The Pennsylvania State University
The Graduate School
Department of Mechanical and Nuclear Engineering

EXTREMA FEATURES FOR GLOBAL-LOCALIZATION AND PATTERN MATCHING OF TIME-SERIES DATA

A Dissertation in
Mechanical Engineering
by
Pramod K. Vemulapalli

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2012

The thesis of Pramod K. Vemulapalli was reviewed and approved* by the following:

Dr. Sean N. Brennan
Associate Professor of Mechanical Engineering
Dissertation Advisor, Chair of Committee

Dr. Henry J. Sommer III
Professor of Mechanical Engineering

Dr. Asok Ray
Distinguished Professor of Mechanical Engineering

Dr. Vishal Monga
Assistant Professor of Electrical Engineering

Dr. Robert T. Collins
Associate Professor of Computer science and Engineering

Karen A. Thole
Professor of Mechanical Engineering
Head of the Department of Mechanical and Nuclear Engineering

*Signatures are on file in the Graduate School.

Abstract

This dissertation describes the use of extrema features for the purposes of localization and pattern matching in time-series data. One of the principle contributions of this work is to develop a wavelet based framework to extract extrema features from raw data. The utility of these features is then demonstrated through different applications in pitch and acceleration-based localization. Another major contribution is the formulation of the filter optimization problem to obtain robust extrema as an eigen value problem with a tractable solution. This optimization framework which was conceived and formulated in a 'deterministic' sense was finally extended to the stochastic domain. This extension resulted in interesting theoretical results that show that the filters of the Discrete Sine Transform are the optimal filters for extracting extrema from Gaussian random walk data. Finally, the nature of the optimal filters for the case of general random walk data was also explored.

Table of Contents

List of Figures

List of Tables

Acknowledgments

Dedication

To all the community activists, organizers and social workers who have made this world a better place and the fruits of whose sacrifice and labor I am currently enjoying and to my parents, Ravindra Vemulapalli, and Vasantalata Vemulapalli.

Chapter 1

Introduction to Feature Vectors

1.1 Introduction

The problem of encoding raw data into a set of feature vectors is important in the context of a wide range of pattern recognition problems. While the main area of concentration in this dissertation is the application of extrema features for localization problems, this chapter aims to introduce features in the more general context of dimension reduction and pattern matching. It is important to provide this background as it makes it easier to appreciate how the techniques developed in this dissertation could apply to a wide range of different pattern recognition problems. Raw data is typically encoded into the form of a vector, with a fixed dimension, and this vector is often referred to as the feature vector. A large body of pattern recognition theory assumes that features in the form of fixed dimension vectors are available for analysis [1]. The ability of the features to

capture the essential information in a dataset is an important factor that affects the performance of most pattern recognition algorithms. The discussion in this section is mainly focused towards feature vectors from time series data but the explanations could apply to other types of data as well.

1.2 Tradeoffs in the feature vector encoding methods

Before introducing the different types of feature vector encoding techniques, it is important to go through some of the underlying tradeoffs involved in creating a feature vector. This overview of the tradeoffs will help the reader gain much better insight into the properties of different feature vectors that will be presented in later sections.

1.2.1 Uniqueness versus Robustness

One of the fundamental tradeoffs involved in creating feature vectors is the choice between uniqueness and robustness. The notions of uniqueness and robustness directly correspond to the ideas of 'signal' and 'noise' in a data stream. The ability of a feature vector to encode the 'signal' information determines its uniqueness, while its ability to reject the effects of noise will determine its robustness. For example, a feature vector created by directly using the raw data is very unique

but is not robust because the match of this data to a database is easily affected by noise (example: constant bias, etc). If the feature vector was encoded by using the mean subtracted values of the raw data then the feature vector would be robust to bias variations in the matching process. However, the uniqueness of this new feature vector is reduced because a single parameter has been removed from the data. It is important to note that the concepts of uniqueness and robustness are discussed here in the context of the encoding step of the feature vector and could be used in other contexts as explained below. The tradeoff between uniqueness and robustness could also arise in certain preprocessing steps in the development of a feature vector. For example, a common preprocessing step is to transform the given signal by using different transforms (Fourier, Wavelet etc) and to use the resulting coefficients in the feature vector. In those situations, one could optimize the preprocessing phase such that it provides more unique and robust coefficients which will in turn lead to better feature vectors.

1.2.2 Computation versus Dimensionality (Better Accuracy)

The dimensionality of feature vector has an effect on the uniqueness of a feature vector. Higher dimensional feature vectors are expected to be more unique, because of the additional volume created in the search space because of their high dimensionality. One of the main drawbacks of using higher dimensional features

is the additional computational and memory requirements needed to handle those features. These problems associated with high dimensional features is often referred to as the 'curse of dimensionality'[2]. Given the tradeoff, it is important to choose a feature vector with the right number of dimensions for a particular application.

1.3 A general overview of the feature encoding method

This section provides an overview of the general procedure that is followed in creating a feature vector. The first step involves preprocessing the raw data to obtain certain parameters of interest and the second step involves encoding those parameters into a feature vector. This process is illustrated by a schematic in Figure 1.1.

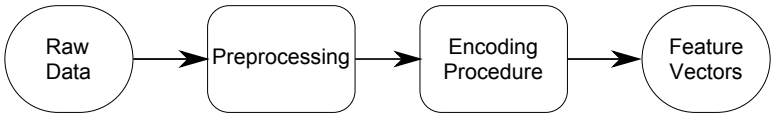

Figure 1.1: A procedural overview of creating feature vectors.

1.4 Preprocessing methods

Given the importance of feature vector representation in pattern recognition there are a wide range of preprocessing methods that have been proposed to address

this problem. Some of these techniques have been presented below to stand as representatives for large families of variants that can be derived based on them.

1.4.1 Raw data as a feature vector

The most direct method of encoding a feature vector is to utilize the raw data as the feature vector. This method is not usually efficient as it is based on the assumption that the raw data only contains 'signal' information and no 'noise' information. The raw data is usually used as a feature vector in techniques where a more elaborate scheme is utilized as a part of the feature vector comparison scheme. This method of encoding will also result in high dimensional feature vectors which will increase the computational burden. Given the combined drawbacks of high dimensionality and less robustness to noise, dimension reduction techniques such as the ones presented below are often utilized in creating feature vectors.

1.4.2 Quantization

A very standard and straightforward technique to perform dimension reduction is through quantization. Figure 1.2 shows the result of quantization when performed over a certain time interval 't' and amplitude 'a'. In quantization methods, the higher dimensional raw data can be viewed as an analog signal and the quantization technique as a process of digitizing the higher dimensional 'analog' signal. A large body of work in signal processing deals with this very specific problem and different

variants of this type of method have been proposed [3].

1.4.3 Transform Based techniques

In signal processing, a common method to extract 'signal' information is to transform the signal to represent it is a set of coefficients of certain basis functions. Depending on the signal and the basis functions that are used, it may be possible to separate out the 'signal' and the 'noise' components of the original data. One could therefore select the coefficients corresponding to the 'signal' component and use them in encoding a feature vector. Figure 1.3 shows an example in which a signal is transformed using the FFT to obtain the coefficients corresponding to the sinusoidal basis functions [4]. In Figure 1.3 , the coefficients corresponding to the lowest frequencies are used in the encoding process and this could have been done because the coefficients corresponding to higher frequencies contained more 'noise' than 'signal' information. A large set of encoding methods that utilize transformations such as the discrete cosine transform [5], the wavelet transform [6] etc have been proposed in literature.

1.4.4 Extrema Based Techniques

In extrema based techniques, the location and the amplitude of the extrema are extracted during the preprocessing stage and are used to encode the feature vector [7] as shown in Figure 1.4. It is common to filter the signal before identifying the

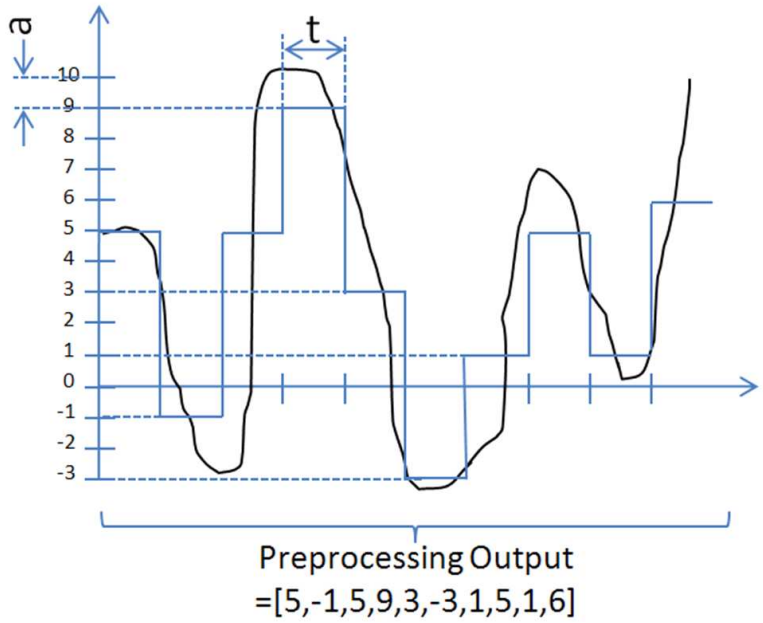

Figure 1.2: Quantization to obtain feature vectors.

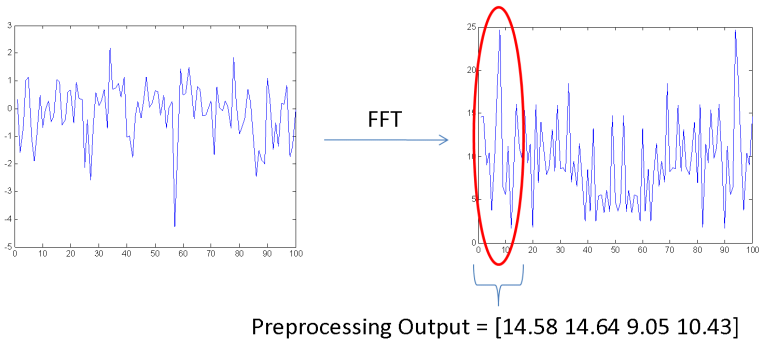

Figure 1.3: Transform based techniques for preprocessing.

locations of the extrema in order to eliminate noise or to create new extrema or for both the previous reasons. In order to choose the most significant extrema in a signal, it is also common to use a parameter that measures the significance of each extrema and selects them by thresholding such a parameter. A common parameter that is used to eliminate insignificant extrema is the amplitude distance between a prospective extrema and its neighboring points.

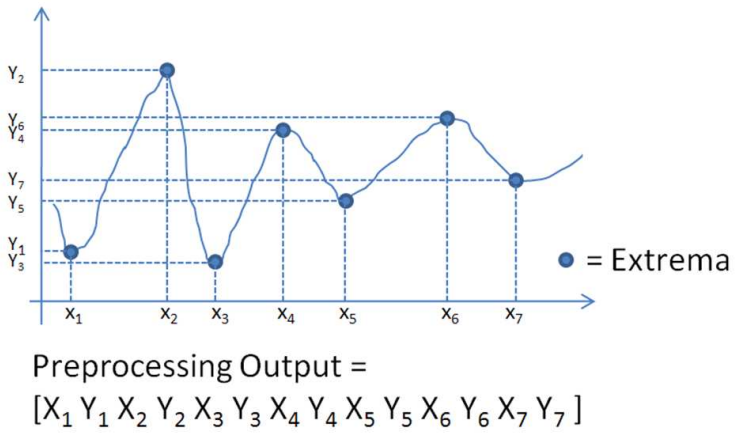

Figure 1.4: Example output of the preprocessing algorithm for extrema features.

1.5 Encoding Methods

The data obtained from the preprocessing stage is usually modified in different ways to obtain a feature vector. The most common modification is to subtract the output of preprocessing step by the mean and divide it by the standard deviation of the data used to generate the feature vector. This normalization step would provide immunity from changes in bias and scale factor variations of a signal such as those shown in Figure 1.5. In some situations, it might be easier to perform the normalization on the input data rather than the output of the preprocessing stage and in those cases the normalization step would be part of the preprocessing stage. In the case of extrema features, it is preferable to encode a feature vector based on the relative distances between the extrema in order to obtain immunity (or invariance) from scale factor and bias distortion. Sometimes the distortions in the signal may be complicated in nature such as a varying bias or scale factor or temporal distortions as shown in Figure 1.5. These distortions may severely affect

the ability of most encoding methods to obtain a proper match. The extrema based methods offer certain advantages in case of such complex distortions because the extrema are usually left intact in spite of these distortions. Distortions such as addition of severe Gaussian random noise tend to alter the extrema present in a signal and could lead to mismatches when the encoding is done with extrema based techniques. In these cases it is customary to filter signal before the extrema extraction process in order to attenuate the noise in the signal.

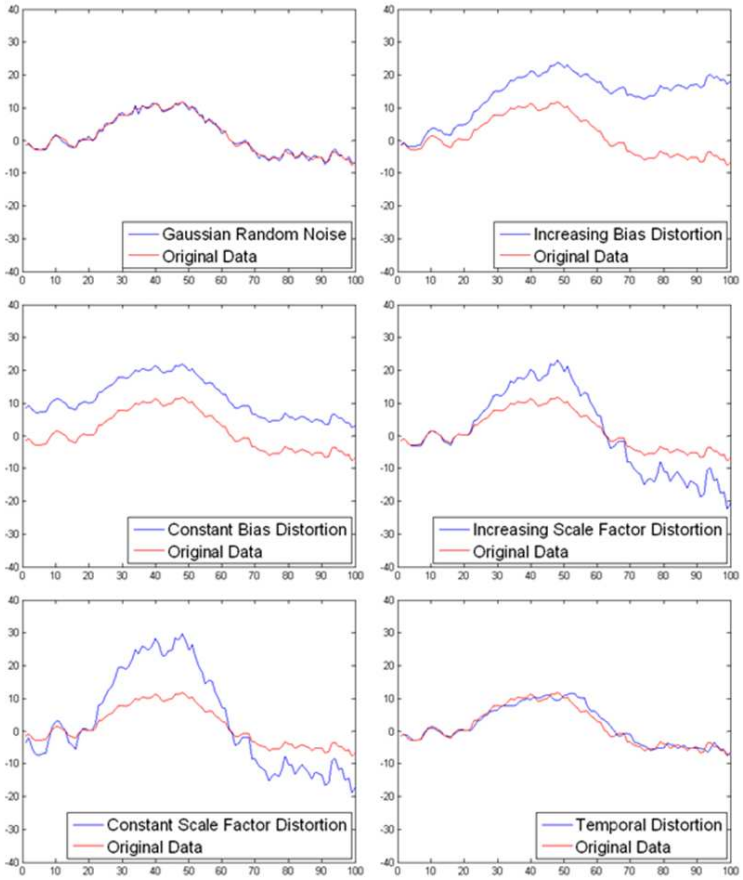

Figure 1.5: Examples of different types of distortions that time series data could experience.

1.6 Encoding Multiple Feature Vectors from Large Datasets

Parameters such as temperature, humidity, stock prices etc tend to vary in a continuous manner and therefore lead to a constant stream of data. In such situations, it is important to generate multiple feature vectors from a single data stream to encode different parts of the data stream. The methods described in section 1.4 assume that only a single feature vector is generated from a time series and so a sliding window method is used to extend these techniques to situations that need analysis of a data stream. In the sliding window method, small portions of the data stream are extracted by windowing the signal and a feature vector corresponding to each windowed signal is evaluated as shown in Figure 1.6. Choosing the sliding window discretization or the step size for the sliding window involves a tradeoff between computation and generating a uniform encoding of a time series. A large step size might make the algorithm susceptible to signals that may start between the individual steps, but a small step size would create a large volume of feature vectors and this would increase the computation. Extrema based methods are able to completely circumvent the need for a sliding window based discretization because the extrema provide natural points to start and stop the encoding process. In the case of the extrema techniques, all sets of a 'fixed number of adjacent extrema' in a signal are used to encode a feature. Another key parameter of the

sliding window method is the size of the window that is used. A small window might result in non-unique features while a large window would not be able to match query signals that are shorter than the window's size. Feature vectors obtained from using different window sizes could be stored and can be used when needed for a particular query signal. Such an approach, though feasible, would increase the memory requirements for the algorithm. It is therefore important to select a window size that would work well for all types of query signals that one may encounter. The extrema features are able to circumvent the limitations of this 'one size fits all' restriction of the sliding window method. This is because a feature is built only if there are adequate numbers of extrema to support it. As the number of extrema in a region of a signal depends on the amount of variation in the signal, the technique automatically has feature vectors that span over large areas of slowly varying parts of a signal and over small areas in the quickly varying regions. Thus the effective 'window size' for each extrema feature is adaptive and is based on the underlying variation in the signal. This adaptive nature of the algorithm also results in efficient computational and memory requirements for the extrema based methods.

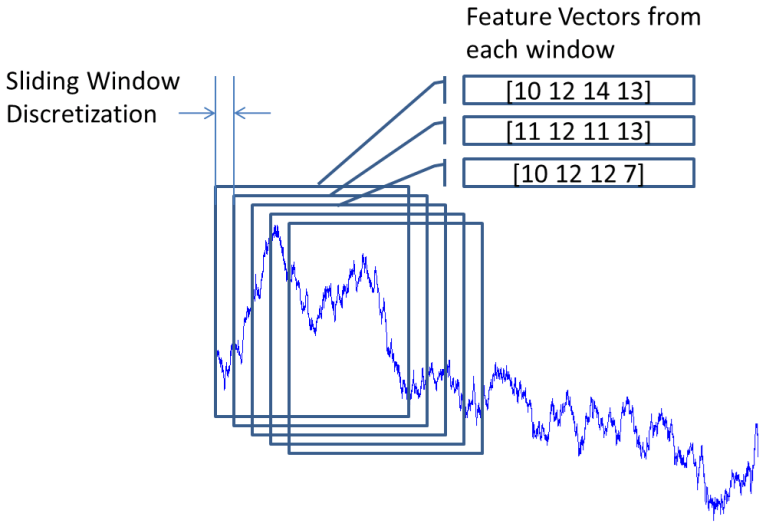

Figure 1.6: Sliding window based encoding technique

1.7 Distance measures for comparing feature vectors

The distance metric that is used to compare two feature vectors is also critical in properly gauging the extent of the match between two feature vectors. The most commonly used distance measure is the Euclidean distance and will be the measure of choice for all extrema based features developed in this dissertation. This subsection briefly describes some of the other distance metrics that are used in comparing time series feature vectors. Most of the non-Euclidean distance metrics are used to compare feature vectors which are obtained by directly encoding raw data as described in section 1.4.1. The Dynamic Time Warping (DTW) distance is the most popular among the non Euclidean distance based metrics because of its capacity to correct for temporal distortion that is observed in many real-life

signals. A comparison of the Euclidean distance and DTW distance for a raw data based feature vector encoding is shown in Figure 1.7. While the Euclidean distance measure shows considerable difference in the signals, the DTW measures [8] shows that the signals are very similar by performing distance calculations after accounting for temporal distortion. The temporal distortion between the two signals can be observed by viewing the gray lines connecting both the signals in Figure 1.7 . Researchers have proposed a host of other distance measures that provide immunity under a variety of signal distortions such as amplitude shifting, temporal distortion etc. Some of the most popular distance measures include the LCSS (Least common subsequence) [9], ERP (Edit distance with penalty) [10] etc and a through overview of the different types of distance measures and their utility on a wide range of datasets has been demonstrated by Keogh [11].

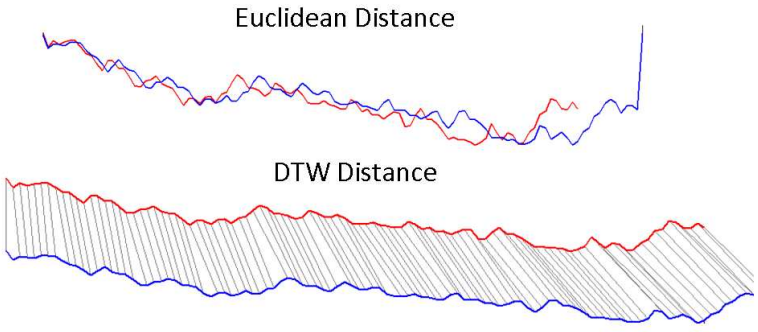

Figure 1.7: A visualization of Euclidean distance and DTW based distance for Raw data based encoding.

1.8 Conclusion

This chapter provides a quick overview of the different types of feature vectors and delves into some of their characteristics. The next chapter introduces the multi scale extrema features that are proposed as a part of this dissertation. The utility of the Multi-scale extrema features is demonstrated in chapter 3, by applying it to solving the problem of global localization using pitch information. The method is also utilized to perform pattern matching using acceleration data as shown in chapter 4. Chapter 5 puts forward a theoretical framework that would allow one to perform optimization to obtain an optimal filter that could be utilized to obtain robust extrema. Chapter 6 provides closed form solutions for the optimal filter in the case where the optimization is performed on Random Walk Data. This document ends with the description of the conclusions and the plans for future work.

Chapter 2

Introduction to Multi-Scale Extrema Features

2.1 Introduction

Given the task of matching a particular time series signal to another time series signal that is not exactly similar, any person would quickly try to match parts or features from one signal to another and then comment on the overall validity of the match. Researchers have utilized this basic human approach as a source of inspiration for their algorithms. The following list summarizes the observations made in literature about the way humans match signals.

1. Humans seem to depend on landmarks in organizing their spatial memory [12]. In [7], it was noted that a relatively successful strategy to duplicate a signal would be to remember its major peaks and valleys (landmarks) and

by joining these points by smooth curves.

2. Humans tend to smooth out the fluctuations from much smaller time scales as noise [7].

3. Humans, when matching two sequences, seem to perform sequential matches of a set of subsequences [13].

4. There is evidence to suggest that human visual systems are sensitive to "primitives" such as tangent and curvature discontinuities [14]. It was also reported in [15] that it is natural to suppose the existence of these primitives or features of second and higher orders, which describe different combinations of first order features, in the visual cortex.

The aim of this chapter is to develop a solution approach that is compatible with the above notions of the way humans match signals, utilizing the mathematical and the algorithmic contributions made in pattern matching. The next section presents a brief overview of the literature in this area. Section 2.3 describes Multi scale extrema features which use a wavelet-based framework to obtain features from time series. The chapter ends with the implications of the proposed framework of time series matching.

2.2 Literature Survey

2.2.1 Commonly used Subsequence Matching Techniques

In [4], Faloutsos posed a very fundamental question regarding time series data. The question is, "Given a certain time series signal (say a "query" signal), what is the procedure to extract similar time series from a signal database?". This problem is often referred to as the subsequence matching problem in literature. It is important to note that the word 'similar' is very critical in the above question. Faloutsos answered this question using Euclidean distance as the definition of similarity. While Faloutsos's work utilized the Discrete Fourier transform, subsequent work using Discrete Wavelet Transform [16], Piecewise Aggregate approximation [17], Adaptive piece wise constant approximation [18] etc used the same paradigm of Euclidean distance between query signal and database signal as the criterion. It was reported that by 2005, more than two dozen techniques based on the Euclidean distance criterion have appeared in literature [19]. Even so, some have found that the distortions that take place in the temporal data impede the techniques that utilize Euclidean distance measure. For example, time warping methods [8] that allow for temporal distortions have been proposed by many researchers. These methods based on dynamic programming have very high computational demands, and some have sought more efficient implementations [20]. In [19] the authors were able to implement Dynamic Time Warping (DTW) via indexing and with

no false dismissals and a relatively tight lower bound. The DTW method of analyzing similarity can also be seen as a modified Euclidean criterion which allows for temporal distortion. Similar to DTW, common subsequence based methods [21, 9] were built to handle noise in the temporal domain but were reported to be more sensitive to amplitude shifting and scaling [22]. While some researchers have built algorithms that can handle amplitude shifting and scaling [6, 23, 22, 24], these methods do not take temporal warping into account. A common underlying characteristic of all these methods is that they utilize a sliding window framework for analysis, and this methodology has certain disadvantages that are elaborated in the next sub section.

2.2.2 Drawbacks of Common Subsequence Matching Techniques

A large majority of the time series data retrieval methods perform full sequence matching [6], that is they assume the signals in the database are of the same size as the query signal. A common approach to achieve sub-sequence matching from these full sequence algorithms is by using the sliding window technique [18, 19] . In this method, a sliding window over the data set is used to create feature vectors at each offset of the window and these feature vectors are efficiently indexed for retrieval purposes [6].The sliding window method has a number of drawbacks, most significantly that it does not extend well to higher dimensions. Given the nature

of this comparison process, one might need to vary the window size in order to accommodate transformations such as temporal scaling in the query signal. While this approach may seem expensive in terms of the computational and memory resources that are required, the one dimensional nature of time series data makes it feasible. The windowing based approach also reduces the flexibility in terms of the length of the query signal that can be used. Any query that is shorter than the prescribed window length would require the entire database to be reevaluated at a finer resolution in order to handle it. Additionally, any query above the prescribed length would mean that certain low frequency features would not be used in the matching process, which in turn leads to higher computation or lower accuracy. Another byproduct of the sliding window approach is that a large amount of processing must be performed, for the sake of mathematical exactness, in order to ascertain which window has the closest match (based on Euclidean criteria or other distance measures) with respect to the query signal. Given the drawbacks of the sliding window approach, there is a clear need to explore alternative notions of similarity between the query signal and the database that may overcome many of the above drawbacks.

2.2.3 The Vision Approach

The problem of Vision-based robot localization [25, 26] and Robust Image Hashing [27, 28, 29] have a number of similarities to the above problem. For example, In

these problems, a query image must be compared to a large set of images present in a database in order to extract those images that closely match the query image. Vision-based localization systems are built to handle large changes due to perspective distortion, scaling, photometric conditions, etc of the query image with respect to the database images. On the other hand, Robust Image Hashing considers matching in the presence of changes due to different compression techniques, shearing, cropping etc. One can see that the above problems are basically higher dimensional versions of the time series matching problem. One can also immediately notice that the Euclidean distance criterion / sliding window approach becomes problematic in the context of image data as it becomes infeasible in terms of the necessary memory and computational resources. The large number of different transformations such as scaling, rotation, perspective distortion, lighting, and occlusions drastically increases the number of different possibilities that have to be covered by a 2-D sliding window. Given these drawbacks, the vision community has developed a feature-based approach to vision based localization and object detection [30, 31]. The feature vectors are designed to be scale, rotation, and/or affine invariant and can be matched in spite of these transformations existing between the query image and the database images. These feature vectors are created in any particular image by identifying unique points (key points) where these features have to be located. A match is established between a query and a database image only if there are sufficient number of feature vector matches, all of which

are spatially coherent [25]. The notion of similarity that is used in this case is the number of feature vector matches that satisfy spatial proximity constraints. This similarity notion is very powerful as it allows for matching to be made under a host of transformations, noisy data and outliers. While most of the vision-based localization papers have used feature vectors that are obtained from local patches at key points [30], some researchers in the image processing community have utilized the relative configuration of the key points with respect to each other to obtain a feature vector [28, 29] . In [28], Monga et.al have used the relative locations of the modulus maxima of the wavelet transform of an image for Robust Image Hashing. This research must be viewed in context of the work done by mathematicians who have shown that the wavelet modulus maxima is a complete and stable signal representation as long as the signal is band limited and the wavelet has compact support [32, 33, 34]. The feature vectors generated from wavelet modulus maxima are thus a much more compact yet complete representation of the signal than the features that are obtained from a sliding window approach.

