INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Experiments in Image Enhancement Using Biological and Artificial Neural Networks

by

Lesa M. Kennedy

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

1998

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

1/17/95
Date

Chair of Examining Committee

7/15/93
Date

Executive Officer

PROFESSOR JOSEPH BARBA

PROFESSOR MICHAEL CONNER

PROFESSOR SRINIVASA VEMURU

DR. HAISHAN WU

Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

Experiments in Image Enhancement Using Biological and Artificial Neural Networks

by

Lesa M. Kennedy

Advisor: Professor Mitra Basu

In this thesis, we report the results of computer experiments carried out to enhance digital images. We consider two areas of image processing: edge/boundary detection and image restoration. In our research on edge detection, we use a special line-weight function (LWF) which is a combination of zero- and second-order Hermite functions. We are motivated by physiological evidence reported in [78] that visual receptive fields are shaped like the sum of a Gaussian function and its Laplacian. This function can also be derived mathematically when the contrast sensitivity experiments of psychophysics are posed as a eigenvalue problem [71]. We provide experimental and mathematical proof that the LWF produces only authentic scale space contours (i.e. it does not detect phantom edges). We also show that the LWF has an excellent localization property. Performance comparison between the Laplacian of Gaussian and LWF operators is presented in a number of computer simulations.

We extend our work from edge to boundary detection through the use of projection

pursuit learning networks (PPLN). PPLNs have been used in many fields of research but have not been widely used in image processing. We demonstrate how a PPLN may be used to produce more continuous boundaries when presented with broken edge segments. We also propose the application of PPLN to deblurring a degraded image when little or no *apriori* information about the blur is available. The PPLN was successful at developing an inverse blur filter to enhance blurry images.

Acknowledgements

I would like to thank Dr. Mitra Basu, my research advisor, for her guidance and invaluable contributions to the completion and success of this thesis.

Thanks to Dr. Rolston Jeremiah, Dr. Michael Lewis and all my colleagues at the City College of New York.

Thanks to Dr. Ronald Brown, Dr. Joseph Barba, Dr. Richard Mammone and all the other members of the faculty whose help have made the completion of this thesis possible.

Thanks to the NSF, CASI and Dean Ramona Brown of the PRES program for their support through this thesis.

Thanks to my parents and the rest of the family for all their love and support.

Contents

List of Figures

Chapter 1

Introduction

The goal of this thesis is to develop computational methods for enhancing important features in images. We propose two methods to help us accomplish this goal: the first is modeled after the Human Visual System (HVS), and the second involve a neural network-like technique.

Two reasons motivated us to take the approach of modeling the human visual system and discuss each briefly.

- Humans have always been superior at recognizing images in comparison to existing algorithmic approaches.
- In most image communication systems, the quality of an image is judged by human viewers.

Certain aspects of image recognition and understanding create formidable tasks for computers. Under adverse situations (e.g., noise, incomplete image, poor light) humans are far better at recognition than computers even with their massive computational power. In the recent past, there have been attempts to take a somewhat different approach to achieve better performance by computers. These attempts may be broadly categorized into two groups, namely:

- **Model-free Approach:** systems with the ability to learn from the environment (e.g., neural network with supervised learning) and
- **Model-based Approach:** systems which mathematically formulate the human visual system.

The second reason played a major role in initiating this investigation. The first revolution in culture and knowledge, stimulated by mass production of printed word, has been superseded by a second revolution fueled by the broadcast of electronic images. The advance of technology for the processing and display of electronic imagery has in recent years offered the possibility of radically new means of communicating visual information. These new means range from commercial high-definition television (HDTV) to packet video, picture phone system, teleconferencing, multimedia, high-density optical digital discs, so-called personal digital assistant, and advanced image communication systems for space, aeronautics, medical, military, industrial, and commercial applications. Bandwidth, memory and computational resources are

the limiting factors for all these systems and in every instance directly affect cost, quality and practicality. In many of these applications the quality of the final image is evaluated by humans.

1.1 Overview of the thesis

We propose a mathematical formulation (the combination of hermite functions) of experimental results from psychophysics and neurophysiology for image enhancement. We show that this method is superior at detecting edges, with precise location information in real images, in comparison to similar edge detection methods. It is computationally as efficient as other existing methods. We propose a novel method to connect these edges to produce a continuous boundary using a highly promising and rather unknown (in the image processing research community) technique: *Projection Pursuit Learning*. We also apply this method to the restoration of images degraded by blurring. The projection pursuit approach is the basis of this thesis. It unifies the edge detection and the boundary detection methods, provides a platform to combine model-free and model-based approaches and has applications in two areas of image processing.

The objective of this thesis has been stated. Chapter 2 begins by mentioning some of the common problems encountered in edge detection. This is followed by a review of

the properties of the Gaussian filter and edge detection techniques which employ this filter. Chapter 3 introduces the proposed method for detecting edges and presents the results with one- and two-dimensional data. In Chapter 4, objective measures are utilized to assess the performance of the proposed edge enhancing operator. Chapter 5 introduces the concept of projection pursuit learning and reviews several works which use this technique. In Chapter 6, we apply projection pursuit learning to edge detection and image restoration, and present the results of several simulations. Chapter 7 summarizes the thesis and outlines a plan for future work.

Chapter 2

Edge Detection - A Brief Overview

The detection of edges in an image has been an important problem in image processing for more than twenty years. In a gray level image, an edge may be defined as a sharp change in intensity. Edge detection is the process which detects the presence and locations of these intensity transitions. The edge representation of an image drastically reduces the amount of data to be processed, yet it retains important information about the shapes of objects in the scene. This description of an image is easy to integrate into a large number of object recognition algorithms used in computer vision and other image processing applications.

Over the years, many methods have been proposed for detecting edges in images. Some of the earlier methods, such as the Sobel and Prewitt detectors [59, 65], used

local gradient operators which only detected edges having certain orientations and performed poorly when the edges were blurred and noisy. Since then more sophisticated operators have been developed to provide some degree of immunity to noise, and to be non-directional. The majority of these are linear operators that are derivatives of some sort of smoothing filter. Median filtering was used in [77] to reduce noise and preserve edges. Shen and Castan [69] used a symmetrical exponential filter in edge detection. However, since it was originally proposed by Marr and Hildreth in 1980 [54], the Gaussian filter is by far the most widely used smoothing filter in edge detection. The reasons for this are presented in a later section.

Since the Gaussian filter is so commonly used, in this chapter we review several Gaussian-based edge detection techniques. In Section 2.1 we outline some of the major problems encountered during an edge detection process. Section 2.2 discusses the features of the Gaussian filter that have contributed to its popularity in edge detection. In Section 2.3, we present a discussion of the two most well-known Gaussian edge operators - the Marr-Hildreth and Canny detectors. We also review several other detectors that have developed in more recent years. Conclusions are given in Section 2.4.

2.1 The problems of edge detection

The separation of a scene into object and background is an essential step in image interpretation. This is a process that is carried out effortlessly by the human visual system. but when computer vision algorithms are designed to mimic this action several problems can be encountered. This section describes some of the problems involved in detecting and localizing edges.

Due to the presence of noise and quantization of the original image. during edge detection it is possible to locate intensity changes where edges do not exist. For similar reasons. it is also possible to completely miss existing edges. The degree of success of an edge detector depends on its ability to accurately locate true edges.

Edge localization is another problem encountered in edge detection. The addition of noise to an image can cause the position of the detected edge to be shifted from its true location. The ability of an edge detector to locate in noisy data an edge that is as close as possible to its true position in the image is an important factor in determining its performance.

Another difficulty in any edge detection system arises from the fact that the sharp intensity transitions which indicate an edge are sharp because of their high frequency components. As a result. any linear filtering or smoothing performed on these edges to suppress noise will also blur the significant transitions. However. some form of

smoothing is necessary since edge detection depends on differentiating the image function and this amplifies all high frequency components of the signal, including those of the noise. Low pass filters are the most widely used smoothing filters. The amount of smoothing applied depends on the size or scale of the smoothing operator. In general, for a small scale, the detector extracts fine details of intensity changes from the image, but tends to be more sensitive to noise. A larger scale extracts coarse details of intensity changes, but some of the detected edges tend to have a large localization error. Selecting a single scale of smoothing which is optimal for all edges in an image is very difficult. One filter size may not be good enough to remove noise while keeping good localization.

Multiscale edge detection offers an alternative. This involves applying smoothing operators of different sizes to an image, extracting the edges at each scale, and combining the recovered edge information to form a more complete picture of the actual edge representation of the image. The basic premise of this technique is that different parts of an image have varying degrees of noisiness and types of edges, therefore each part of the image needs to be smoothed differently. However, there are problems associated with this technique, namely, how many filters should be used, how to determine the scales of the filters, and how to combine the responses from each filter so as to create a single edge map.

2.2 Significance of the Gaussian filter

The most widely used smoothing filters are Gaussian filters. Such filters have been shown to play an important role in edge detection in the human visual system, and to be extremely useful as detectors for edge and line detection. Marr and Hildreth [54] demonstrated that the Gaussian filter (along with the Laplacian operator) is very similar to the difference of Gaussians (DOG) filter. This is a well known approximation to the shape of spatial receptive fields in the visual system of cats that has also been proposed for humans. By variational methods, Canny [10] derives an optimal edge detection operator which turns out to be well approximated by the first derivative of a Gaussian function.

Babaud, et al. [2] proved that when one dimensional signals are smoothed with a Gaussian filter, the scale space representation of their second derivatives shows that existing zero-crossings disappear when moving from a fine-to-coarse scale, but new ones are never created. They also proved that for a wide category of signals, the Gaussian function is the only filter that has this property. This unique property makes it possible to track zero-crossings over a range of scales, and also gives the ability to recover the entire signal at sufficiently small scales. Yuille and Poggio [80] extended this work to two-dimensional signals and proved that with the Laplacian, the Gaussian function is the only filter in a wide category that does not create zero-crossings as the scale increases. They also showed that for nonlinear directional

derivatives along the gradient, there is no filter that does not create zero-crossings as the scale increases.

Another extraordinary property of the Gaussian filter is that it is the only operator that satisfies the uncertainty relation:

$$\Delta x \Delta \omega \geq \frac{1}{2} \tag{2.1}$$

where Δx and $\Delta \omega$ are its variance in the spatial and frequency domains, respectively. This property allows the Gaussian operator to give the best trade-off between the conflicting goals of the localization in the spatial and frequency domains simultaneously.

The two dimensional Gaussian filter is also the only rotationally symmetric filter that is separable in Cartesian coordinates. Separability is important for computational efficiency when implementing the smoothing operation by convolutions in the spatial domain.

2.3 Gaussian-based edge detection techniques

Marr and Hildreth [54] originally proposed the spatial coincidence assumption which led to the idea of multiscale edge analysis. Having observed that significant intensity changes occur at different scales within an image, they concluded that optimal

detection of these changes would require the use of a filter that operates at several different scales. They further argued that the edge maps of the different scales each contained important information about physically significant phenomena. If an edge is present at several different scales, then it represents the presence of an image intensity change due to a single physical phenomenon. If an edge is present at only one scale, then it may be that two independent physical phenomena are operating to produce intensity changes in the same region of the image. However, this assumption is mainly based on intuition and some aspects of it still remain open for further research.

Marr and Hildreth also argued that the optimal smoothing filter for images should be localized in both the spatial and frequency domains, thereby satisfying the uncertainty relation given in Equation (2.1). They suggested the Gaussian operator which in two dimensions is given by:

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \tag{2.2}$$

where σ is the standard deviation of the Gaussian function and (x, y) are the Cartesian coordinates of the image. By applying Gaussian filters of different scales (σ) to an image, a set of images with different levels of smoothness can be obtained. To detect the edges in these images it is necessary to find the zero-crossings of their second derivatives. Marr and Hildreth achieved this by using the Laplacian of a Gaussian (LOG) function as a filter:

$$\begin{aligned}\nabla^2 g(x,y) &= \frac{d^2}{dx^2}g(x,y) + \frac{d^2}{dy^2}g(x,y) \\ &= \frac{x^2+y^2-2\sigma^2}{2\pi\sigma^6}e^{-(\frac{x^2+y^2}{2\sigma^2})}\end{aligned} \tag{2.3}$$

This is an orientation independent operator and its scale is given by σ. The Marr-Hildreth operator is the most well-known edge operator and has become a standard gauge for rating the performance of other operators.

It has been found that zero-crossings are only reliable in locating edges if they are well separated, otherwise the problem of detecting false edges arise. The reason is that zero-crossings correspond to local maxima and minima in the first derivative of an image function, whereas only local maxima indicate the presence of a real edges [13, 50]. LOG filtered images also suffer from the problem of missing edges - edges in the original image may not have corresponding edges in a filtered image. In addition, it turns out to be very difficult to combine LOG zero-crossings from different scales, primarily because [60]: (1) a physically significant edge does not match a zero-crossing for more than a few and very limited number of scales (2) zero-crossings in larger scales move very far away from the true edge position due to poor localization of the LOG operator, and (3) there are too many zero-crossings in the small scales of a LOG filtered image, most of which is due to noise. Furthermore, physiological experiments have found no evidence to support zero crossings as a

model for biological vision.

Canny [10] developed an operator that has also become a standard gauge in edge detection. His technique is based on optimizing three criteria desired for any edge detection filter: good detection, good localization, and only one response to a single edge. For good detection, the probability of false detection should decrease with increasing signal-to-noise-ratio (SNR). The SNR at an edge point is given by:

$$SNR = \frac{\left| \int_{-W}^{W} G(-x) f(x) dx \right|}{n_0 \sqrt{\left(\int_{-W}^{W} f^2(x) dx \right)}} \tag{2.4}$$

where $f(x)$ denotes the optimal operator, $G(x)$ denotes the edge and n_0 is the root mean-squared value of the noise. It is assumed that the optimal filter has a finite impulse response bounded by [-W, W].

For good localization the expression to be maximized is:

$$Localization = \frac{\left| \int_{-W}^{W} G'(-x) f'(x) dx \right|}{n_0 \sqrt{\int_{-W}^{W} f'^2(x) dx}} \tag{2.5}$$

where $f'(x)$ is the derivative of $f(x)$.

The criterion for eliminating multiple responses to a single edge is designed to minimize the number of spurious maxima within the operator's spatial extent. The expression for the distance between adjacent maxima of the filter output in response

to noise is:

$$x_{max} = 2\pi \left(\frac{\int_{-\infty}^{\infty} f'^2(x)dx}{\int_{-\infty}^{\infty} f''^2 dx} \right)^{\frac{1}{2}} = kW \tag{2.6}$$

Fixing the constant k to be as large as possible results in the distance between spurious maxima being as great as possible.

In order to simplify the analysis for step edges, Canny computed the function $f(x)$ so that the localization is maximized while keeping the single response criterion constant. By variational methods, he derived a set of filters corresponding to various values of the single response criterion. Canny showed that for a one-dimensional step edge, the derived optimal filter can be approximated by the first derivative of a Gaussian function with variance σ:

$$f(x) \approx -\frac{x}{\sigma^2} e^{-\frac{x^2}{\sigma^2}} \tag{2.7}$$

In two dimensions, his edge detector uses two filters representing derivatives along the horizontal and vertical directions to optimally detect the edges in an image with additive Gaussian white noise. Canny's operator is not unique as it varies with the edge profile (e.g., step edge or ramp edge). However, for step edges, Canny's optimal operator is similar to the LOG operator because the maxima in the output of a first derivative operator corresponds to the zero crossings in the Laplacian operator used by Marr and Hildreth.

Canny also proposed a scheme for combining the outputs from different scales. His strategy is fine-to-coarse and the method is called *feature synthesis*. It starts by marking all the edges detected by the smallest operators. It then takes the edges marked by the small operator in a specific direction and convolves them with a Gaussian normal to the edge direction of this operator so as to synthesize the large operator outputs. It then compares the actual operator outputs to the synthesized outputs. Additional edges are marked if the large operator detects a significantly greater number of edges than what is predicted by the synthesis. This process is then repeated to mark the edges from the second smallest scale that were not marked by the first, and then to mark the edges from the third scale that were not marked by either of the first two, and so on. In this way, it is possible to include edges that occur at different scales even if they do not spatially coincide.

Canny's edge detector uses adaptive thresholding with hysteresis to eliminate streaking of edge contours. Two thresholds are involved, with the lower threshold being used for edge elements belonging to edge segments already having points above the higher threshold. The thresholds are set according to the amount of noise in the image, which is determined by a noise estimation procedure.

The problem with Canny's edge detection is that his algorithm marks a point as an edge if its amplitude is larger than that of its neighbors without checking that the differences between this point and its neighbors are higher than what is expected for

random noise. His technique causes the algorithm to be slightly more sensitive to weak edges, but it also makes it more susceptible to spurious and unstable boundaries wherever there is an insignificant change in intensity (e.g. on smoothly shaded objects and on blurred boundaries).

In [68], Schunk introduces an algorithm for the detection of step edges using Gaussian filters at multiple scales. The initial steps of Schunk's algorithm are based on Canny's method. The algorithm begins by convolving an image with a Gaussian function. The gradient magnitude and gradient angle are then computed for each point in the resulting smoothed data array. Next, the gradient ridges in the results of the convolution are thinned using non-maxima suppression. Then, the thinned gradient magnitudes are thresholded to produce the edge map.

Multi-resolution edge detection is incorporated by repeating the steps above for several scales of the Gaussian filter. The gradient magnitude data at the largest scale will contain large ridges which correspond to the major edges in the image. As the scale decreases, the gradient magnitude data will contain an increasing number of ridges, both large and small. Some of these correspond to major edges, some to weaker edges, and the rest are due to noise and unwanted details. The gradient magnitudes over the chosen range of scales are multiplied to produce a composite magnitude image. Ridges that appear at the smallest scale and correspond to major edges will be reinforced by the ridges at larger scales. Those that do not, will be

attenuated by the absence of ridges at larger scales. Therefore, in the combined magnitude image, the ridges that correspond to major edges are much higher than the ridges that do not. Non-maxima suppression is then performed using sectors obtained from the gradient angle of the largest filter.

Schunk's algorithm chooses the width of the smallest Gaussian filter to be around $w = 7$, and the filters that are used differ in width by a factor of 2. However, he did not discuss how to determine the number of filters to use. In addition, by choosing such a large size for the smallest filter, Schunk's technique loses a lot of important details which may exist at smaller scales.

Bergholm [8] proposed an algorithm which uses the Gaussian filter and combines edge information moving from a coarse-to-fine scale. His method is called *edge focusing*, and uses a rule-based approach for detecting local features and for tracking and predicting a possible scale parameter. Both the Marr-Hildreth and Canny edge detectors are possible schemes that can be used in edge focusing.

The image is first smoothed with a large scale Gaussian filter and then the edge detection process is performed using adaptive thresholding. Assuming that edge contours rarely move by more than two pixel for a unit change in the scale parameter, the exact location of the edges is determined by tracking them over decreasing scales. Therefore, the results from one scale of the edge detector is used to predict the locations of edges in the next, smaller, scale.

The idea behind edge focusing is to reverse the effect of the blurring caused by the Gaussian operator. Blurring is not a desired feature as it results in poor edge localization. It is simply used as a means of removing the noise and other unnecessary features. The most obvious way of undoing the blurring process is to start with edges detected at the coarse scale and gradually track or *focus* these edges back to their original locations in the fine scale.

There are several problems associated with edge focusing, the foremost being how to determine the starting and ending scales of the Gaussian filter. Bergholm suggests the range ($3 \leq \sigma < 6$) for the maximum scale, but did not specify a minimum scale. He also did not discuss in detail how to choose the threshold which is used at the coarsest level, and this is a parameter which is critical in determining how well the algorithm performs. If it is too high, a number of true edge points will be eliminated right from the start. If it is too low, the output of the edge focusing could be very noisy. In addition, since edge focusing is obtained at a finer resolution, some edges (i.e. the blurred ones, such as shadows) present a juggling effect at small scales. This is due to the splitting of a coarse edge into several finer edges, and tends to give rise to broken, discontinuous edges.

Lacroix [48] avoids the problem of splitting edges by tracking edges from a fine-to-coarse resolution. His algorithm detects edges using the Canny method of non-maxima suppression of the magnitude of the gradient in the gradient direction. His

method then considers three scales, σ_0, σ_1 and σ_2. The smallest scale, σ_0 is the detection scale, and is the finest resolution at which an edgel (i.e. a short, linear edge element) first appears. The largest scale, σ_2 is the blurring scale, and is the coarsest resolution at which the edgel still remains. The intermediate resolution is computed as:

$$\sigma_1 = \sigma_0 + \frac{(\sigma_2 - \sigma_0)}{3} \tag{2.8}$$

An edgel is validated if (1) it is the local maximum of a Gaussian gradient, and (2) the two regions it separates are homogeneous and significantly different from one another. Only validated edgels are then tracked through the scales.

Although Lacroix avoids the problem of splitting edges, he introduces the problem of localization error as it is the coarsest resolution that is used to determine the location of the edges. He also provides no explanation as to how to decide which scales are to be used and under what conditions.

Williams and Shah [76] devised a scheme to find edge contours using multiple scales. They analyzed the movement of edge points smoothed with a Gaussian operator of different sizes, and used this information to determine how to link edge points detected at different scales. Their method, following the lead of Canny, uses a gradient of Gaussian operator to determine gradient magnitude and direction, followed by non-maxima suppression to identify ridges in the gradient map. Since the re-

sulting ridges are often more than one pixel wide. the gradient maxima points are thinned and then linked using an algorithm which assigns weights based on four measures: noisiness. curvature. contour length and gradient magnitude. The set of points having the highest average weight is chosen.

