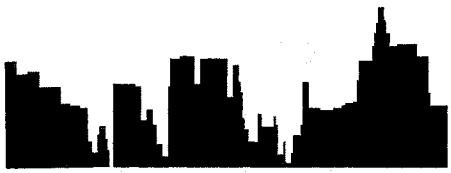
(h) Idempotent result by others

Figure 3.11: Results produced by CFs generating "Mixed" fluxes.

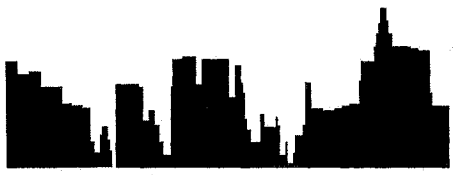


(a) Original image

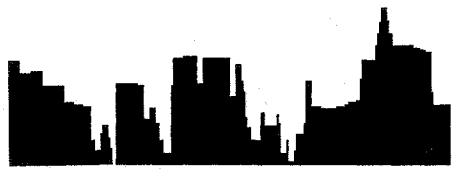
(b) By "exp" at t=10^2



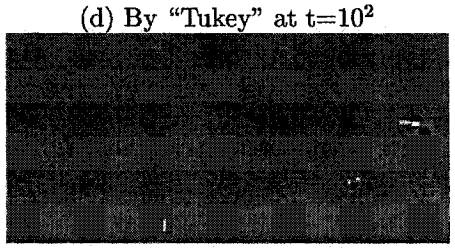
(c) By "frac" at t=10^2



(d) By "Tukey" at t=10^2



(e) By "ramp" at t=10^2

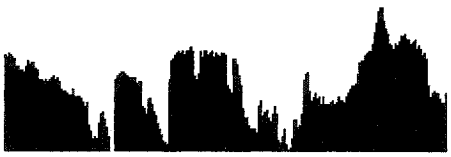


(f) Difference between (b) and (c)

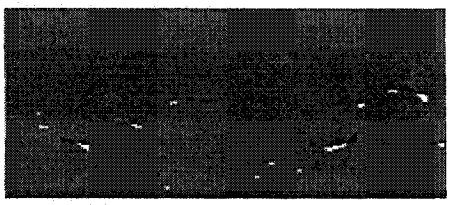
(g) Idempotent result by "frac"

(h) Idempotent result by others

Figure 3.12: Results produced by CFs generating "Mixed" fluxes.



(a) Original image



(b) Difference between (c) and (d)

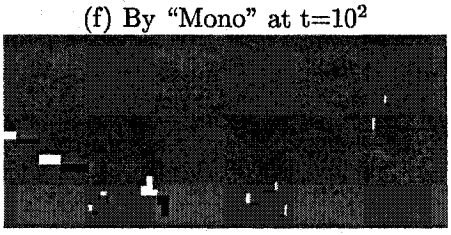
(c) By "Mixed" at t=10

(d) By "Mono" at t=10

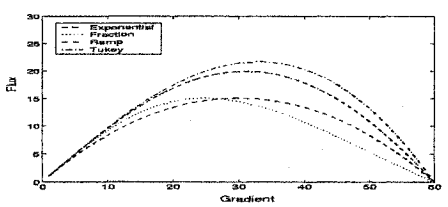
(e) By "Mixed" at t=10^2

(f) By "Mono" at t=10^2

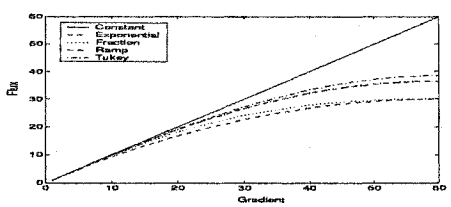
(g) Difference between (e) and (f)



(h) "Mixed" result vs. "Mono" result

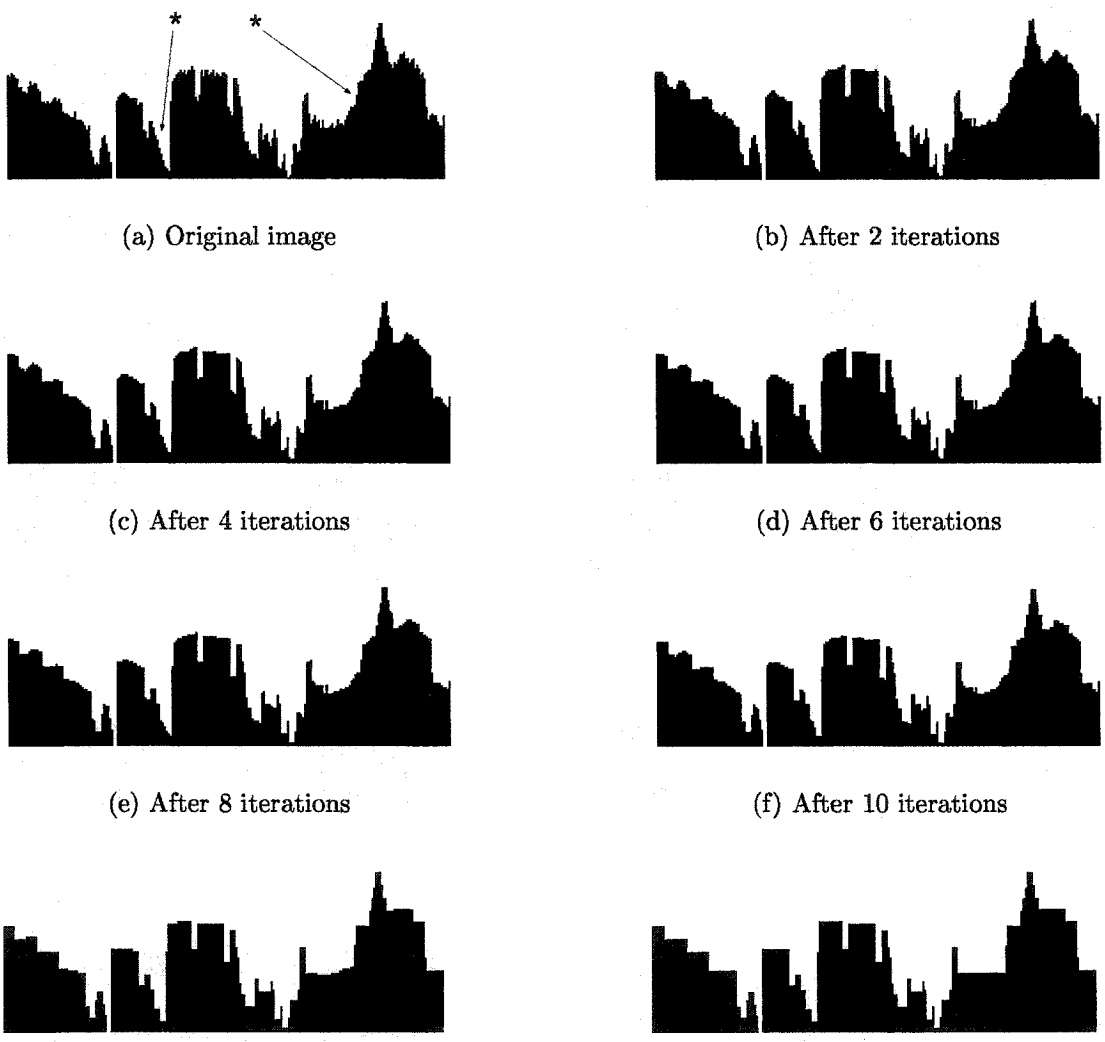


(i) "Mixed" result vs. original image



(j) "Mono" result vs. original image

Figure 3.13: Comparision between results of "exponential" of different flux.



(a) Original image

(b) After 2 iterations

(c) After 4 iterations

(d) After 6 iterations

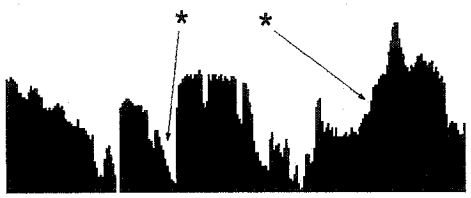
(e) After 8 iterations

(f) After 10 iterations

(g) After 10^2 iterations

(h) Idempotent at 1688 iterations

Figure 3.14: 1D results from the IAD filter.

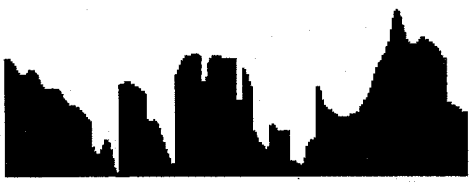


(a) Original image

(b) After 2 iterations

(c) After 4 iterations

(d) After 6 iterations



(e) After 8 iterations

(f) After 10 iterations

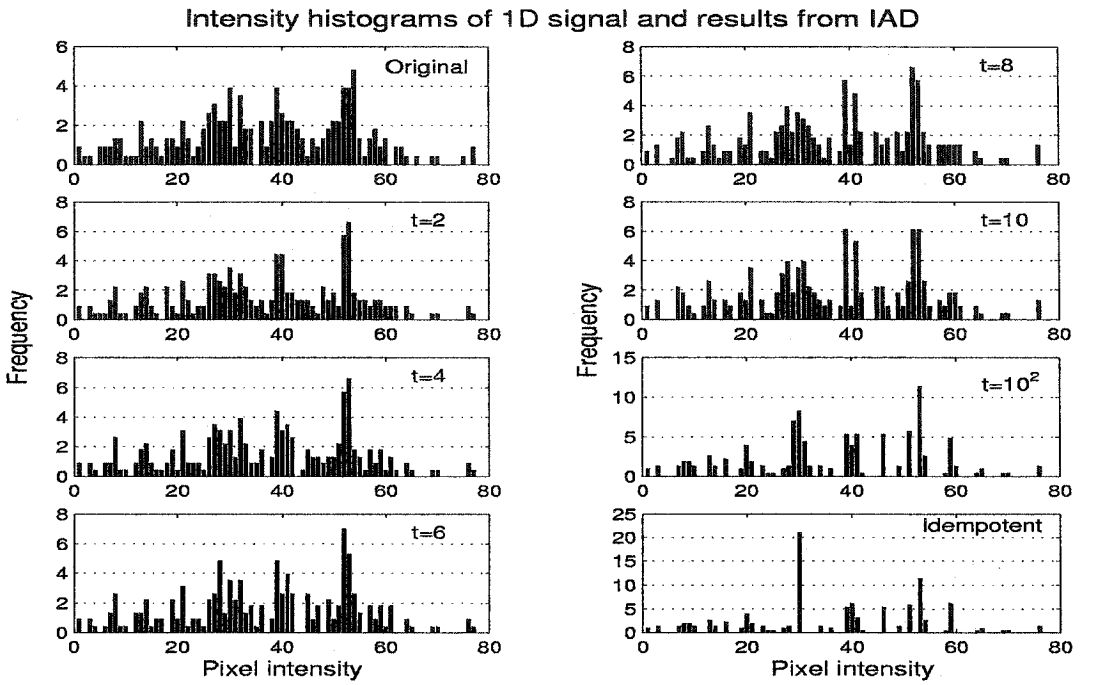
(g) After 10^2 iterations

(h) After 10^3 iterations

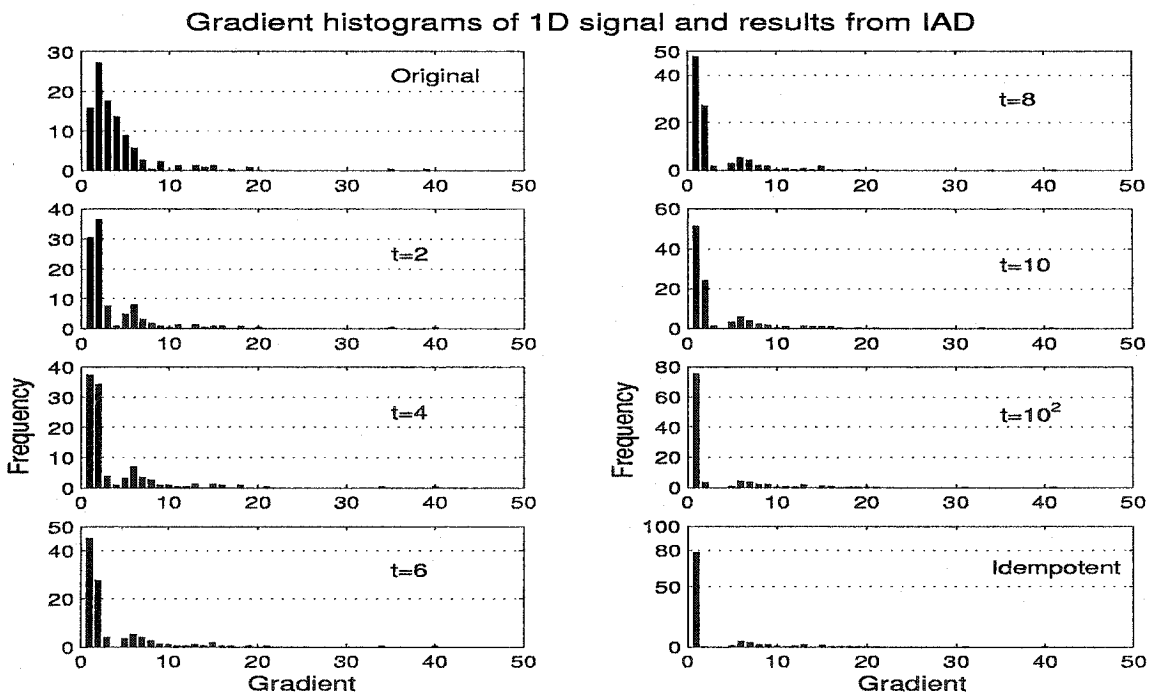
(i) After 10^4 iterations

(j) Idempotent at 88710 iterations

Figure 3.15: 1D results from the TAD filter.

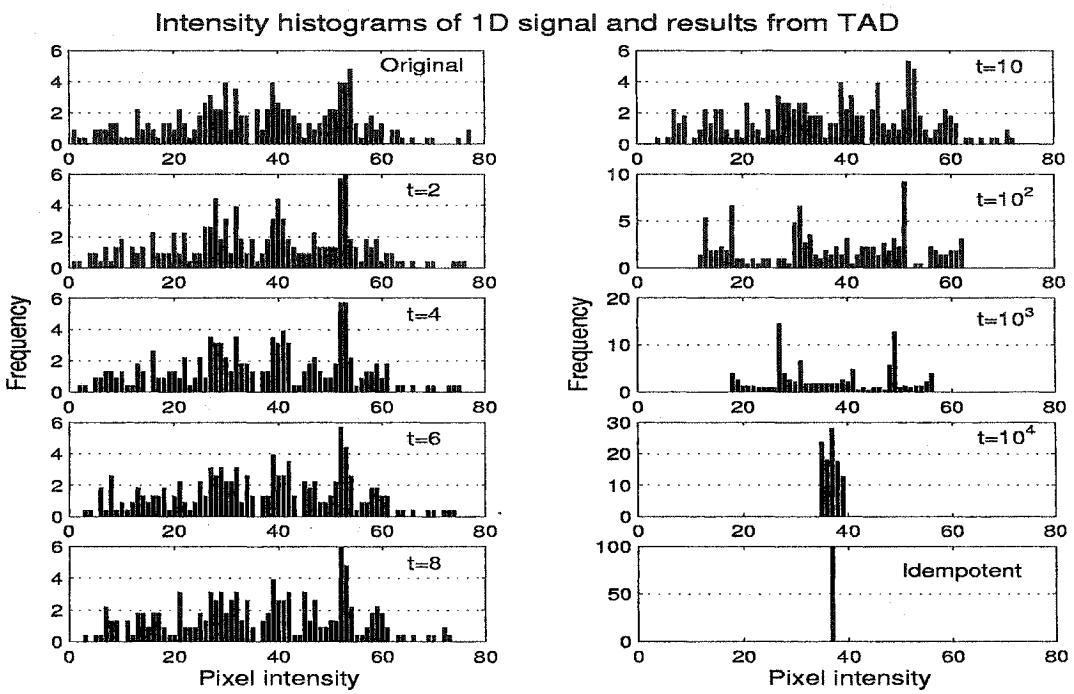


(a)

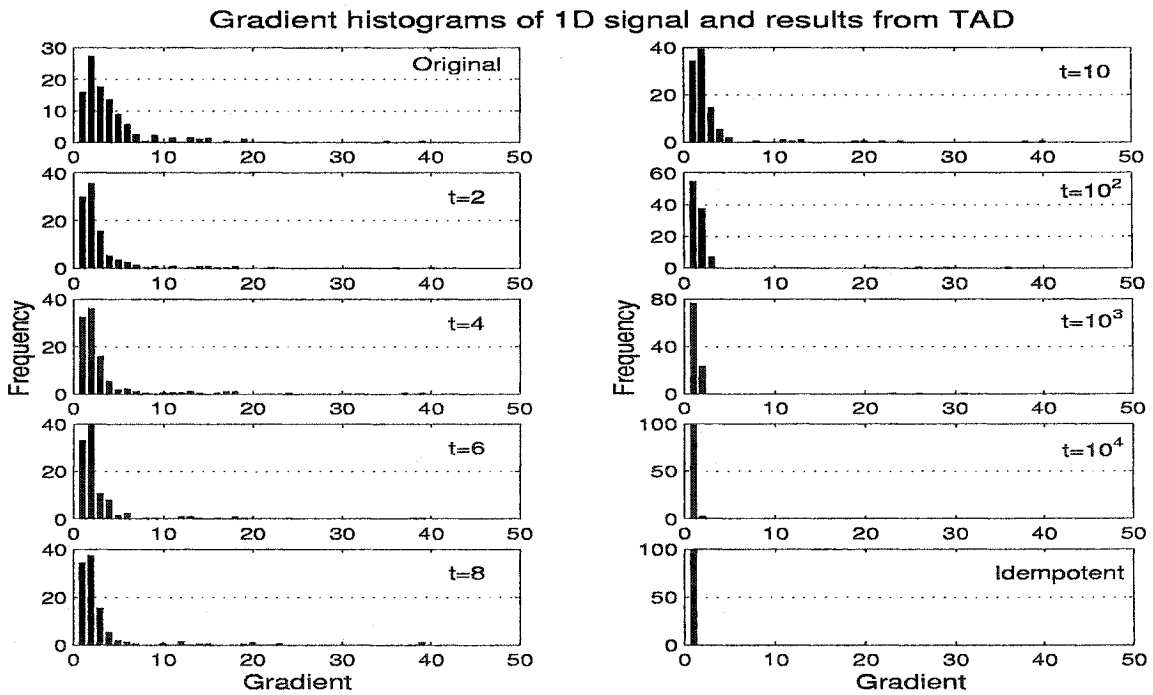


(b)

Figure 3.16: Histogram results from the IAD filter on 1D signal.

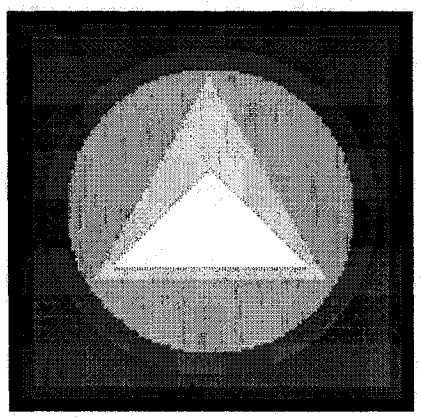


(a)

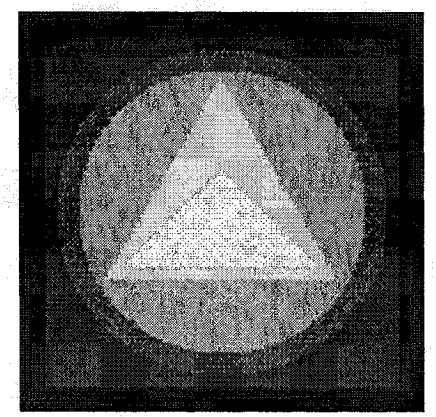


(b)

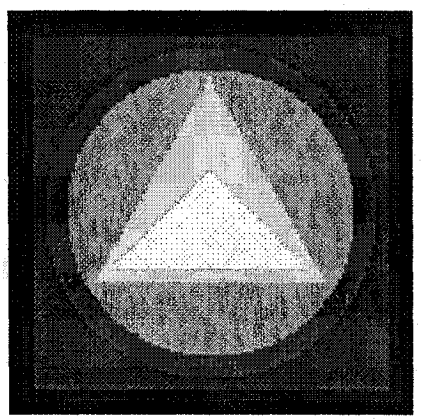
Figure 3.17: Histogram results from the TAD filter on 1D signal.



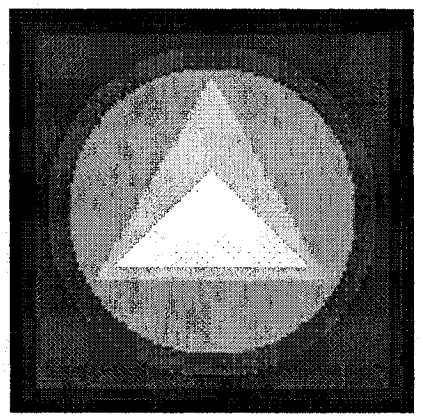
(a) Original image



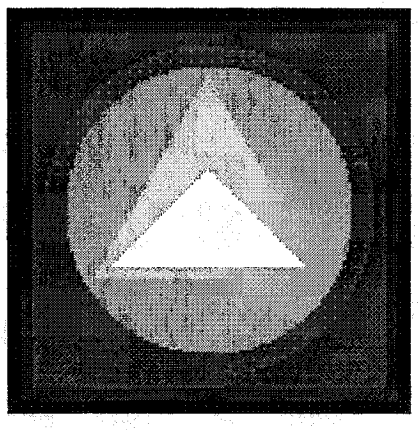
(b) Image noised for test



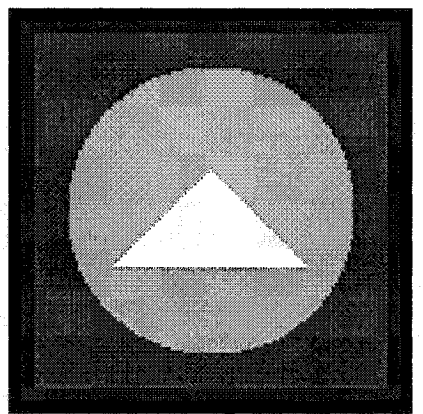
(c) Result at t=10



(d) Result at t=10^2

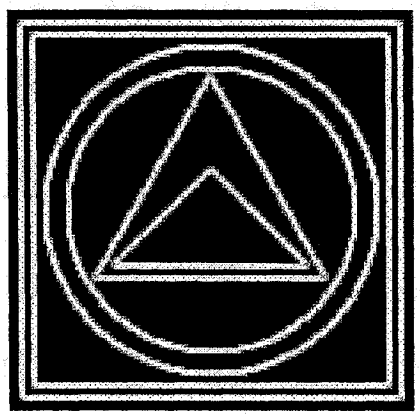


(e) Result at t=10^3

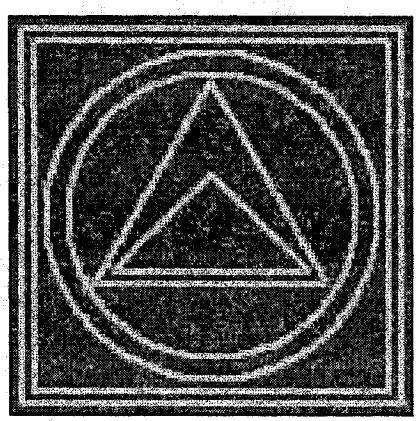


(f) Idempotent at t=7428

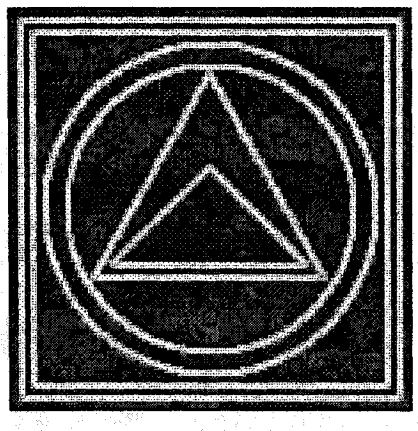
Figure 3.18: Image results produced by the "Mono" flux.



(a) Original gradient

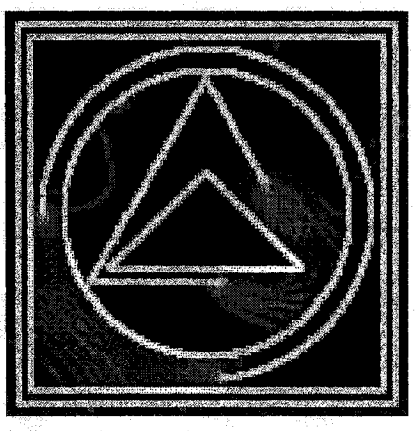


(b) Gradient of the test image

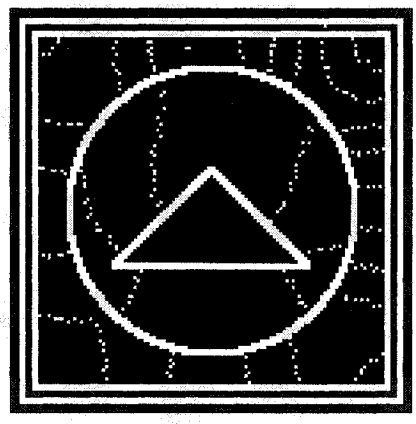


(c) Result at t=10

(d) Result at $t=10^2$

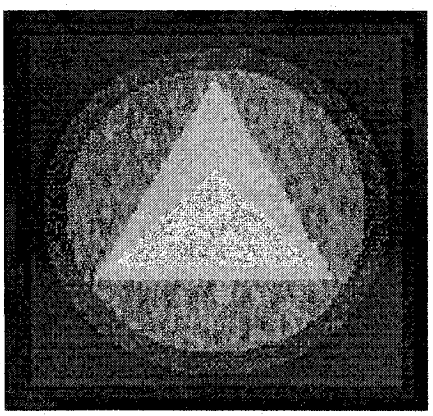


(e) Result at $t=10^3$

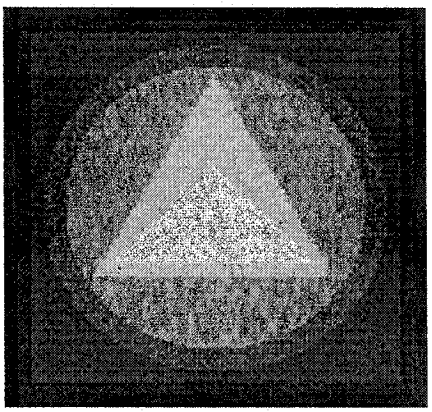


(f) Idempotent at t=7428

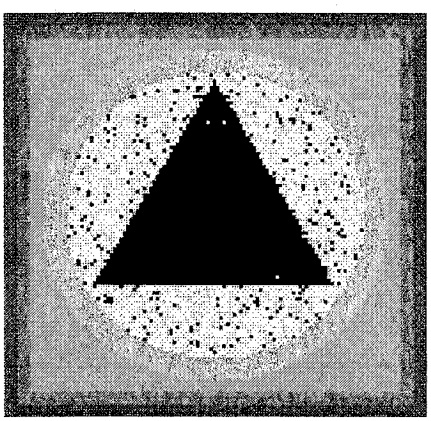
Figure 3.19: Gradient results produced by the "Mono" flux.



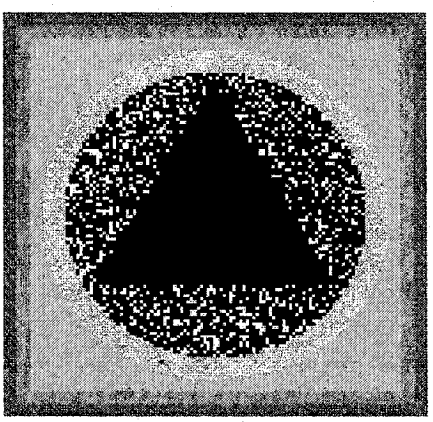
(a) $\lambda = 0.1$ at $t = 5$



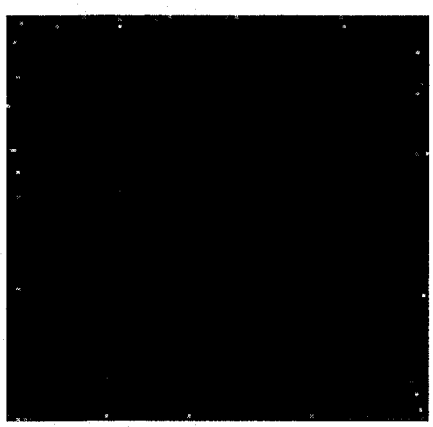
(b) $\lambda = 0.9$ at $t = 10$



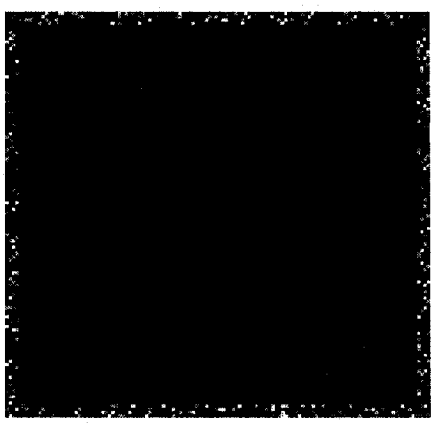
(c) $\lambda = 0.1$ at $t = 10$



(d) $\lambda = 0.9$ at $t = 40$

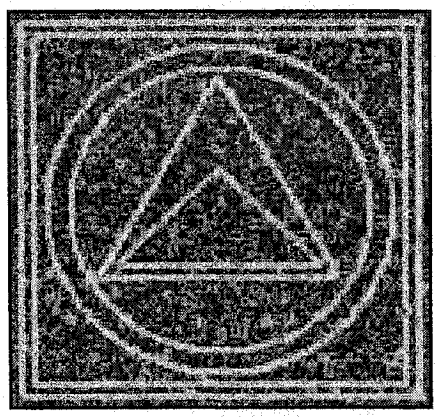


(e) $\lambda = 0.1$ at $t = 20$

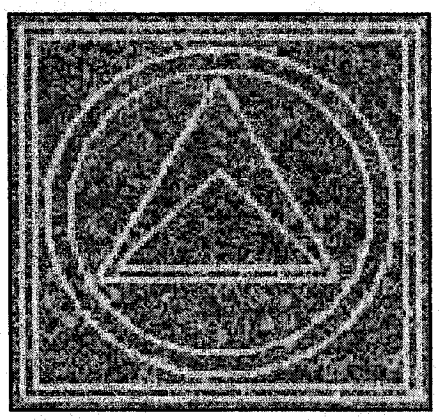


(f) $\lambda = 0.9$ at $t = 80$

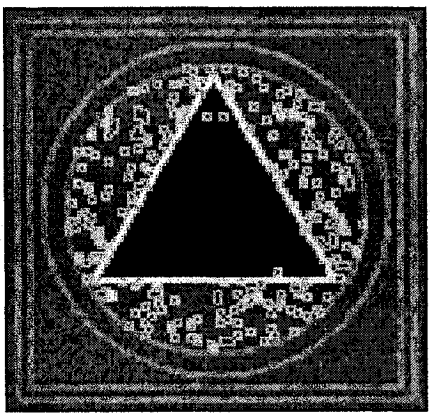
Figure 3.20: Image results by the "Gilboa" FAB filter with a fidelity term λ added.