2.2.4 Feature-Based Approaches in Subsequence Matching

There have been few instances where researchers in different fields have suggested methods for time series data that are analogous to those utilized in vision. Perhaps the closest is in the field of audio retrieval methods [35, 36]. This is a rather special domain-specific time series sub sequence matching problem, and the same problems

as above can be expected. It must be noted that an audio retrieval method that utilizes a feature vector that was based on relative distances between adjacent peaks in the spectrogram of the audio signals was reported in [37, 38] and has certain similarities to the proposed method. It is interesting to note that this algorithm is the technology behind 'Shazham', a popular audio identification app on smart phones. The use of "landmarks", which are the same as extrema, in time series data was suggested in [7] to facilitate full sequence matching. This method is interesting as it allows the matching process even if the query signal is transformed in a number of different ways. The main drawback of this method is that it does not adequately explore the possibility of creating these landmark-based features at multiple spatial resolutions and the possibility of using such techniques for subsequence matching. The use of 'perceptually important points' for subsequence matching was suggested by [39] for stock time series data, but this method ultimately reverted to using a sliding window approach for subsequence matching. Specifically, this paper suggested a sliding window method with different window sizes in order to extract patterns at multiple spatial resolutions. While this approach is computationally taxing, the attempt underlines the importance of extracting features at different resolutions for certain datasets such as stock market data. Both the above methods [39, 7] utilize key points (landmarks/perceptually important points) and suggest further research along the same direction in handling time series data.

2.2.5 Proposed Approach

One of the aims of this work is to utilize this notion of similarity that has been developed by the vision community, and modify it to handle time series data. Specifically, this work seeks to borrow 2-D image processing methods to implement a feature vector based on the relative distance of the wavelet modulus maxima key points with respect to each other within 1-D time series data. The first step in this direction was made by choosing the extrema as the basis for feature encoding. This use of extrema features has the advantages of not requiring a sliding window method and being resilient to severe distortions as described in sections 1.6 and 1.7. In order to enable analysis at multiple resolutions, a wavelet-based approach is utilized. One of the other advantages of this approach is that a signal of any length can be input as a query. This is because the signal is represented in the feature space rather than the signal space and so the matching process is not constrained to using an input signal of a particular length. The whole framework itself is much more robust to noise and outliers as these entities might effect a few features, while the rest of the features generated by algorithm can still be used to obtain a match.

2.3 Multi-Scale Extrema Features

This section provides the details of the proposed method for extrema feature generation.

2.3.1 Algorithm Overview

To frame the problem succinctly, a query signal must be compared to a database of signals in order to obtain the locations of maximum agreement. Implementation of an algorithm to achieve this can be divided into two phases: a Preprocessing Phase and an Online Phase. In the preprocessing phase, the signal-database is processed to obtain feature vectors which are stored in an index database. In the online phase, the query signal is used to create feature vectors which are used in conjunction with the index database in order to identify locations of similar signals in the signal-database. A schematic that illustrates the above process is shown in Figure 2.1.

2.3.2 Extrema Features

This sub-section introduces the 'Multi Scale Extrema Feature', which presents a very efficient wavelet based scheme to generate feature vectors that capture the behavior of the signals over different resolutions. The individual steps involved in generating this feature vector are shown in Figure 2.2 and the corresponding descriptions are given below.

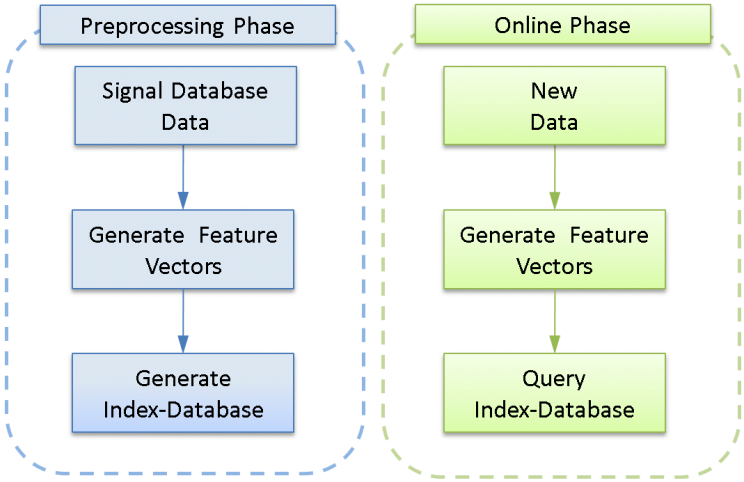

Figure 2.1: The two phases involved in the proposed localization scheme and the central role played by the feature vector in these phases.

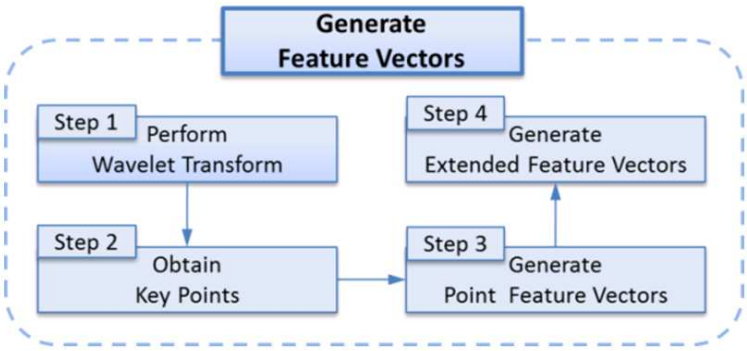

Figure 2.2: The step by step process involved in obtaining the Multi Scale Extrema Features.

1. Wavelet decomposition: To separate high-frequency noise from low-frequency features, wavelet decomposition is performed to partition the signal into its components corresponding to dyadic frequency bands. Next, feature vectors corresponding to each frequency band are computed. This method is used to restrict the effect of frequency-selective noise by limiting it to those feature vectors which have been extracted from the noise-affected frequency bands.

The Wavelet transform is performed by using the so-called "Derivative of Gaussian wavelet" whose Fourier transform is shown below.

$$\hat{\psi} = i\omega e^{-\omega^2/2} \tag{2.1}$$

It has been shown that the wavelet transform [32] is equivalent to a multi scale differential operator.

$$W_f(u,s) = s^n \frac{\mathrm{d}^n (f * \theta_s)(u)}{\mathrm{d}x^n} \tag{2.2}$$

Where the wavelet $\psi(t) = (-1)^n \frac{\mathrm{d}^n \theta(t)}{\mathrm{d}x^n}$ and $\theta(t)$ is typically chosen as the Gaussian function. In the present case n = 1, as we are utilizing the Derivative of Gaussian ('DerGauss') Wavelet. Thus, the output signals, obtained from the wavelet transform, contain peaks corresponding to the high variation points of different Gaussian-smoothed versions of the original signal. Step 1 in Figure 2.3 presents an example of this wavelet transform of a signal.

2. Obtaining Keypoints: Local maxima of the output from the wavelet transform are then selected as candidate "key points". These local maxima are calculated from the wavelet transform at each scale. This implies that, if a local maxima exists at time u_o and scale s_o , then:

$$\frac{\partial W f(u,s)}{\partial u} = 0 \mid (u = u_o, s = s_o) \tag{2.3}$$

These key points are found from the finite-difference implementation of this equation. A heuristic measure for the susceptibility of a certain local maxima to noise is evaluated by measuring its distance to the neighboring peaks. A threshold for this heuristic measure is used to decide if a local maxima is to be designated as a key point.

Step2 in Figure 2.3 shows that the key points of the wavelet transform at different scales. This entire process of taking the wavelet transform and finding the local maxima in the above manner is called 'Wavelet Modulus Maxima' [32]. By encoding the shape information at recognizable key points, this algorithm is able to achieve shift invariance. This procedure does away the need for encoding closely-offset overlapping frames, thus reducing the number of feature vectors required to encode a particular stretch of data.

An underlying assumption in this analysis is that the data is composed of different regions of constant slope (straight lines) and that the key points are the high curvature 'bridge' points between these straight lines. Road data tends to exhibit this effect, at least for the six thousand km measured by the authors.

3. Computing the point feature vector: Once the key points are obtained, the distance of a key point to its adjacent neighbors is used to compute a point feature vector (PFV). By using the neighboring key points, the PFV is able to expand to a scale suited to the underlying variation present in the signal.

Thus, the signal length that is encoded is larger if the key points are far apart because of little variation in the data, and vice versa. This adaptive nature of the proposed PFV enables it to overcome 'the one size fits all' restriction of the sliding window technique. For a one dimensional signal, the distance between two points in that signal is given by distance along the abscissa and the ordinate. Thus, four numerical quantities are required to describe the location of both the neighbors present on each side of a point. Let a and c denote the distance of a key point along the abscissa to each of its neighbors and let b and d be its distance along the ordinate to the same neighbors as shown in step3 of Figure 2.3. Depending on the nature of invariance that one would like to incorporate into the matching process, the PFV at that particular key point could be encoded in a variety of ways as shown in Table 2.1. For example, the first encoding scheme in Table 2.1 allows the PFV to be scale invariant with respect to the input data. Thus, a PFV computed on a signal which is scaled by different amounts along the abscissa and the ordinate will be the same as that computed on the un-scaled signal. As only the relative distances between key points are used to compute the PFV, all the above PFVs are bias invariant. It is important to note that "Time Bias" invariance is not mentioned in the above list as that corresponds to the basic subsequence matching problem and is present in each one of those cases. Both bias and scale errors are commonly encountered in a number of

Table 2.1: Feature vector formulations and the associated types of invariance.

Types of Invariance	**Feature Vector Encoding**
Amplitude Bias Amplitude Scale Time Scale	$\left[\frac{a}{\sqrt{a^2+c^2}} \quad \frac{c}{\sqrt{a^2+c^2}} \quad \frac{b}{\sqrt{b^2+d^2}} \quad \frac{d}{\sqrt{b^2+d^2}} \right]$
Amplitude Bias Amplitude Scale	$\left[\frac{a}{\sqrt{a^2+c^2}} \quad \frac{c}{\sqrt{a^2+c^2}} \quad b \quad d \right]$
Amplitude Bias Time Scale	$\left[a \quad c \quad \frac{b}{\sqrt{b^2+d^2}} \quad \frac{d}{\sqrt{b^2+d^2}} \right]$
Amplitude Bias	$\left[a \quad c \quad b \quad d \right]$

datasets and the point feature vector can be designed to be immune to them.

4. Creating the extended feature vector: Finally, adjacent PFVs are bundled together to create an extended feature vector in order to obtain an adequately unique representation of the shape around the key point. As certain dimensions may be identical between adjacent PFVs, it is beneficial to remove these redundant dimensions to improve the computational performance of the algorithm. Higher dimensions typically imply more computational effort and this phenomenon is often described as the 'Curse of Dimensionality' [2]. Choosing the length of an extended feature vector is a tradeoff between increasing the uniqueness of a feature and restricting the effect of an erroneous key point on

the recognition of its neighborhood. Through implementation, it was found that at least three adjacent features in each extended feature vector are necessary for robust localization. Combining three 'amplitude bias' PFVs and removing their redundant dimensions leads to an '8' dimensional extended feature vector. The extended feature vector can be seen as containing only the first and the third PFV of the three PFVs as the dimensions of the second PFV are present either in the first or third PFV when the 'amplitude bias' formulation is used. An example extended feature vector of this nature is shown in step 4 of Figure 2.3.

2.3.3 Feature Matching

Once the feature vectors are created they are used differently in the preprocessing phase and the online phase.

1. Preprocessing phase: In the pre-processing phase, the extended feature vectors are used to create a KD-tree in order to be able to perform an efficient search through the database of features. As the primary aim of this dissertation is to explore the efficacy of the feature vectors for localization, a generic tree was used for testing the vectors. A more detailed treatment of the various types of tree data structures that can be used for localization is presented in [40, 41]. An interesting new data structure called vocabulary tree [42] has been reported to perform vision based localization very efficiently, and could

easily be extended to the proposed method as well.

2. Online phase: In the online phase, the feature vectors are tested for a match within the database to obtain a match for a particular query signal. Each query signal generates multiple feature vectors that are tested for a match within the database to determine their corresponding position estimate in the signal database. Each position estimate was compiled into a histogram and the position with the highest value in the histogram is output as the best position estimate for a query signal. For applications which need multiple outputs, the histogram can be used to output multiple position estimates by identifying peaks in a sequential manner based upon their height.

2.4 Conclusion

Given the premise that one of the motivations for designing this algorithm is to obtain features that enable a more 'human-like' matching technique, the below table identifies how the algorithm described in the previous section embodies the leads enumerated in section 2.1.

1. • Intuition: Humans rely on landmarks in organizing their spatial memory [12]. In [7], it was noted that a relatively successful strategy to duplicate a signal would be to remember it major peaks and valleys (landmarks) and by joining these points by smooth curves.

- Algorithm: In proposed method key points (wavelet modulus maxima) are identified and the relative distance between these key points become the basis for the feature vector.

2. - Intuition: Humans tend to smooth out the fluctuations from much smaller time scales as noise [7].
   - Algorithm: The proposed uses the wavelet transform and breaks a signal down into dyadic frequency bands, and thus a signal in a low frequency band will have its high frequency content filtered out.

3. - Intuition: Humans, when matching two sequences, seem to perform sequential matches of a set of subsequences [43].
   - Algorithm: The proposed method utilizes multiple feature vectors which span over different parts of the query signal and these multiple feature vectors can be seen as being extracted from different subsequences of the query sequence and be used for the matching process.

4. - Intuition: There is evidence to suggest that human visual systems are sensitive to "primitives" such as tangent and curvature discontinuities [14]. It was also reported in [15] that it is natural to suppose the existence of these primitives or features of second and higher orders, which describe different combinations of first order features, in the visual cortex.

- Algorithm: In the proposed method, small primitives known as the point feature vectors are extracted from the key points and different combinations of these point feature vectors are used to generate the final feature, analogous to the human visual process.

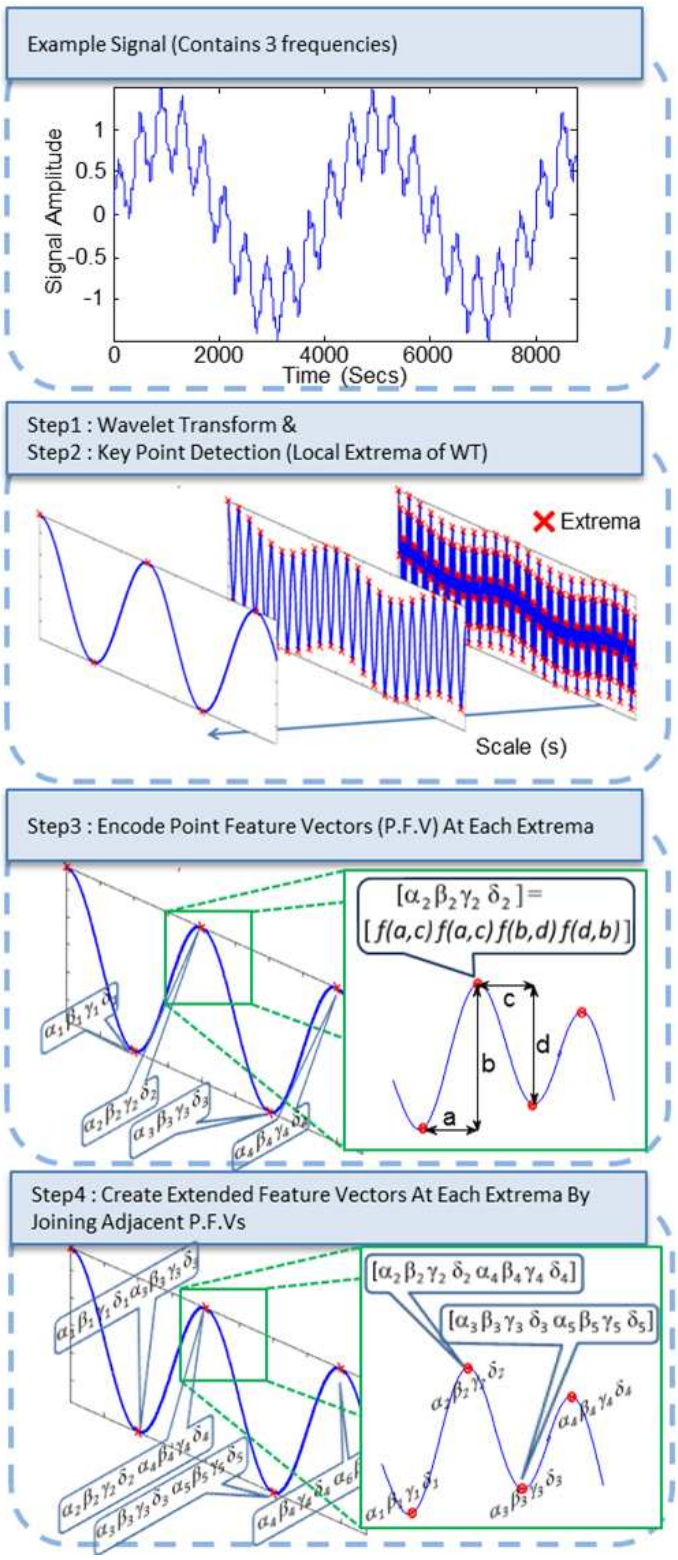

Figure 2.3: The feature vector creation process by example.

Chapter 3

Global Localization Using Pitch Data with Extrema Feature Matching

3.1 Introduction

Advanced vehicle systems such as autonomous vehicles, driver-assist systems, and collision warning systems, etc. all benefit from accurate estimates of vehicle location. While GPS provides position information, it is quite susceptible to intentional attack [44], outages, and signal reception problems. As the GPS signal is weak it could be easily jammed by using relatively lower power jamming equipment. Consequently, there is growing interest to develop alternatives to GPS such as map-based localization techniques [45], including some which rely on onboard LIDAR [46] and vision sensors [47] in order to increase the reliability and robustness of the localization systems. The problem of map-based localization can be broken

down into two phases: global localization and local tracking. Global localization [46, 47] tries to estimate the position of the vehicle in the initial phase, during which it could be present anywhere within a map. Once the vehicle has been localized on a global scale, the second phase, i.e. local tracking, is usually initiated. In local tracking [48, 49] the current position estimate of the vehicle must be determined from previous, nearby position estimates and current sensor information. Popular approaches to local tracking include Particle Filtering (PF) and variants of Kalman Filtering [50], both of which have been implemented by the authors for vehicle localization using pitch information[51, 52]. Global localization tends to be a harder problem to solve than local tracking because maps create a large search space. This requires tremendous computational resources to implement a particle-filter or multiple Kalman Filter solution when the initial vehicle position is completely unknown, thus these "standard" solutions are not practical. The methods for non-GPS global localization methods can be classified into categories based on the type of sensor that is utilized.

3.1.1 Global Localization Using LIDAR

Fox et al [53] pioneered a particle filter approach to global localization and local tracking with the aid of LIDAR sensors in indoor environments. Adaptive techniques for controlling the population of particles [54] were later introduced to make the method more efficient in terms of computation and memory requirements. A

feature-based approach to global localization utilizing LIDAR data was proposed by Bosse et al in the past few years [55, 56]. This LIDAR feature approach has enabled localization in large outdoor environments covering a roadway network of 164 Km.

3.1.2 Global Localization Using Vision Sensors

A wide range of approaches have been proposed for localization using vision sensors, primarily due to their low cost. Dellaert et al [56] used a particle filter to perform global localization and local tracking on an indoor robot using a camera looking at the ceiling. The invention of SIFT features by Lowe [30] greatly simplified matching images taken from different viewpoints. Se et al applied these SIFT features for localization of indoor mobile robots [25]. The SIFT features were then used to perform outdoor localization for an area of 20 km by Schindler et al. Murillo et al [57] proposed the use of the global 'gist descriptor' for localization and demonstrated accurate performance over a stretch of 21 km of an urban area. David et al [58] have proposed the use of satellite images to create the map by encoding 'orientation descriptors' that can then be used to perform matches with images taken from the ground view.

3.1.3 Global Localization Using Other Sensors

Recent research has focused on the developing adhoc vehicular networks (VANETs) [59] and Drawil et al have demonstrated the possibility of improving localization by using those networks [60]. Such wireless communication approaches could be used for global localization in case of faulty sensors but depend on the presence of other vehicles on the road network. The authors have previously demonstrated the possibility of pitch based localization using particle filters [51, 52]. Figure 3.1 provides an illustration of the pitch angle of a road and an INS sensor installed in a vehicle to measure it. The particle filter approach could be used for global localization over small roadway networks but becomes computationally infeasible for larger networks.

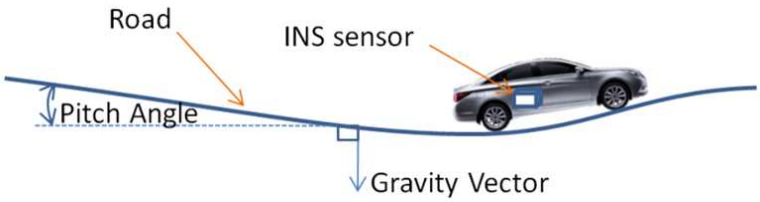

Figure 3.1: The schematic illustrates the pitch angle of the road and an INS sensor installed in a vehicle to measure it.

3.2 Overview

In summation, the previous approaches to global localization have mainly involved the use of LIDAR and vision sensors [46, 47, 50, 61]. While both the above sensors are proven to improve the localization of a vehicle, they typically tend to fail under

rainy and dusty conditions; moreover, vision sensors also tend to be unreliable during poor lighting conditions such as nighttime driving. These sensors are also expensive and can be blocked by dirt or snow.

This work overcomes these problems by proposing the use of road grade data measured from in-vehicle pitch or inertial navigation system (INS) sensors, which are robust under all the above conditions assuming vehicles are operating on known (e.g. mapped) roadways. Among the different global localization methods that have been suggested in literature [46, 47], the proposed method is quite practicable for implementation on roadways as it is comparably simple to compute, and is immune to external signal jamming conditions, e.g. it is "signal free."

The data density per unit distance traveled also affects the sensor choice for localization. This choice is dependent on a trade-off between computation and travel distance for localization. The previously mentioned vision and/or LIDAR approaches [46, 47] utilize high densities of data per unit distance traveled, values several orders of magnitude more than in this work. They hence require higher computational resources to localize, even within a small roadway network. However, these other approaches are advantageous as they require a smaller travel distance before localization, and can simultaneously estimate multiple vehicle state parameters (position and orientation, for example). In contrast, this paper utilizes pitch data which is one dimensional in nature and lane-specific. While pitch data requires only a moderate amount of computation and memory for localiza-

tion within considerably large road networks, the tradeoff is that the vehicle must travel a larger distance before successful localization and only a single parameter (longitudinal roadway location) can be estimated.

In order to implement a global localization scheme with pitch data, a signal retrieval method (based on time series subsequence matching) has been implemented. This approach is similar to recent approaches in vision and LIDAR-based global localization [46, 47] which have utilized techniques from image retrieval [61, 42]. This novel application of pitch information requires new time series subsequence matching tools because of the rather unique distortions and noise typically present in this data. Therefore, the other major contribution of this paper is the 'Multi-Scale Extrema Feature' which is a feature vector that has been specially designed to facilitate pitch data retrieval from vehicles on roads [62, 63]. The feature-based approach that has been developed for global localization could also be applied to local tracking problems [51, 52] to take advantage of the computational and memory benefits that are obtained by using this method. The remainder of this paper presents an algorithm that achieves global localization within very large road networks, using pitch information. Section 3.3 presents the literature survey for current signal retrieval techniques and explains the need to develop a new feature vector that is particularly effective in handling the challenges presented by pitch data. This section also presents the proposed 'Multi-Scale Extrema Feature' and combines it with a KD-tree framework for global localization. A theoretical ap-

proach to selecting the right wavelet is presented in Section 3.4, and its predictions are confirmed with simulation. Section 3.5 shows the algorithm's results in localizing a vehicle's position without initialization within a road network spanning 6000 km. Section Section:chap3SimulationResults demonstrates the algorithm's immunity to typical types of sensor noise. Conclusions then summarize the main results of this work.

3.3 Multi Scale Extrema Features

3.3.1 General overview of feature vector based localization

In this paper, pitch data measured on a vehicle is compared to features stored in a map in order to search for the locations of maximum agreement. Implementation of the localization algorithm can be divided into two phases: a preprocessing phase and an online phase. In the preprocessing phase, mapped data is processed to obtain feature vectors which are stored in a database. In the online phase, data collected on the vehicle is used to create feature vectors which are used in conjunction with the database in order to obtain the location of the vehicle. A schematic that illustrates this process is shown in Figure 3.2. It is can be seen that the feature vector plays a critical role in both phases.

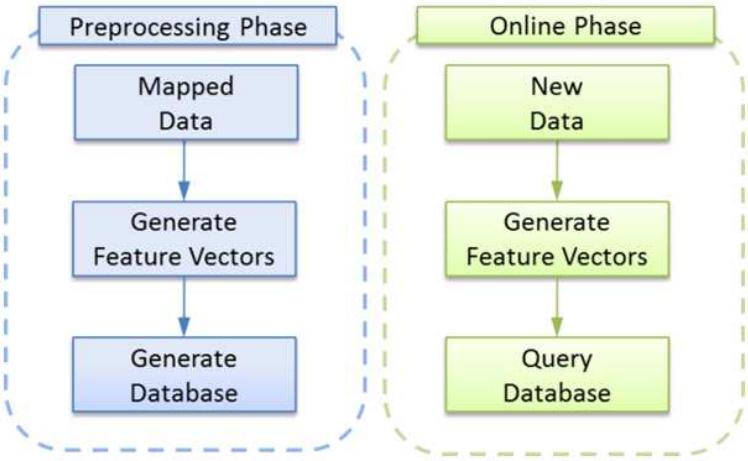

Figure 3.2: The schematic illustrates the pitch angle of the road and an INS sensor installed in a vehicle to measure it.

3.3.2 Previous methods from the literature

One can observe that this map-matching problem is quite analogous to the time series subsequence matching problem, for which a large number of different solutions have been proposed [4, 64, 19, 16]. Unfortunately, there are several drawbacks when applying these approaches to map-based localization. For example, all the above methods use the Euclidean distance or a modified Euclidean distance such as Dynamic Time Warping (DTW) to calculate the distance between the database signals and the query signal. The Euclidean distance criterion and its variants are ill suited for handling data with outliers. In the proposed formulation, the extrema based features are expected to be robust to outliers and temporal distortions. Further, it is difficult to implement a real-time version of many methods proposed in the literature [4, 64, 19, 16]. To create feature vectors in the above techniques, signals are usually sampled at regular intervals which are closely offset

in a technique commonly known as the sliding window method. This technique results in information being stored redundantly across a large number of feature vectors, which will result in unnecessarily large databases when applied to road networks. Some methods such as Longest Common Subsequence (LCSS) [9] are robust to outliers and can be indexed for fast retrieval purposes but still have the drawback of using the sliding window framework like all the above methods. In the sliding window method, the signal length that is used to generate a feature vector is fixed, regardless of the variation within the signal. This 'one-size-fits-all' approach is adequate for signals which exhibit significant variation over "small" time scales (e.g. music). However, for map-based localization, there may be sections of road which are very smooth and which have little variation in pitch information. Thus a fixed length segment of the signal might not create adequately unique feature vectors. The proposed algorithm creates features which span to an appropriate extent based on the variation present in the underlying signal and the method is also able to handle query signals of any length.

3.3.3 Feature Matching

The algorithm's details are provided Once the feature vectors are created they are used differently in the preprocessing phase and the online phase.

1. Preprocessing phase: In the pre-processing phase, the extended feature vectors are used to create a KD-tree in order to be able to perform an efficient

search through the database of features. As the primary aim of this paper is to explore the efficacy of the feature vectors for localization, a generic tree was used for testing the vectors. A more detailed treatment of the various types of tree data structures that can be used for localization is presented in [40, 41]. An interesting new data structure called vocabulary tree [42] has been reported to perform vision based localization very efficiently, and could easily be extended to the proposed method as well.