The algorithm extends to multi-resolution by convolving the image with the Gaussian filter at three scales: σ. $\sqrt{2}\sigma$ and 2σ. First. the best partial edge contours are found using the largest scale. Then the next smaller scale is used. and the regions around the end points of the contours are examined to determine if there are possible edge points at the smaller scale having similar directions to the end points of the contours. The original algorithm is then carried out for each of these possible edge points. and the best are chosen as an extension to the original edge contour. The scale is decreased to the smallest scale. and the process is repeated.

Although Williams and Shah specify the number of scales to be used and the relationship between these scales. they did not suggest the best way to choose the value of σ and under what conditions.

To avoid the common problems associated with integrating edges detected at multiple scales. Jeong and Kim [39] proposed a scheme that automatically determines the optimal scales for each pixel before detecting the final edge map. To find the optimal scales for a Gaussian filter. they define an energy function that quantitatively determines the usefulness of the possible edge map. They approach the edge

detection problem as finding the size of the Gaussian filter, $\sigma(x, y)$, which minimizes the energy function over σ, $E(\sigma)$. The parameter σ is chosen so that:

1. It is large at uniform intensity areas, thereby smoothing out random noise
2. It is small at locations where the intensity changes significantly, thus retrieving edges accurately, and
3. It does not change sharply from pixel to pixel, therefore avoiding broken edges due to random noise.

If g and I denote the two dimensional Gaussian filter and image function, respectively, the equation to be minimized is:

$$E(\sigma) = \int\int ((I * g)^2 + \lambda|\nabla\sigma^{-1}|^2)dxdy \tag{2.9}$$

Since the Jeong-Kim algorithm is designed to adaptively find the optimal scale of the Gaussian filter for every location in the image function, it can be easily incorporated into any Gaussian-based edge detection technique. However, this algorithm does result in reduced performance when it comes to detecting straight lines in vertical or horizontal directions. The algorithm also has the disadvantage of low speed performance [7].

Deng and Cahill [16] also use an adaptive Gaussian filtering algorithm for edge detection. Their method is based on adapting the variance of the Gaussian filter

to the noise characteristics and the local variance of the image data. Based on observations of how the human eye perceives edges in different images, they concluded that in areas with sharp edges, the filter variance should be small to preserve the sharp edges and keep the distortion small. In smooth areas, the variance should be large so as to filter out noise. They proposed that the variance of a one dimensional Gaussian filter at location x is:

$$\sigma^2(x) = \frac{k{\sigma_n}^2}{\sigma_f^2(x) + \sigma_n^2} \tag{2.10}$$

where k is a scaling factor, σ_n^2 is the noise variance, and $\sigma_f^2(x)$ is the local variance of the signal.

The major drawback of this algorithm is that it assumes the noise is Gaussian with known variance. In practical situations, however, the noise variance has to be estimated. The algorithm is also very computationally intensive.

In [7], Bennamoun *et al.* present a hybrid detector that divides the tasks of edge localization and noise suppression between two sub-detectors. This detector is the combination of the outputs from the Gradient of Gaussian and Laplacian of Gaussian detectors. The hybrid detector performs better than both the first order and second order detectors alone, in terms of localization and noise removal [7]. The authors extended the work to automatically determine the optimal scale σ and threshold Th, of the hybrid detector. They do this by:

1. Finding the probability of detecting an edge for a signal with noise. $P(A)$
2. Finding the probability of detecting an edge in noise only. $P(B)$, and
3. Deriving a cost function which maximizes $P(A)$ and minimizes $P(B)$, thereby giving the optimal σ and Th values.

The minimization of $P(B)$ is equivalent to the maximization of its complement, therefore the cost function to be maximized is

$$CF = \eta P(A) - (1 - \eta)[1 - P(B)] \tag{2.11}$$

where $0 \leq \eta \leq 1$. When $\eta = 0$, all the emphasis is placed on noise suppression. When $\eta = 1$, the emphasis is on edge detection. Choosing η to be around 0.5 results in σ and Th values which try to balance noise suppression and edge detection. However, as the authors results show, their technique is still susceptible to false edge detection, especially in the presence of high noise levels.

In [60] and [61], Qian and Huang introduce a 2-D edge detection scheme. The scheme detects edges in images with an edge detection functional that uses the LOG zero-crossing contours as a guide to the true edge locations. The proposed edge functional is based on a parametric 2-D edge model, and was developed to be optimal in terms of signal-to-noise ratio (SNR) and edge localization accuracy (ELA).

The procedure begins by convolving an image with the LOG operator and finding

the zero-crossing points. Zero-crossing contours are then segmented at points with large curvatures, and the true edge locations are determined using the 2-D edge detection functional. Adaptive thresholding based on the global noise estimation and physiological evidence from biological visual systems is then performed on the edge segments. This is followed by an edge regularization technique for continuity and smoothness of the detected edges.

Edge segments are combined from different scales using a fine-to-coarse strategy. Since the smallest scale of the detection functional gives the best ELA, and only edges with high SNR survive the edge detection procedure described above, the final edge map always contains all the edges from the smallest scale. The results from larger scales are incorporated into the final edge map if they produce new edge segments.

The scales of the operator which Qian and Huang use, were selected to span the range of filters that operate in the human fovea. They used seven scales between $\sigma_{min} = 2.5$ and $\sigma_{max} = 6.7$. However, this may not be the ideal range for computational methods. In addition, the range may also change depending on the type of image and the amount of noise it contains.

2.4 Summary

The Gaussian filter has several desirable features which accounts for its wide use in many image processing applications. However, research clearly demonstrates that edge detection techniques involving this filter do not give satisfactory results. All of the methods presented in this chapter suffer from problems associated with Gaussian filtering, namely, edge displacement, vanishing edges and false edges. The introduction of multiscale analysis further complicates the issue by creating two major problems : (1) how to choose the size of the filters, and (2) how to combine edge information from different scales. Adaptive approaches which avoid the multi-resolution problem, all tend to be computationally intensive. As it currently stands, use of the Gaussian filter requires making compromises when developing algorithms to give the best overall edge detection performance.

Chapter 3

The Line Weight Function (LWF)

In this chapter, a special line weight function (LWF) for enhancing edges in digital images is discussed. This operator is a combination of a Gaussian function and its second derivative, or equivalently, zero- and second-order Hermite functions. The development of this operator is based on biological and mathematical evidence that the visual receptive fields of primates are of this form. The LWF has demonstrated the ability to provide good edge enhancement and edge localization, with significant removal of noise from the edge description.

3.1 Motivations for using the LWF

One of the first works on how light intensities affect what we see can be found in [51]. Consider a visual scene with light intensity distribution $I = f(x, y)$. For a point in that scene, Mach stressed that the output of the retina at a corresponding point is given not just by I, but also by the Laplacian of I.

Another early work on how the human visual system responds to light intensities can be found in [20]. Fiorentini and Mazzantini studied the visual response of human subjects to spatial interactions between dark and light line targets. Their results indicate that the LWF of humans is shaped like the linear combination of zero- and second-order Hermite functions.

As discussed in Chapter 2, Marr and Hildreth modeled the Laplacian of a Gaussian (LOG) operator after the difference of Gaussian (DOG) filter which is a widely known description for the retinal ganglion cells of cats. However, when Young [78] performed quantitative tests of the LOG filter with actual biological data (monkey, cat and human) he made some discoveries: (1) a new mechanism called the difference of offset Gaussians (DOOG) provides a better description of the receptive field structures of the test data, (2) the DOOG is closely fitted by the weighted sum of a Gaussian and its Laplacian, indicating that a Gaussian term needs to be added to the LOG in order to accurately describe the data, and (3) the Gaussian derivative model (i.e. the sum of a Gaussian and its Laplacian) provided the best approximation to all the

test data when compared to the Gabor and other contending models. Young also pointed out that the Gaussian derivative model provides edge and line enhancement, while retaining all the information in the original image.

Martens [55, 56] presented a new system for the local processing of images called the Hermite transform. In order to make the processing of an image local, its information is usually multiplied by a window function. When the window function is Gaussian, the associated polynomial transform is called a Hermite transform. The Hermite transform involves filter functions that are derivatives of Gaussians which, as previously mentioned, have been shown to play a role in modeling the human visual system.

Motivated by psychophysical studies, Malik and Perona [53] suggested that the first stage of visual processing can be regarded as a convolution between an image and a bank of linear filters f_i, where f_i are of the following type: (1) Laplacian of Gaussian (2) $f(x,y) = G'_{\sigma 1}(x)G_{\sigma 2}(y)$, and (3) $f(x,y) = G''_{\sigma 1}(x)G_{\sigma 2}(y)$, where $G_{\sigma i}$ denotes a Gaussian function with standard deviation σ_i, and $G'_{\sigma i}$ and $G''_{\sigma i}$ denote the first and second derivatives of $G_{\sigma i}$, respectively. The kernels of these filters are first and second derivatives of Gaussians and have been shown by Canny [10] to be fairly close to optimal (for suitably defined performance criteria) for detecting and localizing edges and lines.

Stewart and Pinkham [71] tackled a classical psychophysics experiment using a math-

ematical approach. Treating the contrast sensitivity experiments of Graham and Robson [28] as an eigenvalue problem. they derived a complete orthonormal set of eigenfunctions. The resulting eigenfunctions are neither sine and cosine nor Gabor functions. Rather. Hermite functions arise as the eigenfunctions of a space-variant differential operator used to model the contrast sensitivity of humans.

3.2 The one-dimensional LWF

In a typical contrast sensitivity experiment in psychophysics. an investigator presents one stimulus pattern at a time. The observer adjusts the amplitude of the pattern until he/she can just detect its contrast. The experimenter then presents another stimulus of a different spatial frequency. Adjusting the amplitude of a sinusoidal pattern is mathematically equivalent to multiplying the sinusoid by a scalar. By definition. the transformation of an eigenfunction is also equivalent to multiplication by a scalar quantity. In this sense. most contrast sensitivity experiments can be thought of as an approximation of an eigenvalue problem. A brief review of the mathematical formulation of the eigenvalue problem [71] is presented in this and the following section.

Define an operator

$$p = -\frac{d^2}{dx^2} + x^2 \tag{3.1}$$

and a test function

$$u = e^{-\frac{x^2}{2}} \tag{3.2}$$

If p is applied to (3.2) the resulting function will be

$$pu = \lambda u \tag{3.3}$$

In other words, u is an eigenfunction of the operator p with corresponding eigenvalue λ. The resulting differential equation is

$$-u'' + x^2 u = \lambda u \tag{3.4}$$

The solutions for (3.4) are of the form

$$u(x) = ch_n(x) = ce^{-\frac{x^2}{2}} H_n(x) \tag{3.5}$$

where H_n is a Hermite polynomial of degree n, and c is a constant. The function h_n is the Hermite function. In order to incorporate multiscale analysis, a scale parameter σ (standard deviation of the Gaussian function) is introduced. The Hermite function h_n with scale parameter σ is:

$$h_n(x/\sigma) = \frac{1}{\sqrt{2^n n!}} \frac{d^n}{d(x/\sigma)^n} \frac{1}{\sigma\sqrt{\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{3.6}$$

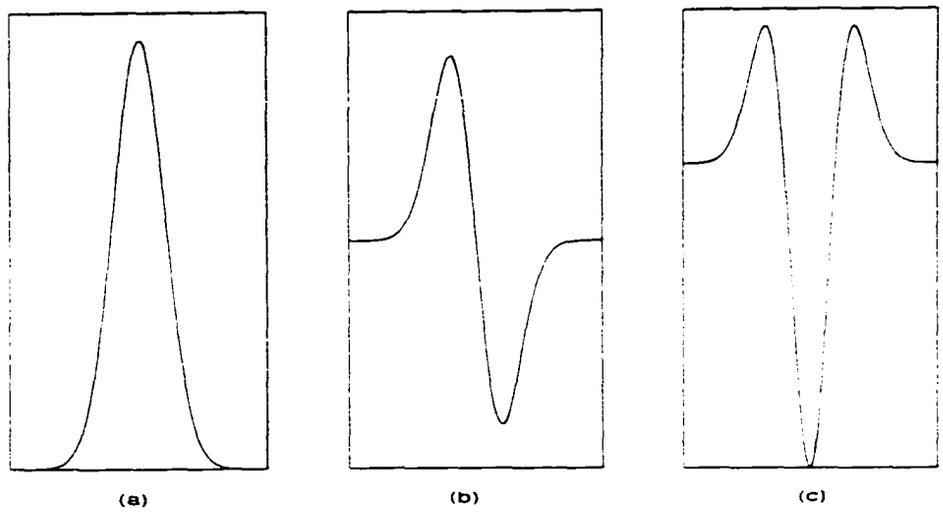


Figure 3.1: The first three Hermite functions: (a)h_0 (b)h_1 and (c)h_2.

Therefore, $h_0(x/\sigma)$ is the Gaussian function

$$h_0(x/\sigma) = \frac{1}{\sigma\sqrt{\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{3.7}$$

and $h_2(x/\sigma)$ is the second derivative of the Gaussian function

$$h_2(x/\sigma) = \frac{1}{\sqrt{8\pi\sigma^2}} \left(-e^{-\frac{x^2}{2\sigma^2}} + \frac{x^2}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}} \right) \tag{3.8}$$

The LWF is the linear combination of $h_0(x/\sigma)$ and $h_2(x/\sigma)$

$$l(x/\sigma) = c_0 h_0(x/\sigma) + c_2 h_2(x/\sigma) \tag{3.9}$$

The LWF is composed only of even-order Hermite functions so that it will be a

symmetric and therefore orientation independent operator. Fig. 3.1 shows the first three Hermite functions.

1 Experimental results in 1-D

The problem of detecting and localizing intensity transitions in grayscale images is usually treated as one of finding step edges. However, it has been suggested in [53] that step edges are not adequate models for the discontinuities that result from the projection of depth or orientation discontinuities in physical scenes. Mutual illumination and specularities have effects on convex or concave object edges. In addition, there is a shading gradient on the image regions bordering the edge. As a result of these effects, real image edges are not step functions but more typically are a combination of step, peak and roof profiles. This has been shown experimentally by Herkovits and Binford in 1970. Quantitative analyses of the associated physical phenomena have been provided by Horn [31] and more recently by Forsyth and Zisserman [22].

The input signal (Fig. 3.2a) is a composite edge with added random noise. The transition representing the edge is located at $x = 500$. The LWF using coefficients $c_0 = 0.065$ and $c_2 = -0.042$, and scale parameter $\sigma = 2$ is shown in Fig. 3.2b. Convolving Figs. 3.2a and 3.2b produces the output signal in Fig. 3.2c. This experimental result shows that the LWF is highly effective as an edge detector. The noise

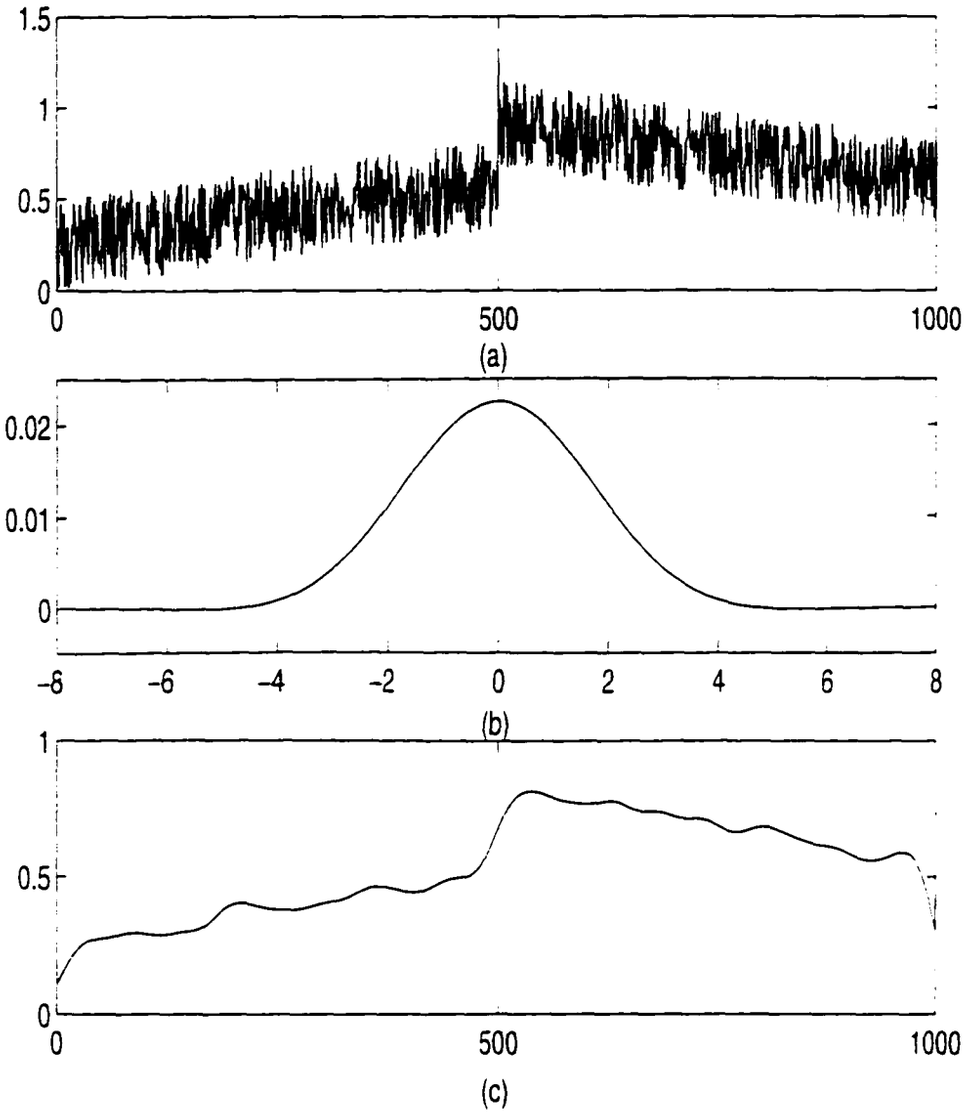


Figure 3.2: (a) Composite edge with random noise. (b) The LWF operator. $\sigma = 2$. (c) Result of convolving (a) and (b).

is eliminated to a considerable extent and the magnitude of the edge is enhanced in comparison to the transitions around it. In addition, the location of the edge has not been shifted.

3.3 The two-dimensional LWF

The formulation of the two-dimensional LWF is an extension of that of its one-dimensional counterpart. The two-dimensional operator is given by

$$P = -\nabla + V \tag{3.10}$$

where in Cartesian coordinates (x, y), ∇ is the Laplacian operator

$$\nabla = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \tag{3.11}$$

and V is the function

$$V = x^2 + y^2 \tag{3.12}$$

For a test function U, the eigenvalue problem then becomes

$$PU = -\nabla U + VU = \lambda U \tag{3.13}$$

The function U is of the form

$$U(x,y) = X(x)Y(y) \tag{3.14}$$

Substituting (3.14) into (3.13) and dividing through by XY gives

$$\left(x^2 - \frac{1}{X}\frac{\partial^2 X}{\partial x^2}\right) + \left(y^2 - \frac{1}{Y}\frac{\partial^2 Y}{\partial y^2}\right) = \lambda \tag{3.15}$$

Since λ is a constant, the left-hand side of the equation can only hold if

$$-X'' + x^2 X = \lambda_x X \tag{3.16}$$

and

$$-Y'' + y^2 Y = \lambda_y Y \tag{3.17}$$

where λ_x and λ_y are constants. Since (3.16) and (3.17) are the same form as equation (3.4), it follows that $U(x,y)$ can be written as

$$U(x,y) = h_m(x)h_n(y) \tag{3.18}$$

where m and n are the degrees with respect to x and y, respectively. The two-dimensional LWF with scale parameter σ is given by

$$L(x/\sigma, y/\sigma) = c_0 h_0(x/\sigma)h_0(y/\sigma) +$$

$$c_2[h_0(x/\sigma)h_2(y/\sigma) + (h_2(x/\sigma)h_0(y/\sigma)] \tag{3.19}$$

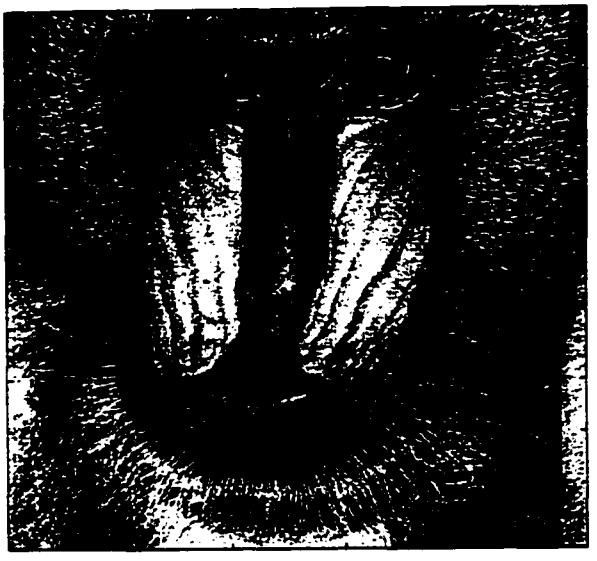
As in the one dimensional case, only even order Hermite functions are included so as to preserve the circular symmetry of this operator.