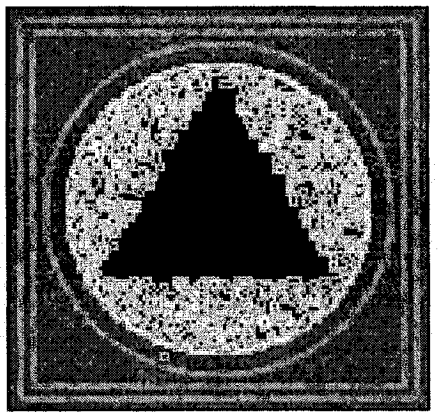
(a) $\lambda = 0.1$ at $t = 5$



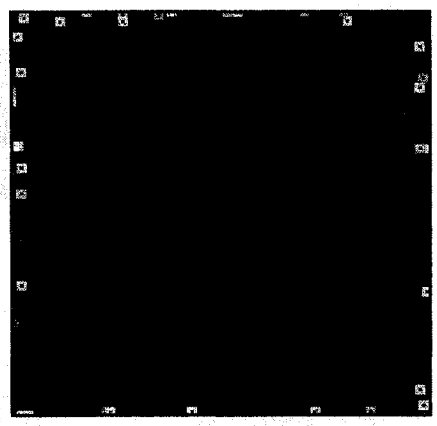
(b) $\lambda = 0.9$ at $t = 10$



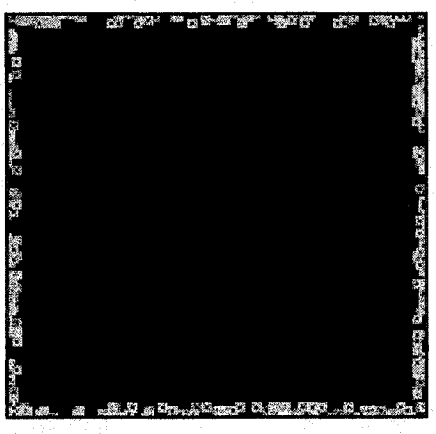
(c) $\lambda = 0.1$ at $t = 10$



(d) $\lambda = 0.9$ at $t = 40$

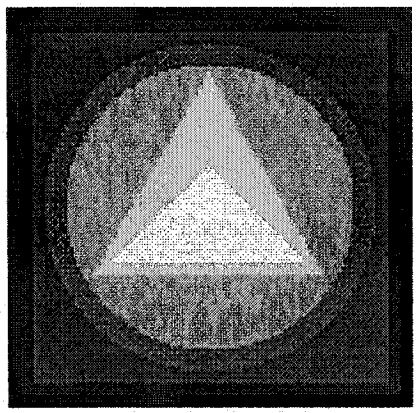


(e) $\lambda = 0.1$ at $t = 20$

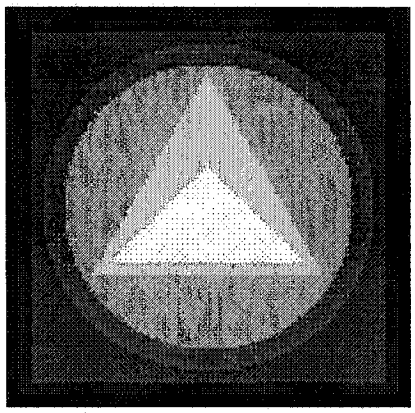


(f) $\lambda = 0.9$ at $t = 80$

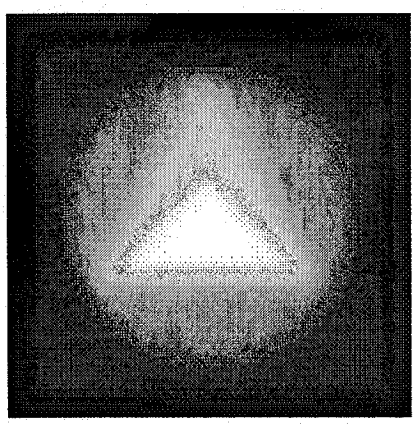
Figure 3.21: Gradient results by the "Gilboa" FAB filter with a fidelity term λ added.



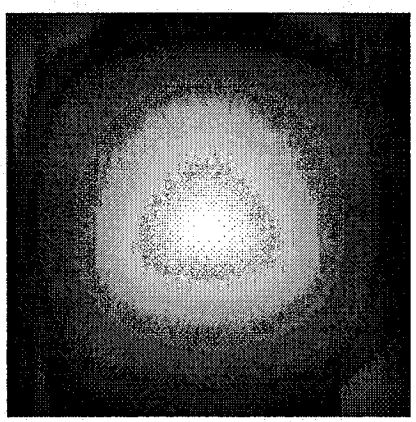
(a) Test image



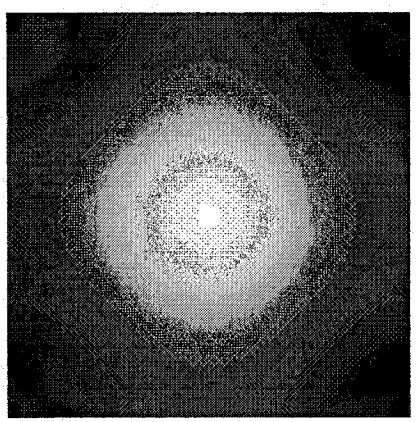
(b) Result at t=10



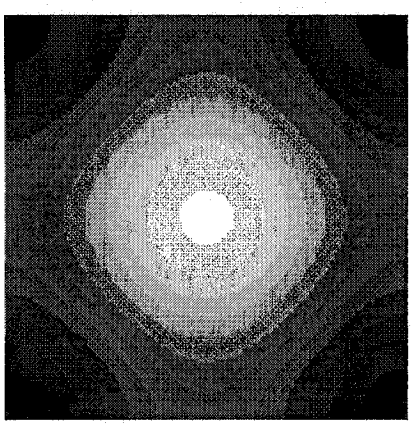
(c) Result at $t=10^2$



(d) Result at $t=10^3$

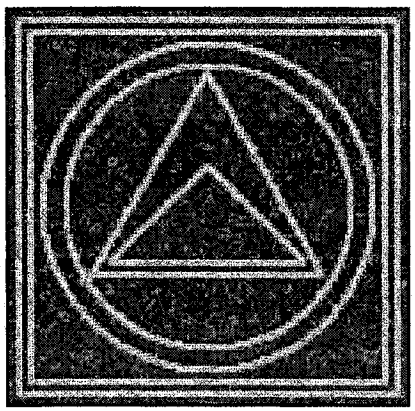


(e) Result at $t=10^4$

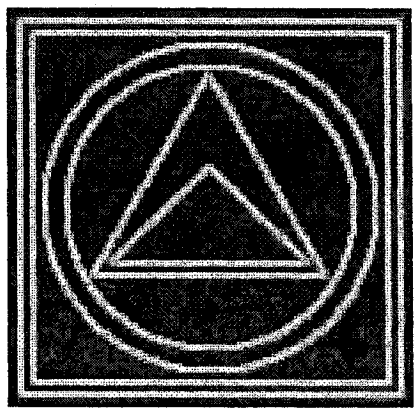


(f) Idempotent at t=14559

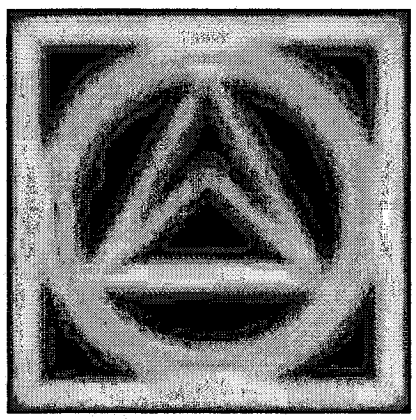
Figure 3.22: Image results produced by the traditional AD techniques.



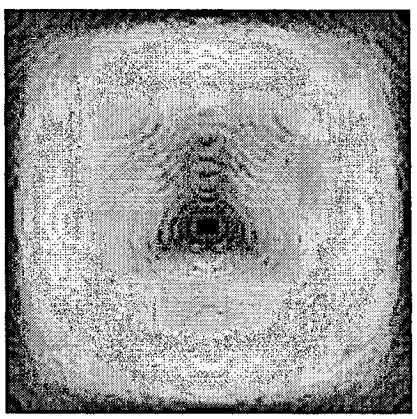
(a) Gradient of the test image



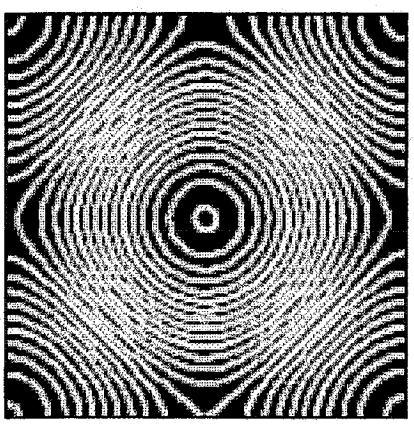
(b) Result at t=10



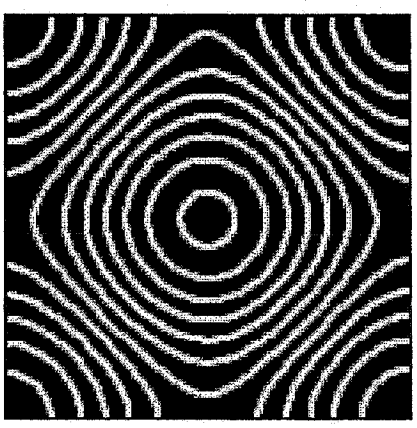
(c) Result at t=10^2



(d) Result at t=10^3

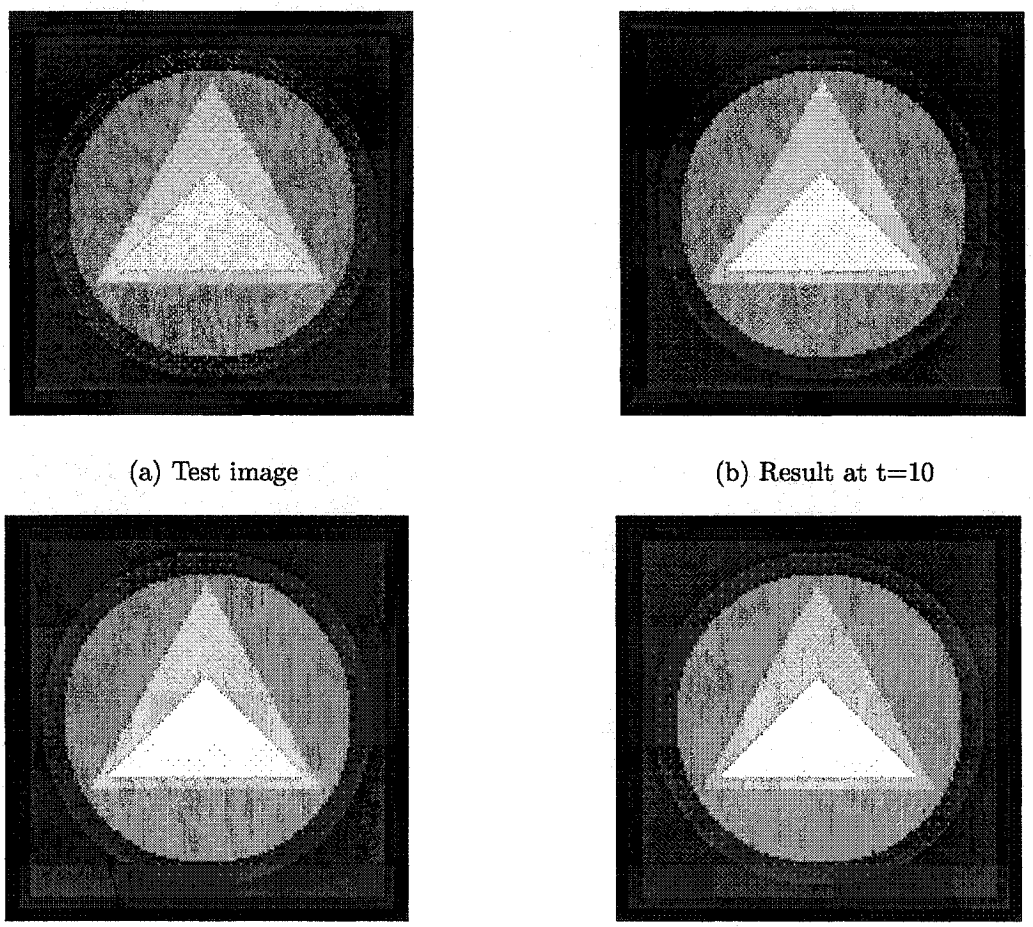


(e) Result at t=10^4



(f) Idempotent at t=14559

Figure 3.23: Gradient results produced by the traditional AD techniques.



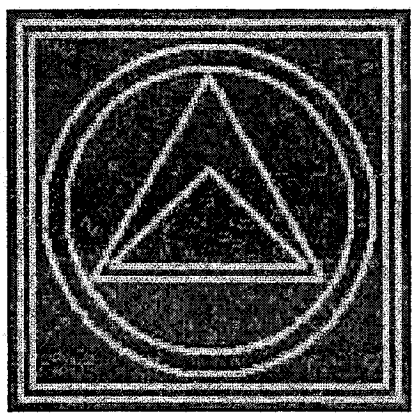
(a) Test image

(b) Result at t=10

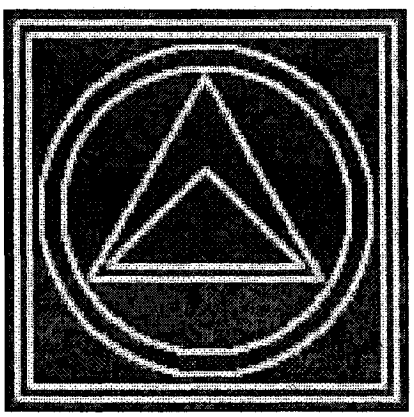
(c) Result at t=10^2

(d) Idempotent at t=372

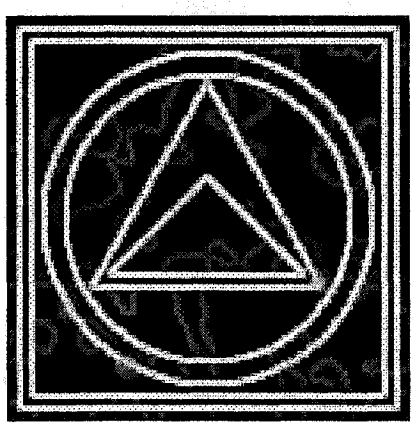
Figure 3.24: Image results produced by the "Mixed" flux.



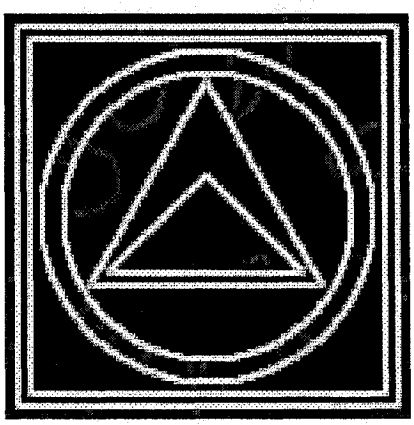
(a) Gradient of the test image



(b) Result at t=10



(c) Result at $t=10^2$



(d) Idempotent at t=372

Figure 3.25: Gradient results produced by the "Mixed" flux.

(a) Original image

(b) Result at t=10

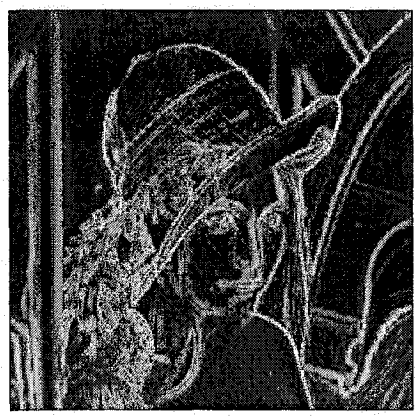
(c) Result at $t=10^2$

(d) Result at $t=10^3$

(e) Result at $t=10^4$

(f) Idempotent at t=46162

Figure 3.26: Results produced by traditional AD techniques.

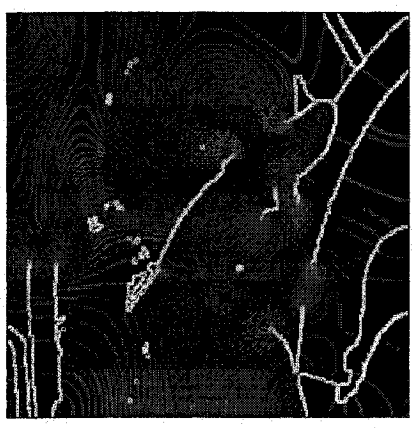


(a) Original gradient

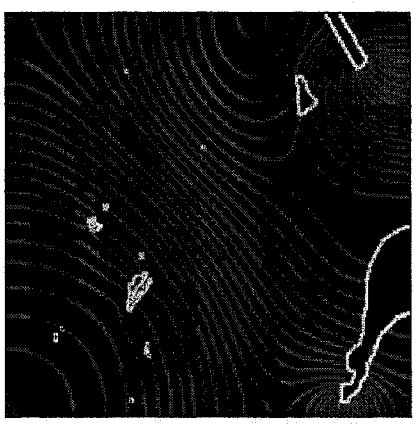
(b) Result at t=10

(c) Result at $t=10^2$

(d) Result at $t=10^3$



(e) Result at $t=10^4$



(f) Idempotent at t=46162

Figure 3.27: Gradient results produced by traditional AD techniques.

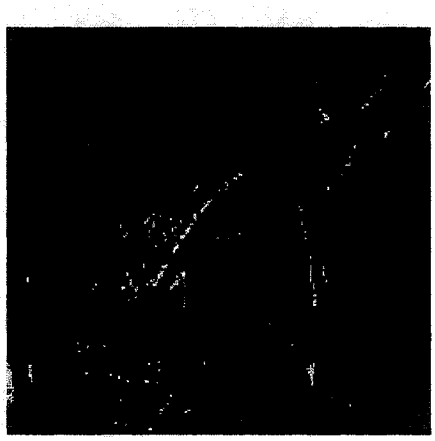
(a) $\lambda = 0.1$ at $t = 5$

(b) $\lambda = 0.9$ at $t = 10$

(c) $\lambda = 0.1$ at $t = 10$

(d) $\lambda = 0.9$ at $t = 30$

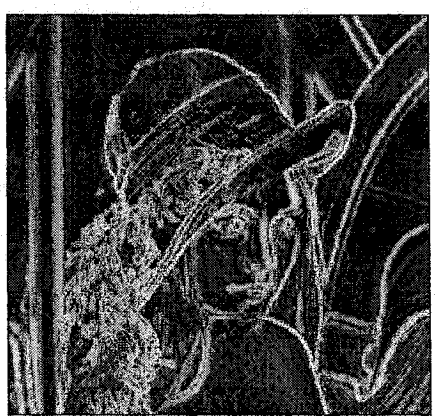
(e) $\lambda = 0.1$ at $t = 15$



(f) $\lambda = 0.9$ at $t = 60$

Figure 3.28: Image results by the "Gilboa" FAB filter with a fidelity term λ added.

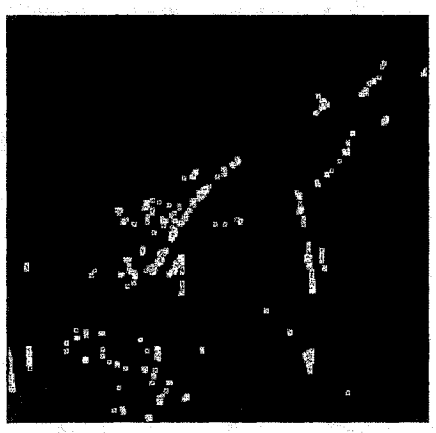
(a) $\lambda = 0.1$ at $t = 5$



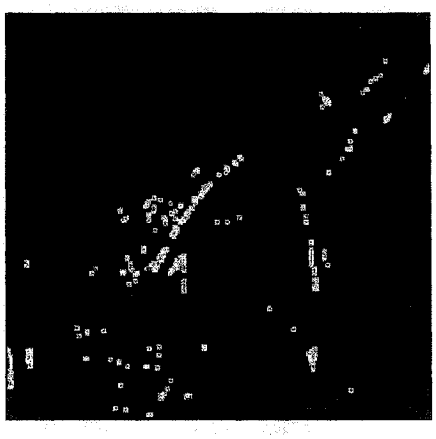
(b) $\lambda = 0.9$ at $t = 10$

(c) $\lambda = 0.1$ at $t = 10$

(d) $\lambda = 0.9$ at $t = 30$



(e) $\lambda = 0.1$ at $t = 15$



(f) $\lambda = 0.9$ at $t = 60$

Figure 3.29: Gradient results by the "Gilboa" FAB filter with a fidelity term λ added.

(a) Original image

(b) Result at t=10

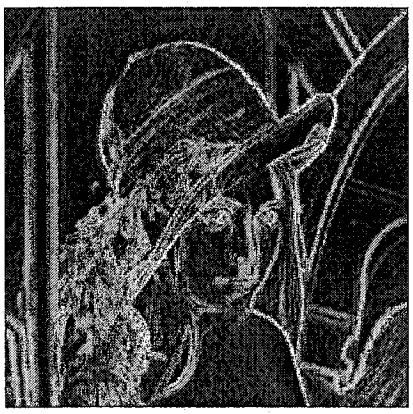
(c) Result at $t=10^2$

(d) Result at $t=10^3$

(e) Result at $t=10^4$

(f) Idempotent at t=107960

Figure 3.30: Results produced by "Mono" flux.



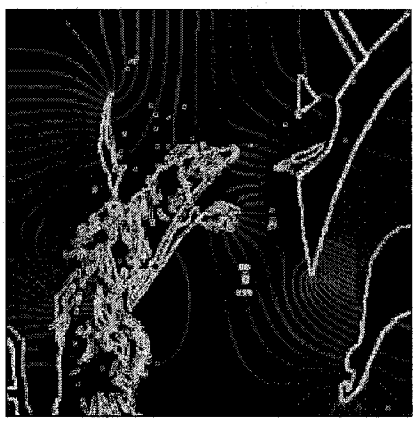
(a) Original gradient

(b) Result at t=10

(c) Result at $t=10^2$

(d) Result at $t=10^3$

(e) Result at $t=10^4$



(f) Idempotent at t=107960

Figure 3.31: Gradient results produced by "Mono" flux.

(a) Original image

(b) Result at t=10

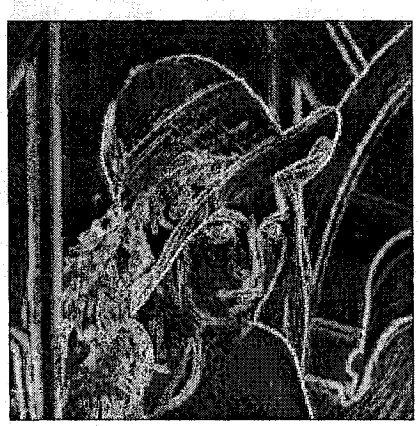
(c) Result at $t=10^2$

(d) Result at $t=10^3$

(e) Result at $t=10^4$

(f) Idempotent at t=27566

Figure 3.32: Results produced by the "Mixed" flux.



(a) Original gradient

(b) Result at t=10

(c) Result at $t=10^2$

(d) Result at $t=10^3$

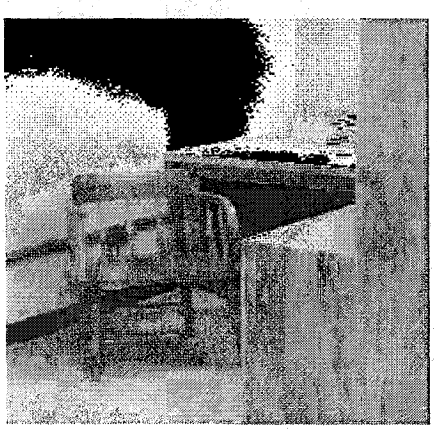
(e) Result at $t=10^4$

(f) Idempotent at t=27566

Figure 3.33: Gradient results produced by "Mixed" flux.

(a) $\lambda = 0.1$ at $t = 5$

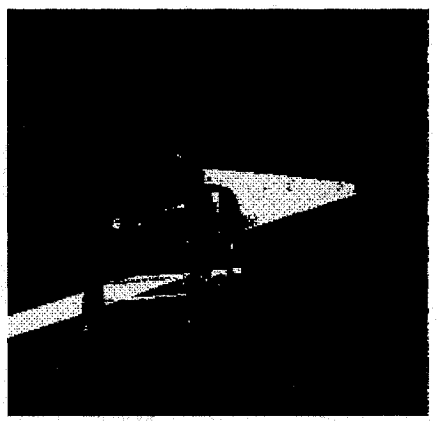
(b) $\lambda = 0.9$ at $t = 5$



(c) $\lambda = 0.1$ at $t = 10$

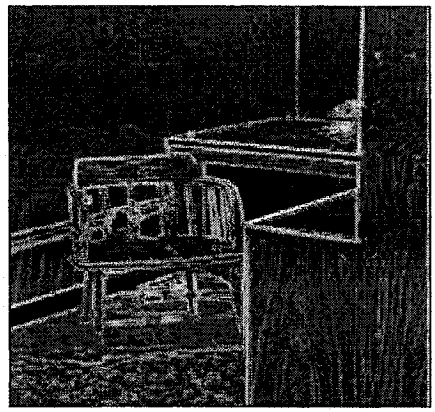
(d) $\lambda = 0.9$ at $t = 10$

(e) $\lambda = 0.1$ at $t = 30$

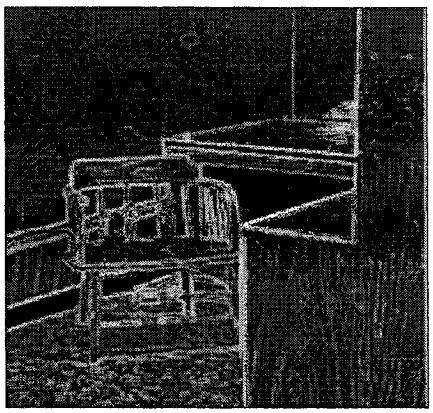


(f) $\lambda = 0.9$ at $t = 90$

Figure 3.34: Image results by the "Gilboa" FAB filter with a fidelity term λ added.

