THE ANALYSIS OF IMAGE SEQUENCE DATA WITH APPLICATIONS TO TWO-DIMENSIONAL ECHOCARDIOGRAPHY

A Thesis

Submitted to the Faculty

of

Purdue University

by

Chee-Hung Henry Chu

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 1988

This thesis is dedicated to the memory of my grandmother,
who expected so much and yet asked so little from me.

ACKNOWLEDGMENTS

I am deeply grateful to many of my friends who have made invaluable contributions to the writing of this thesis and to my education in general. It would not be possible to include them all here; in particular, I thank:

Professor Edward Delp whose patience, support, guidance, and encouragement I shall always treasure;

Professors Leah Jamieson, Michael D. Zoltowski, John S. Sadowsky, H.J. Siegel, Jan P. Allebach, O. Robert Mitchell, and Lawrence G. Brown for their advice and help at various stages of my graduate career;

Nirwan Ansari for teaching me so much and for putting up with me for the past few years;

Mr. and Mrs. Dat Yun Chu for their kindness, generosity and hospitality extended to me since my freshman year in college.

Most of all, I thank my parents for their love and support, and my brother for the constant supply of coffee beans, among other things equally precious.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

ABSTRACT

Chu, Chee-Hung Henry. Ph.D., Purdue University, August 1988. The analysis of image sequence data with applications to two-dimensional echocardiography. Major Professor: Edward John Delp III.

Digital two-dimensional echocardiography is an ultrasonic imaging technique that is used as an increasingly important noninvasive technique in the comprehensive characterization of the left ventricular structure and function. Quantitative analysis often uses heart wall motion and other shape attributes such as the heart wall thickness, heart chamber area, and the variation of these attributes throughout the cardiac cycle. These analyses require the complete determination of the heart wall boundaries. Poor image quality and large amount of noise makes computer detection of the boundaries difficult.

An algorithm to detect both the inner and outer heart wall boundaries is presented. The algorithm was applied to images acquired from animal studies and from a tissue equivalent phantom to verify the performance. Different approaches to exploiting the temporal redundancy of the image data without making use of results from image segmentation and scene interpretation are explored. A new approach to perform image flow analysis is developed based on the Total Least Squares method. The result of this processing is an estimate of the velocities in the image plane. In an image understanding system, information acquired from related domains by other sensors are often useful to the analysis of images. Electrocardiogram signals measure the change of electrical potential changes in the heart muscle and provide important information

such as the timing data for image sequence analysis. These signals are frequently plagued by impulsive muscle noise and background drift due to patient movement. A new approach to solving these problems is presented using mathematical morphology. Experiments addressing various aspects of the problem, such as algorithm performance, choice of operator parameters, and response to sinusoidal inputs, are reported.

CHAPTER 1
INTRODUCTION

1.1. Digital Two-Dimensional Echocardiography

Digital two-dimensional echocardiography is used as an increasingly important noninvasive technique in the comprehensive characterization of the left ventricular structure and function. Two-dimensional echocardiograms are ultrasonic images depicting cross-sectional views of the heart. The cross-sections are taken either longitudinally, commonly referred to as the *long-axis view*, or latitudinally, commonly referred to as the *short-axis view*. The short-axis view is frequently used to monitor the left ventricle, which is the main pumping chamber of the heart [Bri83]. The wall of the left ventricle in a short-axis view is defined by an inner *endocardial* boundary and an outer *epicardial* boundary. Under normal conditions of the heart, the left ventricular wall contracts and relaxes uniformly from diastole to systole back to diastole. At end systole, or full contraction, up to 80% of the inner chamber area as measured at end diastole is ejected. The motion is such that the inner boundary contracts more than the outer boundary, resulting in a thickening of the ventricular wall as it contracts. When a heart is diseased, such as after myocardial infarction, the left ventricle demonstrates irregular motion due to parts of the heart wall moving much less than other parts. The significance of this irregular heart wall motion and decreased ejected volume or area is that there is insufficient blood supplied to the circulation system. To consider the effects of therapy or drug treatment, for example, monitoring of the left ventricle throughout the cardiac cycle. is required.

Quantitative analysis of cardiac function makes use of the shapes of the boundaries, the heart wall thickness, the area enclosed by the inner boundary, and monitors their changes throughout the cardiac cycle [Eat79]. These and other applications such as three-dimensional organ modeling of the left ventricle [Gei82] require the detection of the inner and outer boundaries of the left ventricular wall from two-dimensional echocardiograms.

Current studies in this area often require the tedious and time-consuming process of having expert operators outline the boundaries. Frequently, only the end-systolic and end-diastolic images are processed by human operators for analysis. This is unsatisfactory since different parts of the ventricle have slightly different peak contraction points in the cardiac cycle, and the differences are even more pronounced when regional ischemia occurs [Col86]. The problem associated with the arbitrary selection of an end-systole frame, together with the need for a description of the systolic and diastolic wall motion pattern, require that every frame of the cardiac cycle be analyzed. Other applications of this labor intensive process to trace the heart wall boundaries include surface reconstruction in organ modeling when a number of images taken from different angles have to be processed to obtain points in three-dimensional space. Automatic determination of the boundaries by computers is thus needed. Furthermore, automating the process would improve the reliability of the quantitative analysis by eliminating the subjectivity of manual tracing [Bud83].

Finding boundaries in echocardiograms automatically by computers is often difficult because of the poor quality of the images. None of the attempts at automating the image segmentation process is reliable enough to replace the human operator completely [Bri83]. Problems such as low image intensity contrast, dropouts in the image, and boundary discontinuity in any given image are due to the intrinsic limitations of echocardiographic imaging. An echocardiogram is formed by first sending a pulse along a ray from a transducer towards the organ that is being imaged. Compared to other imaging techniques, the pulse used in echocardiography is of relatively low energy and low frequency. When the pulse hits a medium with an acoustic impedance different from that of the medium in which it is traveling, a copy of the pulse with reduced energy is reflected while the remaining portion travels on. The amount of energy that is reflected back is a measure of the difference of the acoustic impedance across the boundary. The transducer acts as a receiver after transmitting the pulse and measures the time it takes the pulse in transit to compute the distance of the boundary. The accuracy with which an echocardiographic system can measure the distance traveled by a pulse, or the *resolution* attainable by that particular pulse, increases with the frequency of the pulse. In practice, since the energy of the pulse diminishes as it travels, the postprocessing of the reflected signal includes time gain control that compensates for the attenuation of the signal over time. The amount of energy attenuation decreases with the frequency of the pulse. Hence, a

tradeoff is necessary between the amount of energy loss and the resolution of the image acquisition system [Bri83].

Assuming the pulse travels at a single speed in the body, and by taking different rays across a plane, a two-dimensional record of the received energy in spatial coordinates represents a cross-sectional view of the organ. Dropouts in the echocardiograms are caused by reverberations of the pulse bouncing between the boundaries of the heart wall, and by speckle noise caused by the backscattering of the incident wavefront after it hits the tissue microstructures. Another limitation of this imaging technique is that the reflection is not very pronounced when the angle between the boundary of the organ and the ray that the pulse is traveling along is small. Hence the lateral parts of the heart wall boundaries are usually not very well defined in the images [Bud85].

1.2. Overview of a Computer Vision Approach

A computer vision system extracts information of a scene from observations made in the form of images. The information loss in the imaging process that produces the two-dimensional images has to be compensated for by some other means. This problem is further complicated by the presence of noise or by images that are of low quality. A widely used method is to use domain specific knowledge to constrain the solution in interpreting the images. This is possible in situations where the variety of contents in an image is restricted, and more importantly, where the objects in a scene can be described in a form suitable for computers. An example of such situations is machine vision in industrial settings, where computer-aided design information of machine parts is available and can be readily incorporated in a computer vision system. The description of objects in natural scenes is less well understood, making specifying *a priori* knowledge more difficult.

Other passive methods for disambiguating images are more desirable than those using directly specified knowledge because the passive methods are potentially applicable to a wider range of situations. An almost intuitive approach to resolve ambiguities in the interpretation process is to increase the amount of observed data in the hope of including more information albeit, at the expense of more processing work. More data can be observed by viewing the scene from different viewpoints, referred to as *stereo vision*, or by viewing the scene for a period of time, referred to as *dynamic scene analysis*. The challenge here is to ensure that more information is in fact included in the extra data, and that there is a known way to recover them. As we shall see,

this is by no means trivial. In most situations, the interpretation process cannot be solved by a single method.

Computer vision has been applied to machine parts inspection, remote sensing and photo interpretation. Each application has its own characteristics; in medical image understanding, the "scene" or "world" is restricted to known normal and abnormal anatomy [Bri83]. Foreign objects are not expected to appear in these images very often. On the other hand, as is true of other natural scene understanding problems, natural objects such as human organs and anatomical relations are difficult to describe precisely. Very few human faces are identical, for example, despite having the same components.

Current echocardiography studies often make use of very traditional image processing methods [Gar85], and some highly problem-specific image analysis algorithms that implicitly make use of heart anatomy knowledge. Finding the heart wall boundaries in echocardiographic images, in terms of a computer vision system, is an *object detection* problem, which is one of finding the boundary of a region on an image plane that corresponds to the image of an object. In this section, the detection of heart wall boundaries is cast in terms of a high level computer vision system. A computer vision system for detecting objects typically employs image processing algorithms for extracting information from images, *a priori* knowledge about the problem domain, and a control strategy for higher level analysis to determine the object location. A typical configuration of such a system is shown in Figure 1.1.

Image processing algorithms are mostly numerical operations and their main goal is to transform the image data so that a symbolic representation can be produced for interpretation processes. Algorithms that detect spatial features such as intensity edges, and those that detect temporal events such as image motion, can provide information for the extraction of heart wall boundaries. Attributes of detected features and events are also useful in interpretation processes. The attributes of an edge segment include edge strength, local orientation, length, and variation of local orientation along its length. The detection of edges and image motion in echocardiograms will be covered in the following chapters.

A control strategy manipulates output from the image processing algorithms to determine the boundary location. An example of the operations taken by the control strategy is the classification of each detected image edge segment as either part of the inner heart wall (endocardial boundary), part of

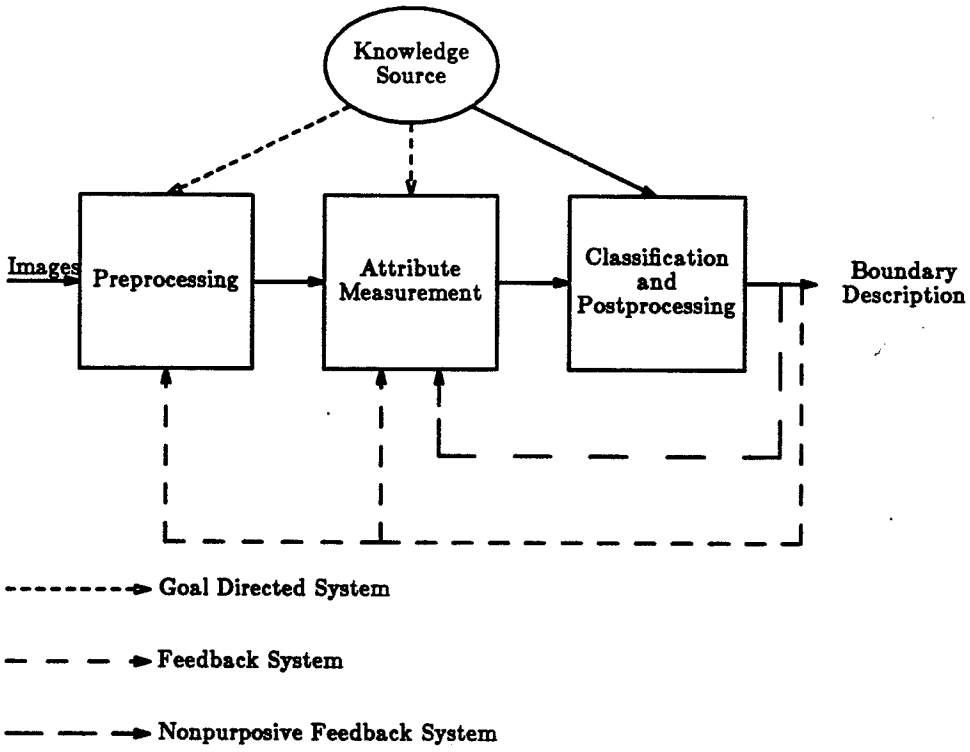


Figure 1.1. Typical configuration of a computer vision system for locating object boundaries.

the papillary muscle, part of the outer heart wall (epicardial boundary), or an artifact due to noise.

High level analysis is used in a computer vision system to integrate multiple sources of information to form an estimate of the heart wall location in the image plane. Sources of information include specific knowledge about the heart anatomy and results obtained from the image data by image processing algorithms. The two most important functions for high level analysis in object detection are classification of detected image features and interpolation to form a complete boundary estimate.

Classification is the process by which detected image features such as edge segments are assigned semantic labels, such as parts of heart wall, papillary muscle, or an noisy artifact. This mapping is useful for recognizing the image contents, and for eliminating noisy artifacts from being included in the interpolation of missing points based on detected data. An individual edge segment is typically labeled based on its measured attributes, specific knowledge about the scene, and, in the case of feedback systems, partial results obtained from a previous classification [Ten77].

The form of knowledge representation plays an important role in determining the choice of algorithms used in higher level analysis. If the object boundary can be represented by exact mathematical forms, curve fitting methods can be used to determine how well an image curve fits the object boundary. While an exact representation of the heart wall boundary is often unavailable, these methods can be used for approximation of the boundary by curves such as circles or ellipses.

Hough transform methods [Bal81] are another class of algorithms that make use of a set of parameters to describe a contour. A parameter space histogram of the observed data is formed, and a peak in the histogram corresponds to the set of parameters that is the best fit for the contour. Disadvantages of these methods are that histogram forming and the subsequent peak finding can get unwieldy when the number of parameters grows, and that the spatial structure of the data is often ignored [Low84]. Again, while an exact representation of the boundary curve may not be available, Hough methods can be used to approximate the boundary curve by ellipses or circles, or by using a boundary found in a previous frame as a template. Those data points that correspond to parameters that deviate substantially from the peak can be treated as noise and rejected.

If the *a priori* knowledge can be represented as rules governing the appearance of an image, a rule-based system can be used for high level analysis. A rule is an *antecedent-consequence* pair [Nil80]: the *antecedent* describes a condition of the observed data, and the *consequence* describes actions that will be taken if the *antecedent* was met. The actions taken can be a decision making or some further processing by image processing algorithms. Rule based analysis has been applied to angiograms [Sta86] and aerial imagery [McK85] for object detection. More recently, there have been attempts at using expert systems concepts for recognizing segmented regions in echocardiographic images [Tuc85]. These attempts have limited success due to the assumption that the images can be segmented successfully.

There are different ways to organize a computer vision system. An obvious approach to search for the heart wall boundaries is to use image processing methods such as edge detectors to detect image edge segments, followed by a classification scheme that makes use of *a priori* knowledge to locate the heart wall boundaries from the detected edge segments. This is commonly referred to as a *data-driven* system. Depending on such factors as the signal-to-noise ratio of the observed data, the degree of precision of the knowledge representation of the object and its background, other choices such as a goal-directed approach may be more suitable [Oht85]. For example, in the presence of a large amount of noise, the control strategy in a data-driven system will have to search for the correct boundary from a large number of candidate segments, most of which are false responses to noise.

A *goal-directed* system uses the problem specific knowledge both for detecting edges, and the subsequent classification of the detected edge segments. A *feedback* system also uses the knowledge of the heart anatomy both for detecting edges and in classifying the detected edge segments. Furthermore, any information gathered after an initial classification is fed back to the edge detection step to improve the detection. Since only edges that are likely to be heart wall boundary segments are detected initially, the control strategy in these systems can be made relatively simpler. The disadvantage is that problem specific edge detectors are usually not stable with respect to noise or imprecise knowledge representation.

A *nonpurposive-segmentation feedback* system uses edge detectors that do not make use of *a priori* knowledge to perform the initial edge detection [Oht85]. High level knowledge is used to further classify the detected edge segments. Information derived from an initial classification is fed back to the classification scheme itself or to the attribute measurement step to improve

the performance.

The problem of detecting heart wall boundaries in the echocardiograms is discussed in Chapter 2. It can be seen that while some success can be expected from processing a single frame by itself, much of the information in the data lie in the temporal redundancy. The use of image sequence analysis techniques is presented in Chapter 3. A new approach based on Total Least Squares method for image flow analysis is presented in Chapter 4. Electrocardiogram signals is often used to provide timing information for echocardiogram data acquisition and for assisting the image sequence analysis. A new approach to perform impulsive noise suppression and background normalization of electrocardiogram signals is presented in Chapter 5. Future work and conclusions are covered in Chapter 6.

CHAPTER 2
DETECTING ENDOCARDIAL AND EPICARDIAL BOUNDARIES

2.1. Introduction

In this chapter, algorithms for detecting heart wall boundaries in echocardiograms are surveyed. Algorithms reported in recent computer vision and image processing literature are surveyed, and their suitability for processing echo images is discussed. A new algorithm developed for detecting heart wall boundaries in a single frame of an echocardiogram sequence is presented.

A typical heart wall boundary detection procedure in echocardiography studies has three steps [Sko85]: (1) preprocessing by smoothing, (2) enhancing the image so that areas with high intensity variations would have higher values whereas areas with more or less constant intensities would have lower values, and (3) identifying the single contour that represents the boundary of interest from the enhanced image. In terms of the model shown in Figure 1.1, steps (1) and (2) are image processing algorithms, while step (3) is the control strategy.

Image processing step is typically performed by using 3×3 operators such as the Sobel or the Laplacian operators. After the image processing step, strategies that have been used for extracting the the heart wall boundary contour include [Sko85]: radial search, binary image forming with subsequent contour tracking, and "prior constraints" of an operator-assigned starting border.

In contour tracking methods, all potential edge points are marked initially; a procedure for tracking the border would then sequentially look for the edge points making up the boundary in the marked pixels. In the "prior constraints" driven methods, the boundary in the initial frame is outlined by an operator. Boundaries in subsequent frames are then found by using the initial boundary and other *a priori* knowledge such as the shape of the ventricle, or the maximum rate of wall motion.

Radial search methods start from a point inside the heart chamber and search radially for the endocardial border. These methods are attractive because the most prominent image feature in the echocardiographic images is often the inner chamber. Due to the physical reason that there is no tissue to reflect the ultrasonic pulse, the image intensity for the heart chamber has a somewhat consistent, typically low, value. The problem of having to pick the contour out from among the many detected edge segments is also avoided.

Other control strategy functions that have been used for postprocessing are mainly used for filling in missing points by interpolating neighbors [Zha84], linking and smoothing detected boundaries [Tam85, Chu86]. A cost function could be used to evaluate every detected boundary point to remove responses due to noise [Eze85].

2.2. Detecting Intensity Edges in Echocardiograms

An important class of image processing algorithms for detecting heart wall boundaries are those that detect edges. Since the heart wall boundaries are imaged as intensity edges, any attempt at detecting these boundaries would necessarily incorporate an edge detection step. Heart wall boundaries are imaged in echocardiograms as intensity edges, which are points where the image intensity changes from one level to another. In the continuous domain, finding edge points is done by locating the signal discontinuities or by differentiating the signal and marking the points where the derivatives have large values. When digital images are processed, these methods have to be approximated by taking finite difference of neighbor values. Finite difference methods usually produce a large number of false responses while missing changes that vary less abruptly.

A typical enhancement/threshold type edge operator enhances the original image at each pixel by using a finite difference method to estimate the gradient magnitude. The finite difference operators are usually implemented as 3×3 or 5×5 masks to produce edge strength values, which are then thresholded to form edge points. An example of this type of edge operator is the Sobel operator, which has been used to process echocardiographic images [Sko81]. The Sobel operator estimates the directional derivative in two orthogonal directions and measures the edge strength by the estimate of the gradient magnitude.

Thresholding gradient magnitude maps results in thick edges, making the location of boundaries ambiguous. Postprocessing such as edge thinning is

needed to produce single pixel wide edges. Nonmaximum suppression of gradient magnitude, or lateral inhibition [Bin81], essentially redefines the edge as the local maximum of the gradient magnitude. Since the local maximum of a function corresponds to a zero-crossing of the derivative of that function, applying a Laplacian operator followed by detecting zero-crossings has been used to detect edges [Eze85].

Since noise is accentuated by taking finite differences, images are often preprocessed by a linear smoothing operation. Torre and Poggio [Tor86] consider numerical differentiation as an ill-posed problem and show that it should be preceded by a filtering step. The combination of smoothing and differencing steps results in larger operator sizes. Shanmugam et al. [Sha79] develop an optimum filter in the sense that it produced maximum energy within a resolution interval of specified width in the vicinity of the edge. Marr and Hildreth [Mar79] use a filter with the shape of a $\nabla^2 G$ function, where ∇^2 is the Laplacian operator and G is a Gaussian function. Edges are found by detecting the zero-crossings of the Laplacian of the image smoothed by a Gaussian shaped filter. Canny's operator [Can86] is designed to maximize the detectability while minimizing the displacement of the detected edge segments. This operator is shown to be well approximated by the derivative of Gaussian operator. Edges are detected by smoothing an image by large Gaussian-shaped masks before detecting zero-crossings of the second derivative along the direction of the gradient of the smoothed image. All of these recently developed operators are much larger than the 3×3 or 5×5 operators commonly employed in echocardiography studies.

The *facet-model* based methods consider image intensities to be noisy observations of a signal that can be expressed as a linear sum of a set of basis functions. The values of the derivatives of the signal are estimated by fitting the derivatives of the basis functions to the observed data [Har84]. A set of cubic polynomials with two variables are chosen as the basis functions and the fitting is done on local 11×11 neighborhoods. An edge point was detected by finding the zero crossing of the second directional derivative along the direction of the gradient. It should be noted that despite their philosophical differences, the facet model based method and the window operator based methods, such as the $\nabla^2 G$ operator or Canny's operator, are equivalent from a data processing point of view.

An important issue when enhancing the images with window operators is to determine the size of the operator to use. Operator size can range from the usual 3×3 to 11×11 [Har85, Har84] or even up to 35×35 [Gri85]. The question

of what can be reliably detected from a noisy image is central to the problem of locating or identifying objects in the image plane. This has a particularly significant impact on the choice of the size of edge detectors. For the purpose of using the edge points to achieve object location later on, only spatially significant features should be detected initially, even if there are relatively few such points. Features that are detected based on information from a large area of image plane or over some significant time period are less likely to have been caused by false responses of the detector to noise. Even though there may be features that are of smaller scale that can be detected after an estimate of the scene has been established, the initial recognition process should certainly not be based on all small scale features that may be found.

Smoothing by small windows tends to result in a lot of false detections due to insufficient noise suppression, while smoothing by large windows tends to result in a lot of true edges being missed. It should be noted that human observers may prefer the results obtained by using small window operators since human vision systems are very proficient at rejecting false detections and deducing structures. Computers have not matched human performance in these two areas due to the sometimes overwhelming amount of data and the difficulty in formulating rules for rejection and deduction.

The window size chosen for preprocessing the echocardiographic images can be much larger than the 3×3 or 5×5 windows commonly reported [Sko85]. For example, the window size is 41×41 for the results reported in Section 2.3 . There might be some concern about using such a large window when the edges are visibly "washed out." A large window should be used precisely because the signal is weak so that more information is available to make a decision. The disadvantage of using a large window is that nearby edges may be merged, thus resulting in displaced detected edges.

The detectors discussed thus far are mainly concerned with devising elaborate schemes to accurately estimate the derivatives of an input image. After the estimates are obtained, the decision of whether an edge is present or not is made based on relatively simple decision strategies such as thresholding the gradient magnitude at local maxima of gradients. The disadvantage of using such a simple strategy is that only local information is used. More sophisticated decision strategies are usually based on employing global information and are thus more desirable in low signal-to-noise situations.

The amount of information used to detect an edge is limited by the size of the operator. As the size of the operator increases, more globally significant

edge segments are detected, at the expense of missing finer details in the image, and possibly at the expense of displacing an edge due to the merging of two nearby edges. Thus, detecting edges at multiple resolutions is desirable for detecting accurate and significant edges. Integrating the results from different resolutions, however, remains difficult. The Marr-Hildreth scheme [Mar79] detected edges at five different levels and combined the results by the heuristic that edges found at two successive levels are marked as true edge points.

An operator with a large spatial support makes no discrimination between information that is relevant to the edge detection process and information that is not. Sequential edge detection methods [Eic85] improves on this by gathering more information in the decision strategy through collecting information along an edge segment instead of around an arbitrary large neighborhood. Since the additional data points taken into account by the decision strategy have more relevant information, the performance of sequential edge detection is improved. Unfortunately, sequential edge detection has not enjoyed as much success in echocardiographic images as compared to other cardiac imaging applications [Eic86] due largely to the signal dropouts in echocardiographic images.

All of the operators discussed thus far detect intensity edges without specific consideration as to whether they are on the heart wall boundary. As noted in Section 1.2, a goal-directed system uses specific problem domain information in the detection of edges. Schudy and Ballard detect boundary points using the *a priori* knowledge of the heart chamber shape [Sch79]. The heart chamber surface is modeled as a linear sum of spherical harmonics, which are generalizations of the Fourier functions to the surface of a sphere. The boundary points are found by fitting the basis function to the image data.

Another example of detecting edges in echocardiograms by using specific problem domain knowledge are those procedures that use a one-dimensional difference operator to detect edge points along each search ray starting from a point inside the heart chamber [Del82, Tam85]. These one-dimensional methods necessarily ignore the information provided by the two-dimensional spatial structure of the images. In the presence of a large amount of noise, it is particularly important that all the information be used for detecting edges or other image features.

2.3. A New Algorithm

A new algorithm for detecting both endocardial and epicardial boundaries in echocardiographic images is now presented. This algorithm serves two purposes: besides extracting high level tokens for interframe matching, we also include a linking and interpolation process to investigate how well single frame processing can perform. The algorithm is based on the nonpurposive segmentation approach. It consists of three steps: (1) edge detecting by a general edge detector, (2) radial search for initial edge estimates, and (3) nonlinearly processing the edge estimates to compensate for dropouts and poor contrast. An overview of the algorithm is shown in Figure 2.1. Partial results at various steps are used to guide further searches for missing points.

Echo data are acquired from animal studies and from a tissue-equivalent phantom with known measurements. Samples are collected in polar coordinates along 98 different rays scanning a horizontal cross-section of the left ventricle of the heart. Each ray consists of 288 samples with 6 bits of gray-level information per sample. The intersample distance is 0.5 mm. Each study consisting of a sequence of 36 images (3 cardiac cycles); an image is interpolated into rectangular coordinates before the portion with size 256×256 that contains the region of interest is extracted for processing.

The enhancement step is performed by applying a window with Gaussian weighting to the images. As noted in Section 2.2, the window size used is 41×41 for the results shown below. After the smoothing operation, the discrete approximation of the Laplacian operator is applied to the entire image. This step, together with the smoothing operation, can be combined as a single $\nabla^2 G$ operation, eliminating the need for a separate approximation of the Laplacian operator [Mar79]. Figure 2.2 shows the sign of the $\nabla^2 G$ operator after it has been applied to an image. Those points where the Laplacian value changes sign from positive to negative are considered to be boundary points. Instead of detecting the edge point based solely on the information along each ray as in [Del82, Tam85], the zero-crossing points found here are based on the information derived from a much larger two-dimensional neighborhood.

After all the edge points are found by the nonpurposive segmentation step, a radial search is used to find the endocardial boundary by locating the zero-crossing points. Since there are not enough intensity variations in some parts of the image, only part of the boundary will be detected initially. This, of course, was in part due to the window size used in the enhancement step.

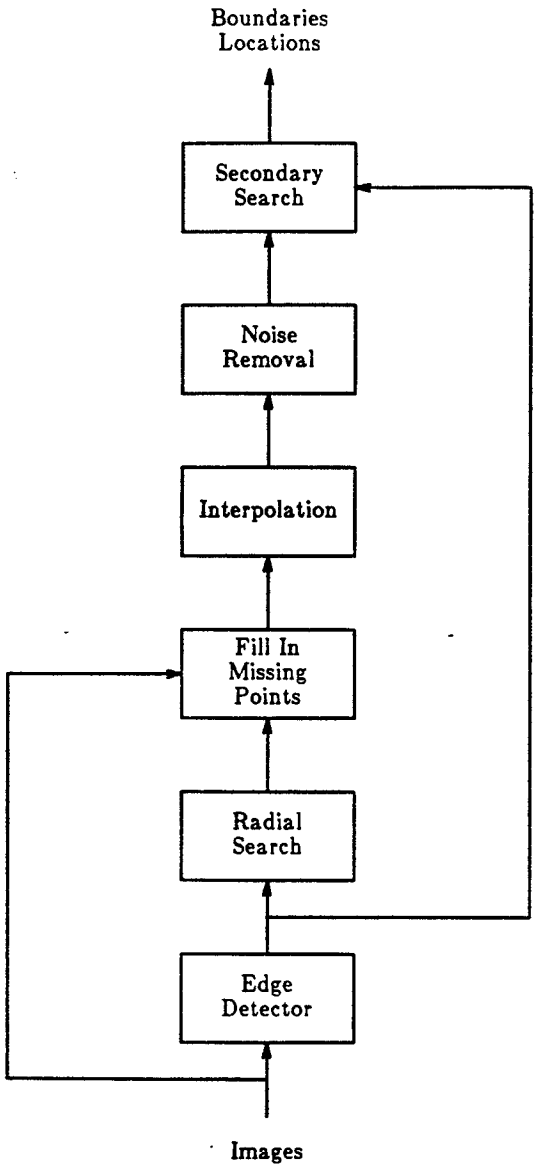


Figure 2.1. Overview of the algorithm for detecting inner and outer heart wall boundaries in echocardiograms.

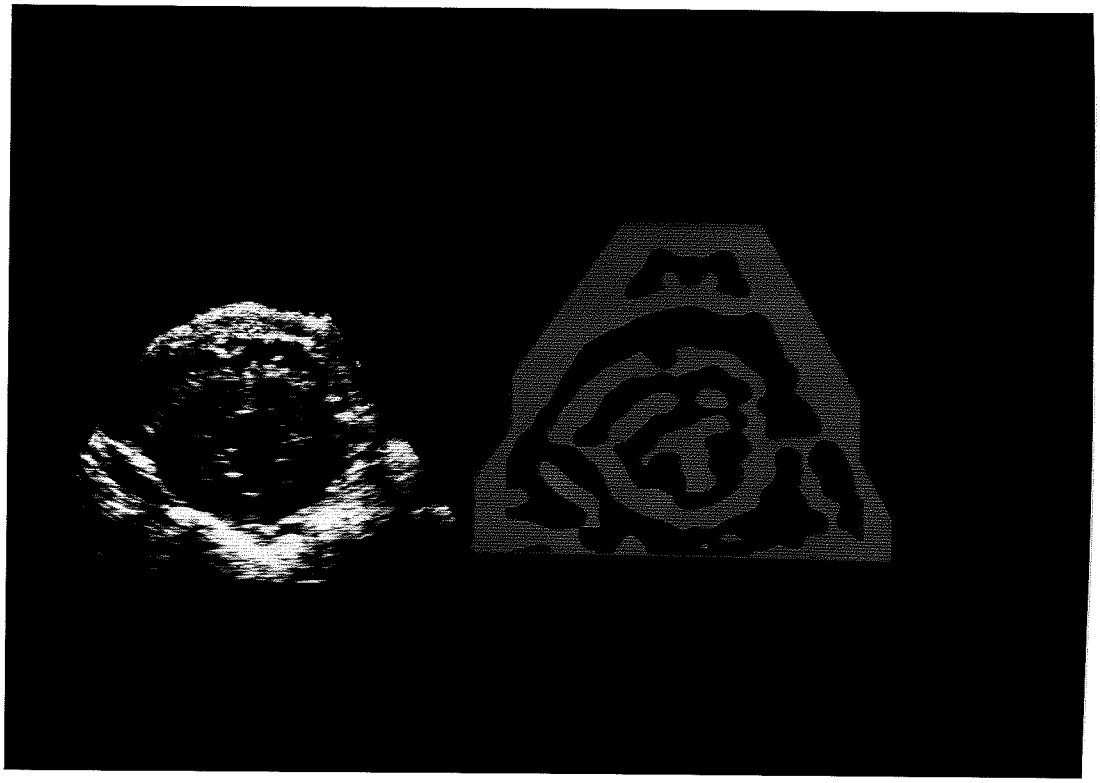


Figure 2.2. Sign map of applying a $\nabla^2 G$ operator. *Left*: Original image. *Right*: Sign of the $\nabla^2 G$ operator after applied to the original image. Bright regions correspond to positive values and dark regions correspond to negative values.

The advantage of using a large window is that the detected segments are very reliable, since they are due to significant gray level contrasts. The search center is defined by a human operator. To avoid detecting the epicardial boundary while looking for the endocardial boundary, a limit is set for the search distance along each ray. A different set of limits is subsequently used for detecting the epicardial boundary using a similar radial search. The search limits are typically set at 20 pixels.

After the radial search, if there are no edge points found for a particular ray, edge points are searched for in the original picture. The search area is defined by the neighbor edge points that are found in the initial radial search. Figures 2.3 to 2.5 show some typical results of the edge points detected using this technique.

After the initial detection of the edge points, further processing had to be done to remove the false edge points and to fill in missing edge regions. The distance of each edge point from the search center is plotted against its angular displacement from a reference axis. This plot is shown in Figures 2.6 and 2.7 for both the inner and outer boundaries found in Figures 2.3 and 2.4, respectively. A one-dimensional median filter is used to remove spurious impulses contained in this distance-angle plot. For the examples shown, the window size is set at 5. Median filtering is known to be effective at removing impulse noise while retaining the original values of the signal when the noise is relatively low in value [Gal81]. To avoid using an overly large window to fill in drop-outs, or missing points, that occur for a large number of successive rays, the missing points are linearly interpolated by the neighboring values before the median filtering operation. This processed distance plot defines a new reconstructed boundary. The processed distance plots are shown in Figures 2.6 and 2.7. Figures 2.3 to 2.5 show the results of the reconstructed boundaries.

Although the radial search method significantly simplifies the detection and classification steps, one major weakness in this method is that part of the boundary can be missed when the search ray is almost parallel to the boundary since it is difficult for the search to look for either gradient or local maximum. Such a situation occurs, e.g., when the curvature of the boundary goes from convex to concave, such as in the region when the papillary muscle protrudes into the heart chamber. In extreme cases, part of the boundary might be occluded by the protrusion (see Figure 2.8).

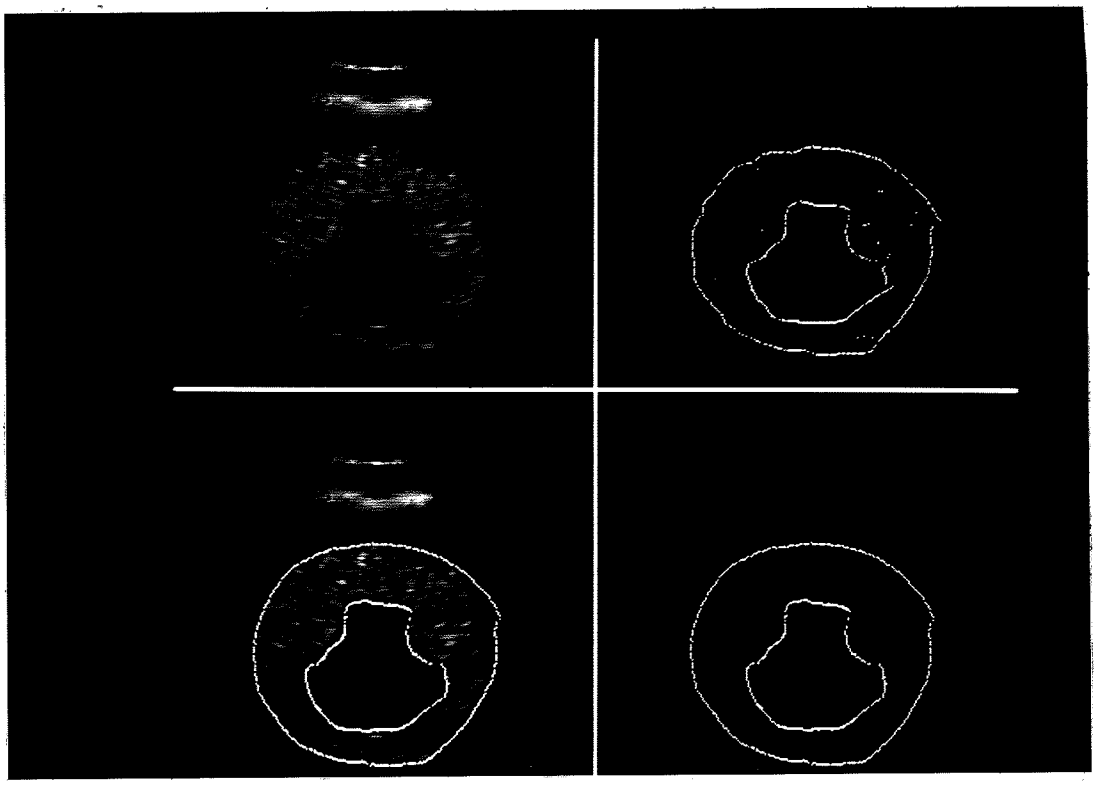


Figure 2.3. Detected boundaries in phantom. *Upper Left:* Original image of phantom. *Upper Right:* Detected raw edge points. *Lower Left:* Edge boundaries after processing. *Lower Right:* Detected boundaries superimposed on the original image.