1 Experimental results in 2-D

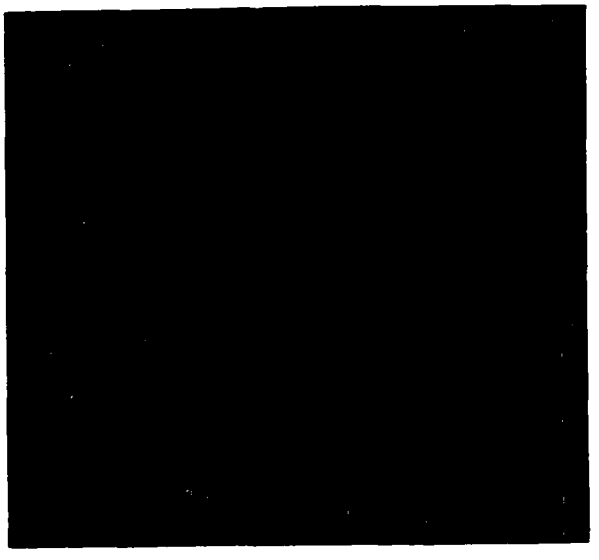
The original image, shown in Fig. 3.3a, is quite complex. The most prominent areas of this image are the two eyes, the nose, the cheeks and the mouth[1]. The image is first filtered with the LWF to enhance the edges and then convolved with the Sobel operator to detect the edges. The Sobel operator finds edges where the first derivative of the enhanced image is a maximum. The results of this process is shown in Fig. 3.3b. For comparison purposes, the results using the Sobel and LOG operators on the original image are shown in Figs. 3.4a and 3.4b, respectively.

The results in Fig. 3.3b show that the outlines of the prominent areas of the image have been identified by the LWF. When compared to Fig. 3.4a which shows the result of using the Sobel operator only, it can be seen that although the Sobel operator identifies the prominent areas, it also picks up a lot of extraneous and unwanted details. Therefore, enhancing the edges of the image with the LWF improved the performance of the Sobel operator. Comparing Figs. 3.3b and 3.4b, it can be seen

[1]These regions stand out when a human being looks at the picture.

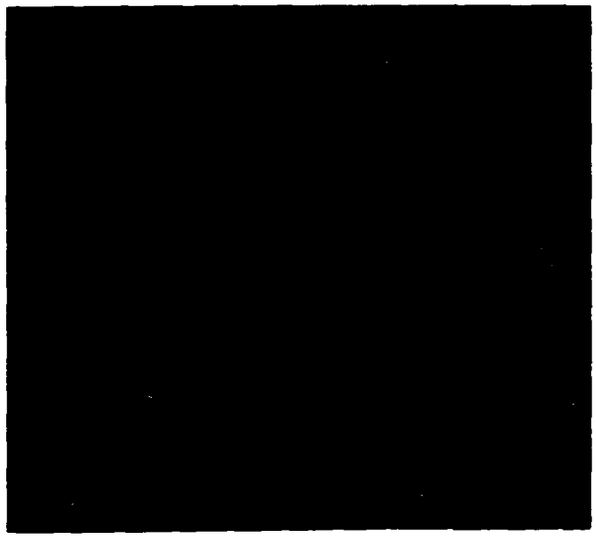


(a)

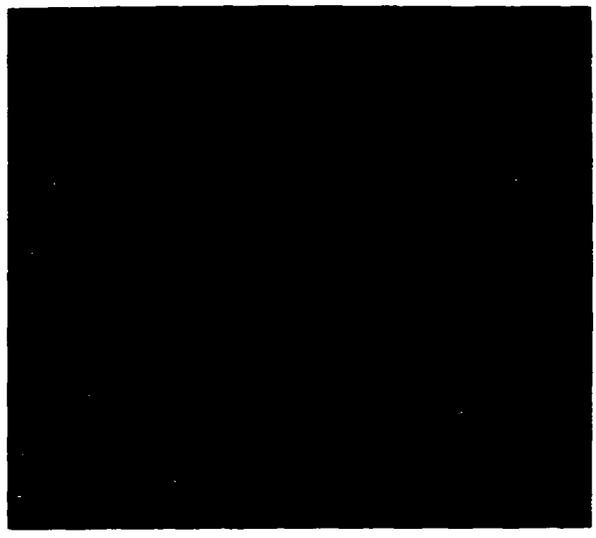


(b)

Figure 3.3: (a) The original image: Mandrill. (b) Result of convolving the LWF ($\sigma = 2$) and Sobel operators with (a).



(a)



(b)

Figure 3.4: (a) Sobel edge detector applied to the Mandrill image. (b) LOG edge detector ($\sigma = 3$) applied to the Mandrill image.

that the performance of the LWF is far superior. The LOG filtered image in Fig. 3.4b is extremely noisy, and the prominent features of the image can hardly be identified.

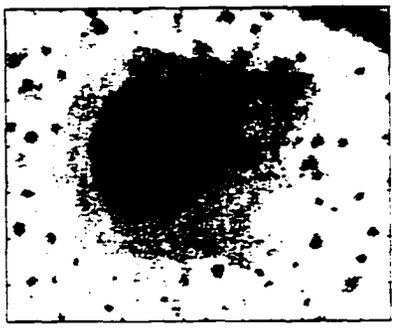
3.4 Applications

1 Applications in image processing

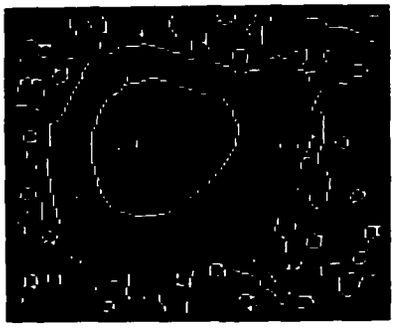
Pathology is the field which deals with the interpretation and diagnosis of changes caused by diseases in cells and tissues. Pathologists review thousands of tissue and cell samples, the primary focus of which is the the detection or confirmation of malignancy. Certain features assist them in their diagnosis, including the size and shape of the cells. Unfortunately, cellular details may be obscured because of clumped cells, mucus and inflammation thereby impeding the diagnosis process.

Segmentation of cell images is an important technique in the applications of image analysis to pathology. Correct cell segmentation can provide useful information about the contour and shape of cells, and assist doctors in the analysis and diagnosis of several diseases. Edge detection is often the first step in image segmentation, therefore an edge detection operator which provides good results can enhance the performance of an image segmentation algorithm.

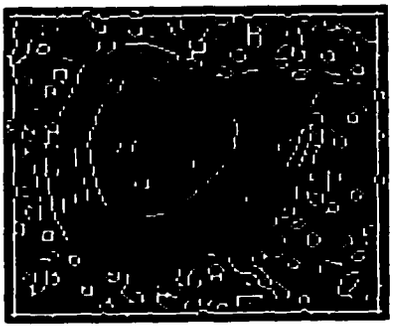
Fig. 3.5a shows a real cell image surrounded by a complex background. Figs. 3.5b



(a)



(b)



(c)

Figure 3.5: (a) The original cell image. (b) LWF operator ($\sigma = 2$) applied to the cell image. (c) LOG operator ($\sigma = 2$) applied to the cell image.

and 3.5c show the edge detection results of the LWF and LOG operators, respectively. The LWF operator is clearly able to detect critical details about the contour of the cell image. The results of the LOG operator, however, are severely obscured by unimportant features in both the cell and its surrounding background. In addition, the LOG outline of the cell's nucleus is not as clearly defined as with the LWF.

2 Applications in speech processing

Speech can be described as a combination of three different sounds: 1) silence, in which no speech sound is produced; 2) unvoiced, in which the vocal cord is not vibrating, so the resulting speech waveform is aperiodic or random in nature, and; 3) voiced, in which the vocal cord is tensed and therefore vibrates periodically when air flows from the lungs. The resulting voiced speech waveform can be considered periodic or quasi-periodic. The cycle of vocal cord vibration consists of two events, the glottal closing and glottal opening. The time interval between successive glottal closures is often defined as an event or a pitch period. Pitch period is a very important parameter in the analysis and synthesis of speech signals.

In event based pitch period detection, the maxima created due to glottal closure is used to determine pitch period. The LWF can detect the transients associated with these maxima and make them more intense. The distance between these successive maxima is used to estimate the pitch period. In addition, when the LWF is convolved

with a speech signal it suppresses the slower varying and noisy parts of the signal. As a result, when a threshold is applied to the results of the convolution, there is less likelihood of detecting maxima due to noise.

Speech was synthesized using the vocal tract and glottal model. The glottal model of the following form is used to generate the glottal excitation signal:

$$g_l(t) = \begin{cases} \frac{1}{2}[1 - \cos(\frac{\pi t}{t_1})] & 0 \leq t \leq t_1 \\ \cos(\frac{\pi(t-t_1)}{2t_2}) & t_1 \leq t \leq t_1 + t_2 \\ 0 & \text{elsewhere} \end{cases}$$

The variables t_1 and t_2 determine the width of the glottal excitation. In order to model the vocal tract we used the equation:

$$V(\omega) = \prod_{k=1}^{m} \frac{1 - 2e^{-\alpha_k T}\cos(\omega_k T) + e^{2\alpha_k T}}{1 - 2e^{-2\alpha_k T}\cos(\omega_k T)e^{-j\omega} + e^{-2\omega_k T}e^{-2j\omega}} \times \prod_{l=1}^{n} \frac{1 - 2e^{-\beta_l T}\cos(\xi_l T)e^{-j\omega} + e^{2\beta_l T}e^{-j2\omega}}{1 - 2e^{-2\beta_l T}\cos(\xi_l T) + e^{-2\beta_l T}} \quad (3.20)$$

where T=0.1ms is the sampling rate, and α_k and β_l are the bandwidths corresponding to the center frequencies of the poles and zeros, ω_k and ξ_l, respectively, of the vocal tract. The glottal function and vocal tract function are convolved which each other in order to produce the synthesized sound. The effects of applying the LWF to a synthesized vowel /e/ with added noise is shown in Fig. 3.6.

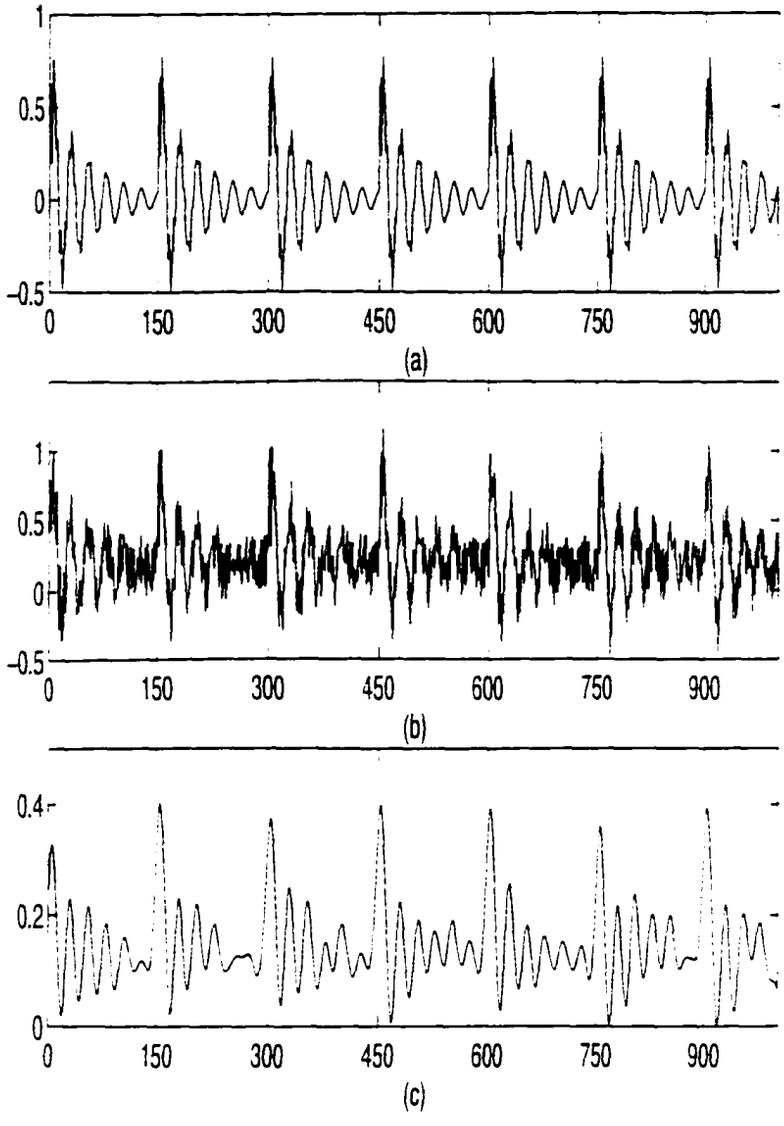


Figure 3.6: (a) A synthesized speech signal /e/ with pitch period of 15ms (the arrows indicate the onset of the true pitch period). (b) Noise is added to (a). SNR=-3.2dB (c) Result of convolving (b) with the LWF. The horizontal axis shows the number of samples with sampling rate 0.1ms.

3.5 Extending the LWF

Many researchers have examined the role of using higher-order Gaussian derivative filters in image analysis and have found this to be important. Pentland [58] showed that the tilt and slant of objects can be determined from their shading by using a third-order Gaussian derivative model. A machine vision system based on this model has been shown to exhibit excellent *subpixel accuracy*, which is the machine equivalent to the precise visual localization capability of humans known as hyperacuity.

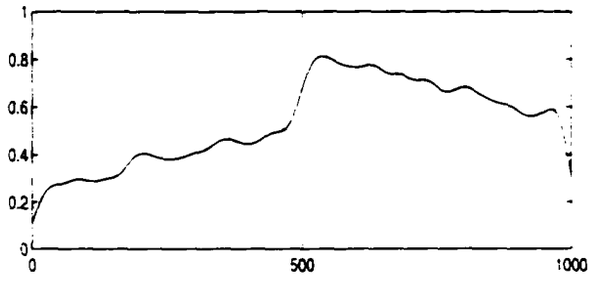
Zucker and Hummel [82] demonstrated both theoretically and experimentally the importance of using higher-order Gaussian derivative filters to deblur images. This has been supported by Martens [57] using the Hermite transform. He showed that deblurring a digital image is equivalent to estimating the coefficients of a polynomial transform. Progressive deblurring is then accomplished by adding successive terms to the polynomial transform.

Our earlier work investigated the roles of the coefficients of the 0-th order and 2-nd order Hermite terms. The results have been reported in [6]. Based on the works mentioned above, the effects of including higher order Hermite terms in the LWF was explored. We found that there was an observed decrease in the performance of the LWF when only one higher-order term was added. Fig. 3.7 demonstrates this. Using the same one dimensional signal as in Fig. 3.2, it is evident that adding a small valued fourth-order Hermite function to the LWF has very little effect on the

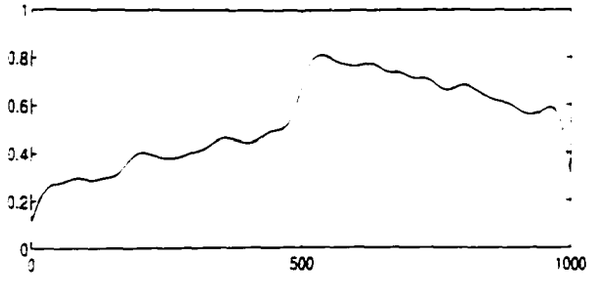
signal when compared to using the "original" LWF. However, as the weight of this fourth-order term is increased, it becomes obvious that the noise removal ability of the LWF decreases. Similar results were obtained in the two dimensional case (Fig. 3.8). Adding higher-order terms larger than four had no significant effect on the performance of the LWF.

3.6 Summary

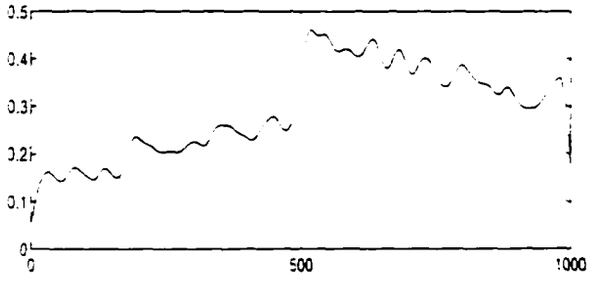
We have proposed an edge enhancement operator, the LWF, which is a weighted combination of a Gaussian and its second derivative. We discussed the motivations behind the development of this operator and described its mathematical formulation. The experimental results show that the LWF is quite effective at enhancing and localizing edges, and removing noise from signals and images.



(a)



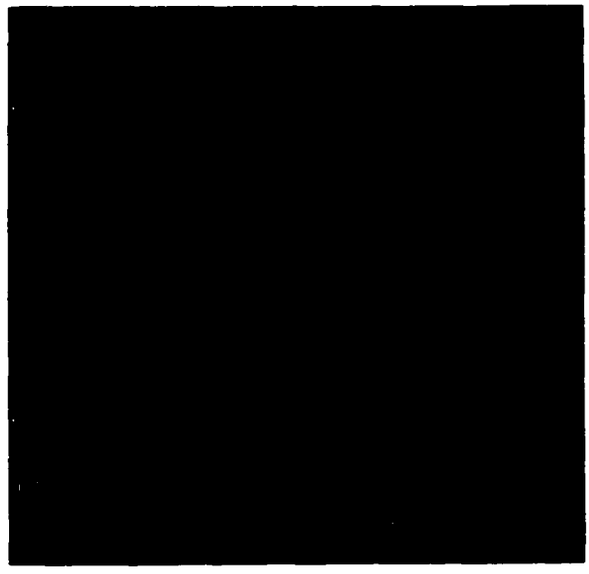
(b)



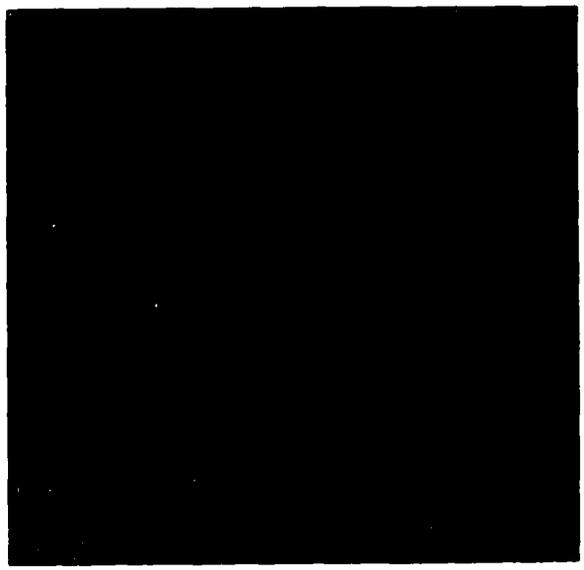
(c)

Figure 3.7: Result of convolving the LWF with composite edge using coefficients: (a) $c_0 = 0.065$, $c_2 = -0.042$ (b) $c_0 = 0.065$, $c_2 = -0.042$, $c_4 = 0.030$ (c) $c_0 = 0.065$, $c_2 = -0.042$, $c_4 = 0.100$

(a)



(b)



(c)

Figure 3.8: (a) The original image: Lenna (b) Edge detection results with LWF coefficients $c_0 = 0.065$, $c_2 = -0.042$ (c) Edge detection results with LWF coefficients $c_0 = 0.065$, $c_2 = -0.042$, $c_4 = 0.080$

Chapter 4

Evaluating the Performance of the LWF

In this chapter, we provide analytical results which show that the LWF produces only the authentic scale space contours. We also show that edges identified by the LWF have excellent localization. We compare our results with those of the Marr-Hildreth zero crossing edge detector [54] for several reasons:

1. Both operators are based on the Gaussian filter.

2. The LWF is an extension of the LOG operator – it is the weighted combination of a second derivative of Gaussian, with an added Gaussian term.

3. The development of both operators was influenced by results from psychophysical research. Marr and Hildreth were inspired by the well-known "difference-of-Gaussian" mechanism used to describe the ganglion cell receptive fields of cats [54]. We were motivated by works such as [51] and [53] which indicate that both the image intensity and its second derivative play a role in visual perception.

4. Both operators require the same number of computational operations to find edges - smoothing, differentiation and detection of zero-crossings or maxima.

The parameter ω in [54] represents the width of the central part of the receptive field of the underlying $\nabla^2 g$ operator.The adjacent positive and negative peaks of the LOG filtered image are approximately $\omega/\sqrt{2}$ apart. An edge is detected by locating such pair of peaks.

The LWF is a combination of a Gaussian and its second derivative. Edges are detected by locating enhanced peaks in a LWF filtered image. It is known that over 99% of the variance in the data is explained by the Gaussian function, while the offset-Gaussian fit to the surround explains about 97% [79]. This provides an approximate relationship between the width of the LWF and the scale parameter σ. In our experiments we used $\omega \approx 6\sqrt{\sigma}$ for effective edge detection. The mask size is carefully chosen to achieve the best operator performance while keeping the complexity under control.

In this chapter we consider only one dimensional operators.

4.1 Edge detection in scale space

As mentioned in Chapter 2, the scale space representation of a signal smoothed with a Gaussian filter has certain desirable properties, one of which is the shape of the contours in the scale space map which allows for the tracking of an edge over several scales. Fig. 4.1, which demonstrates this property, depicts the zero-crossings scale space map of an arbitrary one dimensional signal convolved with the Gaussian operator. The scale space contours are smooth, start at zero,and either proceed forever toward infinite scale or gracefully turn back to zero scale. This well behavedness can be rather misleading, however, as not all the detected zero-crossings correspond to authentic edges in the signal. An explanation of this behavior follows.