2. Online phase: In the online phase, the feature vectors are tested for a match within the database to determine their corresponding position estimate for the vehicle. Each query signal generates multiple query feature vectors and each of these feature vectors is matched with the KD-tree database to determine their corresponding position estimates for the vehicle. The position estimate that is obtained from each feature match is then subtracted by the distance in the query signal at which the query feature vector was extracted and this gives an estimate for the location in the database where the query signal could be matched. All such position estimates are compiled into a histogram and the position with the highest value in the histogram is output as the best position estimate for a query signal. For applications in which local tracking follows the global localization scheme, the histogram can be used to output multiple position estimates which can be used to initiate a particle filter or multiple Kalman filters.

3.4 Multi Scale Extrema Analysis

The purpose of this section is to present the analysis that led to the selection of the 'DerGauss' wavelet for generating the extrema from pitch data. In the following experiments, a large set of pitch data (75 Km) are used to create a database. A smaller subset of those signals are corrupted with Gaussian noise and are used to generate features which are then evaluated, both analytically and experimentally, to obtain the probability of them being matched correctly to the database. While there are a wide range of noise sources that can be the basis for analyzing the performance, this section is particularly focused on testing the performance of the matching scheme under the condition that the query signal is corrupted by Gaussian noise as this is a common situation. There are two approaches that can be used to evaluate the performance: Simulation Approach and Theoretical Approach.

3.4.1 Simulation Approach

In this approach, Gaussian noise is added to the query signal, which has been extracted from the database, and is used to perform subsequence matching as described in the previous section. This computation is repeated a number of times in order to calculate the probability of match for features from each scale. The main advantage of this technique is that it is able to simulate complex processes that might be analytically intractable. The main drawback of this method is that

a large number of experiments must be performed in order to get a valid result and that the whole procedure is essentially a 'black box' approach where a query signal is input and the output result is noted, thus giving very little intuition about the underlying process that might be benefiting or degrading the performance.

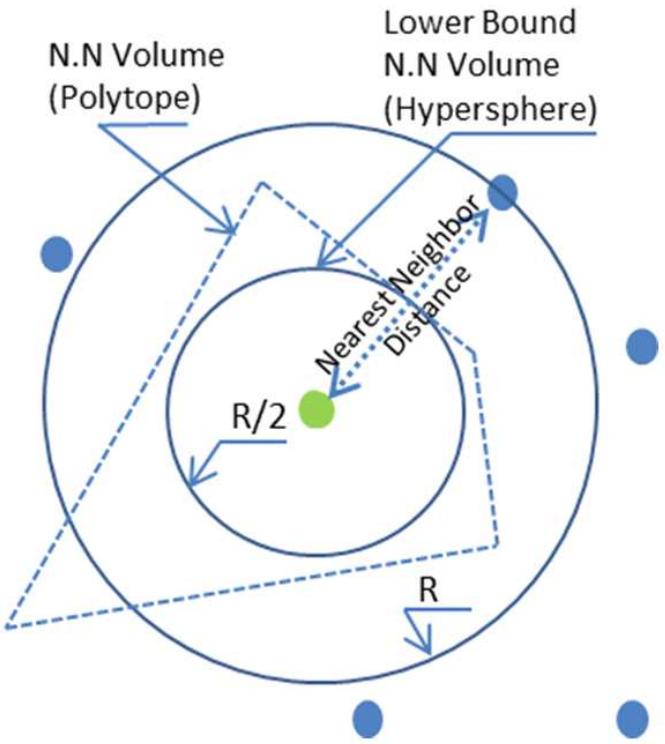

Figure 3.3: A two dimensional projection of a polytope and its corresponding lower bound hypersphere for a given point (green).

3.4.2 Theoretical Approach

In the theoretical approach, one tries to understand the performance of the algorithm from theoretical calculations. The main advantage of this method is that it provides details about the inner workings of the algorithm and can be used to un-

derstand some of the underlying issues. The main disadvantage of this method is that general closed-form solutions to the algorithms are mathematically intractable and usually several approximations are required. Thus, the results obtained from the theoretical method are only as good as the approximations that are made. For example, if either of the first two point feature vectors shown in Table 2.1 in the previous section is used, it becomes mathematically unwieldy to model the distribution of this feature vector, when Gaussian noise is introduced into the query signal. In this case, the only recourse is to use the simulation approach. The other two feature vectors that are given in Table 2.1 can however be modeled under certain approximations and they provide interesting analytical results.

3.4.3 Theoretical analysis for choosing the right wavelet

In this subsection, we attempt to evaluate the relative performance of three different wavelets for the pitch dataset using the theoretical method. The 'Amplitude Bias' point feature vector is used in this analysis. In this case the probability of matching a feature vector, for each wavelet type, depends on two factors:

1. The uniqueness of the feature vectors (or how far each feature vector is from its neighboring features in the database)

2. The robustness of the feature vector, e.g. the immunity of the feature vectors to the introduction of Gaussian noise into the original signal.

A very unique feature vector can be matched correctly in spite of being adversely affected by noise. Similarly, a non-unique feature vector can be matched correctly if it is very robust to noise. The theoretical approach allows us to break down the contribution from both uniqueness and robustness as shown below.

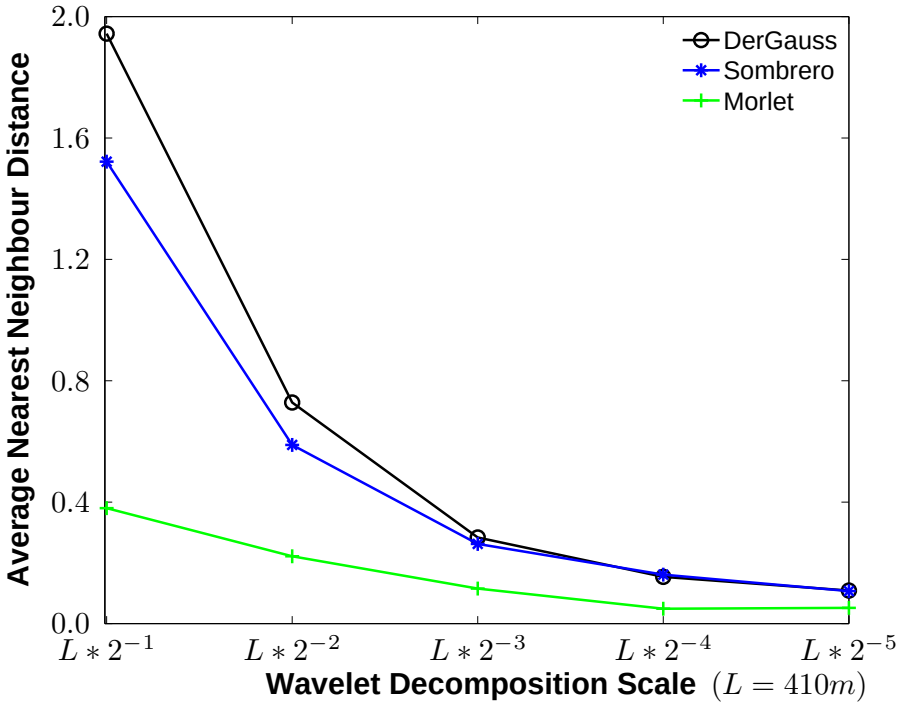

Figure 3.4: The average nearest neighbor distance for feature vectors obtained from different wavelets as a function of their wavelet decomposition scale.

3.4.3.1 Uniqueness of the feature vector

The uniqueness of the feature vector can be gauged by measuring its distance from its nearest neighbor. An exact measure for the uniqueness would be the average 'nearest neighbor' volume for the feature vector in a feature space containing all the feature vectors from the dataset. The 'Nearest Neighbor'(N.N) volume is a

region around a feature vector in a feature space in which the given feature vector is the nearest neighbor for any query point present in that volume. Geometrically, the nearest neighbor volume is expected to be a polytope surrounding each feature as shown in Figure 3.3. As it is computationally expensive to extract the corners of the polytope and calculate its volume, the nearest neighbor distance is used as a measure for the uniqueness of each feature. The nearest neighbor distance can be construed as a diameter of a certain 'lower-bound' hypersphere as shown in Figure 3.3. This hypersphere is the lower bound for the nearest neighbor volume for a point with the given nearest neighbor distance. It can also be visualized as the largest hypersphere that can be fit into the nearest neighbor volume while being centered on the given feature vector. Figure 3.4 shows the average nearest neighbor distance for 'Amplitude Bias' features extracted from different scales for the pitch dataset. The scale information in Figure 3.4 is denoted by length of the filter used at each scale. A total of 5 different scales starting from 205 meters (410/2 meters) to 12.8 meters (410/32 meters) were used in all the experiments shown in this paper. It was experimentally verified that scales below 12.8 meters did not have enough information content to merit feature generation. The result can be understood from an intuitive point of view where we would not expect variations over distances smaller than 12.8 meters to contain unique pitch signatures. It can be seen that the feature vectors generated from the 'DerGauss' wavelet are much more unique than those generated from the other wavelets. One can also notice

that the low frequency features are much more unique than their high frequency counterparts. This low-frequency uniqueness is a general characteristic of many datasets as they are likely to have fewer low frequency features simply because the low frequency features encode longer distances which make them more unique.

3.4.3.2 Robustness of the feature vector

The robustness of a feature vector is measured as the covariance of the feature vectors when the original signal is corrupted by a certain amount of noise. This subsection presents the derivation of the covariance for the 'Amplitude Bias' feature vector. It must be noted beforehand that critical approximations and assumptions are made in this derivation in order to make the problem mathematically tractable. The entire derivation is broken down into four steps, each of which corresponds to the steps described in the previous section for constructing the feature vector.

3.4.3.2.1 Wavelet decomposition Let $y(t)$ be a given signal that is corrupted by Gaussian noise $n_{\sigma^2}(t)$ whose variance is $z(t)$. The resultant corrupted signal be denoted by $z(t)$ and is shown below.

$$z(t) = y(t) + n_{\sigma^2}(t) \tag{3.1}$$

Taking the continuous wavelet transform (CWT) on both sides of the above

equation

$$W_z(t,s) = W_y(t,s) + n_{k\sigma^2}(t) \tag{3.2}$$

Where $k = |h_s(t)|^2$,$n_(k\sigma^2)$denotes noise of variance $k\sigma^2$, and $h_s(t)$ is such that

$$W_y(t,s) = y(t) * h_s(t) \tag{3.3}$$

For the CWT, $h_s(t)$ is scaled such that k is a constant across all scales 's'.

3.4.3.2.2 Obtaining Key Points Let t_i and s_i denote locations on $W_y(t,s)$ such that

$$\begin{aligned} |W_y(t_i,s_i)| &> |W_y(t_i,s_i)| \\ |W_y(t_i,s_i)| &> |W_y(t_(i-1),s_i)| \end{aligned} \tag{3.4}$$

The locations (t_i,s_i) denote the extrema (maxima and minima) occurring on $W_y(t,s)$. Given the set of locations (t_i,s_i), a smaller subset of these locations (t_k,s_k) are designated as key points such that

$$\begin{aligned} (t_k,s_k) = \{t_k,s_k \quad & |t_k \in t_i, s_k \in s_i, and \\ & |W_y(t_i,s_i) - W_y(t_(i+1),s_i)| > T*k, \\ & |W_y(t_i,s_i) - W_y(t_(i-1),s_i)| > T*k, \\ & Tisathresholdfactor\} \end{aligned} \tag{3.5}$$

The value of T was chosen to be 0.25 for all the experiments presented in this paper. All the subsequent analysis is made under the assumption that the locations

(t_k, s_k) are such that they satisfy the above conditions not only for $W_y(t, s)$ but also for $W_z(t, s)$. This assumption implies that the below derivation for noise present in the feature vector is only valid in those cases in which noise has left the key point locations intact so that they could be identified in $W_z(t, s)$. The above thresholding scheme for obtaining key points also ensures that the locations of the key points are not easily susceptible to change and this further ensures the validity of this derivation. Also, the effect of the above assumption is expected to be less at smaller noise levels as these noise levels are unlikely to affect the position of the key points. Finally, without the above assumption a theoretical derivation would need to study the appearance and disappearance of extrema in the presence of noise. Modeling this behavior might improve the exactness of the result but it would eliminate the simplicity and the ease of understanding that the current approximate derivation provides. It must be noted that the assumption that the key points do not undergo any time-displacement implies that the resulting covariance understates the effect of the Gaussian noise on the feature vector.

3.4.3.2.3 Computing the point feature vector Given the locations of the key points (t_k, s_k), the 'amplitude bias' point feature vector for the uncorrupted signal $f_y(t_k, s_k)$ is given by

$$\begin{aligned} f_y(t_k, s_k) = \quad & [W_y(t_k, s_k) - W_y(t_{(k-1)}, s_{(k-1)}) \quad t_k - t_{(k-1)} \ldots \\ & W_y(t_{(k+1)}, s_{(k+1)}) - W_y(t_k, s_k) \quad t_k - t_{(k+1)}] \end{aligned} \tag{3.6}$$

Given the above assumption that the locations of the key points are unaltered, the feature vector corresponding to the corrupted signal $f_z(t_k, s_k)$ is given by

$$\begin{aligned} f_z(t_k, s_k) = \quad & [W_z(t_k, s_k) - W_z(t_{(k-1)}, s_{(k-1)}) \quad t_k - t_{(k-1)} \ldots \\ & W_z(t_{(k+1)}, s_{(k+1)}) - W_z(t_k, s_k) \quad t_k - t_{(k+1)}] \end{aligned} \tag{3.7}$$

The dimensions denoting the temporal terms remain the same between the feature vectors for the original signal and the corrupted signal because of the assumption that the locations of the key points are unaltered between both the signals. Given Equation 3.2, we can conclude that $f_z(t_k, s_k)$ has a bivariate normal distribution whose mean is given by $f_y(t_k, s_k)$ and whose covariance matrix is given by $\begin{bmatrix} 2k^2 & -k^2 \\ -k^2 & 2k^2 \end{bmatrix}$.

3.4.3.2.4 Creating the extended feature vector The extended feature vector is constructed by combining three adjacent point feature vectors and removing the redundant dimensions as described in the previous section. Therefore, the extended feature vector is expected to contain four non constant terms and it consequently has a quarto-variate normal distribution. The distribution is denoted by $F_z(t_k, s_k)$ and its properties are given below:

$$Mean(F_z(t_k, s_k)) = [f_y(t_k, s_k) f_y(t_{(k+2)}, s_k)] \tag{3.8}$$

$$Cov(F_z(t_k, s_k)) = \begin{bmatrix} 2k^2 & -k^2 & 0 & 0 \\ -k^2 & 2k^2 & k^2 & 0 \\ 0 & -k^2 & 2k^2 & k^2 \\ 0 & 0 & -k^2 & 2k^2 \end{bmatrix} \tag{3.9}$$

Table 3.1: Value of the 'k' Parameter for Different Wavelets

Wavelets	'k' Value
Der-Gauss	0.1410
Sombrero	0.2116
Morlet	0.1410

The covariance matrix given above only pertains to the amplitude terms in $F_z(t_k, s_k)$, the expanded covariance matrix that includes the values for the temporal terms can be obtained by inserting rows and columns of zeros at the appropriate locations as the 'temporal' terms do not change between the corrupted and the uncorrupted signal. This section concludes by noting that the above derivation has been able to identify the distribution for the extended feature vectors under the assumption that the key point locations remain intact.

As the key parameter in the covariance matrix is the term k one can calculate the value of this constant for different wavelets and this result is shown in Table. 3.1. One can see from Table. 3.1 that the DerGauss wavelet and the Morlet wavelet have the lowest noise variance and are therefore good candidates for generating extrema. As the DerGauss wavelet has the highest uniqueness as shown in Figure 3.4, one could infer that the DerGauss wavelet is likely to give the best performance in the

pattern matching task.

3.4.3.3 Comparison of the theoretical and simulation approaches.

The purpose of this section is to compare the results obtained from using the theoretical model to those obtained from simulation.

3.4.3.3.1 Simulation Approach The results corresponding to the simulation approach are shown in both Figure 3.6 and Figure 3.7 and were obtained by repeatedly (30 times) corrupting the query signal with Gaussian noise (std dev = 0.5 deg) and running the algorithm in order to calculate the probability of obtaining a correct match for a feature from each scale. The results from the simulation approach constitute the true matching accuracy for a feature vector as no approximations have been made and hence provide a complete evaluation of the techniques.

3.4.3.3.2 Theoretical Approach The theoretical approach combines the uniqueness and the robustness calculations that were presented before to compute the accuracy for matching a feature vector in the presence of noise. In this analytical evaluation, the quarto-variate distribution of each extended feature vector, corresponding to when the signal is corrupted by gaussian noise (std dev = 0.5 deg), is integrated over the approximate nearest neighbor volume in order to calculate the probability for that feature vector to be correctly matched. As it is computationally expensive to extract the corners of the nearest neighbor polytope and as

the integration of the quarto-variate distribution over this polytope would be hard to perform, the nearest neighbor volume is approximated with other geometric elements, as shown in Figure 3.5, to obtain different estimates of the probability of obtaining a correct match. The probability estimates for all the features from a particular wavelet decomposition scale are used to obtain the accuracy of obtaining a correct match for a feature vector from that scale. The accuracy estimates at each wavelet decomposition scale were obtained for different wavelets, using different nearest neighbor approximation volumes are shown in Figure 3.6 and Figure 3.7.

3.4.3.3.2.1 Hyperplane Approximation In order to obtain a probability estimate that is greater than the true probability, the quarto- variate distribution is integrated over all the space on one side of a particular hyperplane. This hyperplane is the perpendicular bisector of a line drawn between the given feature vector and its nearest neighbor. The hyperplane is shown in Figure 3.5 for an example feature vector. The underlying assumption behind using the hyperplane is that the nearest neighbor is the only feature that is present within the region where the probability from the quarto-variate distribution is significant. In fact, the above defined approximate nearest neighbor volume is the true nearest neighbor volume for the given point under certain conditions (ex: if the current point and its nearest neighbor were the only two points present in the entire space). The resulting probabilities for each wavelet scale are shown in Figure 3.6. It can

be seen that the probabilities obtained from this evaluation are much higher than those obtained from simulation because of two assumptions that went into the theoretical approach. The assumptions are:

1. The locations of the key points are unchanged due to the noise.

2. All the other feature vectors, other than the nearest neighbor, are substantially far away from the given feature vector.

Both these assumptions will lead to an over estimation of the probability value. Thus, the theoretical probability values obtained from the above calculation can be construed as being an upper bound for the actual probability value as the calculations have been made under assumptions which will lead to an over estimation of the probability values.

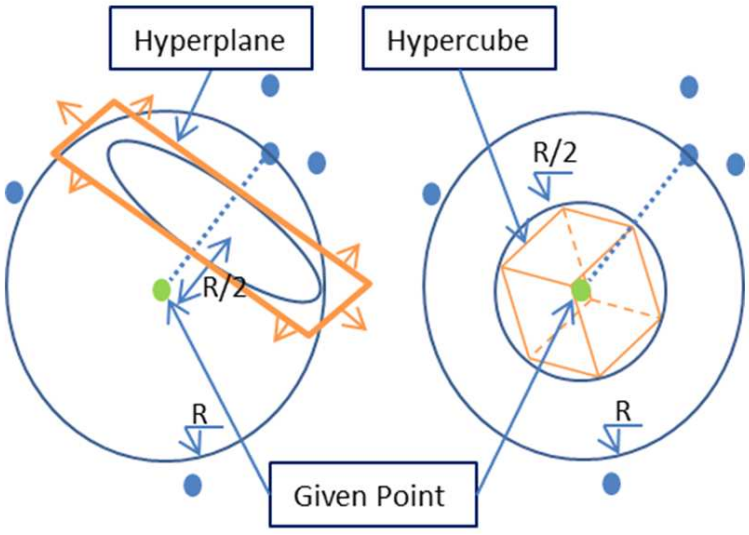

Figure 3.5: An illustration of the geometric elements (Hyperplane and Hypercube) that are used to approximate the nearest neighbor polytope.

3.4.3.3.2.2 Hypercube Approximation The hyperplane approximation considered previously was the largest nearest neighbor volume that could exist for a feature vector in a database given its nearest neighbor. The smallest nearest neighbor volume that can exist is a hyper sphere whose radius is half the nearest neighbor distance. In the presence of a large number of feature vectors at a nearest neighbor distance R, the resulting polytope for the feature vector converges to a hyper sphere. The probability density function is integrated over the largest hypercube that resides in this hyper sphere, as shown in Figure 3.5, for the sake of computational ease. The resulting probability estimate is very conservative as the nearest neighbor space is always larger than the hypercube. The probability estimates for feature vectors from each wavelet scale are shown in Figure 3.7, and they are lower than the actual simulation results in most of the cases because of the underestimation of the nearest neighbor space. It can be seen that at lower frequencies the theoretical probability estimate is higher than the simulation and this is because the conservativeness of the hypercube assumption is not enough to offset the assumption that none of the peaks are being dislocated due to noise. From the comparison of the theoretical and simulation approaches, one can see that the results from both the approaches are consistent. Both methods indicate that the 'DerGauss' wavelet would give the best pattern matching performance among the given wavelets for the pitch dataset. The above analysis also reveals the complementary nature of the simulation and theoretical approaches. The theoretical

approach is able to provide insight into the uniqueness and the robustness of the feature vector, while the simulation approach is able to give an estimate about the performance of the feature vector by accounting for all aspects of the system. From the graphs in Figure 3.6 and Figure 3.7, it is clear that the low frequency features are more accurate than high frequency features, therefore one should preferentially use low frequency scales when choosing between different wavelet decomposition scales for this particular dataset. This pattern is seen in a number of different datasets where the noise is mostly present in the higher frequencies. The theoretical and the simulation approaches show that the 'DerGauss' wavelet gives the best feature matching performance for Pitch Data within the given wavelets.

3.5 Experimental Results For Pitch-Based Global Localization

An experiment was performed on actual highways to evaluate the feasibility of this method for global localization. For the experiment, "map" data was collected once over 6000 km of roadway, and then "test" data was collected on a small portion of the same road way, across just 6 km. The full roadway, shown in Figure 3.8, was used in the map building process while the second run was used in the testing procedure. The 'DerGauss' wavelet and the five wavelet scales shown in Fig 6-Fig 10 are used to generate the features for this experiment. The 'Amplitude

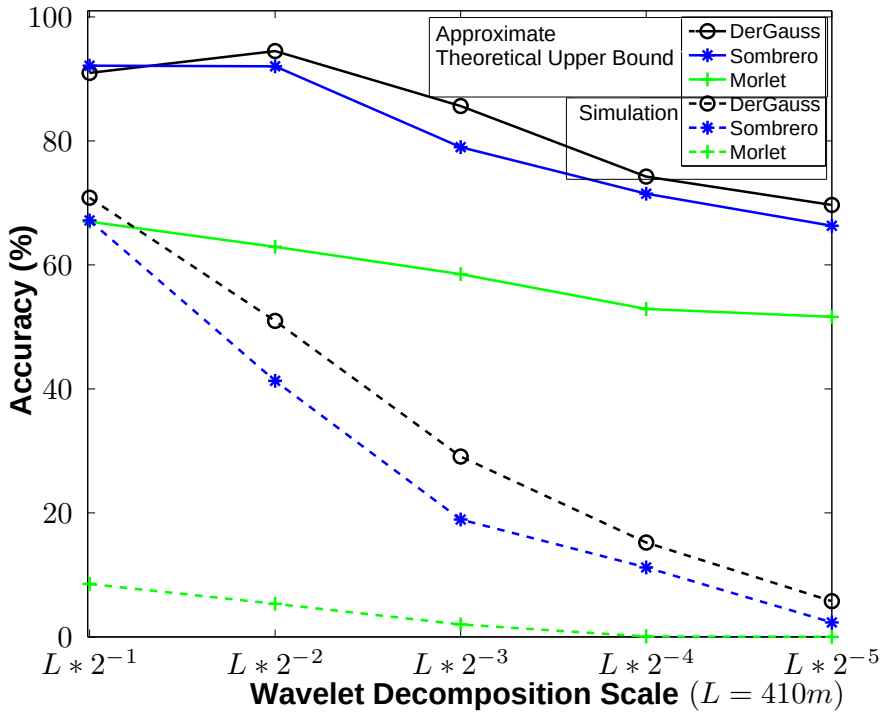

Figure 3.6: A comparison of the theoretical accuracy obtained from the hyperplane NN approximation with the simulation results.

Bias' feature vector from Table 2.1 is used in this analysis as no significant scale distortions were observed in the pitch data. Three point features are combined and an extended feature vector is created by removing the redundant dimensions.

Thirty different query signals were extracted from the "test" data. To check the accuracy of localization, the ground truth for the 6 km stretch in the datasets was obtained from a DGPS system with a positional accuracy of 0.1 meters. It must be noted that while the mapping phase used the GPS information for representing the pitch information as a function of distance, the testing phase used the pitch

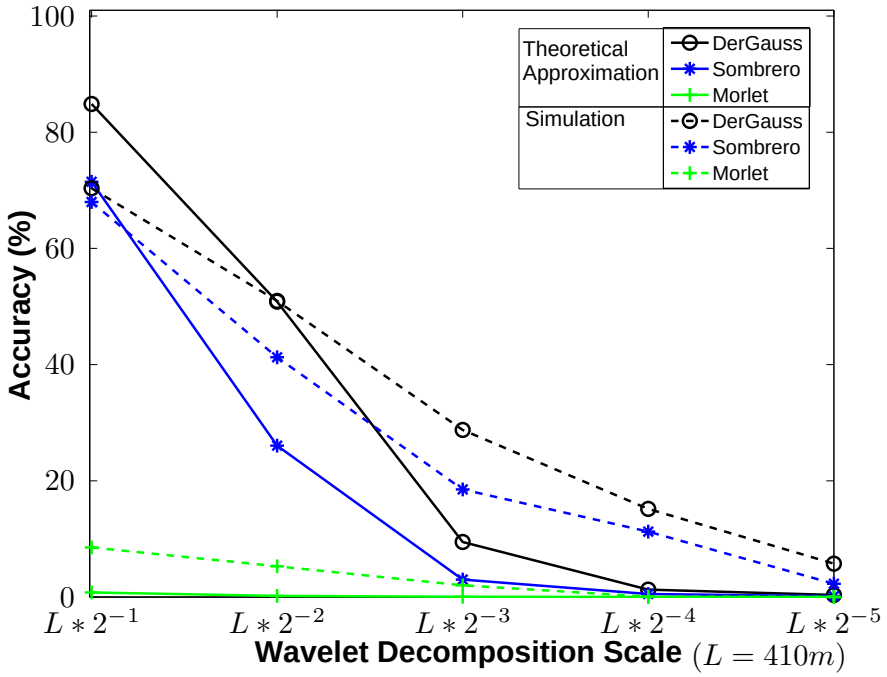

Figure 3.7: A comparison of the theoretical accuracy obtained from the hypercube NN approximation with the simulation results.

information and the wheel encoder data to do the same. The localization estimate that is obtained from this representation of the pitch data is verified by using the GPS data that was collected as a part of the testing phase. A threshold of ten meters was used to determine if a certain match was accurate or not. Figure 3.9 and Figure 3.10 shows the experimental setup and Figure 3.11 shows the accuracy curves that were obtained for the feature tree based method. The accuracy curves illustrate the localization accuracy that was obtained as a function of query signal length. It must be noted that for a query length of 200 meters, the correct position estimate was amongst the first five position estimates that were obtained from the histogram of the position estimates for ninety percent of the cases. The mean and

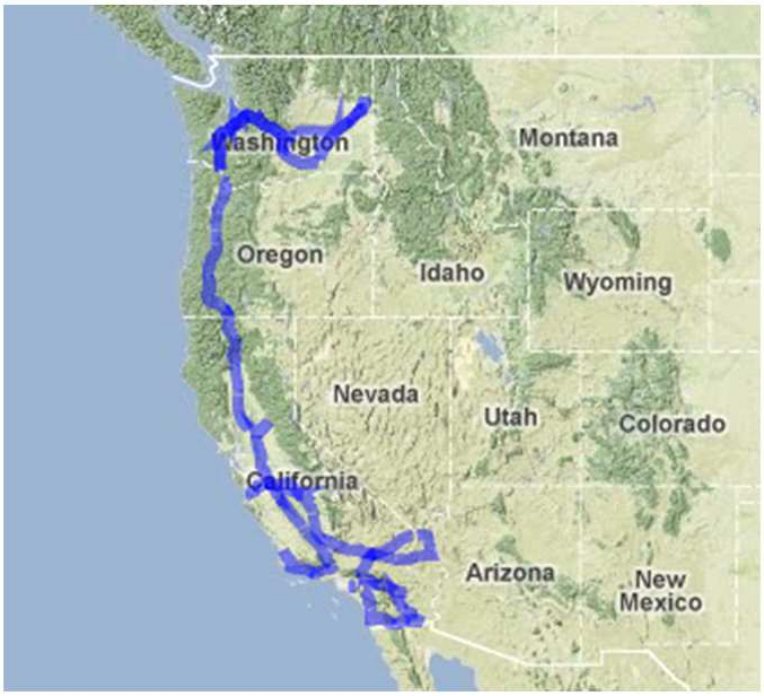

Figure 3.8: The roadway network that was used as a part of the experiments.

the standard deviation of the error in the location estimate, for a 400 m query signal, were 2.49 meters and 2.20 meters respectively. It can be clearly seen that a threshold value of 10m is not very critical and slight changes to the threshold will not affect the accuracy curves in a significant manner. When tested on the 6000km database it was found that a single feature vector match took about 0.01 seconds. The entire matching process for a 400 meter signal took about 1.16 seconds on a 3.16GHz dual core computer and implemented by using a non-optimized MATLAB code.