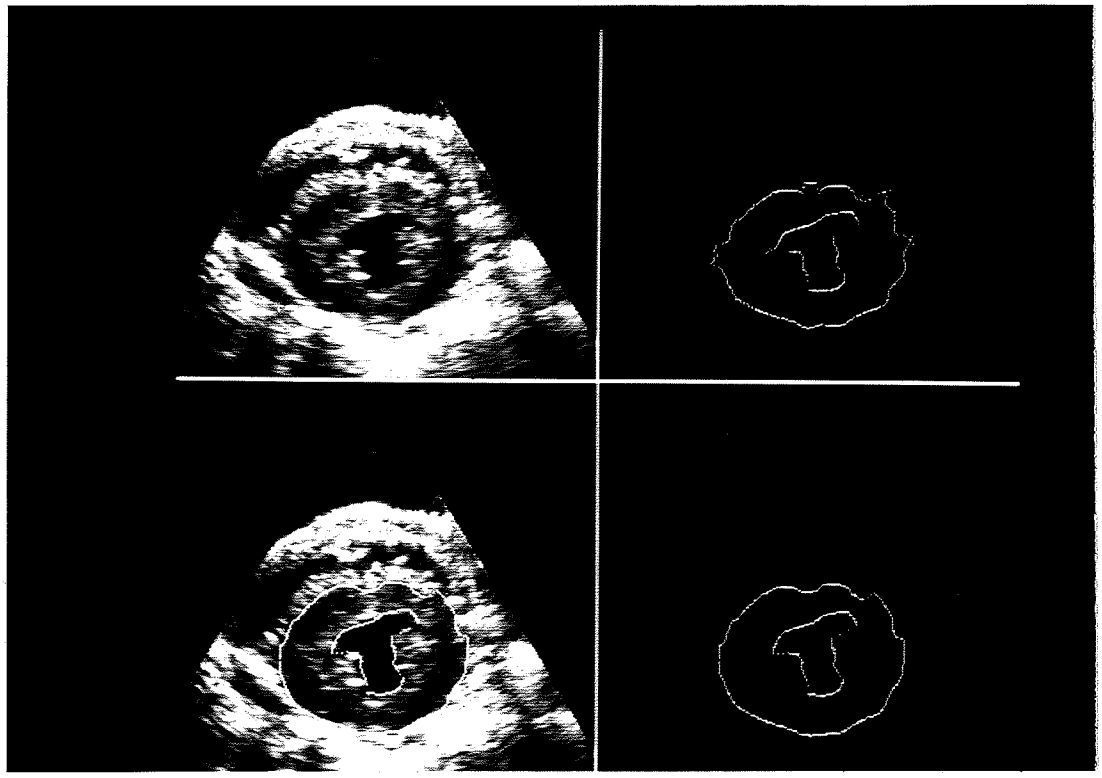


Figure 2.4. Detected boundaries in "baseline" dog study. *Upper Left:* Original image. *Upper Right:* Detected raw edge points. *Lower Left:* Edge boundaries after processing. *Lower Right:* Detected boundaries superimposed on the original image.

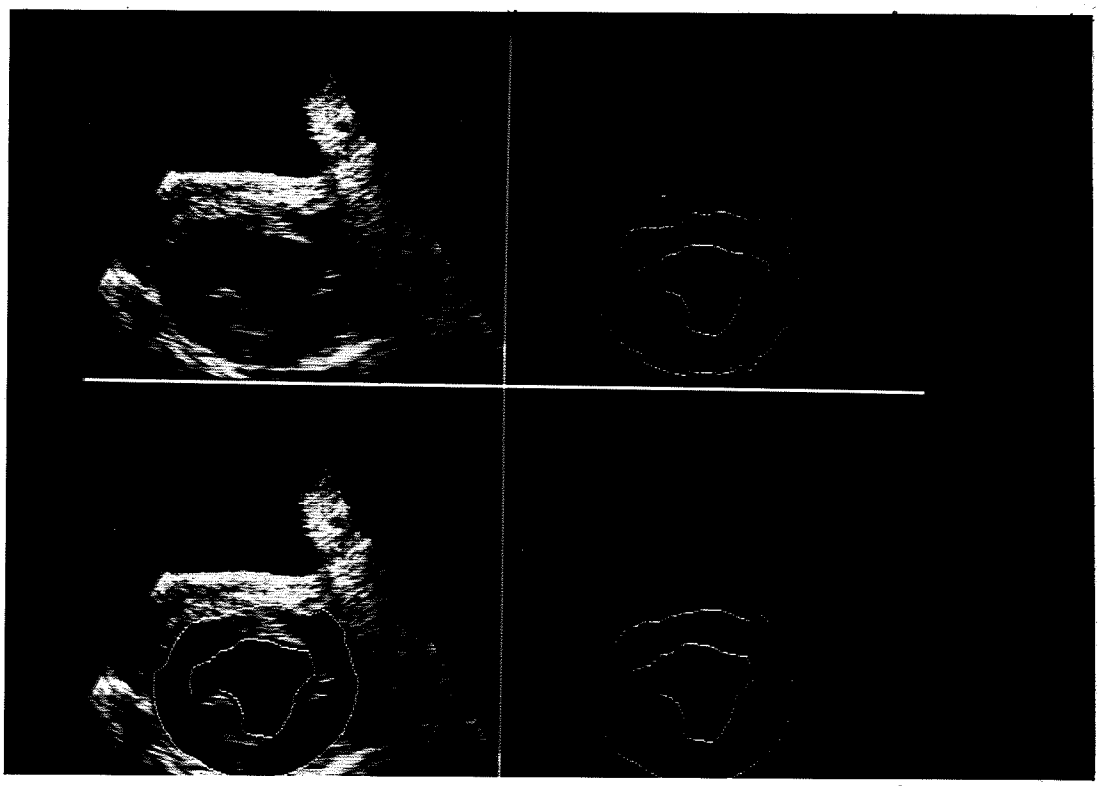


Figure 2.5. Detected boundaries in "post occlusion" dog study. *Upper Left:* Original image. *Upper Right:* Detected raw edge points. *Lower Left:* Edge boundaries after processing. *Lower Right:* Detected boundaries superimposed on the original image.

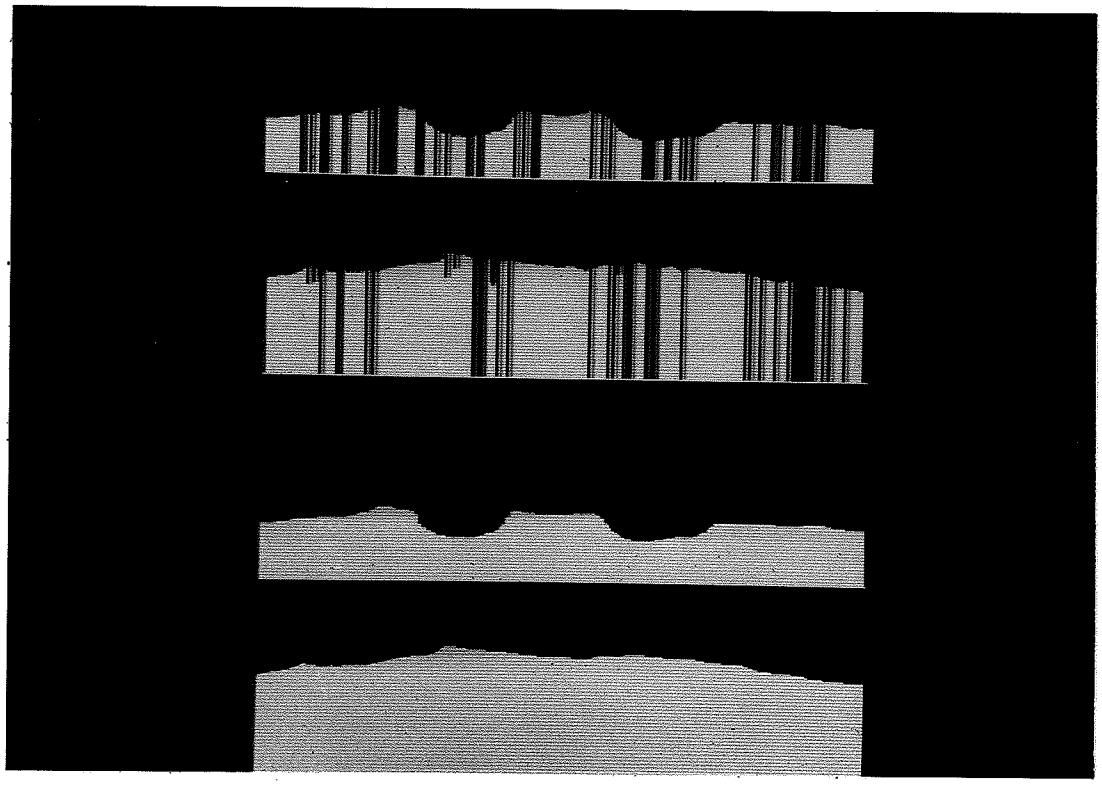


Figure 2.6. Distance-angle plots for phantom study. *Starting from the top:* Distance-angle plot of the inner boundary in Figure 2.3. Distance-angle plot of the outer boundary in Figure 2.3. Processed distance-angle plot of the inner boundary in Figure 2.3.

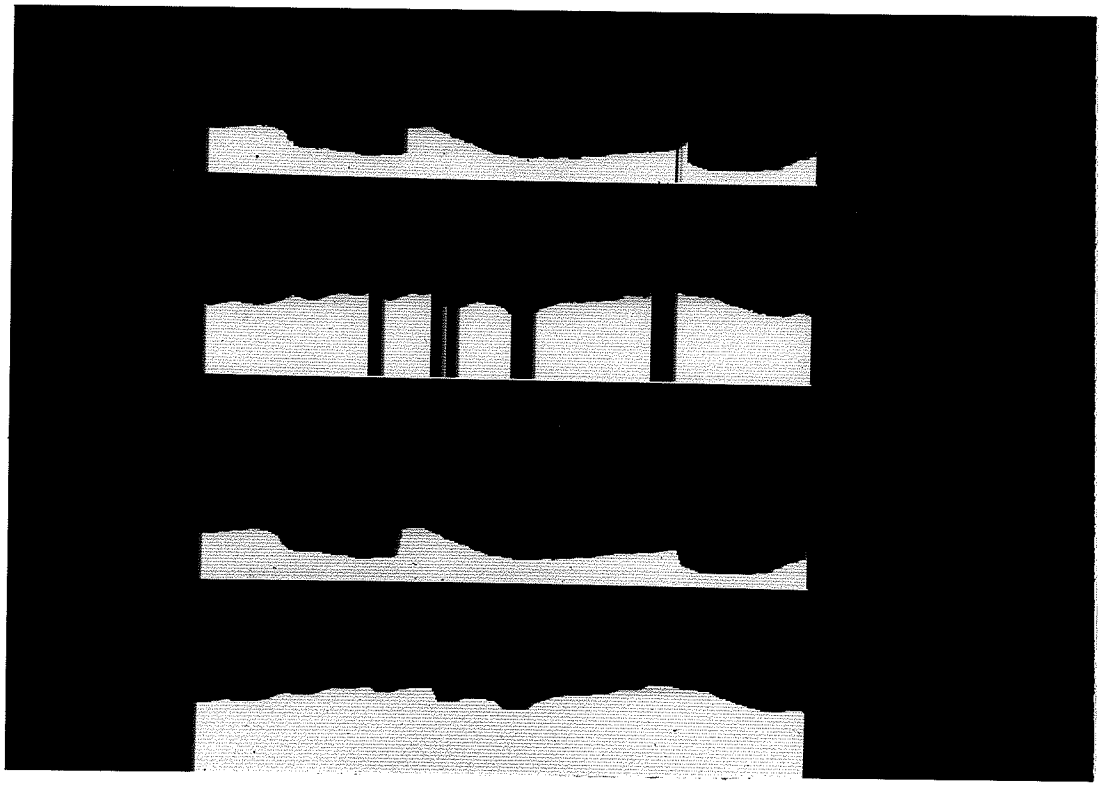


Figure 2.7. Distance-angle plots for dog study. *Starting from the top:* Distance-angle plot of the inner boundary in Figure 2.4. Distance-angle plot of the outer boundary in Figure 2.4. Processed distance-angle plot of the inner boundary in Figure 2.4. Processed distance-angle plot of the outer boundary in Figure 2.4.

To improve the estimate, a secondary search is conducted whenever the distance between successive endocardial boundary points is above a threshold, which is set at 5 in our experiments. This simple secondary search is not intended to be a general purpose linking procedure and does not guarantee a resulting connected boundary linking the two points. Instead, it is restricted to our application by assuming that only a small portion of the endocardial boundary is missing, and that the missing portion has a simple shape. As discussed before, after processing the original image by an edge detector, such as the $\nabla^2 G$ operator, there is an initial edge map marking the edge segments. In the discussion that follows, we assume the edge detector is the $\nabla^2 G$ operator and will refer to the edge segments as the zero-crossings. The secondary search algorithm works on the edge map and always extends the endocardial boundary along zero-crossings. By doing this, the search is quite conservative in that it either terminates or is not conducted in areas where the intensity variation is too weak.

The algorithm seeks to link up two points, at least one of which is assumed to lie on a zero-crossing segment, although they may not necessarily be linked by a single zero-crossing. We shall denote the two end points where a gap exists on the endocardial boundary as A and B. If neither A nor B lies on a zero-crossing, the search is not conducted. We can now assume A is initially a zero-crossing point. The algorithm attempts to track from A to B by moving A, point by point, along the zero-crossing segment. A move is made if it will bring A closer to B. The algorithm stops if A is moved to B, or if A cannot be moved any closer to B. The search terminates if B is not a zero-crossing point; otherwise, the algorithm tries to move B closer to A. This repeats alternately until neither A nor B can be moved any closer to each other, or until they are linked to each other.

The limitation of moving the two end points closer to each other at all times is justified since we do not want the searching path to wander too far from the initial boundary. A direct result of this constraint is that the angle between the line segments AB and AA' is never larger than $\pi/4$, where A' denotes the point to which A is to be moved. This is consistent with the goal of linking A and B by the simplest and shortest possible path guided by the zero-crossings. Figure 2.8 shows the result of conducting this secondary search to extend the endocardial boundary.

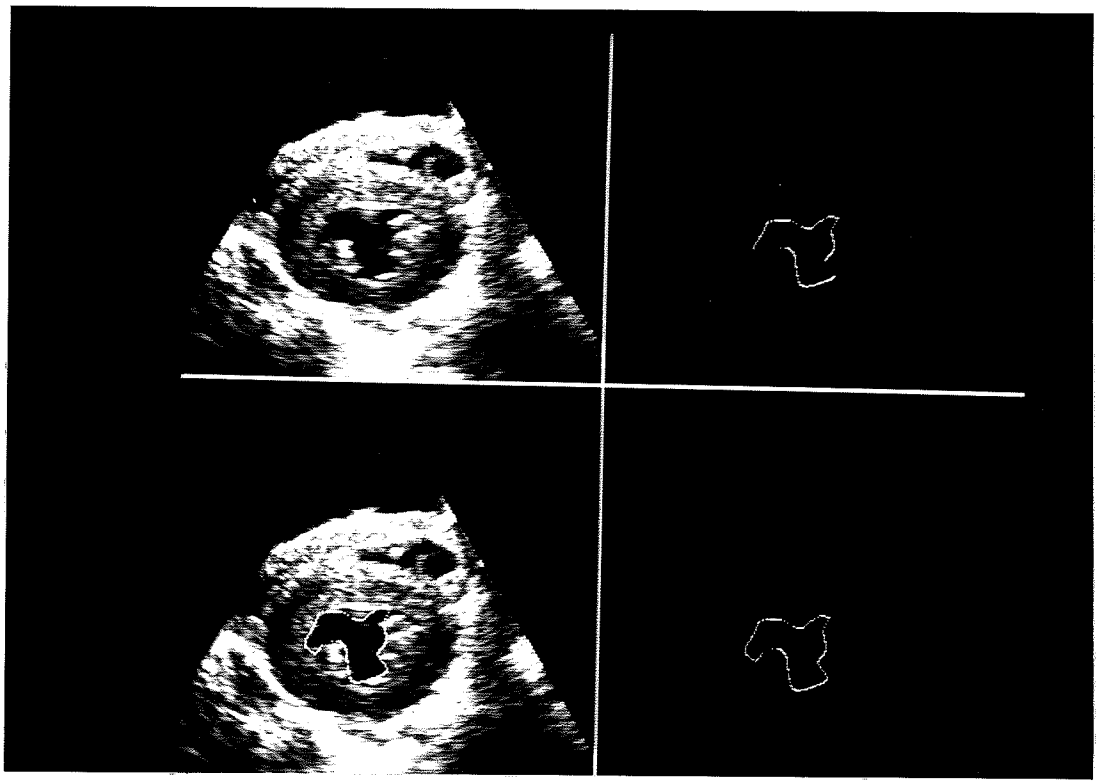


Figure 2.8. Boundaries found after secondary search. *Upper Left:* Original image. *Upper Right:* Detected raw edge points for endocardial boundary. *Lower Left:* Endocardial boundary after secondary search. *Lower Right:* Detected endocardial boundary superimposed on the original image.

2.4. Verification of Detected Boundaries

Validating the accuracy of the detected boundaries from the edge detection algorithm is difficult to perform because it is impossible to directly measure the left ventricular boundaries *in vivo*. Different methods of validating results obtained by computer processing of echocardiograms include comparing the results with those hand-traced by trained experts, and with direct measurements made from excised hearts [Col86].

An alternative method of verification is to process images of a tissue-equivalent phantom with known dimensions. However, even in this case, the accuracy of the results, as determined by comparing them with known dimensions, is limited by the accuracy of the imaging system that acquires the data. The algorithm was applied to 16 images of a phantom with increasing inner and outer diameter to simulate an image sequence depicting a cardiac cycle. The diameter of the automatically determined boundaries correlated highly with the known phantom dimensions. For the inner boundary, the correlation coefficient was 0.995 with a 0.88 mm. root-mean-squared error over a range of 30 to 50 mm. true diameter. For the outer boundary, the correlation coefficient was 0.997 with a 3.83 mm. root-mean-squared error over a range of 60 to 72 mm. true diameter. Figure 2.9 shows a plot of the diameters obtained experimentally compared to known dimensions.

2.5. Summary

To briefly summarize, a new algorithm based on the use of early processing methods to detect features, subsequent processing such as classification and interpolation to correct the expected errors in the early detection, and a further secondary search process to ensure an accurate estimate was presented. We have also overviewed the computer vision area and discussed the efficacy of applying some of the techniques to identify the left ventricular wall.

For the results shown, a $\nabla^2 G$ operator is used for early detection of edges. Other operators such as the directional zero-crossing detector [Har84] had also been used and equally satisfactory results were obtained. Using window operators larger than the conventional small windows allows multiple resolution methods to be incorporated in the edge detection procedure. The algorithm can be developed into a multiple resolution processing scheme by using a large window smoothed image for initial detection of major edge segments, followed by smaller and smaller windows until a complete boundary

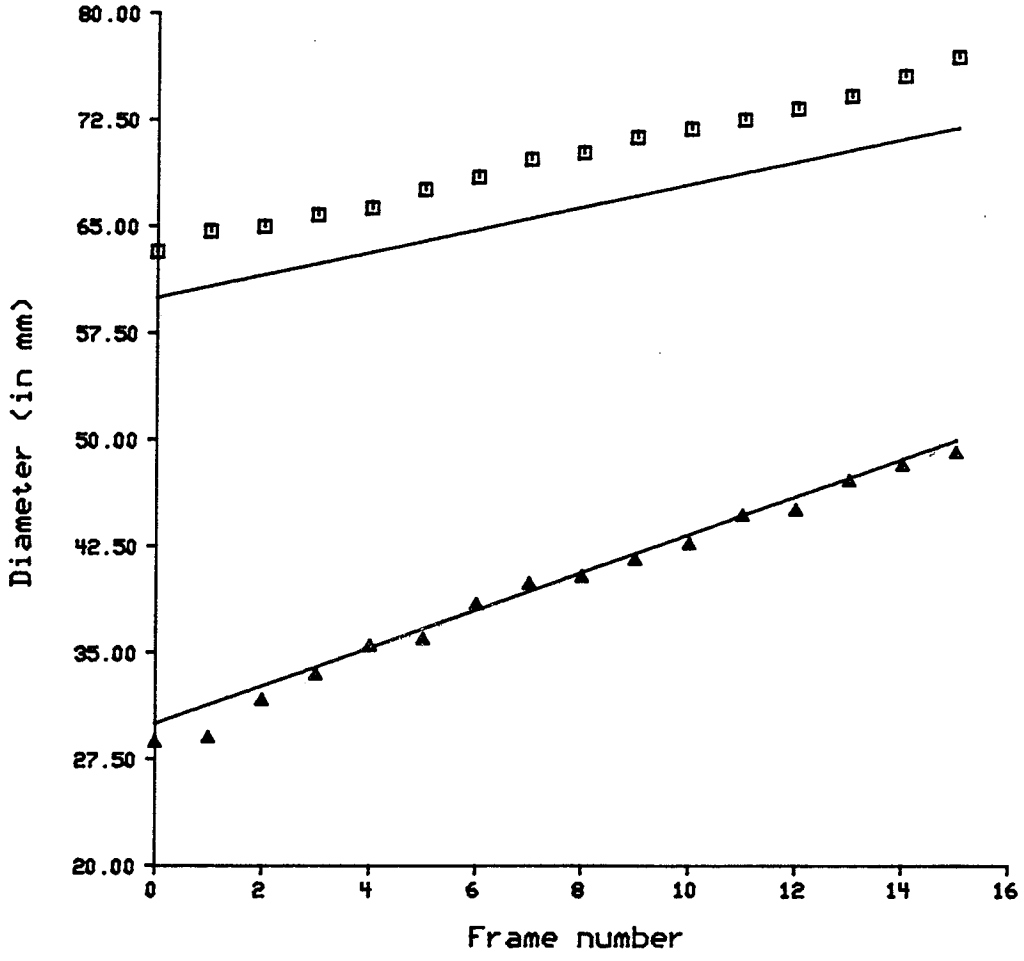


Figure 2.9. Comparing experimentally obtained phantom diameters to known dimensions. The lines in the plot are the known dimensions of the epicardial and endocardial diameters plotted against frame numbers. The experimentally obtained epicardial and endocardial diameters are marked by squares and triangles, respectively.

is found.

The algorithm presented in Section 2.3 has its limitations, although it could form the basis for further advances in methods of boundary detection in echocardiograms. As it is basically extended from the radial search method, the algorithm incorporated much of the limitations, such as the needs for operator-assigned search centers and search limits. The improvements primarily come from more advanced image processing procedures, in the form of the larger $\nabla^2 G$ operator which provides more reliable initially detected edge points; an effective postprocessing procedure to reject noise points; and a secondary search necessitated by the limitations of the radial search.

The limitations can be overcome by using the results obtained in previous frames to establish the search center and limits for the current frame. It should also be noted that even the ideal image gray level edge does not always correspond to the true left ventricular boundary. The solution to achieve a completely automatic segmentation of the echocardiographic images is through the use of high level symbolic reasoning. The work reported here should be considered as the first step of the signal-to-symbol translation process that is essential to the building of the overall system for identifying the left ventricular wall from its background. To be successful, such a system needs to exploit the temporal redundancy of the image data, which requires that it establish the attributes of image features, such as their velocities in the image plane. In the next chapters, we shall be considering the determination of such information from an image sequence.

CHAPTER 3
IMAGE SEQUENCE ANALYSIS

3.1. Introduction

Information such as the motion pattern of the ventricular walls in an echocardiogram sequence is inherently dynamic, necessitating the processing of a sequence of echocardiograms. Furthermore, extracting information from an image sequence is particularly useful in situations where the signal-to-noise ratio of each image frame is low; this is similar to the approach taken in classical detection theory that seeks to improve the performance at low signal-to-noise ratio by increasing the number of observations contributing to the decision. In this chapter, we shall first examine past work in processing echocardiogram sequence, followed by a broader view of image sequence analysis in computer vision. Our concentration will be on the extraction of motion information without using results from image segmentation or scene interpretation.

A straightforward way to reduce noise in echocardiograms is to take the average of several image frames. Due to the gross motion of the heart walls, averaging several successive frames [Eze85] tends to smooth out edges that are moving. An alternative is to average several image frames taken from the same location of the cardiac cycle [Sko81, Bud83]; aligning the image frames is difficult to implement accurately in practice, however. Moreover, the heart is moving in a three-dimensional space, with lateral motion across the image plane and vertical motion. This would introduce errors when taking averages of several image frames, offsetting the advantage gained by noise suppression.

In [Zha84], an image frame is thresholded to locate areas with image motion for detecting edges. A local threshold is determined for an image region from a temporal cooccurrence matrix, which is a matrix defined for a pair of image frames. The (i,j)th entry of a temporal cooccurrence matrix is the number of pixels that has, at the same location, intensity value i in the first frame and intensity value j in the second frame. The threshold θ is selected by maximizing

$$P(\theta) = \sum_{i=1}^{\theta} \sum_{j>\theta}^{K} M_{i,j} + \sum_{i>\theta}^{K} \sum_{j=1}^{\theta} M_{i,j}$$

where $M_{i,j}$ is the (i,j)th entry of the temporal cooccurrence matrix, and K is the maximum image intensity level. All pixels with values below this threshold θ represent stationary points, while points above the threshold represent moving points. This method is sensitive to drop-outs and spurious noise that often plague echo images largely because thresholding, as is true of all other pointwise image processing operations, ignores the two-dimensional spatial information of an image.

Most other efforts at using the temporal information in processing echocardiographic images have been *ad hoc* in nature, such as treating a binary ultrasonic image sequence as three-dimensional data and applying nonlinear processing techniques for removing noise and filling in dropouts [Ver79]; using edge points found in the neighboring frames to fill in dropout edge points [Zha84]; and frame differences to aid edge searching [Bud83]. These techniques have often failed due to the gross motion of the heart caused by temporal undersampling.

Since edge detection is perceived as an important step of image processing, a natural extension to edge detectors that operate in a single image frame is an enhancement operator that finds time varying edges, which are edges that have moved from one frame to another. In [Her78], a three-dimensional edge detector is used as a time varying edge detectors by treating the temporal axis as the third axis. Since the edge model used by this operator does not differentiate between the temporal dimension and spatial dimensions, the operator tends to produce two responses when the intensity jump across time frames is larger than the intensity jump in the spatial coordinates. When the movement of the edge is larger the spatial edge strength, the operator produces responses at two different locations: where the edge was initially, and where the edge ended up after the motion [Hay83]. This makes it difficult to distinguish between the above scenario and the case where there are actually two moving edges.

In [Hay83], moving edges are found by combining the temporal difference and the edge strength in a single frame by a logical *AND* operator. This operator is implemented in practice by forming the product of a difference picture for two frames and the spatial gradient magnitude map for one of the frames. While this is an improvement to using a three-dimensional edge detector, it can be problematic when the edge is not well formed in one frame.

A pair of echocardiograms are shown in the top half of Figure 3.1. Moving edges are detected by forming the product of a difference picture and a Sobel edge magnitude map. The product map is shown in the lower left of Figure 3.1. The pair of images are also processed by smoothing using a 21×21 Gaussian shaped filter prior to the moving edge detection. The result of this processing is shown in the lower right of Figure 3.1. The temporal difference alone may not be enough evidence to allow the detection of the moving edges, as can be seen in Figure 3.1.

There have been increasing interest in image sequence analysis for general computer vision. Analysis of an image sequence can be conducted at different levels of a vision system [Nag83a]. Motion information extracted from a sequence of images provides important cues for processes at different levels in a vision system. At the signal level, motion information is useful to augment processes such as image segmentation; at a higher level, processes such as scene interpretation will benefit from information such as the three-dimensional structure of objects as recovered by analyzing their motion. One of the goals of image sequence analysis is to establish the motion parameters of objects in a scene. Image sequence analysis, in general terms, shares a surprisingly large number of issues encountered by other vision tasks such as feature detection and object identification. For example, instead of matching subimages to special features such as edges, or matching objects in a scene to a machine part stored in a library, image sequence analysis matches items found in an image to those found in another image. After such matches are established, the motion parameters can be computed. Depending on the level that the analysis is operating on, the matching item could be pixel values, image features, or parts of a scene object. The search for matches in an image plane prior to recognition of scene contents is commonly known as the *correspondence* problem.

There are at least two ways of detecting motion: following landmarks in an image sequence, and deducing motion in scenes without explicit landmarks. Correspondingly, there are two mechanisms in human vision systems for image sequence analysis: a long range and a short range mechanism [Ull79]. Landmark tracking is done by first detecting edges or gray level corners [Nag83b] from each image and matching them from frame to frame [Roa79]. The immediate problem of applying these processing in echocardiogram sequences is the lack of gray level corners in the images. More importantly, unlike images containing man-made objects, gray level corners in different echocardiographic images do not necessarily correspond to the same point in

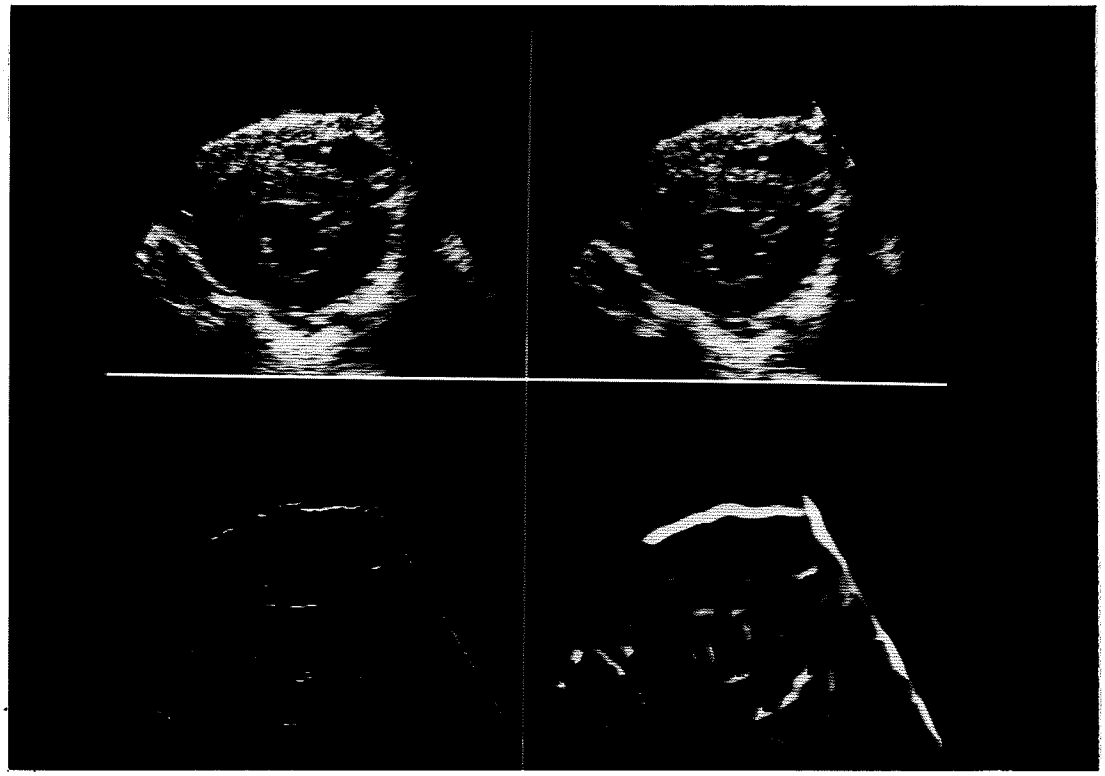


Figure 3.1. Detecting moving edges. *Upper Left:* First frame of original image pair. *Upper Right:* Second frame of original image pair. *Lower Left:* Detected moving edge from image pair. *Lower Right:* Detected moving edge from image pair after smoothing by 21×21 window. Moving edges were detected by forming the product of a difference picture and a Sobel edge magnitude map.

the scene. The matching rules for features are obviously dependent on the level of the features. High level features are easier to match, but they cannot be detected reliably in echocardiographic images. Examples of higher level features are the tip of the papillary muscles (defined to be the point with maximum curvature along the boundary, for instance) and points where the papillary muscles meet the endocardial boundary. As we progress to higher level feature matching, problems such as an appropriate data structure, reliability of feature detection, formulating matching rules would have to be addressed. The boundaries between motion analysis, image segmentation and scene interpretation are less defined at this stage; it is reasonable to assume that these processes would mutually benefit each other in a complete system.

The long range mechanism by tracking landmarks, or tokens, is used in human vision system primarily for "maintaining the perceptual identity of moving objects" [Ull79]. The problem of determining structure from motion is aimed at recovering the three-dimensional relationship of object parts based on analyzing the object motion in the image plane [Tsa84]. The correspondence problem is considered to have been solved by some other methods; the application of these long range methods to echocardiograms is obviously limited.

The issues associated with the correspondence problem in image motion analysis are encountered in another image analysis domain, *image registration and mapping*, which is concerned with images of the earth surface taken from high altitude by either satellites or aeroplanes. In *remote sensing*, matching is performed on images taken at different times. In *photogrammetry*, the matching is performed on images taken from different viewing angles. Other applications include matching observed images to a reference map. In a survey of image registration and mapping techniques [Kas83], it was concluded that syntactic, or landmark based, methods are more suited for high signal-to-noise ratio images while statistical methods are more suited for low signal-to-noise ratio images. An equivalent view is that there is a lower bound of signal-to-noise ratio for the reliable detection of landmarks. Furthermore, it was concluded that no one technique is universally applicable and that a mixture of both methods are needed in most applications. Since one of the motivations of analyzing an image sequence is to handle situations where scene interpretation or image segmentation cannot be performed reliably, image sequence analysis is most important at the early stages of a vision system. In the remainder of this chapter, we shall analyze the problem of determining motion information without using the result of image

segmentation or scene interpretation.

The *displacement vector* of a point on an image plane is the position change of the projection of an object point due to the change of viewing angle between the sensor and the object point. A change in the viewing angle can be caused by the displacement of the sensor, or the movement of the object point, or both. The displacement vector is thus an approximation to the velocity of the image point. At the signal level, the processing performed to estimate the velocity is based on the assumption that the intensity values that correspond to a region in a scene do not change drastically from one frame to another. This assumes, among other things, that the lighting condition stays the same in general computer vision. In echocardiography, this means that the time gain compensation setting has to remain unchanged during the entire data acquisition period.

Displacement vectors are determined by matching image points obtained in different image frames. Since the intensity values are not reliable enough for pointwise matching, the matching is based on a local neighborhood of the point. Estimators of this nature cannot have arbitrarily high accuracy and resolving power simultaneously. Performance of any estimation procedure can be measured by the resolution capability and the accuracy of the result. *Accuracy* is determined by how close the computed displacement vector values are to the correct values. *Resolution* refers to how well two different displacement vectors can be resolved. In practice, resolution amounts to how valid a constant velocity can be assumed for a patch on the image plane.

The correspondence problem can be viewed by considering the following model. Let

$$I_1(\vec{p}) = P\ (\ S\ (\ \vec{x}\ )\ ) \tag{3.1a}$$

and

$$I_2(\vec{p}) = P\ (\ S\ (\ D(\vec{x})\ )\ ), \tag{3.1b}$$

where $\vec{p}$ denotes a point on the image plane, $\vec{x}$ denotes a point in the scene, I_1 and I_2 are the intensity values observed at two different instances. S is the object that generates the image intensity values. In a general vision problem, it would be the reflectance of the object surface that is being illuminated. In echocardiography, it is the amount of energy that is being reflected back to the receiver at a particular point of the organ. P is the operator that represents how S is mapped onto the image plane. D is the operator that represents the deformation that S is undergoing. In general, $\vec{x}$ and $\vec{p}$ will

have different dimensions. Since echocardiograms are cross-sectional views, they are both of dimension two, however. Equation (3.1) can be simplified to

$$I_1(x,y) = S(x,y) + n_1(x,y) \tag{3.2a}$$

and

$$I_2(x,y) = S(D(x,y)) + n_2(x,y) \tag{3.2b}$$

Here, we assumed the image projection operator P only adds noise, represented by n_1 and n_2. A further assumption is that we are considering cross-sectional views or parallel projection, where the distance of the object from the image plane does not affect its appearance, or its projection, on the image plane. We have also used x and y to represent locations both on the image plane and in the scene. The problem then becomes: recover D by observing I_1 and I_2 in the presence of noise.

D is generally modeled as an affine transform, which can represent rigid body rotation and translation. An *affine* transform D that maps points in $\mathbf{R}^2$ to points in $\mathbf{R}^2$, where $\mathbf{R}$ is the set of real numbers, can be written as

$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}.$$

$(x_i \;\; y_i)^T$, i=1,2 represents two points on the R^2 plane. $\mathbf{A}$ is a 2×2 matrix and it represents the linear transformation component of D, constrained by the fact that its determinant is nonzero. Denoting the i,jth entry of $\mathbf{A}$ as a_{ij}, if (1) $a_{11}=a_{22}$, (2) $a_{12}=-a_{21}$, and (3) the determinant of $\mathbf{A}$ is equal to 1, $\mathbf{A}$ represents a rotational matrix. If only conditions (1) and (2) are met, $\mathbf{A}$ represents a rotational matrix with a scale factor equal to the determinant of $\mathbf{A}$. The translation component is represented by $(t_x \;\; t_y)^T$. An affine transform is known for mapping lines to lines, collinear points to collinear points, and noncollinear points to noncollinear points. These properties make it a suitable choice to represent rigid body motion in a plane that is parallel to the image plane. *Rigid* body motion here refers to the case where different parts of the body having the same image plane motion parameter. Deformation of objects can be approximated as locally rigid [Web83].