Fig. 4.3a shows an ideal step edge. When this edge is smoothed with a Gaussian filter, Fig. 4.3b results. The position of the edge can be localized by finding the point where the first derivative of the smoothed signal has a maximum value (Fig. 4.3c). It can also be located by looking for the zeros of the second derivative of the smoothed signal, which is the basis of the zero-crossing (LOG) edge detection method.

Fig. 4.4a shows a staircase step edge which consists of two step edges, and Fig. 4.4b shows the result of smoothing it with a Gaussian operator. Fig. 4.4c corresponds

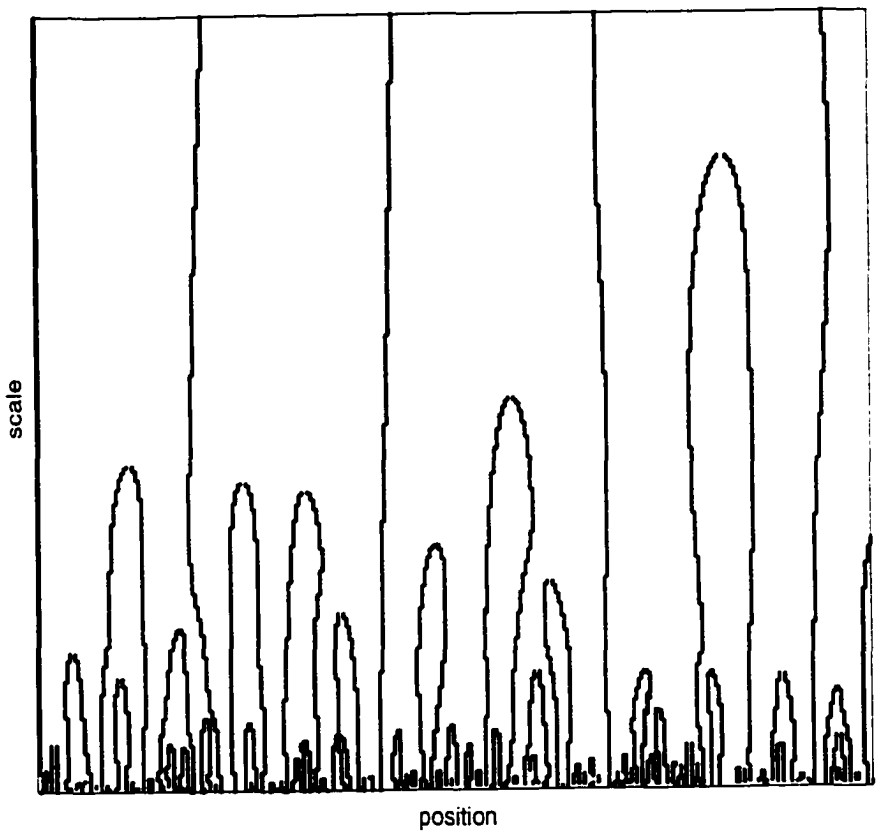


Figure 4.1: A one dimensional scale space map for the LOG filter.

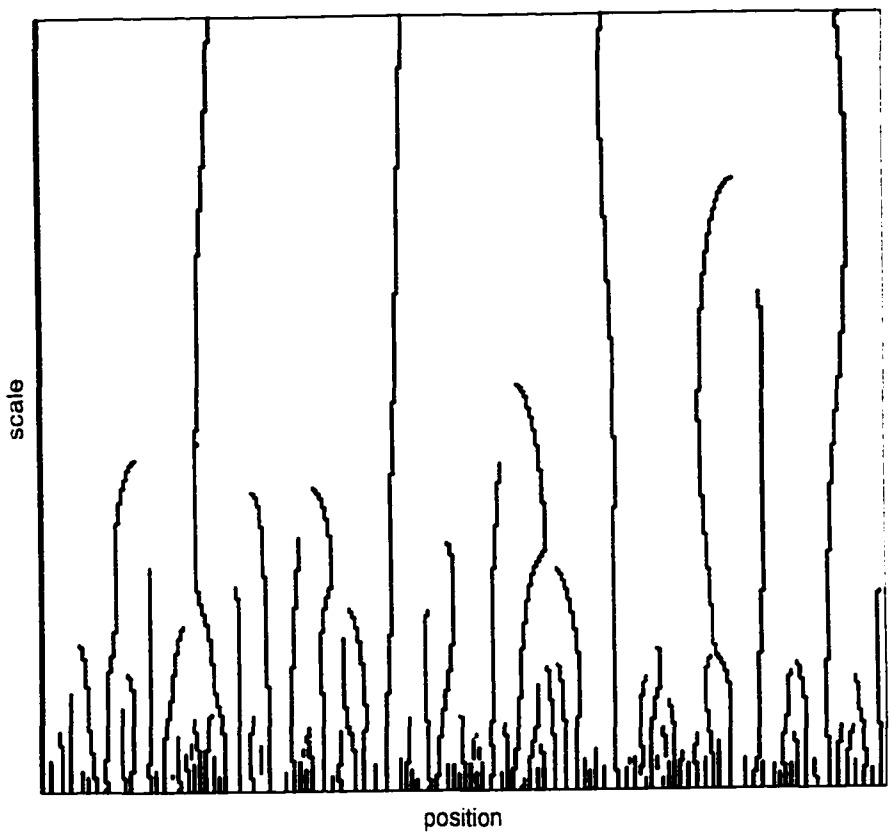


Figure 4.2: A one dimensional scale space map for the LWF filter.

to taking the first derivative of Fig. 4.4b, and it can be seen that three extrema have been identified in this process. This means that when the second derivative of Fig. 4.4b is taken, three zero-crossings will be seen. Two of these zero-crossings correspond to the authentic step edges, but the third zero-crossing which occurs between them is a *phantom* or false edge. This false edge corresponds to the positive minimum in the gradient signal.

The LWF does not detect such false edges. As shown in Fig. 4.6c, the first derivative of a staircase signal smoothed with the LWF identifies the position of the two step edges, and does not give rise to a positive minimum between them. Fig. 4.2 shows the scale space map for the LWF using the same signal as in Fig. 4.1. There are fewer edges detected in Fig. 4.2 because the LWF detects only authentic edges. An in-depth analysis of the differences in the two scale maps is presented in the next section.

4.2 Analysis of the staircase function

In a gray level image, an edge is defined as a sharp change in intensity. Such a discontinuity can be modeled by a unit step function. In Section 4.1, experiments with the LWF and Marr-Hildreth operators on a step edge produced similar results. However, when we considered discontinuities some distance apart, as in a staircase

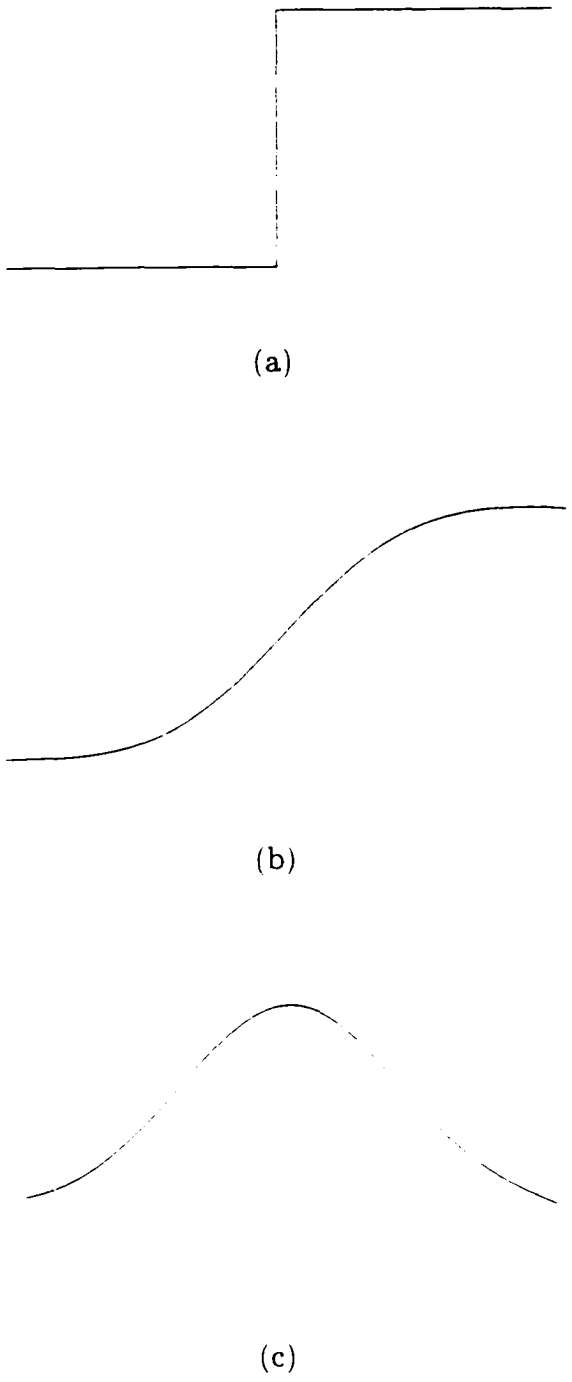
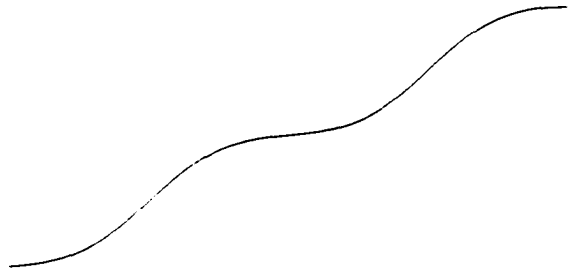
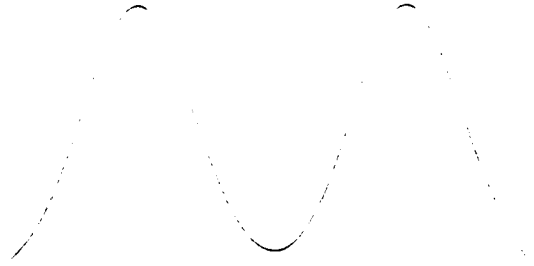


Figure 4.3: (a) A step edge. (b) The step edge smoothed with a Gaussian filter. $\sigma = 2$. (c) The first derivative of (b).

(a)



(b)



(c)

Figure 4.4: (a) A staircase edge. (b) The staircase edge smoothed with a Gaussian filter. $\sigma = 2$. (c) The first derivative of (b).

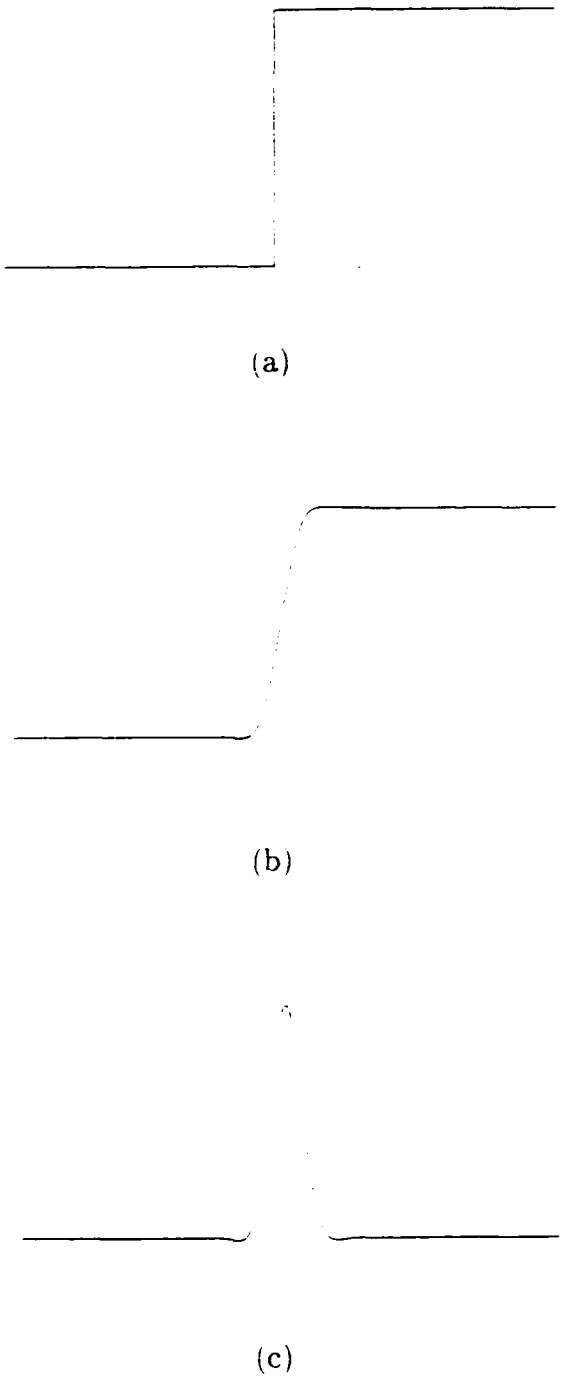
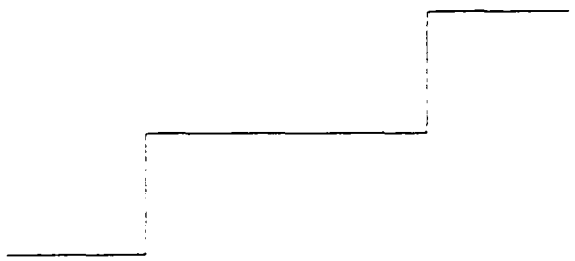
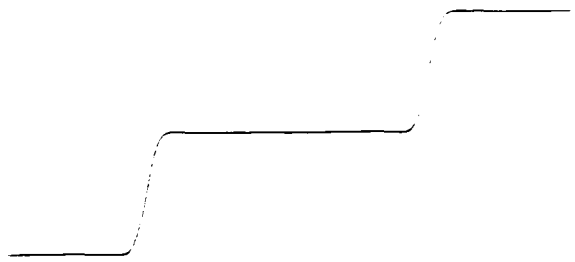


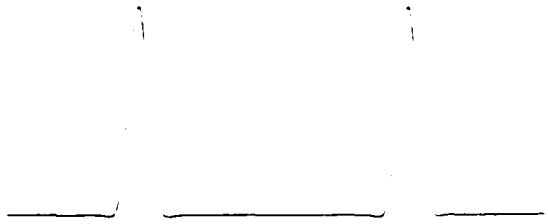
Figure 4.5: (a) A step edge. (b) The step edge smoothed with the LWF filter. $\sigma = 2$. (c) The first derivative of (b).



(a)



(b)



(c)

Figure 4.6: (a) A staircase edge. (b) The staircase edge smoothed with the LWF filter. $\sigma = 2$. (c) The first derivative of (b).

edge, there were marked differences in performance. An analysis of the staircase function follows.

The Gaussian function with scale parameter σ is given by

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-x^2}{2\sigma}} \tag{4.1}$$

The unit step function can be defined as

$$U(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases} \tag{4.2}$$

The staircase function can be represented as the summation of two step functions

$$f(x) = U(x) + AU(x - w) \tag{4.3}$$

where w is the distance between the two steps expressed in units of σ, and A is the ratio of the magnitude of the two steps. For a staircase edge, A is positive.

Convolving the staircase function with $g(x)$ gives

$$\begin{aligned} F_1(x) &= f(x) * g(x) \\ &= \int_{-\infty}^{x} g(t)dt + A\int_{-\infty}^{x-w} g(t)dt \end{aligned} \tag{4.4}$$

Setting the second derivative of equation(4.4) to zero gives

$$-xg(x) - A(x-w)g(x-w) = 0 \tag{4.5}$$

After some algebraic manipulations we get

$$w - x = \frac{xe^{\frac{w^2}{2}-wx}}{A} \tag{4.6}$$

For simplicity and clarity, we will analyze the result in equation(4.6) geometrically. The LOG operator detects edges by finding the zeros of the second derivative of a smoothed signal. This is equivalent to locating the extrema of the first derivative of the smoothed signal. If $w < 2$, overlapping plots of both sides of equation(4.6) intersects once. This is demonstrated in Fig. 4.7a . When the point of intersection is mapped to a plot of the first derivative of the LOG-staircase convolution (Fig. 4.7b), we see that it corresponds to the maxima. This means that the LOG operator detects only one authentic edge.

If $w > 2$, overlapping plots of both sides of equation(4.6) intersect three times as shown in Fig. 4.8. Two of these points coincide with the position of the two maxima in Fig. 4.8b, and are therefore authentic edges. However, the third intersection point corresponds to the minimum between these maxima. Since both maxima and minima identify edge points detected with the LOG operator, the third edge which corresponds to the minimum point is considered a phantom edge.

Performing similar calculations for the LWF gives

$$-\sqrt{2}c_0x-\frac{c_2}{2}(-3x+x^3) = \frac{e^{\frac{w^2}{2}+wx}}{A}\left(\sqrt{2}c_0(x-w)+\frac{c_2}{2}(-3(x-w)+(x-w)^3)\right) \quad (4.7)$$

Edge detection with the LWF is achieved by locating only the maxima in the first derivative of a filtered signal. For $w < 2$, when we look at overlapping plots of both sides of equation(4.7) (Fig. 4.9a)and compare it to a plot of the first derivative of the LWF-staircase convolution (Fig. 4.9b), we see that only one of the intersection points correspond to a maximum: the others coincide with minima. For $w > 2$ (Figs. 4.10a and b), two of the intersection points correspond to maxima. Since only maxima indicate the presence of edges detected with the LWF operator (and the detected minima have no significance), this means that the LWF detects either one ($w < 2$) or two ($w > 2$) authentic edges: it never detects a phantom edge.

1 Experimental results

Experiments were performed using real signals to demonstrate that the LWF does not detect phantom edges. An example is shown in Fig. 4.11 comparing results from the LWF and LOG operators. The size of both operators in this example is $\sigma = 1.5$. In Fig. 4.11a there is an arbitrary signal which has several edges present. An edge exists wherever there is a significant change in the amplitude of the signal. In Fig. 4.11a, the points at which these significant transitions end has been marked

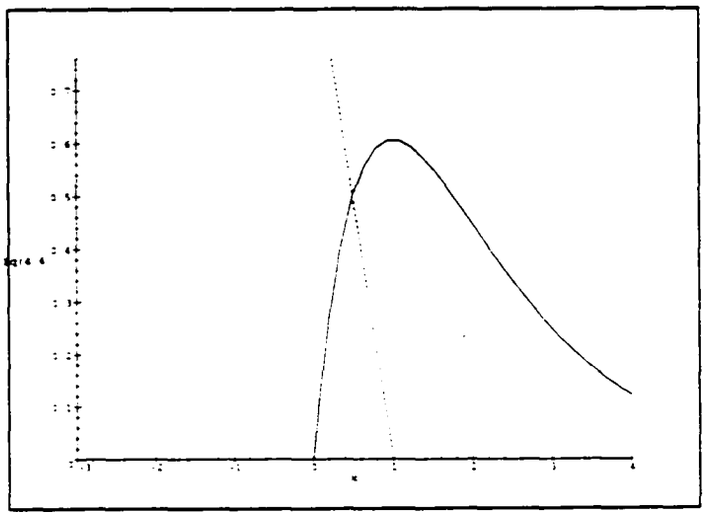
with an 'x'.

Fig. 4.11b shows the result of convolving (a) with the LWF and taking the first derivative. Edges detected by this operator are indicated by peak values in the resulting function, and have been marked with an 'x'. It can be seen that the LWF has identified all the authentic edges in the original signal.

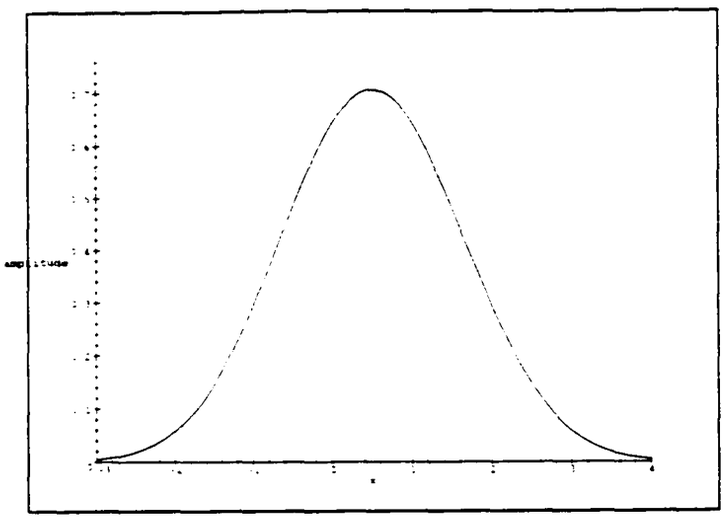
In Fig. 4.11c, the results of the first derivative of the LOG operator convolved with (a) is displayed. The maxima points in this plot which correspond to edges in the original signal have been marked with an 'x'. However, it can be seen that there are other maxima (for example, at the position $x = 29$) which do not correspond to an edge in the original signal. Furthermore, all the minima values in this plot also indicate the presence of edges which do not exist in the original signal.

4.3 Edge Localization Error

An edge detector exhibits good localization if the position of the detected edge is close to that of the true edge. In reference [44], Koplowitz and Greco provide an analysis to measure the location error of edge detectors. The result of their analysis is presented and used to compare the performance of the LWF and Marr-Hildreth operators.