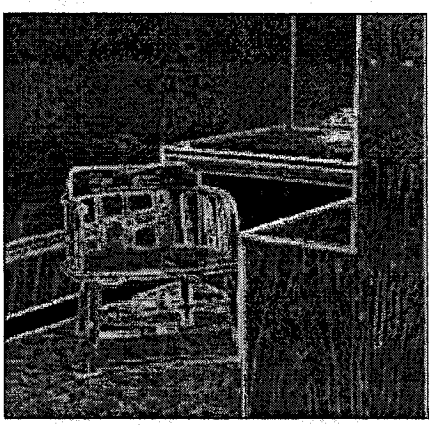


(a) $\lambda = 0.1$ at $t = 5$

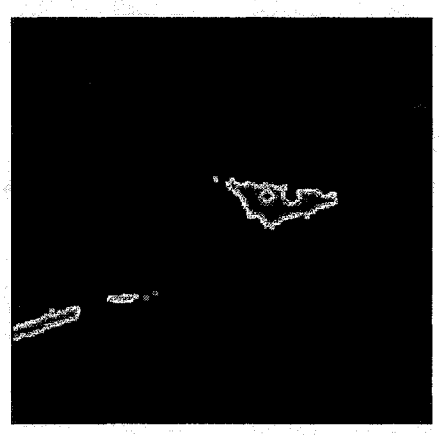


(b) $\lambda = 0.9$ at $t = 5$

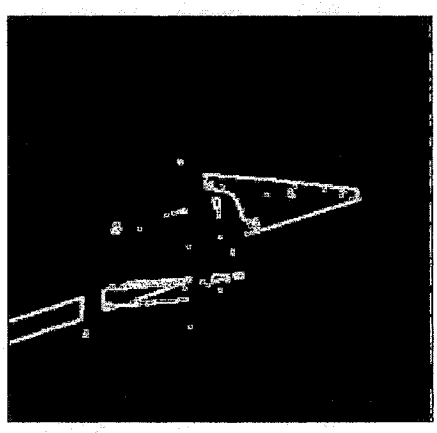
(c) $\lambda = 0.1$ at $t = 10$



(d) $\lambda = 0.9$ at $t = 10$



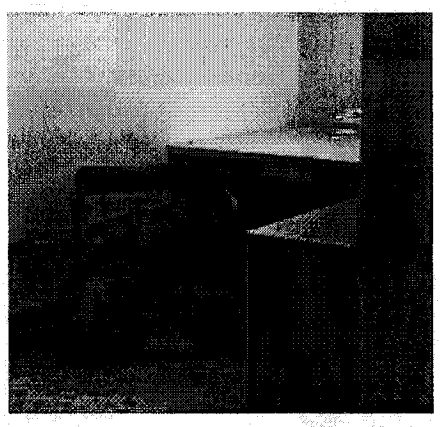
(e) $\lambda = 0.1$ at $t = 30$



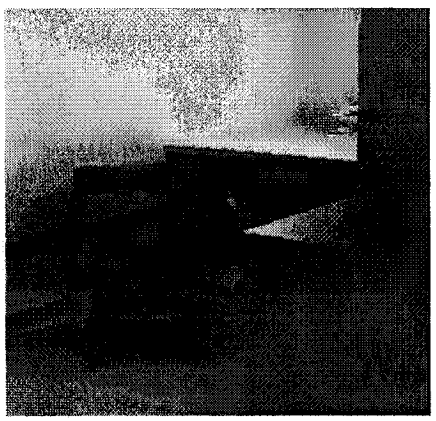
(f) $\lambda = 0.9$ at $t = 90$

Figure 3.35: Gradeint results by the "Gilboa" FAB filter with a fidelity term λ added.

(a) Original image

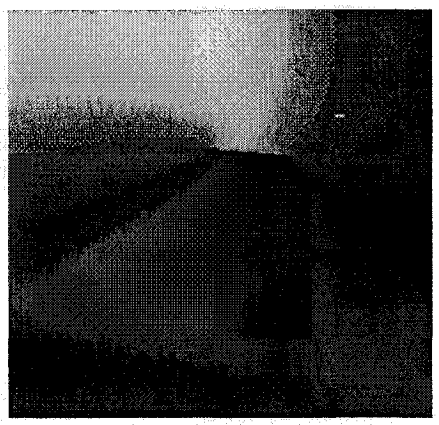


(b) After 10 iterations

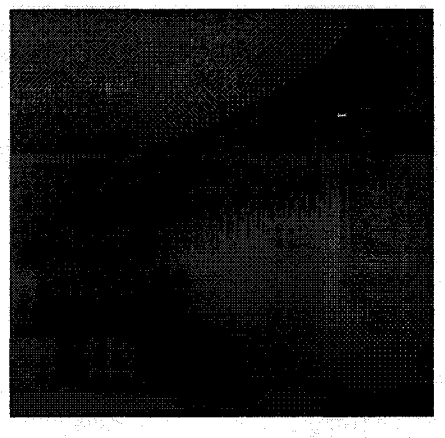


(c) After 10^2 iterations

(d) After 10^3 iterations

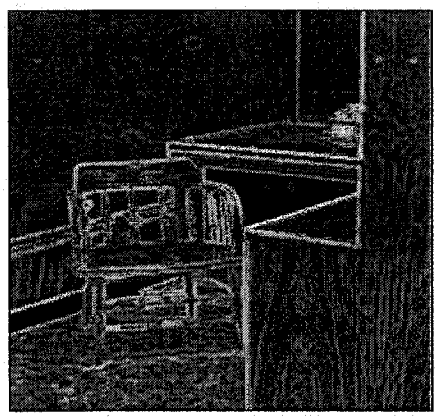


(e) After 10^4 iterations

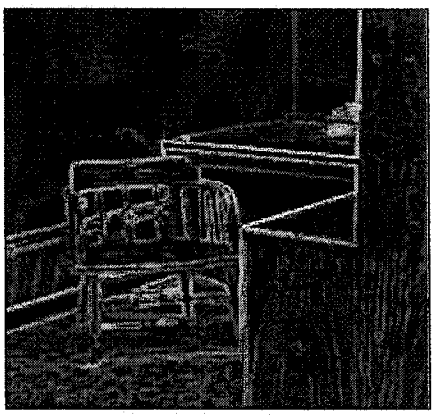


(f) After 10^5 iterations

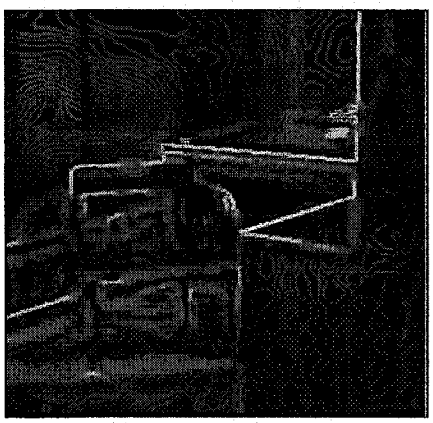
Figure 3.36: Results of TAD on 2D images.



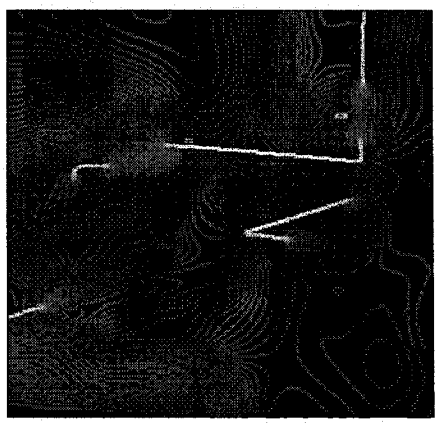
(a) Original image



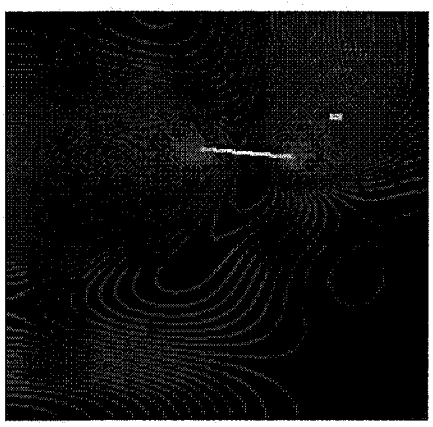
(b) After 10 iterations



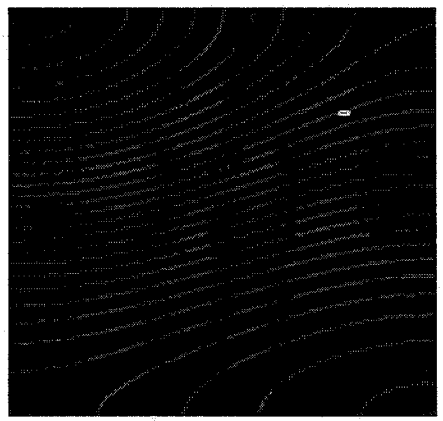
(c) After 10^2 iterations



(d) After 10^3 iterations



(e) After 10^4 iterations



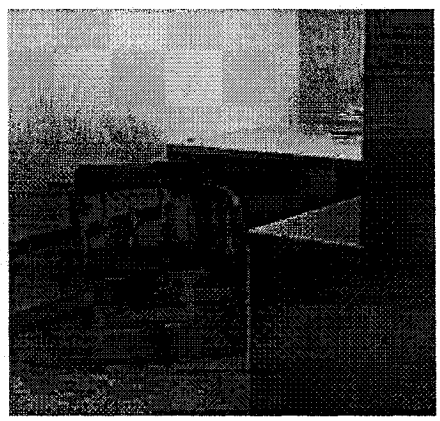
(f) After 10^5 iterations

Figure 3.37: Gradients resulting from the TAD on 2D images.

(a) Original image

(b) After 10 iterations

(c) After 10^2 iterations

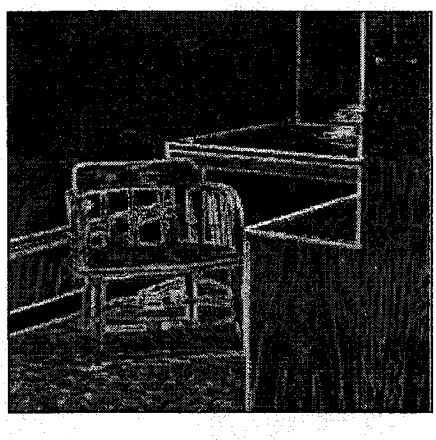


(d) After 10^3 iterations

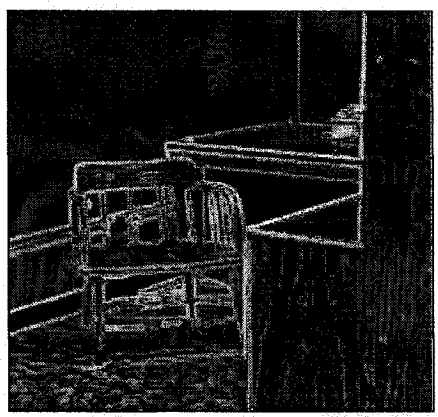
(e) After 10^4 iterations

(f) After 10^5 iterations

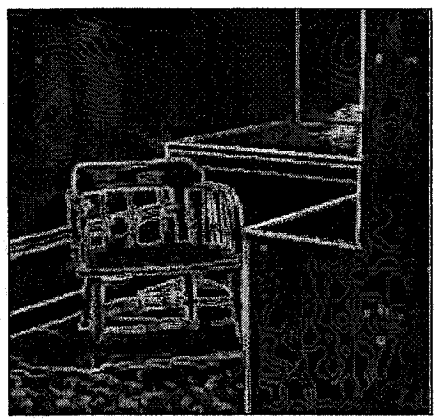
Figure 3.38: Results of IAD on 2D images.



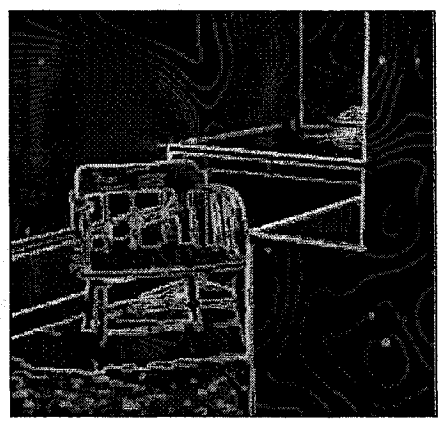
(a) Original image



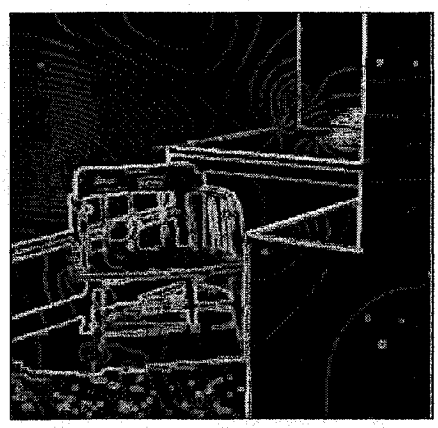
(b) After 10 iterations



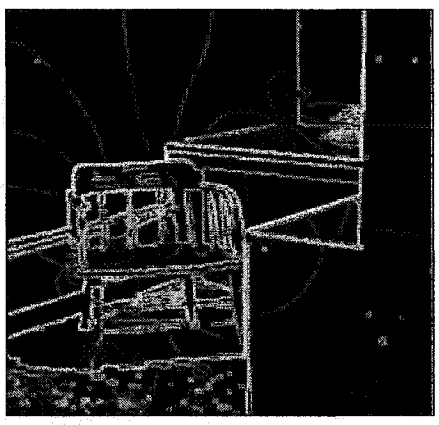
(c) After 10^2 iterations



(d) After 10^3 iterations



(e) After 10^4 iterations



(f) After 10^5 iterations

Figure 3.39: Gradients resulting from the IAD on 2D images.

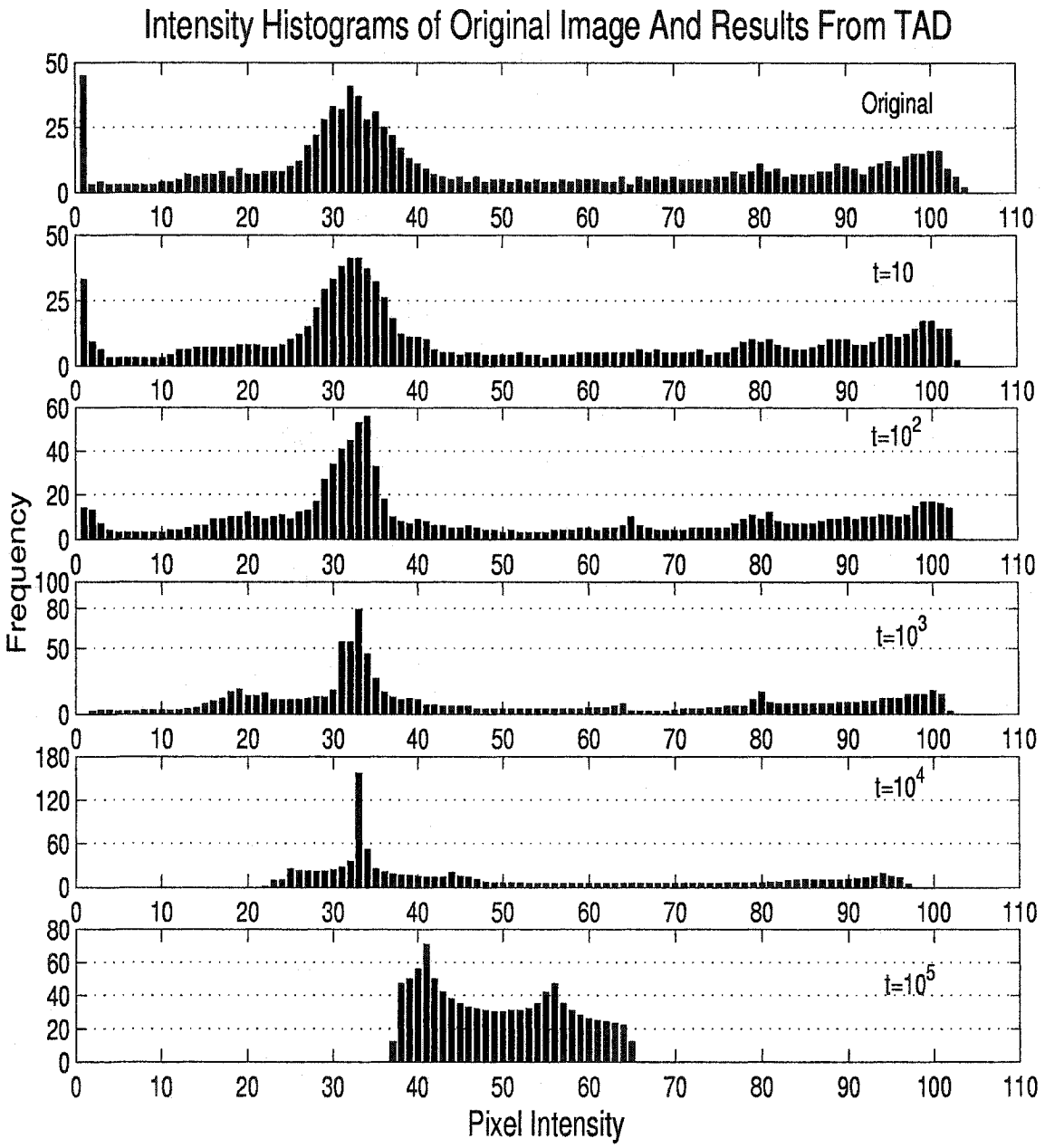


Figure 3.40: Intensity Histogram of Image Results from TAD.

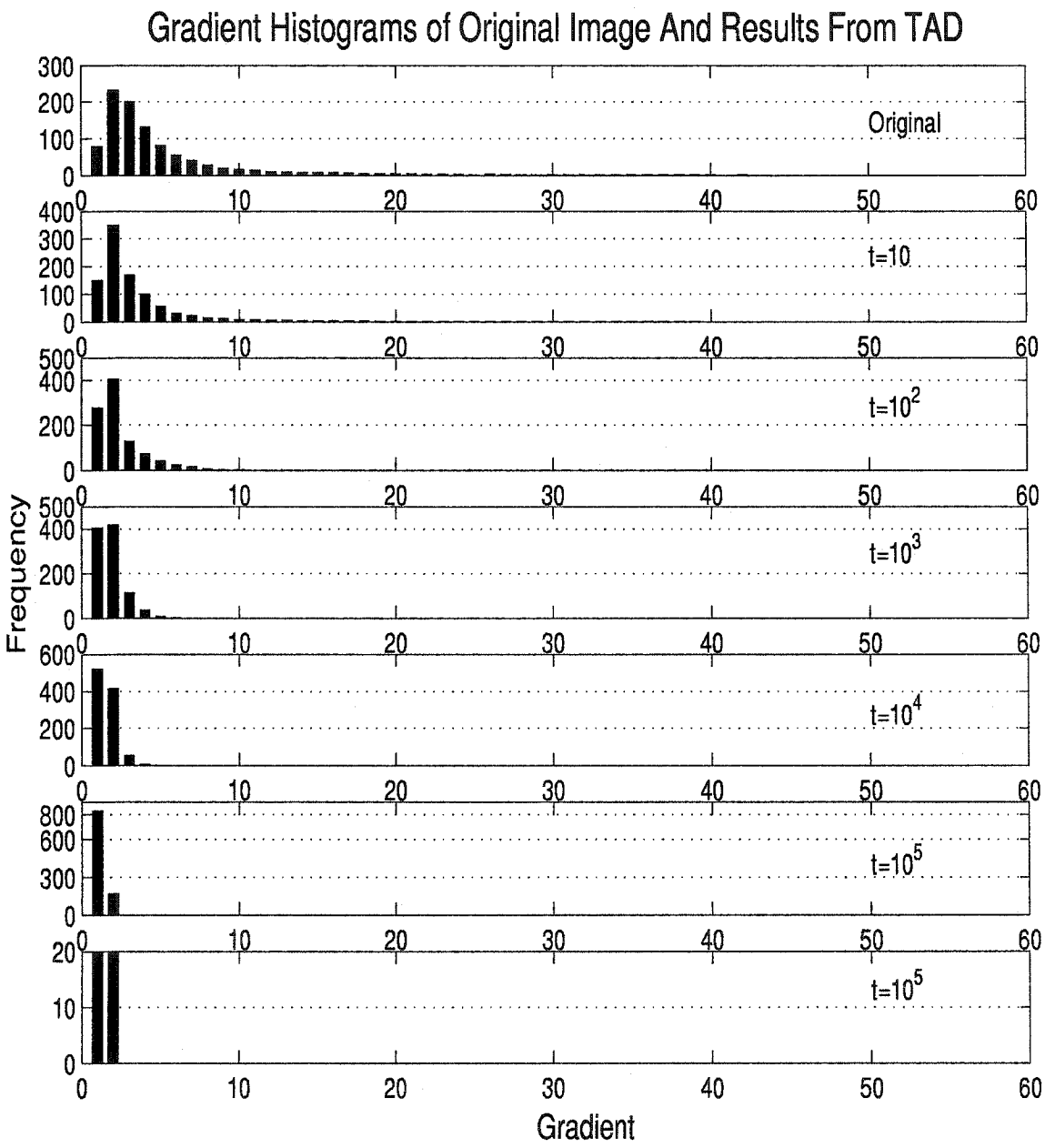


Figure 3.41: Gradient Histogram of Image Results from TAD.

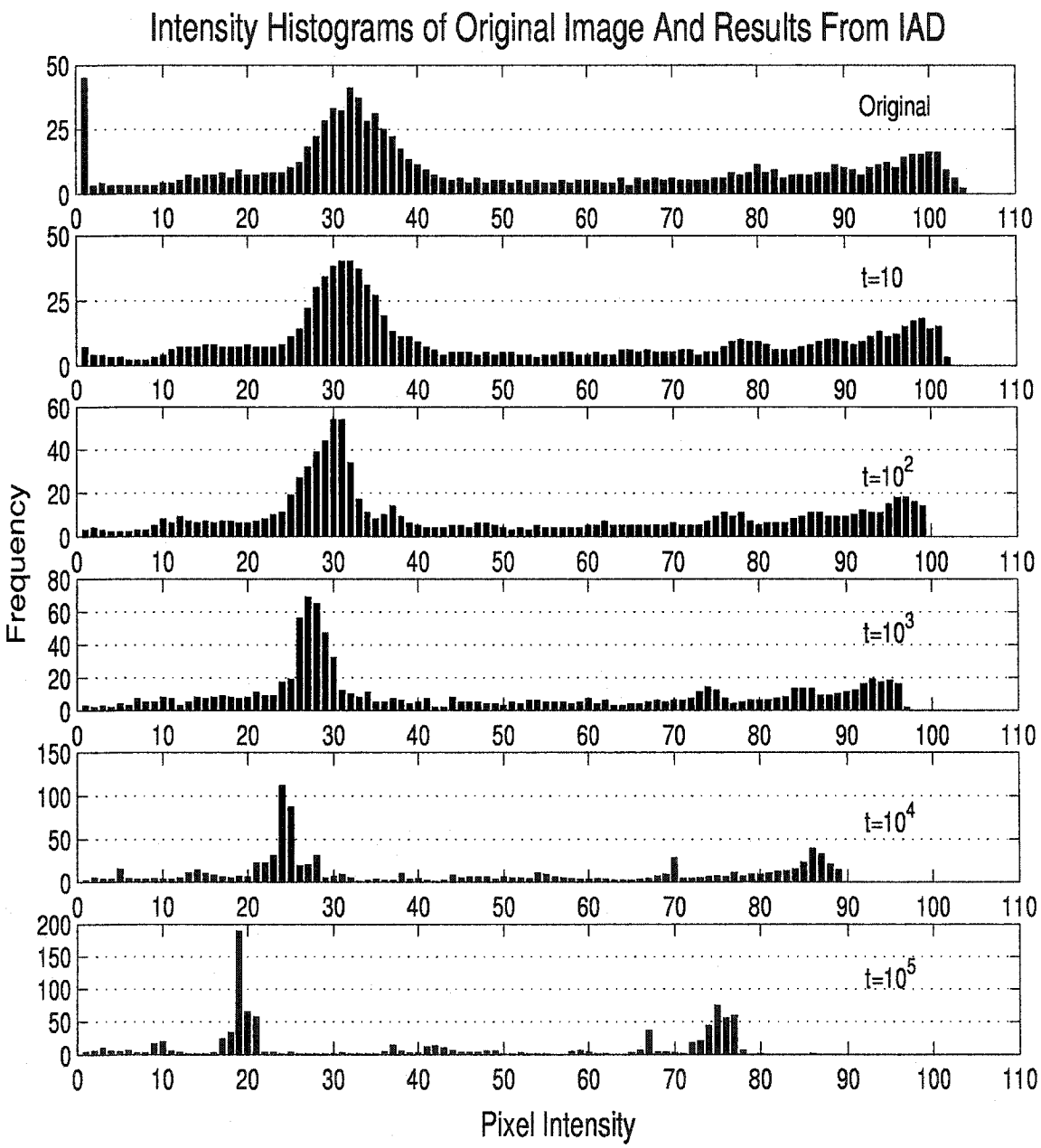


Figure 3.42: Intensity Histogram of Image Results from IAD.

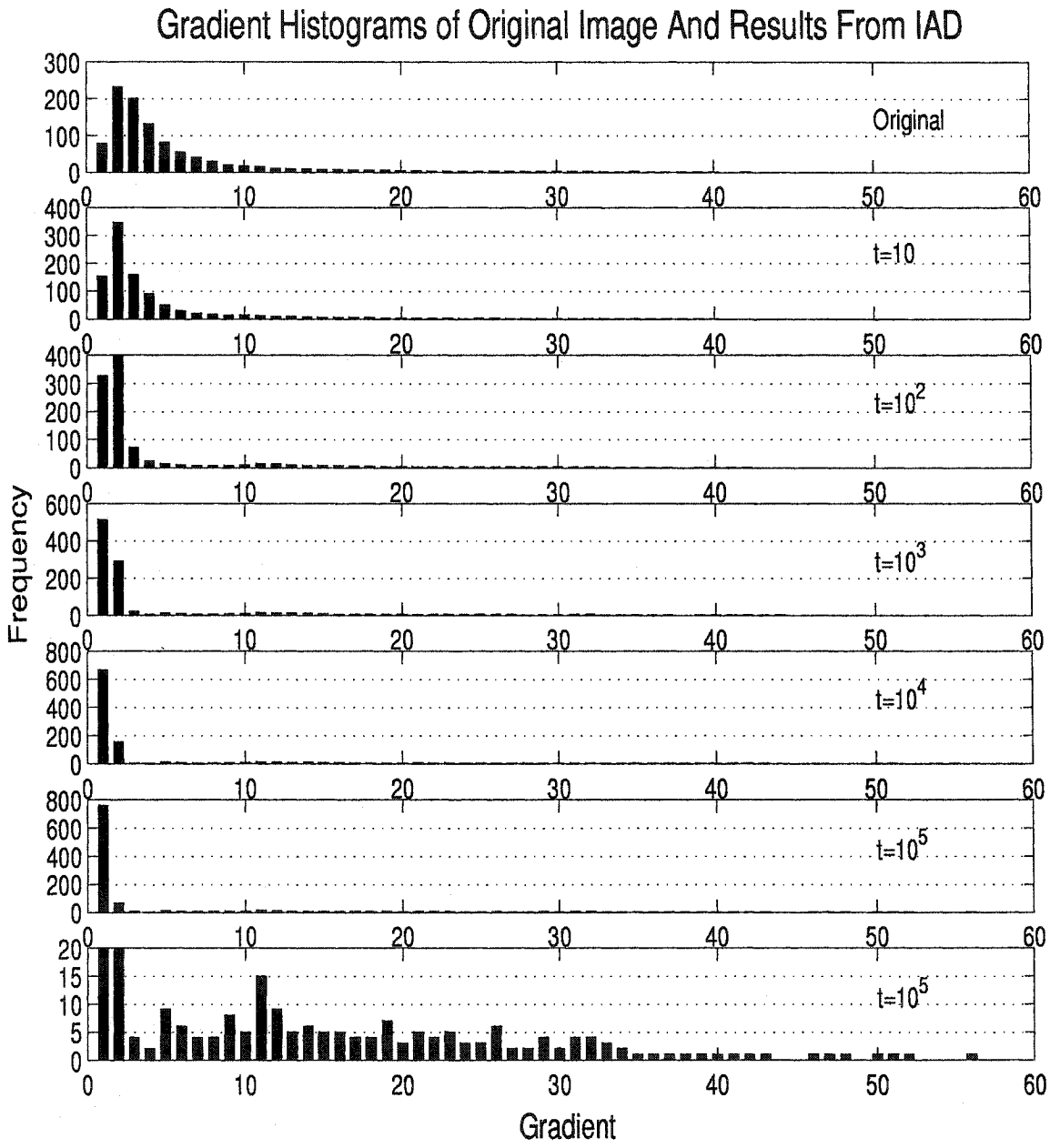
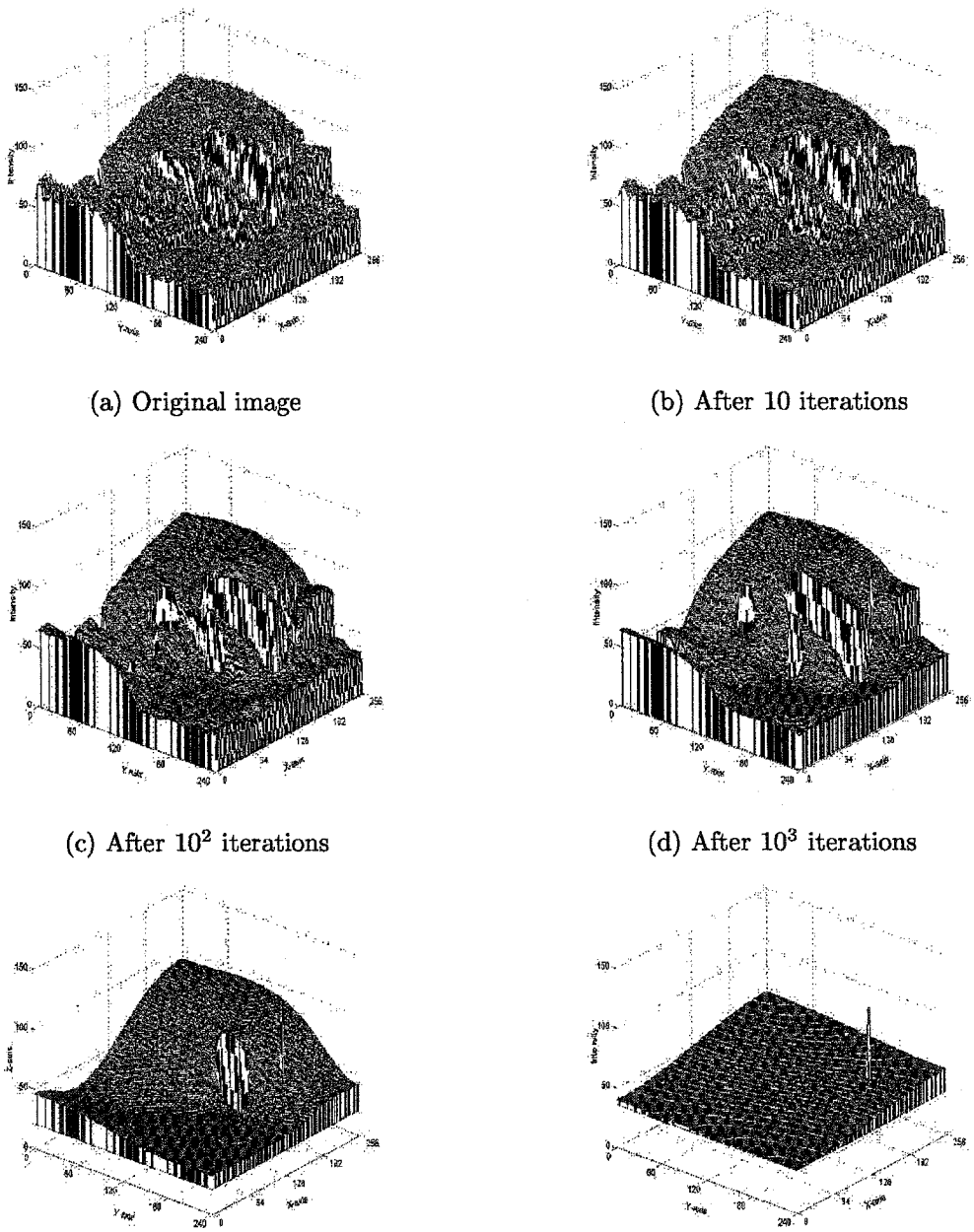


Figure 3.43: Gradient Histogram of Image Results from IAD.