In general, without a good model of the image formation process, i.e., without knowing how S maps $\vec{x}$ into intensity values, it is difficult to invert the process. Hence, the solution is usually limited to searching for two image regions with similar variation characteristics.

Intuitively, matching based on local neighborhoods has to depend on the information in the form of intensity variations in a particular neighborhood. Moreover, in a two-dimensional setting, a match is only possible in the direction where there is enough variation. In the vicinity of an ideal edge where the only intensity variation is along the gradient direction, only one component of the displacement can be estimated. This is referred to as the *aperture effect* [Ull79]. In Figure 3.2, we see a square being translated. When considering a local neighborhood, as represented by the ellipse, only one component of the true velocity can be obtained. It can also be seen that the only points with enough information to recover the true velocity are the four corners, hence the prominence of gray-level corners in image sequence analysis [Nag83b].

3.2. Region Based Approach

The region based approach selects a subimage containing the point for which it is computing the displacement vector. It then searches in the second image a subimage that best fits the first subimage according to certain criterion. The most commonly used criterion is the mean squared error, which leads to the correlation of a subimage with a larger search area for matches. Correlation methods have been used in other computer vision applications such as stereo matching [Yak78]. Common criticisms include [Arn83]: computational expense, incorrect results when dealing with occlusion, poor accuracy when compared to feature based methods, and lack of guidelines to choose matching neighborhoods. Typically a constant size, such as 8×8, window is used for correlation. The use of the autocorrelation function is noted for being helpful in evaluating the suitability of a subimage for matching [Yak78]. In [Mor77], an interest point detector, which evaluates the directional sampled variance of a subimage, is used to find suitable points for establishing matches between two different frames of an image sequence.

It is interesting to note that except for the dimension of the data, the correspondence problem is similar to the original echocardiographic data acquisition or the radar signal processing problem. As we have mentioned in Chapter 1, in radar or echocardiographic systems, a signal is transmitted and an echo is received after bouncing off a target. The receiver has to detect the existence of a returned signal from the observed data and, in some applications, to estimate relevant parameters of the returned signal. In terms of the model represented by Equation 3.2, n_1 is identically zero, since the reference signal is known. It is the aspect of detecting a known signal in noise

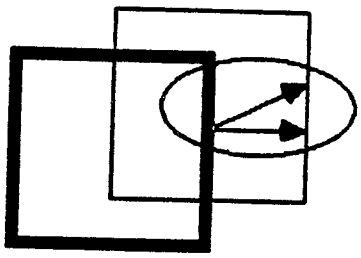


Figure 3.2. The aperture effect.

that is similar to the correspondence problem. Due mainly to the assumption of white Gaussian noise, correlation is established in radar signal processing as the optimal method for detection. Radar signals are designed based on their ambiguity functions [Fra81], which, in the time-domain, are the autocorrelation functions of the radar signal. The problem of choosing a subimage for matching by correlation can then be viewed as choosing, from a large collection of signals, the suitable ones based on the radar signal design criteria.

We can look to other signal processing disciplines to find issues similar to those facing correspondence problems. In sonar signal processing, the correlation method is used to determine the time delay between signals received at two spatially separated sensors in the presence of uncorrelated noise [Kna76]. This problem, which conforms to the model represented in Equation 3.2, appears to be even more similar to the correspondence problem than does the radar detection problem; after all, the radar signal is itself noiseless while both received signals in sonar are noisy. The main emphasis in sonar thus far has been on tackling the noise problem by the design of prefilters to accentuate the signal passed to the correlator at frequencies for which the signal-to-noise ratio is highest and simultaneously to suppress the noise power. On the other hand, the major obstacle in correlating subimages is not noise, but rather that one of the subimages is frequently a rather severely distorted version of the other.

Perhaps the major factor for restricting the use of correlation techniques in image motion analysis is the problem of rotation of portions of an image. In [Mos81], optimum windows are found for registering two images based on two-dimensional correlation. Geometric distortions included in the analysis is modeled by a linear transform of the coordinate axes, which include rotation and scaling of the coordinate axes. Performance measures used to derive the optimum windows include a peak-to-sidelobe measure and registration error. Maximizing the peak-to-sidelobe measure improves the detection of peaks in the correlation function; minimizing the registration error improves the accuracy of the match after the true peak is found. A small amount of rotation (up to 5°) is considered to be tolerable in in choosing optimal window. One possible solution to larger scale rotation might be to use a bank of matching signals, each of which is a rotation of the original subimage, to match in the second image. The number of such matching signals is dependent on the tolerance of each to rotation distortion; or equivalently, the number is dependent on the performance demanded.

Correlation methods is claimed to produce incorrect results in areas where occlusion occurs [Arn83]. To be specific, consider the situation when a portion of an object which is occluded in one image is uncovered in another image. Suppose the subimages that are to be matched were chosen such that one of the subimages contained the uncovered portion. Since parts of the two subimages are indeed different, the correlation method will fail. This has been used as a convincing example of why a feature based method should be used. It is usually claimed that since occlusion occurs in areas with edges, by detecting edges and subsequently matching them, feature based methods are immune to the occlusion problem. On closer examination, however, the performance of the correlation method is degraded, but it does not actually break down.

Suppose the first signal is made up of a nominal part and an occluded part, and the second signal comprises the same nominal part and a different occluded part. The correlation of the two signals is the sum of the correlation of the nominal part, the cross correlation of the nominal signal with the two occluded parts, and the cross correlation of the two different occluded parts. The degradation of the performance can be viewed in terms of the cross correlation terms, and is also dependent on the ratio of the extent of the nominal part to the occluded part in each signal. It is important to note that the cross correlation terms are constrained by the fact that the typical case involves the occluded parts and the nominal signal being on opposite sides of an edge. Hence the correlation function will be flattened, since the nominal part will not have much intensity variation; yet, there is a limit to the degradation of the performance since the peak will still be in the vicinity that corresponds to a correct match. This is further justified by the fact that without such constraints, other methods such as feature based methods would also fail due to insufficient information for feature detection.

3.3. Transformation Approach

The transformation approach computes the motion parameters that by characterizing the difference between two image regions. Unlike the region based approach, the transformation approach does not explicitly search for matching subimages. One of the image regions is assumed to be a deformed version of the other, and the nature of the deformation is either known or is modeled by some known distortions. The deformation is commonly modeled as translation, rotation, or scale change. Here, scale change refers to the distortion of the coordinates, not the scaling of the image intensity values.

Parameters of the deformation is related to the motion parameters, and they are computed by transforming the two regions into an appropriate domain where the parameters of the deformation are made explicit. For example, if the deformation is translation, the translation component can be easily determined by using the Fourier transform and its well known relationship of spatial translation and frequency phase shift.

In [Jai87], a complex logarithm transformation is used to normalize the rotation and scale factors to compute depth of objects from the sensor. Image points in rectangular coordinates, denoted by x and y, are converted to the complex logarithm space by first representing (x,y) as a complex number $z = x + jy$. z is then mapped to the complex logarithm space by $w = \log z$, where $w = u + jv$. u and v are determined by:

$$u(r,\theta) = \log r$$

and

$$v(r,\theta) = \theta,$$

where

$$z = re^{j\theta}.$$

The transformation approach to the correspondence problem is similar to the pattern recognition problems, which frequently have to match observed signals to a reference signal in a position, rotation, and scale invariant domain. It is interesting to note that computationally, the complex logarithm mapping differs from the Fourier Mellin transform [Cas77] only by a last step of taking the Fourier transform along the log$r$ coordinates. Indeed, this has been conjectured in [Jai87] but was dismissed for lack of evidence that such a transform takes place in biological visual systems.

The limitation of having to model the deformation as one of translation, rotation, and scale change amounts to constraining the scene objects to be undergoing rigid body motion. Moreover, a large number of pixels is needed for the computation of the transformations. Typically the the entire image, or at least a significant part of it, is used to compute the motion parameter (see, e.g., [Jai87]). Hence it is further limited by the constraint that the motion parameter which is to be computed is uniform over the entire image region that is transformed. This approach only allows one to determine a coarse estimate of the global motion parameter.

3.3 Image Flow Approach

The image flow approach evolves from the optical flow research, which originally at least, addresses the situation where the sensor is in motion and the scene is stationary [Mar82]. The significance is that every point in the image plane has a nonzero velocity, except for the *focus of expansion*, which is the apparent origin of all velocity vectors in the image plane. The image flow approach is based on determining a velocity component from the spatiotemporal gradients. This approach is based on an "implicit match," which assumes that the same intensity value is registered in the image plane at different frames for the same point in the scene. Hence, we can write:

$$I(x,y;t) = I(x+d_x, y+d_y; t+d_t). \tag{3.3}$$

By using Taylor expansion on the right hand side of Equation (3.3) and ignoring the higher order terms, the velocity information can be determined from an image sequence by measuring the spatiotemporal intensity change [Fen79]:

$$\frac{\partial I}{\partial x}\cdot\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\cdot\frac{\partial y}{\partial t} = -\frac{\partial I}{\partial t}, \tag{3.4}$$

where I represents the intensity as a function of spatial coordinates x and y and time t. The left hand side of Equation (3.4) can be interpreted as the inner product of the velocity vector

$$\vec{v} = \left(\frac{\partial x}{\partial t} \quad \frac{\partial y}{\partial t}\right)^T$$

with the spatial gradient vector

$$\nabla I = \left(\frac{\partial I}{\partial x} \quad \frac{\partial I}{\partial y}\right)^T.$$

By computing ∇I and $\partial I/\partial t$ from the given images, the component of $\vec{v}$ along the direction of ∇I can be determined.

Equation (3.4) is derived based on the following assumptions [Fen79]:

(a) the same image intensity is registered for a fixed point of a physical object from different viewing angles, hence a change in the intensity values over time at a fixed pixel location must equal the change over space at some fixed time;

(b) the image intensities can be modeled as a linear function locally;

(c) the image motion can be modeled as a linear function locally; i.e., the object is undergoing rigid body translation on the image plane.

Since only one component of the velocity vector can be determined from Equation (3.4), the component of the velocity vector that is perpendicular (in the image plane) to the gradient direction has to be found using some other information. Horn and Schunck [Hor81] assume that the velocities vary smoothly and average the velocity components over a neighborhood to calculate the velocity vectors. Hildreth [Hil83] computes the initial velocity components for points along a contour, and then estimates the true velocity vectors by minimizing the variation of the velocity along the contour. All of these methods are computationally expensive, highly dependent on whether the smoothness of velocity variation assumption holds, and crucially dependent on the accuracy of the initial estimates of the velocity components based on Equation (3.4).

Besides the difficulty of estimating the missing velocity component, there are other disadvantages of this method. In practice, ∇I is usually approximated by some numerical differentiation methods, while $\partial I/\partial t$ is usually estimated by simply subtracting one image from another. Numerical differentiation of noisy data should be preceded by data smoothing to compensate for the accentuating effect of the differentiation step on noise, as discussed in [Tor86]. However, the temporal sampling rate is usually very low in echocardiography images, making averaging a large number of consecutive frames impractical. While the data can be spatially smoothed by a relatively large window, smoothing the data over time, if performed at all, is still limited to a very small number of frames, typically no more than three frames. We shall take a closer look at an implementation of the image flow approach in the next chapter.

CHAPTER 4
TOTAL LEAST SQUARES BASED IMAGE FLOW ANALYSIS

4.1. Estimating Displacement Vectors

Estimating displacement vectors in image sequence analysis is used to derive velocity vectors on the image plane. One approach is to directly search for matches from frame to frame; i.e., solve the correspondence problem. An alternative is to use the image flow approach, which assumes implicitly a match exists for every point on the image plane. This basic assumption means that any change in the intensity values over time at a fixed pixel location must equal the change over space at some fixed time (Equation (3.3)). From this simple assumption an image flow equation (Equation 3.4) can be derived [Fen79]. A one dimensional example is shown in Figure 4.1 to illustrate the principle of the image flow equation. The graph of a line with slope m is shown being displaced from left to right by an amount d_x from time t_0 to t_1. Denoting the line at instances t_0 and t_1 as g_0 and g_1, respectively, it can be seen that at a fixed point x_0:

$$m \times d_x = -(\, g_1(x_0) - g_0(x_0)\,). \tag{4.1}$$

The linearity of g_0 and g_1 and the uniformity of translation is obviously essential to the validity of Equation (4.1). The image flow equation relates the spatial gradient of image intensity I to the temporal intensity gradient, and is now restated as:

$$I_x\, v_x + I_y\, v_y = -I_t, \tag{4.2}$$

where $I_x = \dfrac{\partial I}{\partial x}$, $I_y = \dfrac{\partial I}{\partial y}$, $I_t = \dfrac{\partial I}{\partial t}$,

$$v_x = \frac{\partial x}{\partial t}, \quad \text{and} \quad v_y = \frac{\partial y}{\partial t}.$$

It is not difficult to visualize Equation (4.2) as an extension of Equation (4.1)

to two dimensions. Indeed, Equation (4.1) is a special case of Equation (4.2) in the vicinity of an ideal edge, i.e., one which has no intensity change in any direction other than that of the gradient direction. In general, Figure 4.1 still depicts correctly the cross-section plane taken along the gradient direction.

Using the same notation as in Chapter 3, we denote the spatial gradient field of the image intensity I as

$$\|\nabla I\| = \begin{pmatrix} I_x \\ I_y \end{pmatrix},$$

and the velocity vector as

$$\vec{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}.$$

By determining the components I_x, I_y, and I_t, the component of the displacement vector in the direction of the local intensity gradient can be determined. The magnitude of this component, $\vec{v_p}$, is given by:

$$\|\vec{v_p}\| = \frac{\|\vec{v}\cdot\nabla I\|}{\|\nabla I\|} = \frac{|I_t|}{\|\nabla I\|}. \tag{4.3}$$

By computing only one component, the results computed using Equation (4.3) have to be used in conjunction with other assumptions to recover the "lost" component, i.e., the component that is perpendicular to the gradient direction.

Computing $\|\vec{v_p}\|$ for individual points using Equation (4.3) have been known to produce noisy results [Jai87] and this has limited the applicability of the results to any subsequent processing. There are many factors that could contribute to the unsatisfactory results, such as the absence of a true match due to the invalidation of the basic assumption by input image noise, unstable data acquisition conditions, or image areas being uncovered or obscured by scene object occlusion. Incorrect matches may exist when either the spatial or temporal sampling rate is too low. Regardless of whether a correct match exists, a numerical value computed according to Equation (4.3) is generated. Whereas in the region based approach, a search for a match can result in an explicit failure, either due to lack of potential matches or due to too many ambiguous matches, there is no such mechanism in the image flow approach to handle these situations. Either post-processing will have to distinguish between correct results and incorrect ones, or preprocessing have to be performed to minimize the incorrect matches.

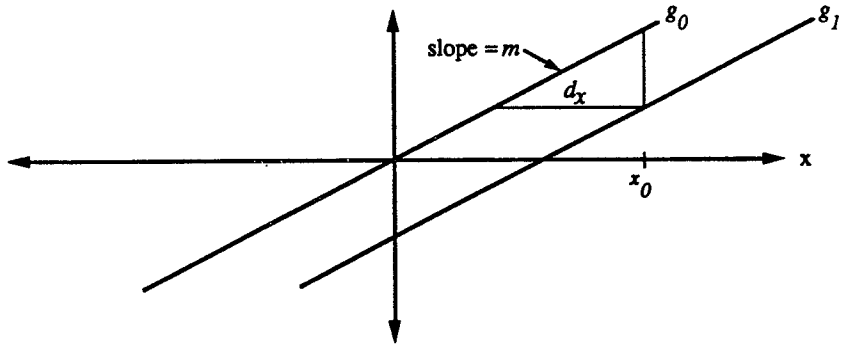


Figure 4.1. **One dimensional view of the image flow equation.**

The effect of the temporal sampling rate on the accuracy of the "implicit" match can be seen from the following simple example. Suppose the signal consists of a single sinusoidal $s(x,t) = \sin(2\pi f_x(x+vt))$ and that we observe s first at $t=0$ and then at $t=T_s$. If we want a match to be correct, the phase term $f_x v T_s$ must be less than one; i.e., the maximum velocity v that can be correctly computed must satisfy:

$$v < \frac{1}{f_x T_s}.$$

It then follows that as the temporal sampling rate drops, the spatial resolution has to decrease. Conversely, if a high spatial resolution is desired, the maximum velocity range is limited, for a fixed temporal sampling rate. Analyses that are more rigorous or those that do not appeal to sinusoidal models of signals are hindered by the fact that alternative models of "real" images are still lacking.

An alternative view of Equation (4.2) is

$$\nabla I \cdot \vec{v} = -I_t. \tag{4.4}$$

The spatial gradient ∇I becomes an operator that maps the velocity $\vec{v}$ to form the observed temporal gradient value. The goal of the estimation problem is to determine $\vec{v}$ by observing I_t and ∇I. This formulation illustrates that the problem is an inverse problem; and it is ill-posed in the Hadamard sense because the solution is not unique. Determining a single component of $\vec{v}$ only constrains the true solution to lie on a line (see Figure 4.2) . Furthermore, in the presence of noisy observations, the solution may not vary continuously with varying observed data. Estimation results of problems of this nature is known to require tradeoffs between accuracy and resolution capabilities [Roo87]. More specifically, one can expect a proper approach is to approximate the correct solution. When a coarse approximate is established, it may not contain fine details, but it does not deviate severely from the correct solution. As more details are resolved, individual estimates may start to be severely corrupted. Factors limiting the level of resolution to contain arbitrarily high detail include signal-to-noise ratio of the data and the error amount tolerated.

In Chapter 3, we described the aperture effect when only local information is available to estimate displacement vectors; from the above discussion, it is obvious that the image flow approach, more than any other introduced in Chapter 3, is especially sensitive to this effect. As we mentioned

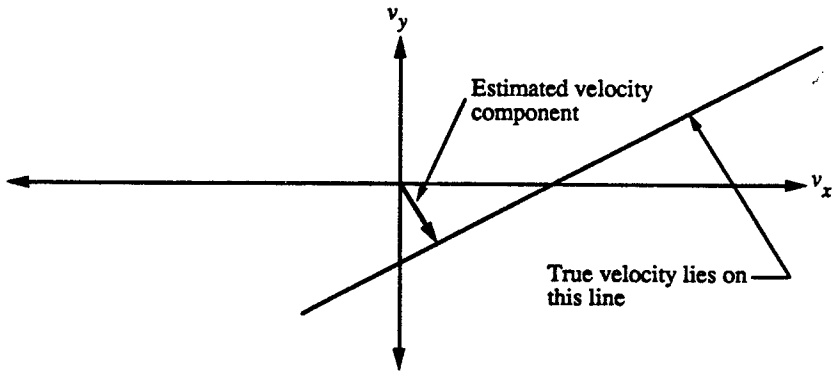


Figure 4.2. Constraining the true velocity.

before, estimating a component of $\vec{v}$ constrains the true solution to lie in a line. If the scene object is undergoing *rigid body motion*, i.e., all parts of the object are moving with the same motion parameter, $\vec{v}$, in the image plane, the true $\vec{v}$ can be uniquely recovered by applying Equation (4.3) to two points of the body that have different gradient directions. This solution obviously has its limitations: natural scene objects are not always rigid. Moreover, even bodies rigid in three dimensional space may have nonrigid (according to the above definition) motion in the image plane. A simple example is a solid plate undergoing rotation about an axis that is not perpendicular to the image plane. The corner of this plate that is nearest to the sensor would have a higher image plane velocity than any other corners.

In [Hor81], the velocity field is assumed to vary smoothly, i.e., velocity vectors for neighboring pixels do not differ significantly. By essentially averaging the perpendicular component, a true velocity vector for every pixel is sought. In [Hil83], smoothing of the vector components is performed not on an image plane neighborhood basis. Instead, the integrated value of the velocity variation magnitude along $\nabla^2 G$ zero-crossings is minimized. One of the motivations is that zero-crossings usually represent object boundaries. It is shown that velocities of blocks world type rigid polyhedral objects will be correctly recovered. It is also noted, however, that the solution is incorrect in the case of smooth curves undergoing general three dimensional motion.

In both of the above solutions, the assumption is that neighboring pixels have different gradient directions, hence allowing the true velocity for a neighborhood to be solved. This is performed by first computing a single component for individual pixels, followed by averaging these individual components. The difference is in the choice of image points used in the averaging. An alternative to the explicit averaging is to combine the temporal and spatial gradient estimates of all the points in a neighborhood to compute a least squares solution as the velocity [Mar86]. This approach will be examined in more detail in Section 4.3.

The *spatiotemporal frequency* approach [Jac87] is one that is similar conceptually to the image flow approach. Time varying image data are represented by their spatiotemporal frequency components. Each component represents a two dimensional sinusoidal grating spatially; this sinusoidal grating moves with a velocity determined by its temporal frequency. The spatial frequencies of the two dimensional grating specifies a direction of motion as being perpendicular to its axis of constant intensity. Hence, by examining a neighborhood at different spatial frequency ranges, one can

constrain the true velocity when spatial frequencies with different directions exist, i.e., except for the case of an ideal edge.

To briefly summarize, in the presence of ideal edge, no true solution can be sought. Nevertheless, all the previous work implicitly or explicitly assumes that in real images, gradient directions in any local neighborhood vary so that a true velocity can be determined.

4.2. Estimating the Spatial and Temporal Gradients

The velocity estimates are computed using the spatial and temporal gradients as input. Computing the spatial gradients in an image has been widely studied for the purpose of detecting intensity edges. We have described in some detail the difficulties of determining spatial gradients in Section 2.3. For purposes of *detecting* the presence of absence of edges, it is acceptable even if the operators produce a function of the gradient, as long as the function is monotonic. When using the gradient estimates as data for further computation, however, results that are more precise are required. Most of the reported results in optical flow estimation do not mention the procedure for estimating spatial and temporal gradients, making comparisons difficult. Our experience with finite difference operators are very discouraging, particularly in estimating temporal gradient. Temporal gradients are typically computed using the difference of two images. Our experience has been that this is inadequate and leads to unsatisfactory results.

Instead of using a finite difference operator, or performing numerical differentiation, we use a basis set of discrete orthogonal polynomials to fit a local neighborhood of image data. Both the spatial and temporal gradients are estimated using a procedure similar to the facet model approach [Har84]. Denoting $d(x)$, $x=1,...,M$, as data points observed in a noisy environment, a set of polynomials, P_i, for $i=1,...,N$, is used to represent the data as:

$$\hat{d}(x) = \sum_{i=1}^{N} \alpha_i \, P_i(x), \tag{4.4}$$

for $x=1,...,M$. α_i, $i=1,...,N$, are the *fitting coefficients*, and N is the *order* of the fit. The fitting coefficients are chosen to minimize some criteria of difference between d and $\hat{d}$. If the criterion is the mean squared difference, and if the set of polynomials are normalized and are orthogonal to each other, a simple procedure to determine the fitting coefficient α_i is by computing the inner product of d and P_i [Fra81]. Since the right hand side of Equation (4.4) is a finite sum, to compute the derivative of $\hat{d}$ with respect to x, $\partial\hat{d}/\partial x$,

one needs only to form the sum of the derivatives of individual P_i, weighted by the corresponding coefficient. It is noteworthy that the value of $\partial\hat{d}/\partial x$ can be computed for any value of x.

A set of third order polynomials in two variables, similar to the one in [Har84], is used to compute the spatial gradients. Using a general three dimensional gradient operator on time varying image data has not been satisfactory because the temporal sampling rate is typically different from the spatial sampling rate. Hence, we use second order polynomials to fit the image data temporally. By fitting the data with second order polynomials temporally, at least three image frames are needed. The time varying image data is denoted as $d(x,y;t)$, for $x=-N/2,...,N/2, y=-N/2,...,N/2$, and $k=0,1,...,K-1$. Here, x and y are the spatial variables and t is the variable corresponding to the frame number. The data is represented as:

$$\hat{d}(x,y;t) = \sum_{i=0}^{3}\sum_{j=0}^{3}\sum_{k=0}^{2} \alpha_{i,j,k} P_i(x) Q_j(y) R_k(t),$$

where P_i , for all i, Q_j, for all j, and R_k, for all k, are three sets of one dimensional polynomials. The temporal gradient at the point $x=0$, $y=0$, and $t=K-1$ is computed. For the results shown in Section 4.4, five image frames are used, corresponding to $K=5$.

4.3. An Algorithm for Estimating Displacement Vectors

Estimation based on multiple observations is frequently used to improve the accuracy. Since the estimation is of an inverse type, the discontinuities in the estimated values are not necessarily introduced by the noisy input data. Moreover, the gradient estimates used in Equation (4.2) are themselves estimates based on numerical differentiation of the input images, a well-known ill-posed procedure [Tor86] that is highly sensitive to noise. Hence spatially smoothing the image data will not improve the results [Sch85]. Along this line of reasoning, since the instability is largely introduced by the inverse operation in the form of a direct division in Equation (4.3), smoothing the estimated gradients and then computing the pointwise magnitude of the velocity component along the local gradient using Equation (4.3) will not be satisfactory, either. Directly smoothing the values computed from Equation (4.3) only degrades the resolution of the estimates without improving the accuracy.

Equation (4.2) can be viewed as a line in a two dimensional v_x-v_y space. Neighboring pixels with different gradient values will have lines intersecting at

the same point, yielding an estimate of velocity. Alternatively, Equation (4.2) represents a plane, parameterized by v_x and v_y, in the three dimensional observation space formed by I_x, I_y, and I_t. The approach presented in this section is that the estimation problem can be viewed as fitting a plane in this three dimensional observation space.

By observing N points, a system of linear equations can be formed:

$$\begin{aligned} I_x^1 v_x + I_y^1 v_y &= -I_t^1, \\ &\vdots \\ I_x^N v_x + I_y^N v_y &= -I_t^N. \end{aligned} \tag{4.5}$$

Writing Equation (4.5) in matrix notation, the problem becomes estimating $\vec{v}$ from

$$\left[\vec{I_x} \;\; \vec{I_y} \right] \vec{v} = \vec{I_t}, \tag{4.6}$$

where $\vec{v} = (v_x \;\; v_y)^T$.

The values of I_x, I_y, and $-I_t$ for the ith observed point correspond to the ith elements of $\vec{I_x}$, $\vec{I_y}$, and $\vec{I_t}$, respectively. Obviously all N points will have to have the same velocity. While this imposes a constraint on the resolution of the estimated vector, the accuracy will be improved.

The motivation for this formulation is that, except in the vicinity of a motion boundary, the velocity is assumed to vary smoothly in a local region. Suppose the velocity of the ith observed point, $\vec{v_i}$, is in fact $\vec{v} + \vec{d_i}$. The assumption is that there is a *nominal* velocity, $\vec{v}$, that is prevalent in the region and that the ith observed point has a *deviation* component $\vec{d_i}$. Consider the vector $\vec{d}$ where the ith element is $I_x^i d_x^i + I_y^i d_y^i$; i.e., the component of the deviation of the ith observed point along the local gradient direction. Estimating the nominal velocity from Equation (4.6) based on minimizing

$$\left\| \left[\vec{I_x} \;\; \vec{I_y} \right] \vec{v} - \vec{I_t} \right\|$$

is equivalent to minimizing $\|\vec{d}\|$, the length of $\vec{d}$.

The solution in this formulation will yield the correct velocity if the scene object is undergoing two dimensional rigid body motion, since the velocity deviation is zero for every point. Furthermore, if more information about the variation were available, such as knowing that the variation is along a certain

direction, the problem can be formulated accordingly. In the absence of such information, the general case is to minimize the mean squared norm of $\vec{d}$.

The "true" velocity can be recovered from Equation (4.6) if there are more than one gradient directions among the N observed points. To increase the likelihood of including points with different gradient directions, and to increase the number of observations to decrease the sensitivity to noise, N should be large. However, when N is overly large, the assumption that all the points have a prevalent nominal velocity will be less likely to hold. Equivalently, $\|\vec{d}\|$ will be large. Hence, the tradeoff is again between resolution and accuracy. When N is large, the resolution is poor, but the overall accuracy is better while ambiguities will be less.

A solution for v_x and v_y from the set of equations in Equation (4.5) requires N to be 2. When the data are observed in noisy environment, N has to be much larger, resulting in a set of overdetermined linear equations. An exact solution does not exist for overdetermined linear equations; a classical solution is by the least squares method. Using the conventional least squares method to solve the image flow problem, such as in [Mar86], only corrects the errors in $\vec{I_t}$ and does not account for errors in $\vec{I_x}$ and $\vec{I_y}$, as will be discussed in the next section. Noting that all three gradients are themselves estimated noisy values, an improved estimation procedure for $\vec{v}$ is via the Total Least Squares method [Gol83, Zol87]. In the Section 4.3.1, key ideas of the Total Least Squares method are summarized.

4.3.1. Total Least Squares Method

In this section, we shall be concerned with fitting a two dimensional plane to N observed points in a three-dimensional space. Throughout this section, we shall denote matrices **G** and **M** as:

$$\mathbf{G} = \left[\vec{I_x}\ \vec{I_y}\right],$$

and

$$\mathbf{M} = \left[\vec{I_x}\ \vec{I_y}\ \vec{I_t}\right].$$

G is a N by 2 matrix, while **M** is a N by 3 matrix. Equation (4.6) can be rewritten as $\mathbf{G}\vec{v}=\vec{I_t}$. A solution to this set of equations exists if **G** is of full rank.

Conventional Least Squares (*LS*) method, in solving for $\vec{v}$ from Equation (4.5), projects $\vec{I_t}$ onto the space spanned by $\vec{I_x}$ and $\vec{I_y}$; i.e., $\vec{I_t}$ is forced to be a

linear combination of $\vec{I_x}$ and $\vec{I_y}$. This can be seen by noting that the derivation leading to the LS solution can be viewed as applying to both sides of Equation (4.6) a projection operator $\mathbf{P}_G$, where $\mathbf{P}_G$ is such that $\mathbf{P}_G\mathbf{G} = \mathbf{G}$. This has the effect of projecting the observed $\vec{I_t}$ onto the I_x-I_y plane before the pseudo-inverse operator $\mathbf{G}^+ = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T$ is applied to obtain the LS solution.

Fitting a plane to the N observed points by the LS method only minimizes the distance along axes that are parallel to the I_t axis. This is suboptimal when all three gradient components used in Equation (4.6) are prone to errors and noise. A "best" fit should be based on minimizing the sum of the shortest distances from every point to the fitted plane. A standard geometrical view of the fitting problem clarifies the motivation for taking a Total Least Squares (*TLS*) approach. Figure 4.3 shows the view for the case of fitting a line to points in two dimensional space. An LS solution in this case minimizes the *vertical* distance—the distance measured along lines parallel to the y-axis from a point to the fitted line. This is acceptable if the x-coordinates of the observed points are noiseless; i.e., noise only perturbs the observations by displacing the points along the vertical direction. When noise can displace the observed points in both directions, the appropriate procedure to fit the line is to find a line that minimizes the perpendicular distances from observed points to the fitted line.

Conceptually, a TLS solution that fits $\mathbf{M}$ is sought by determining a projection operator $\mathbf{P}$ that projects onto the "best" two dimensional subspace. This criterion can be expressed as determining $\mathbf{P}$ such that the following is maximized:

$$\|\mathbf{P}\vec{I_x}\|^2 + \|\mathbf{P}\vec{I_y}\|^2 + \|\mathbf{P}\vec{I_t}\|^2.$$

The solution of $\mathbf{P}$ is given by [Zol87]

$$\mathbf{P} = \left[\vec{u_1}\ \ \vec{u_2}\right],$$

where $\vec{u_1}$ and $\vec{u_2}$ are the two eigenvectors of $\mathbf{M}\mathbf{M}^T$ associated with the two largest eigenvalues. The TLS solution is then given by

$$\vec{v_{TLS}} = (\mathbf{G}^T\mathbf{P}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{P}\vec{I_t}.$$

Upon simplification,

$$\vec{v_{TLS}} = (\mathbf{G}^T\mathbf{G} - \sigma^2\mathbf{I}_{22})^{-1}\mathbf{G}^T\vec{I_t}, \tag{4.7}$$

where $\mathbf{I}_{22}$ is a 2×2 identity matrix and σ^2 is the smallest eigenvalue of the

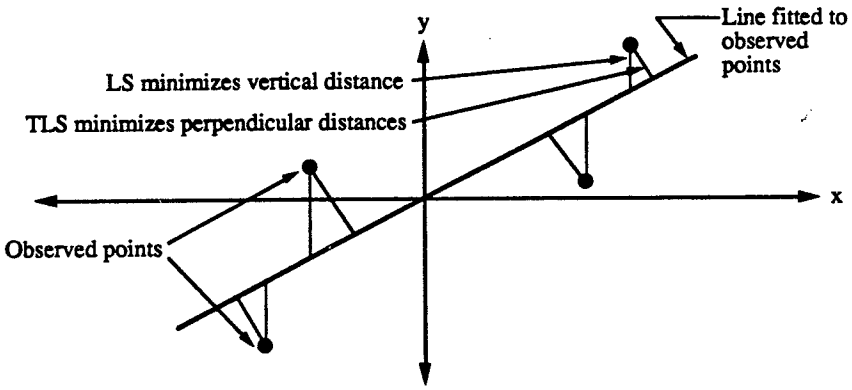


Figure 4.3. Two dimensional view of TLS method compared to LS method.

matrix $\mathbf{M}^T\mathbf{M}$.

4.3.2. Application of the TLS Method to Estimating Displacement Vector

It is interesting to compare Equation (4.7) with the LS solution to Equation (4.6). The LS solution is

$$\vec{v_{LS}} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\vec{I_t}. \tag{4.8}$$

Hence, the TLS solution improves on the LS solution by a "correcting" term, σ^2, which can be shown to be the minimum average magnitude of the spatiotemporal directional gradient as follows. Let $\vec{\alpha}$ be a directional unit vector in the three dimensional (spatiotemporal) space. Consider a constrained minimization problem:

$$\text{find } \vec{\alpha} \text{ that minimizes } \vec{\alpha}^T\mathbf{M}^T\mathbf{M}\vec{\alpha} \text{ subject to } \|\vec{\alpha}\| = 1. \tag{4.9}$$

Using the Lagrange multiplier technique, Equation (4.9) can be rewritten as

$$\text{find } \vec{\alpha} \text{ that minimizes } \vec{\alpha}^T\mathbf{M}^T\mathbf{M}\vec{\alpha} + \lambda(1-\vec{\alpha}^T\vec{\alpha}) \tag{4.10}$$

where λ is the Lagrange multiplier. The solution to Equation (4.10) is

$$\mathbf{M}^T\mathbf{M}\,\vec{\alpha} = \lambda\vec{\alpha}. \tag{4.11}$$

Hence it can be seen from Equation (4.11) that the Lagrange multiplier λ is in fact the smallest eigenvalue of the matrix $\mathbf{M}^T\mathbf{M}$, or $\lambda = \sigma^2$. Furthermore, recall that the associated eigenvector $\vec{\alpha}$ is a directional vector. Consider the matrix $\mathbf{M}^T\mathbf{M}$:

$$\mathbf{M}^T\mathbf{M} = \begin{bmatrix} \vec{I_x}\cdot\vec{I_x} & \vec{I_y}\cdot\vec{I_x} & \vec{I_t}\cdot\vec{I_x} \\ \vec{I_x}\cdot\vec{I_y} & \vec{I_y}\cdot\vec{I_y} & \vec{I_t}\cdot\vec{I_y} \\ \vec{I_x}\cdot\vec{I_t} & \vec{I_y}\cdot\vec{I_t} & \vec{I_t}\cdot\vec{I_t} \end{bmatrix}.$$

The quantity that is to be minimized in Equation (4.9), $\vec{\alpha}^T\mathbf{M}^T\mathbf{M}\vec{\alpha}$, can be expressed as

$$\sum_{i=1}^{N}(\vec{\alpha}^T\cdot\nabla_{st}^i I)^2 \tag{4.12}$$

where ∇_{st}^i represents the three dimensional spatiotemporal gradient operator at the ith observed point. Equation (4.12) then represents the average magnitude of the spatiotemporal gradient along the direction of $\vec{\alpha}$. Hence σ^2 is the minimum average magnitude among all directional spatiotemporal

gradients.

4.3.3. Rank Deficient Cases

Since **G** has only two columns, it has either rank 0 or rank 1 if it does not have full rank. When it has rank 0, it corresponds to the case where the spatial gradient is zero in the neighborhood. Thus it is not surprising that no matching can be done.

When **G** has rank 1, the neighborhood being considered contains an ideal edge. This can be seen by noting that since $\vec{I_y} = c\vec{I_x}$, $I_y^i = cI_x^i$ for some constant c and for all i, $i = 1,2,...,N$. Then

$$\tan\theta_i = \frac{I_y^i}{I_x^i} = c, \text{ for all } i,$$

where θ_i is the gradient direction of the ith observed point. Hence, the gradient direction for every observed point is the same.

The minimum norm solution is given by:

$$\vec{v_{MTLS}} = \alpha(v_{11}\ v_{12})^T, \tag{4.8}$$

where $\alpha = -v_{13}/(1-v_{13}^2)$, and

$$\vec{v_1} = (v_{11}\ v_{12}\ v_{13})^T$$

is the right singular vector associated with the largest singular value of **M**. The minimum norm result is the $\vec{v}\cdot\nabla I$ component.