Figure 3.9: The vehicle setup that was used as a part of the data collection effort.

3.6 Simulation Results For Pitch-Based Global Localization

The objective of the simulation process is to test the ability of the designed feature vector to withstand various types of sensor noise typical of vehicle sensors and road measurement. Each type of sensor noise is represented by a corresponding parameter in the sensor model. In this paper, the pitch and the encoder sensors are modeled with the sensor models taken from [65] and [52], which are given by Eqns 3.10 and 3.11 whose terms are defined in Table 3.2. It must be noted that the bias term for the pitch sensor error in [65] is modeled as a slow varying bias and is approximated as a constant for the purposes of this simulation. The 'Amplitude

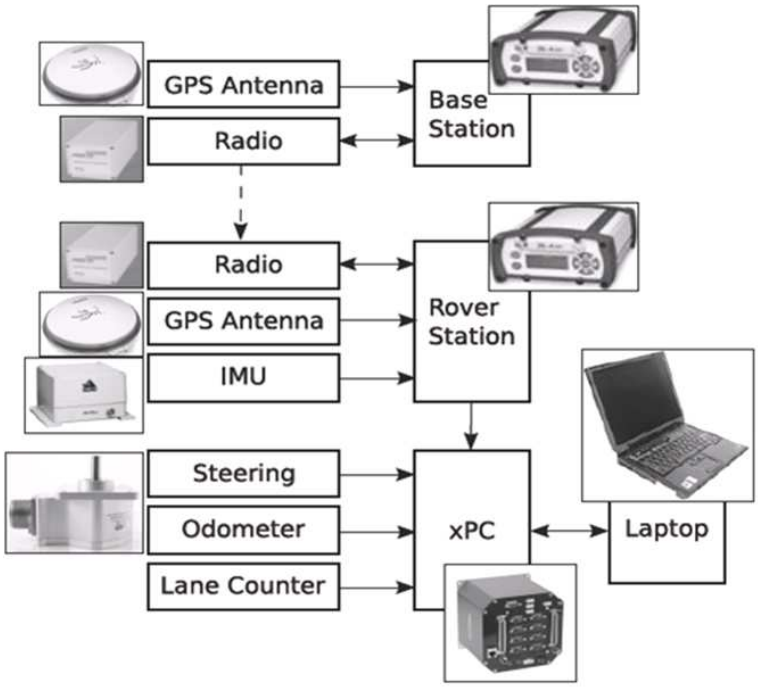

Figure 3.10: The overview of the data acquisition setup that was used as a part of the experiments.

Bias' point feature vector is used in this analysis for all the distortion cases except the scale factor distortion. In case of the scale factor error the 'Amplitude Bias and Amplitude Scale' point feature vector from Table 2.1 is utilized as it provides robustness against scale distortion.

Table 3.2: Nomenclature for Equations 3.10 and 3.11

Symbol	Quantity
$Pitch_t$	True Pitch
$Pitch_m$	Measured Pitch
B	Constant Bias Error
S_f	Constant Scale Factor Error
ν_{w1}	Zero mean band limited pitch noise
ν_{w2}	Zero mean band limited encoder noise
$Encoder_m$	True encoder value
$Encoder_t$	Measured encoder value

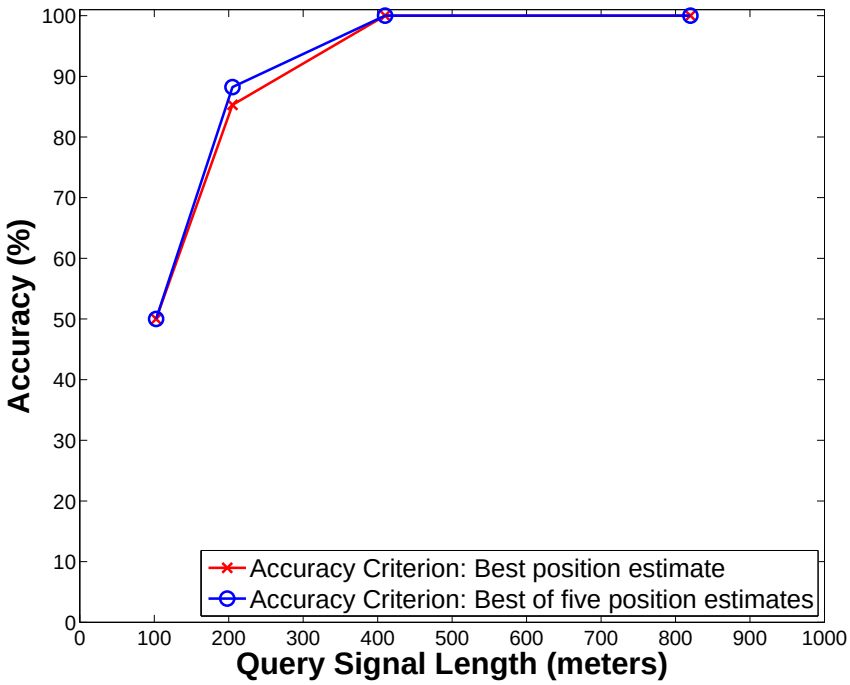

Figure 3.11: Accuracy curves for localization in a roadway network of 6000 Km.

$$Pitch_m = (1 + S_f)Pitch_t + B + \nu_{w1} \tag{3.10}$$

$$Encoder_m = Encoder_t + \nu_{w2} \tag{3.11}$$

These sensor models contain a total of four different error parameters $(B, S_f, \nu_{w1}, \nu_{w2})$, each of which represents particular types of noise. The error sources that are modeled by these parameters can come from both the sensor and the data collection process. For example, B includes the bias error in the pitch sensor and any inclination angle error in mounting the sensor to the vehicle. Similarly, $\nu_w 1$ and $\nu_w 2$ represents the zero-mean band limited white noise from the pitch and encoder

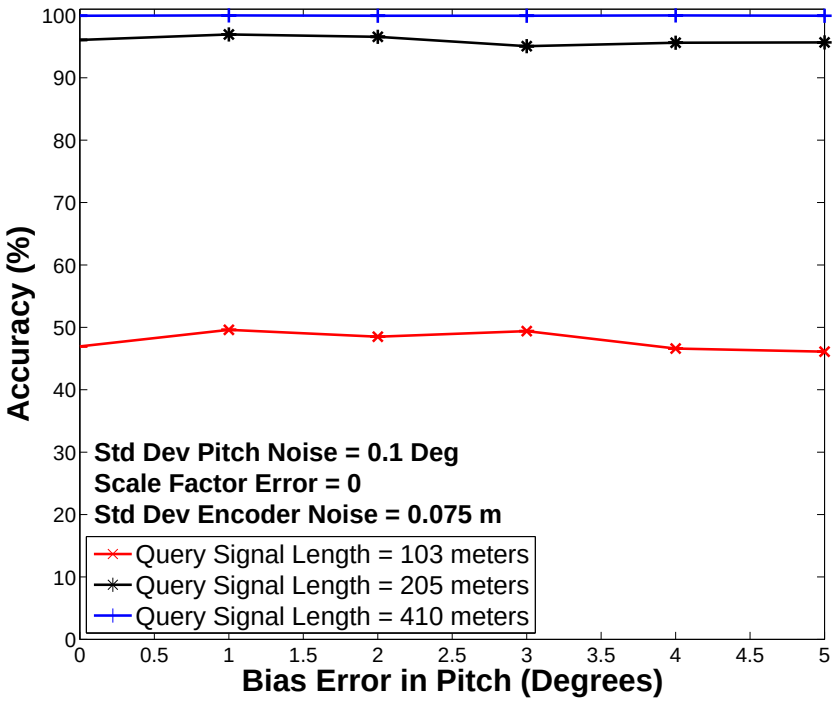

Figure 3.12: The localization accuracy of the 'amplitude bias' feature vector is immune to the bias noise present in the sensor.

sensors and also the noise from the vehicle chassis vibration and measurement errors due to slight differences in the lateral lane position of a vehicle during data collection. The bias and scale factor errors are expected to be mainly from the pitch sensor and the contributions from the data collection process are expected to be small if the sensors are properly mounted and calibrated. The standard deviation of the band limited noise for pitch data is again expected to be mainly from the sensor and the contribution from the data collection process was estimated as 0.057 degrees (std dev) for the pitch data from experiments.

For this simulation, pitch data obtained from an integrated GPS-IMU system

was used to generate feature vectors that were stored in a feature tree. Figure 3.8 shows the 6000 km of roadway that was used in creation of the feature tree. Small portions of the original signal were taken and corrupted with each of the different noise types $(B, S_f, \nu_{w1}, \nu_{w2})$ up to varying degrees to create a "query signal". The feature vectors obtained from this query signal were matched with the original database to estimate the position in the database from which this query signal was extracted. The correctness of this position estimate was decided on the basis of a threshold distance (10 m) from the true point of extraction of the signal, e.g. any final estimate within this threshold is considered correct (local tracking algorithms can "lock" easily within this range). For each of the four parameters, the simulation was performed by varying the parameter of interest while keeping others constant at their expected value for low cost sensors. The query signal was extracted from sixty different points of the original signal. Each query signal was corrupted and tested thirty times in order to obtain a statistical estimate of algorithm performance that accounts for the random nature of the errors introduced. The tests were performed for query signals of lengths 103, 205 and 410 meters.

Figure 3.12 shows that the estimation process is unaffected by bias (B), a result that was expected as the feature vector was designed for bias invariance.Figure 3.13 (top) shows that the scale factor error (S_f) severely affects the performance of the 'amplitude bias' feature vector. The 'Amplitude Bias and Amplitude Scale' feature vector which has been designed to be scale invariant is completely immune to

scale factor error (S_f) as shown in Figure 3.13 (bottom). Figure 3.14 (top) shows that the estimation procedure was largely invariant to distance measurement (encoder) noise (v_{w1}) that one would encounter at highways speeds (60mph) which was estimated to be 0.076m (std dev) for each encoder tick at 100 Hz [52]. The addition of band-limited random noise in the pitch sensor (v_{w2}) was also investigated (Figure 3.14, bottom). In [65], it was found that low-cost sensors used for pitch measurement had a standard deviation of v_{w2} of 0.1 degrees, so variations in pitch noise around this deviation were considered. This noise type appears to have a significant effect on the accuracy of the result. Both the plots in Figure 3.14 consider localization performance over several different query lengths. The results show that the performance of a more accurate pitch sensor can be achieved by a low-cost pitch sensor if one simply collects data over a longer period of time to obtain a longer query signal.

3.7 Conclusions and Future Work

Firstly, the paper demonstrates the possibility of using pitch data for global localization in large roadway networks. By generating feature vectors for one dimensional pitch data, localization has been effectively performed for a road way network that is an order of magnitude larger than what has been previously demonstrated [47, 40]. Table 3.3 provides a comparison of the proposed method with localization methods using other sensors. While pitch information has low data density and

hence can be utilized for localization over a large roadway network, the tradeoff is that a vehicle needs to travel a longer distance before localization is achieved.

Table 3.3: Comparision Of Different Sensor Modalities For Localization

Sensor Citation	Map Size(Km)	Numbers of Features/Km	Travel distance for query signal (m)
Vision [47]	20	5 x 10^6	< 1
LIDAR [40]	165	9 x 10^3	$< 20 - 60$
Pitch	6000	3 x 10^1	$<$ 400

This work also enables the use of other inertial measurements from vehicles, such as roll and yaw data, for localization. Future work could be directed towards implementing a feature vector that combines multiple sources of inertial data, thus reducing the query length because of the higher data density. The paper also introduces 'Multi Scale Extrema Features' which are designed to overcome the expected drawbacks of using current time series subsequence matching techniques for inertial data. These features are robust to sensor noise and future work could involve demonstrating their capabilities by performing localization with low-cost inertial sensors. Overall, this paper presents a promising new technique to perform global localization in order to compensate and/or replace GPS position estimates on roadway networks.

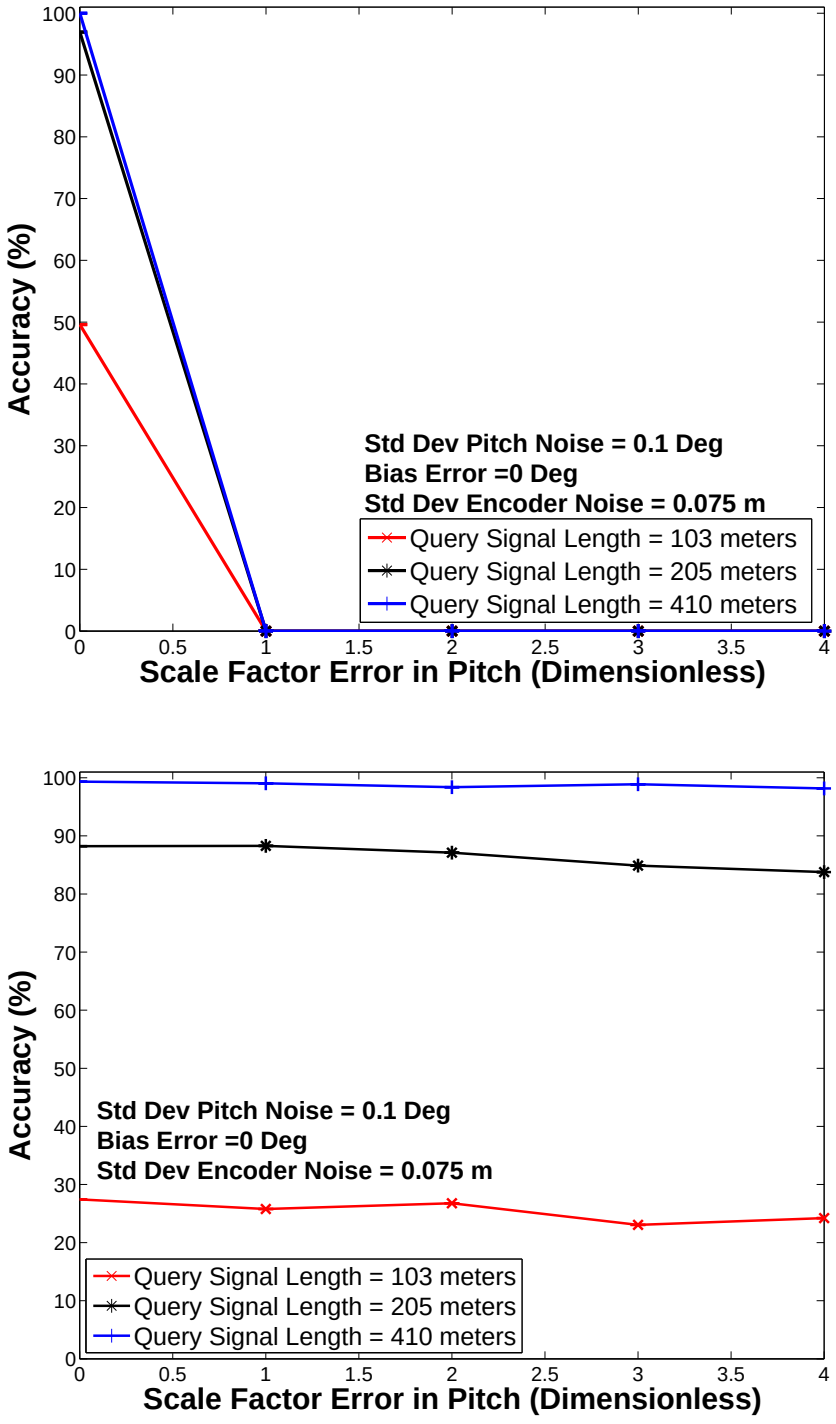

Figure 3.13: The plots show that the localization accuracy for the 'amplitude bias' feature vector (top) is severely affected by scale factor noise while the 'amplitude bias, amplitude scale' feature vector (bottom) is completely immune to it.

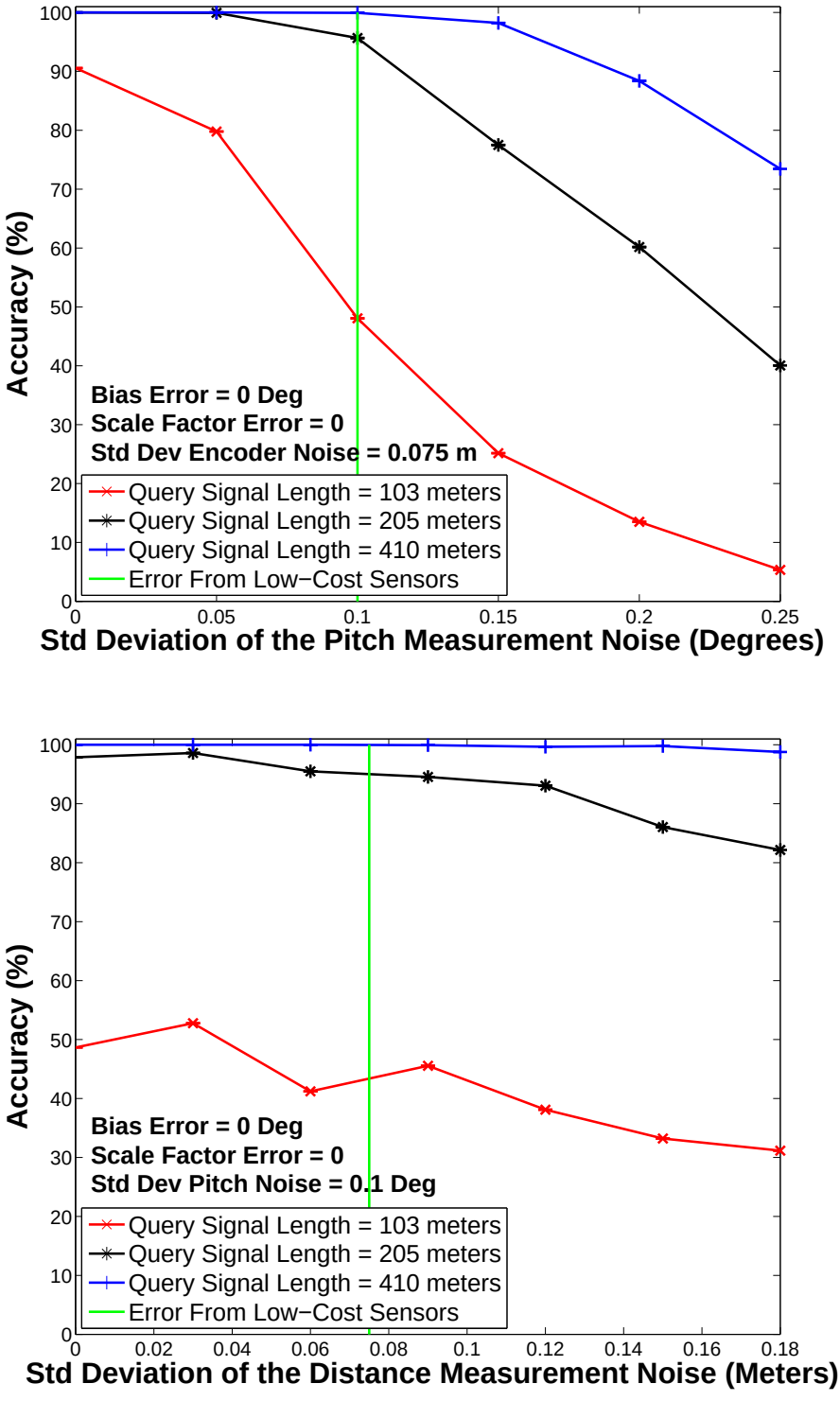

Figure 3.14: The plots examine the effects of band-limited white noise in the encoder and pitch measurements on localization accuracy.

Chapter 4

Pattern Matching of In-Vehicle Acceleration Time Series Data

4.1 Introduction

Previous work that identified patterns in acceleration data was aimed at identifying the nature of terrain that the vehicle was traveling on [66] or identifying the driver behavior [67]. Recently, acceleration data is also being used to identify the activity of a person [68] as many smartphones come with accelerometers built into them. In this work we aim to extract location information by correlating patterns from acceleration data across multiple vehicles. This problem was encountered as a part of a broader research project where a large set of vehicle data (service records, acceleration data, etc) were collected over many years and on a number of different vehicles. While the acceleration data from these vehicles is known, the GPS data

is unavailable due to privacy reasons. For purposes of prognostics and diagnostics, one of the tasks of the original project was to identify patterns in the data collected from fleet vehicles that were operated as a part of convoys or which were operated in the same routes. As the GPS information was unavailable, it was deemed necessary to explore the possibility of being able to cluster vehicles into common locations on the basis of measured acceleration data. If this grouping can be performed, then a possible application could be to pair this new information with the service records and other information from the vehicles to identify relationships between driving behavior patterns, positions, and repair histories. Before solving the problem of clustering the vehicles, one must check feasibility of a signal-feature solution approach through a preliminary test where true positions are measured during the test. A preliminary test in this case would be to check whether the acceleration signatures of two vehicles that have travelled on the same road can be matched in the presence of vehicles. Also, the preliminary test data can be used to determine the most effective variant of a feature vector that can be used to solve the clustering problem. The main objective of this chapter is to perform this preliminary test and identify feature vector formulations that will be effective in matching acceleration data collected from different vehicles. The general overview of the preliminary test is as follows. Two sets of vehicle data including both acceleration and GPS (for ground truth) are collected on the same set of roads with two different vehicles. One set of acceleration data is used to create a database while

portions of different length are extracted from the second dataset and are used to obtain a match to the first acceleration profile. The GPS locations from both the datasets are used to measure the accuracy of the matching process. In its essence, the preliminary test consists of matching a signal with a large database of signals to find the most similar matches. This problem is often referred to as the subsequence matching problem [4]. The preliminary test is also similar to the map-based Global Localization problem in robotics [46, 47, 51]. In the Global Localization problem, a map (or database) is given to a robot, and robot must establish its location within the map by collecting sensor data and matching it with the map. Thus, this technique can be seen not only as a method to perform the feasibility test for grouping vehicles based on acceleration data, but also as a feasibility test for acceleration-based localization. A variety of sensors have been used for Global Localization over the years. LIDAR [46] and Vision sensors [47] are the most commonly used in robotics to perform Global Localization. Both LIDAR and Vision systems provide high dimensional data and the nature of these sensors is substantially different from inertial sensors such as an accelerometer and so the methods applicable cannot be directly utilized for pattern matching with acceleration data. In recent research, Vemulapalli et al [62] have reported that global localization can be performed using pitch data. Global Localization with pitch data [62, 63] has many similarities with respect to the 'preliminary test' problem using accelerations, but there are a number of key challenges that are specific to acceleration data.

Pitch is generally easier to use because the pitch plotted against odometry does not change significantly with speed. The acceleration data, however, can undergo substantial distortion based on external conditions such as traffic. Moreover, the bias and scale factor variations in the pitch data are generally smaller than that for acceleration data. While this technique utilizes the 'Multi-Scale Extrema Feature' vector framework developed for the pitch-based localization method, it evaluates different variants of the above feature vector and provides insights into the specific requirements and possibilities for acceleration data. This chapter also proposes a novel 'Multi-Scale Encoding' method that enhances the performance of the feature matching algorithm. While the current work presents the results in the context of in-vehicle acceleration data matching, this feature vector could potentially be used for other acceleration matching applications. Section 4.2 explains the challenges in matching acceleration data from two different vehicles and presents a literature survey of the current subsequence matching techniques and their abilities to handle the above challenges. Section 4.3 presents the novel encoding method which is a generalization of the encoding technique previously described for the case of 'Multi-Scale Extrema Features'. Section 4.4 describes the experimental setup used to collect the data required to test the algorithm. Section 4.5 presents the results obtained from applying different variants of the MSE features to acceleration data. Conclusions then summarize the main results of this work.