(a) Original image

(b) After 10 iterations

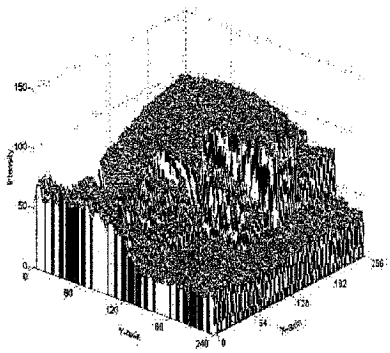
(c) After 10^2 iterations

(d) After 10^3 iterations

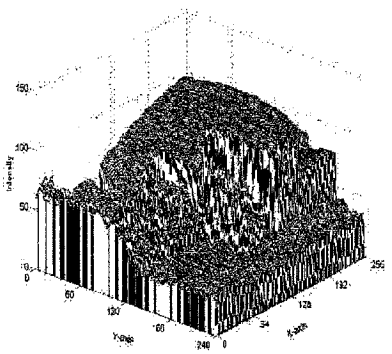
(e) After 10^4 iterations

(f) After 10^5 iterations

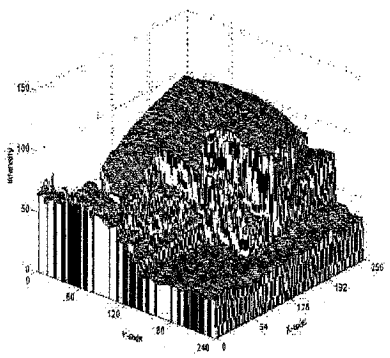
Figure 3.44: Three-dimensional view of resulted intensities by TAD.



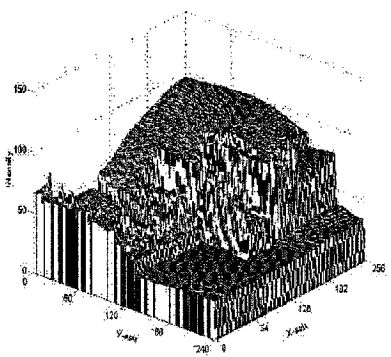
(a) Original image



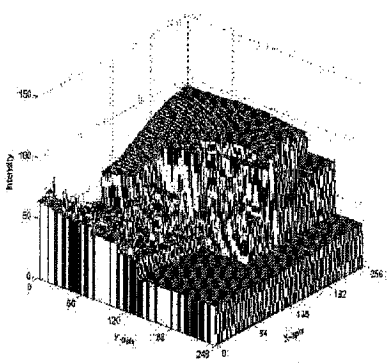
(b) After 10 iterations



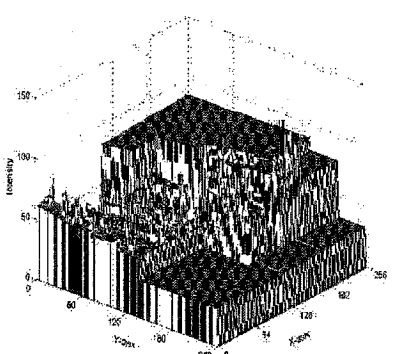
(c) After 10^2 iterations



(d) After 10^3 iterations



(e) After 10^4 iterations



(f) After 10^5 iterations

Figure 3.45: Three-dimensional view of resulted intensities by IAD.

Chapter 4

Idempotent, Direction-Consistent AD

Edge detection is the first step in discovering information from an image. Edges are very important for estimating the structure and properties of objects and for identifying their features in images. Edge detection has continued to be an active research field in image processing.

Edges indicate areas in an image where there are significant local intensity changes. In image processing, an image can been regarded as composed of regions of different intensity levels. A good edge detector is expected to be able to locate only the boundaries between the different regions as edges. Of course, it is unreasonable to expect that an edge detector does not make any wrong decisions. An edge found by an edge detector may be a true edge or a false edge. Also, an edge in an image could be detected or missed by an edge detector. A good edge detector should have a high probability of making correct detections and a low probability of making false detections, even when images are very noisy.

4.1 Analysis of Gradient-Magnitude-Based Anisotropic Diffusion

K plays a very important role in conducting the diffusion process. It determines the fate of every pixel of the processed images. The problem of choosing a suitable value of K to guide the diffusion process has remained open since the concept of AD was introduced in 1987.

4.1.1 Why Most AD Methods Detect Edge by Gradient

Among the research on AD, nearly all approaches use gradient magnitude for estimating edge locations to conduct the smoothing process. This is because:

1. Traditionally, edges have been regarded as discontinuities in gradient magnitude or sharp local change in image intensity. Detecting edges by gradient magnitude seems more straightforward and easier to understand. Typically edge detectors have been developed for measuring peak values in the first derivative, or sometimes zero-crossing in the second derivative. Published literature has concentrated on discussion of gradient-magnitude-based edge detectors (GMEE) with methods employing edge direction almost always ignored [44, 88, 113, 108, 142, 152, 176].

2. In digital image processing, edge detection usually proceeds by first convolving the image with convolution masks. These masks approximate the first or the second derivative operation. They are easy to implement. If edge detection techniques that are not based on gradient magnitude require more calculation, they meet with resistance. Thus, though it is known that in principle it is difficult to develop a GMEE not susceptible to the effects of noise, new edge detection methods have been sought, but most are still based on gradient

magnitude.

4.1.2 The Difficulty of Choosing K

Determining the threshold K is the dominant problem for implementing AD since the stopping problem was solved by the proposed IAD technique. What makes the determination of a suitable value for K so difficult?

Noise is one of the major problems that make it so hard to find a suitable value for the threshold K. Image filters based only on pixels' gradient magnitudes do not distinguish noise from meaningful edges. High gradient noise, if its gradient magnitude reaches the threshold defined for edges, will be treated in the same way as edges. It will be preserved or even enhanced throughout the diffusion process. Such gradient-magnitude-based smoothing techniques are susceptible to the effects of noise and tend to generate erroneous edge information. Unfortunately, this drawback is a problem intrinsic to GMEE.

Another factor that has an immediate influence on the threshold is the change of intensity. Changes of intensity have a direct influence on the value of the threshold. The threshold scales with intensity. For example, we are required to process a group of photos that have been taken of the same scene but at different times, such as in the morning and in the afternoon. We have to choose different thresholds carefully for every photo because of its unique illumination intensity. This can easily be shown by equation (2.3), which is is rewritten here for convenience:

$$g(\|\nabla I\|) = e^{-(\frac{\|\nabla I\|}{K})^2} \tag{4.1}$$

where:

$\|\nabla I\|$ = gradient magnitude,

K = threshold for conducting the diffusion.

Apparently, a constant K is not able to track changes of illumination intensity.

Determining a suitable threshold is so sensitive to image content that one cannot tell whether a pixel of high gradient magnitude is part of an edge or just noise. *a priori* knowledge is helpful for determining the threshold K for gradient-magnitude-based anisotropic diffusion (GMAD). Even if a threshold is chosen carefully for an image, it is inevitable that large-amplitude noise pixels will be also detected. Requiring some knowledge about the image in advance makes AD unsuitable for implementing many computer vision tasks where it is the machine that must make the decisions. This problem is intrinsic to GMAD and is nearly impossible to solve using smoothing and GMEE, thereby making difficult the implementation of AD techniques in dealing with tasks in computer vision. Obviously, this stubborn problem, inherent to GMAD, is the motivation for a new criterion for the AD technique.

4.1.3 Review of Ways of Choosing K

The most common way of choosing K, by basing it on a percentage of image pixels, is easy to carry out. It provides a simple and effective way for researchers to test their proposed scheme. The main concern is that it is sensitive to the effects of noise. Pixels with high gradient magnitude are all detected as edges although some of them could be due to noise. It also requires knowledge of the number of edge pixels in the image.

Applying a filter to smooth the image before anisotropic diffusion makes the diffusion process less susceptible to noise. This method yields more consistent performance by estimating the gradient of the smoothed image, of course at the cost of implementing these filters and their side effects. For example, by implementing a

Gaussian filter in advance, though it shows an improvement in the performance of edge detection, true edge locations tend to be shifted or distorted, especially when noise is considerable or a Gaussian filter with a large spatial support is employed.

Of methods employing other techniques to assist choosing K, using a histogram is popular, but it is not able to deal with the diverse conditions in real images. In principle, what it makes is a trade-off between detecting edges and non-edges. In other words, the K it determined will unavoidably treat part of noise as edges and part of edges as noise. The problem of sensitivity to noise, together with susceptibleness to changes of illumination intensity, makes the application of GMAD to the tasks of computer vision very problematic.

Incorporating contextual information may be a solution for choosing K. Of these techniques, using edge direction may provide a reliable way to separate edges and noise. The scheme of using edge direction to locate edge regions was presented as early as in 1993 [89]. But its merits have been ignored in most research for edge detectors during the years. The direction-consistency-based edge estimator (DCEE) is more robust to noise than GMEE. It distinguishes noisy areas from areas containing edges by their difference in edge direction consistency. In the next section, our proposed idempotent, direction-consistency-based anisotropic diffusion (IDCAD) exploits the merits of DCEE and shows its advantages over that of GMAD.

4.2 Itempotent, Direction-Consistent Anisotropic Diffusion

4.2.1 Gradient's Attributes

"Edgeness" is a local property of a pixel and its immediate neighborhood. Two attributes can be used to describe "edgeness": gradient magnitude and gradient direction. For two-dimensional images, the gradient is defined as the vector:

$$\begin{aligned} \nabla I_{x,y} &= \begin{bmatrix} G_x \\ G_y \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \end{aligned} \tag{4.2}$$

with gradient magnitude:

$$|\nabla I_{x,y}| = \sqrt{G_x^2 + G_y^2} \tag{4.3}$$

and gradient direction:

$$\alpha_{(x,y)} = tan^{-1}\left(\frac{G_y}{G_x}\right) \tag{4.4}$$

where:

I = intensity value of a pixel,

∇ = gradient operator,

x, y = position of pixel.

Gradient-magnitude-based image processing techniques are based on only one of the attributes while our proposed IDCAD presented in the following section exploits both of them.

4.2.2 Direction-Consistency-Based Anisotropic Diffusion

As discussed before, GMEE algorithms have the stubborn problem of selecting a suitable value for the threshold to divide pixels in an image into two groups: uniform regions and edges. Choosing the threshold is image dependent and is often complicated by the presence of noise which is introduced during the process of image capture, image transmission or even image processing. In other words, GMAD cannot been expected to be immune to the effects of noise.

The other attribute of a pixel, the edge direction, carries useful information about that pixel and is very helpful for determining the locations of edges. Edges differ significantly from noise in terms of gradient direction. This can be illustrated with Figure 4.1 which picks out two squares of size 5x5 for explanation. Region "A" is a flat area consisted of noise while region "B" contains an edge. Figure 4.2 shows gradient vectors for both regions. As Figure 4.2(b) shows, in areas immediately surrounding edges, the pixels exhibit high gradient-direction consistency, while in areas of noise as Figure 4.2(a), the pixels show high variation in their gradient directions. As discussed in [89]: "an image edge is a curve through points having attribute gradient magnitudes sufficiently large and gradient directions approximately orthogonal to the tangent to the putative edge curve.". In other words, gradient direction consistency can be used as a new criterion to locate regions containing edges in images. This is the basis of the DCEE approach.

In the following discussion of our proposed IDCAD algorithm, we assume that the signals to be processed are two dimensional images. For every pixel in the image, both the x gradient component and the y gradient component are calculated. They are expressed as a vector for each pixel and are the basis for calculating the direction consistency.

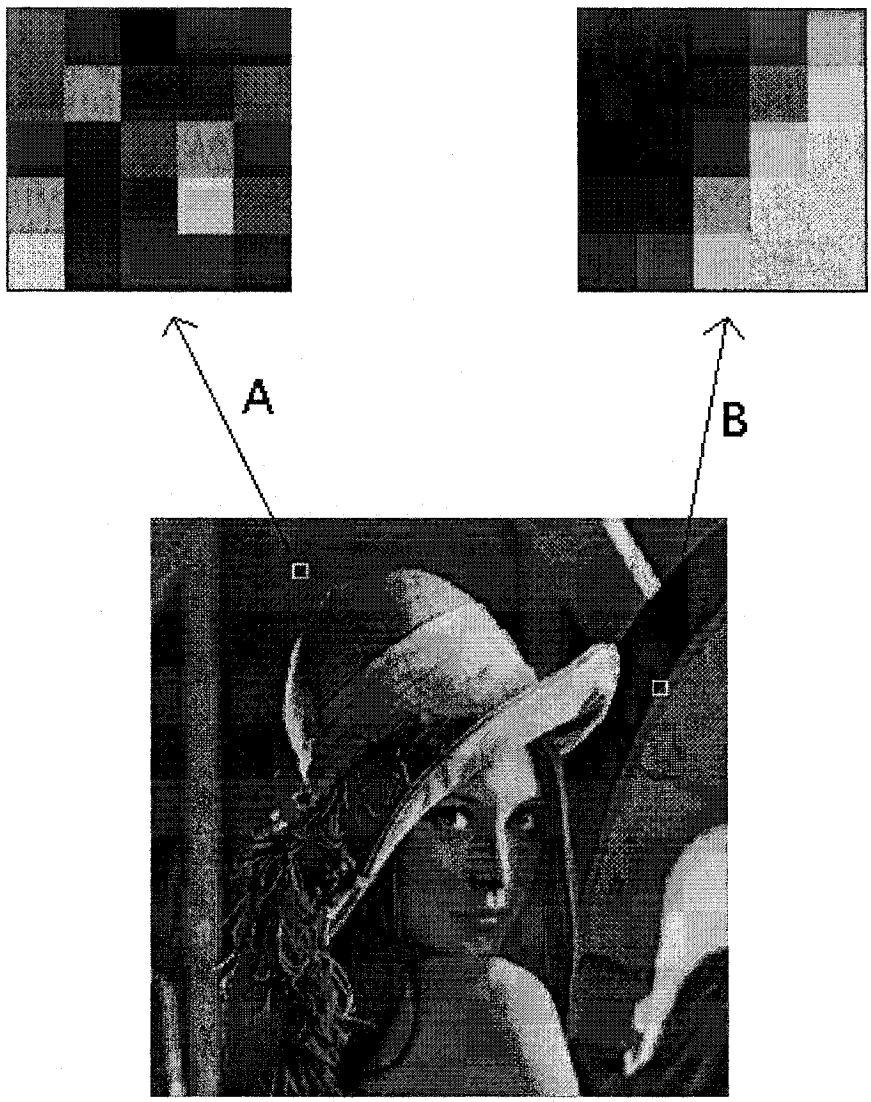
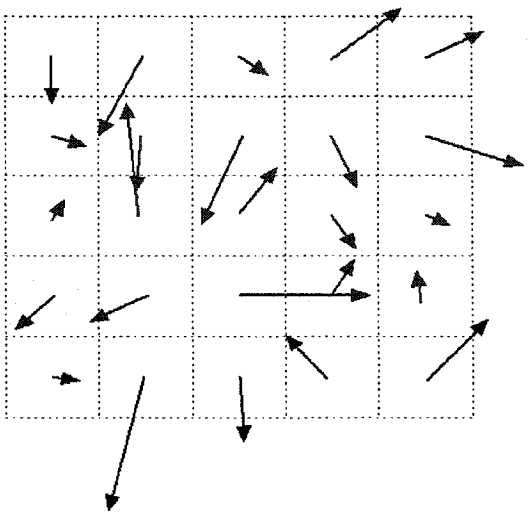
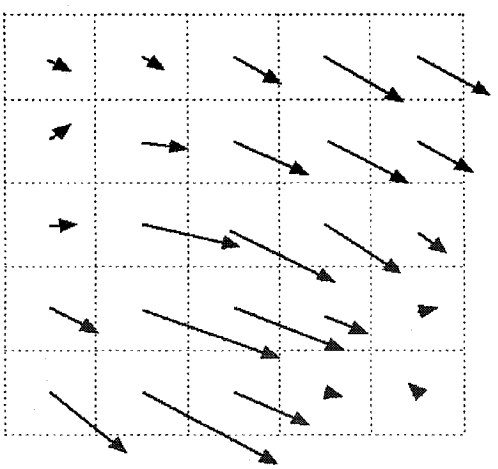


Figure 4.1: Two squares used for showing their differency in edge directions

Note: "A": A noisy area; "B": An area containing an edge



(a) In noisy region "A"



(b) In region "B" containing an edge

Figure 4.2: Gradient vectors

The magnitude of the sum of gradient vectors and the sum of the gradient vector magnitude in a neighborhood of the pixel are as follows, respectively:

$$M1 = \left[\left(\sum_{(i,j)\in n} G_{i,j,h} \right)^2 + \left(\sum_{(i,j)\in n} G_{i,j,v} \right)^2 \right]^{\frac{1}{2}} \tag{4.5}$$

$$M2 = \sum_{(i,j)\in n} \sqrt{G_{i,j,h}^2 + G_{i,j,v}^2} \tag{4.6}$$

where:

i, j = the pixel's position,

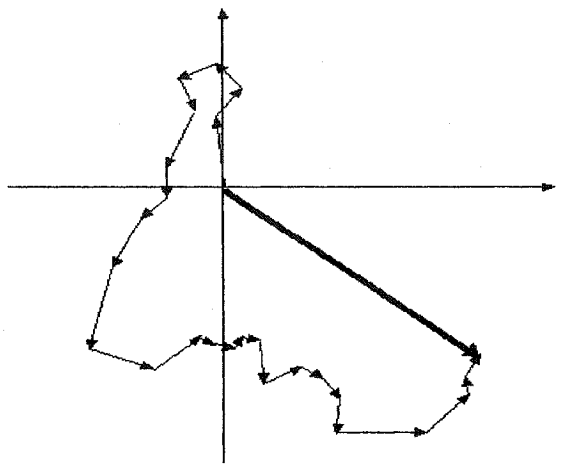
$G_{i,j,h}$ = gradient component in horizontal direction for the pixel (i, j),

$G_{i,j,v}$ = gradient component in vertical direction of the pixel (i, j),

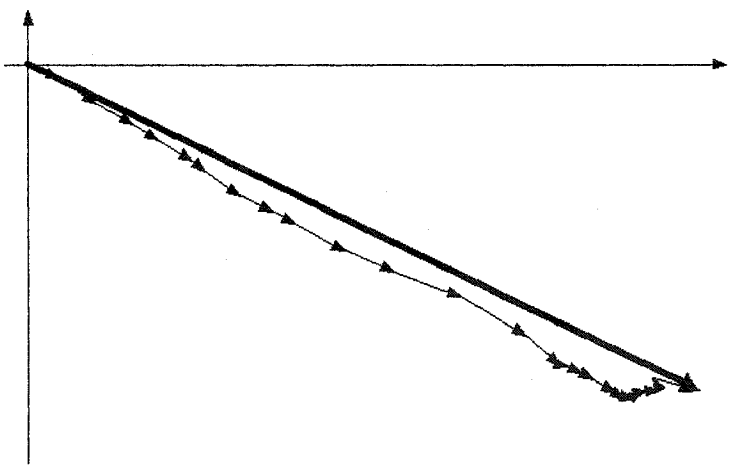
n = spatial neighborhood of pixel (i, j).

The similarity between the magnitude of the sum of gradient vectors and the sum of the gradient vector magnitude can be used as an indicator of direction consistency. If M1 and M2 for pixel (i, j) are similar, this pixel is considered to be within an area of high gradient direction consistency and thus will be treated as a part of an edge. On the other hand, if its M1 is much less than its M2, that pixel will be deemed to be noise due to its low gradient direction consistency, and so it will be smoothed. This is clearly presented in Figure 4.3 which shows that the sum of gradient vectors of the noise region "A" is much different from that of area "B" containing an edge. The sum of gradient vectors is drawn with a thick arrow while the gradient vectors of each square are drawn in thin arrows. Thus for the pixel (i, j), the ratio ρ of M1 to M2 can be employed to measure the gradient direction consistency within its neighborhood:

$$\rho = \frac{M1}{M2} \tag{4.7}$$



(a) Of the noisy region "A"



(b) Of the region "B" containing an edge

Figure 4.3: Sum of the gradient vectors

Equation (4.7) clearly shows that ρ is totally unaffected by the changes of intensity levels. This makes the use of ρ more realistic for real image processing. It is clear that $0 \leq \rho \leq 1$ since $M1 \leq M2$. Obviously, the larger ρ is, the higher the possibility that the pixel is part of an edge. It is unlikely that high gradient magnitude noise will be treated as part of an edge due to its small value of ρ. Based on edge direction consistency, DCEE shows much better performance than those strategies based on gradient magnitude because of its immunity to changes of intensity and low sensitivity to noise. Using DCEE, we can construct a robust anisotropic diffusion technique based on edge direction consistency. It is much better than using the gradient magnitude as the indicator of an edge region because it reflects directly the spatial characteristics of real edges.

Based on the regions of high ρ value, smoothing is implemented using IAD with the DC function as follows.

$$c(x, y, t) = e^{-(\frac{\rho}{K})^2} \tag{4.8}$$

where:

$c(x, y, t)$ = the DC function controlling the smoothing process,

K = threshold for edges.

The proposed IDCAD strategy has been developed by combining the merits of DCEE and those of the IAD technique. DCEE shows much improved performance in locating edge regions and being robust to noise. It conducts IAD for removing noise and preserving edges. Although DCEE makes choosing K much easier, the size of the region for consistency estimation must be selected. The IAD technique, though it belongs to the group of GMAD algorithms, prevents over-blurring and keeps the meaningful features of images throughout the diffusion process. IAD is applied to noisy regions detected by DCEE.

4.3 Experiments

4.3.1 Source of Test images

Experiments are carried out on two real images: "lena" and "cameraman" which are widely used in the field of image processing. Image "lena" is used as a standard for image processing techniques [150]. It is used to test various image processing algorithms due to its mixture of features: straight lines, curves, shading, bright regions, flat regions, texture regions, strong edges, weak edges and more.

IAD, based on the gradient magnitude edge estimator, is used to draw a comparison between IDCAD and GMAD. Models of noise commonly used in images are white noise, Gaussian noise, impulse noise and salt-and-pepper noise. Here we have chosen Gaussian noise for our experiments because Gaussian noise "is a very good model for many kinds of sensor noise, such as the noise due to camera electronics" [108] and "Gaussian noise is a very good approximation to noise that occurs in many practical cases." [208].

4.3.2 Simulation Functions

For digital images, masks provide a practical way of approximation to the derivatives. For the gradient magnitudes, standard edge-detector masks include Roberts operators, Sobel operators, Prewitt operators, Frei-Chen operators, etc. Masks of size 3x3 are better in dealing with noise and more convenient in implementation than masks of size 2x2. Of 3x3 edge detectors, Sobel masks is used for our experiments due to their convenience of calculation.

The performances of GMAD and IDCAD filters are compared in the experiments. Floating-point simulations have been implemented with the following simulation

$$Gx = \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad Gy = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

Figure 4.4: 3x3 Sobel operators.

functions:

GMAD filter	Using equations (4.1), (3.89), (2.9) and (2.10)
IDCAD filter	Using equations (3.89) and (4.5) - (4.8)

4.3.3 Analysis of Experimental Results

Figure 4.5 contains the original images and gradient images obtained with the Sobel operator. Figure 4.6 shows the images after adding Gaussian noise of $mean = 10$ and $\sigma = 15$. Here we should note that IDCAD uses the same set of parameters throughout the experiments on both original images and noisy images, while for GMAD the value of the threshold had to be adjusted for every image to achieve a good result. Figure 4.7 presents the regions located by direction consistency. They act as indicators for conducting smoothing process. Experimental results on both images show the same conclusion. Thus for simplicity, we use the results on image "lena" for our explanation.

As for the original image, the GMAD technique implements anisotropic diffusion based on the gradient magnitudes shown in Figure 4.5(b). The IDCAD technique uses a map of regions located by DCEE shown in Figure 4.7(a) to control the diffu-

sion process. Despite a few strong noise pixels, both of the subfigures, Figure 4.5(b) and Figure 4.7(a), locate the boundaries of regions properly. If a suitable threshold is chosen carefully for GMAD, we can expect that both GMAD and IDCAD produce good results. This can be seen in results Figures 4.9 and 4.8. The feather is much more smoothed in Figure 4.9 then in Figure 4.8 because of its high gradient magnitudes but low direction consistency. Note that several spots caused by strong noise pixels remain in the resultant images from GMAD as the gradient images in Figure 4.8 show. Figures 4.11 and 4.10 shows this difference clearly.

The difference becomes more obviously in the noisy images. By comparing Figure 4.5(b) with Figure 4.6(b), we note that some edges are damaged by noise. This means that noise regions with high gradient magnitudes will inevitably be treated as edges by GMAD. As Figure 4.12 shows, GMAD is unable to smooth the high gradient magnitude noise, and removes part of the edges damaged by noise. Due to its low direction consistency, noise does not produce such a significant effect on edges estimated with DCEE, even for high gradient magnitude noise. In Figure 4.7(a) and (b), it is seen that edges remain complete and clear for both the original and noisy images due to their high direction consistency. Figure 4.13 shows that the results of IDCAD were almost completely insensitive to the added noise. This shows that edge direction consistency is a better edge estimator than gradient magnitude.

Finally, it is worthwhile to emphasize the improvement in robustness to noise made by IDCAD. IDCAD uses the same set of parameters throughout all experiments on all images while GMAD had to have the value for the threshold adjusted for each test image for good results. IDCAD shows its advantages over GMAD through its low sensitivity to noise.