4.3.4. Measure of Fitting

As discussed in Section 4.3, using the image flow approach, a match is implicitly assumed and a numerical value is computed regardless of whether a match is correct. Hence it is reasonable to evaluate the computed values by determining the closeness of the TLS fit. One measure of the fit is by computing the sum of distances of each observed point (in the spatiotemporal gradient space) to the plane determined by the computed v_x-v_y values. By noting that the vector $(v_x\ v_y\ 1)^T$ is a normal vector of the plane, the distance of any point (I_x^i, I_y^i, I_t^i) to that plane is given by: $I_x^i v_x + I_y^i v_y + I_t^i$.

4.4. Experimental Results

Test Images

Different types of test images have been used to test the image flow algorithms; examples are images with ramp and jump edges [Mar86], geometric figures [Hil83], and sinusoidal gratings [Jac87]. A test image containing two sinusoidals is used here to demonstrate the efficacy of the method described in the previous section to compute the true velocity. The images are noiseless, hence any input error to the velocity estimation procedure originates from the gradient estimators. The test image has size 64×64, and has intensity values $I(m,n)$ defined by:

$$I(m,n) = k + A\ \cos(2\pi f_x(m+\theta_x) + 2\pi f_y\ (n+\theta_y))$$

$$+ A\ \cos(2\pi f_x(m+\theta_x) - 2\pi f_y\ (n+\theta_y)).$$

k and A are constants; f_x and f_y are functions of the spatial frequencies of the sinusoidals and the spatial sampling rates; θ_x and θ_y are the velocity components that are to be recovered, assuming unit temporal sampling period. The top row of Figure 4.4 shows a test sequence of five images; the parameters used are:

f_x=4/50; f_y=9/50;θ_x=3;θ_y=2;k=A=60.

A fitting neighborhood of 7×7 was used in the TLS algorithm, and the velocity map is shown in the center row of Figure 4.4. The bottom row of Figure 4.4 shows the velocity map superimposed on the original five images.

The same image sequence is used to demonstrate the advantage of using the TLS fitting compared to LS fitting. Figures 4.5 and 4.6 show the histograms of the estimated velocities using TLS and LS fitting, respectively. The results estimated by both methods exhibit some spreading out effect, but the TLS results are clustered around the correct values, whereas the LS results are biased to lower values.

The test sequence is also used to examine the effect of varying the fitting neighborhood size. Since there is a unique motion parameter in the image, increasing the neighborhood should always improve the performance. The results for estimating v_y corresponding to neighborhood sizes of 9×9, 13×13, 17×17, and 19×19 are shown in Figure 4.7. As expected, the estimated results cluster sharpens as the neighborhood size is increased.

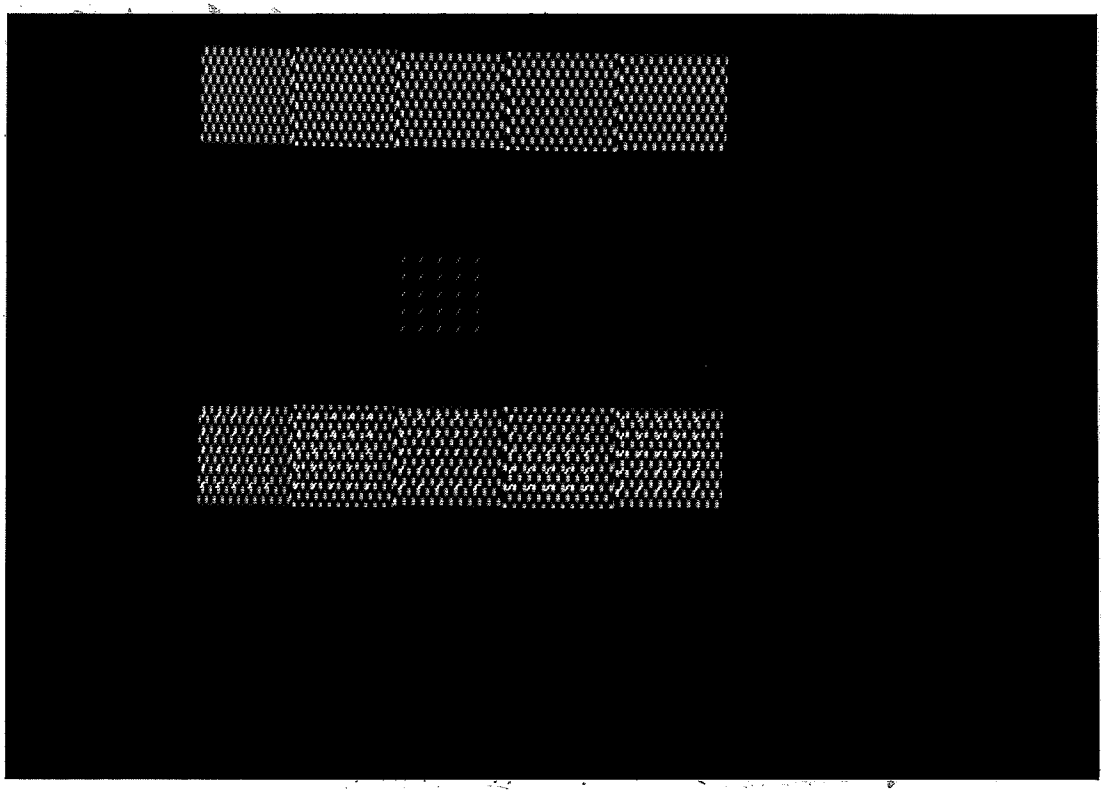


Figure 4.4. Image velocity estimation using test image sequence.

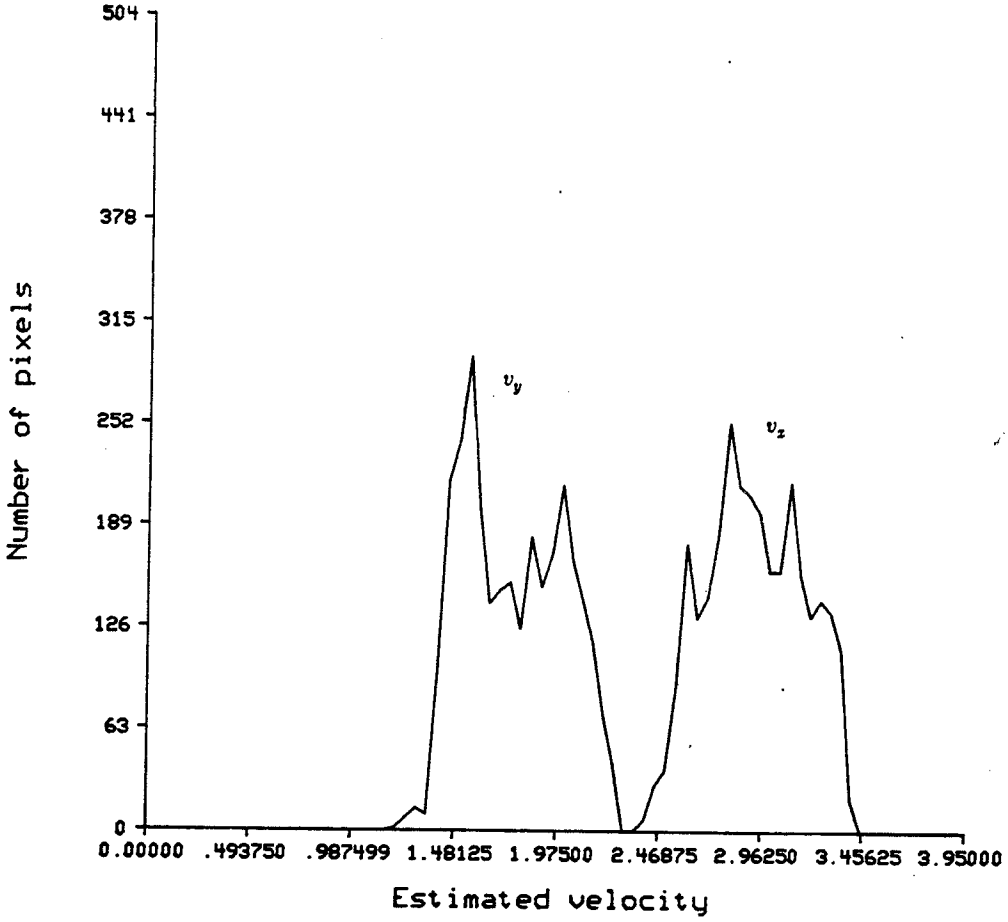


Figure 4.5. Histogram of the estimated velocities using TLS fitting.

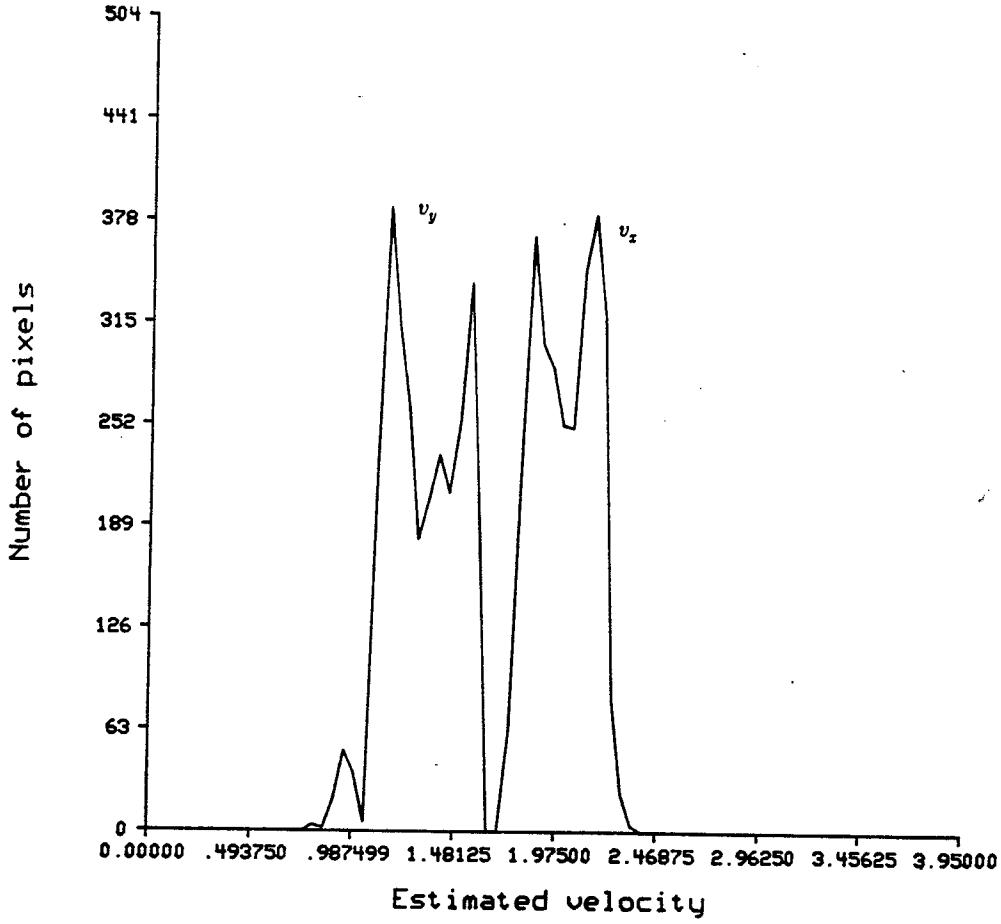


Figure 4.6. Histogram of the estimated velocities using LS fitting.

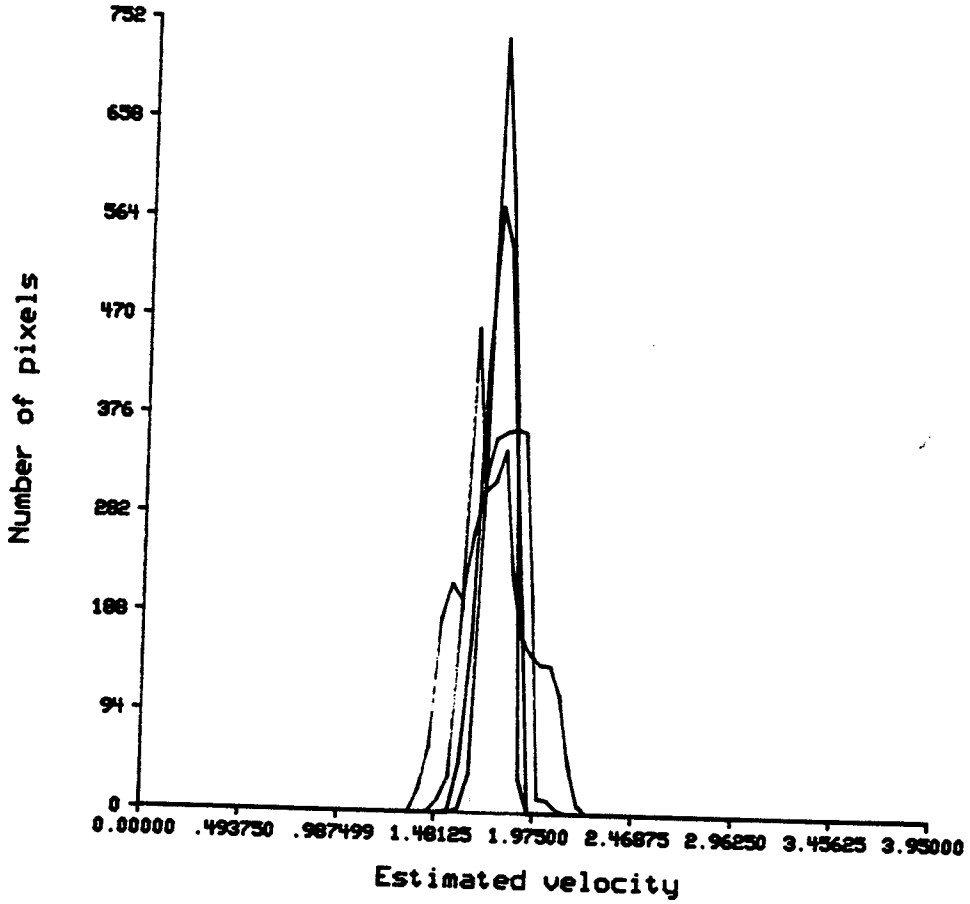


Figure 4.7. Histogram of the estimated velocities using TLS fitting and different neighborhood sizes.

Real Images

A sequence of real echocardiographic images with synthesized motion is next used to test the algorithm. With a known motion parameter, the abilities of both the gradient estimators and the TLS algorithm to handle real images are tested. The synthesized motion is pure translation along the diagonal from the lower right hand corner to the top left corner. The original sequence is shown in the top row of Figure 4.8. The images are first smoothed by a 21×21 Gaussian shaped filter, followed by the gradient estimators. The resulting velocity map is shown in Figure 4.8. Some mistakes are made, notably in the regions where the spatial gradients are low.

The algorithm is also applied to a real sequence of echocardiograms depicting the left ventricular during diastole. The original sequence is shown in the top row of Figure 4.9. The images are smoothed by a 35×35 Gaussian shaped filter, followed by the gradient estimators. The resulting velocity map is shown in Figure 4.9 and overlaid onto the original images at the bottom row. Since the motion pattern in the image sequence is quite complicated, it is difficult to judge the results. The diastole effect is not apparent, however, the indicated velocities do seem to agree with the intensity variations.

4.5. Concluding Remarks

The problem of estimating displacement vectors from an image sequence using the image flow approach was examined. Pointwise estimation results are typically badly corrupted by noise, which limits their utility in subsequent processing. The estimation result at each point will be more reliable if it were based on more observed points. By assuming that the velocity field does not change abruptly, an overdetermined set of linear equations is formed to relate the intensity changes in a local neighborhood. Conventional Least Squares method that has been used to solve for the local velocity does not account for all of the errors in the estimated gradient values. A Total Least Squares method is described that will provide a more accurate estimation. The primary issue addressed in this chapter is that of *initial* velocity estimation. In certain areas of the image plane, especially in the absence of gradients in one or more directions, the estimation is less reliable.

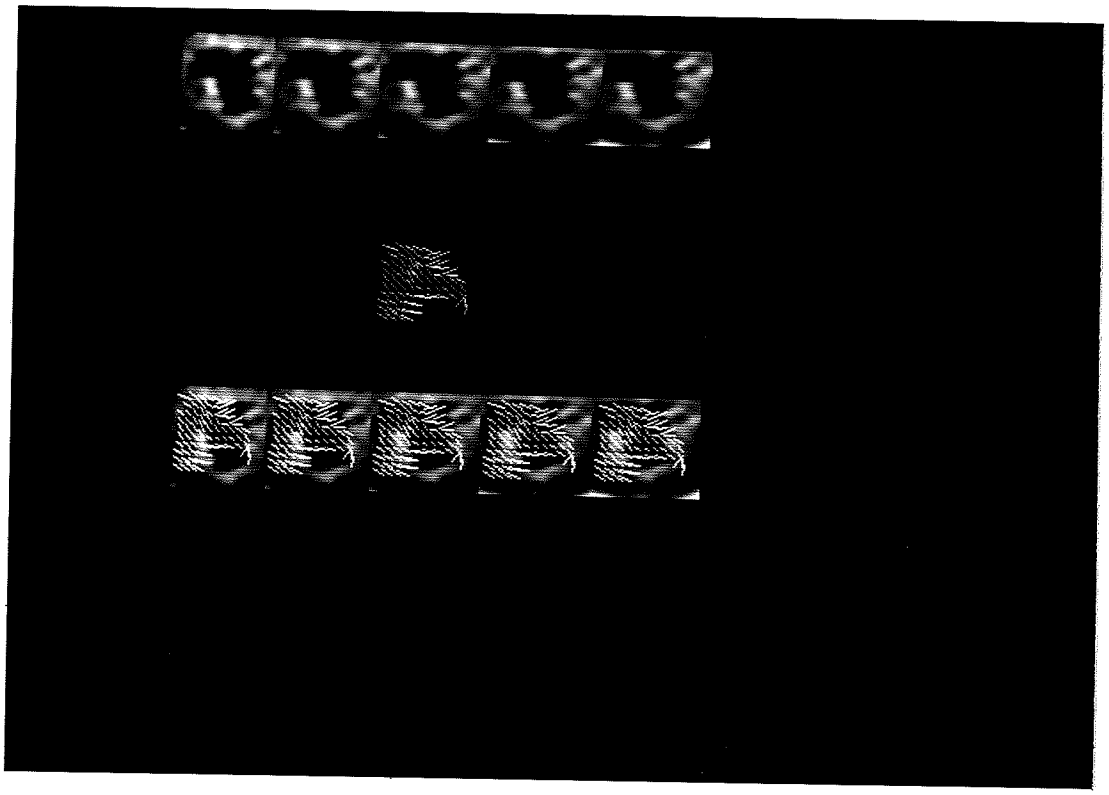


Figure 4.8. Estimated velocities of an echocardiographic image sequence with synthesized motion.

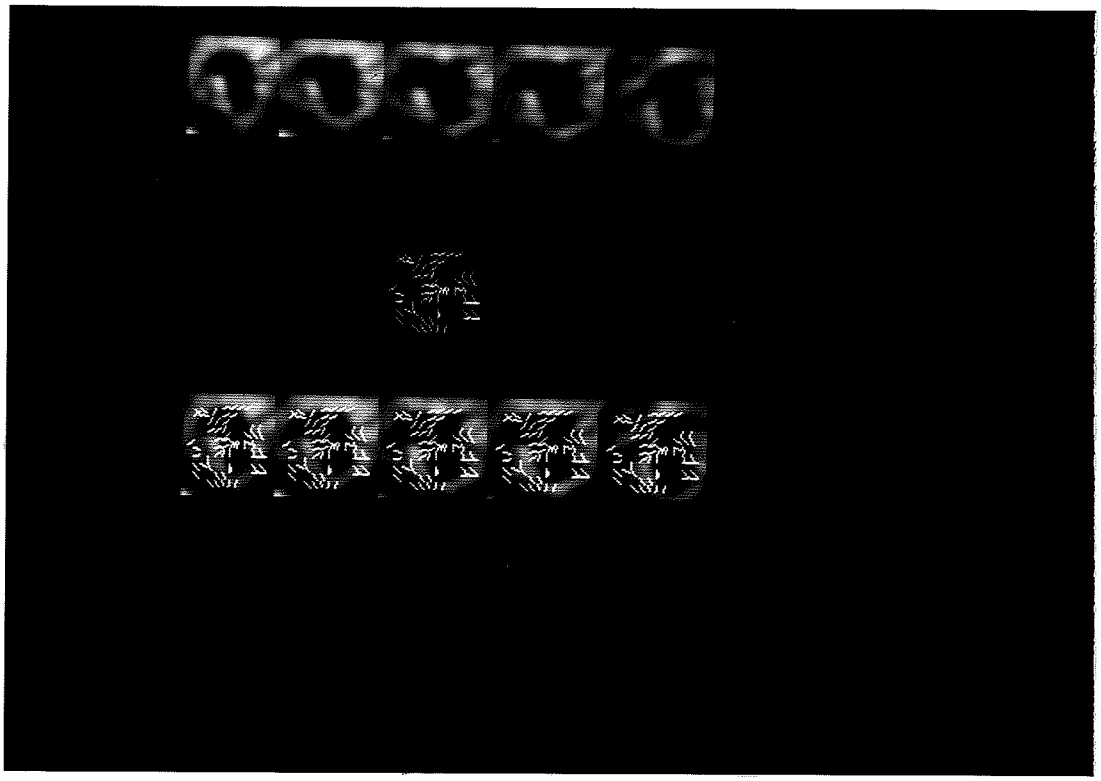


Figure 4.9. Estimated velocities of a real echocardiographic image sequence.

CHAPTER 5
ELECTROCARDIOGRAM SIGNAL PROCESSING

5.1. Introduction

Except in a very well controlled environment, knowledge used in a computer vision system cannot always be specified in the form of static a priori knowledge. Information extracted by other sensors or from other related domains are often useful to the extraction of information from images. An important source of information in understanding an echocardiogram is knowing the position of its acquisition time in the cardiac cycle. For example, in echo image sequence analysis, knowing that the heart is going from diastole to systole allows the direction of the heart wall motion to be predicted or verified. Electrocardiogram (*EKG*) signals, which provide important timing information for cardiac imaging, are measures of the changes in the electrical potential of the heart muscle. They are frequently used to synchronize the acquisition of digitized echocardiogram sequences. "Gating" the echocardiograms, or knowing that certain images are obtained at the same point in the cardiac cycle, allows those images to be compared, or to be averaged for reducing the noise in the images [Col86]. Furthermore, knowing the cardiac rhythm allows a computer vision system to determine the approximate range of image velocity.

A new approach to impulsive noise suppression and background normalization of digitized EKG signals is presented in this chapter. EKG signals are frequently plagued by impulsive noise, e.g., due to muscle activities and power line interference [Moo84]. Moreover, background normalization is needed to correct the baseline drift of the signal caused by the respiration and motion of the subject [Ahl85]. An EKG signal consists of a number of lobes, or *waves*, corresponding to the polarization actions of the heart muscle. The most prominent waves are labeled as the Q, R, and S waves, and together they form the *QRS complex*. Noise suppression is typically the first step performed in the processing of EKG signals [Pah87]. It is important to limit the distortion of the EKG signal by noise suppression algorithms before such tasks

as QRS detection or temporal alignment. The main objective in our processing is to produce an output that can facilitate detection of the QRS waves; hence, of primary importance in the processing is to preserve the main QRS complex. Other applications include real-time processing of EKG signals acquired in "hostile" environments such as in ambulances or on board spacecraft. We note that issues such as artifact suppression and preserving the subtle notches and slurs in individual waves are significant for clinical diagnostics use.

The most common approach to noise suppression is by low-pass filtering [War70, Pah84], which is ineffective for reducing impulsive noise. Since the baseline drift is assumed to have relatively low frequency, baseline correction is typically performed by high-pass filtering the EKG data [Als85]. Effective alternatives to conventional linear filtering, particularly when dealing with impulsive noise, are nonlinear operators such as median filtering [Gal81] or other ranked ordering methods [Wen86].

This chapter presents new algorithms that make use of a class of nonlinear signal processing operators, known as mathematical morphology, for processing EKG data. Morphological operators have been used in the field of image processing and are known for their robust performance in preserving the shape of a signal while suppressing noise [Mar87a]. An introduction to the morphological operators is given in Section 5.2. The new algorithm is described in Section 5.3. Experimental results addressing various aspects of the problem are presented in Sections 5.4 to 5.8.

5.2. Morphological Operators

Mathematical morphology, which is based on set operations, provides an approach to the development of nonlinear signal processing operators that incorporate the shape information of a signal [Ser82]. In mathematical morphological operations, the result of a set transformed by another set depends on the shapes of the two sets involved. The *shape* of a signal is determined by the values that the signal takes on. The shape information of a signal is extracted by using a *structuring element* to operate on a data sequence.

Morphological operators were developed in the image processing field for machine vision and medical imaging applications [Har87]. Operators that are used to process the EKG data for noise suppression and background

normalization as described in this chapter are known in the image processing literature as gray-scale morphological operators [Har87]. Morphological operators have been used in a limited way for background normalization of biological signals [SkB85]. There are two basic morphological operators: erosion and dilation. These operators are usually applied in tandem; opening and closing are two derived operators that are defined in terms of erosion and dilation. These operators are described in detail below.

5.2.1. Erosion and Dilation

Throughout this section, f and k denote two discrete functions defined on $F = \{\ 0, 1, \ldots, N-1\ \}$ and $K = \{\ 0, 1, \ldots, M-1\ \}$, respectively; i.e., $f : F \rightarrow \mathbf{I}$ and $k : K \rightarrow \mathbf{I}$, where $\mathbf{I}$ denotes the set of integers. It is further assumed that $N > M$.

The *erosion*[†] of a function f by another function k, which we shall call the structuring element, denoted $f \ominus k$, is defined as

$$(f \ominus k)(m) = \min_{n=0,\ldots,M-1} f(m+n) - k(n),$$

$$\text{for } m = 0, \ldots, N-M.$$

Erosion is a "shrinking" operator in that values of $f \ominus k$ are always less than those of f. To determine the value of $f \ominus k$ at a point m, the procedure is to:

(1) translate the structuring element to m,

(2) subtract the structuring element from the input sequence, and

(3) find the minimum value of the differences.

An example of erosion is shown in Figure 5.1. The structuring element has a length of 3 and a constant value of 1. The original data consists of a sinusoidal signal corrupted by unwanted impulsive notches. The data is shown marked with squares; the result after erosion is shown marked with circles.

The *dilation*[‡] of f by k, $f \oplus k$, is defined as

$$(f \oplus k)(m) = \max_{n=m-M+1,\ldots,m} f(n) + k(m-n),$$

[†] This definition of erosion is commonly called grayscale erosion in the literature.
[‡] This definition of dilation is commonly called grayscale dilation in the literature.

$$\text{for } m = M-1, M, \ldots, N-1.$$

The dilation operation is an "expansion" operation in that the values of $f \oplus k$ are always greater than those of f. The procedure to determine the value of $f \oplus k$ at m is to

(1) left-right reverse the structuring element k,

(2) translate the reversed structuring element to m,

(3) add the reversed structuring element to the input sequence, and

(4) find the maximum value of the sums.

An example of dilation is shown in Figure 5.2. The structuring element has a length of 3 and constant values of 1. The original signal is shown marked with circles. The result after dilation is shown marked with triangles.

The complexity of an erosion or a dilation is comparable to that of discrete convolution. The role of a structuring element is analogous to that of the window kernel of a convolution. Within the window defined by the structuring element, instead of performing a pointwise multiplication, pointwise subtraction or addition is performed. The resulting value for that window is determined by a minimization or maximization instead of a summation.

5.2.2. Opening and Closing

The two basic operations, erosion and dilation, are usually applied in tandem. Opening and closing are two operations defined in terms of the basic operations. *Opening* of a data sequence by a structuring element is defined as erosion followed by a dilation. *Closing* of a data sequence by a structuring element is defined as dilation followed by an erosion. The opening of a data sequence can be interpreted as sliding the structuring element along the data sequence from beneath and the result is the highest points reached by any part of the structuring element. Similarly, the closing of a data sequence can be interpreted as sliding a "flipped-over" version of the structuring element along the data sequence from above and the result is the lowest points reached by any part of the structuring element.

It can be seen then that the shape of the output of either opening or closing is affected by the shape of the structuring element. Depending on the shape characteristics of the signal that is to be preserved, a specific structuring element has to be designed for processing the data. In most applications, opening is used to suppress peaks while closing is used to suppress pits. For

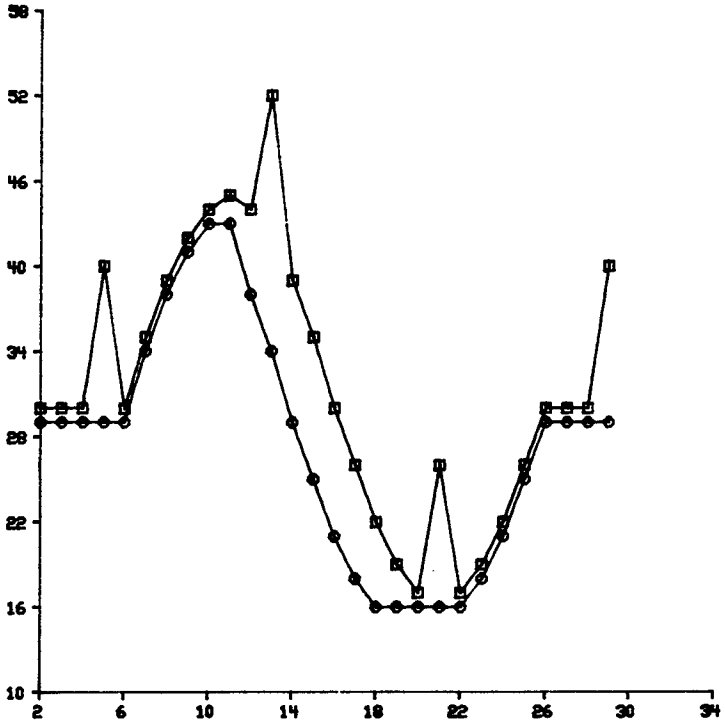
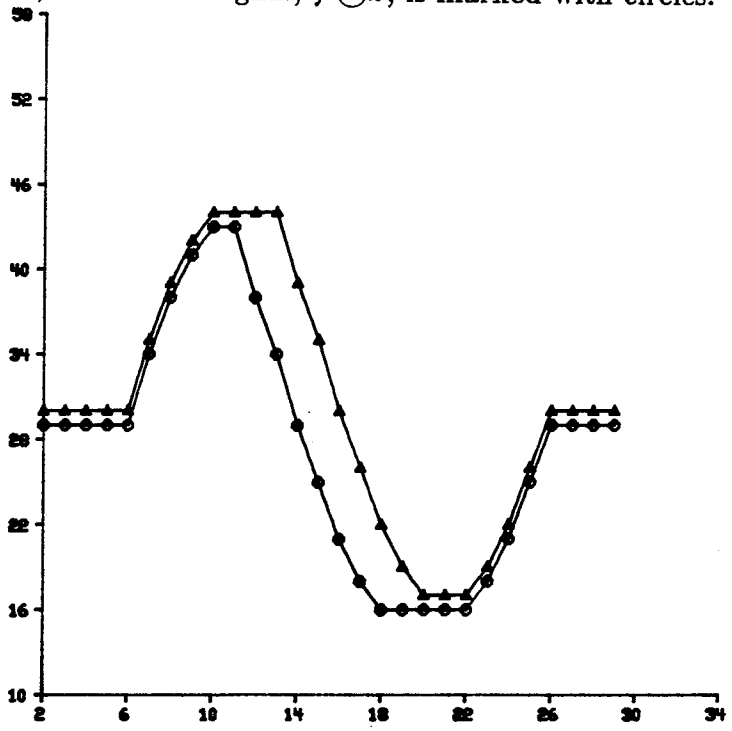


Figure 5.1. Example of erosion. The structuring element, k, has length 3 and constant values of 1. The original signal, f, is marked with squares; the eroded signal, $f \ominus k$, is marked with circles.



Figures 5.2. Example of dilation. The structuring element, k, has length 3 and constant values of 1. The original signal, f', is marked with circles; the dilated signal, $f' \oplus k$, is marked with triangles.

example, the result of opening any sequence with a structuring element that is flat and has a length of M will not contain any peak within any interval of length $M-1$; while the result of closing any sequence with such a structuring element will not contain any pit within any interval of length $M-1$.

Noting that an opening operation is an erosion followed by a dilation, the examples of erosion and dilation described above can be seen as the two steps that make up an opening operation example. The original data sequence formed by a sinusoidal signal corrupted by impulsive noise is shown in Figure 5.3 marked with squares. The data is shown in Figure 5.3 marked with squares and is eroded and dilated by the same structuring element. The partial result after erosion is shown in Figure 5.1. The result after opening, with the spurious peaks suppressed, is shown in Figure 5.3 marked with triangles. Figure 5.4 shows an example of applying the closing operator to a signal. The original data is shown marked with squares while the result after closing is shown marked with triangles. It can be seen that the negative impulsive peaks are removed by the closing operation.

5.3. A New Algorithm

The algorithm uses two steps to process the EKG signal: (1) impulsive noise suppression, and (2) background normalization. The overview of the algorithm is shown in Figure 5.5.

Impulsive noise suppression is performed by processing the data through a sequence of opening and closing operators. The algorithm for noise suppression is shown in Figure 5.6. The EKG signal, as well as any baseline drift, is estimated by processing the data using an opening operator followed by a closing operator. A second estimate of the signal is formed by processing the data using a closing operator followed by an opening operator. The result from this step is the average of the two estimates. If the amount of processing is a concern, either one branch of the block diagram can be deleted with some performance degradation.

The design of the structuring element depends on the shape of the signal that is to be preserved. Since the opening and closing operations are intended to remove impulses, the structuring element must be designed so that the waves in the EKG signal are not removed by the process. A structuring element is characterized by its "shape," width, and height. Its width, or length, is largely determined by the duration of the major waves and the

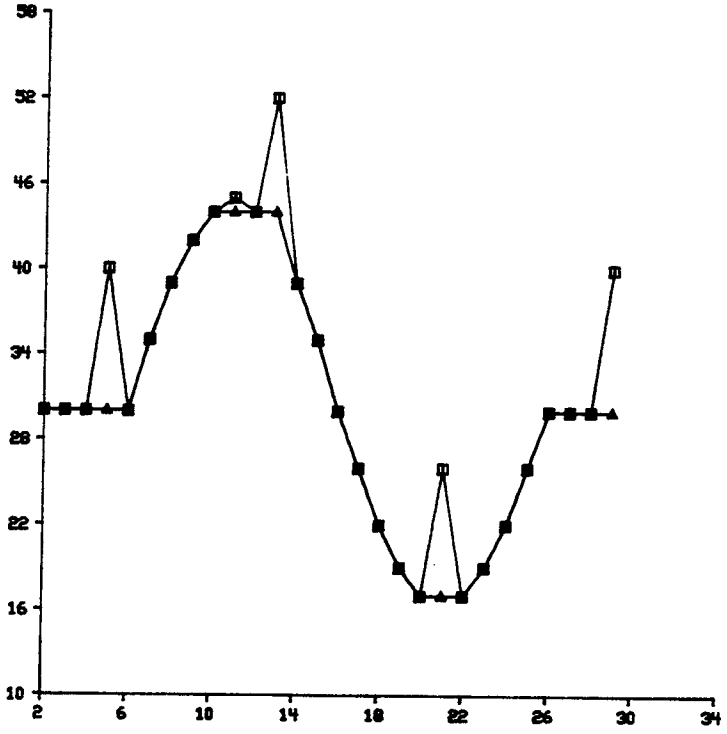
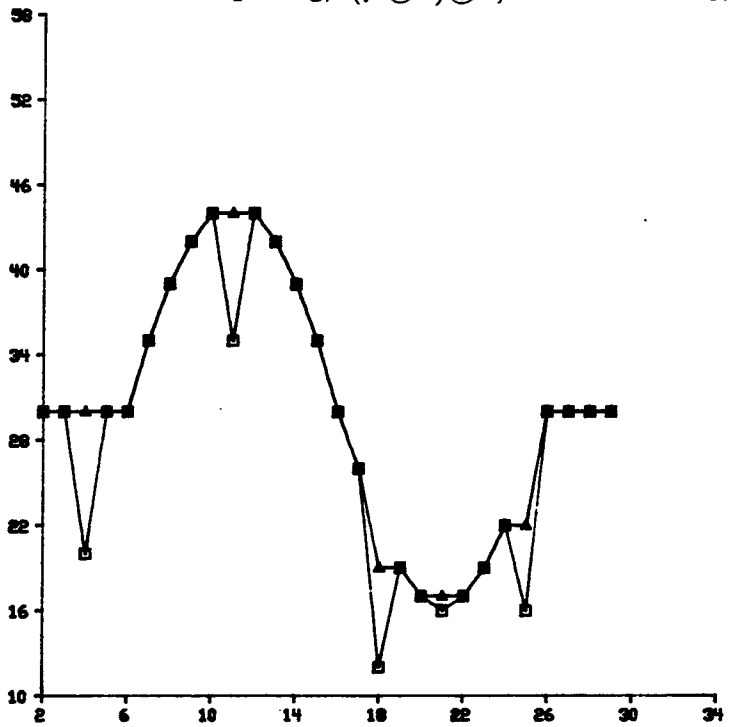


Figure 5.3. Example of opening. The structuring element, k, has length 3 and constant values of 1. The original signal, f, is marked with squares; the result after opening, $(f \ominus k) \oplus k$, is marked with triangles.