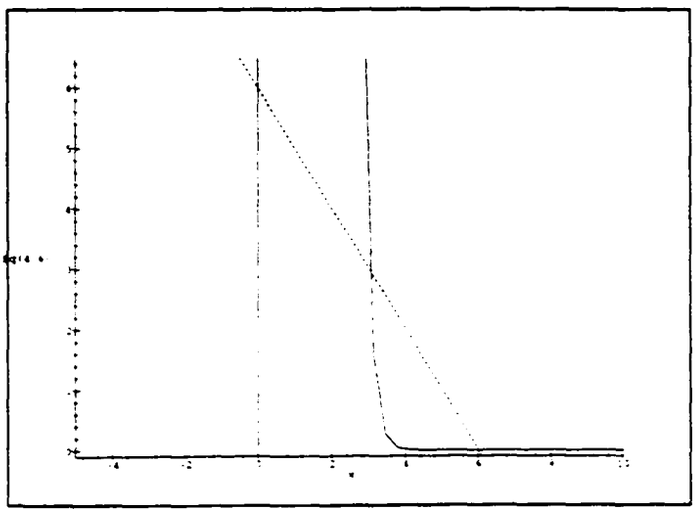


(a)

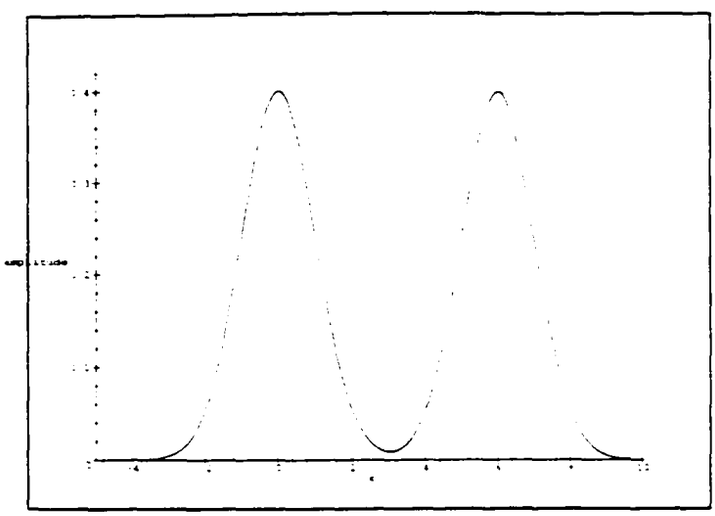


(b)

Figure 4.7: For a staircase edge, $w = 1$: (a) plot of both sides of equation(4.6). The dashed line represents the left-hand side. (b) first derivative of the staircase edge convolved with the LOG operator.

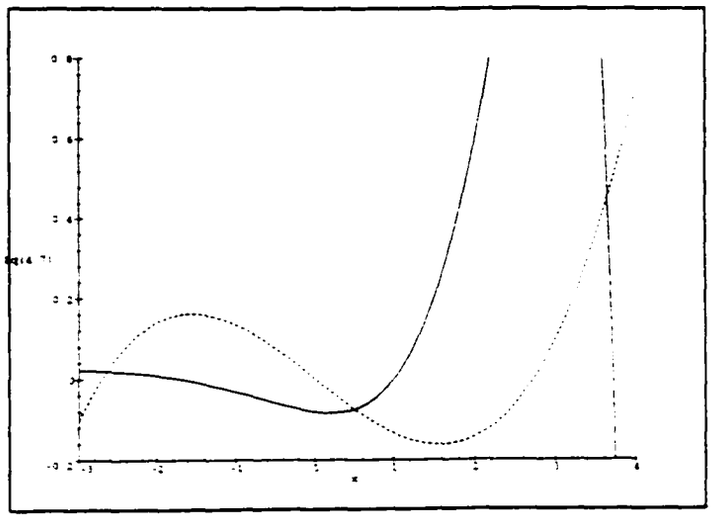


(a)

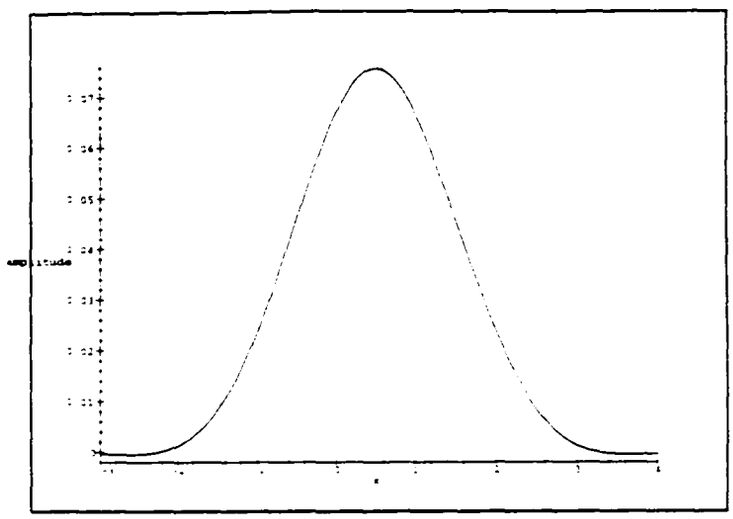


(b)

Figure 4.8: For a staircase edge, $w = 6$: (a) plot of both sides of equation(4.6). The dashed line represents the left-hand side. (b) first derivative of the staircase edge convolved with the LOG operator.

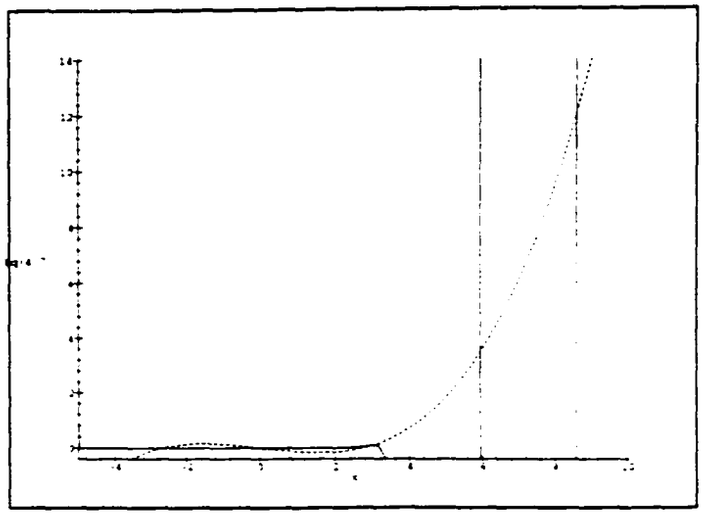


(a)

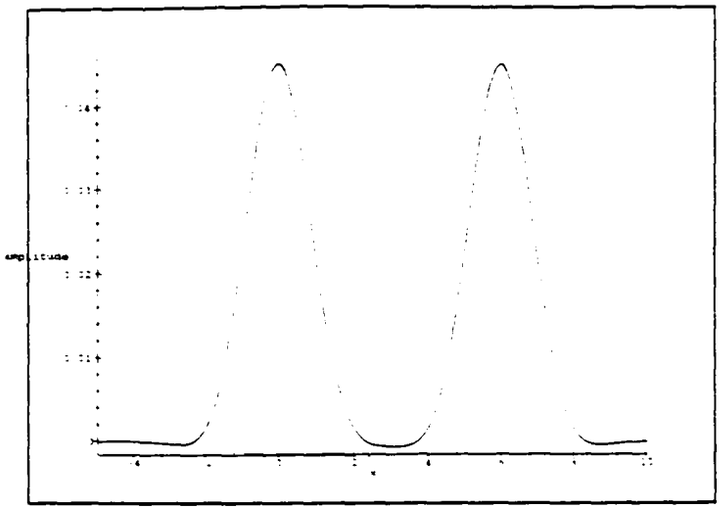


(b)

Figure 4.9: For a staircase edge, $w = 1$: (a) plot of both sides of equation(4.7). The dashed line represents the left-hand side. (b) first derivative of the staircase edge convolved with the LWF operator.

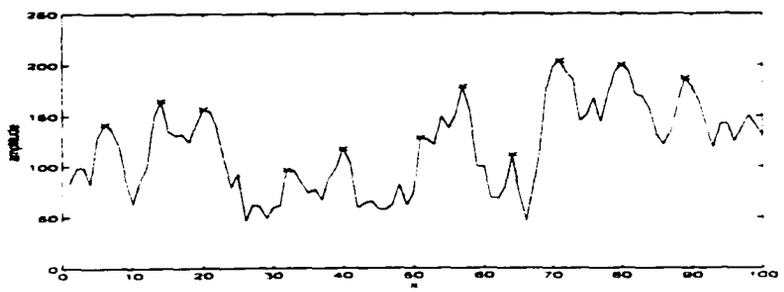


(a)

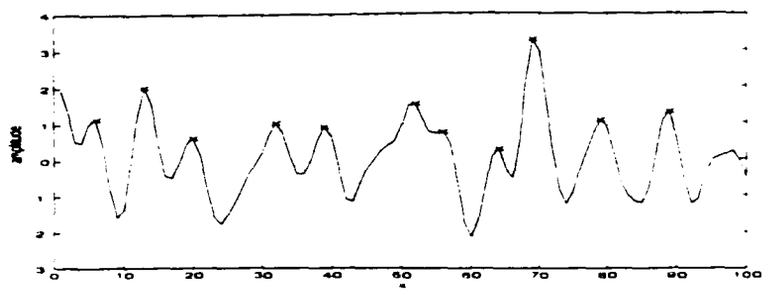


(b)

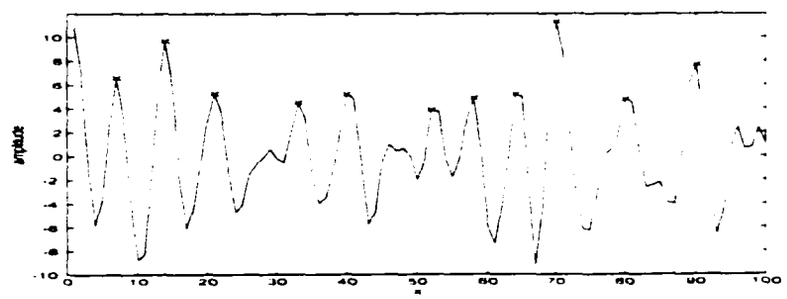
Figure 4.10: For a staircase edge, $w = 6$: (a) plot of both sides of equation(4.7). The dashed line represents the left-hand side. (b) first derivative of the staircase edge convolved with the LWF operator.



(a)



(b)



(c)

Figure 4.11: Experimental results with real data: (a) original signal (b) gradient of convolution with the LWF operator. $\sigma = 1.5$ (c) gradient of convolution with the LOG operator. $\sigma = 1.5$.

Consider a filter $f(x)$ which has a finite impulse response bounded by $[-W, W]$. The response of the filter to noise is

$$H_N(x) = \int_{-W}^{W} f(\alpha) N(x - \alpha) d\alpha \tag{4.8}$$

and the response to an ideal step edge, $G(x)$ is given by

$$H_G(x) = \int_{-W}^{W} f(\alpha) G(x - \alpha) d\alpha \tag{4.9}$$

where

$$G(x) = \begin{cases} 0 & \text{if } x < 0 \\ A & \text{if } x \geq 0 \end{cases} \tag{4.10}$$

The total response of the filter to both the step edge and noise is given by

$$H_T(x) = H_G(x) + H_N(x) \tag{4.11}$$

Define x_0 as the point at which there is a local maximum in the total response. If the detected edge shifts from its true position at $x = 0$ to $x = x_0$, the value x_0 represents the edge location error. Taking a Taylor series expansion of the derivative of the total response of the filter about $x = 0$, and solving for the probability density function of x_0 [44] leads to:

$$Pr(x_0) = \frac{\sigma_1 \sigma_2}{\pi(\sigma_2^2 x_0^2 + \sigma_1^2)} exp - \left(\frac{\mu_1^2}{2\sigma_1^2} + \frac{\mu_2^2}{2\sigma_2^2} \right)$$

$$+\left(\frac{\mu_2\sigma_1^2-\mu_1\sigma_2^2x_0}{\sqrt{2\pi}(\sigma_2^2x_0^2+\sigma_1^2)^{\frac{3}{2}}}\right)$$
$$\times exp-\left(\frac{(\mu_1+\mu_2x_0)^2}{2(\sigma_2^2x_0^2+\sigma_1^2)}\right)$$
$$\times erf\left(\frac{\mu_2\sigma_1^2}{\sqrt{2}\sigma_1\sigma_2}\right) \tag{4.12}$$

where

$$\mu_1 = E[H_N(0)] \tag{4.13}$$

$$\mu_2 = E[H'_G(0)] \tag{4.14}$$

$$\sigma_1 = E[H'^2_N(0)]^{\frac{1}{2}} \tag{4.15}$$

$$\sigma_2 = E[H^2_N(0)]^{\frac{1}{2}} \tag{4.16}$$

The probability density functions of the edge location error for the LWF and LOG filters are shown in Figs. 4.12 and 4.13. Both operators were tested under similar conditions of low noise ($A/\sigma_N = 2$) and high noise ($A/\sigma_N = 10$), where A is the magnitude of the step edge and σ_N^2 is the variance of the additive noise. Both operators also have the same scale value of $\sigma = 1$.

The high noise case occurs when $H'_N(0)$ is significant relative to $H'_G(0)$ and is depicted in Fig. 4.12. The width of a density function can be defined in different ways.

however, for simplicity, it will be defined here as the width where $Pr(x_0)$ drops by 1/2. A comparison of the results in Fig. 4.12 reveal that the density function for the LOG operator (dashed line) is 1.4 times as wide as that of the LWF (solid line). Similar results are obtained in the low noise case (Fig. 4.13) where the density function of the LOG operator is approximately 1.9 times wider than that of the LWF. From this analysis, it can be concluded that edges enhanced with the LWF are less likely to shift from their true locations. In addition, the number of large edge location errors, which may be considered as missed edges, is significantly less for the LWF.

4.4 Summary

We have analyzed the performance of a Gaussian derivative edge operator. We have shown experimentally and verified analytically that the LWF does not detect phantom edges in scale space. An analysis of the edge location error for the LWF shows that it has the advantage of high edge location accuracy in both low and high noise conditions, as compared to the LOG edge detector.

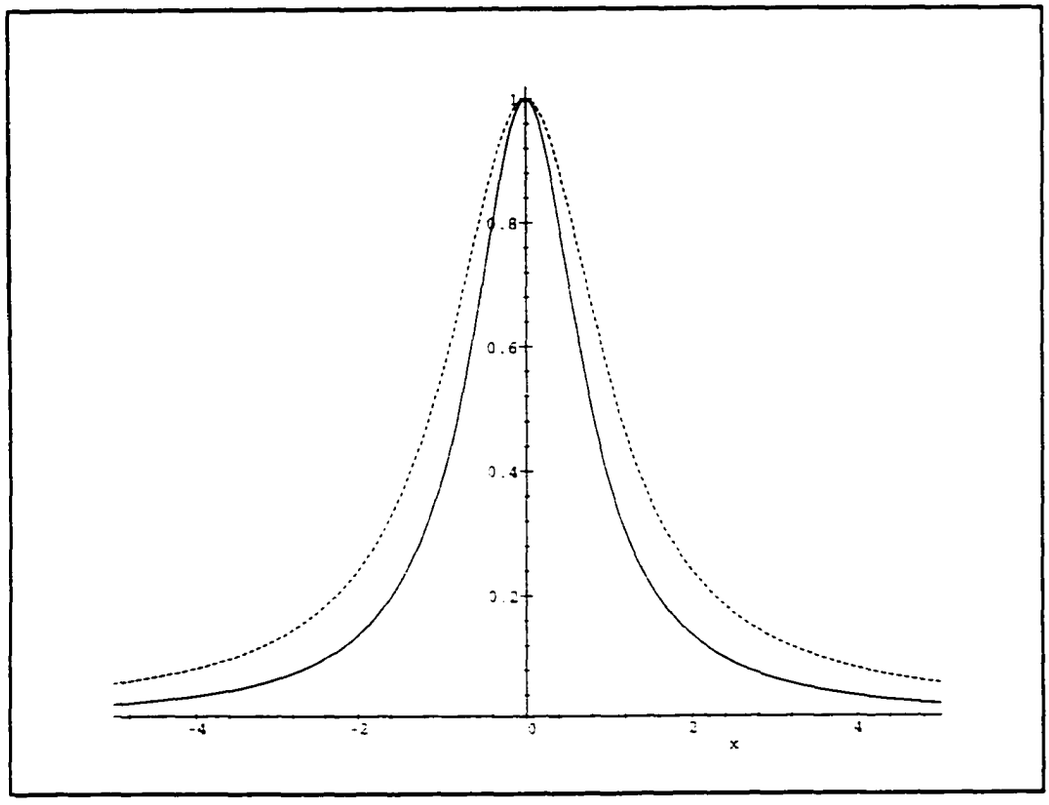


Figure 4.12: Edge location error probability density functions in high noise conditions. $A/\sigma_N = 2$. The dashed and solid curves represents the LOG and LWF operators, respectively.

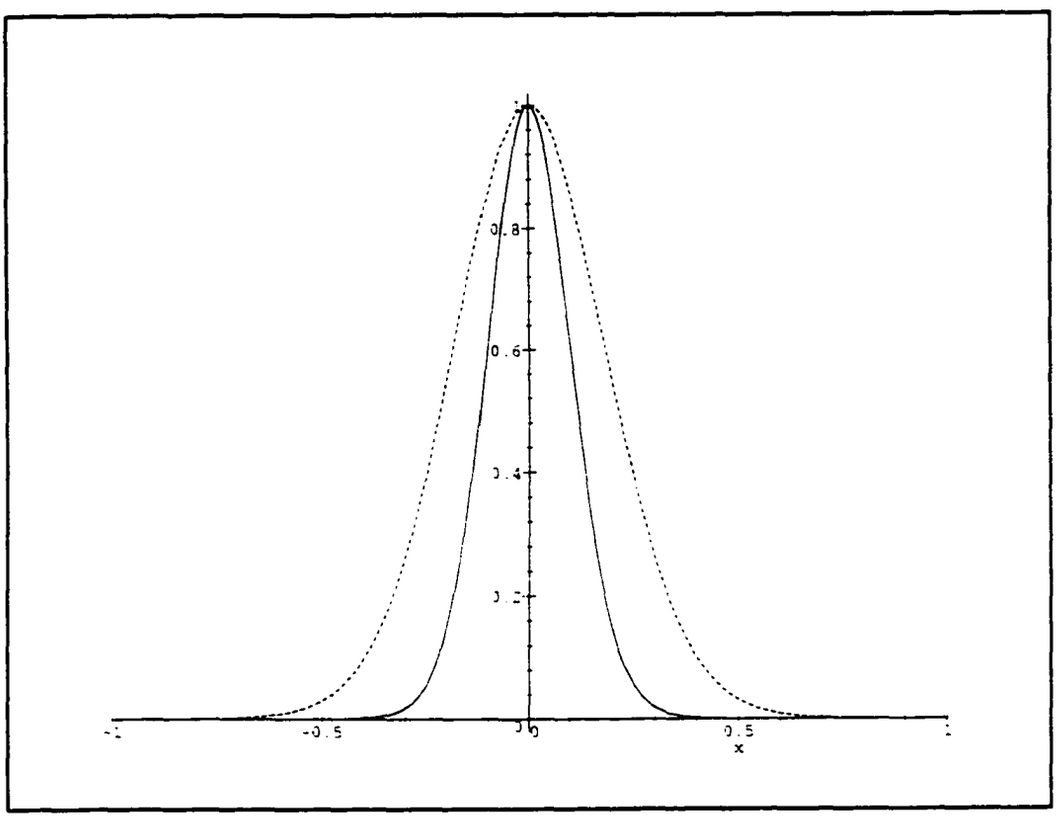


Figure 4.13: Edge location error probability density functions in low noise conditions. $A/\sigma_N = 10$. The dashed and solid lines represent the LOG and LWF operators, respectively.

Chapter 5

Projection Pursuit

One of the main challenges in multivariate data analysis is dealing with the inherent complexities of high dimensional data. This problem is usually tackled by finding and excluding unnecessary dimensions so as to obtain a smaller data set that retains as much information as possible about the original data. Projection pursuit has become one of the most exciting multivariate data analysis techniques due to its ability to avoid the *curse of dimensionality.* The curse results from the fact that data in high-dimensional space is thinly distributed, as illustrated by the following example: assume that a large number of points are distributed uniformly in a 10-dimensional unit ball. The radius of the ball containing 5% of the points is $(0.05)^{0.1} = 0.74$. This implies that kernel smoothers and similar procedures will detect small features only if the sample size is huge [32].

Projection pursuit avoids the curse by finding interesting low-dimensional projections of the high-dimensional data sets. In addition, the more interesting projection pursuit methods ignore extraneous (i.e. noisy and information-poor) variables. This is a definite advantage over techniques based on interpoint distances such as minimal spanning trees, multi-dimensional scaling and most clustering techniques [32].

Projection pursuit methods were first introduced by Kruskal [45, 46]. Kruskal recognized that all properties of the multivariate data form clusters within a data set, and described an interesting linear projection as any projection that displays clustering of the data points regardless of the arrangement of the cluster. He used a projection index that was based on the coefficient of variation, i.e. the data's standard deviation divided by its mean value. This projection index combined the global and local features of the data into a single calculation. However, Kruskal did not propose a practical application of these concepts [27].