4.3.4 Experiment Results

(a) Original image "lena"

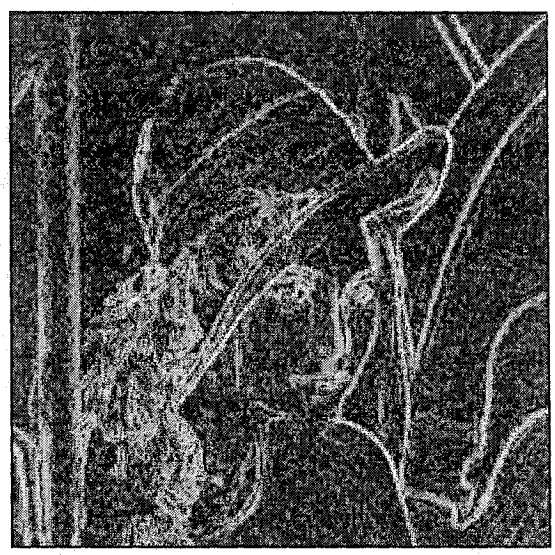
(b) Gradient magnitude of (a)

(c) Original image "Cameraman"

(d) Gradient magnitude of (c)

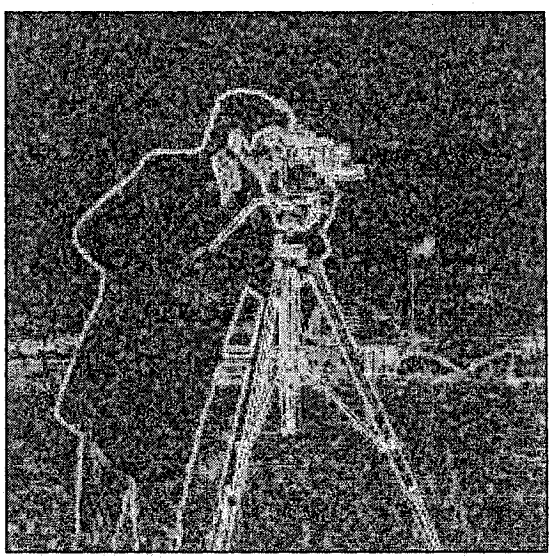
Figure 4.5: Original images for the experiments

(a) Noisy image "lena"



(b) Gradient magnitude of (a)

(c) Noisy image "cameraman"

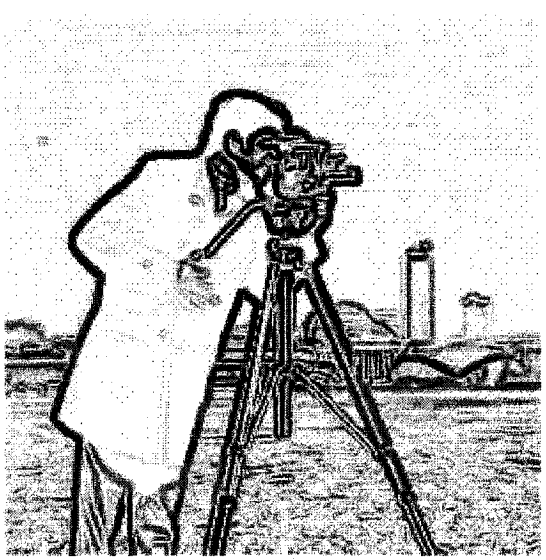


(d) Gradient magnitude of (c)

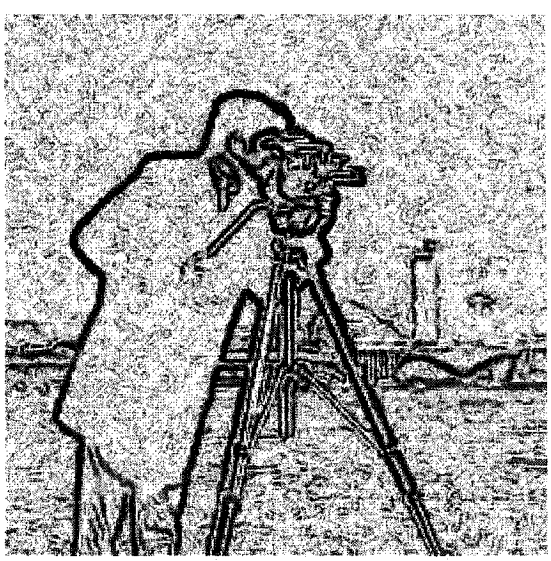
Figure 4.6: Noisy images for the experiments

(a) For original image "lena"

(b) For noisy image "lena"



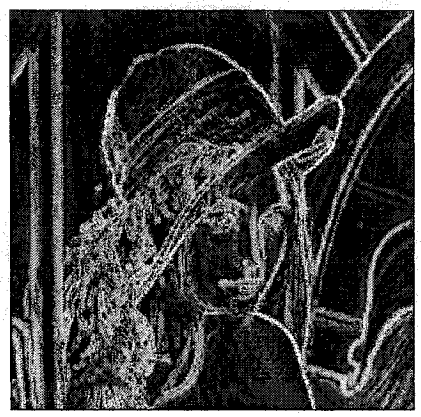
(c) For original image "cameraman"



(d) For noisy image "cameraman"

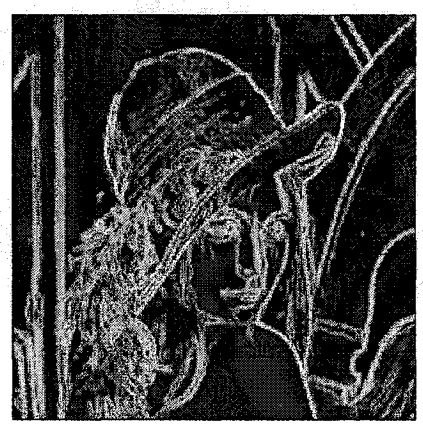
Figure 4.7: DC maps estimated by DCEE

(a) At iteration 10



(b) Gradient magnitude of (a)

(c) At iteration 10^2



(d) Gradient magnitude of (c)

(e) At iteration 10^3

(f) Gradient magnitude of (e)

Figure 4.8: Experiment results obtained by GMAD on original image "lena".

(a) At iteration 10

(b) Gradient magnitude of (a)

(c) At iteration 10^2

(d) Gradient magnitude of (c)

(e) At iteration 10^3

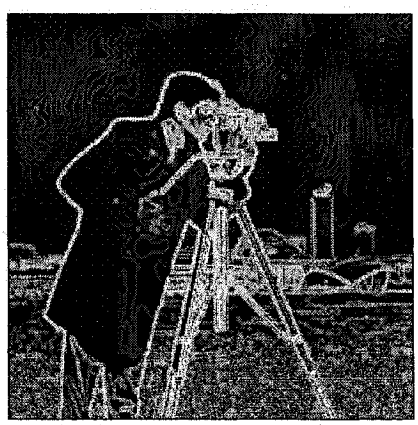
(f) Gradient magnitude of (e)

Figure 4.9: Image and gradient results obtained by IDCAD on original image "lena".

(a) At iteration 10

(b) Gradient magnitude of (a)

(c) At iteration 10^2



(d) Gradient magnitude of (c)

(e) At iteration 10^3

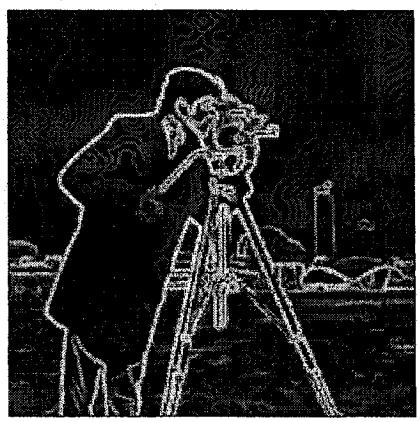
(f) Gradient magnitude of (e)

Figure 4.10: Experiment results obtained by GMAD on original image "cameraman".

(a) At iteration 10

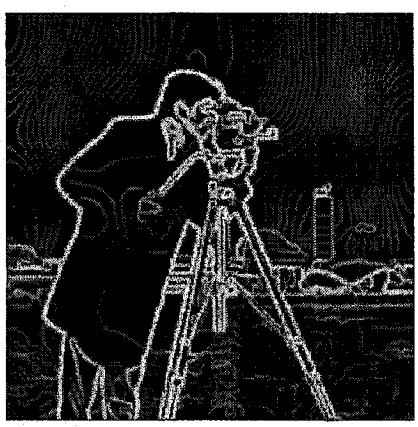
(b) Gradient magnitude of (a)

(c) At iteration 10^2



(d) Gradient magnitude of (c)

(e) At iteration 10^3



(f) Gradient magnitude of (e)

Figure 4.11: Experiment results obtained by IDCAD on original image "cameraman".

(a) At iteration 10

(b) Gradient magnitude of (a)

(c) At iteration 10^2

(d) Gradient magnitude of (c)

(e) At iteration 10^3

(f) Gradient magnitude of (e)

Figure 4.12: Experiment results obtained by GMAD on noisy image "lena".

(a) At iteration 10

(b) Gradient magnitude of (a)

(c) At iteration 10^2

(d) Gradient magnitude of (c)

(e) At iteration 10^3

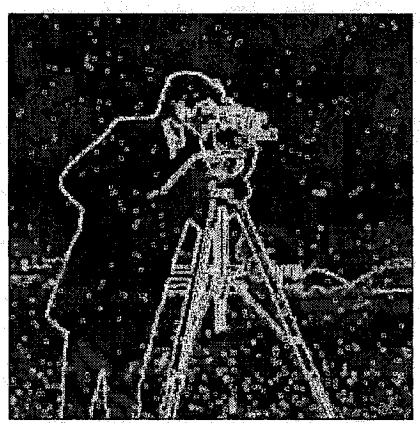
(f) Gradient magnitude of (e)

Figure 4.13: Experiment results obtained by IDCAD on noisy image "lena".

(a) At iteration 10

(b) Gradient magnitude of (a)

(c) At iteration 10^2



(d) Gradient magnitude of (c)

(e) At iteration 10^3

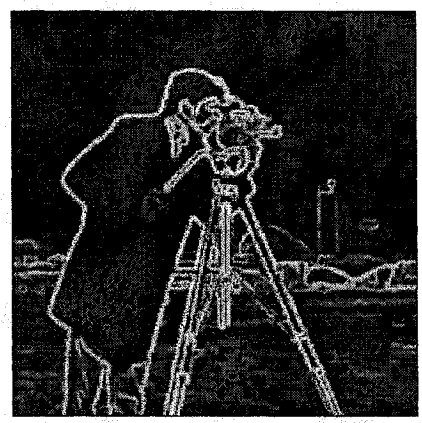
(f) Gradient magnitude of (e)

Figure 4.14: Experiment results obtained by GMAD on noisy image "cameraman".

(a) At iteration 10

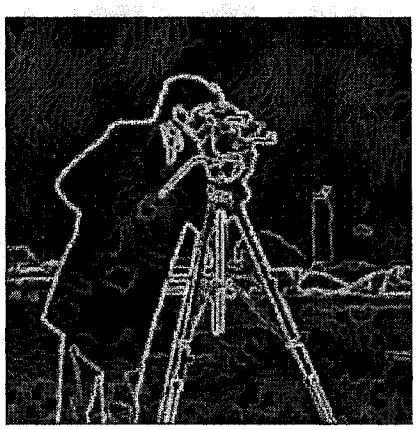
(b) Gradient magnitude of (a)

(c) At iteration 10^2



(d) Gradient magnitude of (c)

(e) At iteration 10^3



(f) Gradient magnitude of (e)

Figure 4.15: Experiment results obtained by IDCAD on noisy image "cameraman".

Chapter 5

Conclusion and Discussion

5.1 Conclusion

In this thesis, we proposed a novel AD technique, idempotent, direction-consistent anisotropic diffusion. It consists of solutions to two of the most critical problems that remained unsolved in the field of AD research since the AD theory was introduced in 1987. Our first proposal, the IAD technique, is a new interpretation of AD. The challenging problem of over-oversmoothing is prevented, thereby making the experimental results agree with the AD theory for the first time since 1987. Our second proposal, the IDCAD technique, creates a new criterion for implementing AD. It provides a solution to the stubborn problem of choosing a suitable value as the threshold and makes AD much more robust to noise.

To develop the IAD strategy, our research has been carried out on AD in both the continuous domain and the discrete domain. Unlike current AD research that is mainly based on the CF, a unique viewpoint is proposed in our study in which AD behavior is derived by the flux. We show for the first time in both the continuous domain and the discrete domain that the DF should be employed to control the smoothing strength. The importance of a threshold for preserving the edges

throughout the diffusion process is emphasized for the first time since 1987. Our technique has the following significant distinguishing features.

- It agrees with AD theory with a desirable combination of forward smoothing implemented in noise regions and zero/backward smoothing carried out on edges.
- It solves the difficult problem of determining a stopping criterion with its idempotent behavior. Thus there is no need to worry about over-blurring since semantically meaningful features will be kept throughout the diffusion process.
- A side-effect of using a negative DC is avoided with the "Maximum Principle" being obeyed.
- The simple algorithm allows for implementation in highly parallel VLSI.

We analyzed the TAD technique and showed that it does not preserve or enhance edges, but continually smooths the image. The result is a meaningless image. Experiments carried out on both 1D signals and 2D images show that the proposed IAD technique has the desirable attributes of preserving edges, enhancing edges, removing noise and flattening trivial details. In contrast, TAD filters continually smooth until a uniform grey image is produced.

In our second proposal, the IDCAD technique, gradient direction consistency, a largely unused but valuable attribute of pixels, has been exploited in conducting the diffusion process. It improves the performance of the IAD technique by distinguishing edges from high-gradient noise. It is based on the idea that edges and noise behave differently with respect to gradient direction consistency. Our proposed IDCAD implements smoothing based on regions located by using the consistency of gradient direction as an indicator of an edge region. This approach reflects directly

the spatial characteristics of real edges and is much better than operators using only the gradient magnitude as the indicator of an edge. Regions likely to contain edges will be preserved and the other regions will be smoothed. This is different from GMAD in which the exact positions of edges are required.

Choosing a suitable threshold to conduct AD is a stubborn problem inherent to the AD technique based on GMEE. Even if a threshold is determined carefully, certain kinds of high gradient magnitude noise will be treated inevitably as edges by GMAD techniques. Our proposed IDCAD conducted by DCEE avoids this problem. Noise, even with a high gradient magnitude, will be smoothed by IDCAD because it does not exhibit high direction consistency.

Besides the benefits that the AD technique brings to the field of image processing, our final goal is to implement AD in real time image processing (RTIP). Our proposed idempotent, direction-consistent AD combines the merits of IAD and those of DCEE. This technique distinguishes itself from common AD techniques in the following ways:

1. Meaningful features of images will be kept throughout the diffusion process. This quality of not over-blurring is very useful, especially when AD is employed in hardware for real time image processing.

2. An edge estimator based on direction consistency has immunity from changes of intensity levels. One application, for example, is the monitoring of scenes of interest, where the camera captures the same scene but with different illumination intensity levels. It is very hard for GMAD to deal with such tasks because of the need for adjusting the value of the threshold. Also, DCEE is much less susceptible to the effects of noise than GMEE.

These merits make it realistic to conduct AD in real time image processing.

In the end, we would prefer using our experimental results over writing to summarize our contributions to the field of AD research. Figure 5.1 shows the improvements in image results, their corresponding gradient results and the DC maps used to conduct the AD. Experimental results are all taken at $2x10^3$ iterations. The DC map used to conduct the GMAD comes from a threshold that regards 95% percent of pixels in the image as noise.

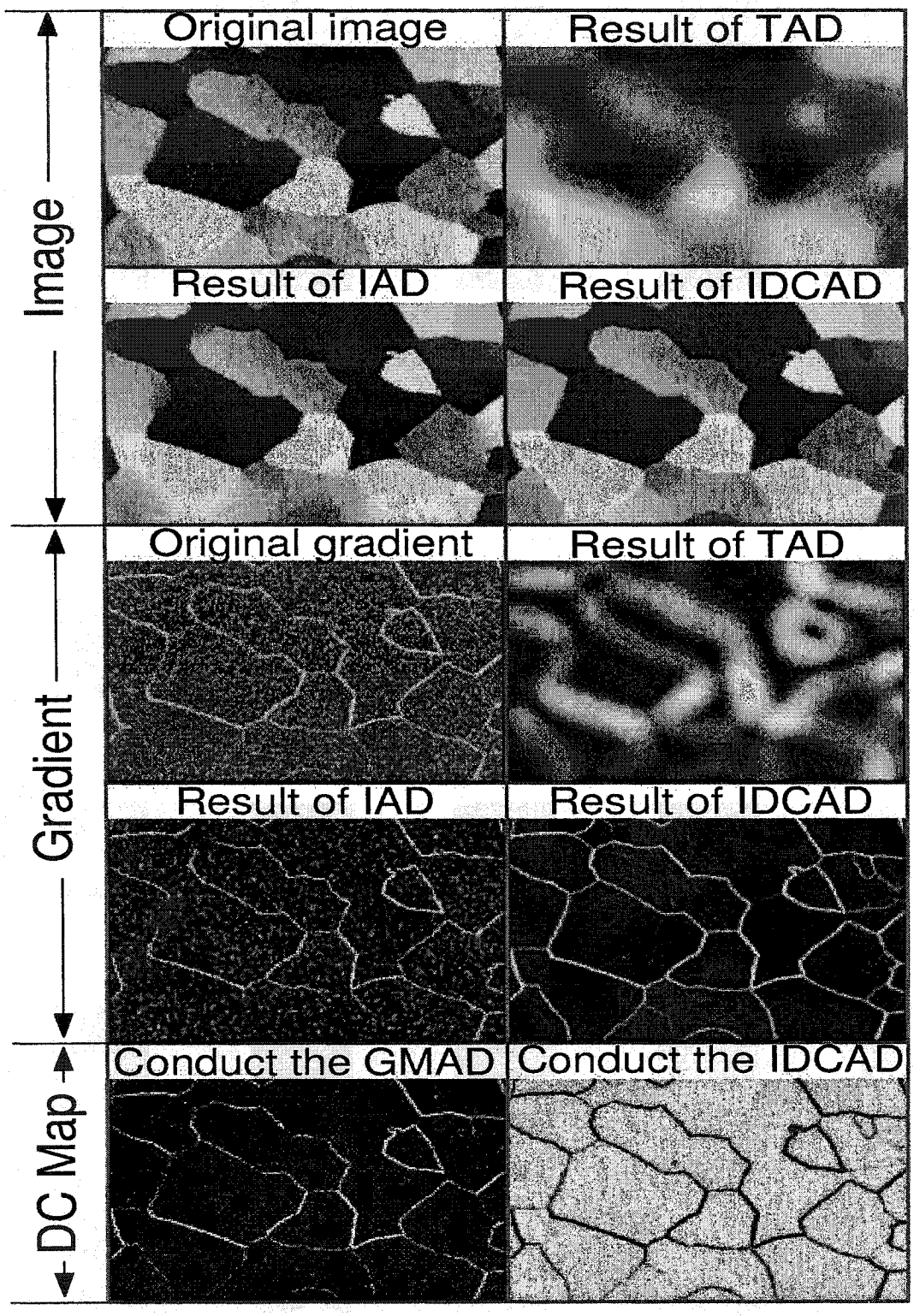


Figure 5.1: Summary of contributions of this thesis.

5.2 Future Work

Implementing AD for dealing with tasks of computer vision has shown promise with the proposed solutions to its remained major problems. However, it is still a long way from practical application. More research is needed in the following areas.

5.2.1 Improving the performance of IDCAD

The basic DCEE is not good at detecting corners or noise areas with approximately the same edge directions. The corner, for example, behaves much like noise in edge direction. Corners act as one of the basic elements for image understanding and pattern recognition. Considerable research has been conducted on detecting corners due to the essential information that they bear. A thorough study of the available strategies for detecting corners will be helpful to find a way to deal with the problem, thus improving the performance of IDCAD.

A flat area with edge direction in approximately the same direction is another problem of DCEE. This could generate incorrect edge indications. Though it is similar to regions containing edges in gradient direction consistency, noisy regions behave much different from regions containing edges in gradient magnitude. For example, it is unlikely for a flat area to have a local gradient magnitude peak. The difference in gradient magnitude between noisy regions and regions containing edges is a valuable means of solving this problem.

The IDCAD algorithm is more complex than that of GMAD. This is because it requires that the contextual information be included when making an edge decision. We may have to accept this fact since IDCAD makes it more realistic to apply AD to tasks in machine vision and RTIP. Of course this does not mean that there is no

simpler way to realize the IDCAD strategy. We expect the performance of DCEE and IDCAD to be improved continually with time.

5.2.2 Implementing AD in VLSI - The Next Hot Topic

The most desirable advantage of AD technique has not been exploited completely if it is not combined with VLSI. The ideal way is to construct a system-on-a-chip. This covers several research areas: image acquisition techniques, image processing techniques, electrical circuit theory and VLSI.

CMOS imagers and charge coupled device (CCD) imagers are the two most popular solid-state image sensors constructed from silicon [239]. Both sense light in the same way by converting incident photons into electronic charges. CMOS image sensors are more suitable for implementing AD than the CCD image sensor for the following reasons:

1. One significant advantage of CMOS sensors over CCD sensors is their ability to have the desired functions integrated onto a single chip. This makes CMOS sensors more suitable for "smart" cameras that implement image sensing and image processing together.

2. Another advantage of CMOS sensors over CCD sensors is that CCD sensors need a clock with strict clock amplitude and shape. This has the drawbacks of specialized clock driver circuits, multiple supply voltage, high power dissipation and hence high cost.

3. CMOS sensors use the basic VLSI technology while CCD sensors need specialized manufacturing processes.

4. The performance of CMOS sensors has been improved continually and are acceptable in practice. For example, fixed-pattern noise (FPN) used to be a

problem of CMOS images [72]. It is produced by uniform/zero illumination and is fixed in position in the generated image. Solutions have been proposed for it [110, 158, 172], and it has already been reduced to acceptable levels [61, 100].

Camera-on-a-chip products based on CMOS sensor are already available. Normally they consist of a lens, photo sensor and related control circuits including functions such as readout, signal processing, analog-to-digital conversion and interfacing, etc. Understanding of research in this field [73, 118, 157, 160] is useful for the implementation of AD in hardware.

Our goal is a camera-on-a-chip with the ability to remove noise and preserve and enhance edges, and to output essential information for subsequent high-level image processing at low cost and low energy dissipation. Such a system combines image sensing and image processing intimately.

New theory needs to be developed for constructing such a system-on-a-chip implementing AD. It must have low energy dissipation with high calculation effectiveness. Current image processing operations are normally carried out in the digital domain. These digital strategies are energy-hungry. Mead [143] noticed the fact that a transistor is orders of magnitude more effective than a digital computer/circuit in the energy cost for doing an operation. In digital schemes, normally much more than one transistor is needed. Not only do these transistors consume energy, but also the traces on the chip need energy to be charged/discharged.

Another factor that makes a difference in energy consumption is the method of operation. Making algorithms more local could reduce the power dissipation. Biological image sensor systems deal quite well at low energy dissipation with real time image processing which demands very high computation. Understanding the prin-

ciple of biological image sensor systems will be very helpful due to their superiorities to digital image sensor systems [146].

AD can be implemented in local parallel structures and VLSI provides the way of integrating the desirable functions on a chip. This provides promise of developing smart cameras for dealing with many tasks of computer vision, especially of RTIP at low cost.

Bibliography

[1] Abd-Elmoniem, K.Z., "Feedback coherent anisotropic diffusion for high resolution image enhancement", *Biomedical Imaging, Proceedings. 2002 IEEE International Symposium*, pp. 693 -696, 2002.

[2] Abd-Elmoniem, K.Z., Youssef, A.B.M. and Kadah, Y.M. "Real-time speckle reduction and coherence enhancement in ultrasound imaging via nonlinear anisotropic diffusion", *Biomedical Engineering, IEEE Transactions*, Vol. 49, Issue: 9, pp. 997-1014, Sept. 2002.

[3] Abd-Elmoniem, K.Z., Kadah, Y. M. and Youssef, A.-B.M., "Real time adaptive ultrasound speckle reduction and coherence enhancement", *Image Processing, International Conference*, vol. 1, pp. 172-175, 2000.

[4] Acton, S.T. and Ley, K., "Tracking leukocytes from in vivo video microscopy using morphological anisotropic diffusion", *Image Processing, 2001 International Conference*, vol. 2, pp. 300-303, Oct. 2001.

[5] Acton, S.T., "Locally monotonic diffusion", *Signal Processing, IEEE Transactions*, vol. 48, Issue. 5, pp. 1379 -1389, May 2000.

[6] Acton, S.T., "Anisotropic diffusion and local monotonicity", *Acoustics, Speech and Signal Processing, Proceedings of the 1998 IEEE International Conference*, vol. 3, pp. 1293-1296, 1998.

[7] Acton, S.T., "Multigrid anisotropic diffusion", *Image Processing, IEEE Transactions*, vol. 7, Issue: 3, pp. 280-291, March 1998.

[8] Acton, S.T., "A PDE technique for generating locally monotonic images", *Image Processing, ICIP 98, Proceedings, 1998 International Conference*, vol.3, pp. 269-273, 1998.

[9] Acton, S.T., "A pyramidal edge detector based on anisotropic diffusion", *Acoustics, Speech, and Signal Processing, ICASSP-96, Conference Proceedings, 1996 IEEE International Conference*, vol. 4, pp. 2215 -2218, 1996.

[10] Acton, S.T, "Edge enhancement of infrared imagery by way of the anisotropic diffusion pyramid", *Image Processing, International Conference*, vol. 1, pp. 865-868, 1996.

[11] Acton, S.T., Bovik, A. C. and Crawford, M.M., "Anisotropic diffusion pyramids for image segmentation", *Image Processing, Proceedings, ICIP-94, IEEE International Conference*, vol. 3, pp. 478 -482, 1994.

[12] Acton, S.T. and Bovik, A.C., "Anisotropic edge detection using mean field annealing", *Acoustics, Speech, and Signal Processing, ICASSP-92, 1992 IEEE International Conference*, vol. 2, 1992.

[13] Aja, S., Alberola, C. and Ruiz, J., "Fuzzy anisotropic diffusion for speckle filtering", *Acoustics, Speech, and Signal Processing, 2001 IEEE International Conference*, vol. 2, pp. 1261 -1264, 2001.

[14] Alacron, (2002), "A Real-Time Inspection Application for Fruit", [online], Available: *http://www.alacron.com/index.html*, April 16, 2003.

[15] Alacron, (2002), "Industrial Mask Inspection Using FastChannel", [online], Available: *http://www.beaulieudesign.com/alacron/final/apps/mask.html*, June 16, 2003.

[16] Alvarez, L., Deriche, R. and Santana, F., "Recursivity and PDEs in image processing", *Pattern Recognition, 15th International Conference*, pp. 242-248, vol. 2000.

[17] Alvarez, L., Guichard, F., Lions, P.L. and Morel, J.M.; "Axioms and fundamental equations of image processing", *Arch. Rotional Mechanics*, vol. 123, pp. 200-257, 1993.

[18] Alvarez, L., Lions, P.L. and Morel, J.M., "Image selective smoothing and edge detection by nonlinear diffusion", *SIAM-JNA*, vol. 29, pp. 845-866, 1992.

[19] anafocus, (2002), "Vision Systems on Chip", [online], Available: *http://www.anafocus.com/*, May 4th, 2003.

[20] Ancin, H., Dufresne, T.E., Ridder, G.M., Turner, J.N. and Roysam, B., "An improved watershed algorithm for counting objects in noisy, anisotropic 3-D biological images", *Image Processing, International Conference*, vol. 3, pp. 172 -175, 1995.

[21] Andersen, J. D., "Methods for modeling the first layers of the retina", *Neuroinformatics and Neurocomputers, RNNS/IEEE Symposium*, vol.1, pp. 179-186, 1992.

[22] Antoine, M.J., Travere, J.M. and Bloyet, D., "Anisotropic diffusion filtering applied to individual PET activation images: a simulation study", *Nuclear Science Symposium and Medical Imaging Conference Record, IEEE*, vol. 3, pp. 1465-1469, 1995.

[23] Bakalexis, S.A., Boutalis, Y.S. and Mertzios, B.G., "Edge detection and image segmentation based on nonlinear anisotropic diffusion", *Digital Signal Processing, DSP. 2002 14th International Conference*, vol. 2, pp. 1203 -1206, 2002.

[24] Barash, D. "Fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 24 Issue: 6, pp. 844 -847, Jun. 2002.

[25] Bayram, E., Ge, Y. and Wyatt, C.L., "Confidence based anisotropic filtering of magnetic resonance images", *Engineering in Medicine and Biology Society, Proceedings of the 23rd Annual International Conference of the IEEE*, vol. 3, pp. 2526-2529, 2001.

[26] Bei, T., Sapiro, G. and Caselles, V., "Chromaticity diffusion", *Image Processing, 2000 International Conference*, vol. 2, pp. 784-787, 2000.

[27] Bennamoun, M., "Edge detection: problems and solutions", *Systems, Man, and Cybernetics, 'Computational Cybernetics and Simulation', 1997 IEEE International Conference*, vol. 4, pp. 3164-3169, Oct 1997.

[28] Berger, M.O., Maurice, N., Winterfeldt, G. and Lethor, J.-P., "Automatic 3D reconstruction of the beating left ventricle using transthoracic echographic images", *Computers in Cardiology*, pp. 641 -644, 1998.

[29] Bhandari, D., Pal, N.R., and Majumder, D.D., "Fuzzy divergence, probability measure of fuzzy events and image thresholding", *Pattern Recognition, Letter*, Vol. 13, pp. 857-867, 1992.

[30] Black, M., Sapiro, G., Marimont, D. and Heeger, D., "Robust anisotropic diffusion", *Image Processing, IEEE Transactions*, vol. 7, Issue: 3, pp. 421-432, March 1998.

[31] Black, M., Sapiro, G., Marimont, D. and Heeger, D., "Robust anisotropic diffusion and sharpening of scalar and vector images", *Image Processing, International Conference*, vol. 1, pp. 263-266, 1997.

[32] Boccignone, G. and Ferraro, M., "Multiscale contrast enhancement", *Electronics Letters*, vol. 37, Issue. 12, pp. 751 -752, June 2001.

[33] Boccignone, G., Ferrearo, M. and Caelli, T., "Encoding visual information using anisotropic transformations",*Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 23, Issue. 2, pp. 207-211, Feb. 2001.

[34] Boccignone, G., Ferraro, M. and Caelli, T., "Visual information from anisotropic transformations", *Image Analysis and Processing, International Conference*, pp. 334-339, 1999.

[35] Boccignone, G., "A multiscale contrast enhancement method", *Image Processing, International Conference*, vol. 1, pp. 306-309, 1997.

[36] Boccignone, G. and Picariello, A., "Multiscale contrast enhancement of medical images", *Acoustics, Speech, and Signal Processing, ICASSP-97, 1997 IEEE International Conference*, vol. 4, pp. 2789 -2792, 1997.

[37] Boccignone, G. and Picariello, A., "Anisotropic enhancement of mammographic images", *Pattern Recognition, Proceedings of the 13th International Conference*, vol. 2, pp. 408 -412, 1996.

[38] Boccignone, G. and Picariello, A., "Enhancement of mammograms: experimental results", *Image Processing, Proceedings, International Conference*, vol. 1, pp. 347-350, 1996.

[39] Bosson, A. and Harvey, R.W., "Using occlusion models to evaluate scale-space processors", *Image Processing, ICIP 98, 1998 International Conference*, vol. 1, pp. 615-619, 1998.

[40] Brancaleoni, F., Mikula, K., Sarti, A. and Lamberti, C., "3D echocardiography pre-processing for ventricular volume estimation" *Computers in Cardiology*, pp. 311-314, 1997.

[41] Burkle, D., Preusser, T. and Rumpf, M., "Transport and anisotropic diffusion in time-dependent flow visualization", *Visualization, VIS '01, Proceedings*, pp. 61 -67, 2001.