4.2 Background and Literature Survey

Before setting out to perform the 'preliminary test', one can visually verify whether acceleration data collected from two different vehicles on the same route have similar characteristics. Figure 4.1 shows the acceleration data collected on a vehicle that has travelled on a certain public route and within normal traffic patterns. One can clearly see the effect of the road layout on the acceleration data, wherein the turns of the route correlate to specific acceleration features. This implies that one can predict the acceleration of a vehicle, to a certain extent, based on the route that the vehicle is traveling on. Or conversely, one can use acceleration features to discern route location. The driver behavior, such as the speed at which one is traveling, external conditions, such as the traffic on the road, and the vehicle dynamics will also affect the acceleration data that is collected on a vehicle. This is in contrast to pitch based localization which is largely immune to these variations. The key to acceleration-based pattern matching is the ability to extract the

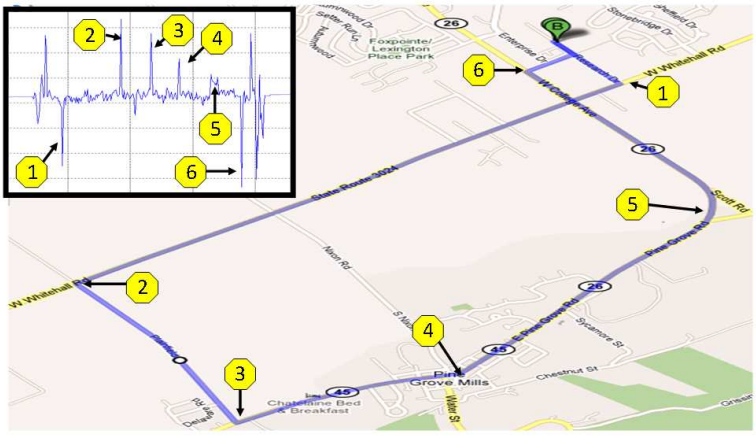

Figure 4.1: The effect of the route on in-vehicle acceleration data

common features that can be matched irrespective of the nature of distortions that will be experienced due to human behavior and external conditions. Figure 4.2 shows the acceleration data collected on the same route as shown in Figure 4.1 on two different runs by different drivers. While one can definitely notice similarities between the data, the nature of distortions that one can observe is substantial: the five distortions that one can easily notice are shown in Figure 4.2 include temporal distortion, outliers, bias distortion, scaling distortion, and random noise. Given the nature of distortions in this case, a robust subsequence matching technique must be deployed. A number of different subsequence matching techniques have been proposed in the literature. Unfortunately, the similarity distance metrics that are used by these techniques have certain drawbacks that preclude them from being effective for acceleration data in particular. For example, the Euclidean distance has been reported as being brittle [19] to temporal distortions. Dynamic Time Warping (DTW) [19, 69] has been introduced as a generalized form of Euclidean distance as it is robust to temporal distortion but it fails in the presence of outliers. This has led to 'Edit Distance' methods such as Edit Distance on Real Sequence (EDR) [69] and Longest Common Subsequence (LCSS) [9], which have drawn inspiration from methods used for matching strings in which dissimilar portions between the two strings are ignored. It has been reported in the literature that the edit distance methods are themselves sensitive to amplitude shifting and scaling [13].The above methods [69, 9, 19] are also computationally burdensome

because of the long length of the query signal in this application. Researchers have recently used local pattern based techniques [64, 39], but most of these methods have relied on a sliding window approach and perform an exhaustive search across all window sizes and are thus computationally expensive. The'Multi-Scale

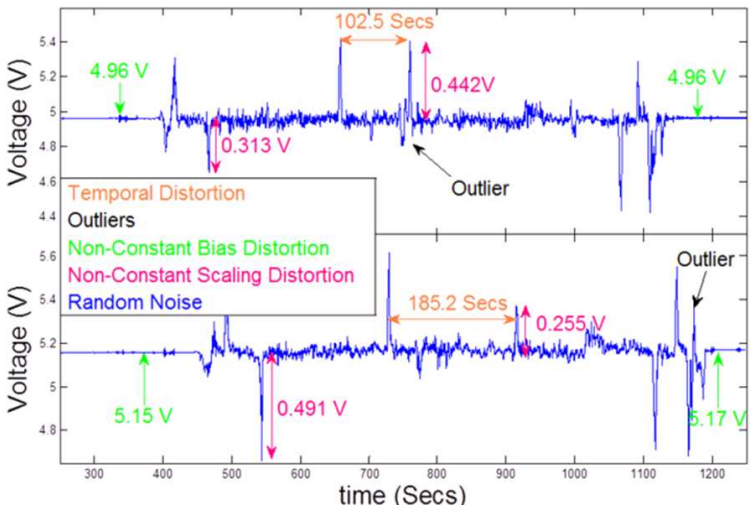

Figure 4.2: The distortions that are exhibited by acceleration data collected on different runs.

Extrema Features' , which have been presented in Chapter 2, have been designed to perform under a range of distortions without using a sliding window approach. The success of these features in the context of pitch based localization makes them promising candidates for the acceleration case. The following section presents a novel 'Multi-Scale Encoding' method which is a generalization of the sequential encoding method used in Chapter 2 and leads to better performance results.

4.3 Multi-scale Encoding

In the subsequent sections, the 'Multi-Scale Extrema Features' presented in Chapter 2 are referred to as 'Sequentially Encoded Multi-Scale Extrema Features' or 'SEMSE' features in order to distinguish them from the 'Multi-Scale Encoded Multi-Scale Extrema Features' or 'MEMSE' features that are described in this section. The term 'Multi-Scale Extrema (MSE) Features' is used to refer to 'SEMSE' and 'MEMSE' features simultaneously. Multi-Scale Encoding is a technique to improve the matching accuracy by encoding more feature vectors for a given signal that in turn captures more information about the signal. In the Multi-Scale Encoding method, point feature vectors from different scales are combined together to form extended feature vectors. Figure 4.3 illustrates the encoding mechanism for SEMSE and MEMSE feature vectors. The figure shows the features vectors that are formed with a point feature vector (P.F.V)(shown in a red glow) in conjunction with other P.F.Vs (shown in a yellow glow) for both the sequential encoding and Multi-Scale Encoding methods. Multi-Scale Encoding allows encoding of feature vectors from even those wavelet scales where there may be insufficient number of extrema in a particular scale to form a sequentially encoded feature vector. This leads to improved performance for shorter query signals. In this particular implementation, two point feature vectors (P.F.Vs) are combined to form a MEMSE feature and the amplitude bias invariant feature vector encoding from Table 2.1 is used to generate the P.F.V. As a large number of combinations of point fea-

ture vectors across multiple scales are possible, it becomes necessary to limit the number of combinations by setting time and scale windows in which suitable combinations can be found. Choosing a larger window size will lead to the creation of a larger number of a features but this would also increase the computational effort required for the pattern matching task. In this particular implementation, each point feature vector (P.F.V) from a given scale was combined with point feature vectors from two subsequent scales. Within these scales, the original P.F.V was combined with other P.F.Vs which were within a certain time threshold interval from the original P.F.V. This time threshold has to be adaptive, as each wavelet scale represents the signal over different time lengths. The threshold for each scale was chosen to be twice the compact support of the wavelet filter at that particular scale. This allows for an adaptive threshold that adjusts itself to an appropriate extent corresponding to the filter. It can be seen that the Multi-Scale Encoding method is a generalization of the sequential encoding method, where the P.F.V's from different wavelet scales and beyond adjacent neighbors are combined. It is also important to note that the extended feature vector will contain two additional dimensions which store information about the difference in the scales and the temporal distance between the two combined P.F.Vs in the case of Multi-Scale Encoding.

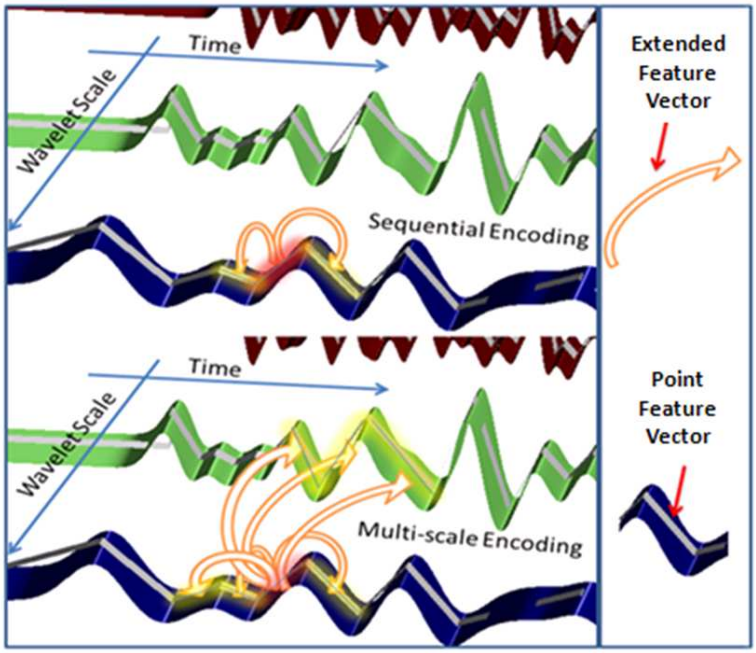

Figure 4.3: An illustration of the sequential encoding method and the multi-scale encoding methods.

4.4 Experimental Setup

Acceleration and GPS data were collected along six predetermined routes. These routes ranged from 15-45 minutes in duration and included diverse driving conditions such as winding roads, mountainous roads, highways, downtown driving, etc. The total distance for all the six routes combined was 135 kms and the routes are shown in Figure 4.4. Each of the six routes was driven in two different manners to test two particular scenarios.

1. Convoy Scenario: In the convoy situations, three cars drove the route simultaneously, with the cars safely following directly behind each other. The data collected from different vehicles in the convoy scenario is expected to have

similar characteristics as all the vehicles were traveling at similar speeds, in similar traffic conditions. However, there will be some variation due to the different drivers involved.

2. Single Vehicle Scenario: In the single car situations, one car drove the route independently with no driving restrictions other than local traffic laws. The route was repeatedly driven under different times of the day (different traffic conditions). This provided a less controlled test where data collected was unique to driving style and traffic patterns.

The equipment used to collect data included a GlobalSat BU-353 GPS antenna sampling at 10 Hz, a SparkFun, three-axis, ADXL335 accelerometer sampling at 9600 Hz, a battery pack, and a data-logging box. The data-logging box housed the accelerometer and stored GPS and acceleration readings. The data-logging box was positioned behind the passenger seat and was firmly fixed to floor of the vehicle. The magnetized GPS antenna was mounted in the rear-window area of the car to provide higher satellite visibility. Throughout the tests, the equipment was positioned in the same orientation for data consistency. Refer to Figure 4.5 for images of the equipment setup. The GPS and accelerometer data were collected separately in the data-logging unit. Post-processing was used to convert the data into MATLAB data files. To compensate for the different sampling rates, the GPS and acceleration data were resampled to 10 Hz for further processing.

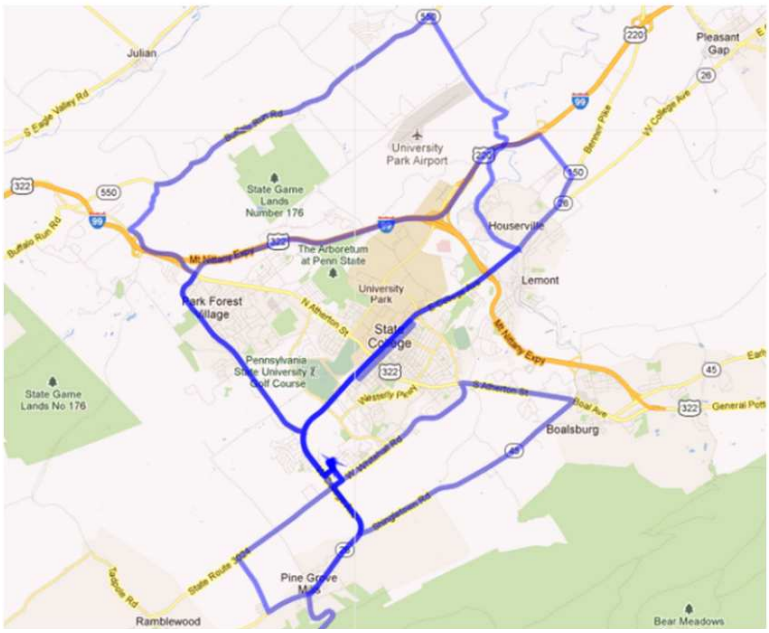

Figure 4.4: The routes covered as a part of the data collection effort

Figure 4.5: The sensors and data acquisition systems used in the experiments.

4.5 Experimental Results

This section presents the experimental results obtained by using Multi-Scale Extrema (MSE) features on the data from the two scenarios mentioned in the previous section. The tests are conducted using the subsequence matching procedure described in Chapter 2. The testing procedure consists of a preprocessing phase in

which the data collected on one of the vehicles is used to build the KD-tree data structure. In the testing phase, acceleration data obtained from another vehicle is used to create feature vectors and these features are matched with the KD-tree data structure. The position estimates from the matches are compiled into a histogram and the match with the highest number of votes provides a position estimate for the current vehicle. The accuracy of this estimate is compared to the separately measured GPS information and in this implementation, a distance threshold of 300 meters, in the database containing over 135000 meters of data, is used to verify if a resulting location estimate is accurate. A relatively lax threshold distance was utilized as this would be sufficient for the prognostics and diagnostics application which is the eventual target for the preliminary acceleration pattern matching problem. As described earlier, the acceleration data collected on a vehicle depends on the route, driver behavior and external traffic conditions. Given these variations, the convoy acceleration data matching problem is easier because all these variations are expected to be similar as the vehicles are traveling in a convoy formation. On the other hand, in the non-convoy acceleration data, only the variations due to the roadways are expected to match while the variations due to driver actions and external conditions are expected to be different and therefore inhibit the matching process. Due to these differences in the data types, one can notice that in all the subsequent tests, the accuracy result for the convoy dataset is higher than the accuracy for the non-convoy dataset. Therefore, the two datasets

are useful to understand the behavior of the algorithms under different levels of noise. Given the two datasets, the next subsection presents evidence to support the parameter choices that have been made in constructing the feature vector. The subsequent subsection delves into the experimental results of the acceleration matching problem using different variants of Multi-Scale Extrema Features.

4.5.1 Parameter Tuning

4.5.1.1 Selecting the Point Feature Vector

The first design choice in constructing the feature vectors is to choose a point feature vector from among the different options presented in Table 2.1. An experimental test was performed using the SEMSE features to decide the appropriate point feature vector from among those listed in Table 2.1 and the results are presented in Figure 4.6 and Figure 4.7. The test followed the methodology described in Chapter 2 and the two datasets described in the previous section were utilized. The results for the convoy dataset are shown in Figure 4.6 and the Amplitude Bias encoding from Table 2.1 results in the best performance. These results can be intuitively explained as the convoy data is expected to matchup very well as all the effects such as route layout, the driver behavior and external conditions are expected to be similar for all the vehicles. This implies that because of the low noise situation, one would not require a high degree of robustness from the feature vector. Therefore, the Amplitude Bias point feature vector which provides a

unique but not very robust feature vector would be very suitable for this situation.

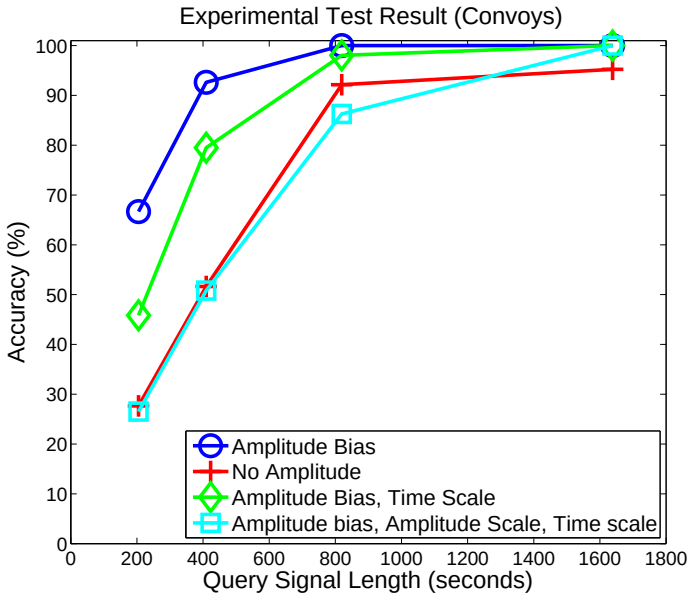

Figure 4.6: Accuracy curves for localization in the convoy dataset using different types of feature vectors.

Figure 4.7 shows that the "Amp-Bias" point feature vector performs well even in the non-convoy situation, but it must be noted that the "Amp bias, time scale" feature vector also performs well. It is quite likely that in the case when non-convoy data is collected with large variations in speed, then "Amp bias, time scale" feature vector can outperform the "Amp bias" feature vector in the non-convoy scenario. The results in Figure 4.6 and Figure 4.7 show that scale information in the feature vector makes an overall positive contribution to matching process because of the uniqueness it imparts in spite of the reduced robustness that might occur because of any scale factor variations between the matched signals.

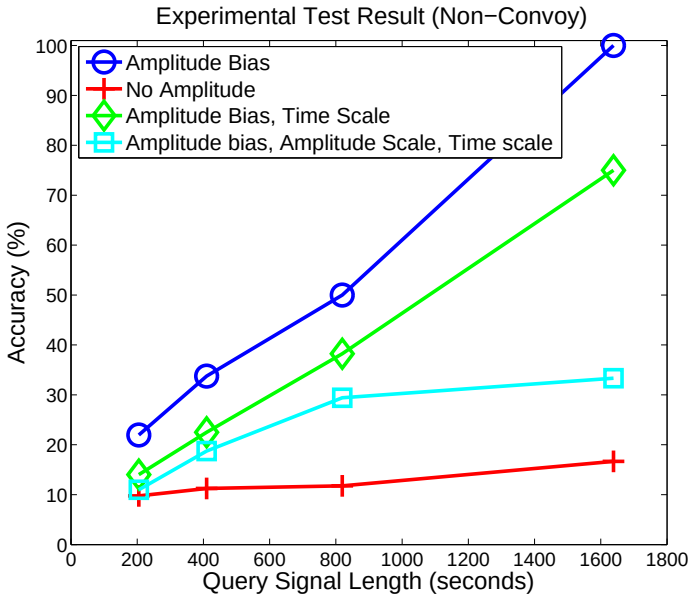

Figure 4.7: Accuracy curves for localization in the non-convoy dataset using different types of feature vectors.

4.5.1.2 Extended feature vector dimensionality

The number of point feature vectors that are utilized to construct an extended feature vector is another important design choice that determines the dimensionality of the extended feature vector. Figure 4.8 and Figure 4.9 show the effects of the feature vector dimension on the retrieval result for the case of SEMSE features. It can be seen that, for both the datasets, the method in which the extended feature vector has three point feature vectors outperforms the other cases. The performance of the 1 point feature vector case can be attributed to the lack of adequate uniqueness in the feature vector. On the other hand, the performance of the 5 point feature vectors case can be attributed to the decrease in robustness as an erroneous artifact such as an outlier is encoded into a larger number of feature

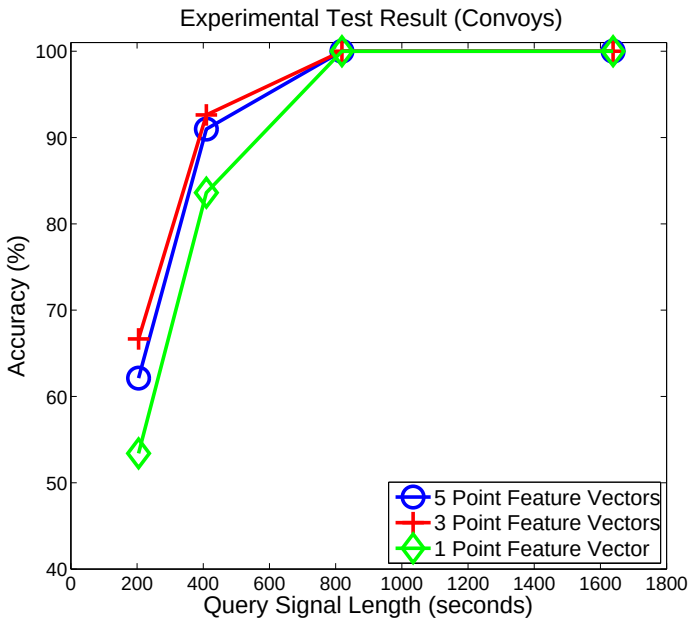

Figure 4.8: Effect of the feature vector dimensionality on the retrieval accuracy for the convoy dataset.

vectors.

4.5.2 Experiments

The pattern matching experiments are divided into two cases. In the single axis acceleration matching case, only the forward acceleration data is utilized in the matching process. On the other hand, the three axis acceleration matching case utilizes data from all X-Y-Z accelerations of a vehicle.

4.5.2.1 Single Axis Acceleration Matching

The pattern matching results comparing the SEMSE, MEMSE feature based matching and the traditional Euclidean distance method are shown in Figure 4.10 and

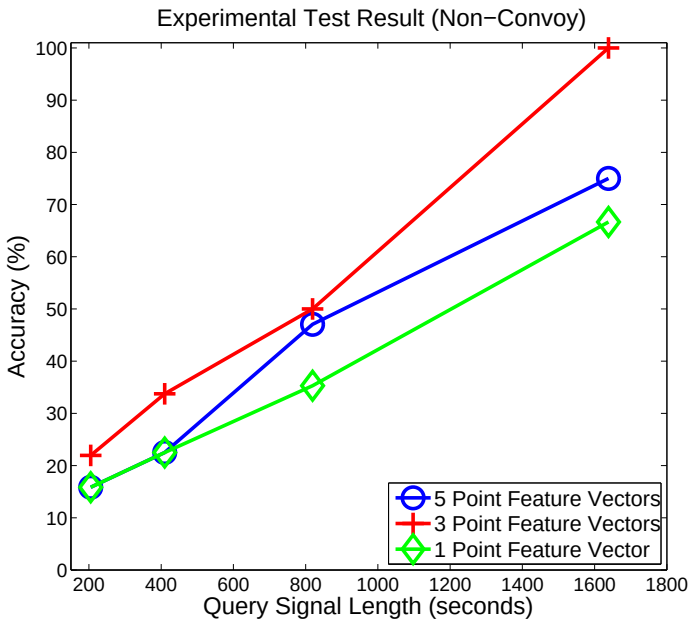

Figure 4.9: Effect of the feature vector dimensionality on the retrieval accuracy for the non-convoy dataset.

Figure 4.11. In order to simplify the explanation, the results of SEMSE method are first compared with the Euclidean distance method and then a comparison between the MEMSE and SEMSE features is delineated.

1. SEMSE features vs. Euclidean distance method: The aim of this analysis is to compare the results of a particular implementation of the MSE feature (Sequentially encoded) with the Euclidean distance method by evaluating them on the same dataset. The sequentially encoded feature has been described in Chapter 2. In this particular implementation, the amplitude bias invariant feature vector from Table 2.1 was chosen on the basis of the analysis performed in Section 4.5.1.1. A total of three point feature vectors were used in each extended feature vector as this gave the best performance as shown in

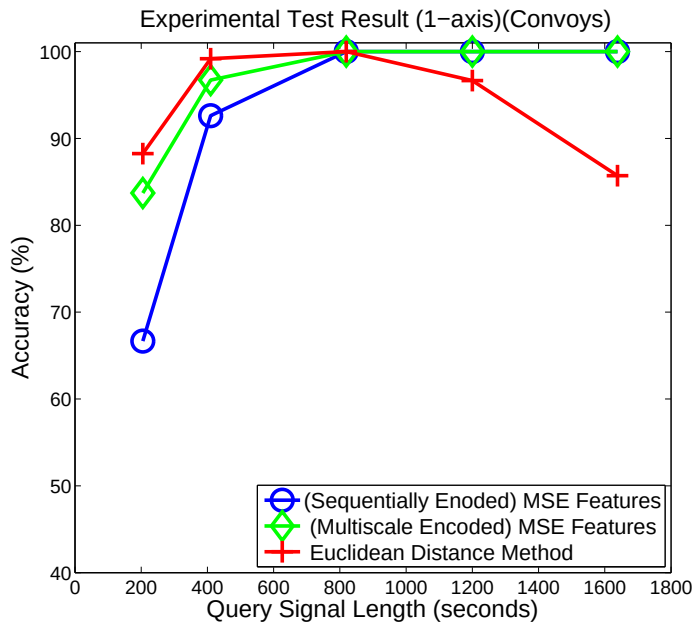

Figure 4.10: Accuracy curves for localization in the convoy dataset using different types of feature vectors.

Section 4.5.1.2. The dataset consists of acceleration data measured along a single axis and the results corresponding to convoy and non-convoy datasets are shown in Figure 4.10 and Figure 4.11 respectively. The low noise level in the query data of the convoy dataset, results in the excellent performance of both the Euclidean and the Sequentially Encoded MSE (SEMSE) feature vector. However, one can notice that the SEMSE feature outperforms the Euclidean method at longer query lengths and this can be attributed to the non-robust nature of the Euclidean distance metric. It must also be noted that the Euclidean distance method performs better than the SEMSE feature vector for shorter query lengths as there may not be adequate number of extrema in shorter query signals in order to create unique feature vec-

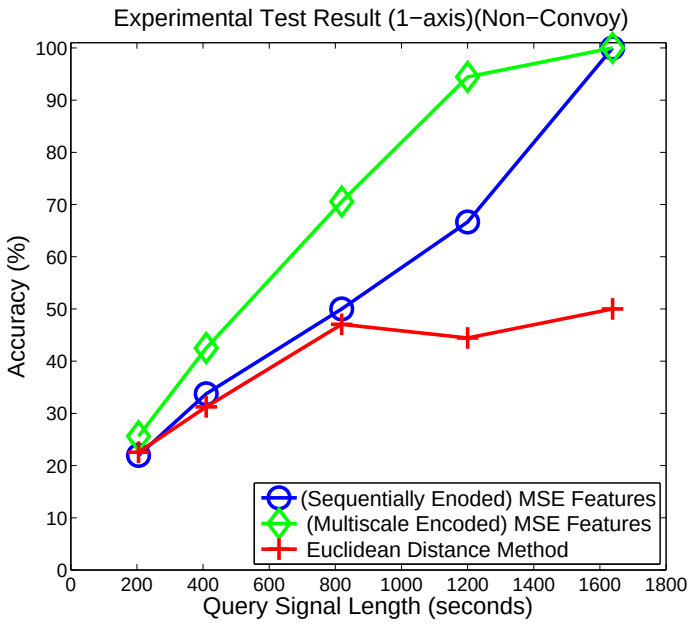

Figure 4.11: Accuracy curves for localization in the non-convoy dataset using different types of feature vectors.

tors. In the case of the non-convoy dataset, the SEMSE feature outperforms the Euclidean distance based method because of its capacity to withstand complex deformations in a signal. The performance difference is stark especially with large query lengths, because of the ability of the MSE method to encode low frequency features which are very unique. While the DTW [8] based methods might result in better performance than the Euclidean data, the high computational demands of these methods makes them infeasible for the current application. It must be noted that MSE method not only results in better accuracy but is also computationally very efficient.

2. MEMSE features vs. SEMSE features: Figure 4.10 and Figure 4.11 also present the results of using different types of encoding techniques to build

the extended feature vector. The MEMSE feature vector clearly leads to better performance than the SEMSE feature vector, but the nature of the Multi-Scale Encoding technique leads to large number of feature vectors and this in turn leads to a slightly larger memory footprint and computational effort in this case.

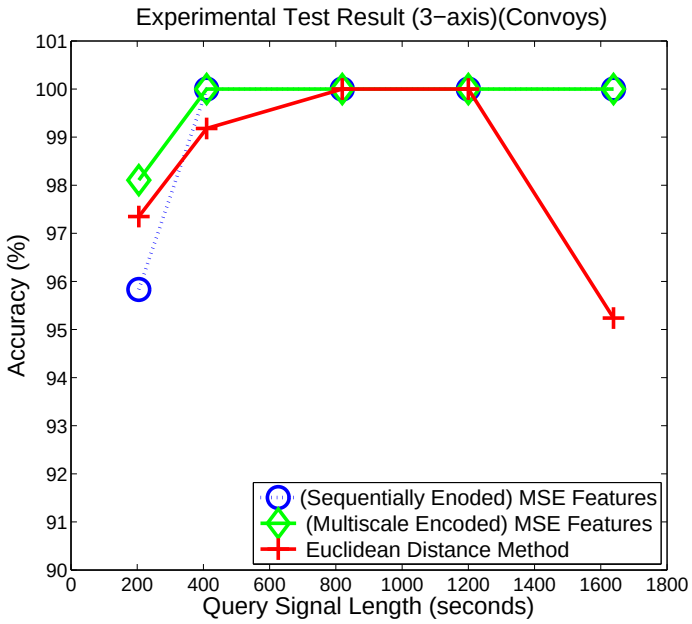

Figure 4.12: Accuracy curves for localization in the convoy dataset using different types of feature vectors.