Figures 5.4. Example of closing. The structuring element, k, has length 3 and constant values of 1. The original signal, g, is marked with squares; the result after closing, $(g \oplus k) \ominus k$, is marked with triangles.

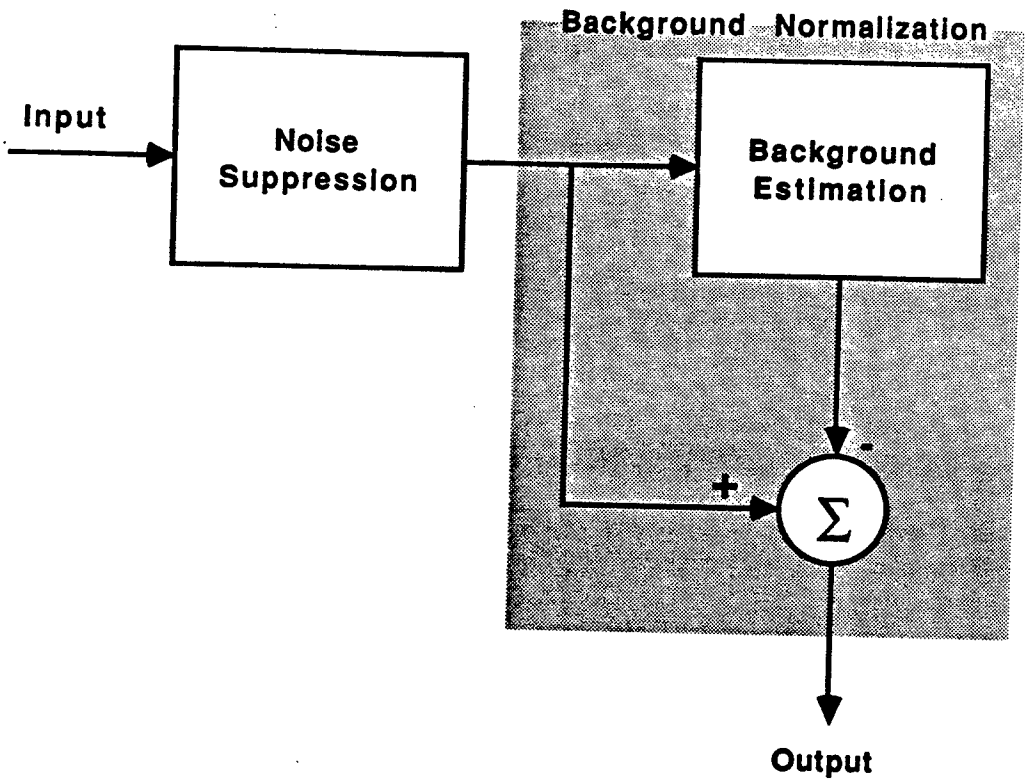


Figure 5.5. Overview of the algorithm for suppressing impulsive noise and normalizing background drift.

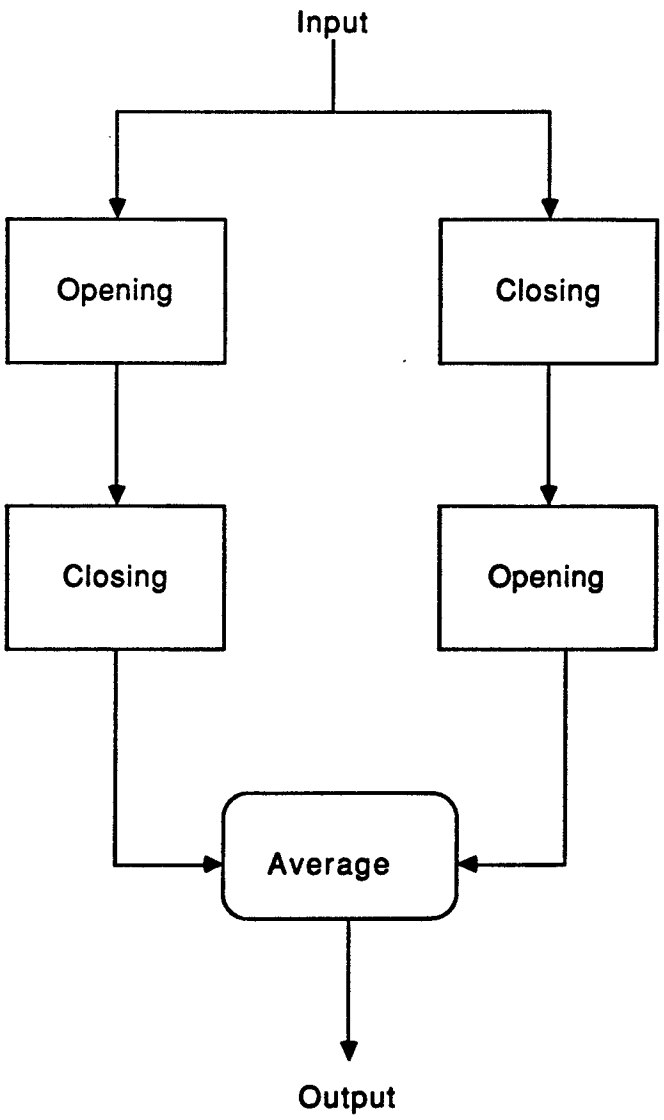


Figure 5.6. Block diagram of the impulsive noise suppression algorithm.

sampling rate. Denoting the duration of one of the waves as T sec, and the sampling rate as S Hz, the number of samples that correspond to a wave is then $T \times S$. Thus, the length of the structuring element must be less than $T \times S$.

The values of the structuring element determines the shape of the output waveform. Since in practice the EKG signal is not an ideal deterministic signal, we can only loosely classify the shape of its waves as triangular or dome-like. Due to this imprecise classification, the structuring element in practice has to be a lot shorter than $T \times S$ samples. Many structuring element implementations with the same width and height can be classified as dome-like. In the next section, we shall use a dome-like structuring element model parameterized by its width, height, and "shape" to see how each of the parameters affects the performance of the algorithm.

Background normalization is performed by estimating the drift in the background and subtracting it from the incoming data. The algorithm for background normalization is shown in Figure 5.7. The background drift is estimated by removing the EKG signal from the data. The data is first opened by a structuring element that removes peaks which results in a pit where the EKG signal is located. This pit is removed by a closing operator using a larger structuring element. The result is then an estimate of the baseline drift.

In this step, two structuring elements are used: one for removing peaks and the other for removing the pit left after the previous operation. The design of the first structuring element is determined by the duration of the waves in the EKG signal. As in the previous discussion, denote the duration of one of the waves by T seconds, and the sampling rate by S Hz, the number of samples of a wave is $T \times S$. To remove the wave, a structuring element must have its length L greater than $T \times S$. The second structuring element is used to remove the pit left by the first operation, thus its length must be roughly $2L$.

A second estimate can be made by first closing the data by a structuring element which results in a hump where the EKG signal is located, followed by an opening operator using a larger structuring element. The two estimates are then averaged to form the baseline drift estimate. Correction of the baseline roll and drift is then done by subtracting the baseline drift estimate from the result obtained from the previous step.

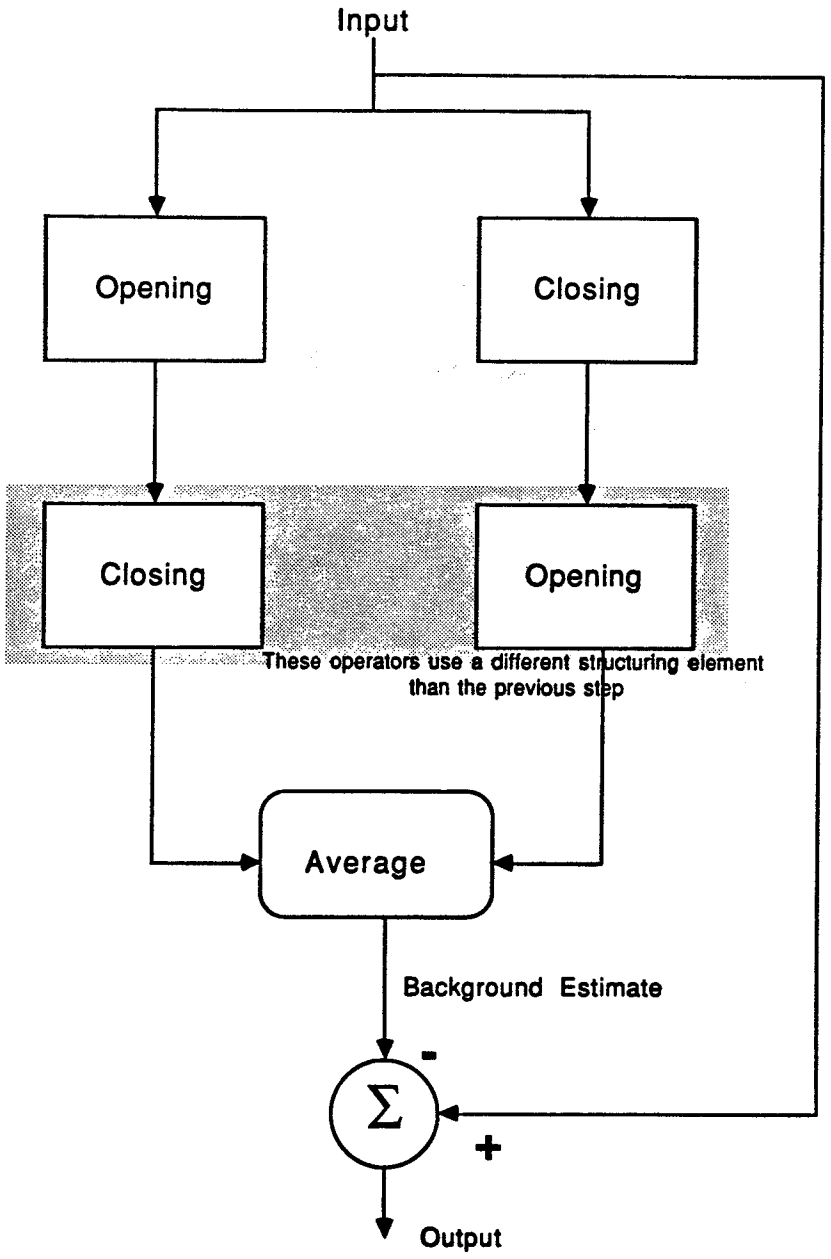


Figure 5.7. Block diagram of the background normalization algorithm.

The block diagram for the overall algorithm is shown in Figure 5.8.

5.4. Experiments with a Known Signal

A number of experiments were performed using test data formed by adding impulsive noise and baseline drift to a digitized signal from an analog EKG simulator. Using a corrupted known signal as test data allows the performance of the algorithm to be evaluated by comparing the recovered signal with the known signal. Moreover, the degree of corruption of the input data can be seen, in part, by considering the difference between the data before and after the corruption.

A noisy sequence of EKG data is modeled as

$$r(n) = s(n) + i(n) + b(n),$$

where $s(n)$ is the signal that includes the QRS waves, $i(n)$ is the noise component, and $b(n)$ is the baseline drift.

Signal

A known EKG signal was obtained from a FOGG Model M310 ECG Simulator that generates a signal with 5 mV amplitude. The signal was sampled at 1 kHz and quantized to 12 bits (i.e., the digitized sample values range from -2048 to 2047). The maximum and minimum values of the EKG signal are 1636 and -1722, respectively. Figure 5.9 shows a sequence of the digitized EKG signal with the heart rate at 100 beats per minute. The unit of the time-axis is 1/1000 seconds.

Noise

Impulsive noise is generated by an ϵ-mixture of Gaussian noise that has a probability distribution function of

$$P_i(y) = (1-\epsilon)\Phi(\frac{y}{\sigma_1}) + \epsilon\Phi(\frac{y}{\sigma_2})$$

where $\Phi(y)$ is the probability distribution function of a Gaussian random variable with zero mean and unit variance. σ_2 is typically much larger than σ_1. With probability $(1-\epsilon)$, the added noise for a sample, $i(n)$, is a Gaussian random variable with standard deviation σ_1, which simulates the background noise; with probability ϵ, $i(n)$ is Gaussian with standard deviation σ_2, which

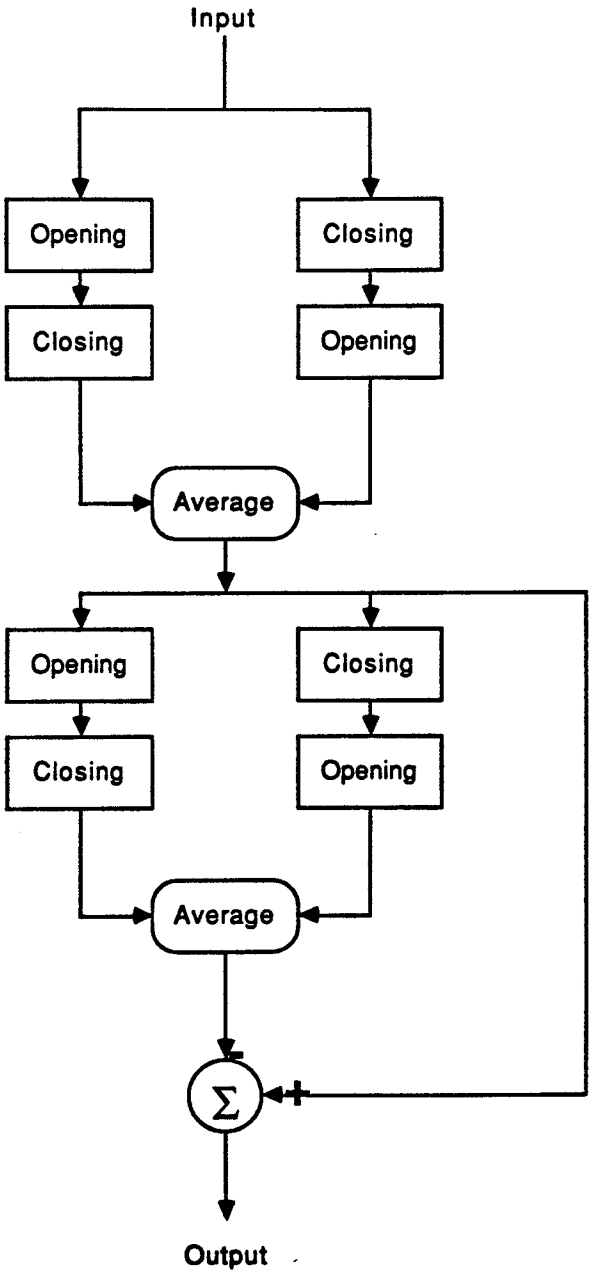


Figure 5.8. Block diagram of the overall algorithm for suppressing impulsive noise and normalizing background drift.

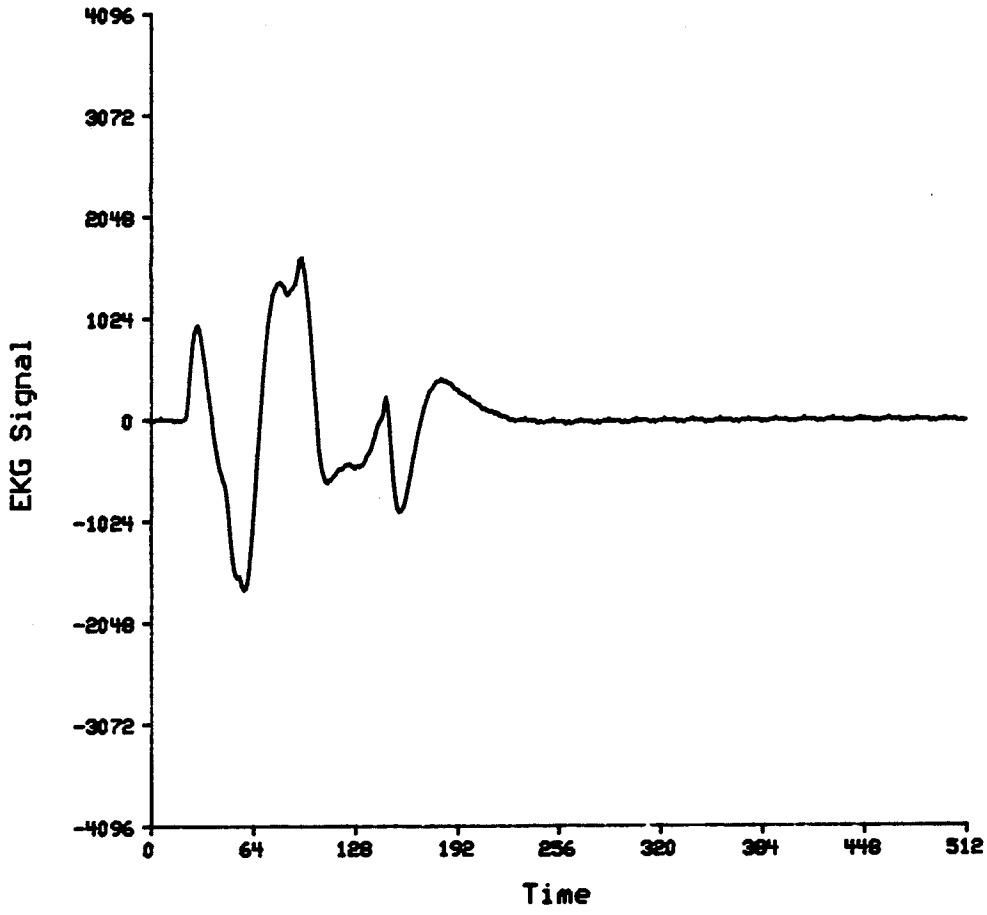


Figure 5.9. A digitized EKG signal sequence from an analog EKG simulator.

simulates the impulsive noise. As σ_1 or σ_2 increases, the noise amplitude increases. As ϵ increases, the frequency of impulse noise occurrence increases. A sample of such a noise sequence with ϵ=0.2, σ_1=65 and σ_2=650 is shown in Figure 5.10.

Baseline drift

The baseline drift is simulated by adding a slanted line to a sinusoid:

$$b(n) = B + m \times n + A \times \cos(2\pi \frac{n}{N} + \phi).$$

The period of the sinusoid, N, controls the severity of the baseline roll while the slope of the line, m, controls the degree of upward or downward drift. Using different values for ϕ allows different baseline drift sequences to be generated with similar characteristics. The bias term B is set so that the sequence values do not get out of range. Figures 5.11 and 5.12 shows test data formed by adding an EKG signal to impulsive noise and a baseline drift sequence. The unit of the time-axis is 1/1000 seconds.

Performance Measures

Three metrics were used to measure the difference between two signals, s and $\hat{s}$, assuming they each have L number of points:

$$d_1(s,\hat{s}) = \frac{1}{R} \frac{1}{L} \sum_{n=1}^{L} | s(n) - \hat{s}(n) |,$$

$$d_2(s,\hat{s}) = \frac{1}{R} \left\{ \frac{1}{L} \sum_{n=1}^{L} | s(n) - \hat{s}(n) |^2 \right\}^{\frac{1}{2}},$$

$$d_\infty(s,\hat{s}) = \frac{1}{R} \max_{n=1,\ldots,L} | s(n) - \hat{s}(n) |.$$

d_2 is a measure of the root-mean-squared difference between two signals and is the most commonly used metric. d_1 is a measure of the mean absolute difference while d_∞ is a measure of the maximum deviation of one signal from another. They are normalized by R, the peak-to-peak value of s.

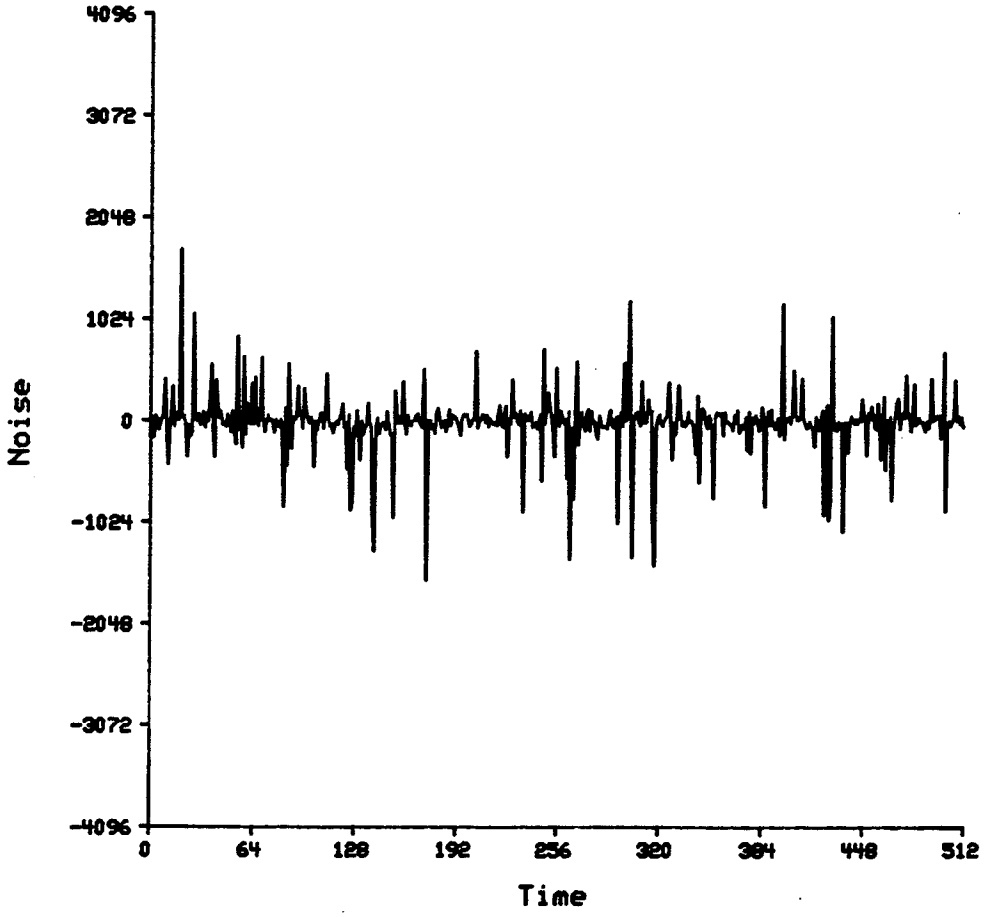


Figure 5.10. A sample noise sequence. Generated by the ϵ-mixture Gaussian noise model, with ϵ=0.2, σ_1=65 and σ_2=650.

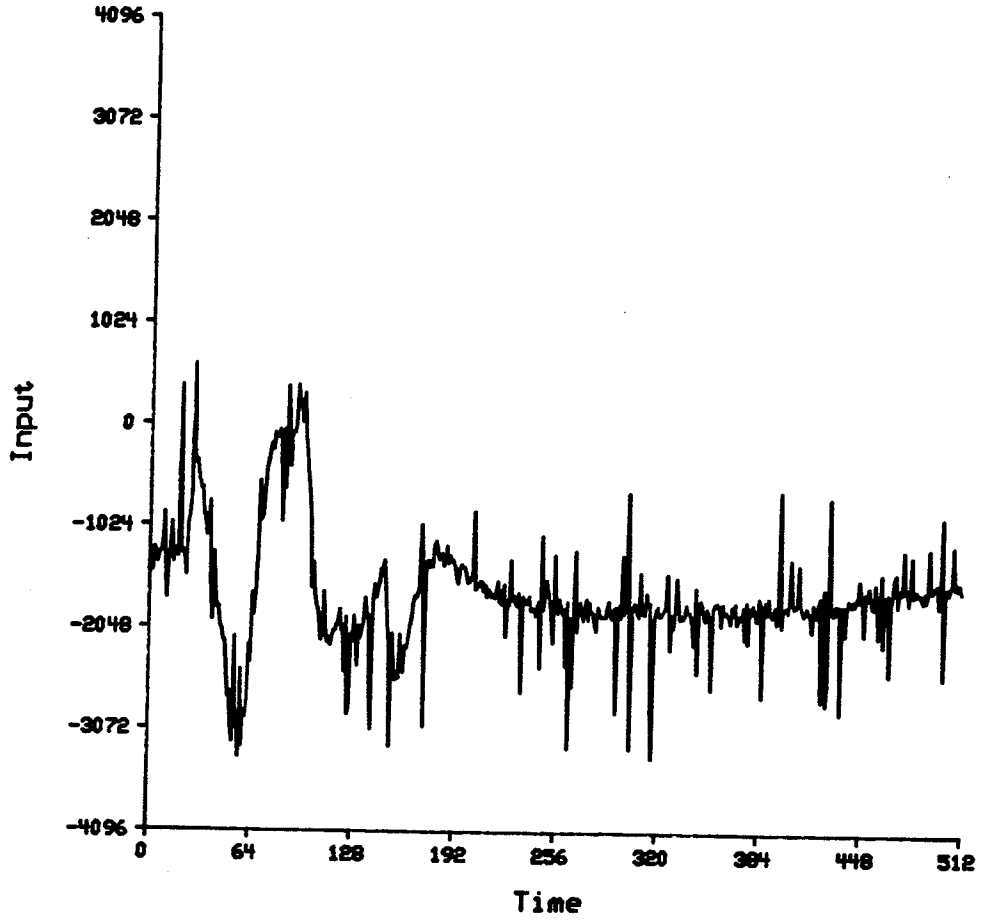


Figure 5.11. An EKG signal corrupted by additive noise and baseline drift (512 data points shown).

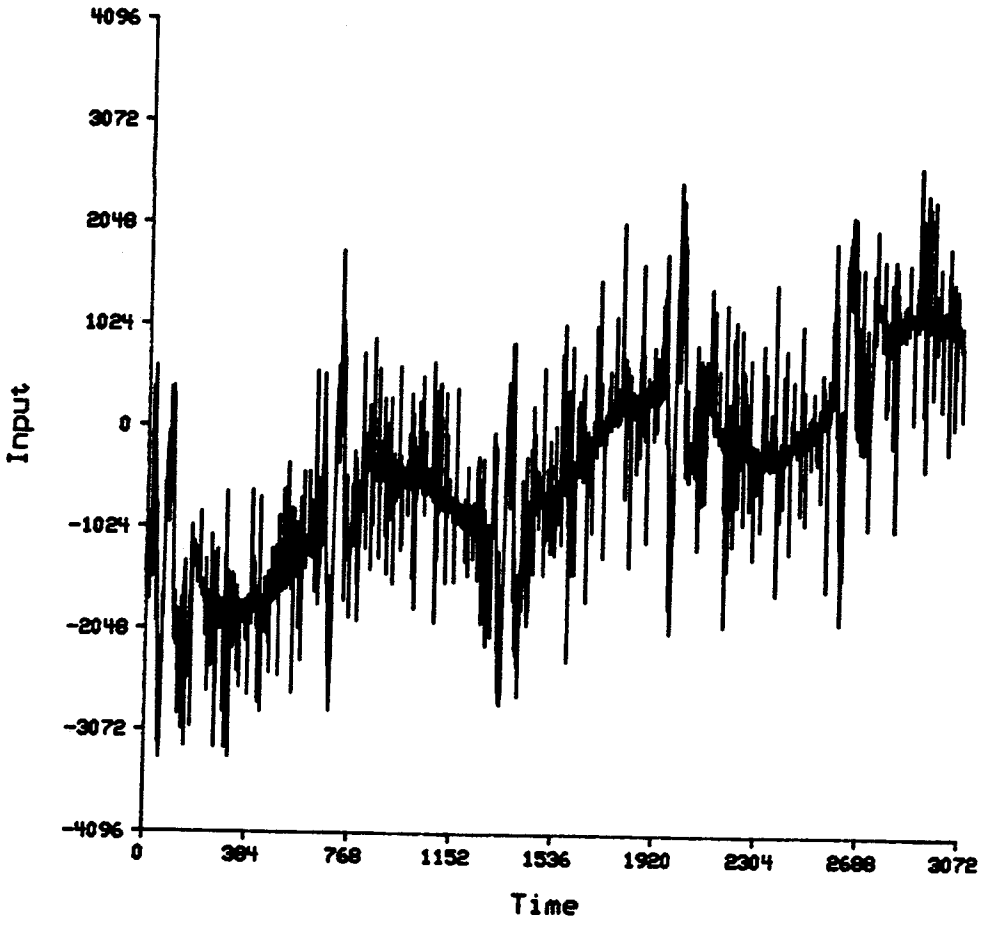


Figure 5.12. An EKG signal corrupted by additive noise and baseline drift (3072 data points shown).

5.4.1. Test Results

Data sequences corrupted by using different sets of noise parameters were used to test the new algorithm. Throughout this section, $s(n)$ denotes the EKG signal, $r(n)$ denotes the corrupted data, and $\hat{s}(n)$ denotes the processed result.

The structuring element used for noise suppression has length 5 and values of (0, 1, 5, 1, 0) as shown in Figure 5.13. The two structuring elements used in background normalization both have triangular shape. In the first stage, the structuring element has length 41 and values as shown in Figure 5.14; the structuring element used in the second stage has length 81 and values as shown in Figure 5.15.

For the input data sequence depicted in Figures 5.11 and 5.12, the parameters are:

ϵ-mixture noise: $\epsilon = 0.2$, σ_1=65 and σ_2=650 ,

baseline-drift: m=0.8, A=500, N=1000, and ϕ=0.8 .

Based on 3000 data points, the differences between s and r as measured by the three metrics are:

$$d_1(s,r) = 0.2433 , \quad d_2(s,r) = 0.2993 , \quad d_\infty(s,r) = 0.9887 .$$

Results of applying the new algorithm to this signal are shown in Figures 5.16 and 5.17. Figure 5.16 shows 512 points of the output, indicating the noise suppression performance. Figure 5.17 shows 3072 points of the same output sequence, illustrating the baseline correction capability of the new algorithm. The performance measures on the processed signal, calculated based on 3000 data points, are:

$$d_1(s,\hat{s}) = 0.02652 , \quad d_2(s,\hat{s}) = 0.04074 , \quad d_\infty(s,\hat{s}) = 0.2239 .$$

As measured by d_2, for example, the deviation of the input data from the signal is 30% of the signal peak-to-peak value. This value is a measure of the power of the noise and the baseline drift. The deviation was reduced to 4% by the processing.

A different set of noise parameters that increased the noise level and baseline drift was used to corrupt the signal, Figure 5.18 shows the input data sequence to the algorithm. The parameters for this signal are:

ϵ-mixture noise: $\epsilon = 0.25$, σ_1=75 and σ_2=750 ,

baseline-drift: m=1.4, A=500, N=800, and ϕ=0.3 .

Based on 3000 data points, the differences between s and r as measured by the three metrics are:

$$d_1(s,r) = 0.3358 , \quad d_2(s,r) = 0.3922 , \quad d_\infty(s,r) = 1.10155 .$$

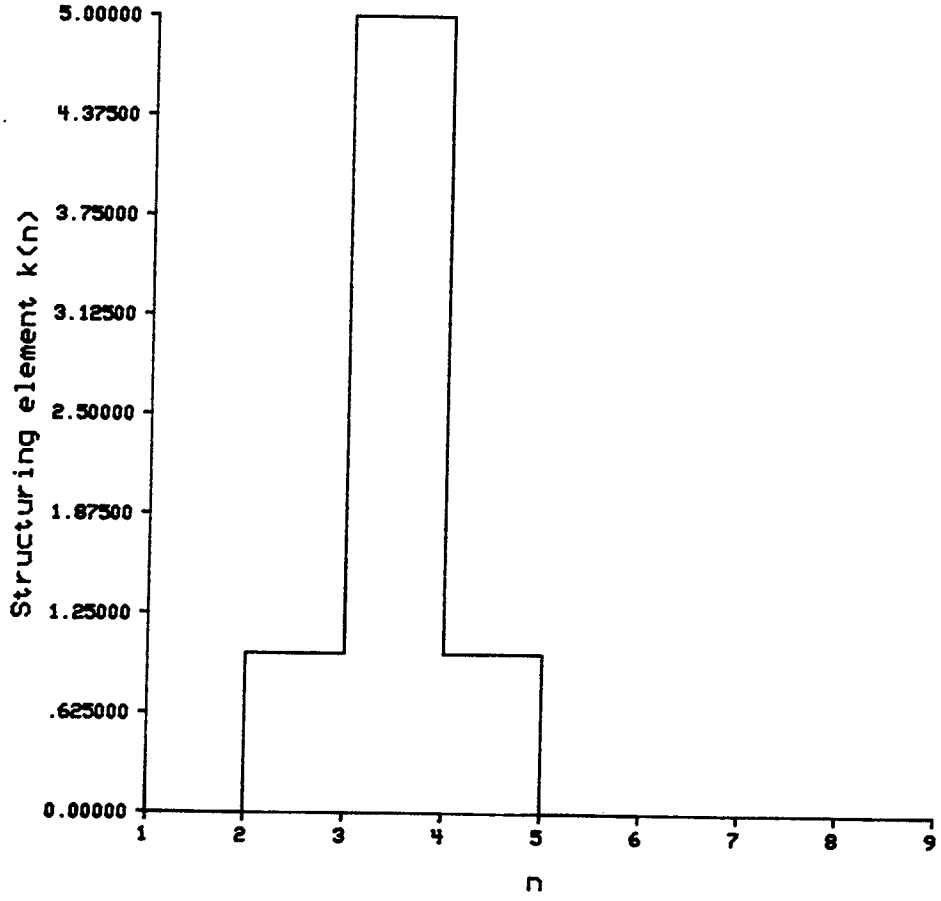


Figure 5.13. Structuring element used in the impulsive noise suppression algorithm.

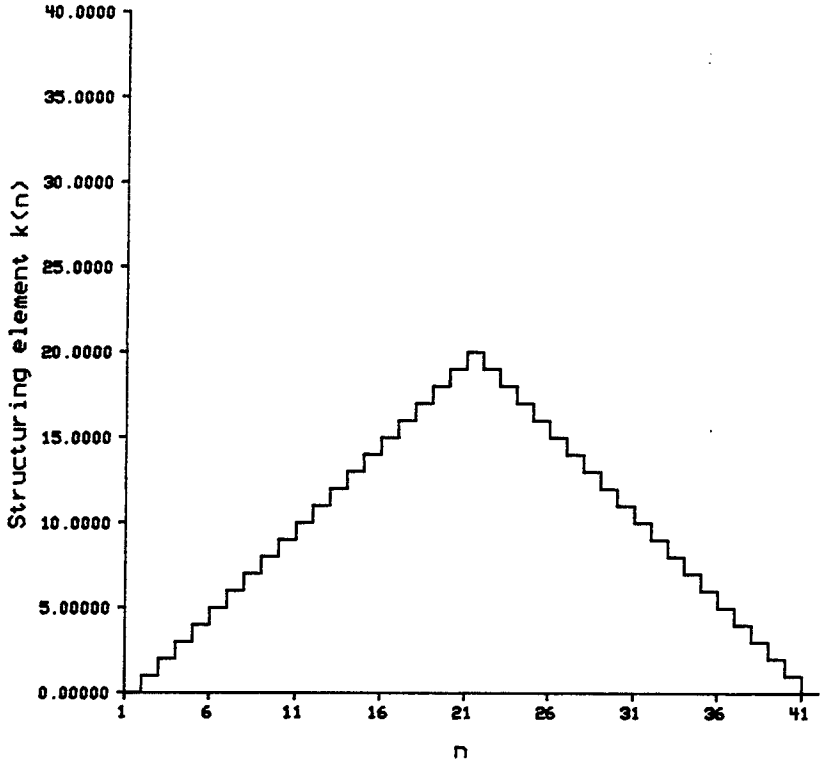


Figure 5.14. Structuring element used in the background normalization algorithm for removing peaks.

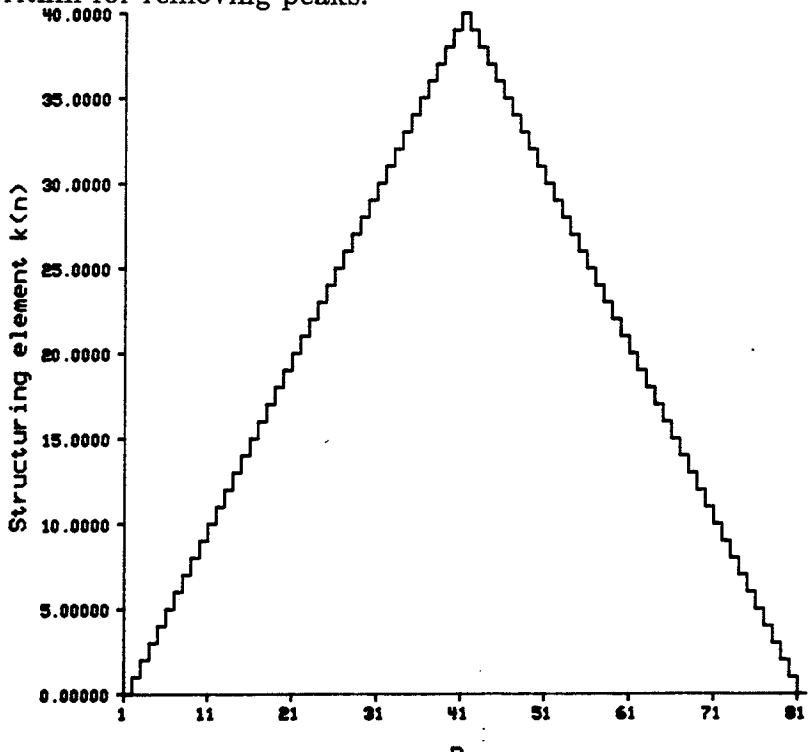


Figure 5.15. Structuring element used in the background normalization algorithm for filling pits.