Ideas related to projection pursuit can also be found in [72, 73]. Switzer [72, 27] proposed that the likelihood ratio test statistic for a two-sided test of the equality of means could be used to find dichotomous clustering in projections of multivariate normal data onto a line. However, Switzer failed to explore interesting projections by relating this measure to an optimization scheme. He believed that such a scheme would find "multiplicity of decidedly suboptimal local maxima" and would only lead to confusing results.

The first successful implementation of projection pursuit was due to Friedman and Tukey [26] who presented an approach called exploratory projection pursuit that combined a projection index with an optimization scheme. Their projection index classified data points into clusters after the multivariate data had been linearly mapped onto a lower-dimensional space. Projections that displayed more clustering in lower-dimensional space were given higher projection indices. By maximizing the projection index. they were able to pinpoint which projection of the multivariate data had the most interesting information [27].

Years later. Friedman and Stuetzle expanded the concepts of exploratory projection pursuit and introduced projection pursuit regression (PPR) [23]. projection pursuit classification [24] and projection pursuit estimation [25]. A discussion on projection pursuit methods is presented in [32].

The rest of the research in this thesis is concerned with projection pursuit regression and its application as a learning network to specific areas of image processing. Theory and background information on PPR and projection pursuit learning networks are presented in the following sections of this chapter.

5.1 Projection pursuit regression

In 1981, Friedman and Stuetzle [23] introduced projection pursuit regression (PPR) as a new technique for nonparametric multivariate regression. In a regression problem, there is a p-dimensional random vector $\mathbf{X}$, whose components are called predictor variables, and a random variable Y, called the response. A regression surface depicts a general relationship between $\mathbf{X}$ and Y. Traditionally, in addressing the regression problem, it was usually assumed that the functional form of the regression surface was known, thereby minimizing the problem to one of determining a set of parameters. However, it is usually difficult to prove the results, and a wrong assumption can cause incorrect or misleading results. This makes the use of nonparametric methods which make only a few general assumptions about the regression surface highly desirable.

Friedman and Stuetzle's concept of projection pursuit regression avoided many difficulties experienced with other existing nonparametric regression procedures. The poor performance in high dimensions (curse of dimensionality) encountered in kernel and nearest-neighbor methods is avoided in PPR since all estimation (smoothing) is carried out in a univariate setting. Unlike recursive partitioning, PPR does not split the predictor space into two regions thereby allowing, when necessary, more complex models. In addition, interactions of predictor variables are directly considered since linear combinations of the predictors are modeled with general smooth functions [23].

Another significant property of PPR is that the results of each iteration can be depicted graphically. The graphical output can be used to modify the major parameters of the procedure: the average smoother bandwidth and the terminal threshold. The PPR method can also be applied to the residuals from any initial model. If the initial model does not fit the data well. PPR will indicate this by augmenting the model [23].

All stepwise procedures have difficulties modeling regression surfaces that can not be adequately described by models of low complexity in their hierarchy. Because models in PPR are sums of functions, each varying only along a single linear combination of the predictor variables. PPR has difficulty modeling regression surfaces that vary with equal strength along all possible linear combinations [23].

Diaconis and Shahshahani [17] examined PPR theoretically. They addressed the projection pursuit algorithm from the perspective of the approximation theory. They analyzed the necessary and sufficient conditions for the functions to be exactly represented as a linear combination of nonlinear functions, and discussed the nonuniqueness of the representation.

Donoho and Johnstone [19] studied the duality between PPR and kernel regression in two dimensions. Their results indicated that PPR produces function estimates which behave well when the underlying function is angularly smooth (oscillating slowly with angle), while kernel regression was suitable for functions with sufficient Laplacian

smoothness, such as harmonic functions and solutions to the inhomogeneous heat equation (oscillations averaging out locally). They also showed that if the function to be estimated has nice tail behavior, PPR lowers the dimensionality of the problem. In their case, it behaved as if the dimension of the problem were 1.5 rather than 2. PPR and kernel regressions turn out to complementary: for a given function, if one method offers a dimensionality reduction, the other does not.

Hall [30] devised a tractable mathematical model to describe PPR. The model allows computation of explicit formulae for bias and error about the mean in orientation estimates and curve estimates. The results indicated that the estimate of orientation has most of its error in the bias, and that the error about the mean is asymptotically negligible in comparison to the bias. Hall also proved that the common form of kernel-based PPR does estimate projections with the same convergence rates as those experienced in one-dimensional problems, although a greater degree of smoothness (an extra derivative) must be assumed to accomplish this.

Chen [11] proposed a PPR scheme with two additional constraints imposed so that the rate of convergence of the estimator is independent of dimensionality. He concluded that to resolve the conflict between the necessity for flexible modeling and the curse of dimensionality due to p-dimensional local averaging, there needs to be a compromise between a parametric regression model and a nonparametric regression model. Let $(\mathbf{X}, Y)$ be a random vector such that $\mathbf{X} = (\mathbf{X_1}, \ldots, \mathbf{X_p})^T$ ranges over R^p.

The regression function of Y on $\mathbf{X}$ is $E(Y/\mathbf{X} = \mathbf{x})$, and is assumed to be the sum of no more than p general smooth functions of $\beta_i{}^T\mathbf{x}$, where $\beta_i \in S^{p-1}$, the unit sphere in R^p centered at the origin. Under suitable conditions, the rate of convergence of the proposed estimator is independent of p.

5.2 Projection pursuit learning networks

A projection pursuit learning network (PPLN) possesses a structure very similar to a one hidden-layer neural network (with sigmoidal nonlinearity). In the case of the PPLN, the sigmoidal functions are replaced by unknown functions to be learned from the data. Since the sigmoidal functions are replaced by more general functions, PPLN can be viewed as a generalization of a one hidden-layer sigmoidal feedforward neural network.

The nonparametric regression problem can be stated as follows: given n vector pairs in a p-dimensional space

$$(\mathbf{y}_l, \mathbf{x}_l) = (y_{l1}, y_{l2}, \ldots, y_{lq}; x_{l1}, x_{l2}, \ldots, x_{lp}), \qquad l = 1, 2, \ldots, n \tag{5.1}$$

that have been generated from unknown models

$$y_{li} = g_i(\mathbf{x}_l) + \epsilon_{li}, \quad l = 1, 2, \ldots, n; \quad i = 1, 2, \ldots, q \tag{5.2}$$

the aim of regression is to construct the estimators, $\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_q$, and to use these estimates to predict a new **y** given a new **x**.

The PPLN for regression [36, 81] is mathematically modeled as a one-hidden layer feedforward network (see Fig. 5.1) and it approximates a function using:

$$\hat{y}_i = \bar{y}_i + \sum_{k=1}^{m} \beta_{ik} f_k(\sum_{j=1}^{p} \alpha_{kj} x_j) \tag{5.3}$$

where β_{ik} are the projection strengths, f_k are the unknown smooth activation functions and α_{kj} are the projection directions. These three parameters are determined by training the network to minimize the mean squared error loss function:

$$L_2 \equiv \sum_{i=1}^{q} W_i E(y_i - \hat{y}_i)^2 \tag{5.4}$$

where E is the expectation operator defined as

$$E(y_i) = \frac{1}{n} \sum_{l=1}^{n} y_{li} \quad = \bar{y}_i \tag{5.5}$$

and the weights W_i indicate the relative contribution of each mean squared output error to the total L_2 loss.

The traditional training algorithm for a PPLN trains the hidden units one at a time, as opposed to all at once as is the case in a backpropagation neural network. The algorithm can be described as follows for the k-th hidden layer neuron:

1. Make initial guesses for α_k, f_k and β_k.

2. Estimate $\hat{\alpha_k} = \alpha_k + \Delta$ using an iterative optimization method.

3. Given $\hat{\alpha_k}$, estimate $\hat{f_k}$ as the smooth curve which best fits the scatterplot $[z_{kl}, f_k^*(z_{kl})]$, where $z_{kl} = \hat{\alpha_k}^T x_l$.

4. Repeat steps 2-3 for several iterations.

5. Use the most recent values of $\hat{f_k}$ and $\hat{\alpha_k}$ to evaluate β_{ik}. (β_{ik} can be computed by setting the derivatives of the loss function L_2 with respect to β_{ik} equal to zero).

6. Repeat steps 2-5 until the loss function is minimized with respect to all β_{ik}, α_k and f_k associated with the k-th neuron.

This procedure is then repeated for the $(k+1)$-th hidden layer neuron.

5.3 Survey on projection pursuit learning networks

Projection pursuit was first introduced in the context of learning networks by Barron and Barron [4]. They analyzed several network schemes and algorithms based on the definition that a *learning network* estimates its function from representative observations of the relevant variables. They considered three types of methodologies - neural networks, adaptive polynomial learning and nonparametric statistical

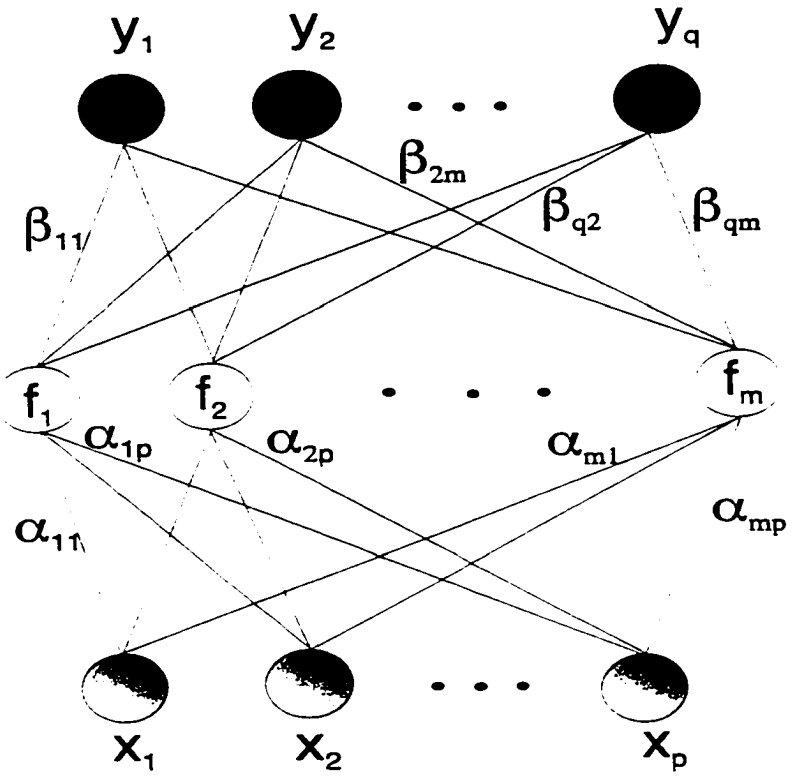


Figure 5.1: A standard PPLN

inference – and found substantial similarities in these fields. They found that the most successful learning network strategies adaptively grew the network structure using all the observational data, and used a suitable model selection criterion to ensure a parsimonious network. The best methodologies also used network structures which were not limited in their approximation capabilities. The primary examples of these successful methodologies were adaptively synthesized polynomial networks and projection pursuit.

Barron and Barron [4] stated that an advantage of projection pursuit networks is that they have been amenable to theoretical examination of some of their approximation properties [18, 32, 40]. In particular, it is known that any square integrable function can be approximated by a theoretical analog of projection pursuit provided sufficiently many levels of the network are utilized.

Intrator [37, 38] applied exploratory projection pursuit to do feature extraction in speech recognition. He defined a measure of multimodality for finding interesting projections based on the observation that high-dimensional clusters translate to multimodal low-dimensional projections. When the data is known in advance to be bimodal, it is relatively straightforward to define a good projection index. However, Intrator considers the case when the structure is not known in advance and defining a general multimodal measure of the projected data is not straightforward.

Intrator formed an objective function whose minimization finds projections with one-

dimensional projected distributions that are not Gaussian. This was done using a loss function that has an expected value that leads to the desired projection index. Consider a neuron with input vector $x = (x_1, \ldots, x_N)$ and synaptic weights $w = (w_1, \ldots, w_N)$, the loss function is given by

$$L_w(x) = \frac{-\mu}{3}\left\{\sigma^3(x \cdot \omega) - E[\sigma^2(x \cdot \omega)]\sigma^2(x \cdot \omega)\right\} \tag{5.6}$$

where μ is the learning rate and σ is usually a sigmoidal function.

Based on this formulation, a network can be created that consists of identical nodes which all receive the same input and inhibit each other so as to extract features in parallel.

Intrator compared the classification performance with speech data of the proposed method and a backpropagation network. He found that less features were extracted with the backpropagation network, as this method concentrated mainly on one particular feature of the speech signal and stopped training when the misclassification error fell to zero. The proposed network did not try to reduce the misclassification error and was therefore able to find a significant structure in the speech signals.

Maechler et al. [52] and Hwang et al. [35, 36] studied several two-dimensional examples to compare projection pursuit learning network (PPLN) and backpropagation neural network (BPNN). They limited their study to the regression problem because there is a better theoretical understanding of the approximation quality of PPLNs.

They point out that both types of networks form projections of the data in directions determined from the interconnection weights. However, a BPNN uses a fixed set of nonlinear activations that are parametrically specified by a fixed finite set of parameters, whereas a PPLN estimates nonlinear activations based on an optimization scheme that uses a one-dimensional nonparametric data smoother.

In a simulation study, Maechler et al. [52] found that PPLNs and sequential gradient descent BPNNs have similar accuracy for independent test data, but PPLNs are two orders of magnitude faster in training. They concluded that the difference in speed was due to the fact that smoothing in the BPNN involves using many parameters estimated by gradient search. PPLN, on the other hand, uses an efficient optimization strategy. Least squares is used to determine the linear part of the weights, and the second Gaussian method is used to determine the nonlinear part. Furthermore, data-smoothing techniques allow fitting non-parametric nonlinear nodal functions, so the network adapts more quickly to the observation data.

When Hwang et al. [35] compared PPLN and batch Gauss Newton BPNN, they found that both networks have comparable training speed and accuracy for independent test data. However PPLNs are significantly more parsimonious as they require fewer neurons.

In the original PPLN for regression, a variable span smoother, called the supersmoother, was used to obtain smooth estimated activation functions. Hwang et al.

[35, 36] pointed out that nonparametric smoothers lead to the use of large regression tables, unstable approximation in calculating derivatives, and piecewise interpolation in computing activation values. As a result, they proposed using parametric smoothers that are a linear combination of Hermite functions. The Hermite polynomial approximation of the non-linear activations outperforms the supersmoother version of the PPLN in several aspects of performance evaluations.

Hwang et al. [36] also compared a cascade-correlation learning network (CCLN) and PPLN. A CCLN is a unit-growing learning network that increases the size and depth of its hidden layer during training. Like a PPLN, a CCLN forms new candidate hidden units one at a time. The weights of each candidate unit are trained by keeping the weights of the other existing hidden units constant. Unlike a PPLN in which candidate units receive only weighted connections from all the input units, a CCLN's candidate unit receives weight connections from all the input units as well as all the other existing hidden units. This enables a CCLN to detect higher-order features.

Hwang et al. [36] found that the cascaded connections used to enable higher-order feature detection in a CCLN are not necessary in a PPLN, since the trainable activations have the same purpose. Training in a CCLN is also more difficult as the cascaded weights increase the input dimensions of the candidate hidden units using a fixed simple nonlinearity activation. A PPLN, on the other hand, works with the same input dimensions, but uses appropriate hidden activation functions. Further-

more, the maximum correlation criterion used in a CCLN to avoid cyclic updating between layers of weights, usually produces saturated hidden units. This results in unsmooth, zigzag classification/regression surfaces, making the CCLN unsuitable for use in most regression applications. The Hermite-based PPLN, on the other hand, uses the minimum L_2 criterion and produces smoother classification/regression surfaces.

Flick et al. [21] use a projection pursuit methodology for pattern classification which is based on likelihood ratio estimation. The authors chose to use this approach because the likelihood ratio is an optimal discriminant in a Neyman-Pearson sense, and the Neyman-Pearson classifier requires only a monotone function of the likelihood ratio. Consequently, they need to estimate only a single function rather than work with the ratio of two estimated probability density functions. The monotone function they chose to estimate is bounded above and below, and presents few numerical problems. They show that a measure of scatter can be associated with this function, and that minimization of this scatter is equivalent to finding the optimal discriminant.

Flick et al. [21] discuss two algorithms - linear series and nonlinear series - for obtaining discriminant functions from training set data. The projection pursuit scheme in both algorithms is iterative, and at each iteration, a single new ridge function (estimator) and associated projection direction is introduced. In the linear

series algorithm, a linear combination of the ridge functions presented up to the current iteration is used to estimate the monotone function. In the nonlinear series algorithm, a well chosen nonlinear combination of ridge functions is used. In both cases, the direction of the new projection and associated ridge function are chosen to minimize the measure of scatter estimated from the training set.

Zhao et al. [81] investigated the implementation issues of projection pursuit regression (PPR). They applied parametric PPLNs to learn the inverse dynamics of robot arms. This is a highly non-monotonic, pure approximation problem. They use the robot arm example to examine the performance of PPR in a high-dimensional space ($d = 6$).

The authors showed that a PPLN can learn simple arm dynamics fairly well, and that parametric PPR with a direct training method achieves better accuracy and training speed than nonparametric PPR. The parametric representation for PPR that they proposed is

$$\hat{y}(\mathbf{x}) = \sum_{\mathbf{j}=1}^{\mathbf{n}} \sum_{\mathbf{i}=1}^{\mathbf{P}} \mathbf{c_{ij}} \mathbf{G}(\mathbf{x} \cdot \alpha_{\mathbf{j}}, \mathbf{t_i}) \tag{5.7}$$

where α_j are direction parameters, t_i are fixed equispaced partial data projections for all directions, and $G(s, t)$ are one-dimensional weight functions which usually take a symmetric form $G(s, t) = G(|s - t|)$.

The parametric projection pursuit network has the advantage of achieving better

accuracy with fewer parameters than a one hidden layer sigmoidal neural network.

Kwok and Yeung [47] improved the PPLN proposed by Hwang et al. [35, 36]. Hwang et al. used a parametric smoother based on Hermite functions of some predefined highest order R. Kwok and Yeung found that sometimes the order R is important for successful approximation, and choosing a wrong value could lead to poor results in training and testing. This is because a PPLN with a fixed R does not have the universal approximation property for any finite R, and therefore cannot converge to the desired function even with an arbitrarily large number of hidden units. Kwok and Yeung suggest that it is possible to keep R fixed and still achieve universal approximation simply by including a bias term into each linear combination of the predictors, i.e.

$$\hat{y}_i = \sum_{j=1}^{n} \beta_j f_j(\alpha_j{}^T \mathbf{x} + \theta_j) \tag{5.8}$$

where θ_j is the bias term.

They also show experimentally that this modification increases the rate of convergence with respect to the number of hidden units and improves the generalization performance.

As of this writing, we have only come across two works in which projection pursuit learning is applied to image processing problems. Safavian *et al.* [66, 67, 62] proposed a new compression algorithm that fits various artificial neural network models to

different segmented image blocks utilizing the theory of PPLNs. The segmentation phase of their algorithm divides the image into variable-size square blocks based on a measure of activity within the block. The size of the finest block is dictated by the combination of the desired bit rate and the desired peak signal-to-noise ratio.

Once the image has been segmented into various size regions, each block can be coded using the projection pursuit image approximation technique. In every iteration in this phase of the algorithm, a function is selected that best approximates the current image in the given block, from a set of fixed pre-determined basis functions. In the first step of the iteration, the current image is the original image and in the k-th step the current image is the residual image that results from subtraction of the original image and linear combination of all the $(k-1)$-th previous approximations. The optimum parameter for each block must then be quantized before encoding. Based on the distribution of the weights and the biases and their dynamic ranges for each block, separate quantizers were designed for each set of parameters at each iteration of the algorithm. The resulting quantized parameters are then coded using an arithmetic coder.

Experimental results show that at rates below 0.5 bits/pixel, this algorithm performs better than JPEG in terms of peak signal-to-noise ratio and subjective image quality.

Hulle [33] proposed a novel approach to nonparametric regression analysis using topographic maps. The maximum entropy learning rule was extended with a neigh-

borhood function, and the extended rule, called eMER was applied in combination with projection pursuit regression learning. The high-dimensional data were interpreted through optimally chosen lower dimensional projections in which topographic maps were trained using eMER. The position of the neurons in the maps were joined using non-smoothing cubic splines.

Hulle applied the eMER/PPR combination to adaptive filtering of gray-scale images to verify the regression performance in high-dimensional spaces. The eMER/PPR combination was trained on a noisy subimage, then the regression model obtained at convergence was applied on the full noisy image. Experimental results show that the model obtained after training reduces the noise content of the full image by more than 20 dB.

Chapter 6

Experiments With Projection Pursuit Learning Networks

Neural networks have been widely used in image processing, one of the most common applications being modeling mammalian visual systems and their effectiveness at pattern recognition. However, very little has been done using PPLNs in this field despite the similarity in theory to neural networks. We have chosen to apply PPLNs to two fundamental areas of image processing – boundary detection and image restoration. In regards to boundary detection, we want to improve on the work presented in the earlier chapters of this thesis by connecting broken edge segments and producing more continuous boundaries. For image restoration, we address the problem of obtaining the accurate representation of an original image when presented

with its blurred version.

Our choice of PPLN over backpropagation neural (or radial-basis function) network is dictated by the following:

1. PPLN is better at interpolation/extrapolation of data sets.
2. PPLN requires fewer hidden neurons to approximate a true function.
3. PPLN is better with noisy high-dimensional data.