[42] Burton, M. and Acton, S., "Target tracking using the anisotropic diffusion pyramid", *Image Analysis and Interpretation, Proceedings of the IEEE Southwest Symposium*, pp. 112-116, 1996.

[43] Canny, J., "A computational approach to edge detection", *IEEE Pattern Analysis and Machine Intelligence 8th*, No. 6, Nov. 1986.

[44] Castleman, K.R., "Digital Image Processing", *Upper Saddle River, N.J, Toronto, Prentice Hall Inc.*, 1996.

[45] Catte, F., Dibos, F. and Koepfler, G., "A morphological scheme for mean curvature motion and applications to anisotropic diffusion and motion of level sets", *Pattern Recognition, 1994, Vol. 3 - Conference C: Signal Processing, Proceedings of the 12th IAPR International Conference*, vol.3, pp. 156-158, 1994.

[46] Catte, F., Lions, P., Morel, J. and Coil, T., "Image selective smoothing and edge detection by nonlinear diffusion", *SIAM J. Num. Anal.*, Vol 29. No. 1, pp. 182-193, Feb. 1992.

[47] Ceccarelli, M., De Simone, V. and Murli, A., "Well-posed anisotropic diffusion for image denoising", *Vision, Image and Signal Processing, IEE Proceedings*, vol. 149, Issue: 4, pp. 244 -252, Aug. 2002.

[48] Chen, Y. and Levine, S.E., "Image recovery via diffusion tensor and time-delay regularization", *Journal of Visual Communication and Image Representation*, Vol. 13, pp. 156-175, 2002.

[49] Chaine, R., Bouakaz, S. and Vandorpe, D., "A graph-anisotropic approach to 3-D data segmentation", *Computer Graphics, Image Processing, and Vision, SIBGRAPI '98, International Symposium*, pp. 262-269, 1998.

[50] Chambolle, A., "Partial differential equations and image processing", *Image Processing, Proceedings, ICIP-94, IEEE International Conference*, vol. 1, pp. 16-20, 1994.

[51] Chou, P.-C. and Bennamoun, M., "Accurate localisation of edges in noisy volume images", *Pattern Recognition, 2000. Proceedings. 15th International Conference*, vol. 4, pp. 760-763, 2000.

[52] Christodoulou, C.I., Loizou, C., Pattichis, C.S., Pantziaris, M., Kyriakou, E., Pattichis, M.S., Schizas, C.N. and Nicolaides, A., "De-speckle filtering in ultrasound imaging of the carotid artery", *Engineering in Medicine and Biology, 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society, EMBS/BMES Conference*, vol. 2, pp. 1027-1028, 2002.

[53] Chung, D. and Sapiro, G., "Segmenting skin lesions with partial-differential-equations-based image processing algorithms", *Medical Imaging, IEEE Transactions*, vol. 19, Issue: 7, pp. 763 -767, July 2000.

[54] Cong, G. and Ma, S.D., "Nonlinear diffusion for early vision", *Proc. 13th Int. Conf. Pattern Recognition*, Vienna, Vol. A, pp. 403-406, Aug. 1996.

[55] Cottet, G.H. and Ayyadi, M., "A volterra type model for image processing", *Image Processing, IEEE Transactions*, vol. 7, Issue. 3, pp. 292-303, March 1998.

[56] Cottet, G.-H. and El Ayyadi, M., "Nonlinear PDE operators with memory terms for image processing", *Image Processing, Proceedings, International Conference*, vol. 1, pp. 481-483, 1996.

[57] Dang, T., Jamet, O. and Maitre, H., "An image segmentation technique based on edge-preserving smoothing filter and anisotropic diffusion", *Image Analysis and Interpretation, Proceedings of the IEEE Southwest Symposium*, pp. 65-69, 1994.

[58] Dautray, R. and Lions, J.-L., "Mathematical Analysis and Numerical Methods for Science and Technology", Volume 6, Evolution Problems II, *Springer-Verlag, Berlin*, 1988.

[59] Demassieux, N., Jutand, F., Saint-Paul, M. and Dana, M., "VLSI Architecture for a one chip video median filter", *Acoustics, Speech, and Signal Processing, IEEE International Conference, ICASSP '85*, Vol. 10, pp. 1001 -1004, April 1985.

[60] Demirkaya, O., "Improving SNR in PET images by using anisotropic diffusion filtration", *Engineering in Medicine and Biology Society, Proceedings of the 22nd Annual International Conference of the IEEE*, vol. 1, pp. 501-503, 2000.

[61] Denyer, P.B., (May, 2001), "CMOS vs CCD", [online], Available at: *http://www.vvl.co.uk/whycmos/whitepaper.htm*, June 8th, 2003.

[62] Diewald, U., Preusser, T. and Rumpf, M., "Anisotropic diffusion in vector field visualization on Euclidean domains and surfaces", *Visualization and Computer Graphics, IEEE Transactions*, vol. 6, Issue: 2, pp. 139-149 , April-June 2000.

[63] Dorati, A., Lamberti, C., Sarti, A., Baraldi, P. and Pini, R., "Pre-processing for 3D echocardiography", *Computers in Cardiology*, pp. 565-568, 1995.

[64] Fernandes, R.A and Jernigan, M.E., "Unsupervised multi-level segmentation of multispectral images", *Neural Networks for Signal Processing [1992] II, Proceedings of the 1992 IEEE-SP Workshop*, pp. 363-372, 1992.

[65] Ferraro, M., Boccignone, G. and Caelli, T., "Isotropic versus anisotropic encoding of visual information", *Image Processing, 2000 International Conference*, vol.3, pp. 905-908, 2000.

[66] Fischl, B. and Schwartz, E.L., "Adaptive nonlocal filtering: a fast alternative to anisotropic diffusion for image enhancement", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 21, Issue. 1, pp. 42 -48, Jan. 1999.

[67] Fischl, B. and Schwartz, E.L., "Learned adaptive nonlinear filtering for anisotropic diffusion approximation in image processing", *Pattern Recognition, Proceedings of the 13th International Conference*, vol. 3, pp. 276-280, 1996.

[68] Fischl, B. and Schwartz, E.L., "Learning an integral equation approximation to nonlinear anisotropic diffusion in image processing", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 19, Issue: 4, pp. 342-352, April 1997.

[69] Fonseca, L.M.G. and Kenney, C.S., "Control point assessment for image registration", *Computer Graphics and Image Processing, XII Brazilian Symposium*, pp. 125-132, 1999.

[70] Fontaine, F.L. and Basu, S., "Multiresolution approach to solving diffusion equation for edge detection", *Circuits and Systems, ISCAS '92, Proceedings, 1992 IEEE International Symposium*, vol. 2, pp. 975 -978, 1992.

[71] Ford, G.E., Estes, R.R. and Chen, H., "Space scale analysis for image sampling and interpolation", *Acoustics, Speech, and Signal Processing, ICASSP-92, 1992 IEEE International Conference*, vol. 3, pp. 165-168, 1992.

[72] Fossum, E.R., "CMOS image sensor: electronic camara-on-a-chip", *IEEE Transactions on Electron Devices*, Vol. 44, No. 10, pp. 1689 -1698, Oct. 1997.

[73] Fossum, E.R., "Low power camera-on-a-chip using CMOS active pixel sensor technology", *Low Power Electronics, IEEE Symposium*, pp. 74 -77, Oct. 1995.

[74] Frangakis, A.S., Stoschek, A. and Hegerl, R., "Wavelet transform filtering and nonlinear anisotropic diffusion assessed for signal reconstruction performance on multidimensional biomedical data", *Biomedical Engineering, IEEE Transactions*, vol. 48, Issue. 2, pp. 213-222, Feb. 2001.

[75] Gallo, G., Zingale, A. and Zingale, R., "Detection of MRI brain contour using nonlinear anisotropic diffusion filter", *Engineering in Medicine and Biology Society, 1996, Bridging Disciplines for Biomedicine, 18th Annual International Conference of the IEEE*, vol. 3, pp. 1062-1064, 1997.

[76] Gao, J., Zhang, J., Fleming, M.G., Pollak, I. and Cognetta, A.B., "egmentation of dermatoscopic images by stabilized inverse diffusion equations", *Image Processing, ICIP 98. Proceedings. 1998 International Conference*, vol.3, pp. 823 -827, 4-7 Oct. 1998

[77] Gao, X., Wang, C. and Zhang, H., "Anisotropic diffusion filtering and phase unwrapping for interferometric SAR", *Geoscience and Remote Sensing Symposium, IGARSS '02. 2002 IEEE International*, vol. 3, pp. 1744 -1746, 2002.

[78] Gealow, J.C., Herrmann, F.P., Hsu, L.T. and Sodini, C.G., "System design for pixel-parallel image processing", *Very Large Scale Integration (VLSI) Systems, IEEE Transactions*, Vol: 4 Issue: 1, pp. 32 -41, March 1996.

[79] Gerado, I.S.-O. and Alison, N., "Fuzzy clustering driven anisotropic diffusion: enhancement and segmentation of cardiac MR images", *Nuclear Science Symposium, 1998 Conference Record, IEEE*, vol. 3, pp. 1873-1874, 1998.

[80] Gerig, G., Kubler, O., Kikinis, R. and Jolesz, F.A., "Nonlinear anisotropic filtering of MRI data", *Medical Imaging, IEEE Transactions*, vol. 11, Issue: 2, pp. 221-232, June 1992.

[81] Giakoumis, I. and Pitas, I., "Digital restoration of painting cracks", *Circuits and Systems, ISCAS '98, Proceedings of the 1998 IEEE International Symposium*, vol. 4, pp. 269-272, 1998.

[82] Gijbels, T., Six, P., Gool, L., Catthoor, F., De Man, H. and Oosterlinck, A., "A VLSI-architecture for parallel non-linear diffusion with applications in vision", *VLSI Signal Processing* VII, pp. 398-407, 1994.

[83] Gilboa, G., Sochen, N. and Zeevi, Y.Y., "Forward-and-backward diffusion processes for adaptive image enhancement and denoising", *Image Processing, IEEE Transactions*, vol. 11 Issue: 7 , pp. 689 -703, July 2002.

[84] Gilboa, G., Zeevi, Y.Y. and Sochen, N., "Resolution enhancement by forward-and-backward nonlinear diffusion process", *Nonlinear Signal and Image Processing, Baltimore, Maryland*, June 2001.

[85] Gilboa, G., Zeevi, Y.Y. and Sochen, N., "Anisotropic selective inverse diffusion for signal enhancement in the presence of noise", *Acoustics, Speech, and Signal Processing, ICASSP '00. Proceedings. 2000 IEEE International Conference*, vol. 1, pp. 221 -224, 5-9 June 2000.

[86] Gilboa, G., Zeevi, Y.Y. and Sochen, N., "Signal and image enhancement by a generalized forward-and-backward adaptive diffusion process", *EUSIPCO-2000, Tampere, Finland*, Sept. 2000.

[87] Glaseby, C.A., "An analysis of histogram based thresholding algorithms", *CVGIP: Graphical Models and Image Processing*, Vol. 55, pp. 532-533, 1993.

[88] Gonzalez, R.C. and Woods, R.E., "Digital image processing", *Reading, Mass, Addison-Wesley*, 1992.

[89] Gregson, P.H., "Using angular dispersion of gradient direction for detecting edge ribbons", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol.15, No.7, pp. 682 -696, July 1993.

[90] Gregson, P.H., "Using gradient orientation for edgel detection", *Communications, Computers and Signal Processing, 1989 Conference Proceeding, IEEE, Pacific Rim Conference*, pp. 293-296, 1989.

[91] Gulino, F.G., "Consistency-Enhanced anisotropic diffusion in analog VLSI", *Master Thesis, Department of Electrical and Computer Engineering, Dalhousie University*, 2001

[92] Hae, Y.K. and Zang, H.C. "Robust anisotropic diffusion to produce clear statistical parametric map from noisy fMRI", *Computer Graphics and Image Processing, Proceedings. XV Brazilian Symposium*, pp. 11 -17, 2002.

[93] Hagen, S. and Hanno, S., "Accurate optical flow in noisy image sequences", *Computer Vision, ICCV 2001. Proceedings. Eighth IEEE International Conference*, vol. 1, pp. 587-592, 2001.

[94] Haris, K., Efstratiadis, S.N., Maglaveras, N. and Pappas, C., "Semi-automatic extraction of vascular networks in angiograms", *Engineering in Medicine and Biology Society, 1996. Bridging Disciplines for Biomedicine, 18th Annual International Conference of the IEEE*, vol. 3, pp. 1067-1068, 1997.

[95] Havlicek, J.P., Sarkady, K.A., Katz, G.R. and McKeeman, J.C., "Fast, efficient median filters with even length windows", *Electronics Letters*, Vol: 26 Issue: 20 , pp. 1736 -1737, Sept. 1990.

[96] He, H., Tian, J., Wang, J., Chen, H. and Zhang, X.P., "Improved MSEL and its medical application", *Pattern Recognition, Proceedings. 16th International Conference*, vol. 2, pp. 597-600, 2002.

[97] Helman, D. and JaJa, J., "Efficient image processing algorithms on the scan line array processor", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, Vol: 17, Issue: 1, pp.47 -56,Jan. 1995.

[98] Hollig, K., and Nohel, J.A., "A diffusion equation with a nonmonotone constructive function", *Proc. Syst. Nonlinear Partial Differential Equation*, J. Ball, ED. Boston: D. Reidel, pp. 409-422, 1983.

[99] Huang, H. and Wang, J.-H., "Anisotropic diffusion for object segmentation", *Systems, Man, and Cybernetics, 2000 IEEE International Conference*, vol. 3, pp. 1563-1567, 2000.

[100] Hurwitz, J.E.D., Denyer, P.B., Baxter, D.J. and Townsend, G., "An 800K-Pixel Colour CMOS Sensor for Consumer Still Cameras", *Electronic Imaging at SPIE Photonics West*, Feb. 1997.

[101] Inglada, J. and Garello, R., "Depth estimation and 3D topography reconstruction from SAR images showing underwater bottom topography signatures", *Geoscience and Remote Sensing Symposium, IGARSS '99 Proceedings, IEEE 1999 International*, vol. 2, pp. 956-958, 1999.

[102] Intajag, S. and Paithoonwatanakij, K., "Automated boundary extraction by gradient fuzzy gray-scale hit-or-miss transformation" *Circuits and Systems, IEEE APCCAS 2000, The 2000 IEEE Asia-Pacific Conference*, pp. 465-468, 2000.

[103] Izquierdo, E. and Ghanbari, M., "Using 3D structure and anisotropic diffusion for object segmentation", *Image Processing and Its Applications, Seventh International Conference on (Conf. Publ. No. 465)*, vol. 2, pp. 532 -536, 1999.

[104] Izquierdo, E. and Ghanbari, M., "Motion-driven object segmentation in scale-space", *Acoustics, Speech, and Signal Processing, 1999 IEEE International Conference*, vol. 6, pp. 3473-3476, 1999.

[105] Izquierdo, M. and Ghanbari, M., "Nonlinear Gaussian filtering approach for object segmentation", *Vision, Image and Signal Processing, IEE Proceedings*, vol. 146, Issue: 3, pp. 137 -143, June 1999.

[106] Izquierdo, M. and Ghanbari, M.E., "Texture smoothing and object segmentation using feature-adaptive weighted Gaussian filtering", *Telecommunications Symposium, ITS '98 Proceedings, SBT/IEEE International*, vol. 2, pp. 650-655, 1998.

[107] Izquierdo, E. and Ghanbari, M., "Fast nonlinear diffusion approach for object segmentation", *Electronics Letters*, vol. 34, Issue: 19, pp. 1836-1837, Sept. 1998.

[108] Jain, R., Kasturi, R. and Schunck, B.G., "Machine vision", *New York, Toronto, McGraw-Hill*, 1995.

[109] Jin, J.S., Wang Y. and Hiller, J., "An adaptive nonlinear diffusion algorithm for filtering medical images", *Information Technology in Biomedicine, IEEE Transactions*, vol. 4, Issue. 4, pp. 298 -305, Dec. 2000.

[110] Joseph, D. and Collins, S., "Modelling, calibration and correction of nonlinear illumination-dependent fixed pattern noise in logarithmic CMOS image sensors", *IEEE Instrumentation and Measurement Technology Conference, Budapest, Hungary*, pp. 1296 -1301, May, 2001.

[111] Kervrann, C., Hoebeke, M. and Trubuil, A., "A level line selection approach for object boundary estimation",*Computer Vision, The Proceedings of the Seventh IEEE International Conference*, vol. 2, pp. 963-968, 1999.

[112] Kim, H., Jang, J. and Hong, K., "Edge-enhancing super-resolution using anisotropic diffusion", *Image Processing, 2001 International Conference*, vol. 3, pp. 130-133, 2001.

[113] Klette, R. and Zamperoni, P., "Handbook of Image Processing Operators", *John Wiley & Sons, Toronto*, 1996.

[114] Kite, T.D., Damera-Venkata, N., Evans, B.L. and Bovik, A.C., "A fast, high-quality inverse halftoning algorithm for error diffused halftones", *Image Processing, IEEE Transactions*, vol. 9, Issue. 9, pp. 1583-1592, Sep. 2000.

[115] Kopilovic, I. and Sziranyi, T., "Non-linear scale-selection for image compression improvement obtained by perceptual distortion criteria", *Image Analysis and Processing, International Conference*, pp. 197-202, 1999.

[116] Kornprobst, P., Deriche, R. and Aubert, G., "Nonlinear operators in image restoration", *Computer Vision and Pattern Recognition, IEEE Computer Society Conference*, 1997.

[117] Krissian, K., "Flux-based anisotropic diffusion applied to enhancement of 3-D angiogram", *Medical Imaging, IEEE Transactions*, vol. 21 Issue: 11, pp. 1440-1442, Nov. 2002.

[118] Krymski, A., Van Blerkom, D., Andersson, A. and Bock, N., Mansoorian, B., Fossum, E.R., "A high speed, 500 frames/s, 1024x1024 CMOS active pixel sensor", *VLSI Circuits, Digest of Technical Papers, Symposium*, pp. 137 -138, 1999.

[119] Kutka, R. and Stier, S., "Extraction of line properties based on direction fields", *Medical Imaging, IEEE Transactions*, vol. 15, Issue: 1, pp. 51-58, Feb. 1996.

[120] Lamberti, C., Sitta, M. and Sgallari, F., "Improvements to the Anisotropic Diffusion Model for 2-D Echo Image Processing", *Engineering in Medicine and Biology Society, vol.14, Proceedings of the Annual International Conference of the IEEE*, vol. 5, pp. 1872 -1873, 1992.

[121] Lamberti, C., Sgallari, F. and Venturelli, G., "Anisotropic diffusion technique for 2-D echocardiography image processing", *Engineering in Medicine and Biology Society, Proceedings of the Annual International Conference of the IEEE*, vol. 13, pp. 233-234, 1991.

[122] Lamberti, C. and Sgallari, F., "A workstation-based system for 2-D echocardiography visualization and image processing", *Biomedical Engineering, IEEE Transactions*, vol. 37 Issue: 8, pp. 796-802, Aug. 1990.

[123] Lamberti, C., Lusvardi, S. and Truzzi, C., "Workstation for 2-D echocardiography image processing", *Computers in Cardiology 1989, Proceedings*, pp. 447-450, 1990.

[124] Laure, B.-F., Pierre, C., Gilles, A. and Michel, B., "Nonlinear image processing: modeling and fast algorithm for regularization with edge detection", *Image Processing, International Conference*, vol. 1, pp. 474-477, 1995.

[125] Lee, S., Kang, M. and Park, K., "CCD noise filtering based on 3-dimensional nonlinear partial differential equation", *Consumer Electronics, IEEE Transactions*, vol. 44, Issue. 3, pp. 1086-1090, Aug. 1998.

[126] Lee, S. and Kang, M., "Spatio-temporal video filtering algorithm based on 3-D anisotropic diffusion equation", *Image Processing, ICIP 98, 1998 International Conference*, vol. 2, pp. 447-450, 1998.

[127] Lennon, M., Mercier, G. and Hubert-Moy, L., "Nonlinear filtering of hyperspectral images with anisotropic diffusion", *Geoscience and Remote Sensing Symposium, IGARSS '02. 2002 IEEE International*, vol. 4, pp. 2477 -2479, 2002.

[128] Li, L.,Zheng, Y., Zhang, L. and Clark, R.A., "Anisotropic diffusion filtering of mammographic image for CAD in digital mammography", *Signal Processing Proceedings, WCCC-ICSP 2000, 5th International Conference*, vol. 2, pp. 1159-1162, 2000.

[129] Li, X. and Chen, T., "Nonlinear diffusion with multiple edginess thresholds", *Pattern Recognition*, Vol. 27, pp. 1029-1037, 1994.

[130] Liang, K.-H., Tjahjadi, T. and Yang, Y.-H.,"Bounded diffusion for multiscale edge detection using regularized cubic B-spline fitting", *Systems, Man and Cybernetics, Part B, IEEE Transactions*, vol. 29, Issue: 2, pp. 291-297, April 1999.

[131] Liang, P. and Wang, Y.F., "Local scale controlled anisotropic diffusion with local noise estimate for image smoothing and edge detection", *Computer Vision, Sixth International Conference*, pp. 193 -200,1998.

[132] Lin, Z. and Shi, Q., "An anisotropic diffusion PDE for noise reduction and thin edge preservation", *Image Analysis and Processing, International Conference*, pp. 102-107, 1999.

[133] Ling, J. and Bovik, A.C., "Modeling and restoration of Raman microscopic images", *Image Processing, 2000 International Conference*, vol.3, pp. 428-431, 2000.

[134] Ling, J. and Bovik, A.C., "Smoothing low-SNR molecular images via anisotropic median-diffusion", *Medical Imaging, IEEE Transactions*, vol. 21 Issue: 4, pp. 377-384, Apr 2002.

[135] Liu, X., Wang, D. and Ramirez, J.R., "Oriented statistical nonlinear smoothing filter", *Image Processing, ICIP 98, 1998 International Conference*, vol. 2, pp. 848-852, 1998.

[136] Loizou, C., Christodoulou, C., Pattichis, C.S., Istepanian, R., Pantziaris, M. and Nicolaides, A., "Speckle reduction in ultrasound images of atherosclerotic carotid plaque", *Digital Signal Processing, DSP, 2002 14th International Conference*, vol. 2, pp. 525-528, 2002.

[137] Lucchese, L. and Mitra, S.K., "Colour segmentation based on separate anisotropic diffusion of chromatic and achromatic channels", *Vision, Image and Signal Processing, IEE Proceedings*, vol. 148, Issue. 3, pp. 141-150, June 2001.

[138] Lucchese, L. and Mitra, S.K., "Color segmentation through independent anisotropic diffusion of complex chromaticity and lightness", *Image Processing, 2001 International Conference*, vol. 1, pp. 746-749, 2001.

[139] Luijendijk, H., "Automatic threshold selection using histograms based on the count of 4-connected regions", *Pattern Recognition, Letter*, Vol. 12, pp. 219-228, 1991.

[140] Maeda, J., Iizawa, T., Ishizaka, T. and Suzuki Y., "Accurate segmentation of noisy images using anisotropic diffusion and linking of boundary edge", *TENCON '97. IEEE Region 10 Annual Conference, Speech and Image Technologies for Computing and Telecommunications, Proceedings of IEEE*, vol. 1, pp. 279-282, 1997.

[141] Manay, S. and Yezzi, A., "Anti-geometric diffusion for adaptive thresholding and segmentation", *Image Processing, 2001 International Conference*, vol. 2, pp. 829-832, Oct. 2001.

[142] Marshall, D., (1994), "Vision Systems", [online], Available: *http://www.cs.cf.ac.uk/Dave/Vision_lecture/Vision_lecture_caller.html*, April 28, 2003.

[143] Mead, C., "Neuromorphic Electronic Systems", Proceeding of the IEEE, Vol. 78, No. 10, pp. 1629 -1636, Oct. 1990.

[144] MIT Intelligent Transportation Research Center, (April 1999), "Pixel-Parallel Image Processing System for Intelligent Transportation System Applications", [online], Available: *http://www-mtl.mit.edu/mtlhome/6Res/ITRC/*, April 8th, 2003.

[145] McQueen, P.G., Jin, A.J., Pierpaoli, C. and Basser, P.J., "Development of a finite element model of molecular diffusion in living brain from in vivo magnetic resonance", *Biomedical Engineering Conference, Proceedings of the 1996 Fifteenth Southern*, pp. 289-292, 1996.

[146] Mhani, A., Sicard, G. and Bouvier, G., "Analog vision chip for sensing edges contrasts and motion", *IEEE International Symposium on Circuits and Systems*, pp. 2769 -2772, June 1997.

[147] Monteil, J. and Beghdadi, A., "A new interpretation and improvement of the nonlinear anisotropic diffusion for image enhancement", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 21, Issue: 9, pp. 940-946, Sept. 1999.

[148] Monteil, J. and Beghdadi, A., "A new adaptive nonlinear anisotropic diffusion for noise smoothing", *Image Processing, ICIP 98, 1998 International Conference*, vol.3, pp. 254-258, 1998.

[149] Mukherjee, D.P. and Acton, S.T., "Conditional anisotropic diffusion for leukocyte identification", *Circuits and Systems, WSCAS-2002. The 2002 45th Midwest Symposium*, vol. 1, pp. 212 -215, 2002.

[150] Munson, D.C., "A Note on Lena", *IEEE Transaction on image processing*, vol.5., NO. 1. Jan. 1996.

[151] Muramatsu, S., Kobayashi, Y., Araoka, M., Naoi, S., Kaneta, T., Hirose, K. and Onizawa, S., "Image processing LSI “ISP-IV” based on local parallel architecture and its applications", *Image Processing, ICIP 98, International Conference*, vol.3, pp. 1000 -1004, Oct. 1998.

[152] Nalwa, V.S., "A guided tour of computer vision ", *Reading, Mass.: Addison-Wesley*, c1993

[153] Neoh, H. and Sapiro, G., "Using anisotropic diffusion of probability maps for activity detection in block-design functional MRI", *Image Processing, 2000 International Conference*, vol. 1, pp. 621-624, 2000.

[154] Ng, C.M., Leung, W.N. and Chun, F., "Edge detection using evolutionary algorithms", *Systems, Man, and Cybernetics, IEEE SMC '99 Conference ,1999 IEEE International Conference*, vol. 4, pp. 865-868, 1999.

[155] Nirenbarg, N., "A strong maximum principle for parabolic equation", *Commun. Pure Appl. Math.*, vol. VI, pp. 167-177, 1953.

[156] Nitzberg, M. and Shiota, T., "Nonlinear image filtering with edge and corner enhancement", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 14, Issue: 8, pp. 826 -833, Aug. 1992.

[157] Nixon, R.H., Kemeny, S.E., Staller, C.O. and Fossum, E.R., "256x256 CMOS active pixel sensor camera-on-a-chip", *Solid-State Circuits Conference, Digest of Technical Papers, 43rd ISSCC., 1996 IEEE International*, Feb. 1997.

[158] Ohsawa, S., Sasaki, M., Miyagawa, R. and Matsunaga, Y., "Analysis of low fixed pattern noise cell structures for photoconversion layer overlaid CCD or CMOS image sensors", *IEEE Transactions on Electron Devices*, Vol. 44, No. 10, pp. 1667 -1671, Oct. 1997.

[159] Owens, R., (Oct. 1997), "Problems with gradient-based edge detectors", [online], Available at: *http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/OWENS/LECT6/node2.html*, May 6, 2003.

[160] Pain, B., Yang, G., Olson, B., Shaw, T., Ortiz, M., Heynssens, J., Wrigley, C. and Ho, C., "A low-power digital camera-on-a-chip implemented in CMOS active pixel approach " *VLSI Design, Proceedings, Twelfth International Conference*, pp. 26 -31, Jan. 1999.

[161] Panin, V.Y., Zeng, G.L. and Gullberg, G.T., "Total variation regulated EM algorithm", *Nuclear Science Symposium, 1998 Conference Record, IEEE*, vol. 3, pp. 1562-1566, 1998.

[162] Paplinski, A.P., "Current improvements in interpretation of posterior capsular opacification images", *Signal Processing and its Applications, Sixth International, Symposium*, vol. 1, pp. 218-221, 2001.

[163] Pardo, A. and Sapiro, G, "Vector probability diffusion", *IEEE Signal Processing Letters*, vol. 8, Issue. 4, pp. 106-109, April 2001.

[164] Pardo, A. and Sapiro, G., "Vector probability diffusion", *Image Processing, 2000 International Conference*, vol. 1, pp. 884-887, 2000.

[165] Pathak, S.D., Haynor, D.R. and Kim, Y., "Edge-guided boundary delineation in prostate ultrasound images", *Medical Imaging, IEEE Transactions*, vol. 19, Issue: 12, pp. 1211-1219, Dec. 2000.

[166] Peng, C., Chan, A. and Wang, J., "Speckle noise removal and edge enhancement on SAR image using anisotropic diffusion and discrete wavelet transform", *Geoscience and Remote Sensing Symposiun, 2000 Proceedings, IGARSS 2000, IEEE 2000 International*, vol. 4, pp. 1663-1665, 2000.

[167] Perona, P., "Orientation diffusions", *Image Processing, IEEE Transactions*, vol. 7, Issue: 3, pp. 457 -467, March 1998.

[168] Perona, P., "Orientation diffusions", *Computer Vision and Pattern Recognition, 1997 IEEE Computer Society Conference*, pp. 710-716, 1997.

[169] Perona, P. and Malik, J., "Scale-space and edge detection using anisotropic diffusion", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, Vol. 12, Issue 7, pp. 629-639, July 1990.