4.5.2.2 Three Axis Acceleration Matching

A multiple KD-tree approach is utilized to incorporate acceleration data from different axis into the matching process. This method is similar to that presented in Chapter 2 except that three separate KD-trees are built to handle data along each axis. The position estimates from each KD-tree are assembled together into a

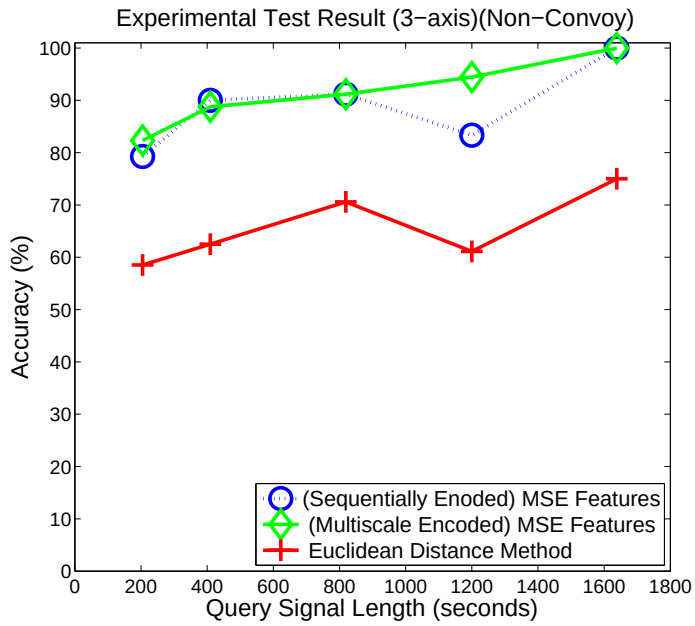

Figure 4.13: Accuracy curves for localization in the non-convoy dataset using different types of feature vectors.

single histogram that is used to decide the best position estimate. Figure 4.12 and Figure 4.13 show the results from this method for the convoy and the non-convoy datasets. One can clearly see that including new data sources into the matching process improves the accuracy of all the methods and that the MEMSE feature vector gives the best performance in all the cases.

4.6 Conclusion and Future Work

Overall, the research work demonstrates the utility of Multi-Scale Extrema Features for encoding acceleration data. The chapter also proposes the Multi-Scale Encoding method which leads to improvements in the performance under certain

conditions, when compared to the sequentially encoded method. This analysis has shown that, given long query signals, the acceleration data from a vehicle traveling on a particular road can be matched in spite of differences in driver behavior and traffic conditions. The performance advantages of using the feature vectors are clear, especially, in the case of longer query signals. The feature vectors that have been developed can not only be used for the originally mentioned clustering task but also be applied to other pattern recognition applications which rely on in-vehicle acceleration data. An interesting direction of future work would be to extract extrema from Iterative Mode Functions (IMFs) obtained from Emperical Mode Decomposition (EMD) [70] because of their ability to handle nonlinear and non-stationary characteristics that were observed in acceleration data. Future work could also be directed towards minimizing driver variability effects by preprocessing or by utilizing encoder data in order to correct for rate of travel through feature sets. Additionally, because the application of these results considers reliability of the same vehicle model in operation, all testing was done using identical Chevy Malibus as the fleet vehicles and therefore the present study does not account for distortions in the acceleration data due to the different dynamics exhibited by different vehicle types. Work is ongoing to study the dynamic influence of vehicle-to-vehicle differences.

Chapter 5

Optimal Extrema Features

5.1 Introduction

The task of extracting feature vectors from time series data is of fundamental importance in accomplishing a wide range of pattern recognition tasks. Examples include similarity-based pattern querying in time-series databases [4], classification of time-series data [71, 68, 72], discovery of anomalous subsequences [73] in time-series etc. The problem of representing and comparing time-series has hence seen a variety of solutions. The various approaches may roughly be categorized into three classes, namely: dimensionality reduction methods, distance metric methods, and interest point methods. A brief description of each approach is provided below.

1) **Dimensionality reduction methods:** In these methods, given a particular time series, a window of a certain length is chosen and the window is slid across the time series to extract all possible subsequences [4]. This initial step is often re-

ferred to as the sliding window method. A dimension reduction technique is then applied to each subsequence to obtain a feature vector to describe it. Different types of dimension reduction techniques have been proposed in literature. These have included extracting coefficients from Discrete Fourier Transform [4] , Discrete Wavelet Transform [6], Discrete Cosine Transform [4], and Singular Value Decomposition [5] of the subsequence. Methods have also been developed to represent a subsequence using piecewise constant values [17, 18], piecewise linear functions [74], chebyshev polynomials [75], and via symbols [76]. Most of the above methods utilize the euclidean distance between the generated feature vectors as a measure for calculating the distance between the feature vectors.

2) **Distance metric methods:** While Euclidean distance is the most straightforward method of comparison, this metric does not exhibit robustness under temporal distortion, outliers etc. which are typically incurred by time-series data. Considerable research effort has hence been directed towards building alternate measures of time-series comparison. One such advance is Dynamic Time Warping methods (DTW) which can tolerate temporal distortion [77]. Euclidean and DTW methods are further not robust to outliers, and this has lead to the development of "edit distance" methods. The concept of edit distance was borrowed from matching strings, and these methods enable matching by ignoring the dissimilar parts of the given time series [9, 71, 10].

3)**Interest point methods:** The use of features developed from key loca-

tions or "interest points" of a signal has been presented in literature in the form of landmarks [7]. From a retrieval standpoint, the fundamental novelty of this method of generating features when compared to the above methods is that it is not necessary to use the sliding window method in this case and this leads to a significant computational benefit. The use of extrema as interest points [78] has become increasing popular for time-series data analysis as evidenced by its recent applications in financial time-series analysis [7], audio hashing [79] and vehicle localization [63, 62].

Of the above time series analysis techniques, our proposed research focuses on interest point methods, and specifically extrema methods as they possess certain inherent capabilities that make them desirable in a variety of different pattern recognition tasks. Some of these properties are:

1) The essential information in a time-series is captured by adapting to its local temporal variation. Extrema features do this naturally since more extrema are found in regions of high variation and fewer where the time series is slowly varying.

2) Extrema methods are invariably economical in terms of signal representation and lead to a concise feature representation. This is because, for relatively well-behaved and structured time series, the number of extrema are much smaller than the overall length of the signal.

3) As demonstrated in [7] extrema or landmark features can be encoded to ex-

hibit invariance under common transformations of shifting and scaling that time-series incur. This, along with resilience to noise, make extrema features *robust*, where robustness is the ability to survive intact in spite of distortions/transformations being introduced into the signal.

Main contributions of this work:

The overarching goal of this chapter is to provide a principled algorithmic framework to handle the process of extrema detection and feature creation. In particular, our goal is to formalize the intuition provided by recent work in extracting and encoding extrema features [7, 79, 63] for various applications. More specifically, our key contributions are:

1) **Algorithmic framework for robust extrema extraction and encoding**: This chapter breaks down the process of generating features from raw data, by utilizing extrema, into three distinct steps: filtering, extrema detection and feature encoding. The proposed framework utilizes the properties of robustness, uniqueness, and cardinality as the basis for controlling each one of the above steps.

2) **Optimization of robustness**: Our central contribution is an optimization technique which extracts robust extrema, i.e. the extrema should be retained even as the time-series goes through distortions of noise, amplitude/time scaling, and shifts and other miscellaneous operations that may occur in capture or incidental processing of time-series data. Note that robustness is a highly desirable property, many practical applications involving time-series comparison/classification

are thwarted because the features do not withstand real-world distortions. Existing work [7] addresses this issue by a careful choice of a smoothing/pre-processing filter applied to the time-series prior to extrema extraction. In the proposed work, we *explicitly optimize* this filter using example (or training) time-series and demonstrate that this optimization reduces to a generalized eigenvalue problem.

3) **Generalized encoding method to control uniqueness and cardinality**: We also develop a new feature encoding method that provides better control over the properties of uniqueness and cardinality of the feature vectors that are generated. The properties of uniqueness and cardinality have direct influence on the accuracy and the computation involved with various pattern recognition problems and are therefore of central importance.

4) **Application of the framework to subsequence matching**: The above framework (optimization for robustness and the generalized encoding method) is then utilized to arrive at a solution to the problem of subsequence matching. The use of extrema methods for subsequence matching enables one to circumvent the need for a sliding window method [4]. Also, the extrema approach to subsequence matching uses fewer feature vectors and does not need post processing unlike the other sliding window methods. Moreover, the extrema methods outperform traditional sliding window methods in situations where the query signal is corrupted with bias noise, scale factor noise, and outliers simultaneously.

The chapter is organized in the following manner. Section 5.2 outlines the pro-

cess of obtaining feature vectors from raw data by using extrema. This section also defines the fundamental properties of robustness, uniqueness, and cardinality and details how these properties can be interrelated. Section 5.3 formulates the design of filter coefficients as a constrained optimization problem, and subsequently solves this as a generalized eigenvalue problem. A new encoding technique is proposed in order to provide a greater level of control over the uniqueness and cardinality of the features created from extrema. Section 5.4 presents results from two different set of experiments. First, we rigorously examine and validate the robustness provided by the optimization method in Section 5.3. Subsequently, an application for the proposed framework is demonstrated where in the performance of the extrema features obtained using the framework are compared with other techniques on both real and simulated data. Section 5.5 summarizes our contributions and concludes the chapter.

5.2 Extrema Features: Principal Issues

We introduce first the relatively generic process that is used to generate feature vectors from raw time-series data while using extrema. Next, we describe properties of extrema features which have a crucial effect on the pattern recognition tasks that utilize them.

5.2.1 Background

Extrema techniques create feature vectors by encoding the amplitude, relative locations or other properties of the extrema. A block diagram illustrating the steps involved in a generic process that seeks to encode features from extrema is shown in Figure 5.1. The first step in the process is filtering the signal to reduce noise and/or to enhance significant aspects of the time series data. The second step involves extracting the *'extrema'* from the filtered signal. The process of extracting the extrema itself could be based on using simple thresholds or by considering additional properties of an extrema that try to ascertain whether each extrema is "significant" or not. Each individual significant extremum is encoded into a *'feature vector'* in the third step. We address the individual elements of this feature vector as *'extrema features'*, and the collection of feature vectors that represent a time series is called the *'feature representation'*.

The above three steps may not always be implemented in conjunction and many pattern recognition problems utilize portions of the three steps described above. It is very common for algorithms to use the first two steps of filtering and extrema detection to detect interest points and subsequently either use alternate encoding methods (that do not rely on extrema)[30][80] or no encoding techniques at all [34] [81] [82].

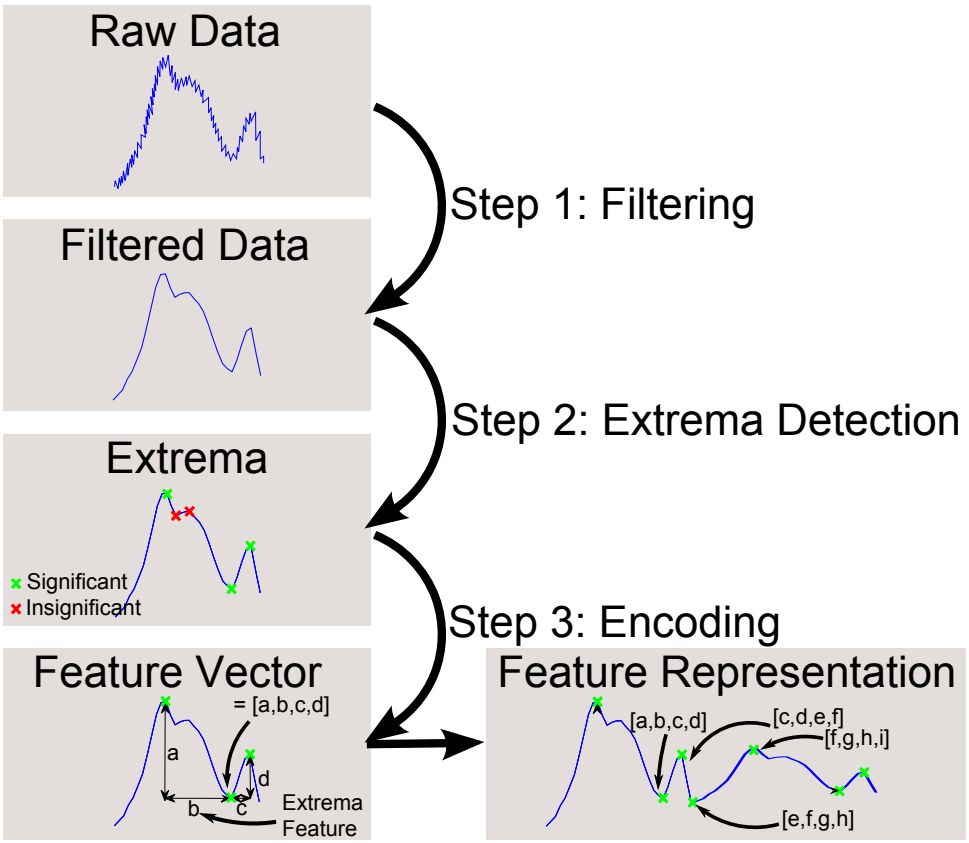

Figure 5.1: The steps involved in creating features from extrema.

5.2.2 Properties Governing Performance

Given the benefits enabled by the use of extrema features, we discuss next the key properties of extrema features which in turn have a critical bearing on their application in pattern matching tasks.

1) **Robustness**: It is highly desirable that the extrema from a time-series signal are robust under some typical distortions to the time-series. These distortions may include additive noise, amplitude/time scaling and shifts, and a host of other miscellaneous distortions. In fact, one of the key motivating factors for the use of extrema in time-series [7] is the fact that extrema features, if well designed,

can enable much better robustness under a variety of distortions over classical alternatives like using Euclidean distance to compare different time series.

2) **Uniqueness**: Certain pattern matching tasks require that the extrema that are extracted lead to unique feature vectors. Intuitively, uniqueness may be interpreted as the feature representation of a given time-series to be considered as its *fingerprint*, i.e. a compact, reduced-dimension identifier of the time-series that is sufficiently distinct from other features.

3) **Cardinality**: Cardinality refers to the number of extrema that are extracted from a signal. The cardinality of extrema is usually directly proportional to the number of feature vectors that are encoded from those extrema. The number of feature vectors that are extracted will in turn affect the computation that is required to perform a given pattern recognition task.

Control over the above properties can profoundly impact the use of extrema features in pattern matching applications. As an example, for classifying time-series, an appropriate balance of robustness and uniqueness is needed. In particular, the robustness should be high enough so that the classification task takes place in spite of noise, but shouldn't be so high as to inhibit the classification process itself. Likewise, a sufficiently high level of uniqueness is needed so that class-specific discriminative information is captured by the extrema feature representation. An alternate application where extremely high uniqueness may be needed is time-series retrieval or subsequence matching - here, the feature representation must (ideally)

be uniquely tied to each time-series in a database of canonical time-series. Finally, cardinality (proportional to the number of feature vectors/size of feature representation) of a time-series may be controlled so as to provide a desirable computation versus performance trade-off. The next section delves into the development of tools and techniques to either explicitly optimize or intuitively control these properties in each of the three steps described in Figure 5.1.

5.3 Tools and Techniques for Adaptive Extrema Feature Extraction

5.3.1 Filtering Step: Optimizing robustness via filter coefficient optimization

The extrema extraction step is usually preceded by filtering in order to reduce noise and to enhance the prominent features of the time series. The earliest approach which introduced landmarks or interest points for time-series analysis [7] advocated the use of a smoothing filter. More recently, applications inspired by wavelet kernels have been utilized in various pattern recognition applications [83] [62]. Kicey et al [34] have shown that the extrema from the wavelet transform can be used to synthesize approximations to the time-series. Extrema in a signal are often used as candidates for generating the hash vectors. For example, the algorithm

behind the shazham audio identification service [37] takes the short term Fourier transform of the music signal and utilizes extrema in the resultant transformed data to create the hash vectors [38]. This is another situation in which linear filtering, followed by extrema detection, is utilized in generating feature vectors.

Next, we describe the process of obtaining the filter that maximizes the robustness of the extrema obtained from the filtering process. This filter is obtained by performing an optimization routine on a training dataset, *thus adapting* the filter to the particular dataset at hand.

5.3.1.1 Derivation of the optimal filter

The most desirable extrema are those that remain identifiable and unaffected when distortions are introduced into a signal and can be referred to as robust extrema. A filter that results in the most robust extrema from being extracted from the filtered signal can be defined as the "optimal filter" in this context. In order to find the optimal filter that maximizes the robustness of the extrema, it is necessary to geometrically visualize the process that occurs when selecting an extrema. The following derivation demonstrates that the extrema selection process is equivalent to a geometric problem of partitioning data points in a hyperspace. It also shows that the filtering operation can be interpreted as bounding the selected extrema by two hyperplanes. The derivation is as follows:

For a given discrete signal $x[n]$ and an acausal FIR filter $h[n]$ with $2N + 1$

coefficients, the corresponding filtered signal is denoted by

$$y[n] = \sum_{i=-N}^{N} b_i x[n-i] \tag{5.1}$$

where b_i are the filter taps for the filter $h[n]$,

$$As\ h[n]\ has\ 2N+1\ taps,\ b_i = 0,\ i \notin [-N, N] \tag{5.2}$$

It is desirable to have an odd number of taps in the filter so that the filter has an identical number of taps on either side of a particular point. This selection is advisable for most signals unless there are any specific reasons to choose a filter with different number of taps on either side of a point.

If $y[n_0]$ is a maxima of the filtered signal then by definition it must satisfy (5.3) and (5.4):

$$y[n_0] > y[n_0 + 1] \tag{5.3}$$

$$y[n_0] > y[n_0 - 1] \tag{5.4}$$

Substituting (5.1) into equation (5.3),

$$\sum_{i=-N}^{N} b_i x[n_0 - i] > \sum_{i=-N}^{N} b_i x[n_0 + 1 - i] \tag{5.5}$$

$$\iff \sum_{i=-N}^{N} b_i x[n_0 - i] > \sum_{i=-N-1}^{N-1} b_{i+1} x[n_0 - i] \tag{5.6}$$

$$\iff \sum_{i=-N-1}^{N+1} b_i x[n_0 - i] > \sum_{i=-N-1}^{N+1} b_{i+1} x[n_0 - i] \tag{5.7}$$

$$as\ b_{-N-1} = b_{N+1} = b_{N+2} = 0 (from\ (5.2))$$

$$\iff \sum_{i=-N-1}^{N+1} (b_i - b_{i+1}) x[n_0 - i] > 0 \tag{5.8}$$

Performing a similar computation for equation (5.4),

$$y[n_0] > y[n_0 - 1] \tag{5.9}$$

We obtain

$$\sum_{i=-N-1}^{N+1} (b_i - b_{i-1}) x[n_0 - i] > 0 \tag{5.10}$$

Let

$$\alpha_i = b_i - b_{i-1} \tag{5.11}$$

Then substituting $i = i + 1$

$$\alpha_{i+1} = b_{i+1} - b_i = -(b_i - b_{i+1}) \tag{5.12}$$

$$\therefore b_i - b_{i+1} = -\alpha_{i+1} \tag{5.13}$$

Substituting (5.11) into (5.10) we obtain

$$\sum_{i=-N-1}^{N+1} \alpha_i x[n_0 - i] > 0 \tag{5.14}$$

Substituting (5.13) into (5.8) we obtain

$$\sum_{i=-N-1}^{N+1} -\alpha_{i+1} x[n_0 - i] > 0 \tag{5.15}$$

Given the two equations (5.14) and (5.15) that need to be satisfied for $y[n_0]$ to be a maxima, these conditions can be interpreted in a geometric manner. Consider the $2N+3$ long sequence of $x[n]$ where $n \in [n_0 - N - 1, n_0 + N + 1]$ as a point x in a $2N+3$ dimensional space. Let the sequence α_i where $i \in [-N-1, N+1]$ be denoted by a vector $\overline{\alpha_1}$ and the sequence $-\alpha_{i+1}$ where $i \in [-N-1, N+1]$ be denoted by vector $\overline{\alpha_2}$ in the same $2N+3$ dimensional hyperspace. Then the conditions given by (5.14) and (5.15) are required for $Y[n_0]$ to be a maxima can be interpreted in the following manner:

1. Imagine a hyper plane passing through the origin and perpendicular to $\overline{\alpha_1}$. Any point $\overline{x}$ ($2N+3$ dimensional) that lies to one side of this hyperplane will satisfy

$$\sum_{i=-N-1}^{N+1} \alpha_i x[n_0 - i] > 0 \tag{5.16}$$

   and all the points that lie on the other side will satisfy

$$\sum_{i=-N-1}^{N+1} \alpha_i x[n_0 - i] < 0 \tag{5.17}$$

2. One can similarly imagine a hyperplane corresponding to $\overline{\alpha_2}$ that divides the

entire hyperspace into two regions corresponding to each of the conditions given in equations (5.18) and (5.19).

$$\sum_{i=-N-1}^{N+1} -\alpha_{i+1} x[n_0 - i] > 0 \tag{5.18}$$

$$\sum_{i=-N-1}^{N+1} -\alpha_{i+1} x[n_0 - i] < 0 \tag{5.19}$$

3. From the above information, the region where equations (5.20) and (5.21) are satisfied is given by the intersection of two of the regions created by the hyperplanes perpendicular to $\overline{\alpha_1}$ and $\overline{\alpha_2}$.

$$\sum_{i=-N-1}^{N+1} \alpha_i x[n_0 - i] > 0 \tag{5.20}$$

$$\sum_{i=-N-1}^{N+1} -\alpha_{i+1} x[n_0 - i] > 0 \tag{5.21}$$

The above explanation is illustrated in Figure 5.2 which shows a two dimensional projection of the $2N + 3$ dimensional space. From the above derivation, it can be seen that the hyperspace is divided into four regions. While one of the four regions is the maxima region the conditions that are satisfied in the other regions will lead to a corresponding minima region, a decreasing region and an increasing region for a particular filter as shown in Figure 5.2. For a given signal x[n], one can extract all possible $2N + 3$ long subsequences of points and populate them in

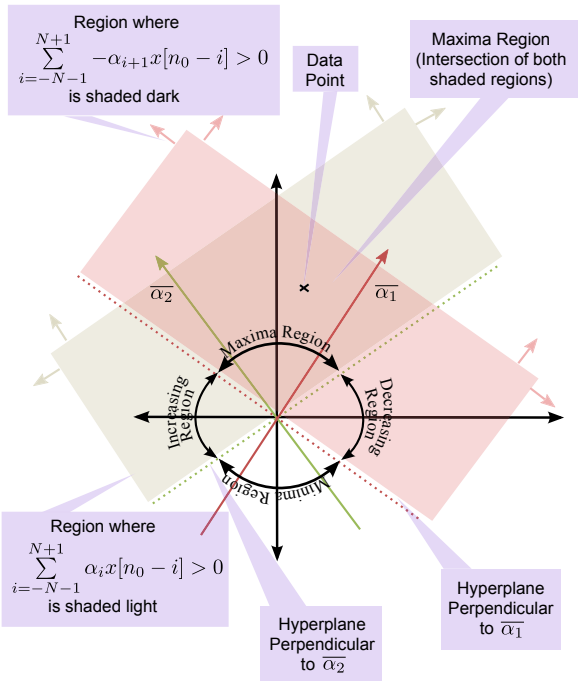

Figure 5.2: A two dimensional projection of the filter hyperplanes and the regions associated with conditions (5.16) and (5.18).

a feature space and build hyper planes corresponding to a particular filter. Then all the subsequences that lie within the maxima region of the feature space will correspond to maxima in the filtered signal.

5.3.1.2 Robustness

Given that noise is expected to be present within the original signal, robustness is defined as the ability of the maxima and minima of the filtered signal to remain intact in spite of the addition of noise. In terms of the above geometric interpretation, that would mean that, after the addition of noise, the subsequences present in the maxima region remain in the maxima region and similarly other subsequences

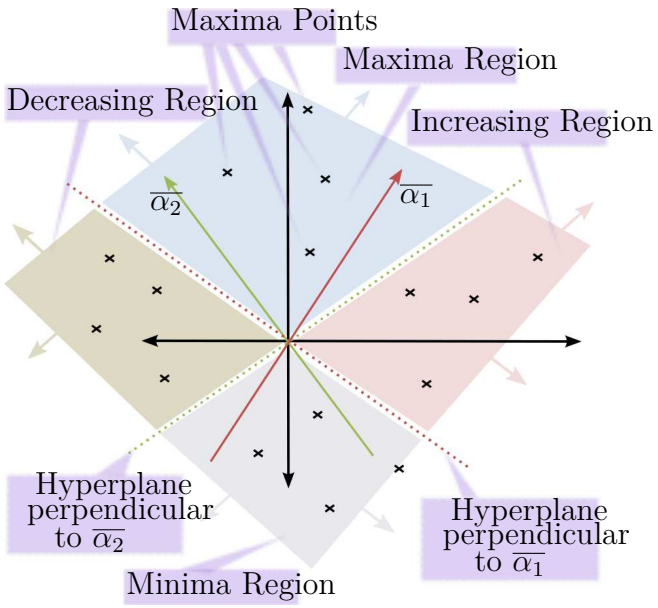

Figure 5.3: The two hyperplanes corresponding to the filter will divide the hyperspace containing the subsequences into four different regions.

from other regions (minima, decreasing, increasing) should remain in their respective regions. This would ensure that all the maxima and minima are intact and no new extrema are formed.

Given the above interpretation, the robustness of a subsequence can now be defined as the proximity to crossing a boundary. A particularly attractive definition is to use the sum of the squared distances of a subsequence to both the hyper planes of a filter. Given this definition, it should be possible to find an optimal filter such that the sum of the squared distances to the hyperplanes of all the subsequences corresponding to that filter is maximized. It is important to note that this particular definition of robustness has implications on the nature of the distortions that the signal is likely to tolerate in pattern matching. For example, if

one assumes that the noise to the signal is i.i.d, then the nature of distribution of a particular subsequence in the $2N + 3$ dimensional hyperspace would be spherical. The outer radius of such a sphere would indicate the severity of the distortion, and as long as this outer radius is less than the perpendicular distance of a given subsequence to the filter hyperplanes, the subsequence would not change its state from being an extrema or non-extrema. Thus, the current definition of robustness is apt in case of an i.i.d noise whose distribution's range is less than twice the perpendicular distance of a point to the closest boundary hyperplane.