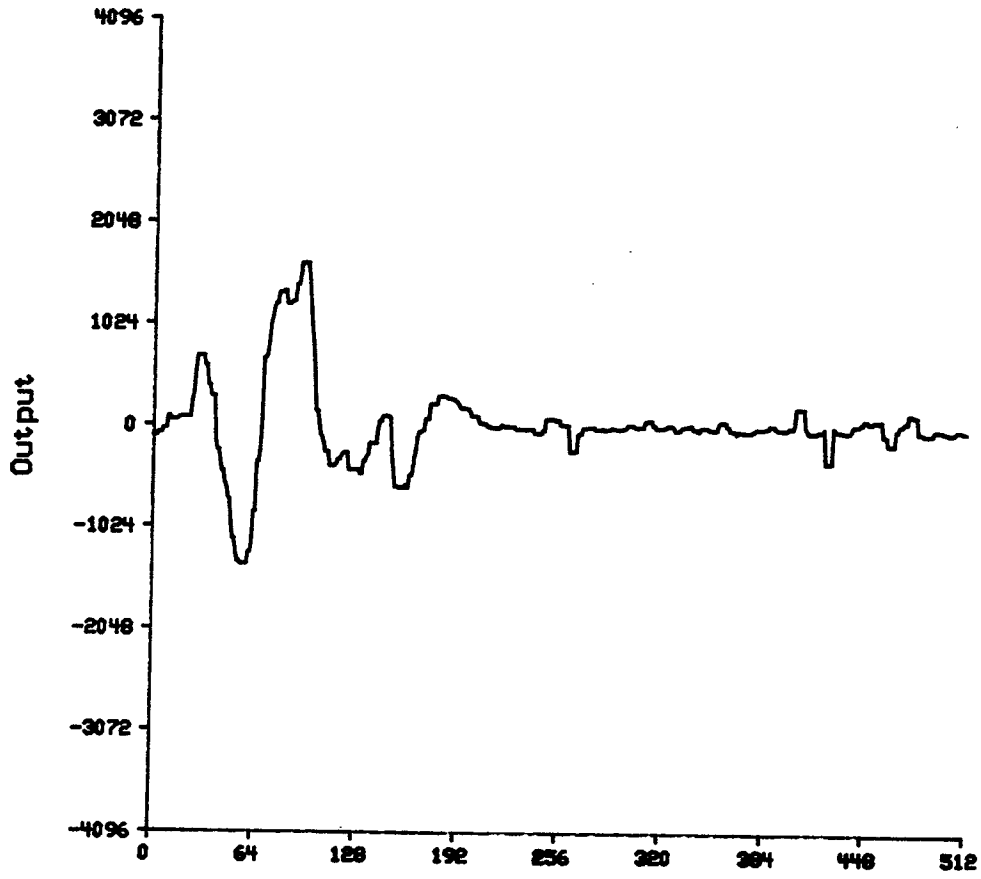


Figure 5.16. Result after processing the input data shown in Figure 5.11 (512 data points shown).

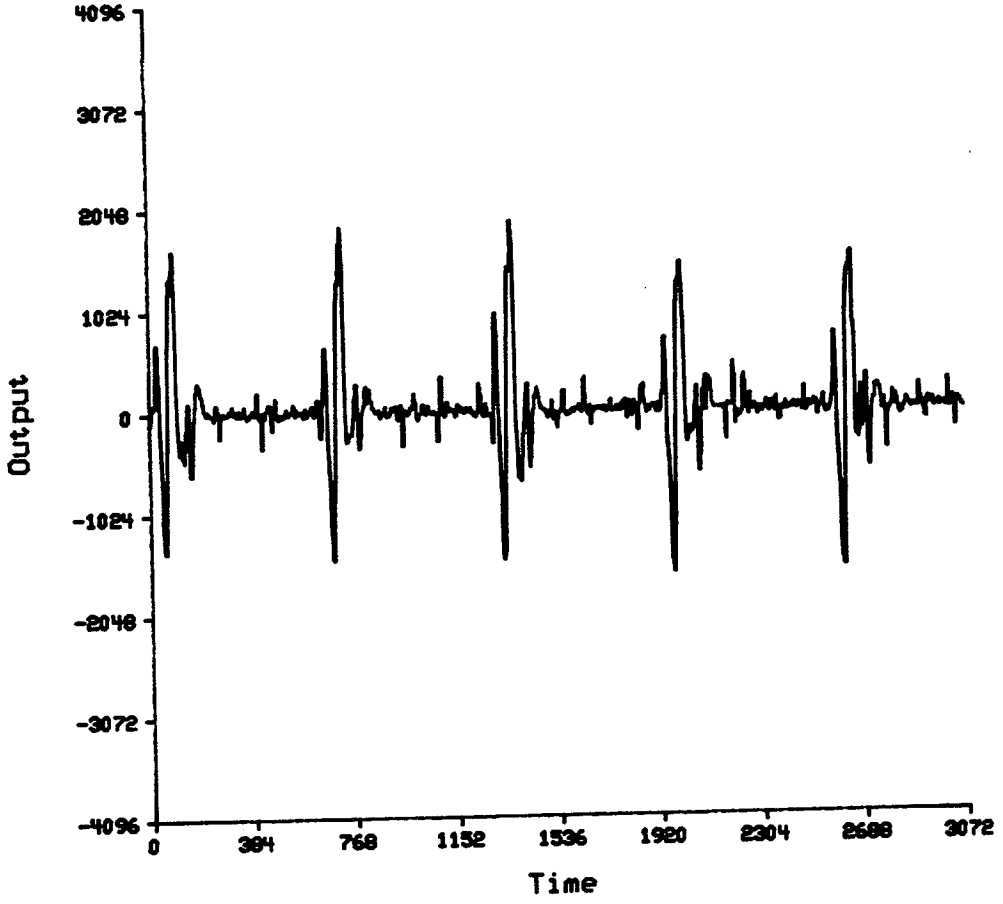


Figure 5.17. Result after processing the input data shown in Figure 5.12 (3072 data points shown).

The result of applying the algorithm to this signal is shown in Figure 5.19. The performance measures were calculated based on 3000 data points to be:

$$d_1(s,\hat{s}) = 0.03087\ , \quad d_2(s,\hat{s}) = 0.04506\ , \quad d_\infty(s,\hat{s}) = 0.2281\ .$$

Even though the deviation of the input data from the ideal signal is increased to 40% of the signal peak-to-peak value, the processed data still achieves a low 4.5% deviation.

The effectiveness of the new algorithm in suppressing noise and correcting baseline drift and roll can be seen through the empirical performance measures and by plots of the results. In the next section, we shall examine the performance of noise suppression under a wider range of situations.

5.5. Noise Suppression Performance

The structuring element with length 5 and values (0, 1, 5, 1, 0) was used to study the noise suppression of the algorithm under a variety of noise situations. Noise as modeled in Section 5.4 was used; the EKG signal was corrupted by noise with different ϵ and σ_2/σ_1 values. The value of ϵ ranged from 0.1 to 0.5, corresponding to an increasing probability of impulsive noise occurrence. At $\epsilon = 0.5$, a sample has equal probability of being corrupted by background Gaussian noise or by higher amplitude impulsive noise. The standard deviation of background Gaussian noise, σ_1, was set at 65. The standard deviation of impulsive Gaussian noise, σ_2, ranged from $2\sigma_1$ to $20\sigma_1$, corresponding to the increasing impulse amplitudes. To consider the severity of the impulsive noise, recall that the peak values of the EKG waves are 1636 and -1722. At $\sigma_2 = 650$, the impulsive noise amplitude is above 1300 with probability 4.5%. At $\sigma_2 = 1300$, the impulsive noise amplitude is above 1300 with probability 31.7%. The resulting values of the performance measures are tabulated in Tables 5.1 to 5.3. These values are shown in Figures 5.20 to 5.22. As can be seen from the tables and the figures, the performance shows no significant deterioration as the noise situation worsens. The expected increases in all three measures as either ϵ or σ_2 increases are gradual and exhibit no abrupt jumps or sharp rises. For example, at the worst case of $\epsilon = 0.5$ and $\sigma_2/\sigma_1 = 20$, d_2 is 0.11024. The input data and processed output for this case are shown in Figures 5.23 and 5.24, respectively. While there are notches that remain in the waves, the QRS complex is isolated from noise.

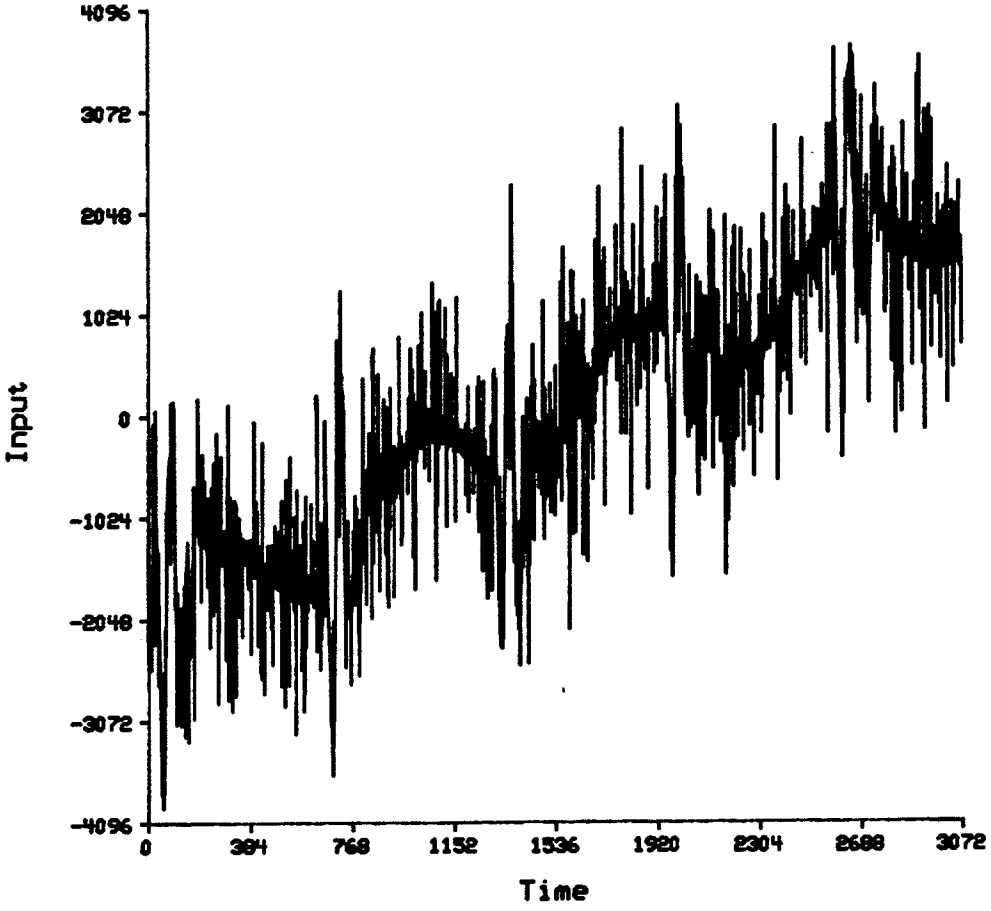


Figure 5.18. An EKG signal corrupted by increased additive noise and baseline drift (3072 data points shown).

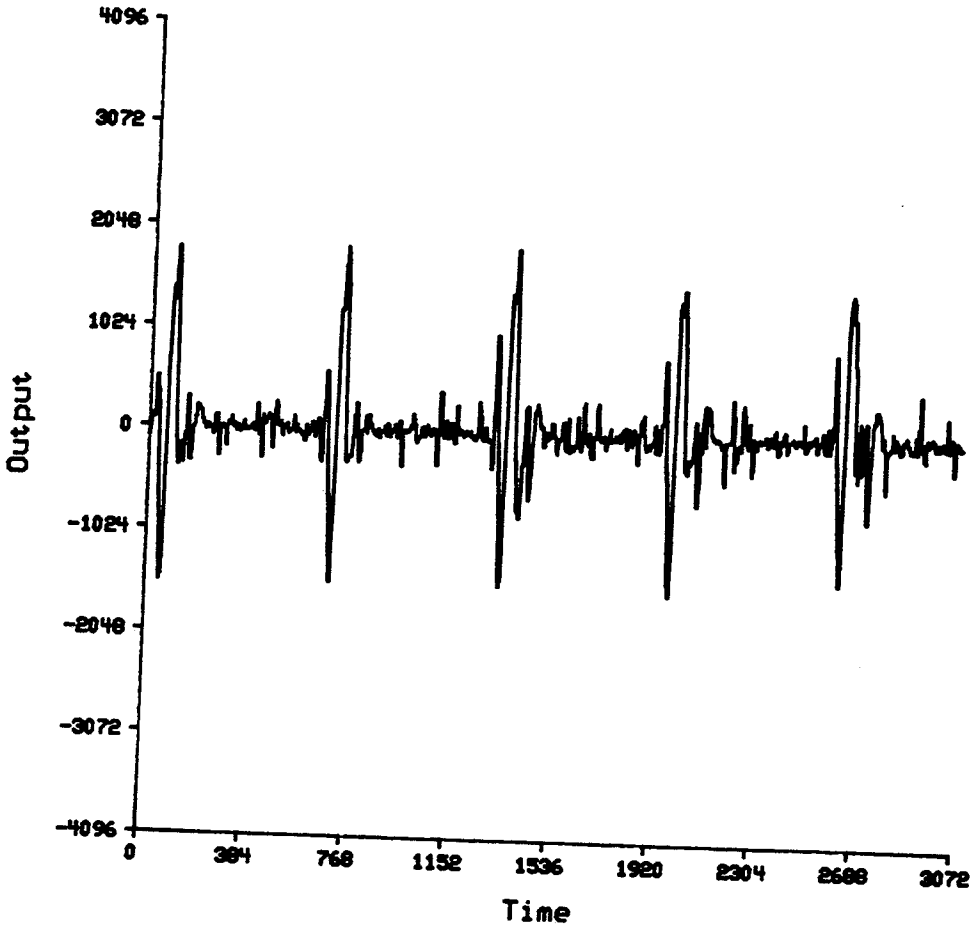


Figure 5.19. Result after processing the input data shown in Figure 5.18.

Table 5.1. Noise suppression performance as measured by d_1. The variance of background Gaussian noise, σ_1, was set at 65.

Noise Suppression Performance as Measured by d_1					
σ_2	ϵ=0.1	ϵ=0.2	ϵ=0.3	ϵ=0.4	ϵ=0.5
130	0.009048	0.009720	0.010283	0.010960	0.011654
195	0.009506	0.010936	0.012177	0.013695	0.015234
260	0.009928	0.012123	0.014118	0.016532	0.019000
325	0.010328	0.013372	0.016100	0.019457	0.022841
390	0.010742	0.014663	0.018134	0.022463	0.026778
455	0.011143	0.015952	0.020190	0.025450	0.030705
520	0.011531	0.017241	0.022255	0.028490	0.034674
585	0.011937	0.018549	0.024344	0.031592	0.038670
650	0.012319	0.019813	0.026391	0.034661	0.042647
715	0.012685	0.021061	0.028413	0.037708	0.046608
780	0.013039	0.022291	0.030431	0.040751	0.050571
845	0.013396	0.023532	0.032476	0.043836	0.054566
910	0.013737	0.024760	0.034528	0.046920	0.058533
975	0.014075	0.025987	0.036584	0.049998	0.062467
1040	0.014416	0.027220	0.038642	0.053054	0.066364
1105	0.014754	0.028436	0.040679	0.056111	0.070229
1170	0.015085	0.029649	0.042696	0.059144	0.074086
1235	0.015399	0.030858	0.044687	0.062129	0.077891
1300	0.015704	0.032055	0.046654	0.065093	0.081683

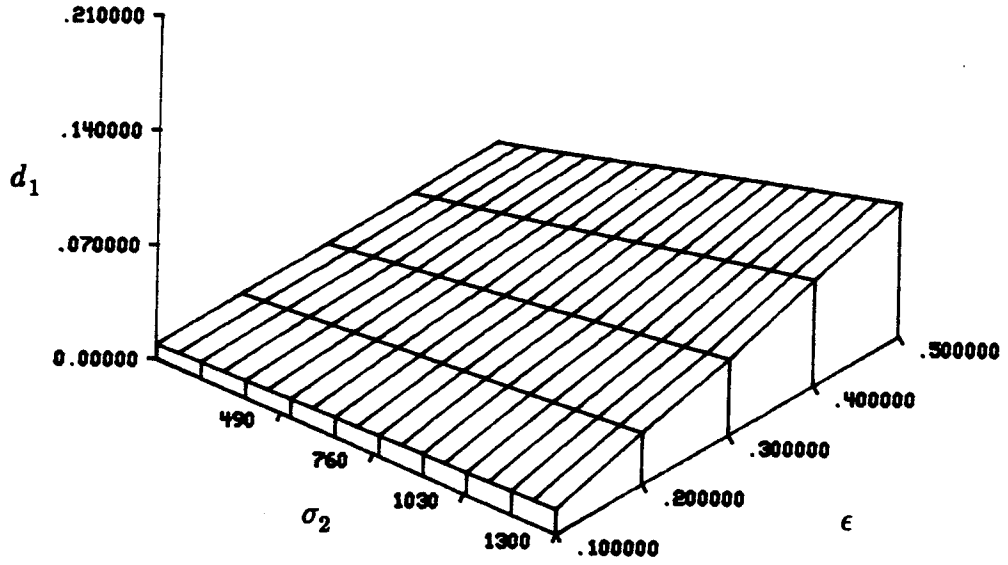


Figure 5.20. Noise suppression performance as measured by d_1. The variance of background Gaussian noise, σ_1, was set at 65.

Table 5.2. Noise suppression performance as measured by d_2. The variance of background Gaussian noise, σ_1, was set at 65.

Noise Suppression Performance as Measured by d_2					
σ_2	ϵ=0.1	ϵ=0.2	ϵ=0.3	ϵ=0.4	ϵ=0.5
130	0.012477	0.013341	0.014018	0.014836	0.015616
195	0.013264	0.015143	0.016801	0.018661	0.020295
260	0.014130	0.017164	0.019808	0.022704	0.025296
325	0.015072	0.019458	0.023000	0.026968	0.030488
390	0.016132	0.021947	0.026351	0.031380	0.035869
455	0.017231	0.024509	0.029809	0.035787	0.041264
520	0.018368	0.027103	0.033328	0.040314	0.046694
585	0.019598	0.029791	0.036921	0.044917	0.052127
650	0.020816	0.032418	0.040512	0.049532	0.057571
715	0.022009	0.035019	0.044117	0.054157	0.063024
780	0.023178	0.037609	0.047722	0.058809	0.068503
845	0.024389	0.040240	0.051362	0.063503	0.074009
910	0.025558	0.042881	0.055031	0.068191	0.079439
975	0.026759	0.045543	0.058728	0.072838	0.084777
1040	0.027979	0.048229	0.062353	0.077368	0.089989
1105	0.029214	0.050925	0.065921	0.081878	0.095085
1170	0.030451	0.053630	0.069489	0.086375	0.100205
1235	0.031673	0.056345	0.072982	0.090742	0.105220
1300	0.032883	0.059055	0.076457	0.095089	0.110236

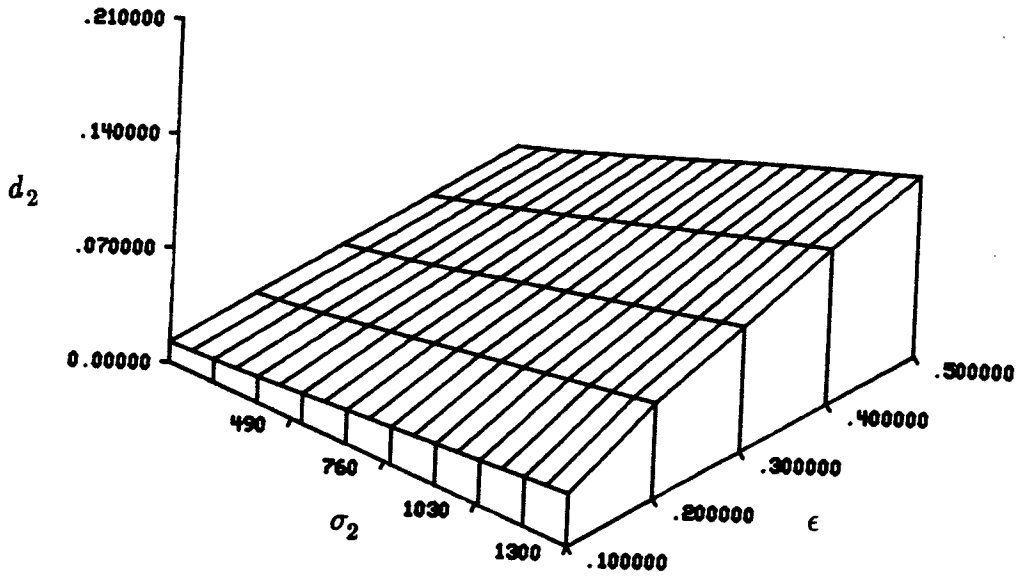


Figure 5.21. Noise suppression performance as measured by d_2. The variance of background Gaussian noise, σ_1, was set at 65.

Table 5.3. Noise suppression performance as measured by d_∞. The variance of background Gaussian noise, σ_1, was set at 65.

Noise Suppression Performance as Measured by d_∞					
σ_2	ϵ=0.1	ϵ=0.2	ϵ=0.3	ϵ=0.4	ϵ=0.5
130	0.075640	0.075640	0.075640	0.077725	0.087850
195	0.075640	0.083979	0.083979	0.098868	0.114652
260	0.079809	0.100357	0.100357	0.122394	0.141453
325	0.094401	0.116736	0.116736	0.146218	0.167362
390	0.108398	0.132519	0.133413	0.166766	0.193270
455	0.122990	0.148898	0.148898	0.166766	0.220071
520	0.136986	0.164681	0.164681	0.179571	0.232281
585	0.151876	0.181060	0.181060	0.196248	0.235855
650	0.158130	0.191781	0.197141	0.206671	0.239428
715	0.158130	0.191781	0.216796	0.225432	0.240917
780	0.158130	0.191781	0.236450	0.246575	0.250447
845	0.171828	0.191781	0.256105	0.267421	0.270697
910	0.185229	0.202204	0.275759	0.288565	0.288565
975	0.198928	0.215307	0.295712	0.298987	0.307624
1040	0.212329	0.228708	0.305241	0.305241	0.317153
1105	0.226027	0.241811	0.305241	0.305241	0.317153
1170	0.239428	0.254914	0.305241	0.305837	0.317153
1235	0.253127	0.268017	0.305241	0.309410	0.317153
1300	0.266528	0.281120	0.305241	0.309410	0.317153

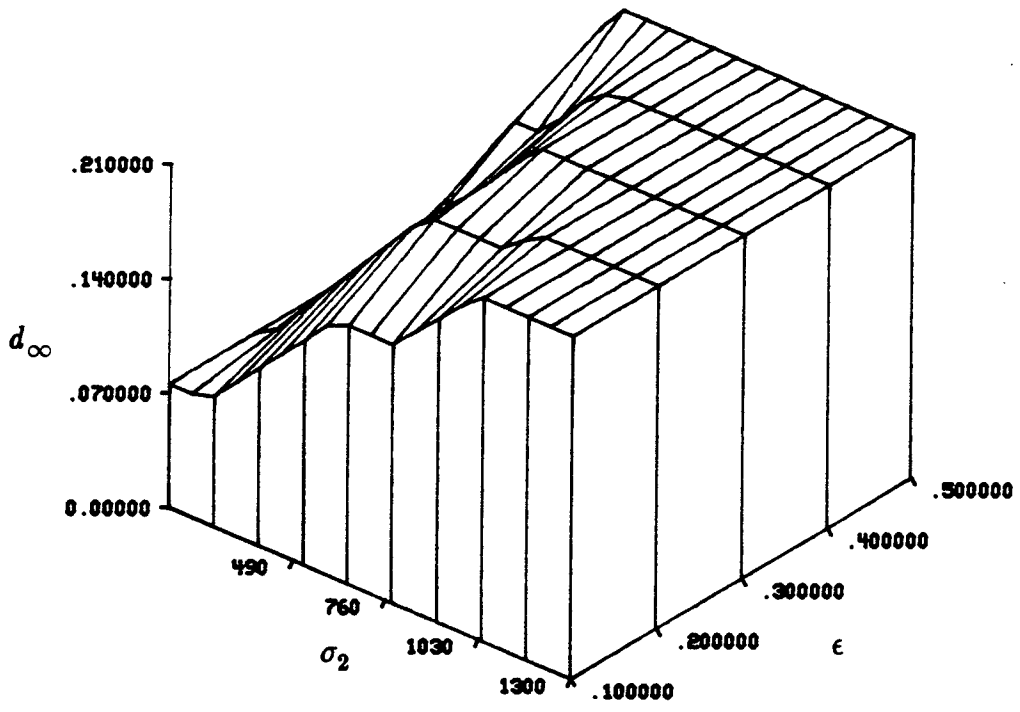


Figure 5.22. Noise suppression performance as measured by d_∞. The variance of background Gaussian noise, σ_1, was set at 65.

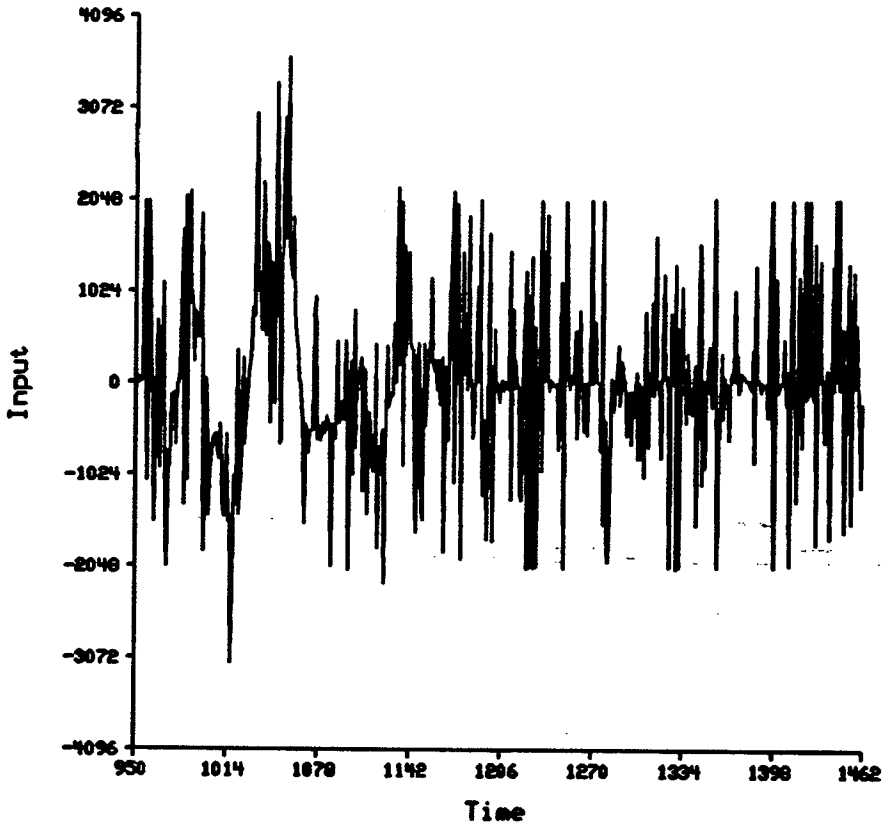


Figure 5.23. EKG signal heavily corrupted by impulsive noise. The noise parameters are: ϵ=0.5, σ_2=1300, and σ_1=65.

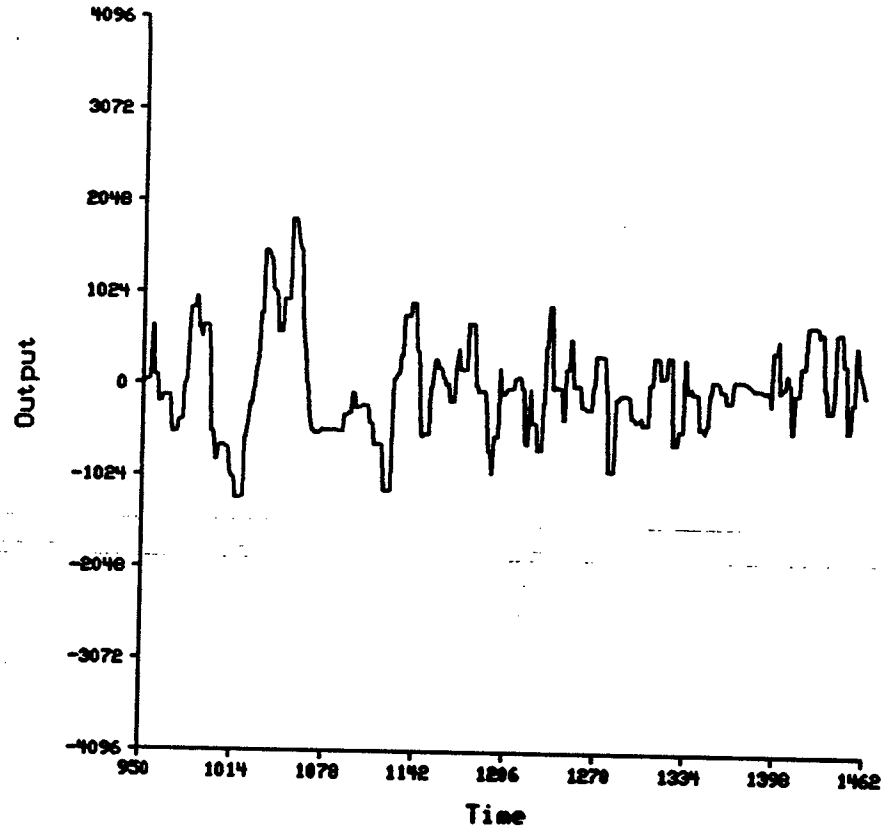


Figure 5.24. The result of processing the data shown in Figure 5.23.

5.6. Experiments with a Parameterized Structuring Element

Unlike the design of linear filters, methods for structuring element design in morphological signal processing is an open research problem [Coy88]. In this section, we use a dome-like structuring element parameterized by its width, height, and "shape" to examine these effects on the noise suppression performance of the algorithm. The parametric structuring element is denoted by $q(n)$, for n=0,1,...,$2N$. N is the parameter that determines the width of q. For n=0,1,...,N, let

$$q(n) = h \times (1 - e^{-\alpha n}) ,$$

and for n=N+1,...,$2N$, let

$$q(n) = q(2N - n) .$$

The structuring element q is then symmetric with respect to the peak at $q(N)$. The height of q is controlled by h; the actual height parameter we used is the peak value $q(N)$, which is related to h. The "shape" of q is controlled by α: which ranges from 0 to ∞. The parameter that we used instead is γ, which ranges from 0 to 1. γ is related to α by

$$\gamma = 1 - e^{-\alpha N} .$$

As γ increases, q changes from thick and round to thin and sharp. Figure 5.25 shows a plot of q with γ varying from 0.01 to 0.99. To illustrate the shape change, the peak value $q(N)$ was set at 10 while the width parameter N was set at 30.

The parametric structuring element with different sets of parameters was used in the noise suppression stage of the algorithm to process the known signal corrupted by impulsive noise as described in Section 4. The parameters for the ϵ-mixture noise are ϵ=0.2, σ_1=65, and σ_2=650. The different parameter values for the structuring element used were:

γ	:	0.99, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1;
$q(N)$	:	1, 3, 5, 7, 9, 11, 13, 15, 17, 19;
N	:	1, 3, 5, 7, 9, 11, 13, 15, 17, 19.

The performance is evaluated by comparing the input to the output relative to the three metrics. From the results, the only parameter that has a significant effect on the performance is the width. Figures 5.26, 5.27, and 5.28 show the results for γ fixed at 0.5. It can be seen that the performance deteriorates with increasing N but shows no change due to varying $q(N)$. Figures 5.29, 5.30, and 5.31 show the results for N fixed at 3. It can be seen

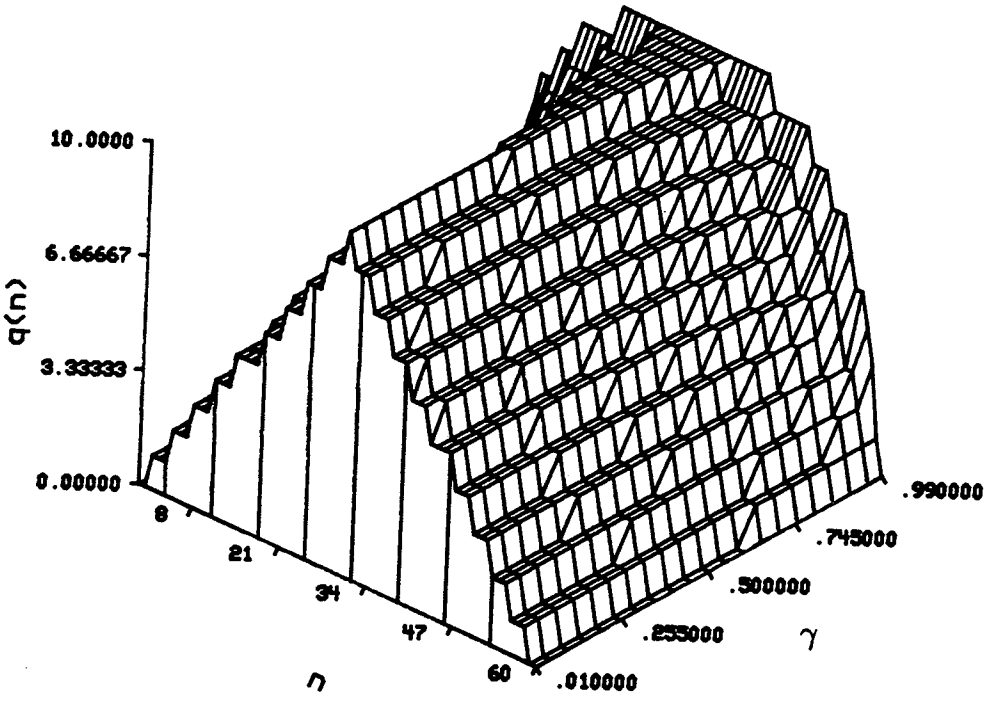


Figure 5.25. Structuring elements with different γ values.

that the performance shows no significant change with different values for $q(N)$ and γ. As there are not a lot of difference between structuring elements with different γ values when the width is as low as 7, it is understandable that γ does not affect the performance. We show the results for $N=3$ because it achieves a performance of $d_2<0.03$, compared to the 0.09 which is how much the input data deviates from the signal. Figures 5.32, 5.33, and 5.34 show the results for $q(N)=5$; again, it can be seen that the performance deteriorates with increasing N but shows no change due to varying γ.

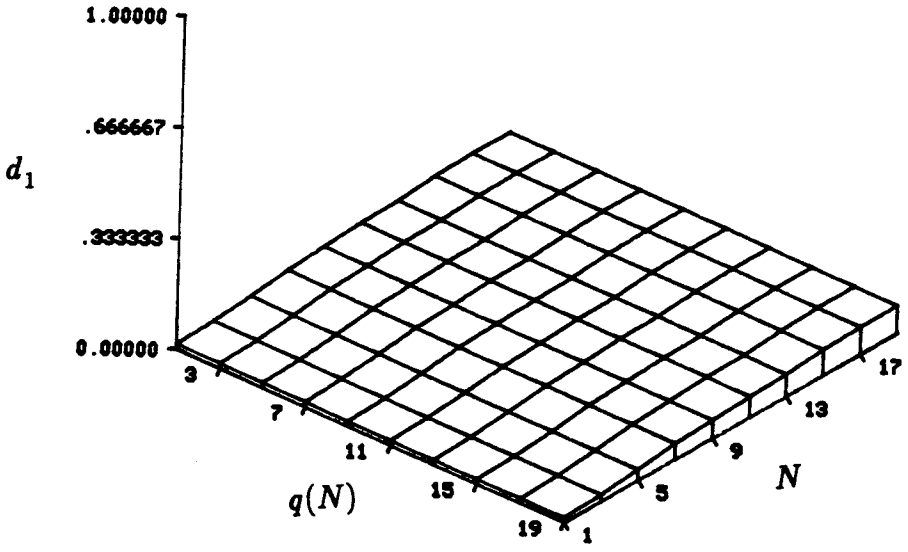


Figure 5.26. Noise suppression performance as measured by d_1 of the parametric structuring element with γ set at 0.5.