6.1 Hermite polynomial-based PPLN

We use a Hermite polynomial-based PPLN as described by Hwang*et al.* in [35, 36]. The orthonormal Hermite functions [29] are defined by

$$h_r(z) \equiv (r!)^{\frac{-1}{2}} \pi^{\frac{1}{4}} 2^{\frac{-(r-1)}{2}} H_r(z)\phi(z) \tag{6.1}$$

where $H_r(z)$ are the Hermite polynomials constructed in a recursive manner

$$\begin{aligned} H_0(z) &= 1 \\ H_1(z) &= 2z \\ H_r(z) &= 2z(zH_{r-1}(z) - (r-1)H_{r-2}(z)) \end{aligned} \tag{6.2}$$

and ϕ is the weighting function

$$\phi(z) = \frac{1}{\sqrt{2\pi}} e^{\frac{-z^2}{2}} \tag{6.3}$$

used to normalize $h_r(z)$.

Recall from the PPLN training algorithm (Section 5.2) that the scatterplot $[z_{kl}, f_k^*(z_{kl})]$ for the k-th neuron has points located at $z_{kl} = \alpha_k^T \mathbf{x}_l$.

Let

$$\begin{aligned} \mathbf{y}_k &= (f_k(z_{k1}), f_k(z_{k2}), \ldots, f_k(z_{kn}))^T \\ \mathbf{h}_{kl} &= (h_1(z_{kl}), h_2(z_{kl}), \ldots, h_R(z_{kl}))^T \end{aligned}$$

$$\mathbf{H}_k = \begin{pmatrix} \mathbf{h}_{k1}^T \\ \mathbf{h}_{k1}^T \\ \vdots \\ \mathbf{h}_{k1}^T \end{pmatrix}$$

$$\mathbf{c}_{kl} = (c_{k1}, c_{k2}, \ldots, c_{kR})^T$$

The least squares (LS) estimate $\hat{\mathbf{c}}_k$ of the coefficient vector $\mathbf{c}_k$ is obtained by

$$\hat{\mathbf{c}}_k = (\mathbf{H}_k^T \mathbf{H}_k)^{-1} \mathbf{H}_k^T \mathbf{y}_k \tag{6.4}$$

The resulting scatterplot is smoothed by LS fitting of the order R Hermite functions

$$\hat{\mathbf{f}}_k(z) = \sum_{r=1}^{R} \hat{\mathbf{c}}_{kr} h_r(z) \tag{6.5}$$

Using the special property of Hermite functions [29] that

$$h_r'(z) = (2r)^{1/2} h_{r-1}(z) - z h_r(z) \tag{6.6}$$

the derivative of $\hat{f}_k(z)$ is given by

$$\hat{f}_k'(z) = \sum_{r=1}^{R} \hat{c}_{kr} [(2r)^{1/2} h_{r-1}(z) - z h_r(z)]. \tag{6.7}$$

6.2 Model selection strategy for PPLNs

The construction of the PPLN consists of two steps - a forward growing procedure and a backward pruning procedure. During the forward growing procedure, hidden layer neurons are added and optimized one at a time using the training algorithm described in Section 5.2. After the parameters associated with the neuron under

consideration have been estimated, a *backfitting* technique is used to update the parameters of previously installed neurons.

Once the network has grown to m^* neurons, a backward pruning procedure is applied to remove overfitting neurons one at a time. The PPLN again uses the backfitting procedure to fit models of decreasing size, $\tilde{m} = m^* - 1, m^* - 2, \ldots, m$, to the data, where m^* and m are specified by the user. The most important $\tilde{m}$ out of $(\tilde{m} + 1)$ hidden neurons are kept at each step. The *importance* is measured by

$$I_k = \sum_{j=1}^{q} W_j |\hat{\beta}_{jk}|, \qquad k = 1, \ldots, \tilde{m} + 1 \tag{6.8}$$

where $\hat{\beta}_{jk}$ are the estimates for the $(\tilde{m} + 1)$-hidden neuron model.

To determine an appropriate number of neurons for the PPLN model, we applied a strategy used in [70] for nonparametric regression. First, we run the PPLN with $m = 1$ and set m^* at a value large enough for the problem at hand. For a relatively small number of variables p ($p \leq 4$), we choose $m^* \geq p$. For large p, we choose $m^* < p$, hoping for a parsimonious representation.

For each neuron $\tilde{m}$, $1 \leq \tilde{m} \leq m^*$, the PPLN evaluates the fraction of unexplained variance

$$e^2(\tilde{m}) = \frac{\sum_{i=1}^{q} W_i \left[y_i - \bar{y}_i - \sum_{k=1}^{\tilde{m}} \hat{\beta}_k \hat{f}_k(\hat{\alpha}_k^T \mathbf{x}_i) \right]^2}{\sum_{i=1}^{n} W_i [y_i - \mathbf{y}_i]^2} \tag{6.9}$$

In a plot of $e^2(\tilde{m})$ versus $\tilde{m}$ which is decreasing, $e^2(\tilde{m})$ often decreases rapidly when $\tilde{m}$ is smaller than a good number of neurons m_0, and then tends to decrease more slowly for $\tilde{m}$ larger than m_0. As a result, we must run the PPLN twice for $\tilde{m} = m, \ldots, m^*$, using two different values of m. The first time $m = 1$ is used to find a good number of neurons m_0. The second time $m = m_0$ is used to obtain the output of the simulations.

6.3 PPLN applied to boundary detection

Edge detection is an important step in most image understanding applications. Its purpose is to identify the areas of an image where large changes in intensity occur. These changes are often associated with some physical boundary in the scene from which the image is derived. The ultimate goal, however, is to develop a continuous set of pixels forming a closed contour around each object of interest in the image. Edge detection alone is not sufficient to accomplish this task. It must usually be combined with more heuristic rule-based algorithms which track the boundaries and link edge segments to form complete contours.

Refer to the edge detected Lenna image in Fig. 6.1. The clear outlines of the original image are well defined in the edge detected version, however, there are also numerous unconnected points in the image. For instance, in the outline of the hat several disconnected edge segments can be seen. This weakness of the edge detection routine

makes it necessary to perform further steps to form continuous boundaries. In this section we address this problem using a PPLN.

We choose to use the Hermite polynomial-based PPLN because of the similarity of our problem to that explored by Hwang *et al.* [35]. They investigated how to obtain a smooth surfaces from noisy data for nonlinear functions. We are investigating how to get smooth, continuous boundaries from scattered edge points in images.

1 Method

We formulate the boundary detection problem as a curve-fitting problem. The edge points (true or otherwise) are detected using the LWF. The problem is to find a smooth curve which *best fits* all the true edge points. We examined several attributes of the edge points, but the features which played the greatest role in determining which points lie on a boundary were the pixel's gradient magnitude and direction. The pixel's gradient magnitude is given by

$$G(x, y) = [G_x^2 + G_y^2]^{1/2} \tag{6.10}$$

where G_x and G_y are the values of the magnitude of the gradient in the x and y direction, respectively. The edge pixel's gradient direction is

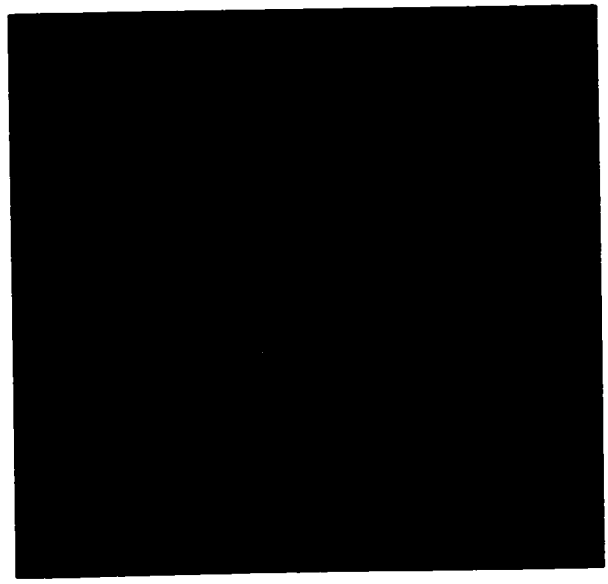
$$\theta_g = tan^{-1}(G_y/G_x) \tag{6.11}$$

We considered images sized 512 x 512 pixels from which we selected a subimage sized 128 x 128 pixels for training. The PPLN had two inputs and one output. During training, the inputs are the gradient magnitude and direction of each pixel in the subimage, and the output is the associated edge value for each pixel (i.e., whether or not the pixel under consideration is an edge). During testing, we used the gradient magnitude and direction for a full image. The data for the training phase was generated from an image convolved with the LWF at a small scale so as to provide sufficient examples for learning. If too few edge points are specified during training the PPLN is not able to predict well when presented with new, unknown data values.

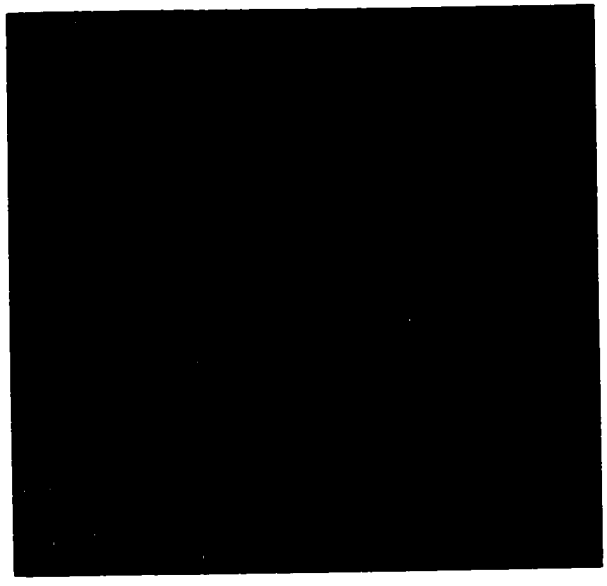
For the simulations, we used a PPLN with 3 hidden neurons with $m^* = 5$ maximum hidden neurons. In most cases, we found that a Hermite polynomial of order R between 9 and 15 provided the best results. For all the simulations shown here, we used $R = 9$. As an example, consider the edge image of Lenna ($\sigma = 1$) in Fig. 6.1a. The training was performed using a 128 x 128 area in the center of the image. We then applied the PPLN model obtained at convergence on the edge image at $\sigma = 3$ in Fig. 6.1b. The edge image obtained after regression is shown in Fig. 6.1c. There is clearly an improvement over the original data in Fig. 6.1b, in the sense that new edge segments have now been incorporated into the edge map, providing a more complete picture. Unfortunately, the regression procedure has also resulted in thickening of the lines, as if trying to create smooth surfaces among the data points. A solution

to this problem would be to apply an edge thinning algorithm to the results of the PPLN regression procedure. The quality of the data at that point, however, is directly related to the effectiveness of the edge thinning algorithm. An example of edge thinning applied to the regression results is shown in Fig. 6.1d.

Once the PPLN was trained, it produced similar results when tested on other images. Fig. 6.2a shows another image – the Peppers image – which was used to test the PPLN after it had been trained on the same 128 x 128 subimage of the Lenna edge map. Again, the PPLN does improve on the original testing data by producing more continuous boundaries and additional edge segments (Fig. 6.2b). The result after edge thinning is shown in Fig. 6.2c.

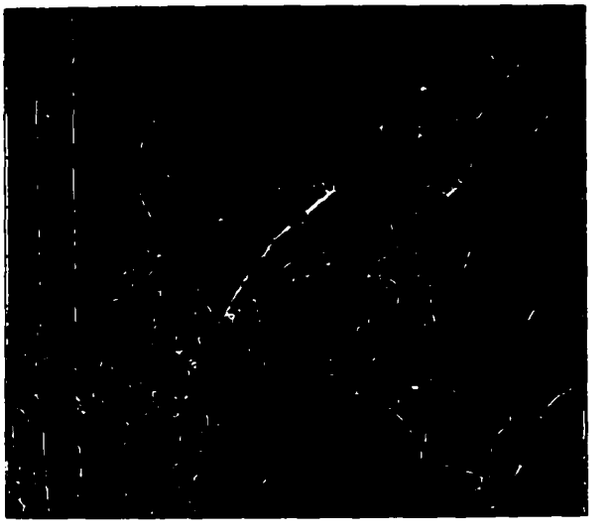


(a)



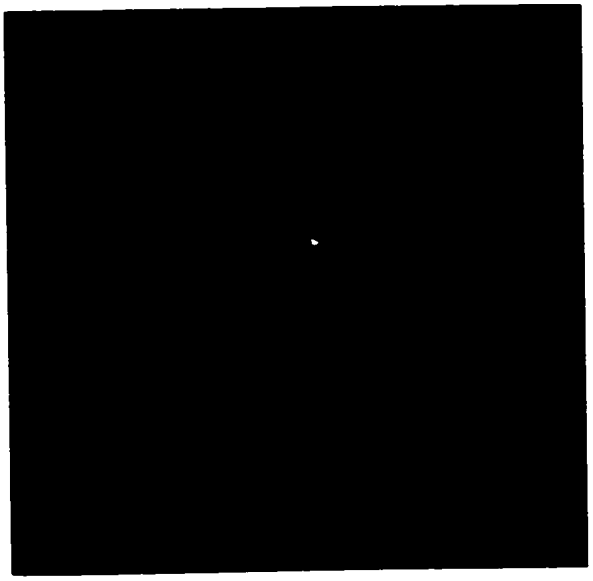
(b)

(c)



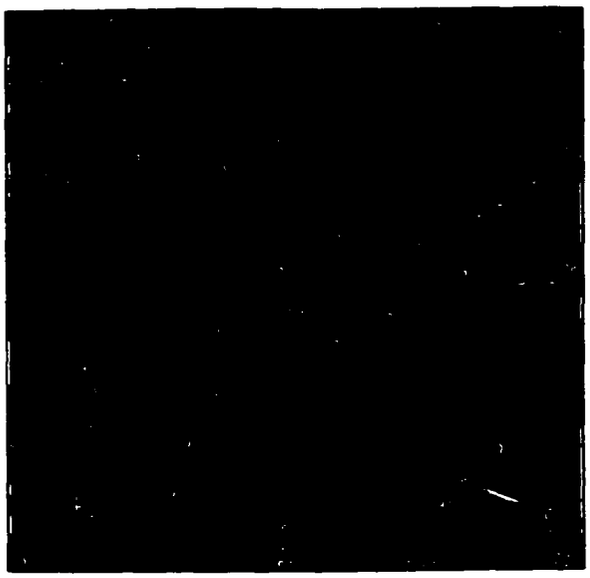
(d)

Figure 6.1: (a) Edge detection of Lenna image, $\sigma = 1$. A 128 x 128 area in the center of the image is used for training (b) Edge detection of Lenna image for testing. $\sigma = 3$ (c) Results of PPLN applied to (b); (d)Results of edge thinning applied to (c).



(a)

(b)



(c)

Figure 6.2: (a) Edge detection of Peppers image for testing, $\sigma = 3$ (c) Results of PPLN applied to (b); (d)Results of edge thinning applied to (c).

6.4 PPLN applied to image deblurring

In image restoration, an image has been degraded in some manner and the objective is to reduce or eliminate the degradation. The most common types of degradation are additive random noise and blurring. The application of PPLN to noise removal in images was explored in [33]. In this section we consider the use of PPLNs for deblurring images.

1 Background

Blurring is hard to avoid in any image acquisition system, and can be caused by many sources such as: (i) atmospheric turbulence, (ii) an out-of-focus optical system (which results in the point spread function of the imaging system not being an impulse function), and (iii) aberrations in the imaging system. Obviously, all these causes cannot be simultaneously captured in a simple model. However, by the central limit theorem of probability theory, the net result of the complex interplay among these independent random sources can, in general, be approximated by a Gaussian blur [75].

Before defining any solution, we must know how the image was formed, i.e. we must have a mathematical model of the image formation system. For this work, the linear shift invariant model will be used. The linear model gives the following relation:

$$g(m,n) = h(m,n) * f(m,n) \tag{6.12}$$

where g, h and f are the blurred image, the point-spread function (PSF) of the blurring system and the original image, respectively. The aim of image restoration is to bring the image back to what it would have been without degradation, i.e. to recover $f(m,n)$ from the information in $g(m,n)$.

In order to extract the original image $f(m,n)$ from the output image $g(m,n)$, a deconvolution is required. In the frequency domain, this can be expressed as an inverse filter

$$G(u,v) = H(u,v)F(u,v) \tag{6.13}$$

The inverse or restoring filter, $Q(u,v)$, may be easily found from $H(u,v)$:

$$Q(u,v) = \frac{1}{H(u,v)} \tag{6.14}$$

However, it is very hard to determine and implement the inverse filter. For this reason an alternative to finding an inverse operator for image restoration is of interest [74].

Several researchers have considered the role of polynomials in deblurring images. One possible way of approaching the problem was considered by Hummel *et al.* [34, 57]. It was noted that the process of reversing blur is unstable and cannot in general be represented as a convolution filter in the spatial domain. They posed the deblurring problem in a variational framework, and constrained the spaces of the original and blurred signals, in an attempt to convert it to a *well-conditioned*

problem. They demonstrated that it is possible to invert the Gaussian blur in a stable manner, if the class of input functions is restricted to polynomials of fixed finite degree. They assume a convolution form for the inverse operator so that the inversion may be realized as a filter operation. The aim was to find an inverse filter which when convolved with the blurred function restores it to the original. The inverse (or deblur) filter $D_N(x)$, which involves a polynomial of degree N, restores the blurred function

$$f(x) = g(x) * D_N(x) \tag{6.15}$$

Martens [57] presented an algorithm for deblurring and interpolating digital images. He assumed that a signal can be locally described by polynomial coefficients, and that these coefficients can be estimated from the sampled signal $S(kT)$ by means of digital filters H_n, for $n = 0, \ldots, N$. These estimated coefficients can then be used to make a deblurred estimate of the original signal. The resulting signal estimate is given by

$$L(x) = \sum_j S(jT) \sum_{n=0}^{N} I_n(x - jT) \tag{6.16}$$

where

$$I_n(x) = \sum_k H_n(k) P_n(x - kT) \tag{6.17}$$

is the n-th order deblurring filter, and P_n are pattern functions. The overall deblur-

ring function

$$I(x) = \sum_{n=0}^{N} I_n(x) \tag{6.18}$$

can be controlled by the order N.

2 Method

The fundamental idea in this work is that of a Hermite polynomial-based PPLN as a filter. In developing a solution, the flexibility of the PPLN should lead to a method which is robust in the sense that it may be applied in situations where little or no apriori information about the degradation is available. For the network to undo the degradation under this condition, however, it is necessary to know something about the image. Therefore, we train the network to develop a deblurring filter on a small region of a blurred image. Provided the blur is shift invariant, then the same inverse filter may be applied to the entire image, as well as different images [41].

This application of PPLN to image deblurring is an example of regression performance in high-dimensional spaces. In the simulations we considered gray-scale images sized 512 x 512 pixels. An image is convolved with the Gaussian filter of spread σ, and from the resulting blurred image we select a 128 x 128 subimage to be used for training. During training, each pixel and its eight surrounding neighbors in a 3 x 3 region of the subimage are presented as a 9 x 1 input vector to the PPLN. The

appropriate output value for each input vector is obtained from the intensity value in the original image corresponding to the location of the center pixel in the 3 x 3 region. For uniformity, all intensity values were normalized. We then applied the PPLN regression model obtained at convergence to full images, each with a different level of blurring.

As an example, we again consider the Lenna image Fig. 6.3. The 9-D input training vector was created from the boxed portion of the blurred image in Fig. 6.4. This image was blurred using a Gaussian function of scale $\sigma = 4$. The 1-D output training vector was created from the original image. During training, the PPLN can be regarded as a filter whose parameters are adapted to find a relationship between the input and output vectors. During testing, the new image is the result of filtering the blurred image. We demonstrate two cases: one in which the test image is less blurred than the training data, and the other in which it is more blurred. Figs. 6.5a and 6.6a show the Lenna image blurred at scales $\sigma = 3$ and $\sigma = 5$, respectively. The restored images after filtering with the PPLN are shown in Figs. 6.5b and 6.6b.

The same trained (converged) PPLN can also be used to deblur other gray-scale images without knowledge of the data in those images. For example, consider the original Peppers image in Fig. 6.7a. Fig. 6.7b shows the blurred image ($\sigma = 4$) and Fig. 6.7c shows the restored image after deblurring.

Figure 6.3: The original Lenna image

Figure 6.4: Blurred Lenna image, $\sigma = 4$. The boxed subimage is used for training.

(a)

(b)

Figure 6.5: (a) Blurred Lenna image for testing, $\sigma = 3$ (b) Restored image.

(a)

(b)

Figure 6.6: (a) Blurred Lenna image for testing, $\sigma = 5$ (b) Restored image.

(a)

(b)

(c)

Figure 6.7: (a) Original Peppers image (b) Blurred Peppers image for testing, $\sigma = 4$ (c) Restored image.

6.5 Summary

We have examined the use of PPLNs in boundary detection. The experimental results indicate that PPLNs perform well, to some extent, at producing more continuous boundaries when presented with disconnected edge points and segments. However, the network also causes thickening of the edges, but this may be compensated for with an effective edge thinning algorithm.

We have also proposed a new approach to the restoration of blurred images by using a PPLN as a deconvolution filter. This method treats deblurring as a high-dimensional regression problem. The proposed procedure seems to work reasonably well even in cases when the amount of blurring is not known *apriori*.