5.3.1.3 Derivation for the optimally robust filter

Let $\overline{\alpha_1}$ and $\overline{\alpha_2}$ described in the previous subsection be defined as follows:

$$\overline{\alpha_1} = [\alpha_{-N-1}\ \alpha_{-N}\ \alpha_{-N+1}\ ...\ ...\ \alpha_0\ ...\ ...\ \alpha_N\ \alpha_{N+1}\ ] \tag{5.22}$$

As the filter is of length $2N + 1$, $\alpha_{-N-1} = 0$. Let

$$\overline{\alpha_2} = [\alpha_{-N}\ \alpha_{-N+1}\ \alpha_{-N+2}\ ...\ ...\ \alpha_1\ ...\ ...\ \alpha_{N+1}\ \alpha_{N+2}\ ] \tag{5.23}$$

Similarly, as the filter is of length $2N + 1$ so $\alpha_{-N+2} = 0$. Let the above two vectors represent perpendicular unit vectors to the hyperplanes for a particular filter. Therefore, the requirement that they be unit vectors imposes a constraint

for $\overline{\alpha_1}$ and $\overline{\alpha_2}$ which is given by (5.24).

$$\sum_{i=-N}^{N+1} \alpha_i^2 = 1 \tag{5.24}$$

From (5.2) and (5.11), one obtains the additional constraint

$$\sum_{i=-N}^{N+1} \alpha_i = 0 \tag{5.25}$$

The above condition can be rewritten as

$$\alpha_{N+1} = -(\alpha_{-N} + \alpha_{-N+1} + ... + \alpha_N) \tag{5.26}$$

The sum of squared perpendicular distances of a particular point represented by a subsequence

$$[x_{-N-1}\ x_{-N}\ x_{-N+1}\ ...\ ...\ x_0\ ...\ ...\ x_{N-1}\ x_N\ x_{N+1}\ ] \tag{5.27}$$

to hyperplanes perpendicular to $\overline{\alpha_1}$ and $\overline{\alpha_2}$ is given by the square of the dot product of the subsequence to $\overline{\alpha_1}$ and $\overline{\alpha_2}$ as shown in Equation (5.28).

$$\left[\sum_{i=-N}^{N+1} \alpha_i x_i\right]^2 + \left[\sum_{i=-N}^{N+1} -\alpha_i x_{i-1}\right]^2 \tag{5.28}$$

Therefore, the total sum of the squared perpendicular distances of all the $'M-N'$ points in the hyperspace (U_R) is given by

$$U_R = \sum_{j=N+1}^{M} \left[\left[\sum_{i=-N}^{N+1} \alpha_i x_{i+j} \right]^2 + \left[\sum_{i=-N}^{N+1} -\alpha_i x_{i+j-1} \right]^2 \right] \tag{5.29}$$

where $x_0, x_1, x_2, x_3, x_4, ..., x_{M+N+1}$ represents the time series for which an optimal filter is being sought. Substituting (5.26) into the above equation we obtain

$$\begin{aligned} U_R = \sum_{j=N+1}^{M} \Bigg[\left[\sum_{i=-N}^{N} \alpha_i (x_{i+j} - x_{j+N+1}) \right]^2 \\ + \left[\sum_{i=-N}^{N+1} \alpha_i (x_{i+j-1} - x_{j+N}) \right]^2 \Bigg] \end{aligned} \tag{5.30}$$

Let y_j denote a $2N+1$ vector whose i^{th} element is denoted by $(x_{-N-1+i+j} - x_{N+1+j})$. Next, let z_j denote a $2N+1$ vector whose i^{th} element is denoted by $(x_{-N-1+i+j-1} - x_{N+j})$. Finally, let α denote a $2N+1$ vector whose i^{th} element is denoted by α_{-N-1+i}. Then the above equation can be written in the matrix form as

$$U_R = \sum_{j=N+1}^{M} \left[\alpha^T y_j y_j^T \alpha + \alpha^T z_j z_j^T \alpha \right] \tag{5.31}$$

$$\iff U_R = \alpha^T \left[\sum_{j=N+1}^{M} [y_j y_j^T + z_j z_j^T] \right] \alpha \tag{5.32}$$

Let

$$X_{Data} = \sum_{j=N+1}^{M} [y_j y_j^T + z_j z_j^T] \tag{5.33}$$

Clearly X_{Data} is positive semi definite from its very definition (5.30), but for most datasets for which $N << M$, the matrix X_{Data} will be positive definite. There are a few times series which are deterministic, in which case the time series satisfies a certain recurrence relation, and which satisfy the condition $N << M$ but would result in $det(X_{Data}) = 0$. As analysis of such 'special' time series is not the objective of this chapter, further derivation is performed under the assumption $N << M$ and that X_{Data} is positive definite. Then, (5.32) can be expressed as

$$U_R = \alpha^T X_{Data} \alpha \tag{5.34}$$

In order to maximize U_R under the condition (5.24) one can utilize the Lagrangian multiplier method [84].Thus L given in (5.35) needs to be minimized

$$L = -\alpha^T X_{Data} \alpha + \nu \left[\sum_{i=-N}^{N+1} (\alpha_i)^2 - 1 \right] \tag{5.35}$$

Using (5.26)

$$L = -\alpha^T X_{Data} \alpha + \nu \left[\sum_{i=-N}^{N} (\alpha_i)^2 + (\sum_{i=-N}^{N} (\alpha_i))^2 - 1 \right] \tag{5.36}$$

Let $j_{1,2N+1}$ denote a $2N+1$ unit vector, then the above equation can be simplified to

$$L = -\alpha^T X_{Data} \alpha + \nu [\alpha^T I \alpha + \alpha^T j_{1,2N+1}^T j_{1,2N+1} \alpha - 1] \tag{5.37}$$

Let J_{2N+1} denote a $2N+1$ by $2N+1$ unit matrix,

$$L = -\alpha^T X_{Data}\alpha + \nu[\alpha^T[I + J_{2N+1}]\alpha - 1] \tag{5.38}$$

Given that there is one quadratic program with one quadratic inequality constraint, this problem is often referred to as the trust region sub problem in optimization theory literature [85][86]. Following the Lagrangian multiplier method, the stationary points are given by

$$\nabla_\alpha L = 0 \; and \; \nabla_\nu L = 0 \tag{5.39}$$

The condition $\nabla_\nu L = 0$ results in the Karush-Kuhn-Tucker condition

$$\alpha^T[I + J_{2N+1}]\alpha - 1 = 0 \tag{5.40}$$

Once the α vector satisfying the condition $\nabla_\alpha L = 0$ is obtained, it can be multiplied by a constant to satisfy (5.40) provided that the modified vector can still satisfy $\nabla_\alpha L = 0$. Solving for $\nabla_\alpha L = 0$

$$\nabla_\alpha L = 2(-X_{Data}\alpha + \nu[I + J_{2N+1}]\alpha) = 0 \tag{5.41}$$

$$\therefore X_{Data}\alpha = \nu[I + J_{2N+1}]\alpha \tag{5.42}$$

As ν is a scalar,

$$[I + J_{2N+1}]^{-1} X_{Data}\alpha = \nu\alpha \tag{5.43}$$

The above equation is the generalized eigenvalue problem. Given that X_{Data} is positive definite, one can rewrite the above equation in the form

$$[I + J_{2N+1}]^{-1} (X_{Data})^{1/2} (X_{Data})^{1/2}\alpha = \nu\alpha \tag{5.44}$$

where $(X_{Data})^{1/2}$ can be obtained from eigenvalue decomposition. Multiplying both sides of (5.44) with $(X_{Data})^{1/2}$

$$\begin{aligned}(X_{Data})^{1/2}[I + J_{2N+1}]^{-1} (X_{Data})^{1/2} (X_{Data})^{1/2}\alpha \\ = \nu (X_{Data})^{1/2}\alpha\end{aligned} \tag{5.45}$$

Substituting $w = (X_{Data})^{1/2}\alpha$ into (5.45) results in the regular eigenvalue problem

$$(X_{Data})^{1/2}[I + J_{2N+1}]^{-1} (X_{Data})^{1/2} w = \nu w \tag{5.46}$$

Thus the eigenvalues (ν_k) and eigenvectors (w_k) corresponding to the symmetric positive semi definite matrix $(X_{Data})^{1/2}[I + J_{2N+1}]^{-1}(X_{Data})^{1/2}$, will lead to the solution $(X_{Data})^{-1/2} w_k$ for the $'\alpha'$ vector.

Given that the following problem is a trust region sub-problem, it has been shown that strong duality is satisfied [85] and so the Lagrangian relaxation for this

non-convex problem is exact [86]. Writing the Lagrange dual function

$$g(\nu) = inf_{a \in Domain} - a^T X_{Data} a + \nu[a^T[I + J_{2N+1}]a - 1] \tag{5.47}$$

As the stationary points satisfy (5.42), (5.47) can be simplified to

$$\begin{aligned} g(\nu) = inf_{a \in Domain} - a^T \nu[I + J_{2N+1}]a + \\ \nu[a^T[I + J_{2N+1}]a - 1] \end{aligned} \tag{5.48}$$

Therefore, the Lagrange dual function is

$$g(\nu) = inf_{a \in Domain} - \nu \tag{5.49}$$

Hence the optimal value for the function in (5.34) is given by the largest eigenvalue. Thus, from the different eigenvector solutions that are obtained, the optimal solution is given by the eigenvector corresponding to the maximum eigenvalue. The above optimization process contained no constraints on the number of extrema that result from the filtered signal. Therefore, this procedure could theoretically result in a filter for which no extrema are created or a situation in which each and every point in the signal is an extrema. In case one obtains no extrema points from the optimal filter (corresponding to the maximal eigenvalue), then one can utilize subsequent eigenvalues and eigenvectors that follow the maximal eigenvalue to obtain optimal filters that result in extrema. In case all the values or a large percentage

of the signal values are chosen as extrema then it is advisable to smooth the signal and then use the optimization process to extract the filter. The above derivation primarily pertains to situations in which extrema detection is performed by linear filtering and subsequent extrema identification. Given the broad range of areas and problems that involve extrema detection, it is clearly of interest to optimize the filtering step in order to shape the properties of the extrema that are detected. Linear filtering followed by extrema identification is not the only method of extrema detection and researchers have put forward nonlinear processing techniques such as Minimal Distance/Percentage principle based smoothing [7], Perceptually Important Points [39] to identify extrema and while these techniques may not directly benefit from the current analysis, one could potentially utilize the proposed framework/definitions for optimizing these techniques. It is important to note that linear filtering can be implemented very efficiently by using the Fast-Fourier Transform and most nonlinear processing techniques require considerable computation which may make them less desirable especially for large databases.

The proposed method has certain conceptual similarities to filter optimization methods for extracting extrema is in the context of edge detection in images. Canny [81] setup an optimization problem for identifying a filter that would result in an edge with desirable properties. The problem was set up in the context of a one-dimensional signal and therefore the edge detection problem became an extrema detection problem. A large body of these methods use iterative or numerical

optimization techniques [81] [82] [87] in order to obtain the optimal filter. This is in contrast to the current problem which has been carefully formulated to yield (non-iterative) closed-form solutions.

This subsection presents the process of optimizing robustness in the filtering step of the extrema feature extraction method. The properties of uniqueness and cardinality haven't been dealt in this step as they do not lend themselves to mathematical tractability in terms of obtaining a closed form solution. The subsequent subsections delve into methods to control all the aforementioned properties (robustness, uniqueness and cardinality) in the extrema detection and the encoding step.

5.3.2 Extrema Detection Step: (Robustness and Uniqueness) versus Cardinality

The most straightforward method of detecting extrema from the filtered signal is by following the conditions given in equations (5.3) and (5.4). However, there are a variety of other techniques to identify extrema in a signal [7, 88]. The central idea behind the different methods of extrema detection is to cull the 'weak extrema' which would easily appear or disappear depending on the distortions introduced into the signal. Thus, the extrema detection step is geared towards decreasing the cardinality, while increasing the robustness and uniqueness of the extrema that are selected.

In our implementation, a two step process was followed for extrema detection. In the first step, potential extrema candidates were detected by using the conditions given in equations (5.3) and (5.4). The second step consists of choosing extrema, from among the potential candidates, based on the threshold of their amplitude distance to each of their neighbors.

5.3.3 Encoding Step: Obtaining better control over the properties by using a generalized encoding process

5.3.3.1 Background

The procedure for encoding feature vectors from extrema plays a critical role in determining the performance of the corresponding feature vectors in the underlying pattern recognition task. The properties that are of interest in the case of the feature encoding step not only include the usual robustness, uniqueness and cardinality but also the dimensionality of the encoded feature vectors. Unlike the other properties (Robustness, Uniqueness and Cardinality) which are relevant in all the three steps described in Figure 5.1, the property of dimensionality is relevant only in the feature encoding step and is therefore dealt solely in this subsection. However, it is important to note that dimensionality of the feature vector not only has implications on the robustness and uniqueness of the eventual feature vector but also has a critical bearing on the computation associated with feature matching. Higher dimensions typically imply more computational effort and this phenomenon

is often described as the 'Curse of Dimensionality'[2].

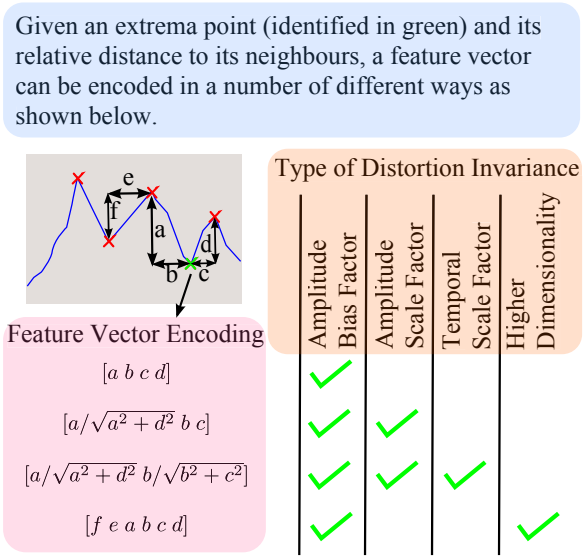

Figure 5.4: Different methods for sequential encoding of extrema.

5.3.3.1.1 Trade-off between robustness and uniqueness via encoding variants A feature vector can be created by encoding the relative distances of an extrema to each of its neighbors [7]. Some well known and widely used techniques [7, 79, 63, 62] of feature encoding from extrema are shown in Figure 5.4. These methods are presented to highlight the trade off between robustness and uniqueness that occurs amongst the different encoding techniques. Typically, an encoding that leads to an increase in robustness to a particular type of distortion will lead to a corresponding decrease in the uniqueness of the feature vector.

5.3.3.2 Proposed generalized encoding process for better control over uniqueness and cardinality

The above encoding techniques can be described as being sequential in nature as sequences of extrema are used to create feature vectors [7, 62]. These sequential encoding methods have the following limitations:

1) The cardinality of the feature vectors is limited to the number of extrema in the signal. The limited cardinality implies that the users do not have the flexibility to increase or decrease the number of features being produced.

2) The sequential nature of the encoding technique is unable to capture certain unique information of the underlying signal that is not sequential in nature.

In order to overcome these drawbacks, a two step generalized encoding procedure is proposed in this section. In the first step, short feature vectors called primitives are created using methods such as those suggested in Figure 5.4. The final feature vectors are created in the second step in which a primitive at an extrema is combined with other primitives in its neighborhood. The total number of combinations performed with a primitive at a given extrema is called the 'combination parameter'. In implementation, the encoding is restricted to either the left neighborhood or the right neighborhood of each extrema in order to avoid encoding the same information into multiple feature vectors.

Figure 5.5 provides an illustration of the two step generalized encoding process and a combination parameter of three is used in this particular case. Therefore,

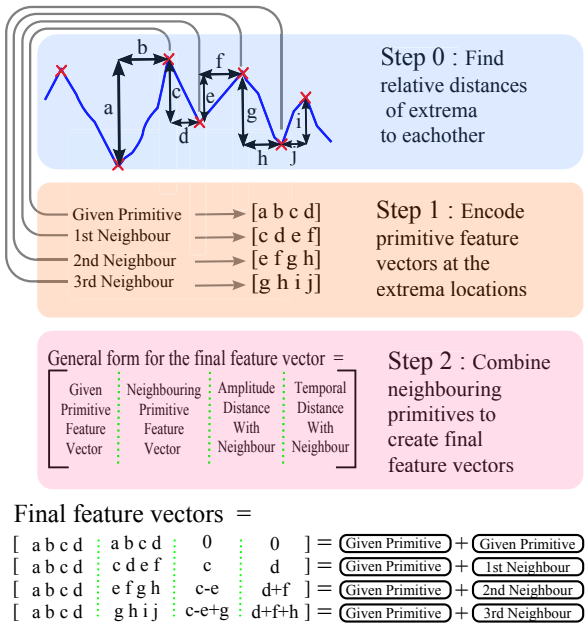

Figure 5.5: The steps involved in the generalized encoding method of creating feature vectors from extrema.

the given primitive is combined with itself and with each of the three primitives present in its immediate right neighborhood in order to obtain four separate feature vectors. The cardinality of the feature vectors can be controlled by controlling the combination parameter. This process of encoding is clearly a generalization of the sequential encoding process because one would obtain sequential features in case the combination parameter is set to zero or one.

The second step of the two stage process leads to an overall increase in the dimensionality and cardinality of the feature vectors. The increase in dimensionality has the effect of increasing the uniqueness. However, an increase in the value of the combination parameter will in turn increase the cardinality of the feature vectors and this will lead to a decrease in uniqueness as more features are entering

the same higher dimensional feature space. Thus the generalized encoding process enables design choices in the first stage corresponding to a trade off between uniqueness and robustness and in the second stage where the control of the combination provides a trade off between uniqueness and cardinality. These capacity to control these properties can be used to obtain better performance at the cost of extra computation as shown in the next section.

5.4 Experimental Results

5.4.1 Experimental Overview

All the datasets and MATLAB code that was utilized to obtain all the subsequent plots (Figure 5.6-Figure 5.23) have been made available online at www.personal.psu.edu/pkv106/robustextrema.html.

Table. 5.1 lists definitions of a few traditional filters that are typically used for extracting extrema from raw data. While Gaussian, Derivate of Gauss (DerGauss) and Sombrero filters have been used for extrema detection [81, 30] , the sinusoidal filters are mostly used in the context of audio data [38]. The delta and the constant filters are used as references to illustrate the effects of no filtering and using a filter with equal taps. The traditional filters as well as the optimal filters (obtained from utilizing the method described in Section 5.3.1) on varied data sets are plotted in Figure 5.6.

5.4.1.1 Datasets used in Experiments

The main purpose of this section is to perform experiments that validate the optimization method presented in section **??** and also demonstrate the advantages obtained from using the optimal filter in a given pattern matching task. The datasets used in the above experiments are described in this subsection. The experimental results for the validation of the optimization methodology and the pattern matching results are presented in sections 5.4.2 and 5.4.3 respectively.

Time-series data from different domains including mechanical, electrical, financial and biomedical fields was chosen to illustrate the broad applicability of the proposed extrema based feature extraction framework. Briefly, the data sets are:

1) Pitch Data (Real): The dataset consists of road pitch data collected from an in-vehicle data acquisition system [62]. The data was collected as a part of the NCHRP 22-21 median design project.

2) Music Data (Real): Music time series was obtained by concatenating music snippets from different audio recordings. The audio data was downloaded from the following website [38].

3) Stock Data (Real) : The data consists of price history data from various stocks that are concatenated with each other. The data was downloaded from the Yahoo website by using a matlab code.

4) EEG Data (Real): EEG time series data obtained from different trials on human subjects is concatenated to obtain the time series used in this analysis. The

data was downloaded from the following source [89].

5) Gaussian Random Walk (GRW) (Simulated): The data was obtained from a MATLAB simulation. The MATLAB code to obtain the simulated data is given by $Data = cumsum(randn(2^{16}, 1))$;

5.4.2 Results A: Validating Robustness as Imparted by Filter Optimization

This test measures the stability of the extrema created by using different filters which are shown in Figure 5.6. The testing procedure, given a particular dataset and a particular filter, is as follows:

1) The signal is filtered with the given filter and the locations of the extrema in the filtered signal are identified by using the conditions given in Eqn. (5.3) and Eqn. (5.4).

2) The signal is now corrupted with a particular level of noise (Gaussian Noise) and the new signal is again filtered and its extrema identified.

3) A true positive (TP) is recorded if an extrema is present at the same location in the filtered versions of both the original signal and the corrupted signal. The True Negatives (TN), False Positives (FP) and False Negatives (FN) are calculated in an analogous manner. The error rate is then calculated using the formula:

$$ErrorRate = (FP + FN)/(TP + TN + FP + FN) \tag{5.50}$$

4) This procedure is iterated for thirty times, in order to obtain a statistical average of the performance for that particular noise level.

5) Finally, the average error rate at different noise levels are used to create the error rate curves that are shown in Figure 5.7 to Figure 5.11.

Table 5.1: Filter Definitions

Filter	Definition
Gaussian	$h[i] = \exp\left(-(\frac{i}{\sqrt{2}std\_dev})^2\right), i \in [-N, N]$
Der-Gauss	$h[i] = i * \exp\left(-(\frac{i}{\sqrt{2}std\_dev})^2\right), i \in [-N, N]$
Sombrero	$h[i] = (1 - (\frac{i}{std\_dev})^2) * \exp\left(-(\frac{i}{\sqrt{2}std\_dev})^2\right),$ $i \in [-N, N]$
Sine	$h[i] = sin(\pi * i/N), i \in [-N, N]$
Const	$h[i] = 1, i \in [-N, N]$
Delta	$h[0] = 1; h[i] = 0, i \in [-N, 0) \cup (0, N]$

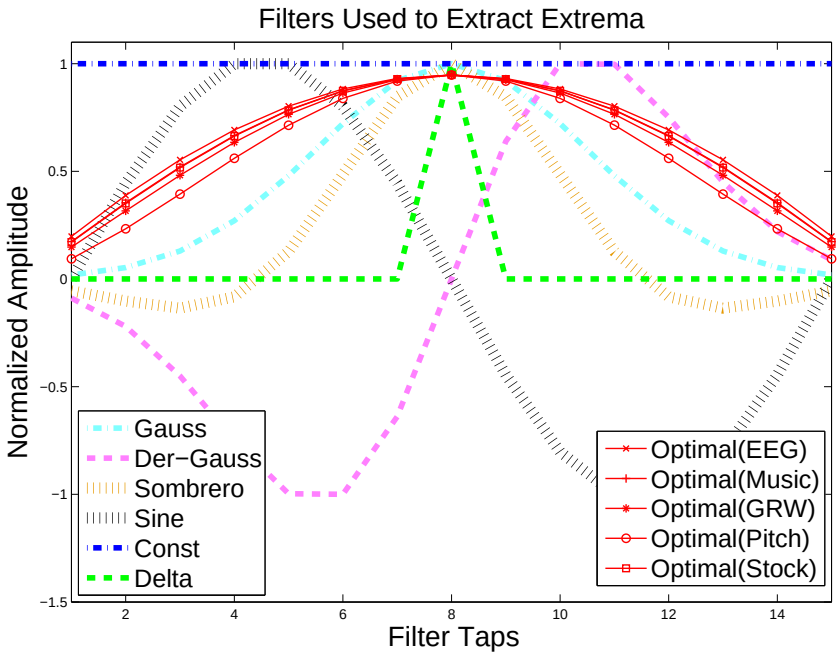

Figure 5.6: The filters used in the experimental tests.

The results presented in Figures 5.7 - 5.11 for each of the 5 datasets reveal

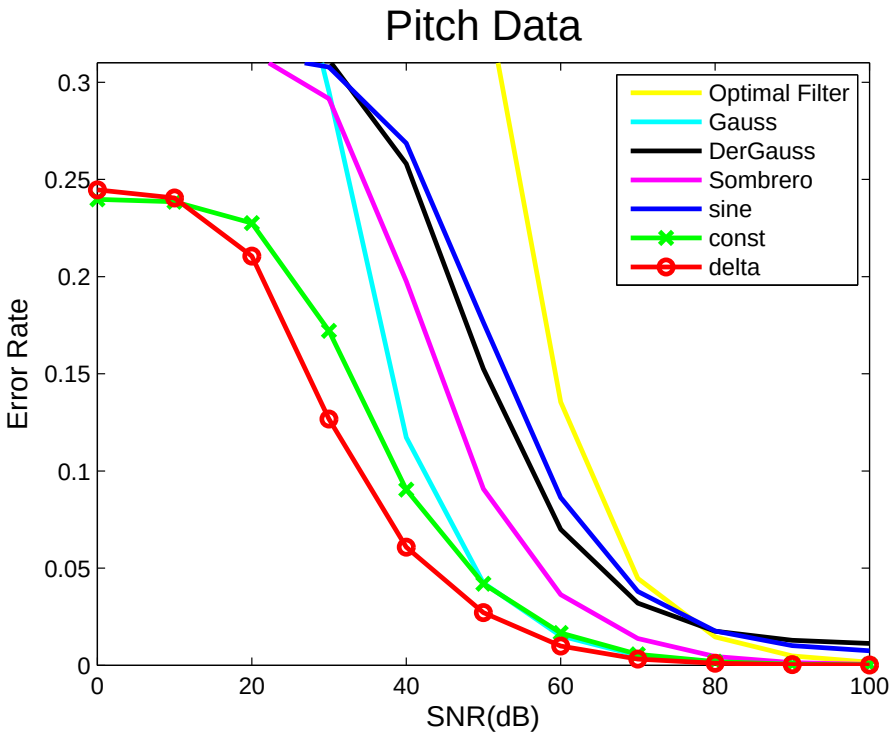

Figure 5.7: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

that the optimal filter outperforms all the other filters in terms of its ability to extract robust extrema which are resilient to noise. Figures 5.7 - 5.11 also show that the closest competitor for the optimal filter for all the datasets is the Gaussian filter. To investigate this further, we separately compare the extrema stability as provided by the Gaussian filter vs. the one optimized using training time-series. As the standard deviation of the Gaussian is critically tied to its performance, it is of interest to obtain the accuracy performance (or equivalently error rates) as a function of standard deviation as well. Plotting the accuracy vs. both noise level (or SNR) and the standard deviation parameter gives rise to surface plots as shown in Figures 5.12 - 5.16. It is observed that our proposed filter optimization procedure

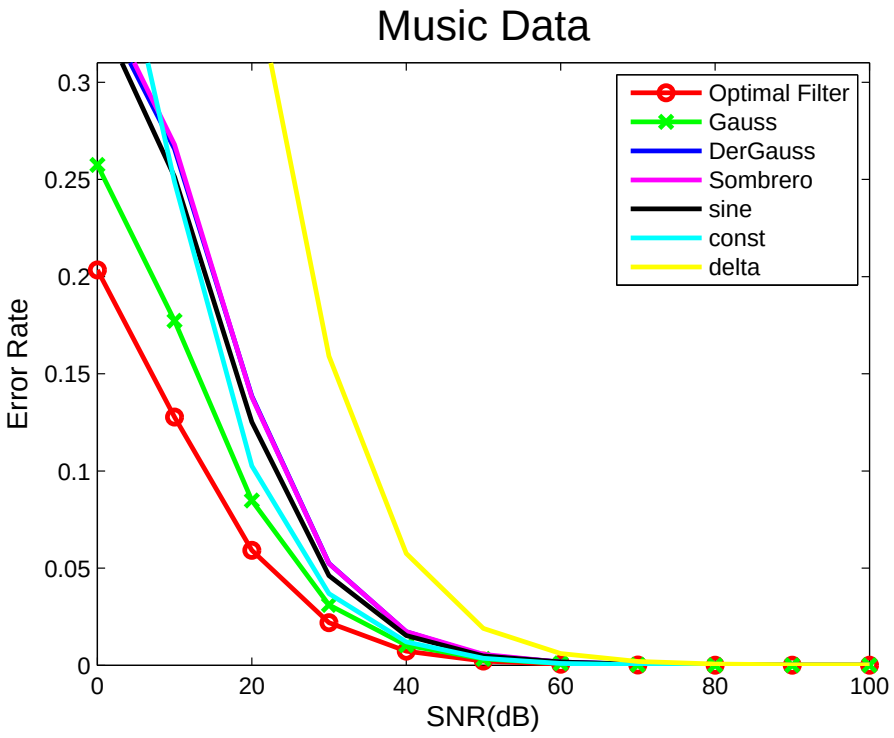

Figure 5.8: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

leads to greater accuracy in the low to medium noise level (or equivalently high SNR) scenarios - which are in fact representative of real-world corruption of time series. As Figures 5.12 - 5.16 reveal, Gaussian filters with some particular choices of the standard deviation parameter do mildly better than the optimized filter in the high noise regime, and for particular data sets (pitch data, stock data and Gaussian Randomwalk data). This is because the optimal filter has been optimized to maximize the distance of data points to partitioning hyperplanes and this does not guarantee that all points exist beyond an epsilon bound from the partition. For certain noise types (ex: i.i.d Gaussian noise) the epsilon bound may be the more appropriate measure for robustness but a closed form solution for

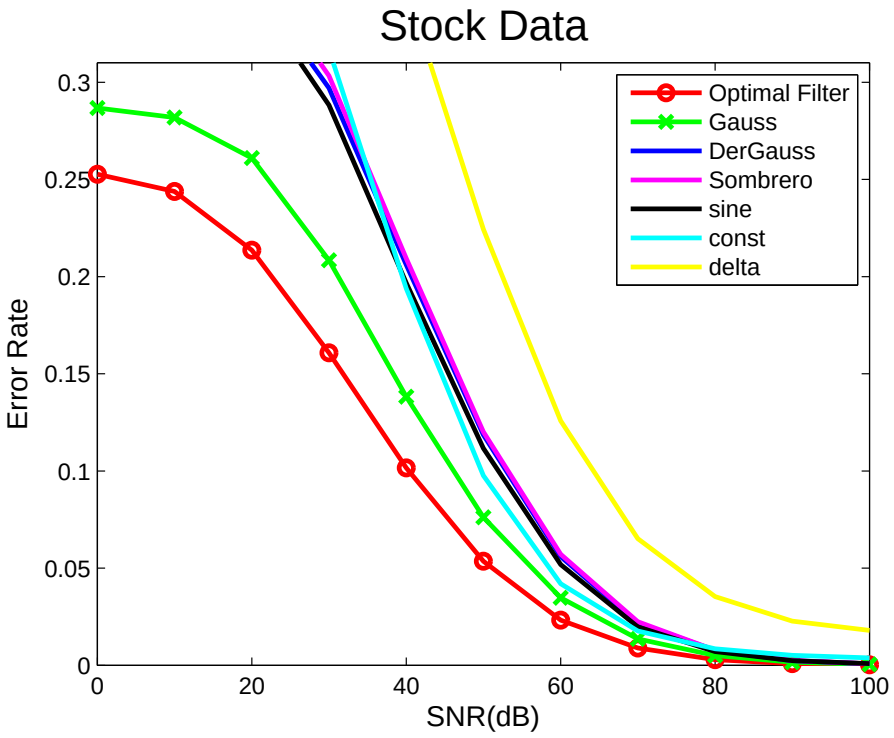

Figure 5.9: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

such a formulation is not evident. However, the definition of robustness based on perpendicular distance, is intuitive, provides a closed form solution for the optimal filter and gives the best results in a number of situations as shown in Figures 5.12 - 5.16.