[170] Perona, P. and Malik, J., "A network for multiscale image segmentation" *Circuits and Systems, IEEE International Symposium*, 1988.

[171] Perona, P. and Malik, J., "Scale-space and edge detection using anisotropic diffusion", *IEEE Computer Society Workshop on Computer Vision - Miami*, 1987.

[172] Peter, W.F., Peter, J.W.N. and Robert J.R., "Fixed-Pattern Noise in Photomatirces", *IEEE Journal of Solid-State Circuits*, Vol. sc-5, No. 5, pp. 250 -254, Oct. 1970.

[173] Pollak, I., Willsky, A.S. and Krim, H.,"Image segmentation and edge enhancement with stabilized inverse diffusion equations", *Image Processing, IEEE Transactions*, vol. 9 Issue: 2 , pp. 256 -266, Feb. 2000.

[174] Pollak, I., Krim, H. and Willsky, A.S., "Stabilized inverse diffusion equations and segmentation of vector-valued images", *Image Processing, ICIP 98. International Conference*, vol.3, pp. 246 -248, 4-7 Oct. 1998.

[175] Pope, K. and Acton, S.T., "Modified mean curvature motion for multispectral anisotropic diffusion", *Image Analysis and Interpretation, 1998 IEEE Southwest Symposium*, pp. 154 -159, 1998.

[176] Pratt, W.K., "Digital image processing", Second edition, *John Wiley & Sons Inc.*, New York, 1991.

[177] Qi, H., Snyder, W.E. and Bilbro, G.L., "Using mean field annealing to solve anisotropic diffusion problems", *Image Processing, International Conference*, vol. 3, pp. 352-355, 1997.

[178] Qian, R. J. and Sezan, M.I., "Video background replacement without a blue screen", *Image Processing, ICIP 99, Proceedings, 1999 International Conference*, vol. 4, pp. 143-146, 1999.

[179] Ramponi, G., "Image processing with rational operators: noise smoothing and anisotropic diffusion", *Image and Signal Processing and Analysis, ISPA 2001, Proceedings of the 2nd International Symposium*, pp. 96 -101, 2001.

[180] Rudin, L.I., Osher, S. and Fatemi, E., "Nonlinear total variation based noise removal algorithms", *Physica D*, vol. 60, pp. 259-268, 1992.

[181] Russ, J.C., "The image processing handbook", Second edition, *Boca Raton : CRC Press*, 1995.

[182] Saint-Marc, P., Chen, J.S. and Medioni, G., "Adaptive smoothing: a general tool for early vision", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 13, Issue. 6, pp. 514 -529, June 1991.

[183] Saint-Marc, P., Chen, J. and Medioni, G., "Adaptive smoothing: a general tool for early vision", *Computer Vision and Pattern Recognition, Proceedings CVPR '89, IEEE Computer Society Conference*, pp. 618 -624, 1989.

[184] Sakaino, H., "Nonlinear robust velocity estimation of vehicles from a snowfall traffic scene", *Pattern Recognition, 2002. Proceedings. 16th International Conference*, vol. 4, pp. 60-63, 2002.

[185] Samsonov, A.A. and Johnson, C.R., "Noise-adaptive anisotropic diffusion filtering of MRI images reconstructed by SENSE (sensitivity encoding) method" *Biomedical Imaging, Proceedings.IEEE International Symposium*, pp. 701 - 704, 2002.

[186] Santarelli, M. F., Positano, V., Landini, L. and Benassi, A., "A new algorithm for 3D automatic detection and tracking of cardiac wall motion", *Computers in Cardiology*, pp. 133-136, 1999.

[187] Sapiro, G. and Ringach, D.L., "Anisotropic diffusion of multivalued images with applications to color filtering", *Image Processing, IEEE Transactions*, vol. 5, Issue: 11, pp. 1582-1586, Nov. 1996.

[188] Sapiro, G., "Vector-valued active contours", *Computer Vision and Pattern Recognition, Proceedings CVPR '96, 1996 IEEE Computer Society Conference*, 1996, pp. 680-685.

[189] Sapiro, G., "From active contours to anisotropic diffusion: connections between basic PDE's in image processing", *Image Processing, International Conference*, vol. 1, pp. 477-480, 1996.

[190] Sapiro, G., "Vector (self) snakes: a geometric framework for color, texture, and multiscale image segmentation", *Image Processing, International Conference*, vol. 1, pp. 817-820, 1996.

[191] Sapiro, G., Tannenbaum, A., You, Y. and Kaveh, M., "Experiments on geometric image enhancement", *Image Processing, Proceedings, ICIP-94, IEEE International Conference*, vol. 2, pp. 472-476, 1994.

[192] Sarti, A., Mikula, K. and Sgallari, F., "Nonlinear multiscale analysis of three-dimensional echocardiographic sequences", *Medical Imaging, IEEE Transactions*, vol. 18, Issue: 6, pp. 453-466, June 1999.

[193] Sawaji, T., Sakai, T., Nagai, H., Kunishima, T. and Matsumoto, T., "Implementing resistive fuse with floating gate MOS transistors", *Neural Networks, International Conference*, vol. 2, pp. 894 -898, 1997.

[194] Scheunders, P. and Sijbers, J., "Multiscale anisotropic filtering of color images", *Image Processing, Proceedings. 2001 International Conference*, vol. 3, pp. 170-173, 2001.

[195] Scheunders, P., "Multivalued image segmentation based on first fundamental form", *Image Analysis and Processing, Proceedings. 11th International Conference*, pp. 185-190, Sep 2001.

[196] Schmid, P. , "Lesion detection in dermatoscopic images using anisotropic diffusion and morphological flooding", *Image Processing, ICIP 99, 1999 International Conference*, vol. 3, pp. 449-453, 1999.

[197] Segall, C.A., Acton, S.T. and Katsaggelos, A.K., "Sampling conditions for anisotropic diffusion", *SPIE Conference on Visual Communications and Image Processing, San Jose, California*, Jan. 1999.

[198] Segall, C.A. and Acton, S.T., "Morphological anisotropic diffusion", *Image Processing, International Conference*, vol. 3, pp. 348-351, 1997.

[199] Segall, C.A., Chen, W. and Acton, S.T., "Nonlinear pyramids for object identification", *Signals, Systems and Computers, 1996, Conference Record of the Thirtieth Asilomar Conference*, vol. 2, pp. 1004-1008, 1997.

[200] Sgro, J., Stanton, P. and Delfino, J., (2002), "Evaluation Framework for Scalable Microprocessor Vision Systems in an Industrical Inspection Application", Available at: *http://www.machinevisiononline.org/public/articles/articles.cfm?cat=85*, May 10, 2003.

[201] Smolka, B. and Szczepanski, M., "Forward and backward anisotropic diffusion filtering for color image enhancement", *Digital Signal Processing, DSP, 2002 14th International Conference*, vol. 2, pp.927 -930, 2002.

[202] Snyder, W., Han, Y., Bilbro, G., Whitaker, R. and Pizer, S., "Image relaxation: restoration and feature extraction", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 17, Issue: 6, pp. 620-624, June 1995.

[203] Sochen, N. and Zeevi, Y.Y., " Super-resolution of grey-level images by inverse diffusion processes", *Electrotechnical Conference, MELECON 98., 9th Mediterranean*, vol. 1, pp. 91 -95, 18-20 May 1998

[204] Sochen, N. and Zeevi, Y.Y., " Resolution enhancement of colored images by inverse diffusion processes", *Acoustics, Speech, and Signal Processing, ICASSP. Proceedings of the 1998 IEEE International Conference*, vol. 5 , pp. 2853 -2856, 12-15 May 1998.

[205] Sohn, H. and Jezek, K.C., "Ice sheet margin detection using ERS-1 synthetic aperture radar", *Geoscience and Remote Sensing Symposium, IGARSS '96, 'Remote Sensing for a Sustainable Future.', International*, vol. 1, pp. 148-150, 1996.

[206] Solo, V., "A fast automatic stopping criterion for anisotropic diffusion", *Acoustics, Speech, and Signal Processing, 2002 IEEE International Conference*, Vol. 2. pp. 1661-1664, 2002.

[207] Solo, V., "Automatic stopping criterion for anisotropic diffusion", *Acoustics, Speech, and Signal Processing, 2001 IEEE International Conference*, vol. 6,pp. 3929 -3932, 2001.

[208] Sonka, M., Hlavac, V. and Boyle, R.,"Image Processing, Analysis, and Machine Vision", *PWS Publishing*, 1998.

[209] Steen, E. and Olstad, B., "Volume rendering in medical ultrasound imaging based on nonlinear filtering", *Nonlinear Digital Signal Processing, IEEE Winter Workshop*, pp. P_6.1-P_6.6, 1993.

[210] Stiliyan, N.K., Bart, M., Harr, R. and Max, A.V., "Detection of elongated structures by using invertible angular representations of scalar images", *Image Processing, ICIP 98, Proceedings, 1998 International Conference*, vol.3, pp. 60-64, 1998.

[211] Sziranyi, T., Kopilovic, I. and Toth, B.P., "Anisotropic diffusion as a preprocessing step for efficient image compression", *Pattern Recognition, Fourteenth International Conference*, vol. 2, pp. 1565-1567, 1998.

[212] Sziranyi, T. and Czuni, L., "Picture segmentation with introducing an anisotropic preliminary step to an MRF model with cellular neural networks", *Pattern Recognition, Proceedings of the 13th International Conference*, vol. 4, pp. 366-370, 1996.

[213] Suri, J.S., Dee Wu, Gao, J., Singh, S. and Laxminarayan, S., "A comparison of state-of-the-art diffusion imaging techniques for smoothing medical/non-medical image data", *Pattern Recognition, 16th International Conference*, vol. 1, pp. 508 -511, Aug. 2002.

[214] Tasdizen, T., Whitaker, R., Burchard, P. and Osher, S., "Geometric surface smoothing via anisotropic diffusion of normals", *Visualization, VIS. IEEE*, pp. 125 -132, 2002.

[215] Tang, B., Sapiro, G. and Caselles, V., "Color image enhancement via chromaticity diffusion", *Image Processing, IEEE Transactions*, vol. 10, Issue. 5, pp. 701-707, May 2001.

[216] Teo, P.C., Sapiro, G. and Wandell, B., "Segmenting cortical gray matter for functional MRI visualization", *Computer Vision, Sixth International Conference*, pp. 292-297, 1998.

[217] Teo, P.C., Sapiro, G. and Wandell, B.A., "Creating connected representations of cortical gray matter for functional MRI visualization", *Medical Imaging, IEEE Transactions*, vol. 16, Issue: 6, pp. 852-863, Dec. 1997.

[218] Terebes, R., Lavialle, O., Baylou, P. and Borda, M., "Mixed anisotropic diffusion", *Pattern Recognition, Proceedings. 16th International Conference*, vol. 3, pp. 760 -763, 2002.

[219] Tilmant, C., Sarry, L. and Boire, J.-Y., "PDE and tensor based approach for parietal dynamic tracking in ultrasound sequences", *Engineering in Medicine and Biology Society, Proceedings of the 23rd Annual International Conference of the IEEE*, vol. 3, pp. 2649-2652, 2001.

[220] Torkamani-Azar, F. and Tait, K.E., "Image recovery using the anisotropic diffusion equation", *Image Processing, IEEE Transactions*, vol. 5, Issue. 11, pp. 1573-1578, Nov. 1996.

[221] Torralba, A. and Sinha, P. , "Accurate optical flow in noisy image sequences", *Computer Vision, ICCV 2001, Eighth IEEE International Conference*, vol.1, pp. 587 -592, 2001.

[222] Tschumperle, D. and Deriche, R., "Regularization of orthonormal vector sets using coupled PDE's", *Variational and Level Set Methods in Computer Vision, IEEE Workshop*, pp. 3-10, 2001.

[223] Tseng, D.C. and Huang, M.Y., "Automatic thresholding based on human visual perception", *Image Vision Comput.*, Vol. 11, pp. 539-548, 1993.

[224] Tsuji, H., Sakatani, T., Yashima, Y. and Kobayashi, N., "A nonlinear spatio-temporal diffusion and its application to prefiltering in MPEG-4 video coding", *Image Processing, International Conference*, vol. 1, pp. I-85 -I-88, Sept. 2002.

[225] Umasuthan, M. and Wallace, A.M., "Outlier removal and discontinuity preserving smoothing of range data", *Vision, Image and Signal Processing, IEE Proceedings*, vol. 143, Issue: 3, pp. 191-200, June 1996.

[226] Valverde, F.L., Guil, N., Munoz, J., Nishikawa, R. and Doi, K., "An evaluation criterion for edge detection techniques in noisy images", *Image Processing, Proceedings. 2001 International Conference*, vol. 1, pp. 766-769, 2001.

[227] Vazquez, L., Sapiro, G. and Randall, G., "Segmenting neurons in electronic microscopy via geometric tracing", *Image Processing, ICIP 98, 1998 International Conference*, vol.3, pp. 814-818, 1998.

[228] Vinitski, S., Gonzalez, C., Burnett, C., Buchheit, W., Mohamed, F., Ortega, H. and Faro, S., "3D segmentation in MRI of brain tumors: preliminary results", *Engineering in Medicine and Biology Society, IEEE 17th Annual Conference*, vol. 1, pp. 481-482, 1997.

[229] Vinitski, S., Gonzalez, C. Burnett, C., Seshagiri, S., Mohamed, F.B., Lublin, F.D., Knobler, R.L. and Frazer, G., "Tissue segmentation by high resolution MRI: improved accuracy and stability", *Engineering in Medicine and Biology Society, Engineering Advances: New Opportunities for Biomedical Engineers, Proceedings of the 16th Annual International Conference of the IEEE*, vol. 1, pp. 557-558, 1994.

[230] Wang, S.-Z. and Lee, H.-J., "Dual-binarization and anisotropic diffusion of chinese characters in calligraphy documents", *Document Analysis and Recognition, Proceedings, Sixth International Conference*, pp. 271-275, 2001.

[231] Weickert, J.A., "Applications of nonlinear diffusion in image processing and computer vision", *Acta Mathematica Universitatis Comenianae*, Vol. 70, No. 1, Invited paper, pp. 33-50, 2001.

[232] Weickert, J.A., "Coherence-enhancing diffusion filtering", *International Journal of Computer vision*, Vol. 31, pp. 111-127, 1999.

[233] Weickert, J.A., Romeny, B.H., Florack, L., Koenderink, J. and Viergever, "A review of nonlinear diffusion filtering", *Scale-Space Theory in Computer Vision, Lecture Notes in Computer Science*, Vol. 1252, Invited paper, Springer, Berlin, pp. 3-28, 1997.

[234] Weickert, J.A., Solina, F., Kropatsch, W.G., Klette, R. and Bajcsy, R., "A semidiscrete nonlinear scale-space theory and its relation to the Perona-Malik paradox", *Advances in Computer Vision*, Springer, Wien, pp. 1-10, 1997.

[235] Weickert, J.A., Berger, M.-O., Deriche, R. Herlin, I., Jaffre, J.-M. and Morel, "Nonlinear diffusion scale-spaces: from the continuous to the discrete setting", *ICAOS'96: Images, Wavelets and PDEs, Lecture Notes in Control and Information Sciences*, Vol. 219, Springer, London, pp. 111-118, 1996.

[236] Weickert, J.A., Romeny, B.M. and Viergever, M.A., "Conservative image transformations with restoration and scale-space properties", *Image Processing, International Conference*, vol. 1, pp. 465-468, 1996.

[237] Whitaker, R. T. and Pizer, S. M., "A multi-scale approach to non-uniform diffusion", *CVGIP: Image Understanding*, vol. 57, No.1, pp. 99-110, Jan. 1993.

[238] Widder, D.V., "The heat equation", *Academic Press, New York*, 1975.

[239] Wilson, A., "CMOS sensors contend for camera designs", *Vision Systems Design*, pp. 43 -49, Sept. 2001.

[240] Wong, E.Q. and Algazi, V.R., "Improved directional algorithm of the nonlinear anisotropic diffusion equation for images", *Image Processing, ICIP 99, Proceedings, 1999 International Conference*, vol. 2, pp. 396-400, 1999.

[241] Wu, Z., Herman, G.T. and Browne, J.A., "Edge preserving reconstruction using adaptive smoothing in wavelet domain", *Nuclear Science Symposium and Medical Imaging Conference, 1993 IEEE Conference Record*, vol. 3, pp. 1917-1921, 1993.

[242] Yang, S. and Hu, Y.-H., "Coding artifacts removal using biased anisotropic diffusion", *Image Processing, International Conference*, vol. 2, pp. 346-349, 1997.

[243] Yi, H. and Gregson, P.H., "A Survey of Anisotropic Diffusion", submitted to *International Journal of Computer Vision* in March, 2002.

[244] You, Y. and Kaveh, M., "Fourth-order partial differential equations for noise removal", *Iage Processing, IEEE Transactions*, vol. 9, Issue. 10, pp. 1723-1730, Oct. 2000.

[245] You, Y. and Kaveh, M., "Blind image restoration by anisotropic regularization", *Image Processing, IEEE Transactions*, vol. 8, Issue: 3, pp. 396-407, March 1999.

[246] You, Y. and Kaveh, M. , "Differences in the behaviors of continuous and discrete anisotropic diffusion equations for image processing", *Image Processing, ICIP 98, Proceedings, 1998 International Conference*, vol.3, pp. 249-253, 1998.

[247] You, Y. and Kaveh, M., "Image enhancement using fourth order partial differential equations", *Signals, Systems & Computers, Conference Record of the Thirty-Second Asilomar Conference*, vol. 2, pp. 1677-1681, 1998.

[248] You, Y. and Kaveh, M., "Formation of step images during anisotropic diffusion", *Image Processing, International Conference*, vol. 3, pp. 388-391, 1997.

[249] You, Y., Xu, W., Tannenbaum, A. and Kaveh, M., "Behavioral analysis of anisotropic diffusion in image processing", *Image Processing, IEEE Transactions*, vol. 5, Issue: 11, pp. 1539-1553, Nov. 1996.

[250] You, Y. and Kaveh, M., "Pyramidal image compression using anisotropic and error-corrected interpolation", Acoustics, Speech, and Signal Processing, ICASSP-96, Conference Proceedings, 1996 IEEE International Conference, pp. 1946 -1949, vol. 4, 1996.

[251] You, Y. and Kaveh, M., "Anisotropic blind image restoration", *Image Processing, International Conference*, vol. 2, pp. 461-464, 1996.

[252] You, Y. and Kaveh, M., "Ringing reduction in image restoration by orientation-selective regularization", *IEEE Signal Processing Letters*, vol. 3, Issue: 2, pp. 29-31, Feb. 1996.

[253] You, Y. and Kaveh, M., "A regularization approach to blind restoration of images degraded by shift-variant blurs", *Acoustics, Speech, and Signal Processing, ICASSP-95, 1995 International Conference*,vol. 4, pp. 2607-2610, 1995.

[254] You, Y., Xu, W., Kaveh, M. and Tannenbaum, A., "On ill-posed anisotropic diffusion models", *Image Processing, International Conference*, vol. 2, pp. 268-271, 1995.

[255] You, Y., Kaveh, M., Xu, W. and Tannenbaum, A., "Analysis and design of anisotropic diffusion for image processing", *Image Processing, Proceedings, ICIP-94, IEEE International Conference*, vol. 2, pp. 497-501, 1994.

[256] Yu, Y. and Acton, S.T., "Speckle reducing anisotropic diffusion", *Image Processing, IEEE Transactions*, vol. 11, Issue: 11, pp. 1260 -1270, Nov. 2002.

[257] Yu, Y. and Acton, S.T., "Segmentation of ultrasound imagery using anisotropic diffusion", *Signals, Systems and Computers, Conference Record of the Thirty-Fifth Asilomar Conference*, vol. 2, pp. 1151-1155, 2001.

[258] Yu, P., Decker, S. J., Lee, H., Sodini, C. G. and Wyatt, J. L., "CMOS Resistive Fuses for Image Smoothing And Segmentation", *IEEE Journal of Solid-State Circuits*, vol. 27, No. 4, pp. 545 -553, April, 1992.

[259] Zhang, Y. and Ma, S., "Unsupervised segmentation of noisy image in a multiscale framework", *Signal Processing Proceedings, WCCC-ICSP 2000, 5th International Conference*, vol. 2, pp. 905-909, 2000.

[260] Zhu, S. and Mumford, D., "GRADE: Gibbs reaction and diffusion equations", *Computer Vision, Sixth International Conference*, pp. 847-854, 1998.

[261] Zhu, S. and Mumford, D., "Prior learning and Gibbs reaction-diffusion", *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 19, Issue: 11, pp. 1236-1250, Nov. 1997.

Appendix A

Programs for TAD

The following programs were written as functions of "*cvlab*" [1]

.....TAD for 1 dimensional signals .....

```
/* tad1 Implementing float point 1 dimensional traditional
        anisotropic diffusion using diffusion function
        exp(-x*x/K/K).   */

/* source = input image for processing.
   result = output image after processing.
   iter   = number of iterations.
   xs     = the width of input image.
   thresh = value of threshold in percent of gradient range.
   wtime  = time for showing the intermediate results.
   accu   = resolution of filter */


/* prompt: source, result, iter, xs, thresh, wtime, accu
   format: %d %d %d %d %f %d %f
   menu:
   helpfile:
   prototype: int tad1 (int source, int result, int iter,
              int xs, float thresh, int wtime, float accu);   */


#include "cvcmd.h"
```

[1] "*cvlab*" is an image processing environment developed by Dr. P. H. Gregson

```
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define XS xs

void _tad1 (void)
{ int source, result, iter, xs, wtime;
  float  thresh, accu;

  source = *((int *) arglist[0]);
  result = *((int *) arglist[1]);
  iter   = *((int *) arglist[2]);
  xs     = *((int *) arglist[3]);
  thresh = *((float *) arglist[4]);
  wtime  = *((float *) arglist[5]);
  accu   = *((float *) arglist[6]);

  tad1 (source, result, iter, xs, thresh, wtime, accu);
}

int tad1 (int source, int result, int iter, int xs,
          float thresh, int wtime, float accu)
{ int  x, y, xsize, ysize, type;
  int   i, j, k, t1=100;
  long t=0,t2=0,k1;
  float a[XS],a1[XS], b[2], c[2], m, m1;
  time_t time1, time2;

  /*  i: number of pixels whose gradient magnitudes are large
         than the threshold defined by 'thresh"  */

  getsize (source, &xsize, &ysize, &type);
  form (result,0,xsize,ysize,type);

  i=(int)((1.0-thresh)*(xsize-1));



  /* read data from the input image */

  for (x=0;x<xsize;x++)
```

```
  { k=0;
    for (y=0;y<ysize;y++)
    if (getpix(source,x,y)==0)
         k++;
    a[x]=(float)k;
    a1[x]=a[x];
    putpix(k,result,x,0);
  }


/* get m, the threshold  */

for (x=1;x<xsize;x++)
   putpix(abs(getpix(result,x,0)-getpix(result,x-1,0)),result,x,1);
putpix(getpix(result,1,1),result,0,1);

while(i>0)
 {k=0;
  for (x=0; x<xsize; x++)
  if (getpix(result,x,1)>k)
      { k=getpix(result,x,1);
        j=x;
      }

  putpix(0,result, j, 1);
  i--;

  for (x=0; x<xsize; x++)
  if (k==getpix(result,x,1))
      { putpix(0,result, x,1);
        i--;
      }
 }

m=(float)k;


/* main program beginnig  */

/* j: stop diffusion process if the differences between
      successive smoothing processes are larger than 10 times
      continuously.
```

```
   k1: number of iterations.                                    */

j=0;

/* a1[x] holds the updated  */

for(k1=0;k1<iter;k1++)
  { for(x=1;x<xsize-1;x++)
    { b[0] = a[x+1]-a[x];
      b[1] = a[x-1]-a[x];

      for (i=0; i<2; i++)
           c[i]=exp(-pow(b[i]/m,2));

      m1=0.0;
      for (i=0; i<2; i++)
           m1+=b[i]*c[i];

      m1/=2.0;
      a1[x] += m1;
      if (fabs(m1)>accu)
          j++;
    }


++t;

/* control the time for showing temporary results */
time (&time1);
do {time (&time2);
           }while (((int) difftime(time2,time1)) < wtime);


/* continuing diffusion?  */

if (j==0)
  { if (t2+1==t)
      { t1--;
        printf("the %dth time total change = 0\n", 100-t1);
      }
    else
      { t1=99;
```

```
            printf("the first time total change = 0\n");
          }
        t2=t;
      }

  /* monitor the diffusion process */

  printf("T=%ld N=%d\n",t,j);

  /* stop diffusion process? */
  if (t1 >= 0)
      for (x=0;x<xsize;x++)
         a[x]=a1[x];
  else
      k1=iter;
 }
  return (TRUE);
}
```

.....TAD for 2 dimensional images .....

```
/* tad2 Implementing float point 2 dimensional traditional
        anisotropic diffusion using diffusion function
        exp(-x*x/K/K).  */

/* source = input image for processing.
   result = output image after processing.
   xsize  = the width of image to be processed.
   ysize  = the height of image to be processed.
   iter1  = if iterations are large than 100000.
   iter2  = times the image will be filtered.
   thresh = value of threshold in percent of gradient range.
   wtime  = time for showing the intermediate results.
   accu   = resolution of filter */


/* prompt: source, result, xsize, ysize, iter1, iter2, thresh,
           wtime, accu
```

```
   format: %d %d %d %d %d %d %f %d %f
   menu:
   helpfile:
   prototype: int tad2 (int source, int result, int xsize,
              int ysize, int iter1, int iter2, float thresh,
              int wtime, float accu); */


#include "cvcmd.h"
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define  XS xsize
#define  YS ysize


void _tad2 (void)
{ int source, result, xsize, ysize, iter1, iter2, wtime;
  float  thresh, accu;

  source = *((int *) arglist[0]);
  result = *((int *) arglist[1]);
  xsize  = *((int *) arglist[2]);
  ysize  = *((int *) arglist[3]);
  iter1  = *((int *) arglist[4]);
  iter2  = *((int *) arglist[5]);
  thresh = *((float *) arglist[6]);
  wtime  = *((int *) arglist[7]);
  accu   = *((float *) arglist[8]);


  tad2 (source, result, xsize, ysize, iter1, iter2, thresh,
        wtime, accu);
}

int tad2 (int source, int result, int xsize, int ysize, int iter1,
          int iter2, float thresh, int wtime, float accu)
{ int  x, y, xs, ys, type;
  int   i, j, k, k1, k2,t1=100,t2=0, t3=0;
  long t=0;
  float a[XS][YS], b[4], c[4],*p[3];
```

```
float m, m1, *buf;
time_t time1, time2;
char filtert[16]="times";
int *p1;
FILE *fp;


getsize (source, &xs, &ys, &type);
form (result,0,xs,ys,type);

/* allocate space for recording temporary data */

buf = (float *) farmalloc (3L*xs*sizeof(float));
if (buf == NULL)
   { response ("Unable to allocate buffer space for
     2D float point traditional anistropic diffusion");
     longjmp (jumpbuf, 1);
   }

for (x = 0; x < 3; x++)
    p[x] = buf+x*xs;

/*  k: number of pixels whose gradient magnitudes are large
       than the threshold defined by ``thresh"  */

k=(int)((1.0-thresh)*(xs-1)*(ys-1));

form (result,0,xs,ys,type);


/* get the gradient magnitude defined for edges */

for (x=1;x<xs-1;x++)
   for (y=1;y<ys-1;y++)
      putpix(getpix(source,x-1,y)+getpix(source,x+1,y)
             +getpix(source,x,y+1)+getpix(source,x,y-1)
             -4*getpix(source,x,y),result,x,y);

while(k>0)
 { k1=0;
   for (x=1; x<xs-1; x++)
      for (y=1; y<ys-1; y++)
```

```
          if (abs(getpix(result,x,y))>k1)
             { k1=abs(getpix(result,x,y));
               i=x;
               j=y;
             }

   putpix(0,result, i, j);
   k--;

   for (x=1; x<xs-1; x++)
      for (y=1; y<ys-1; y++)
         if (k1==abs(getpix(result,x,y)))
            { putpix(0,result, x,y);
              k--;
            }
 }

m = (float)k1;


/* Creating array for implementing float point operations */

for (x=0;x<xs; x++)
   for (y=0; y<ys; y++)
       a[x][y]=(float)getpix(source,x,y);


/*  main program   */

/* k: stop diffusion process if the differences between
      successive diffusion processes are larger than t1 times
      continuously.
   k1: in case if more than 100000 iter1s are required.
   k2: times of iter1s.   */

k=0;

for(k1=0;k1<iter1;k1++)
   for (k2=0; k2<iter2; k2++)
      { for(y=1;y<ys-1;y++)
         { for (x=1;x<xs-1;x++)
            { b[0] = a[x+1][y]-a[x][y];
```

```
        b[1] = a[x-1][y]-a[x][y];
        b[2] = a[x][y+1]-a[x][y];
        b[3] = a[x][y-1]-a[x][y];

        for (i=0;i<4; i++)
             c[i]=exp(-1.0*b[i]/m*b[i]/m);

        m1=0.0;
        for (i=0; i<4; i++)
                m1+=b[i]*c[i];

        m1/=4.0;
        *(p[1]+x)= a[x][y]+m1;

        if (fabs(m1)>accu)
              { k++;
                t2++;
              }

      }

    if (y>=2)
       { for (x=0;x<xs;x++)
         a[x][y-1]=*(p[0]+x);

         p[2]=p[0];
         p[0]=p[1];
         p[1]=p[2];
       }
    else
       { p[2]=p[0];
         p[0]=p[1];
         p[1]=p[2];
       }

    if (y==ys-2)
       { for (x=0;x<xs;x++)
    a[x][y]=*(p[0]+x);
       }

  }
```

```
/* monitor the process of diffusion */

++t;
printf("T=%ld; N=%d\n",t,t2);


/* control the interval of showing each result */

time (&time1);
do {time (&time2);
           }while (((int) difftime(time2,time1)) < wtime);


/* determining whether continue diffusion   */

if (k==0)
  { if (t3+1==t)
       { t1--;
         printf("The %dth time meet the accuracy requirement.
                \n", 100-t1);
       }
    else
       { t1 = 99;
         printf("The first time meet the accuracy requirement.\n");
       }
    t3=t;
          }

if (t1==0)
  { k1=iter1;
    k2=iter2;
   }
t2=0;
k=0;

}

/* Record iterations */

p1=filtert;
```

```
 if((fp = fopen(p1,"ab")) == NULL )
    { printf("cannot open file\n");
      exit(0);
    }

 fprintf(fp,"%ld", t-100);
 fprintf(fp,"\n",0);
 fclose(fp);

 return (TRUE);
}
```