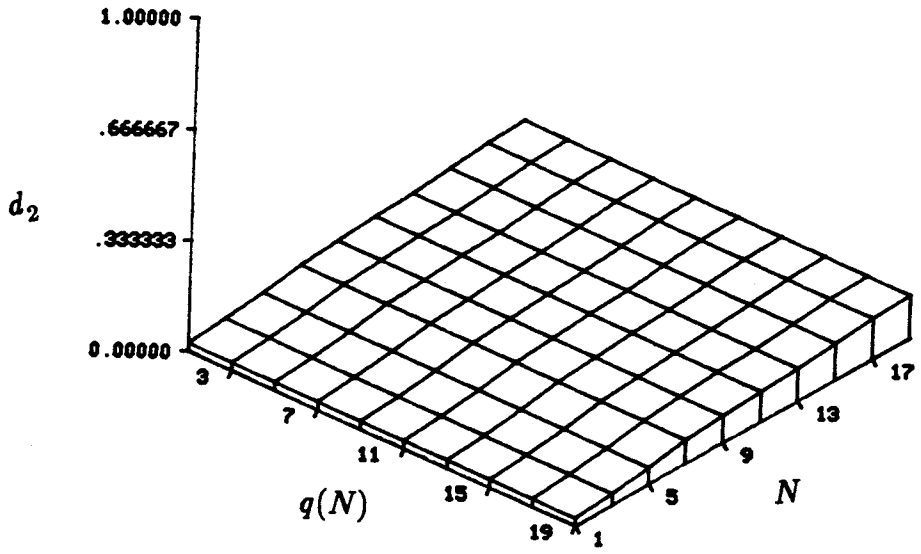


Figure 5.27. Noise suppression performance as measured by d_2 of the parametric structuring element with γ set at 0.5.

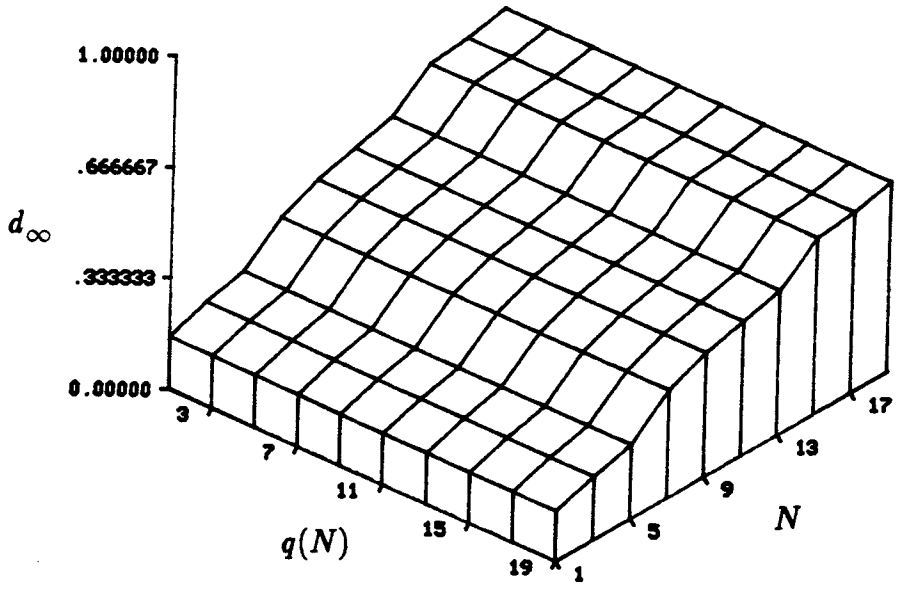


Figure 5.28. Noise suppression performance as measured by d_∞ of the parametric structuring element with γ set at 0.5.

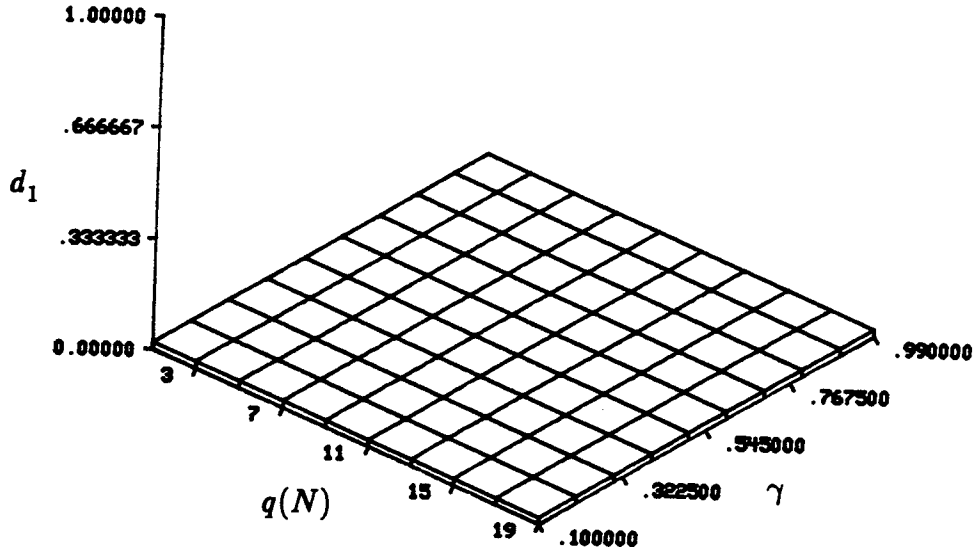


Figure 5.29. Noise suppression performance as measured by d_1 of the parametric structuring element with N set at 3.

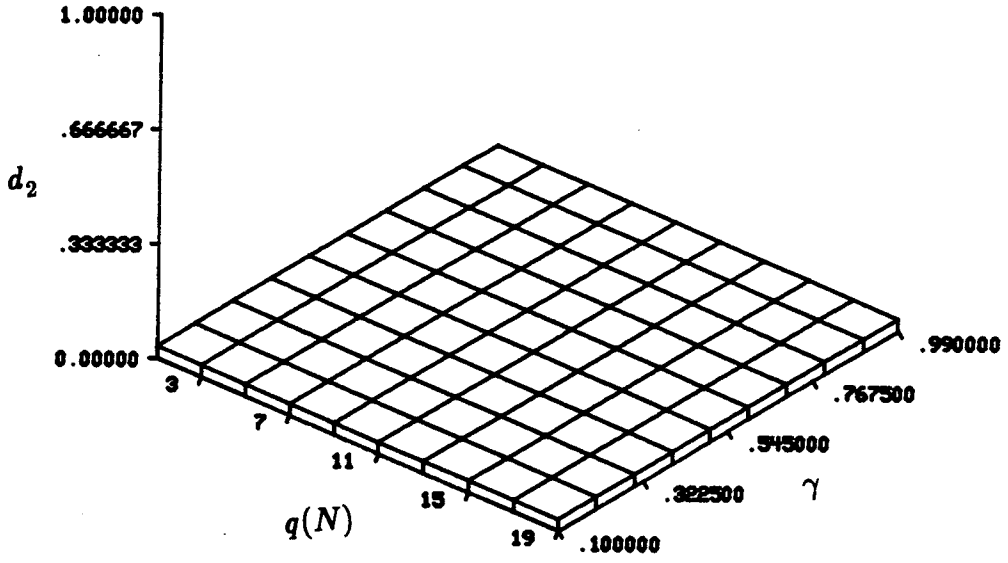


Figure 5.30. Noise suppression performance as measured by d_2 of the parametric structuring element with N set at 3.

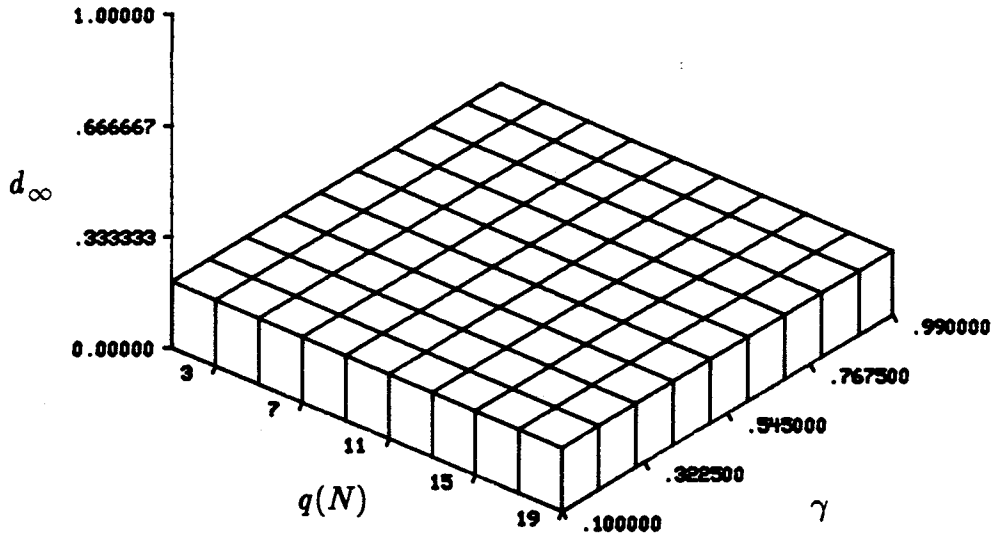


Figure 5.31. Noise suppression performance as measured by d_∞ of the parametric structuring element with N set at 3.

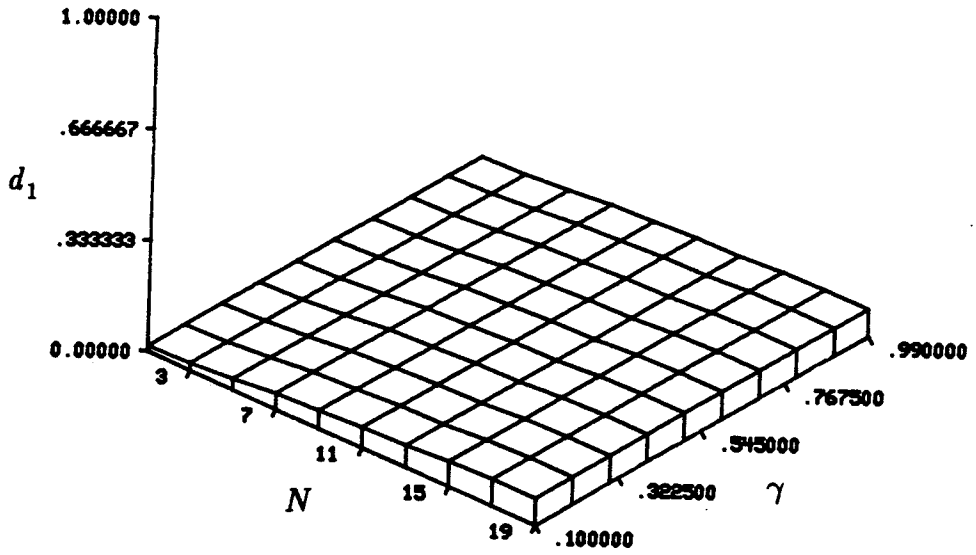


Figure 5.32. Noise suppression performance as measured by d_1 of the parametric structuring element with $q(N)$ set at 5.

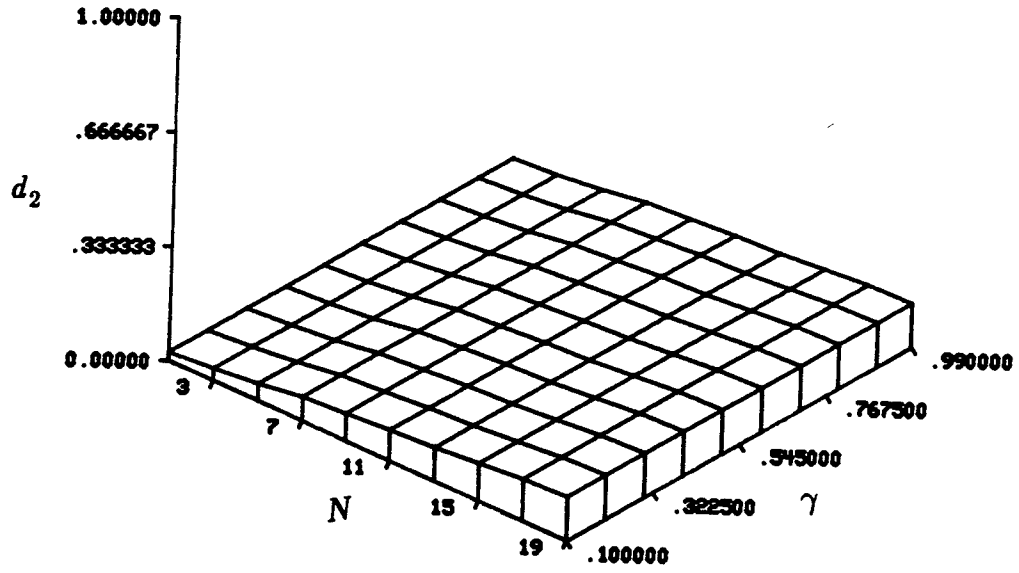


Figure 5.33. Noise suppression performance as measured by d_2 of the parametric structuring element with $q(N)$ set at 5.

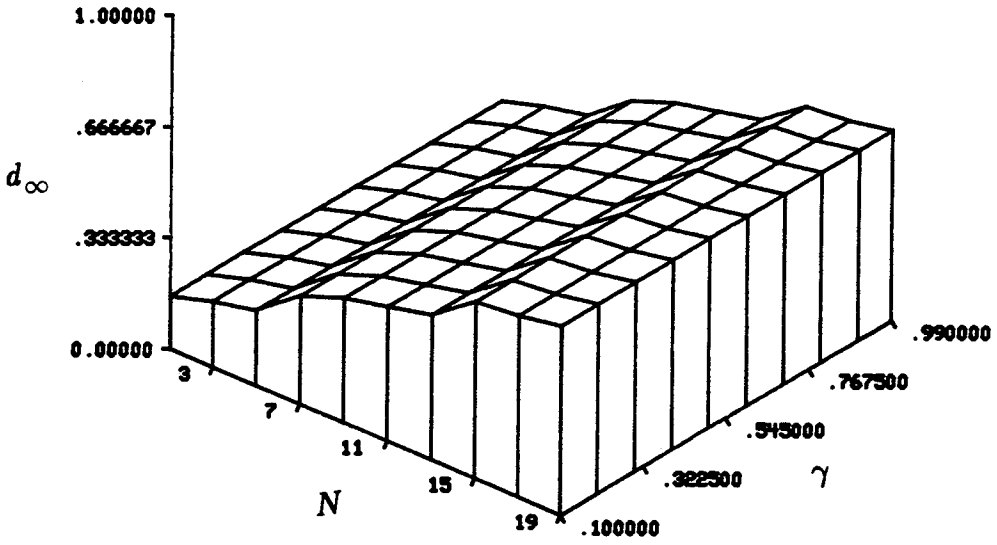


Figure 5.34. Noise suppression performance as measured by d_∞ of the parametric structuring element with $q(N)$ set at 5.

5.7. Sinusoidal Response

Guidelines have been established by the American Heart Association for EKG signal processing governing acceptable frequency response of linear filters [Pip75]. It is difficult, however, to show that the method described in this chapter conforms to these guidelines since nonlinear filtering cannot be analyzed in terms of frequency responses. Linear filtering, via the superposition principle, allows frequency response analysis by decomposing the input and output into a sum of sinusoidals. The superposition principle does not apply to nonlinear filters, hence "bandwidth" is a meaningless quantity. There is ongoing work in the theoretical analysis of morphological operators based on a weaker form of superposition principle, known as threshold decomposition [Mar87 ....., Wen86]. The general problem of investigating the response of morphological operators to a sinusoidal input remains an open research problem [Nee88]. In this section, we address this problem by examining the sine wave response of the new algorithm to study the extent that these signals are modified relative to the type of structuring element.

According to the American Heart Association committee report on electrocardiography [Pip75], the recommended bounds to frequency response is flat from 0.14 Hz to 50 Hz, reduced from unit gain by no more than 6%. Since the morphological processing is discrete in nature, if a sinusoidal signal with a certain frequency is unmodified, any signals with a lower frequency would also be unmodified. Sinusoidal signals at frequencies ranging from 0 Hz to 180 Hz, sampled at rates ranging from 360 Hz to 1800 Hz were processed by the algorithm described in Figure 5.6. The structuring elements used were as shown in Figure 5.15. The processed result is compared to the original sinusoidals using the performance measures described in Section 5.4. The resulting values are tabulated in Tables 5.4 to 5.6 and are plotted in Figures 5.35 to 5.37. As the results indicate, increasing the sampling rate produces better results. All three performance measures show steady rises as the sampling rate drops and as the input frequency increases, eventually flattening out.

It should be emphasized that there is no direct relation between these plots and the common amplitude frequency response plot used to describe linear filterss. For example, from Table 5.5, a sampling rate in the range of 1000 Hz is needed to obtain performance that achieves $d_2 < 0.1$ for input with a frequency of up to 50 Hz, which is the upper frequency limit for the flat response required by the American Heart Association for "faithful

reproduction of the electrocardiographic waveform " [Pip75]. It is difficult to set a general performance standard, since it depends on other factors such as the intended application of the processed data. The choice of 0.1 here is based on the observation that for a fixed sampling rate, d_2 tends to rise relatively quickly after it reaches 0.1.

Since the processing is discrete, the signal bandwidth, the sampling rate, and the structuring element length are all related. The highest frequency of the signal is usually dictated by the application: e.g., 50 Hz in EKG signal processing. Given that the signal frequency is fixed, either the structuring element can be shortened or a higher sampling rate can be used to minimize the distortion of the waveform. The minimum length of the structuring element is determined by the nature and characteristics of noise expected. For example, for impulsive noise, a typical choice of the minimum length is from 3 to 5; for other artifact suppression, the minimum length might have to be increased depending on the situation.

The problem s examined below from the viewpoint of determining the required sampling rate of the EKG data for a given required performance. A structuring element with length $2N+1$ modifies a wave of an EKG signal by truncating its top. The worst case is for a structuring element to have constant values of 0. Let the tolerated attenuation factor be $1-\tau$; i.e., the tip of the wave should not be truncated by a factor more than τ. Suppose we model a wave by a raised cosine waveform with frequency f_0 Hz. A reasonable estimate of f_0 is 50, since it then follows that the duration of a wave is 20 msec. Typically, a QRS complex with three waves lasts from 80 to 100 msec. It can easily be shown that the sampling rate f_s can be found by:

$$f_s = \frac{2\pi f_0 N}{\cos^{-1}(1-\tau)}. \tag{5.1}$$

For example, with $N = 2$, $f_0 = 50$Hz, and $\tau = 0.06$, we found f_s to be 1.8 kHz. Alternatively, we can model a wave by a Gaussian pulse with σ^2 as its variance. Denoting the bandwidth of the signal as f_0 Hz, and noting that the Fourier transform of a Gaussian pulse is again a Gaussian pulse with variance $(2\pi\sigma)^{-2}$, we see that the sampling rate f_s can be determined by:

$$f_s = \frac{2\pi f_0 N}{(-2\log(1-\tau))^{1/2}}. \tag{5.2}$$

Using $\tau = 0.06$, $f_0 = 50$Hz, $N = 2$, we again obtain f_s as 1.8 kHz. From Equations (5.1) and (5.2), we see that the required sampling rate varies

directly with N and f_0. It varies inversely and nonlinearly with τ; i.e., with increasing τ, f_s drops off quite rapidly. Consequently, f_s decreases rather rapidly as τ increases.

Table 5.4. Amount of modification of sampled sinusoidal input as measured by d_1.

Sinusoidal Input Modification as Measured by d_1						
Input Frequency (in Hz)	Sampling Rate (in Hz)					
	360	680	1000	1320	1560	1800
0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5	0.00046	0.00004	0.00002	0.00000	0.00000	0.00000
10	0.00405	0.00055	0.00016	0.00005	0.00002	0.00003
15	0.01369	0.00296	0.00090	0.00024	0.00013	0.00008
20	0.03200	0.00483	0.00149	0.00063	0.00035	0.00022
25	0.08871	0.01390	0.00297	0.00190	0.00114	0.00046
30	0.10256	0.02334	0.00744	0.00218	0.00130	0.00083
35	0.22223	0.03791	0.01216	0.00532	0.00323	0.00207
40	0.31885	0.05638	0.01815	0.00783	0.00480	0.00308
45	0.30086	0.07800	0.02559	0.01092	0.00654	0.00297
50	0.31265	0.10383	0.02352	0.01527	0.00932	0.00405
60	0.33264	0.16702	0.05703	0.01779	0.01083	0.00703
70	0.31120	0.25688	0.09098	0.04127	0.02550	0.01663
80	0.33438	0.29642	0.13282	0.06082	0.03778	0.02480
90	0.24962	0.30601	0.18121	0.08222	0.05102	0.02352
100	0.30635	0.31826	0.16847	0.11314	0.07063	0.03200
110	0.32256	0.31344	0.30407	0.10256	0.09356	0.06202
120	0.49919	0.31371	0.29787	0.18912	0.12022	0.08043
130	0.32075	0.31108	0.30757	0.22976	0.10256	0.09974
140	0.30604	0.31048	0.31730	0.27700	0.17978	0.12236
150	0.31100	0.31850	0.30878	0.30073	0.21457	0.10256
160	0.32185	0.32174	0.31471	0.30049	0.25670	0.17627
170	0.30069	0.24962	0.31518	0.30891	0.29757	0.20657
180	0.49939	0.30574	0.31048	0.31275	0.29480	0.16847

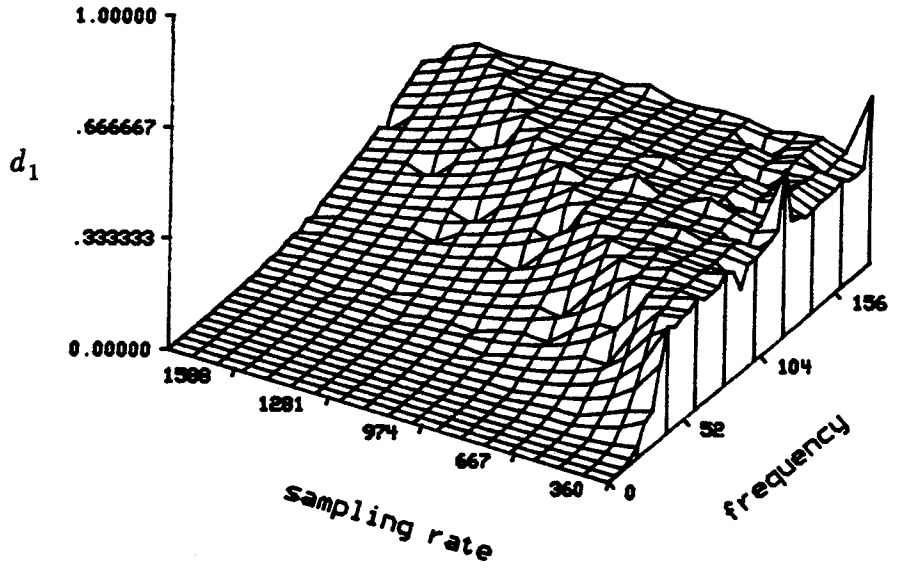


Figure 5.35. Amount of modification of sampled sinusoidal input as measured by d_1 plotted against input frequencies and sampling rates.

Table 5.5. Amount of modification of sampled sinusoidal input as measured by d_2.

Sinusoidal Input Modification as Measured by d_2						
Input Frequency (in Hz)	Sampling Rate (in Hz)					
	360	680	1000	1320	1560	1800
0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5	0.00161	0.00021	0.00010	0.00004	0.00003	0.00003
10	0.01005	0.00188	0.00064	0.00022	0.00014	0.00014
15	0.02768	0.00759	0.00280	0.00093	0.00056	0.00036
20	0.05601	0.01157	0.00433	0.00209	0.00127	0.00086
25	0.12754	0.02751	0.00772	0.00520	0.00338	0.00161
30	0.14655	0.04240	0.01636	0.00598	0.00387	0.00265
35	0.27034	0.06312	0.02454	0.01232	0.00812	0.00559
40	0.37414	0.09077	0.03549	0.01767	0.01172	0.00810
45	0.35253	0.11456	0.04557	0.02254	0.01471	0.00772
50	0.34421	0.14527	0.04337	0.02969	0.01969	0.01005
60	0.35272	0.21495	0.08887	0.03438	0.02273	0.01590
70	0.35376	0.30421	0.13018	0.06775	0.04539	0.03182
80	0.36678	0.33788	0.17907	0.09417	0.06347	0.04472
90	0.35302	0.33522	0.22893	0.12015	0.08107	0.04337
100	0.34703	0.34708	0.22025	0.15577	0.10579	0.05601
110	0.35074	0.34488	0.34838	0.14655	0.13312	0.09490
120	0.49919	0.35257	0.33656	0.24531	0.16940	0.12178
130	0.35039	0.35287	0.33744	0.27775	0.14655	0.14025
140	0.35040	0.35113	0.34783	0.32334	0.22740	0.16595
150	0.35341	0.35286	0.34149	0.34284	0.26303	0.14655
160	0.37052	0.35274	0.34404	0.33809	0.30442	0.22418
170	0.33785	0.35302	0.35129	0.33957	0.34225	0.25469
180	0.49939	0.34293	0.34981	0.33805	0.33606	0.22025

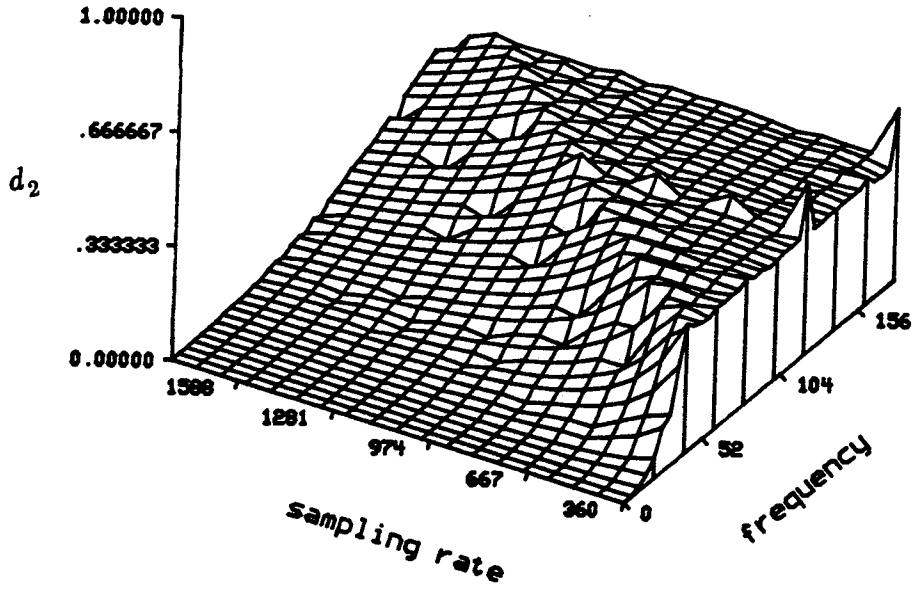


Figure 5.36. Amount of modification of sampled sinusoidal input as measured by d_2 plotted against input frequencies and sampling rates.

Table 5.6. Amount of modification of sampled sinusoidal input as measured by d_∞.

Sinusoidal Input Modification as Measured by d_∞						
Input Frequency (in Hz)	Sampling Rate (in Hz)					
	360	680	1000	1320	1560	1800
0	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5	0.00606	0.00121	0.00061	0.00030	0.00030	0.00030
10	0.02880	0.00697	0.00273	0.00121	0.00091	0.00091
15	0.06547	0.02395	0.01091	0.00364	0.00243	0.00182
20	0.11549	0.03213	0.01425	0.00758	0.00485	0.00333
25	0.24098	0.07123	0.02304	0.01879	0.01334	0.00606
30	0.24856	0.09730	0.04516	0.01879	0.01303	0.00940
35	0.43801	0.14004	0.06608	0.03789	0.02728	0.02031
40	0.57237	0.19046	0.09038	0.05194	0.03704	0.02762
45	0.49864	0.22673	0.10943	0.05850	0.04153	0.02304
50	0.50924	0.26735	0.09397	0.07578	0.05426	0.02880
60	0.49864	0.35314	0.17581	0.07790	0.05577	0.04183
70	0.47408	0.47954	0.24886	0.14823	0.10761	0.08124
80	0.55892	0.51483	0.32958	0.19988	0.14602	0.11077
90	0.49924	0.48742	0.38921	0.22310	0.16338	0.09397
100	0.54198	0.50561	0.34404	0.28190	0.20824	0.11549
110	0.47348	0.47924	0.53653	0.24856	0.25614	0.19703
120	0.49919	0.46866	0.49574	0.41553	0.31129	0.24027
130	0.49106	0.49197	0.47924	0.45438	0.24856	0.26857
140	0.51228	0.47135	0.52137	0.50197	0.38406	0.30039
150	0.49894	0.53228	0.54380	0.51440	0.42377	0.24856
160	0.55236	0.51544	0.47322	0.48663	0.49332	0.39272
170	0.51016	0.49924	0.48681	0.47530	0.53289	0.42710
180	0.49955	0.51106	0.45953	0.47954	0.48469	0.34404

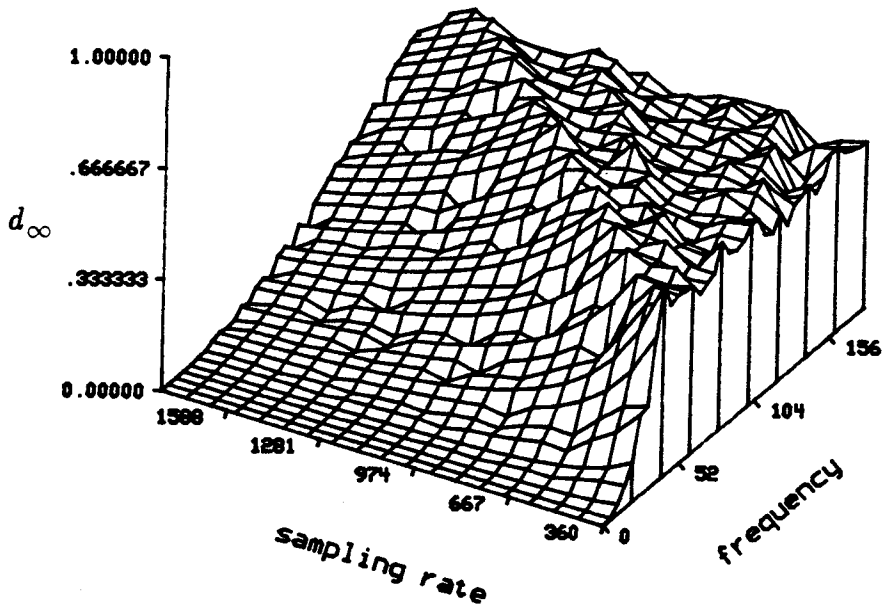


Figure 5.37. Amount of modification of sampled sinusoidal input as measured by d_∞ plotted against input frequencies and sampling rates.

5.8. Experiments with Acquired Data

EKG data from the MIT-BIH Arrhythmia Database [Ins82] were used to evaluate the algorithm performance. Each set of data was digitized at 360 Hz without interruption from a single patient using a modified EKG "lead 2" in which electrodes were placed at the right shoulder and the left abdomen of the patient [Ins82, Bas86].

The processing is done using structuring elements as shown in Figures 5.13, 5.14, and 5.15. The unit of the time-axis for the plots shown in Figures 5.38 to 5.45 is 1/360 seconds. Figure 5.38 and 5.39 show two sequences that were considered to be of "excellent quality" (Tapes 117 and 219 of the MIT-BIH Database). Figures 5.40 and 5.41 show the result after noise suppression processing, the original signal is not significantly modified. Figures 5.42 and 5.43 show the result after background normalization, the original signal is modified to some degree, most notably in the areas of the ST segment. Due to the nature of background normalization, the reference level of the signal is also biased to zero.

Figure 5.44 shows a sequence that was labeled showing bursts of "baseline wander" (Tape 111 of the MIT-BIH Database). Figure 5.45 shows the result after processing. Again, we see very slight modification of the signal, but the background drift is corrected. The reference level of the signal is also biased to zero.

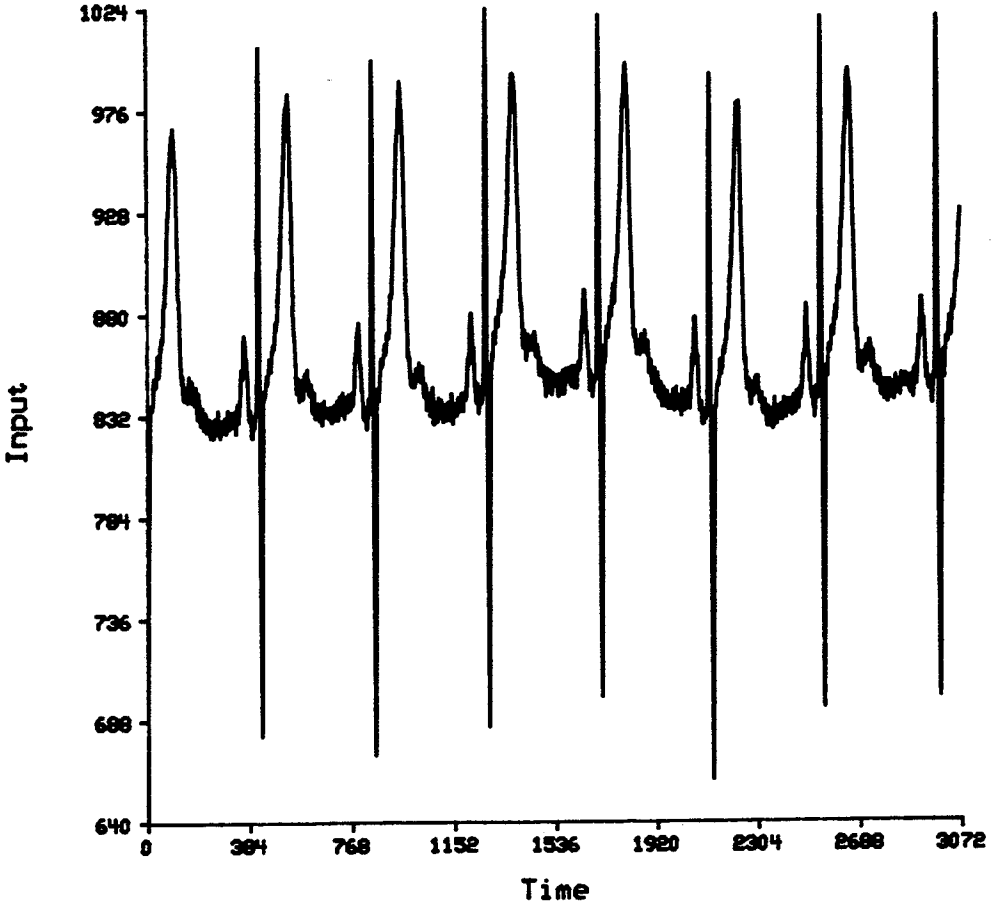


Figure 5.38 A sequence of EKG signal classified as "of excellent quality," from Tape 117 of the MIT-BIH Database.

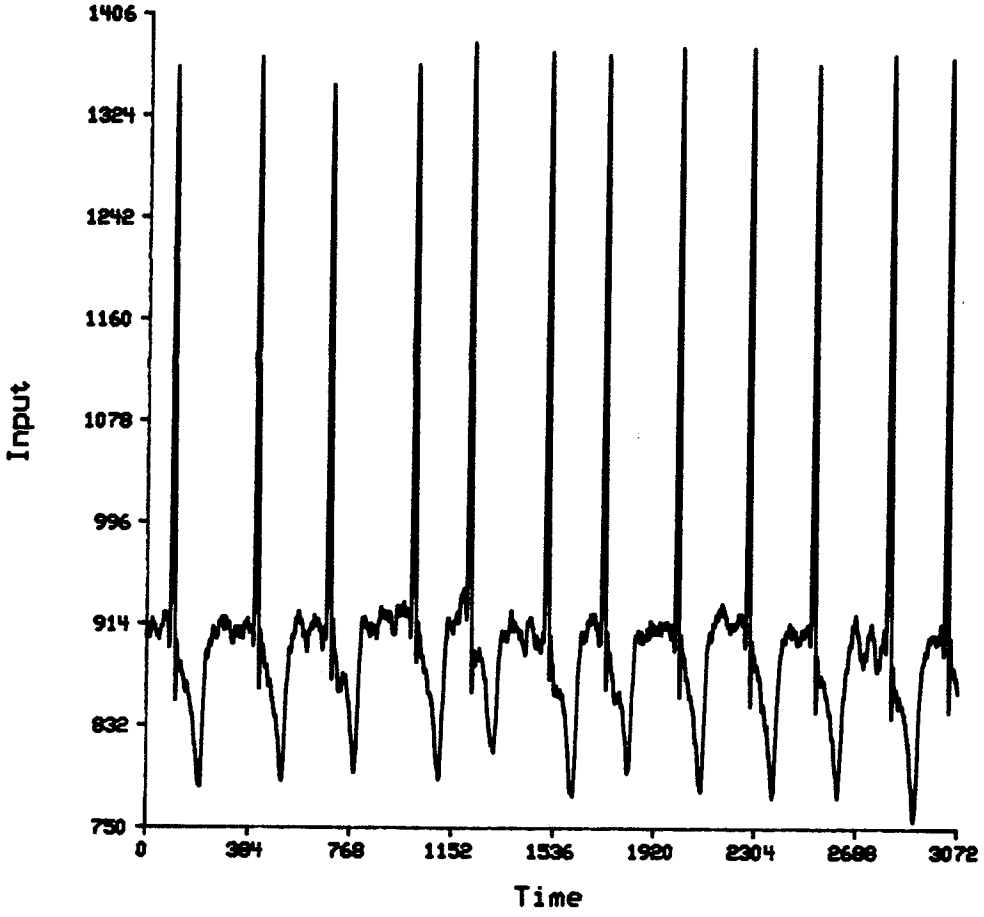


Figure 5.39 A sequence of EKG signal classified as "of excellent quality," from Tape 219 of the MIT-BIH Database.