Chapter 7

Conclusions

7.1 Future Work

We propose to investigate the following problems in the next phase of this research:

A. A neural network formulation of the LWF so that weights for best performance can be learnt from the data.

B. A neural network formulation to learn Hermite coefficients for image compression.

C. An automatic selection technique to determine the best order for the Hermite polynomial term in the PPLN.

1 A neural network implementation of LWF

The proposed LWF is a weighted combination of 0th and 2nd order Hermite functions. The problem is to find the correct combination of the weights, c_0 and c_2, so that the best result is obtained. Analysis of these coefficients have shown that c_0 is the major agent in suppressing noise whereas c_2 is responsible for maximizing the edge magnitude. Through trial and error it has been determined that the best results are obtained when c_0 is close to 0.1 and c_2 is close to -0.05. The development of a systematic approach to optimize these coefficients will be studied.

We plan to explore the following approach. The edge detection problem can be thought of as a **decision-making** problem (i.e., Is there an edge at the spatial location (x, y)?). The proposed network will have three layers:

Input Layer: The input vector size will be decided by the size of the processing window (e.g., 3 by 3 window will lead to 9 by 1 input vector etc.).

Hidden Layer: We will take the approach of pruning the number of neurons in this layer with initial number of neurons equal to the number of neurons in the input layer.

Output Layer: This layer contains just one neuron to yield the decision (**yes/no**).

We also note that:

- The weight vector/matrix needs to be related to the coefficients c_0 and c_2.
- The structure (i.e., the nature of the weight elements: inhibitory/excitory) of this network is partially known.
- The design of the network for this section and the following section are interdependent.

2 Image compression with Hermite functions

Our objective is to extend the LWF concepts to describe a new image transform [9] and to implement it using neural network. The image transform introduced in [9] decomposes an image using Gaussian derivatives and results in a spatial/spatial-frequency domain. From the image processing perspective, the locality of the Gaussian derivatives can be a significant benefit. This allows the processing of an image based on local image properties rather than on the global makeup. In image compression, for example, it is common to apply square, nonoverlapping blocks in order to parse the image into many smaller local images. Each block is then transformed with a global process (e.g., DCT). Gaussian derivatives are inherently local, therefore, this partitioning of an image is unnecessary. It is reasonable to expect that the local nature of the Gaussian derivatives and their ability to model the early response of the HVS will result in a transform that is well suited for image applications. The strength of this approach is that the proposed HVS model will produce compressed images that look *better* to human eye.

The one-dimensional discrete signal $f(k)$ can be decomposed into a weighted sum of elementary functions $g_n(k)$, regardless of the orthogonality of g_n:

$$f(k) = \sum_{n=0}^{N-1} c(n) g_n(k) \quad \text{where} \quad 0 \leq k \leq N-1.$$

The expression can be written in matrix notation as

$$\mathbf{f} = \mathbf{G}\mathbf{c}$$

We can define the error as

$$E = \mathbf{G}\mathbf{c} - \mathbf{f}$$

The goal is to find $\hat{\mathbf{c}}$ which minimizes E in some sense. We can easily extend the one-dimensional approach to two-dimensions by considering separable two-dimensional functions. Our objective is to develop a neural network to transform a given image using a set of Hermite functions g_n. Decomposition and reconstruction of an image are not possible using standard inner product techniques since the basis set is not orthogonal. Bloom and Reed [9] use LMS linear systems approach. Our motivation comes from a related work by [15] where, a neural network has been used to transform image data into generalized non-orthogonal two-dimensional Gabor representation for image compression. The strategy here, is to use c_n as the weights in the proposed neural network and estimate them from the image data such that the error E is minimized. What we hope to show from this neural network implementation using

Hermite functions is that such a network may have neuroscientific relevance.

3 Selection strategy for order of the Hermite polynomial

The orthonormal Hermite polynomial functions were chosen due to their parametric orthonormal property and the ease of calculation of the functional values using the least-squares estimation. The Hermite polynomial approximation of the non-linear activations have also been shown to outperform the supersmoother version of the PPLN [35, 36]. The choice of the order is equivalent to choosing a global bandwidth parameter for smoothing. To get the equivalent of supersmoothers in parametric regression, several automatic variable selection methods exist, e.g. AIC [1], MDL [64] and PSE [3]. However, no method exists to automatically determine the order of the Hermite polynomial in a data driven manner. The order of the polynomial did not play a significant role in our experiments on image deblurring, however, in boundary detection we had to use a trial-and-error approach to determine which order produced the best results.

Lay *et al.* investigated the cascaded projection pursuit network (CPPN), which incorporates cascaded connections to a PPLN to further enable its high order feature approximation capability without using high order Hermite polynomial smoothers. Although the proposed CPPN helped significantly in detecting high order features without the critical requirement of adequate pre-selection of polynomial order, there

is an obvious degradation in regression performance when compared with an adequately pre-selected PPLN.

The future work will investigate several automatic variable selection methods to determine how they may be modified and applied to Hermite polynomials without compromising the performance of the PPLN.

7.2 Conclusion

In this thesis, we presented new techniques for image enhancement. We reviewed the features of the Gaussian filter and its role in edge detection, as well as the problems encountered when it is used in edge detection. We then proposed a novel edge operator based on the human visual system, which is a combination of a Gaussian function and its second derivative. Initial experimental results with one- and two-dimensional data indicate that this operator, the LWF, provides excellent edge enhancement and edge localization, with significant removal of noise from the edge description. We then did further analytical studies which proved that the LWF had good edge location accuracy in both low and high noise conditions, and that it also does not detect phantom edges in scale space.

The next phase of the work in this thesis involved the use of projection pursuit regression, a well-known technique in statistical data analysis, but seldomly used in

image processing until recently. We applied the network formulation of this technique, the PPLN, to boundary detection and Gaussian deblurring of images. Experimental results show that the PPLN does produce more continuous boundaries when presented with broken edge segments, but also causes thickening of the lines in the edge image. The PPLN is also effective at restoring blurred images, even when the amount of blurring is not known *apriori*.

7.3 Publications

Journal papers

1. L. M. Kennedy and M. Basu, "Image enhancement using a human visual system model," *Pattern Recognition*, vol. 30, no. 12, pp. 2001-2014, 1997.

2. L. M. Kennedy and M. Basu, "A Gaussian derivative operator for authentic edge detection and accurate edge localization," accepted for publication in *Intl. J. Pattern Recognition and Artificial Intelligence*

3. L. M. Kennedy and M. Basu, " Gaussian-based edge detection methods," submitted to *Pattern Recognition*.

4. L. M. Kennedy and M. Basu, "Application of projection pursuit learning to boundary detection and deblurring in images," to be submitted to *IEEE Trans. Image Processing*.

Conference papers

1. M. Basu and L. M. Kennedy, "An experiment with Gaussian derivatives for Image Enhancement." *Proc. IEEE Int. Conf. Systems, Man and Cybernetics.* pp. 3778-3783. 1995.

2. L. M. Kennedy and M. Basu, "Scale space contours and localization property of a Gaussian derivative edge enhancement operator." *Proc. IEEE Intl. Conf. Systems, Man and Cybernetics.* vol. 1. pp.643-648. 1997.

3. L. M. Kennedy and M. Basu. "Experiments with projection pursuit learning on gray-scale images." to appear in *Proc. Intl. Joint Conf. Information Sciences.* Research Triangle Park. North Carolina. USA. Oct. 23-28. 1998.

Bibliography

[1] H. Akaike. "Information theory and an extension of the maximum likelihood principle," *2nd Intl. Symp. on Information Theory, Akademia Kiado.* Budapest. pp.267-281. 1973.

[2] J. Babaud, A. P. Witkin, M. Baudin and R. O. Duda. "Uniqueness of the Gaussian kernel for scale-space filtering," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8. no. 1. pp. 26-33. 1986.

[3] A. R. Barron. "Predicted squared error: A criterion for automatic model selection." *Self-Organizing Methods in Modeling*, S. Farlow and Marcel Dekker. Eds. chap. 4. 1984.

[4] A. R. Barron and R. L. Barron."Statistical learning networks: A unifying view." in *Proccedings 20th Symposium Interface*. Ed Wegman. Ed. Washington. D.C.: American Statist. Assoc., 1988. pp. 192-203.

[5] M. Basu and L. M. Kennedy. "An experiment with Gaussian Derivatives for Image Enhancement." *Proc. IEEE Int. Conf. SMC.* pp. 3778-3783. 1995.

[6] M. Basu. "A gaussian derivative model for edge enhancement." *Pattern Recognition.* vol. 27. pp. 1451-1461. 1994.

[7] M. Bennamoun. B. Boashash and J. Koo. "Optimal parameters for edge detection." *Proc. IEEE Intl. Conf. SMC.* vol. 2. pp. 1482-1488. 1995.

[8] F. Bergholm, "Edge focusing." *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-9. no. 6. pp.726-741. 1987.

[9] J. A. Bloom and T. R. Reed. "A Gaussian derivative-based transform." *IEEE Trans. IP.* vol. 5. no. 3. 1996.

[10] J. Canny. "A computational approach to edge detection." *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8. no.6. pp. 679-698. 1986.

[11] H. Chen. "Estimation of a projection-pursuit type regression model." *Annals Statis.*, vol. 17, no. 2, pp.453-555, 1989.

[12] J. J. Clark, "Singularity theory and phantom edges in scale space," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-10, no. 5, pp. 720-727, 1988.

[13] J. J. Clark, "Authenticating edges produced by zero-crossing algorithms," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-11, no. 1, pp. 43-57, 1989.

[14] J. G. Daugman, "Two-dimensional spectral analysis of cortical receptive field profiles, " *Vision Research*, Vol. 20, pp. 847-856, 1980.

[15] J. G. Daugman, "Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression," *IEEE Trans. ASSP*, vol. 36, no. 7, pp. 1169-1179, 1988.

[16] G. Deng and L. W. Cahill "An adaptive Gaussian filter for noise reduction and edge detection," *IEEE Nuc. Sci. Symp. Med. Im. Conf.*, pt. 3, pp. 1615-1619, 1994.

[17] P. Diaconis and M. Shahshahani, "On nonlinear functions of linear combinations," *SIAM J. Sci. Statist. Comput.*, vol. 5, no. 1, pp. 175-191, 1984.

[18] D. Donoho and I. M. Johnstone, "Discussion on projection pursuit," *Annals Statist.*, vol. 13, pp. 496-500, 1985.

[19] D. L. Donoho and I. Johnstone, "Projection-based approximation and a duality with kernel methods," *Annals Statist.*, vol. 17, no. 1, pp. 58-106, 1989.

[20] A. Fiorentini and L. Mazzantini, "Neuron inhibition in the human fovea: a study of interaction between two line stimuli," *Atti Fond G Ronchi*, vol. 21, pp. 738-747, 1966.

[21] T. E. Flick, L. K. Jones, R. G. Priest and C. Herman, "Pattern classification using projection pursuit," *Pattern Recognition*, vol.23, no. 12, pp. 1367-1376, 1990.

[22] D. Forsyth and A. Zisserman, "Mutual illumination," *Proc. CVPR*, pp. 466-473, 1989.

[23] J. H. Friedman and W. Stuetzle, "Projection pursuit regression," *J. Amer. Statist. Assoc.* vol. 76, no. 376, pp. 817-823, 1981.

[24] J. H. Friedman, "Classification and multiple response regression through projection pursuit," Dep. Statist., Stanford Univ., Rep. LCM006, 1984.

[25] J. H. Friedman, W. Stuetzle and A. Schroeder, "Projection pursuit density estimation," *J. Amer. Statist. Assoc.*, vol. 79, pp. 599-608, 1984.

[26] J. H. Friedman and J. W. Tukey, "A projection pursuit algorithm for exploratory data analysis," *IEEE Trans. Comput.*, vol. C-23, pp.881-889, 1974.

[27] D. M. Glover and P. K. Hopke, "Exploration of multivariate atmospheric particulate compositional data by projection pursuit," *Atmosph. Environ.*, vol. 28, no. 8, pp.1411-1424, 1994.

[28] N.Graham, J. G. Robson, "Grating simulation in fovea and periphery," *Vision Research*, vol. 18, pp. 815-825, 1978.

[29] P. Hall, "On polynomial-based projection indices for exploratory projection pursuit," *Annals Statist.*, vol. 17, no. 2, pp.589-605, 1989.

[30] P. G. Hall, "On projection pursuit regression," *Annals Statist.*, vol. 17, no. 2, 1989.

[31] B. K. P. Horn, "Image intensity understanding," *Artificial Intelligence*, vol. 8, pp. 201-231, 1977.

[32] P. J. Huber, "Projection pursuit," *Annals Statist.*, vol. 13, no. 2, pp.435-525, 1985.

[33] M. M. Van Hulle."Nonparametric regression analysis achieved with topographic maps developed in combination with projection pursuit learning," *IEEE Trans. Signal Processing*, vol. 45, no. 11, pp.2663-2672, 1997.

[34] R. A. Hummel, B. B. Kimia and S. W. Zucker, "Deblurring the Gaussian blur," *Comput. Vision Graphics Image Process.*, vol. 38, pp.66-80, 1987.

[35] J. N. Hwang, S. R. Lay, M. Maechler, R. D. Martin and J. Schimert, "Regression modeling in backpropagation and projection pursuit learning," *IEEE Trans. Neural Networks*, vol. 5, no. 3, pp. 342-353, 1994.

[36] J. N. Hwang, S. S. You, S. R. Lay and I. C. Jou, "The cascade-correlation learning: a projection pursuit learning perspective," *IEEE Trans. Neural Networks*, vol. 7, no. 2, pp. 278-289, 1996.

[37] N. Intrator, "Exploratory feature extraction in speech signals," in *Advances in Neural Information Processing Systems 3*, R. P. Lippman, J. E. Moody and D. S. Touretzky, Eds. San Mateo, CA: Morgan Kauffman, 1991.

[38] N. Intrator, "Feature extraction using an unsupervised neural network," *Neural Computa.*, vol. 4, pp.98-107, 1992.

[39] H. Jeong and C. I. Kim, "Adaptive determination of filter scales for edge detection," *IEEE Trans. PAMI*, vol. 14, no. 5, 1992.

[40] L. Jones, "On a conjecture of Huber concerning the convergence of projection pursuit regression," *Annals Statist.*, vol. 15, pp. 880-882, 1987.

[41] C. M. Jubien and M. E. Jernigan, "A neural network for deblurring an image," *IEEE Pacific Rim Conf. on Comm., Comput. and Signal Proc.*, pp.457-460, 1991.

[42] L. M. Kennedy and M. Basu,"Scale space contours and localization property of a Gaussian derivative edge enhancement operator," *Proc. IEEE Intl. Conf. Sys. Man Cyb.*, vol. 1, pp.643-648, 1997.

[43] L. M. Kennedy and M. Basu, "Image enhancement using a human visual system model," *Pattern Recognition*, vol. 30, no. 12, pp. 2001-2014, 1997.

[44] J. Koplowitz, V. Greco, "On the edge location error for local maximum and zero-crossing edge detectors," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, no. 12, pp. 1207-1212, 1994.

[45] J. B. Kruskal, "Toward a practical method which helps uncover the structure of a set of multivariate observations by finding the linear transformation which optimizes a new index of condensation," in *Statistical Computation*, R. C. Milton and J. A. Nelder, Eds. New York: Academic, 1969.

[46] J. B. Kruskal, "Linear transformation of multivariate data to reveal clustering," in *Multidimensional Scaling: Theory and Applications in the Behavorial Sciences I-Theory*. New York: Seminar, 1972.

[47] T. Y. Kwok and D. Y. Yeung, "Use of bias term in projection pursuit learning improves approximation and convergence properties," *IEEE Trans. Neural Networks*, vol. 7, no. 5, pp. 1168-1183, 1996.

[48] V. Lacroix, "The primary raster: a multiresolution image description," *Proc. 10th Int. Conf. on Pattern Recognition*, pp. 903-907, 1990.

[49] S. R. Lay, J. N. Hwang and S. S. You, "Extensions to projection pursuit learning networks with parametric smoothers regression," *Proc. Intl. Conf. Neural Networks*, pp.1325-1330, 1994.

[50] Z. Q. Liu, "Scale space approach to directional analysis of images," *Applied Optics*, vol. 30, no. 11, pp. 1369-1373, 1991.

[51] E. Mach, "On the influence of spatially and temporally varying light stimuli on visual perception," *Mach Bounds: Quantitative Studies on Neural Networks in the Retina*, F. Ratlif (Ed.), pp. 321-332, Holden-Day, San Francisco, 1965.

[52] M. Maechler, D. Martin, J. Schimert, M. Csoppenszky, and J. N Hwang, "Projection pursuit learning networks for regression," in *Proc. 2nd Int. IEEE Conf. Tools Artificial Intell.*, Nov. 1990, pp. 350-358.

[53] J. Malik and P. Perona, "Finding boundaries in images," *Neural Networks for Perception*, vol. 1, H. Wechsler (Ed.), pp. 315-344, 1992.

[54] D. Marr and E. Hildreth, "Theory of edge detection," *Proc. R. Soc. Lond.*, vol. B 207, pp.187-217, 1980.

[55] J. B. Martens, "The Hermite transform - theory," *IEEE Trans. ASSP*, vol. 38, pp. 1595-1606, 1990.

[56] J. B. Martens, "The Hermite transform - applications," *IEEE Trans. ASSP*, vol. 38, pp. 1607-1618, 1990.

[57] J. B. Martens, "Deblurring digital images by means of polynomial transforms," *CVGIP*, vol. 50, pp.157-176, 1990.

[58] A. P. Pentland, "Shading into texture," *Artificial Intelligence*, vol. 29, pp. 147-170, 1986.

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

[60] R. J. Qian and T. S. Huang, "Optimal edge detection in two-dimensional images," *IUW*, pp. 1581-1588, 1994.

[61] R. J. Qian and T. S. Huang, "Optimal edge detection in two-dimensional images," *IEEE Trans. Image Processing*, vol. 5, no. 7, pp. 1215-1220, 1996.

[62] H. R. Rabiee, R. L. Kashyap and S. R. Safavian, "Multiresolution segmentation-based image coding with hierarchical data structures," *Proc. ICASSP*, vol. 4, pp.1870-1873, 1996.

[63] J. Richter and S. Ullman, "Non-linearities in cortical simple cells and the possible detection of zero crossing," *Biological Cybernetics*, Vol. 53, pp. 195-202, 1986.

[64] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Ann. of Stat.*, vol. 11, no. 2, pp.416-431, 1983.

[65] A. Rosenfeld and A. C. Kak, *Digital Picture Processing* (2nd ed.), Academic Press, New York, 1982.

[66] S. R. Safavian, H. R. Rabiee, M. Fardanesh and R. L. Kashyap, "Low bit rate image compression with orthogonal projection pursuit neural networks," *Proc. IEEE Intl. Conf. Neural Networks*, vol. 3, pp.1518-1522, 1997.

[67] S. R. Safavian, H. R. Rabiee and M. Fardanesh, "Projection pursuit image compression with variable block size segmentation." *IEEE Signal Processing Letters*, vol. 4, no. 5, pp.117-120, 1997.

[68] B. G. Schunk, "Edge detection with Gaussian filters at multiple scales." *Proc. IEEE Comp. Soc. Work. Comp. Vis.*, pp. 208-210, 1987.

[69] J. Shen and S. Castan, "Towards the unification of band-limited derivative operators for edge detection." *Signal Processing*, vol. 31, pp. 103-119, 1993.

[70] Statistical Science Inc., *S-Plus Guide to Statistical and Mathematical Analysis*. (Version 3.2). Seattle, WA.

[71] A. L. Stewart and R. Pinkham, "A spatial-variant differential operator for visual sensitivity." *Biological Cybernetics*, vol. 64, pp. 373-379, 1991.

[72] P. Switzer, "Numerical classification." in *Geostatistics*. New York: Plenum. 1970.

[73] P. Switzer and R. M. Wright, "Numerical classification applied to certain Jamaican eocene nummulitids." Math. Geology, vol. 3, pp 297-311, 1971.

[74] H. Tang and L. W. Cahill, "A new approach for the restoration of noisy blurred images." *Proc. IEEE Intl. Symp. Circuits and Systems*, vol. 1, pp.520-523, 1991.

[75] M. Vairy and Y. V. Venkatesh, "Deblurring Gaussian blur using a wavelet array transform." *Pattern Recognition*, vol. 28, no. 7, pp.965-976, 1995.

[76] D. J. Williams and M. Shah, "Edge contours using multiple scales." *Comp. Vis. Grap. Image Proc.*, vol. 51, pp. 256-274, 1990.

[77] G. J. Yang and T. S. Huang, "The effect of median filtering on edge location estimation." *Computer Graphics and Image Processing*, vol. 15, pp.224-254, 1981.

[78] R. A. Young, "The Gaussian derivative model for spatial vision: I. Retinal Mechanisms," *Spatial Vision*, vol. 2, no. 4, pp. 273-293, 1987.

[79] R. A. Young, "Oh say, can you see? The physiology of vision." *SPIE Vol. 1453. Human Vision, Visual Processing, and Digital Display II*, pp. 92-123, 1991.

[80] A. L. Yuille and T. A. Poggio, "Scaling theorems for zero crossings." *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, no. 1, pp. 15-25, 1986.

[81] Y. Zhao and C. G. Atkeson, "Implementing projection pursuit learning." *IEEE Trans. Neural Networks*, vol. 7, no. 2, pp. 362-373, 1996.

[82] S. W. Zucker and R. A. Hummel, "Receptive fields and the representation of visual information," *Proc. 7th. Intl. Conf. Pat. Recog.*, Montreal, Canada, pp. 515-517, 1984.