Appendix B

Program for AD by CFs of "Mono" flux

The following program was written as a function of "*cvlab*". [1]

```
/* adi Implementing 1D AD with the CF producing only increasing flux*/

/* itertion = times the image will be filtered, set at the beginning.
 edge = the threshold for edge (percentage of pixels).
 accu = smoothing is stopped if the difference is less than the
accu for 100 times */

/* prompt: source, result, iteration, edge, accu
format: %d %d %d %f %f
menu:
helpfile:
prototype: int ydfb1 (int source, int result, int iteration,
float edge, float accu); */


#include "cvcmd.h"
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define K 1

void _adi (void)
```

[1] "*cvlab*" is an image processing environment developed by Dr. P. H. Gregson

```
      {  int source, result, iteration;
 float  edge, accu;

source = *((int *) arglist[0]);
result = *((int *) arglist[1]);
iteration = *((int *) arglist[2]);
edge = *((float *) arglist[3]);
accu = *((float *) arglist[4]);

adi (source, result, iteration, edge, accu);
      }

int adi (int source, int result, int iteration,
      float edge, float accu)
    {   int  x, y, xsize, ysize, type;
int   i, j, k, k1, k2, t1=99;
long t=0,t2;
float a[300],a1[300], b[2], c[2], m, m1, w;
time_t time1, time2;

/* i: how many pixels are treated as edges: (1-edge)% of total pixels  */

getsize (source, &xsize, &ysize, &type);
form (result,0,xsize,ysize,type);

i=(int)((1.0-edge)*(xsize+1));

for (x=0;x<xsize;x++)
 { k=0;
   for (y=0;y<ysize;y++)
     if (getpix(source,x,y)==0) k++;
   a[x]=(float)k;
   a1[x]=a[x];
   putpix(k,result,x,0);
 }


/* get m, the threshold  */

for (x=1;x<xsize;x++)
     putpix(abs(getpix(result,x,0)-getpix(result,x-1,0)),result,x,1);
putpix(getpix(result,1,1),result,0,1);
```

```
while(i>0)
{k=0;
 for (x=0; x<xsize; x++)
    if (getpix(result,x,1)>k)   {k=getpix(result,x,1); j=x;}

 putpix(0,result, j, 1);
 i--;

 for (x=0; x<xsize; x++)
 if (k==getpix(result,x,1))   { putpix(0,result, x,1); i--;  }
}

m=(float)k;


/*  main program beginnig  */
for (k2=0;k2<K;k2++)
  for (k=0; k<iteration; k++) /* k control the times of iterations  */
    { for(x=1;x<xsize-1;x++)   /* calculating b[2]=dEW, c[2]=cEW */
      { b[0] = a[x+1]-a[x];
        b[1] = a[x-1]-a[x];

for (i=0; i<2; i++)
    if (fabs(b[i])>m) c[i]=0.0;
    else c[i] = 1.0;

m1=0.0;
for (i=0; i<2; i++) m1+=b[i]*c[i];

a1[x] += m1/4.0;
      }

   b[0] = a[1]-a[0];
   b[1] = a[xsize-2]-a[xsize-1];

   for (i=0;i<2; i++)
      if (fabs(b[i])>m) c[i]=0.0;
      else c[i]=1.0;

   a1[0] += b[0]*c[0]/2.0;
   a1[xsize-1] += b[1]*c[1]/2.0;
```

```
/* up to new, a1[x] is updated while a[x] still contains data of
   last time */

/* continuing diffusion?  */

j=0;
for (i=0;i<xsize;i++)
    if (fabs(a1[i]-a[i]) >= accu)   j++;

if (j==0)
    { if (t2+1==t)
{ t1--;
  printf("the %dth time = 0\n", 99-t1);
}
      else
{ t1=9;
  printf("the first time = 0\n");
}
      t2=t;
    }

printf("T=%ld N=%d accu=%f\n",t,j,accu);

if (t1 >= 0)
    for (x=0;x<xsize;x++) a[x]=a1[x];
else {k=iteration;k2=K;}

      } /* End of the main programing */

/*  saving the result          */
        form (result,1,xsize,ysize,type);
        for (x=0;x<xsize;x++)
            { i=(int)(floor(a1[x]+0.16));
              for (y=0; y<i; y++) putpix(0,result,x,y);
            }

return (TRUE);
}
```

Appendix C

Programs for IAD

The following programs were written as functions of "*cvlab*" [1]

..... IAD for 1 dimensional signals .....

```
/* iad1 Implementing float point 1 dimensional idempotent
        anisotropic diffusion using diffusion function
        exp(-x*x/K/K).   */

/* source = input image for processing.
   result = output image after processing.
   iter   = number of iterations.
   xs     = the width of input image.
   thresh = value of threshold in percent of gradient range.
   wtime  = time for showing the intermediate results.
   accu   = resolution of filter */


/* prompt: source, result, iter, xs, thresh, wtime, accu
   format: %d %d %d %d %f %d %f
   menu:
   helpfile:
   prototype: int iad1 (int source, int result, int iter,
              int xs, float thresh, int wtime, float accu);   */
```

[1] "*cvlab*" is an image processing environment developed by Dr. P. H. Gregson

```
#include "cvcmd.h"
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define XS xs

void _iad1 (void)
{ int source, result, iter, xs, wtime;
  float  thresh, accu;

  source = *((int *) arglist[0]);
  result = *((int *) arglist[1]);
  iter   = *((int *) arglist[2]);
  xs     = *((int *) arglist[3]);
  thresh = *((float *) arglist[4]);
  wtime  = *((float *) arglist[5]);
  accu   = *((float *) arglist[6]);

  iad1 (source, result, iter, xs, thresh, wtime, accu);
}

int iad1 (int source, int result, int iter, int xs,
          float thresh, int wtime, float accu)
{ int  x, y, xsize, ysize, type;
  int   i, j, k, t1=100;
  long t=0,t2=0,k1;
  float a[XS],a1[XS], b[2], c[2], m, m1;
  time_t time1, time2;

  /*  i: number of pixels whose gradient magnitudes are large
         than the threshold defined by 'thresh"  */

  getsize (source, &xsize, &ysize, &type);
  form (result,0,xsize,ysize,type);

  i=(int)((1.0-thresh)*(xsize-1));



  /* read data from the input image */
```

```
for (x=0;x<xsize;x++)
  { k=0;
    for (y=0;y<ysize;y++)
    if (getpix(source,x,y)==0)
         k++;
    a[x]=(float)k;
    a1[x]=a[x];
    putpix(k,result,x,0);
  }


/* get m, the threshold  */

for (x=1;x<xsize;x++)
   putpix(abs(getpix(result,x,0)-getpix(result,x-1,0)),result,x,1);
putpix(getpix(result,1,1),result,0,1);

while(i>0)
 {k=0;
  for (x=0; x<xsize; x++)
  if (getpix(result,x,1)>k)
      { k=getpix(result,x,1);
        j=x;
      }

  putpix(0,result, j, 1);
  i--;

  for (x=0; x<xsize; x++)
  if (k==getpix(result,x,1))
      { putpix(0,result, x,1);
        i--;
      }
 }

m=(float)k;


/* main program beginnig  */

/* j: stop diffusion process if the differences between
      successive smoothing processes are larger than 10 times
```

```
      continuously.
   k1: number of iterations.                                    */

j=0;

/* a1[x] holds the updated  */

for(k1=0;k1<iter;k1++)
  { for(x=1;x<xsize-1;x++)
    { b[0] = a[x+1]-a[x];
      b[1] = a[x-1]-a[x];

      for (i=0; i<2; i++)
           c[i]=(1-pow(b[i]/m,2))*exp(-pow(b[i]/m,2));

      m1=0.0;
      for (i=0; i<2; i++)
           m1+=b[i]*c[i];

      m1/=2.0;
      a1[x] += m1;
      if (fabs(m1)>accu)
          j++;
    }


++t;

/* control the time for showing temporary results */
time (&time1);
do {time (&time2);
          }while (((int) difftime(time2,time1)) < wtime);


/* continuing diffusion?  */

if (j==0)
  { if (t2+1==t)
      { t1--;
        printf("the %dth time total change = 0\n", 100-t1);
      }
    else
```

```
        { t1=99;
          printf("the first time total change = 0\n");
        }
      t2=t;
    }

  /* monitor the diffusion process */

  printf("T=%ld N=%d\n",t,j);

  /* stop diffusion process? */
  if (t1 >= 0)
      for (x=0;x<xsize;x++)
         a[x]=a1[x];
  else
      k1=iter;
 }
  return (TRUE);
}
```

..... IAD for 2 dimensional images .....

This program was used also as the code for "Mixed" flux method and gradient-magnitude-based anisotropic diffusion (GMAD).

```
/* iad2 Implementing float point 2 dimensional idempotent
        anisotropic diffusion using diffusion function
        exp(-x*x/K/K).  */

/* source = input image for processing.
   result = output image after processing.
   xsize  = the width of image to be processed.
   ysize  = the height of image to be processed.
   iter   = times the image will be filtered.
   thresh = value of threshold in percent of gradient range.
   wtime  = time for showing the intermediate results.
   accu   = resolution of filter */
```

```
/* prompt: source, result, xsize, ysize, iter, thresh,
           wtime, accu
   format: %d %d %d %d %d %f %d %f
   menu:
   helpfile:
   prototype: int iad2 (int source, int result, int xsize,
              int ysize, int iter, float thresh,
              int wtime, float accu); */


#include "cvcmd.h"
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define  XS xsize
#define  YS ysize


void _iad2 (void)
{ int source, result, xsize, ysize, iter, wtime;
  float  thresh, accu;

  source = *((int *) arglist[0]);
  result = *((int *) arglist[1]);
  xsize  = *((int *) arglist[2]);
  ysize  = *((int *) arglist[3]);
  iter   = *((int *) arglist[4]);
  thresh = *((float *) arglist[5]);
  wtime  = *((int *) arglist[6]);
  accu   = *((float *) arglist[7]);


  iad2 (source, result, xsize, ysize, iter, thresh,
        wtime, accu);
}

int iad2 (int source, int result, int xsize, int ysize, int iter,
          float thresh, int wtime, float accu)
{ int  x, y, xs, ys, type;
```

```
int   i, j, k, k1,t1=100,t2=0;
long t=0,k2,t3;
float a[XS][YS], b[4], c[4],*p[3];
float m, m1, *buf;
time_t time1, time2;
char filtert[16]="times";
int *p1;
FILE *fp;


getsize (source, &xs, &ys, &type);
form (result,0,xs,ys,type);

/* allocate space for recording temporary data */

buf = (float *) farmalloc (3L*xs*sizeof(float));
if (buf == NULL)
   { response ("Unable to allocate buffer space for
     2D float point traditional anistropic diffusion");
     longjmp (jumpbuf, 1);
   }

for (x = 0; x < 3; x++)
    p[x] = buf+x*xs;

/*  k: number of pixels whose gradient magnitudes are large
       than the threshold defined by ``thresh"  */

k=(int)((1.0-thresh)*(xs-1)*(ys-1));

form (result,0,xs,ys,type);


/* get the gradient magnitude defined for edges */

for (x=1;x<xs-1;x++)
   for (y=1;y<ys-1;y++)
      putpix(getpix(source,x-1,y)+getpix(source,x+1,y)
             +getpix(source,x,y+1)+getpix(source,x,y-1)
             -4*getpix(source,x,y),result,x,y);

while(k>0)
```

```
{ k1=0;
  for (x=1; x<xs-1; x++)
     for (y=1; y<ys-1; y++)
        if (abs(getpix(result,x,y))>k1)
           { k1=abs(getpix(result,x,y));
             i=x;
             j=y;
           }

  putpix(0,result, i, j);
  k--;

  for (x=1; x<xs-1; x++)
     for (y=1; y<ys-1; y++)
        if (k1==abs(getpix(result,x,y)))
           { putpix(0,result, x,y);
             k--;
           }
}

m = (float)k1;


/* Creating array for implementing float point operations */

for (x=0;x<xs; x++)
   for (y=0; y<ys; y++)
       a[x][y]=(float)getpix(source,x,y);


/*  main program   */

/* k: stop diffusion process if the differences between
      successive diffusion processes are larger than t1 times
      continuously.
   k2: number of iterations.   */

k=0;

  for (k2=0; k2<iter; k2++)
      { for(y=1;y<ys-1;y++)
         { for (x=1;x<xs-1;x++)
```

```
    { b[0] = a[x+1][y]-a[x][y];
      b[1] = a[x-1][y]-a[x][y];
      b[2] = a[x][y+1]-a[x][y];
      b[3] = a[x][y-1]-a[x][y];

      for (i=0;i<4; i++)
         c[i]=(1.0-2*b[i]/m*b[i]/m)*exp(-b[i]/m*b[i]/m);

      m1=0.0;
      for (i=0; i<4; i++)
              m1+=b[i]*c[i];

      m1/=4.0;
      *(p[1]+x)= a[x][y]+m1;

      if (fabs(m1)>accu)
            { k++;
              t2++;
            }

    }

   if (y>=2)
      { for (x=0;x<xs;x++)
        a[x][y-1]=*(p[0]+x);

        p[2]=p[0];
        p[0]=p[1];
        p[1]=p[2];
      }
   else
      { p[2]=p[0];
        p[0]=p[1];
        p[1]=p[2];
      }

   if (y==ys-2)
      { for (x=0;x<xs;x++)
   a[x][y]=*(p[0]+x);
      }

}
```

```
/* monitor the process of diffusion */

++t;
printf("T=%ld; N=%d\n",t,t2);


/* control the interval of showing each result */

time (&time1);
do {time (&time2);
          }while (((int) difftime(time2,time1)) < wtime);


/* determining whether continue diffusion   */

if (k==0)
  { if (t3+1==t)
       { t1--;
         printf("The %dth time meet the accuracy requirement.
                \n", 100-t1);
       }
    else
       { t1 = 99;
         printf("The first time meet the accuracy requirement.\n");
       }
    t3=t;
          }

if (t1==0)
    k2=iter;

t2=0;
k=0;

}

/* Record iterations */

p1=filtert;
```

```
 if((fp = fopen(p1,"ab")) == NULL )
    { printf("cannot open file\n");
      exit(0);
    }

 fprintf(fp,"%ld", t-100);
 fprintf(fp,"\n",0);
 fclose(fp);

 return (TRUE);
}
```

Appendix D

Programs for GFAB

The following programs were written as functions of "*cvlab*" [1]

.....GFAB with no fidelity term added for 1D signals .....

```
/* Implementing forward-and-backward diffusion proposed by Gilboa.
   The fidelity term is not employed  */

/*  itertion = times the image will be filtered, set at the beginning.
    accu = smoothing is stopped if the difference is less than the
           accu for 100 times */


/* prompt: source, result, iteration, accu
format: %d %d %d %f
menu:
helpfile:
prototype: int gfab1 (int source, int result, int iteration,
           float accu); */

#include "cvcmd.h"
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define K 1
```

[1] "*cvlab*" is an image processing environment developed by Dr. P. H. Gregson

```
void _gfab1 (void)
     {  int source, result, iteration;
 float  accu;

source = *((int *) arglist[0]);
result = *((int *) arglist[1]);
iteration = *((int *) arglist[2]);
accu = *((float *) arglist[3]);

gfab1 (source, result, iteration, accu);
     }

int gfab1 (int source, int result, int iteration, float accu)
    {   int  x, y, xsize, ysize, type;
int   i, j, i1, k, k1, k2, t1=100;
long t=0,t2, k3=0;
float a[300],a1[300], a2[300], b[2], c[2], m, m1, m2, m3, m4, w;
time_t time1, time2;

getsize (source, &xsize, &ysize, &type);
form (result,0,xsize,ysize,type);

/* get m, the MAG  */

        for (x=0;x<xsize;x++)
         { k=0;
           for (y=0;y<ysize;y++)
             if (getpix(source,x,y)==0) k++;
           a[x]=(float)k;
           a1[x]=a[x];
   a2[x]=a[x];
           putpix(k,result,x,0);
         }

for (x=1;x<xsize;x++)
     putpix(abs(getpix(result,x,0)-getpix(result,x-1,0)),result,x,1);
putpix(getpix(result,1,1),result,0,1);

for (x=0; x<xsize; x++)
    k3 += getpix(result,x,1);
```

```
k = k3/xsize;
printf("MAG=%f\n", m=(float)k);
m1 = 2.0*m;
m2 = 4.0*m;
w = m;
m3 = 0.5*m1/(m2+w);

/*  main program beginnig  */
j=0;
for (k2=0;k2<K;k2++)
  for (k=0; k<iteration; k++) /* k control the times of iterations  */
   {for(x=1;x<xsize-1;x++)  /* calculating b[2]=dEW, c[2]=cEW */
     { b[0] = a[x+1]-a[x];
       b[1] = a[x-1]-a[x];

for (i=0; i<2; i++)
    c[i] = 1.0/(1.0+b[i]/m1*b[i]/m1*b[i]/m1*b[i]/m1)
           - m3/(1.0+(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w);

m4=0.0;
for (i=0; i<2; i++) m4+=b[i]*c[i];

a1[x] += m4/4.0;
      }

   b[0] = a[1]-a[0];
   b[1] = a[xsize-2]-a[xsize-1];

   for (i=0;i<2; i++)
    c[i] = 1.0/(1.0+b[i]/m1*b[i]/m1*b[i]/m1*b[i]/m1)
           - m3/(1.0+(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w);

   a1[0] += b[0]*c[0]/2.0;
   a1[xsize-1] += b[1]*c[1]/2.0;

/* up to new, a1[x] is updated while a[x] still contains data of
   last time */

/* continuing diffusion?  */
for(x=0;x<xsize;x++)
if(fabs(a1[x]-a[x])>accu) j++;
```

```
if (j==0)
    { if (t2+1==t)
{ t1--;
  printf("the %dth time = 0\n", 100-t1);
}
      else
{ t1=99;
  printf("the first time = 0\n");
}
      t2=t;
    }

if (t1 >= 0)
    for (x=0;x<xsize;x++) { a[x]=a1[x]; j=0;}
else {k=iteration;k2=K;}
      } /* End of the main programing */


/*  display the result */
form (result,1,xsize, ysize,type);

i1=0;
for (x=0;x<xsize;x++)
            if ((int)(floor(a1[x]+0.16)) < i1)
i1 = (int)(floor(a1[x]+0.06));

i1=abs(i1);

if (i1==0)
          {for (x=0;x<xsize;x++)
             { i=(int)(floor(a1[x]+0.06));
for (y=0; y<i; y++) putpix(0,result,x,y);
             }
           }
else
      {for (x=0;x<xsize;x++)
             { i=(int)(floor(a1[x]+0.06));
          if (i>0)
          { for (y=i1; y<i+i1; y++) putpix(0,result,x,y);}
        else
          { for (y=i1-1; y>i1-abs(i); y--) putpix(0,result,x,y);}
      }
```

```
            }
return (TRUE);
}
```

.....GFAB with fidelity term added for 1D signal .....

```
/* Implementing forward-and-backward diffusion proposed by Gilboa
   with the fidelity term "A" added  */

/* itertion = times the image will be filtered, set at the beginning.
 accu = smoothing is stopped if the difference is less than the
accu for 100 times */


/* prompt: source, result, iteration, accu
format: %d %d %d %f
menu:
helpfile:
prototype: int gfab2 (int source, int result, int iteration,
                                    float accu); */


#include "cvcmd.h"
#include "math.h"
#include <stdlib.h>
#include "stdio.h"
#include <time.h>
#define K 1
#define A 0.1

void _gfab2 (void)
      {  int source, result, iteration;
 float  accu;

source = *((int *) arglist[0]);
result = *((int *) arglist[1]);
iteration = *((int *) arglist[2]);
accu = *((float *) arglist[3]);
```

```
gfab2 (source, result, iteration, accu);
      }

int gfab2 (int source, int result, int iteration, float accu)
    {   int  x, y, xsize, ysize, type;
int   i, j, i1, k, k1, k2, t1=100;
long t=0,t2, k3=0;
float a[300],a1[300], a2[300], b[2], c[2], m, m1, m2, m3, m4, m5, w;
time_t time1, time2;

getsize (source, &xsize, &ysize, &type);
form (result,0,xsize,ysize,type);


/* get m, the MAG  */

        for (x=0;x<xsize;x++)
         { k=0;
           for (y=0;y<ysize;y++)
             if (getpix(source,x,y)==0) k++;
           a[x]=(float)k;
           a1[x]=a[x];
   a2[x]=a[x];
           putpix(k,result,x,0);
         }

for (x=1;x<xsize;x++)
     putpix(abs(getpix(result,x,0)-getpix(result,x-1,0)),result,x,1);
putpix(getpix(result,1,1),result,0,1);

for (x=0; x<xsize; x++)
    k3 += getpix(result,x,1);

k = k3/xsize;
printf("MAG=%f\n", m=(float)k);
m1 = 2.0*m;
m2 = 4.0*m;
w = m;
m3 = 0.5*m1/(m2+w);

/*  main program beginnig  */
```

```
j=0;
for (k2=0;k2<K;k2++)
  for (k=0; k<iteration; k++) /* k control the times of iterations */
   {for(x=1;x<xsize-1;x++)  /* calculating b[2]=dEW, c[2]=cEW */
     { b[0] = a[x+1]-a[x];
b[1] = a[x-1]-a[x];

for (i=0; i<2; i++)
    c[i] = 1.0/(1.0+b[i]/m1*b[i]/m1*b[i]/m1*b[i]/m1)
           - m3/(1.0+(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w);

m4=0.0;
for (i=0; i<2; i++) m4+=b[i]*c[i];

m5 = A*(a[x]-a2[x]);
m5 = m4/4.0 -m5;
a1[x] += m5;
      }

   b[0] = a[1]-a[0];
   b[1] = a[xsize-2]-a[xsize-1];

   for (i=0;i<2; i++)
    c[i] = 1.0/(1.0+b[i]/m1*b[i]/m1*b[i]/m1*b[i]/m1)
           - m3/(1.0+(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w*(b[i]-m2)/w);

   m5 = A*(a[0]-a2[0]);
   m5 = b[0]*c[0]/2.0 -m5;
   a1[0] += m5;
   m5 = A*(a[xsize-1]-a2[xsize-1]);
   m5 = b[1]*c[1]/2.0 -m5;
   a1[xsize-1] += m5;

/* up to new, a1[x] is updated while a[x] still contains data of last time
*/

/* continuing diffusion?  */
for(x=0;x<xsize;x++)
if(fabs(a1[x]-a[x])>accu) j++;

if (j==0)
    { if (t2+1==t)
```

```
{ t1--;
  printf("the %dth time = 0\n", 100-t1);
}
      else
{ t1=99;
  printf("the first time = 0\n");
}
      t2=t;
    }

if (t1 >= 0)
    for (x=0;x<xsize;x++) { a[x]=a1[x]; j=0;}
else {k=iteration;k2=K;}

      } /* End of the main programing */

/*  Enter the part of display  */
form (result,1,xsize, ysize,type);

i1=0;
for (x=0;x<xsize;x++)
            if ((int)(floor(a1[x]+0.16)) < i1)
i1 = (int)(floor(a1[x]+0.06));

i1=abs(i1);

        if (i1==0)
           {for (x=0;x<xsize;x++)
              { i=(int)(floor(a1[x]+0.06));
for (y=0; y<i; y++) putpix(0,result,x,y);
              }
            }
else
           {for (x=0;x<xsize;x++)
              { i=(int)(floor(a1[x]+0.06));
if (i>0)
                   { for (y=i1; y<i+i1; y++) putpix(0,result,x,y);}
else
                   { for (y=i1-1; y>i1-abs(i); y--) putpix(0,result,x,y);}
              }
            }
```

```
return (TRUE);
}
```

Appendix E

Program for IDCAD

The following program was written as a function of "*cvlab*" [1]

```
/* dcad implementing anisotropic diffusion conducted by
        direction consistency based edge extimator   */

/* prompt: source, result, iter,xsize, ysize, thresh resolu
   format: %d %d %d %d %d %f %f
   helpfile:
   menu:
   prototype: int dcad (int source, int result, int iter, int xsize,
                        int ysize, float thresh, float resolu) ;  */

#include "cvcmd.h"
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define  XS xsize
#define  YS ysize
#define  OS 10

void _dcad (void)
{ int source, result, iter, xsize, ysize;
  float thresh, resolu;

  source = *((int *) arglist[0]);
  result = *((int *) arglist[1]);
  iter   = *((int *) arglist[2]);
```

[1] "*cvlab*" is an image processing environment developed by Dr. P. H. Gregson

```
  xsize  = *((int *) arglist[3]);
  ysize  = *((int *) arglist[4]);
  thresh = *((float *) arglist[5]);
  resolu = *((float *) arglist[6]);
  dcad (source, result, iter, xsize, ysize, thresh, resolu);
}


int dcad (int source, int result, int iter, int xsize, int ysize,
          float thresh, float resolu)
{ int x, y, xs, ys, type, n, n1, t1=100, k1;
  long t=0, k;
  float deltax[XS][YS], deltay[XS][YS], coeff[XS][YS], a[XS][YS];
  float sumx, sumy, sumv, b[4], c[4], diffu[XS][YS];


  getsize (source, &xs, &ys, &type);
  form (result,0,xs,ys,type);


  /* read in image for float point operation */

  for (x=0;x<XS;x++)
     for (y=0;y<YS;y++)
          a[x][y]=(float)getpix(source,x,y);


  /* calculate deltax, deltay and magnitude array */

  for (x=1;x<XS-1;x++)
     for (y=1;y<YS-1;y++)
        { deltax[x][y]=a[x+1][y-1]+2.0*a[x+1][y]+a[x+1][y+1]
                       -a[x-1][y-1]-2.0*a[x-1][y]-a[x-1][y+1];
          deltay[x][y]=a[x-1][y+1]+2.0*a[x][y+1]+a[x+1][y+1]
                       -a[x-1][y-1]-2.0*a[x][y-1]-a[x+1][y-1];
        }


  /* calculate coefficient array */

  for (x=2;x<XS-2;x++)
     for (y=2;y<YS-2;y++)
```

```
    { sumx = 0;
      sumy = 0;
      sumv = 0;
      for (n=-1;n<=1;n++)
         for (n1=-1;n1<=1;n1++)
             { sumx += deltax[x+n][y+n1];
               sumy += deltay[x+n][y+n1];
               sumv += sqrt(deltax[x+n][y+n1]*deltax[x+n][y+n1]
                       +deltay[x+n][y+n1]*deltay[x+n][y+n1]);
             }
      coeff[x][y] = sqrt(sumx*sumx+sumy*sumy)/(sumv+OS);
    }


/* calculate the DC array */

for (x=0;x<XS;x++)
   for (y=0;y<YS;y++)
      { sumx = coeff[x][y]/thresh;
        diffu[x][y]=(1.0-sumx*sumx)*exp(-1.0*sumx*sumx);
       }


/* opearte diffusion */

for(k=0;k<iter;k++)
   { k1=0;
     for (x=2;x<XS-2;x++)
        for(y=2;y<YS-2;y++)
          { b[0]=a[x+1][y]-a[x][y];
            b[1]=a[x-1][y]-a[x][y];
            b[2]=a[x][y-1]-a[x][y];
            b[3]=a[x][y+1]-a[x][y];

            c[0]=(diffu[x][y]+diffu[x+1][y])/2.0;
            c[1]=(diffu[x][y]+diffu[x-1][y])/2.0;
            c[2]=(diffu[x][y]+diffu[x][y-1])/2.0;
            c[3]=(diffu[x][y]+diffu[x][y+1])/2.0;

            sumx = 0;
            for (n=0;n<4;n++)
                 sumx += b[n]*c[n];
```

```
            sumx = sumx/4.0;

            if (sumx > resolu)
                  k1++;

            coeff[x][y] = a[x][y]+sumx;
          }



/* monitor the diffusion process */

printf("T=%d, N=%d\n", k, k1);


/* stop diffusion process? */

if (k1==0)
  { if (t+1==k)
       { t1--;
         printf("The %dth time meet the accuracy requirement.
                 \n", 100-t1);
       }
    else
       { t1 = 99;
         printf("The first time meet the accuracy requirement.\n");
       }
    t=k;
  }

k1=0;

if (t1==0)
    k=iter;
else  { for(x=0;x<XS;x++)
           for(y=0;y<YS;y++)
              { a[x][y]=coeff[x][y];
                if (fabs(diffu[x][y])>0.001)
                       diffu[x][y]*=(1-exp(-10*diffu[x][y]));
                else diffu[x][y] = 0;
              }
      }
```

```
 }


 return (TRUE);
}
```