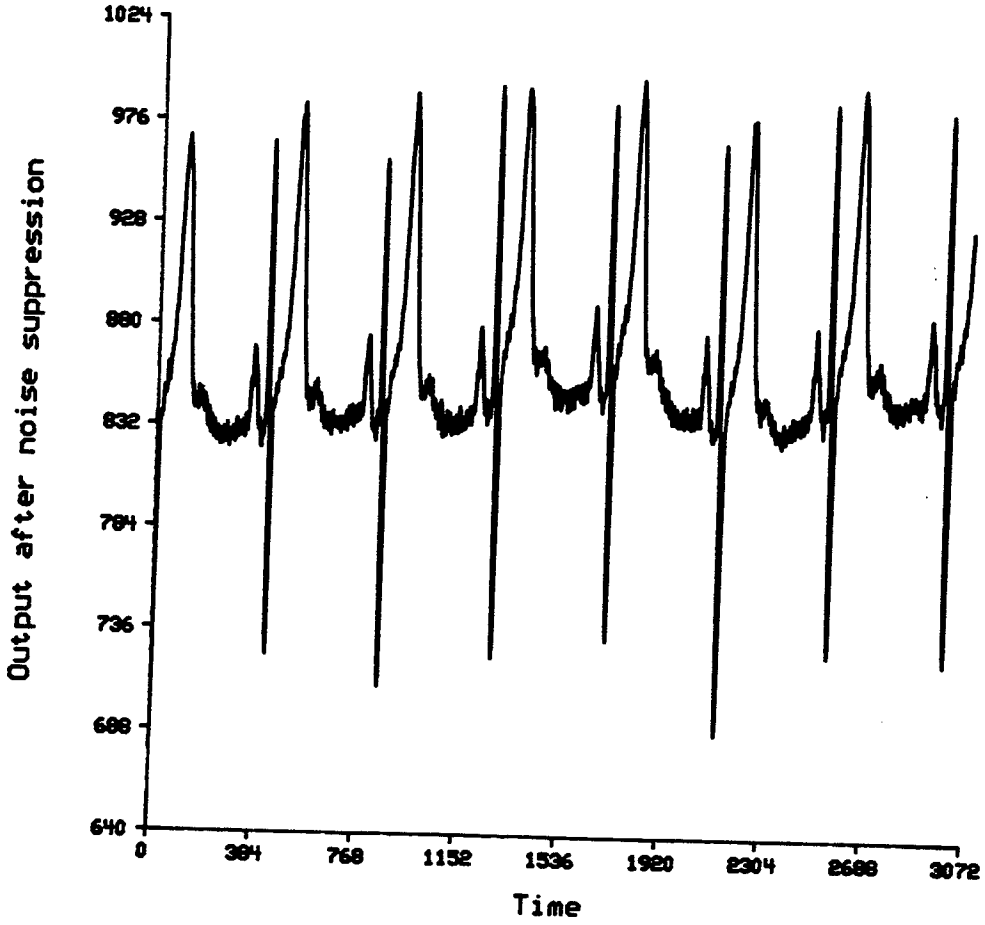


Figure 5.40 Result of noise suppression on the data sequence shown in Figure 5.38.

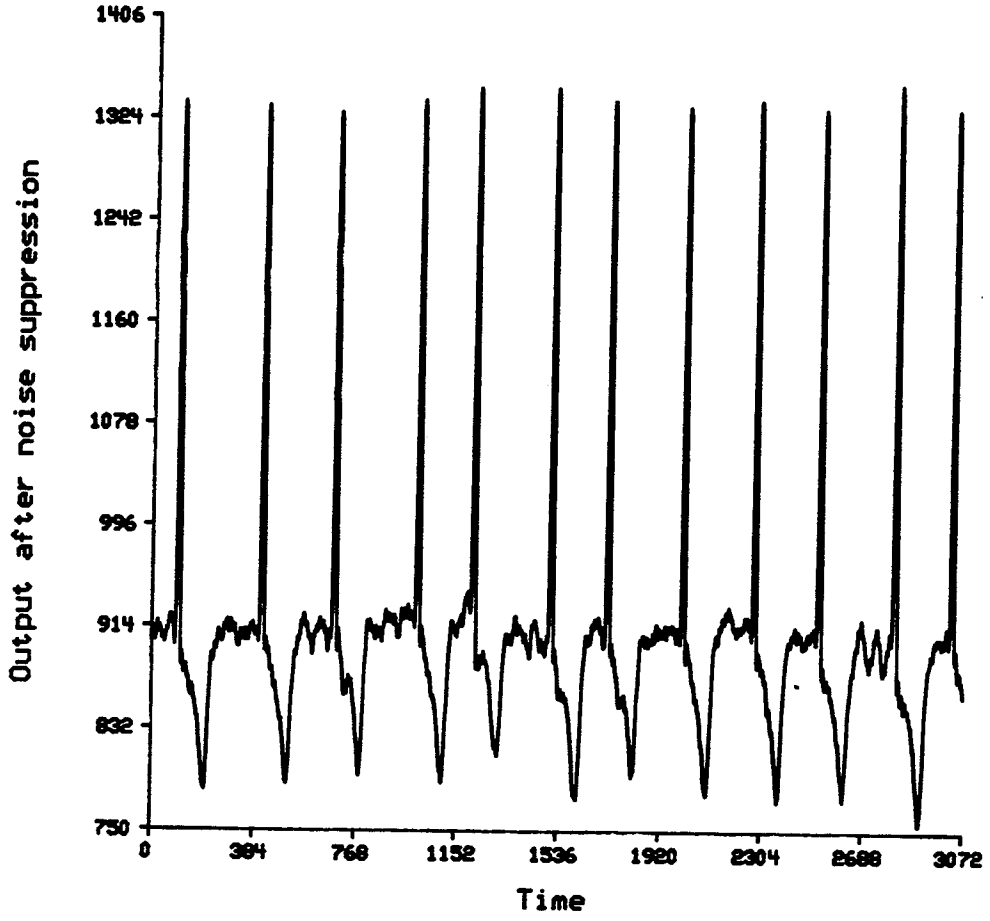


Figure 5.41 Result of noise suppression on the data sequence shown in Figure 5.39.

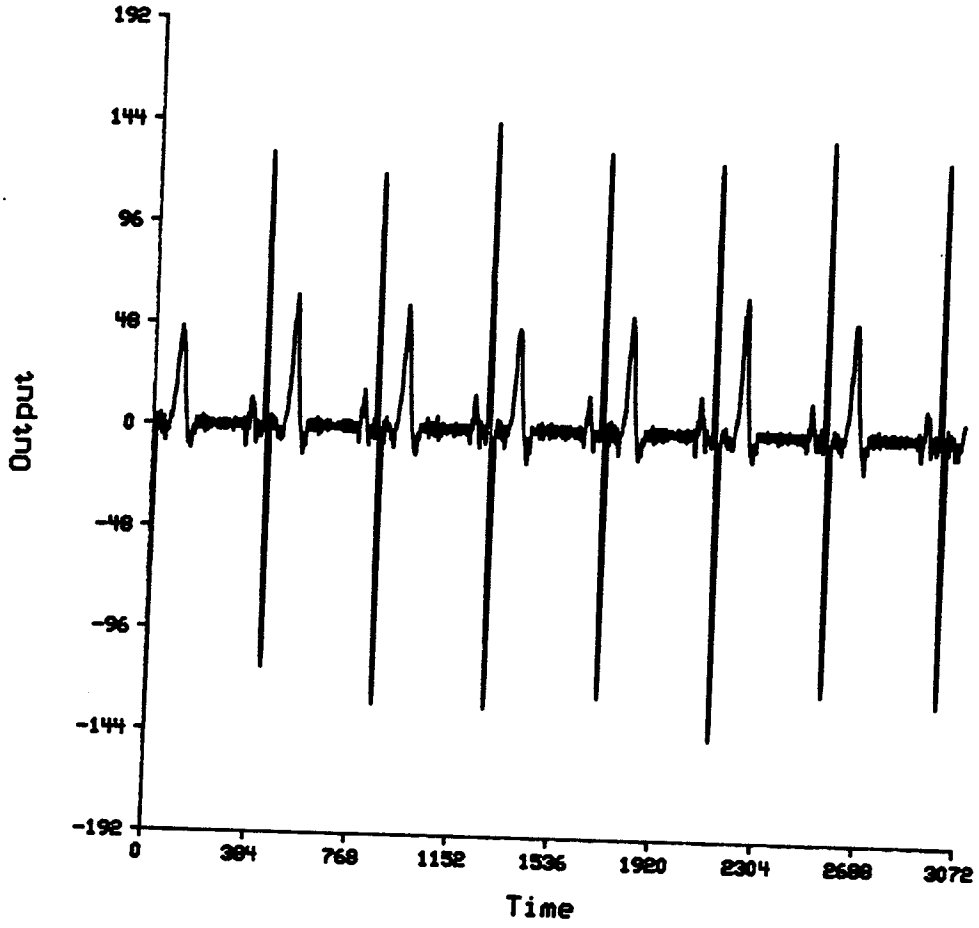


Figure 5.42 Result after baseline correction and noise suppression on the data sequence shown in Figure 5.38.

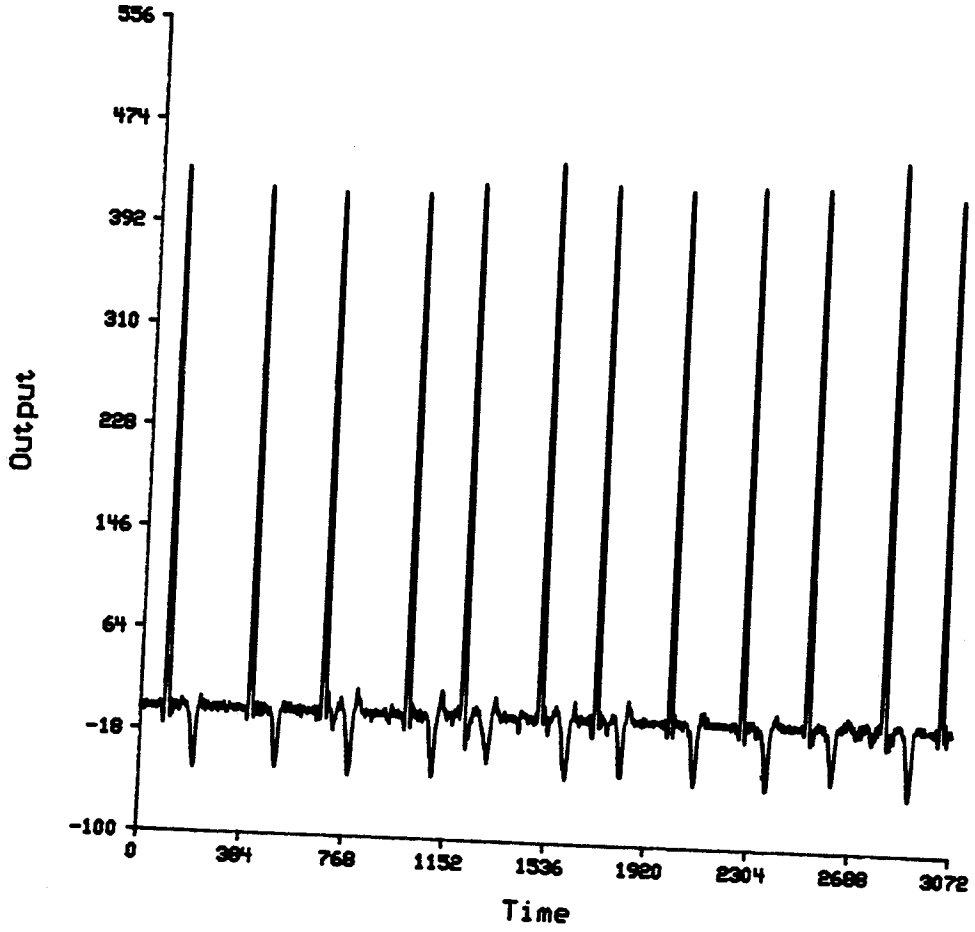


Figure 5.43 Result after baseline correction and noise suppression on the data sequence shown in Figure 5.39.

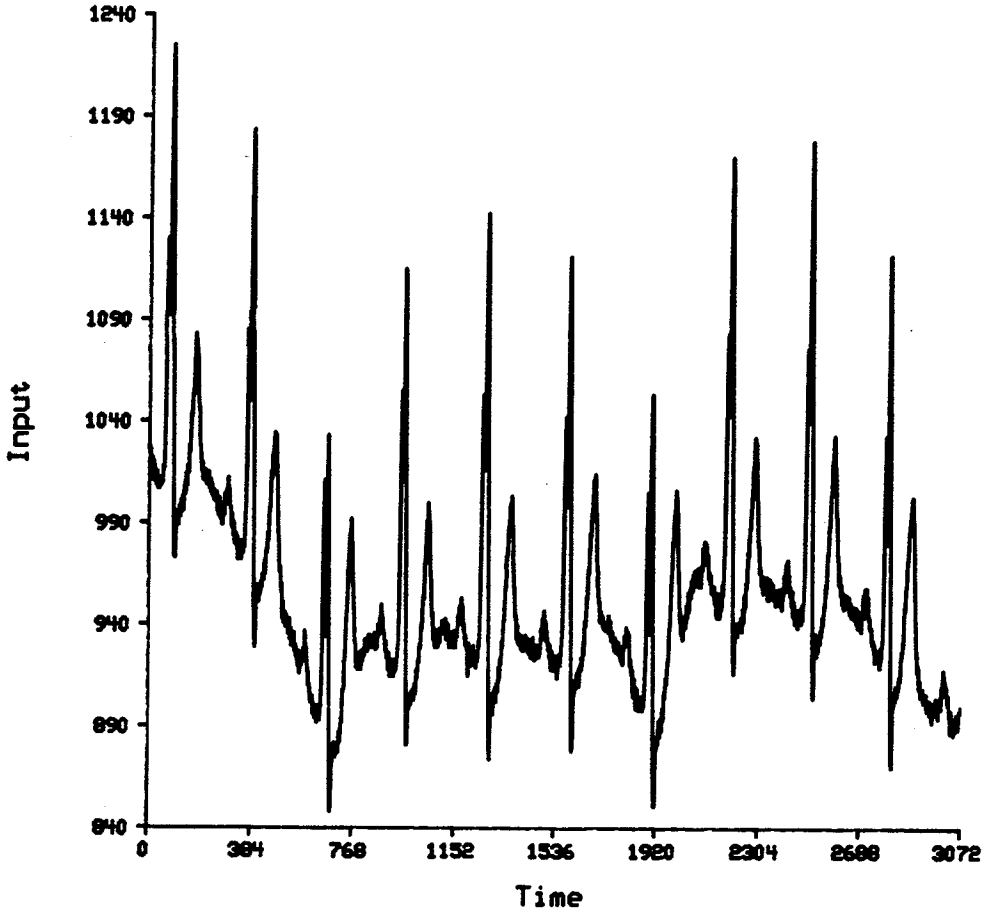


Figure 5.44 A sequence of EKG signal showing baseline wander, from Tape 111 of the MIT-BIH Database.

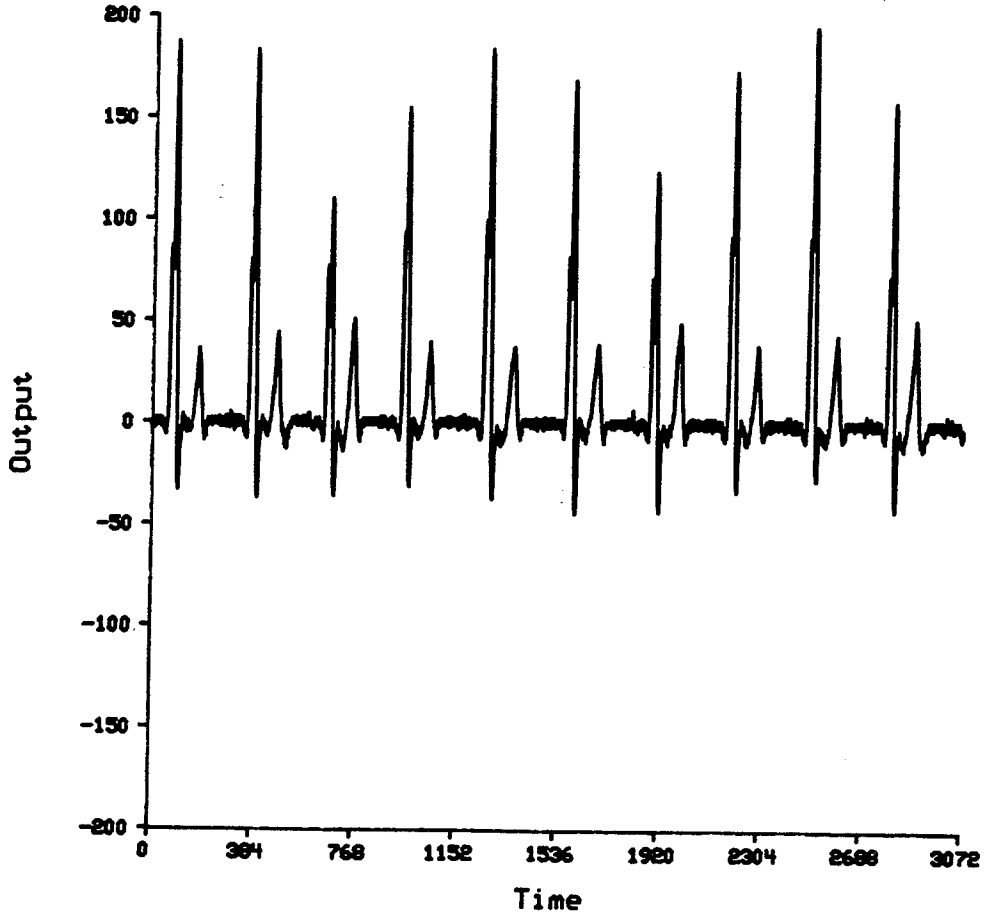


Figure 5.45 Result after baseline correction and noise suppression on the data sequence shown in Figure 5.44.

5.9. Concluding Remarks

A new approach to EKG signal processing is presented in this chapter using mathematical morphological operators. The effectiveness of the new algorithm in impulsive noise suppression and background normalization was first demonstrated by using a corrupted known signal and measuring the difference between the known signal and the processed result. The performance was further examined by processing clinically acquired EKG data.

The performance of the algorithm is dependent on three related factors: amount of noise, choice of structuring element, and sampling rate of the signal. Results in Section 5.5 show that a 5-element structuring element can handle a large amount of impulsive noise. In Section 5.6, a structuring element parameterized by its width, height, and shape was used to study the effect of each of them on noise suppression performance. The most important factor was found to be the length of the structuring element. This result is consistent withthe use of median filters relative to EKG data [Yu85].

The morphological operators are also attractive for their relatively simple computational demands: the computation is made up of addition, subtraction, and logical comparison. It has been shown that these types of algorthms can be implemented in VLSI for real-time processing [Coy86]. While morphological operators have not yet been as widely used in one-dimensional signal processing as in image processing, they seem particularly well suited for processing EKG data, since the signal components are well characterized by their shapes.

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

In the context of developing a system capable of monitoring the heart wall motion and change in two-dimensional echocardiograms, we have presented work in two different areas: computer vision and morphological signal processing. In this chapter, we shall take a critical look at our work in these two fields and suggest future work.

Extensions to the algorithm reported in Chapter 2 would be based on symbolic manipulation. A natural extension would be to automatically search for the boundaries in the next frame based on the detected boundaries in the current frame. Some issues that need to be addressed are: given that the boundary in the current frame may not be correct, how to evaluate the boundary and store the information; how to handle conflicts between boundaries detected in different frames. Although the amount of data that would be needed will be increased substantially, it would probably be easier for an echocardiogram understanding system to be concerned with the entire image, instead of concentrating on the heart wall boundaries.

The work reported in Chapters 3 and 4 has much potential to be further explored. Theoretical bounds to the performance of the TLS based image flow approach is one of many topics worth considering. Our experience is that although the TLS based algorithm is sensitive to the input image data, in part through modeling the image data as polynomials in the gradient estimators, the TLS fitting component of the algorithm is an appropriate tool for solving the image flow equation. We note that none of the image flow results reported in the literature have successfully demonstrated applications to real image data with general motion. In the early processing stage of computer vision, the solutions to many problems are tremendously underconstrained. It seems almost inevitable that image segmentation and image motion analysis must complement each other. It would be interesting to examine how each can make use of results from the other.

Our work in processing the EKG data using morphological operators has applications in a wide range of other domains. Some topics worth exploring are: how to reconstruct a signal after it has been passed through different morphological operators; how to use an adaptive structuring element, one perhaps similar to the parametric structuring element introduced in Chapter 5, to eventually change its shape as it adapts to the noise or signal characteristics.

LIST OF REFERENCES

LIST OF REFERENCES

[Ahl85] M. L. Ahlstrom and W. J. Tompkins, "Digital filters for real-time ECG signal processing using microprocessors," *IEEE Trans. on Biomedical Engineering*, vol. BME-32, no. 9, pp. 708-713, September 1985.

[Als85] J. A. van Alste and T. S. Schilder, "Removal of base-line wander and power-line interference from the ECG by an efficient FIR filter with a reduced number of taps," *IEEE Trans. on Biomedical Engineering*, vol. BME-32, no. 12, pp. 1052-1060, December 1985.

[Arn83] R. D. Arnold, *Automated Stereo Perception,* Report No. STAN-CS-83-961, Dept. of Computer Science, Stanford University, March 1983.

[Bal81] D. H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes," *Pattern Recognition*, vol. 13, no. 2, pp. 111-122, 1981.

[Bas86] S. C. Bass, *Some Introductory Comments to the Users of the MIT-BIH Arrythmia Database,* School of Electrical Engineering, Purdue University, West Lafayette, Indiana, September 1986. Internal document.

[Bin81] T. O. Binford, "Inferring surfaces from images," *Artificial Intelligence*, vol. 17, pp. 205-244, 1981.

[Bri83] J. F. Brinkley, *Ultrasonic Three-dimensional Organ Modelling,* Technical Report No. CS-1001, Dept. of Computer Science, Stanford University, 1983.

[Bud83] A. J. Buda, E. J. Delp, C. R. Meyer, J. M. Jenkins, D. N. Smith, F. L. Bookstein, and B. Pitt, "Automatic computer processing of digital 2-dimensional echocardiograms," *The American Journal of Cardiology*, vol. 52, pp. 384-389, August 1983.

[Bud85] A. J. Buda and E. J. Delp, eds., *Digital Cardiac Imaging,* Martinus-Nijhoff, the Hague, the Netherlands, 1985.

[Can86] J. F. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698, November 1986.

[Cas77] D. Casasent and D. Psaltis, "New optical transforms for pattern recognition," *Proceedings of the IEEE*, vol. 65, no. 1, pp. 77-84, January 1977.

[Chu86] C. H. Chu, E. J. Delp, and A. J. Buda, "Detecting left ventricular endocardial and epicardial boundaries by digital two-dimensional echocardiography," in *Proc. Computers in Cardiology*, pp. 393-396, Boston, Mass., October 1986.

[Col86] S. M. Collins and D. J. Skorton, eds., in *Cardiac Imaging and Image Processing*, p. 190, McGraw-Hill, New York, NY, 1986.

[Coy86] E. J. Coyle, "The theory and VLSI implementation of stack filters," in *VLSI Signal Processing II*, IEEE Press, New York, N.Y., November 1986.

[Coy88] E. J. Coyle, "Rank order operators and the mean absolute error criterion," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, no. 1, pp. 63-76, January 1988.

[Del82] E. J. Delp, A. J. Buda, M. R. Swastek, D. N. Smith, J. M. Jenkins, C. R. Meyer, and B. Pitt, "The analysis of two-dimensional echocardiograms using a time varying image approach," in *Proc. Computers in Cardiology*, pp. 391-394, Seattle, Wash., 1982.

[Eat79] L. W. Eaton, W. L. Maughan, A. A. Shoukas, and J. L. Weiss, "Accurate volume determination in the isolated ejecting canine left ventricle by two-dimensional echocardiography," *Circulation*, vol. 60, no. 2, pp. 320-326, August 1979.

[Eic85] P. H. Eichel and E. J. Delp, "Sequential edge detection in correlated random fields," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 14-21, San Francisco, Calif., 1985.

[Eic86] P. H. Eichel, E. J. Delp, K. Koral, and A. J. Buda, "A method for fully automatic definition of coronary arterial edges from cineangiograms," in *Proc. Computers in Cardiology*, pp. 201-204, Boston, Mass., October 1986.

[Eze85] A. Ezekiel, E. V. Garcia, J. S. Areeda, and S. R. Corday, "Automatic and intelligent left ventricular contour detection from two-dimensional echocardiograms," in *Proc. Computers in Cardiology*, pp. 261-264, Linkoping, Sweden, September 1985.

[Fen79] C. L. Fennema and W. B. Thompson, "Velocity determination in scenes containing several moving objects," *Computer Graphics and Image Processing*, vol. 9, pp. 301-315, 1979.

[Fra81] L. E. Franks, *Signal Theory,* Dowden and Culver, Stroudsburg, Penn., 1981.

[Gal81] N. C. Gallagher and G. L. Wise, "A theoretical analysis of the properties of median filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, no. 6, pp. 1136-1141, December 1981.

[Gar88] E. Garcia-Melendo and E. J. Delp, *The Use of Image Processing Techniques for the Analysis of Echocardiographic Images,* Tech. Rep. TR-EE 88-29, School of Electrical Engineering, Purdue University, West Lafayette, Ind., July 1988.

[Gar85] E. V. Garcia and A. Ezekiel, "Digital processing in cardiac imaging," *International Journal of Cardiac Imaging*, vol. 1, no. 1, pp. 3-27, 1985.

[Gei82] E. A. Geiser, M. Ariet, D. A. Conetta, S. M. Lupkiewicz, L. G. Christie, and C. R. Conti, "Dynamic three-dimensional echocardiographic reconstruction of the intact human left ventricle: technique and initial observations in patients," *American Heart Journal*, vol. 103, no. 6, pp. 1056-1065, 1982.

[Gol83] G. Golub and C. Van Loan, *Matrix Computations,* Johns Hopkins University Press, Baltimore, Maryland, 1983.

[Gri85] W. E. L. Grimson and E. C. Hildreth, "Comments on 'Digital step edges from zero crossings of second directional derivative'," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, no. 1, pp. 121-127, January 1985.

[Har84] R. M. Haralick, "Digital step edges from zero-crossings of second directional derivative," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 1, pp. 58-68, January 1984.

[Har85] R. M. Haralick, "Author's reply," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-7, no. 1, pp. 121-127, January 1985.

[Har87] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology: part I," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 4, pp. 532-550, July 1987.

[Hay83] S. M. Haynes and R. C. Jain, "Detection of moving edges," *Computer Vision, Graphics, and Image Processing*, vol. 21, pp. 345-367, 1983.

[Her78] G. T. Herman and H. K. Liu, "Dynamic boundary surface detection," *Computer Graphics and Image Processing*, vol. 7, pp. 130-138, 1978.

[Hil83] E. C. Hildreth, *The Measurement of Visual Motion,* The MIT Press, Cambridge, Mass., 1983.

[Hor81] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185-203, 1981.

[Ins82] Biomedical Engineering Center for Clinical Instrumentation, *MIT-BIH Arrhythmia Database Tape Directory and Format Specification,* Harvard University-Massachusetts Institute of Technology, Division of Health Sciences and Technology, Cambridge, Mass., October 1982.

[Jac87] L. Jacobson and H. Wechsler, "Derivation of optical flow using a spatiotemporal-frequency approach," *Computer Vision, Graphics, and Image Processing*, vol. 38, pp. 29-65, 1987.

[Jai87] R. Jain, S. L. Bartlett, and N. O'Brien, "Motion stereo using ego-motion complex logarithmic mapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 3, pp. 356-369, May 1987.

[Kas83] B. G. Kashef and A. A. Sawchuk, "A survey of new techniques for image registration and mapping," in *Applications of Digital Image Processing VI*, ed. A. G. Tescher, Proc. SPIE 432, pp. 222-239, 1983.

[Kna76] C. H. Knapp and G. C. Clifford, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-24, no. 4, pp. 320-327, August 1976.

[Low84] D. G. Lowe, *Perceptual Organization and Visual Recognition*, Report No. STAN-CS-84-1020, Dept. of Computer Science, Stanford University, September 1984.

[Mar87a] P. Maragos and R. W. Schafer, "Morphological filters—part I: their set-theoretic analysis and relations to linear shift-invariant filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-35, no. 8, pp. 1153-1169, August 1987.

[Mar87b] P. Maragos and R. W. Schafer, "Morphological filters—part II: their relations to median, order-statistic, and stack filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-35, no. 8, pp. 1170-1184, August 1987.

[Mar79] D. Marr and E. C. Hildreth, *Theory of Edge Detection*, A.I. Memo No. 518, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, April 1979.

[Mar82] D. Marr, *Vision*, p. 212, Freeman and Co., San Francisco, Calif., 1982.

[Mar86] D. M. Martinez, *Model-Based Motion Estimation and Its Application to Restoration and Interpolation of Motion*

Pictures, Ph.D. Dissertation, Dept. of Elec. Eng. and Comp. Sci., Mass. Inst. of Tech., Cambridge, Mass., 1986.

[McK85] D. M. McKeown, Jr., W. A. Harvey, Jr., and J. McDermott, "Rule-based interpretation of aerial imagery," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-7, no. 5, pp. 570-585, September 1985.

[Moo84] G. B. Moody, W. K. Muldrow, and R. G. Mark, "A noise stress test for arrhythmia detectors," in *Computers in Cardiology,* pp. 381-384, Park City, Utah, September 1984.

[Mor77] H. P. Moravec, "Toward automatic visual obstacle avoidance," in *Proc. 5th International Joint Conference on Artificial Intelligence,* p. 584, Cambridge, Mass., August 1977.

[Mos81] H. Mostafavi, T. L. Steding, F. W. Smith, and R. S. Poulsen, "Optimum windows for image registration," *IEEE Transactions on Aerospace and Electronic Systems,* vol. AES-17, no. 1, pp. 101-109, January 1981.

[Nag83a] H.-H. Nagel, "Overview on image sequence analysis," in *Image Sequence Processing and Dynamic Scene Analysis,* ed. T. S. Huang, pp. 2-39, Springer-Verlag, Heidelberg Berlin, 1983.

[Nag83b] H.-H. Nagel, "Displacement vectors derived from second-order intensity variations in image sequences," *Computer Vision, Graphics, and Image Processing,* vol. 21, pp. 85-117, 1983.

[Nee88] J. Neejarvi, P. Heinonen, and Y. Neuvo, "Sine wave responses of median type filters," in *Proceedings of the 1988 IEEE International Symposium on Circuits and Systems,* pp. 1503-1506, Helsinki, Finland, June 1988.

[Nil80] N. J. Nilsson, *Principles of Artificial Intelligence,* Morgan Kaufmann Publishers, Los Altos, Calif., 1980.

[Oht85] Y. Ohta, *Knowledge-based Interpretation of Outdoor Natural Color Scenes,* Pitman Publishing, Marshfield, Mass., 1985.

[Pah84] O. Pahlm and L. Sornmo, "Software QRS detection in ambulatory monitoring—a review," *Medical and Biological Engineering and Computing,* vol. 22, pp. 289-297, July 1984.

[Pah87] O. Pahlm and L. Sornmo, "Data processing of exercise ECG's," *IEEE Trans. on Biomedical Engineering*, vol. BME-34, no. 2, pp. 158-165, February 1987.

[Pip75] H. V. Pipberger, R.C. Arzbaecher, A. S. Berson, S. A. Briller, D. A. Brody, N. C. Flowers, D. B. Geselowitz, E. Lepeschkin, G. C. Oliver, O. H. Schmitt, and M. Spach, "Recommendations for standardization of leads and of specifications for instruments in electrocardiography and vectorcardiography: report of the Committee on Electrocardiography, American Heart Association," *Circulation*, vol. 52, pp. 11-31, 1975.

[Roa79] J. W. Roach and J. K. Aggarwal, "Computer tracking of objects moving in space," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, pp. 127-135, 1979.

[Roo87] W. L. Root, "Ill-posedness and precision in object-field reconstruction problems," *Journal of the Optical Society of America*, vol. 4, no. 1, pp. 171-179, January 1987.

[Sch79] R. B. Schudy and D. H. Ballard, "A computer model for extracting moving heart surfaces from four-dimensional cardiac ultrasound data," in *Proc. 6th Conference on Computer Applications in Radiology and Computer-aided Analysis of Radiological Images*, pp. 366-376, Newport Beach, Calif., June 1979.

[Sch85] B. G. Schunck, "Image flow: fundamentals and future research," in *Proc. of the 3rd Workshop on Computer Vision Representation and Control*, pp. 560-571, Bellaire, Mich., October 1985.

[Ser82] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, New York, 1982.

[Sha79] K. S. Shanmugam, F. M. Dickey, and J. A. Green, "An optimal frequency domain filter for edge detection in digital pictures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-1, no. 1, pp. 37-49, January 1979.

[SkB85] M. M. Skolnick and D. Butt, "Cellular array algorithms for the analysis of EKG signals," in *Proc. 1985 IEEE Workshop on Computer Architectures in Pattern Analysis and Image Database Management*, pp. 438-443, Miami, Fla., November 1985.

[Sko81] D. J. Skorton, C. A. McNary, J. S. Child, F. C. Newton, and P. M. Shah, "Digital image processing of two-dimensional echocardiograms: identification of endocardium," *American Journal of Cardiology*, vol. 48, pp. 479-486, September 1981.

[Sko85] D. J. Skorton, S. M. Collins, E. Garcia, E. A. Geiser, W. Hillard, W. Koppes, D. Linker, and G. Schwartz, "Digital signal and image processing in echocardiography," *American Heart Journal*, vol. 1, no. 6, pp. 1266-1283, December 1985.

[Sta86] S. A. Stansfield, "ANGY: a rule-based expert system for automatic segmentation of coronary vessels from digital subtracted angiograms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 2, pp. 188-199, March 1986.

[Tam85] S. Tamura, S. Nakano, M. Matsumoto, T. Shimazu, M. Fujiwara, T. Matsuyama, and P. Hanrath, "Three-dimensional reconstruction of echocardiograms based on orthogonal sections," *Pattern Recognition*, vol. 18, no. 2, pp. 115-124, 1985.

[Ten77] J. M. Tenenbaum and H. G. Barrow, "Experiments in interpretation-guided segmentation," *Artificial Intelligence*, vol. 8, pp. 241-274, 1977.

[Tor86] V. Torre and T. Poggio, "On edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 2, pp. 147-163, March 1986.

[Tsa84] R. Y. Tsai and T. S. Huang, "Uniqueness and estimation of three-dimensional motion parameters of rigid objects and curved surfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 1, pp. 13-26, January 1984.

[Tuc85] D. M. Tucker, W. Siler, V. G. Powell, and A. W. H. Stanley, Jr., "FLOPS: a fuzzy expert system used in unsupervised

echocardiograms analysis," in *Proc. Computers in Cardiology*, pp. 341-344, Linkoping, Sweden, September 1985.

[Ull79] S. Ullman, *The Interpretation of Visual Motion*, The MIT Press, Cambridge, Mass., 1979.

[Ver79] P. W. Verbeek and S. Lobregt, "The application of 3-d logical neighbour operators to sequences of 2-d ultrasound images," in *Proc. 6th Conference on Computer Applications in Radiology and Computer-aided Analysis of Radiological Images*, pp. 362-365, Newport Beach, Calif., June 1979.

[War70] J. Wartak, *Computers in Electrocardiography*, C. C. Thomas, Springfield, Ill., 1970.

[Web83] J. A. Webb and J. K. Aggarwal, "Shape and correspondence," *Computer Vision, Graphics, and Image Processing*, vol. 21, pp. 145-160, 1983.

[Wen86] P. D. Wendt, E. J. Coyle, and N. C. Gallagher, "Stack filters," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, no. 4, pp. 898-911, August 1986.

[Yak78] Y. Yakimovsky and R. Cunningham, "A system for extracting three-dimensional measurements from a stereo pair of TV cameras," *Computer Graphics and Image Processing*, vol. 7, pp. 195-210, 1978.

[Yu85] B. C. Yu, C. S. Liu, M. Lee, C. Y. Chen, and B. N. Chiang, "A nonlinear digital filter for cardiac QRS complex detection," *Journal of Clinical Engineering*, vol. 10, no. 3, pp. 193-201, 1985.

[Zha84] L.-F. Zhang and E. A. Geiser, "An effective algorithm for extracting serial endocardial borders from 2-dimensional echocardiograms," *IEEE Transactions on Biomedical Engineering*, vol. BME-31, no. 6, pp. 441-447, June 1984.

[Zol87] M. D. Zoltowski, "Signal processing application of the method of total least squares," in *Proc. 21st Annual Asilomar Conf. on Signals, Systems and Computers*, Pacific Grove, Calif., November 1987.

VITA

VITA

Chee-Hung Henry Chu was born on 31 August 1959 in Happy Valley, Hong Kong. He received a B.S.E. degree in Computer Engineering in 1981 and his M.S.E. degree in Computer, Information and Control Engineering in 1982, both from the University of Michigan, Ann Arbor. He enrolled at Purdue University in 1984 to pursue a doctoral degree in electrical engineering and to find something that is worth doing into the next century.

Mr. Chu is a member of the Institute of Electrical and Electronics Engineers, the Optical Society of America, the Association for Computing Machinery, American Institute of Physics, Eta Kappa Nu, and Tau Beta Pi.