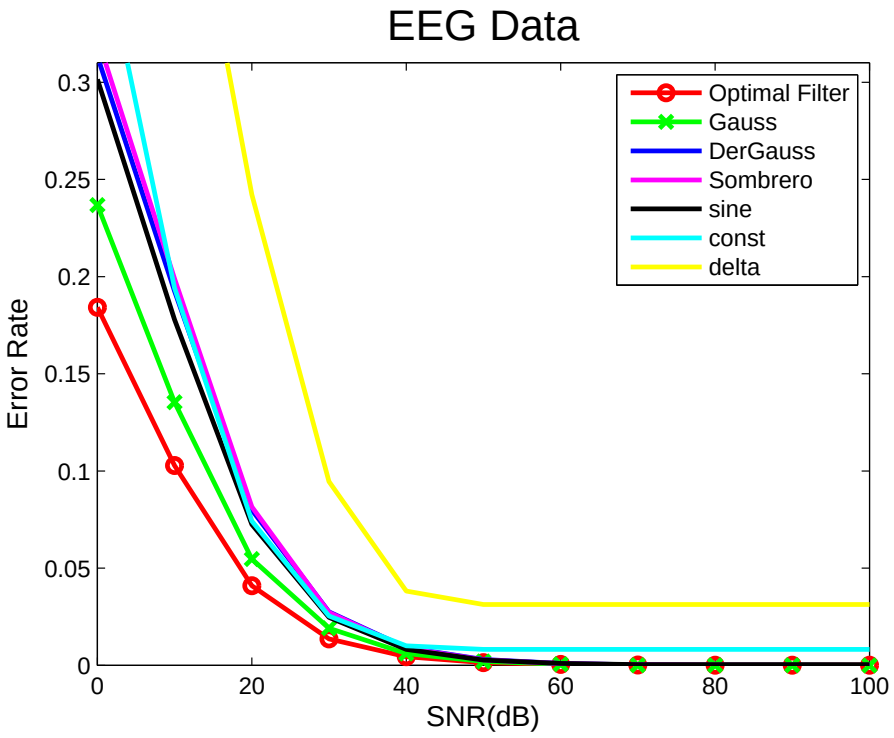

Figure 5.10: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

5.4.3 Results B: Application to the subsequence matching problem

5.4.3.1 Brief review of time series subsequence matching

Time series subsequence matching is a key research area in a large number of disciplines including mechanical, electrical, financial and medical domains. Improvements in sensor technology and data acquisition systems have lead to an exponential increase in the amount of signal data that is acquired in different disciplines. Given a new time series signal, a common problem that is often encountered is that of finding similar signals from within an existing time series repository. Specifi-

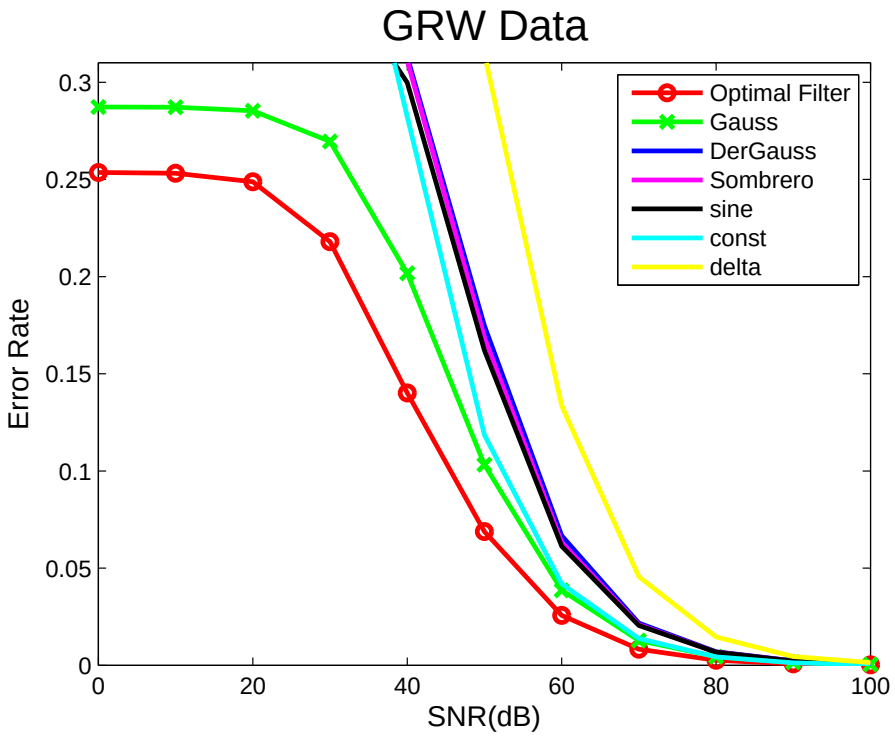

Figure 5.11: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

cally, the problem could be posed as finding 'k' time series in a dataset, which are most similar to a given query time series. This fundamental question was posed by Faloutsos et al in their seminal paper [4] and the proposed solution utilized the Euclidean distance measure of similarity while operating within a sliding window framework.

Given the fundamental nature of this problem, a wide variety of different techniques have been proposed in literature [4, 19, 90, 7]. The techniques themselves can be broken down into exact and inexact methods. The exact methods can guarantee retrieval of all subsequences in a database which are within a certain threshold distance from the given query sequence. Typically, these methods utilize

the sliding window framework for accomplishing this task [4]. On the other hand, the inexact matches do not provide such a guarantee but they usually provide advantages in terms of computational and memory resources needed to implement the system. These methods could also deliver advantages in terms of their ability to handle complex distortions in the query signal. The extrema methods are inexact methods which offer certain computational advantages and also provide robustness against different distortions [7]. The framework proposed in Section **??** is utilized to build an 'inexact' extrema based subsequence matching method.

5.4.3.2 Extrema based subsequence matching method

Like all the common subsequence matching methods, extrema methods (exact and inexact methods) also have two distinct phases in their implementations: 1) Indexing phase, and 2) Online phase.

In the indexing phase, feature vectors are extracted by using the extrema obtained from time series data and these features are used to build a data structure that enables quick retrieval and matching. Any of the wide range of different retrieval techniques that are available [41] can be used for this application and a KD-tree was used in this particular implementation. The choice of a particular data structure will not affect the final accuracy result of the subsequence matching method and is only expected to change the speed at which the matching task is performed.

Differences from and benefits over sliding window techniques: The online phase for the extrema method, as proposed here, is very different from the traditional sliding window techniques. The traditional sliding window methods employ a two step procedure in the online phase. The first step involves retrieval of all subsequences which satisfy a lower bounding distance criterion and the second step consists of post processing the retrieved subsequences to obtain the closest match. In the proposed extrema method, a one step process is followed. After extracting the features from a signal and matching them against the data structure, each feature match results in different estimates for the position of the query time series in the database. These position hypotheses from all the feature matches are assembled into a histogram and the locations with a large number of votes are considered to be good matches.

Unlike the traditional sliding window techniques, no major post processing is required to remove the false positives after the initial index based search and so this reduces the amount of computation that is involved. Also, as no further post processing is required one need not store the original data and this reduces the memory requirements for the extrema algorithm. The piecemeal nature of the feature matching technique allows the method to handle query signals of any length, unlike the sliding window methods which are restricted by their window size.

As a counter balance to the advantages associated with lower computation,

lower memory requirements, increased robustness [7] and the ability to handle to query signals of different lengths while using extrema, the main drawback of using extrema based approach is that it does not utilize the uniqueness of the query signal to the maximum possible extent as done by the sliding window based methods. This is because this method performs matching in a piecemeal fashion while the sliding window method can perform matching using the entire query signal at once. This trade off will have implications (both positive and negative) for the extrema based subsequence matching approach depending on the dataset and the nature of distortions at hand. Also, as no post processing is performed, the results from this method of subsequence matching depend solely on the ability of the features to obtain a proper match and can thus be a useful comparison tool for testing extrema based feature encoding techniques. This in turn implies that the feature vector must have maximum robustness towards the nature of distortion that the query signal may experience and maximum uniqueness so as to be able to obtain a proper match. Also the extrema based methods normalize themselves based on the bias and scale at the locations of extrema and this may give added robustness in case the bias or scale factor change along the signal.

5.4.3.3 Experimental Process

The purpose of this section is to compare the subsequence matching results of traditional linear-filter based extrema methods and sliding window based methods

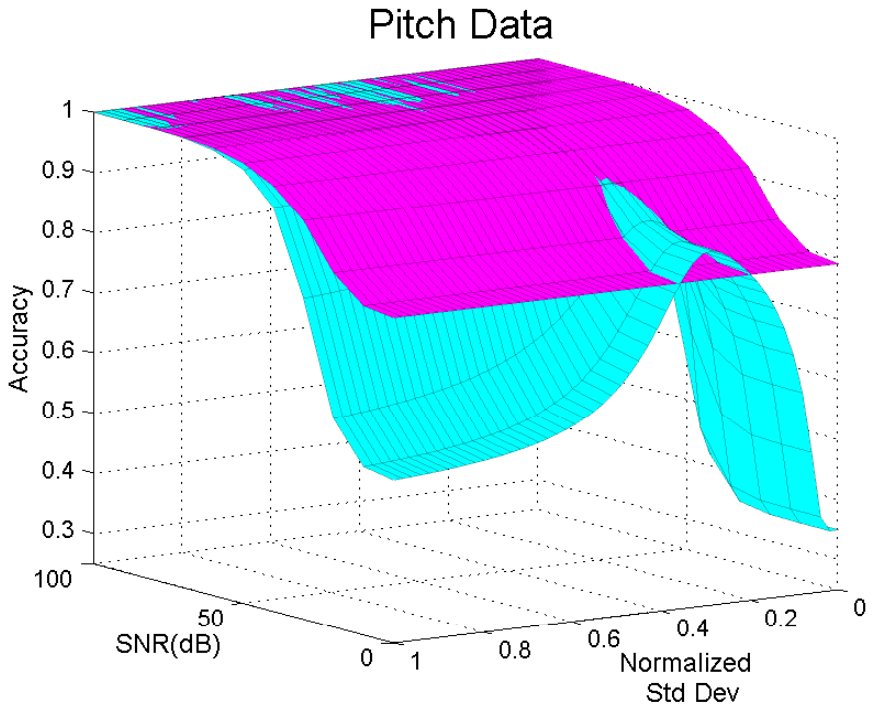

Figure 5.12: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

with the extrema method based on the framework presented in section 4. The experiments are performed on real and simulated data that was described in section 5.4.1.1.

A wide variety of solutions utilizing sliding window methods for dimension reduction or using different types of distance measures have been proposed [4, 19, 9] for subsequence matching. However, extrema based methods offer certain advantages while solving the subsequence matching problem because of their ability to handle complex distortions [7]. For example, scale factor noise, bias noise and outliers are commonly encountered distortions in time series data. However, common index-based subsequence matching methods based on euclidean distance [4] and

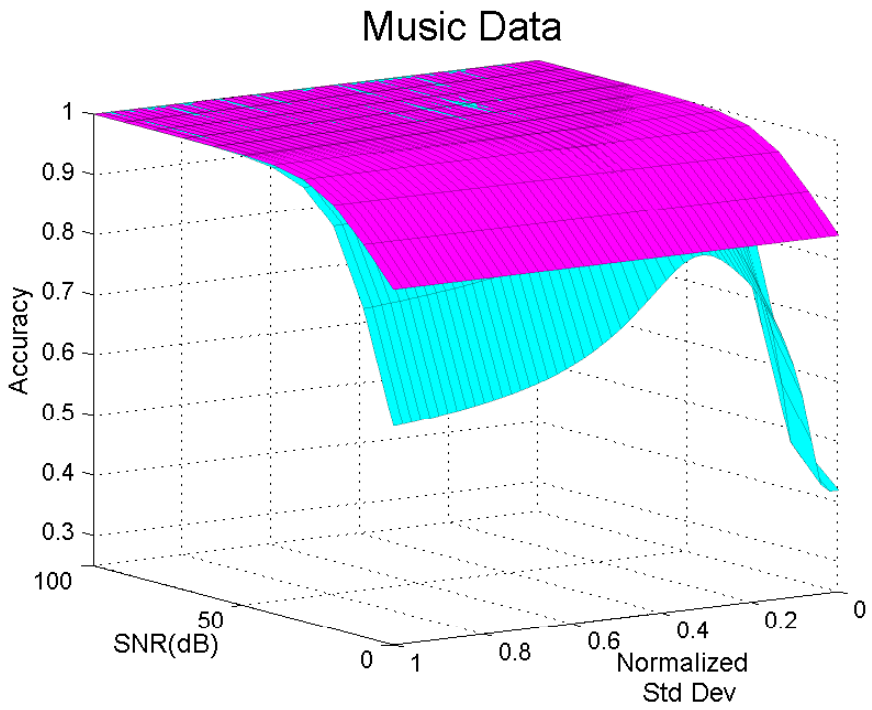

Figure 5.13: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

DTW [19] cannot handle outliers while certain edit distance based methods like LCSS [9] are not designed to handle scale and outlier noise simultaneously [43].

We next compare the proposed "optimally robust extrema features" against state of the art alternatives for the problem of time-series subsequence matching. In these experiments, we particularly look to examine how each method fares in retrieving similar time-series even as the time-series are observed under practically significant distortions. The entire experimental process is as follows:

1) 31 query signals of length 256 are extracted from a time series of 65536 points obtained from the datasets in section 5.4.1.1 and are corrupted with bias noise, scale factor noise, outliers, and Gaussian noise before being tested for subsequence

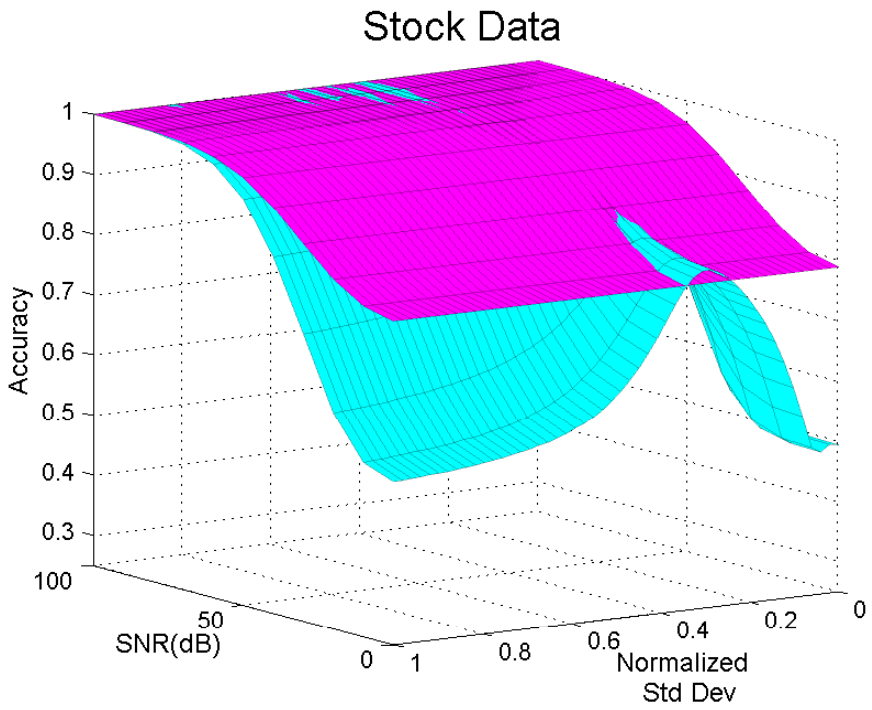

Figure 5.14: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

matching. The bias noise for a query signal was obtained by randomly selecting a value from a uniform distribution in the following interval [-std dev(query signal),+std dev(query signal)]. Similarly, the scale factor noise was obtained by randomly selecting a value from a uniform distribution in the following interval [0.33,3]. The outlier is simulated as a Gaussian random noise with the same mean and standard deviation as the query signal and is multiplied by a large scale factor (5). The query signal is also corrupted with a small amount of Gaussian noise (approximately 14dB SNR) that is often encountered in time series data in order to make the matching process more realistic. Figure 5.17 provides an example of a query signal that is obtained by applying the above distortions to a given

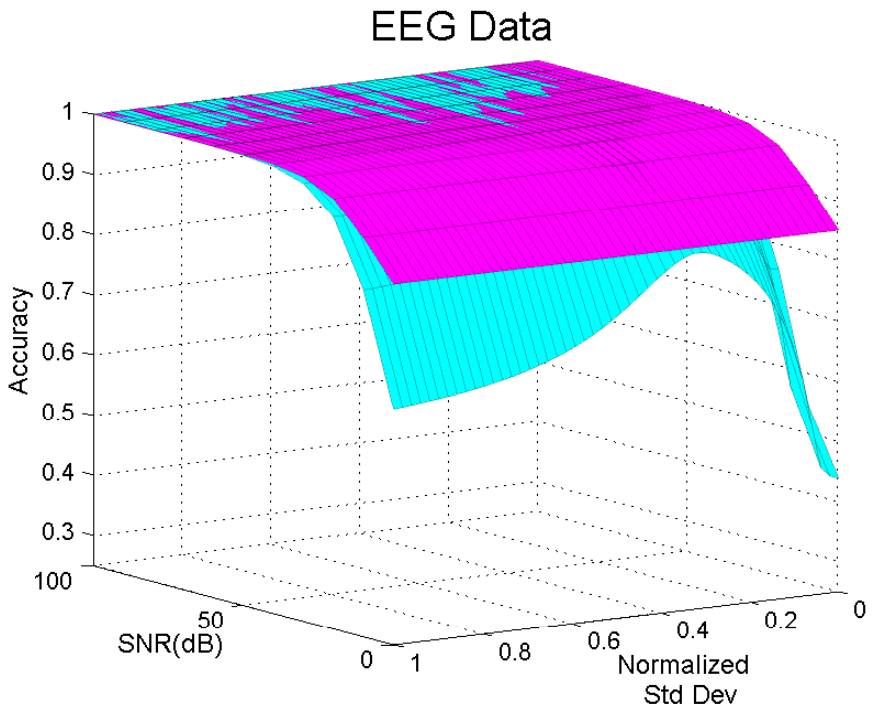

Figure 5.15: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

time series. Similarities between the two time-series in Figure 5.17 can be visually ascertained but the challenges in quantitatively enabling robust comparisons are readily apparent.

2) The spatial length of the outlier is varied from 0% to 20% of the length of the query signal and tested. The whole process is repeated thirty times in order to get a statistical average of the accuracy.

3) The accuracy results are presented in Figures 5.18 - 5.22 for different datasets when using different types of subsequence matching methods.

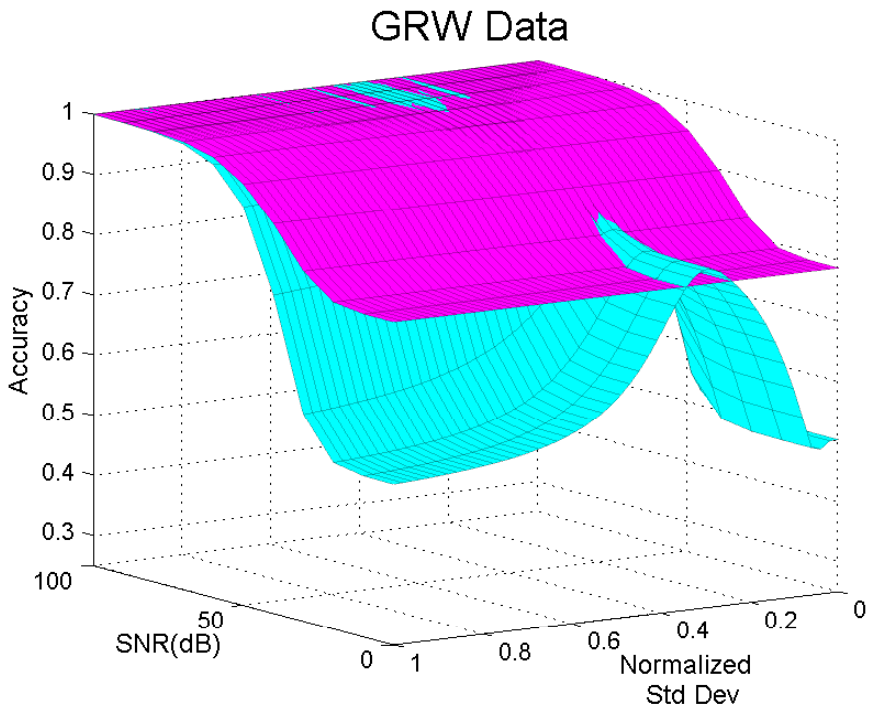

Figure 5.16: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

5.4.3.4 Implementation of well-known sliding window methods for time-series subsequence matching

Given a query signal that is corrupted in the aforementioned manner, the sliding window methods are implemented in the following manner. A sliding window is used to extract each subsequence of 256 data points from the time series and its distance is computed with respect to the given query signal. The subsequence which gave the least distance was considered the best match and if the position of this subsequence was within a threshold distance (+/- 25 data points = 10% of query signal length) of the point of extraction of the query signal then the match was deemed accurate. The sliding windows methods were *deliberately* implemented

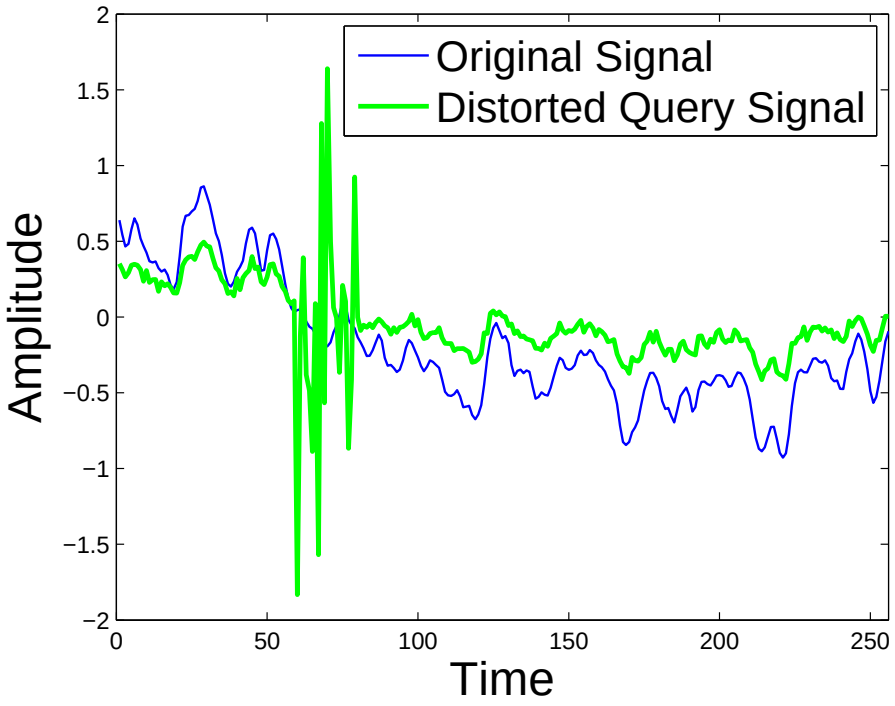

Figure 5.17: The distorted query signal is obtained by adding bias, scale factor, outlier and Gaussian noise to the original signal.

without indexing[1] to allow the implementation of the above methods in a form that is closest to the original intent of the algorithm's authors. The euclidean distance and dynamic time warping methods do not have any explicit parameters and were implemented in a standard manner. The LCSS distance, however, has a parameter corresponding to the amplitude change and another parameter corresponding to temporal distortion. The amplitude parameter was chosen as the minimum of the standard deviations of the two matching signals and the temporal parameter was chosen as 20% of the length of the query signal as suggested in the paper [9].

[1]It is worth re-emphasizing that the lack of indexing does not influence the sliding-window techniques from an accuracy of retrieval standpoint which is the focus of this chapter. State of the art indexing techniques for these techniques are of course well researched, examples may be found in [4] [19] [90].

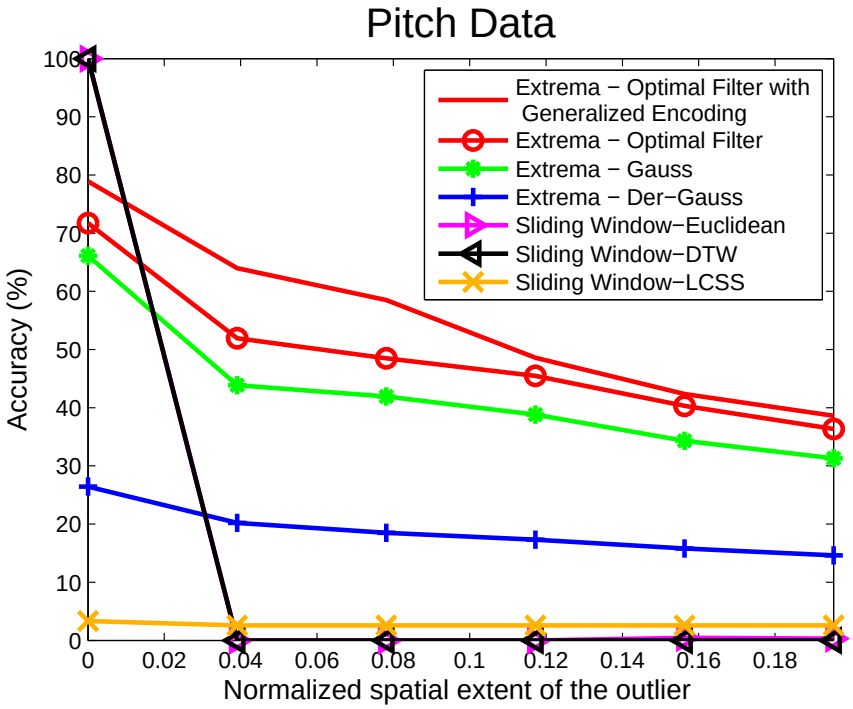

Figure 5.18: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

Also, as the implementation of LCSS corresponded to the D1 distance mentioned in paper [9] rather than the more complicated D2 distance which is expected to incorporate bias invariance, the query signal for the LCSS method was not corrupted with bias noise inorder to avoid disadvantaging the LCSS method on this count.

5.4.3.5 Implementation of extrema methods for time-series subsequence matching

The first step (Indexing phase) in the extrema method is to generate features from the time series as described in section 5.4.3.2. In the feature generation

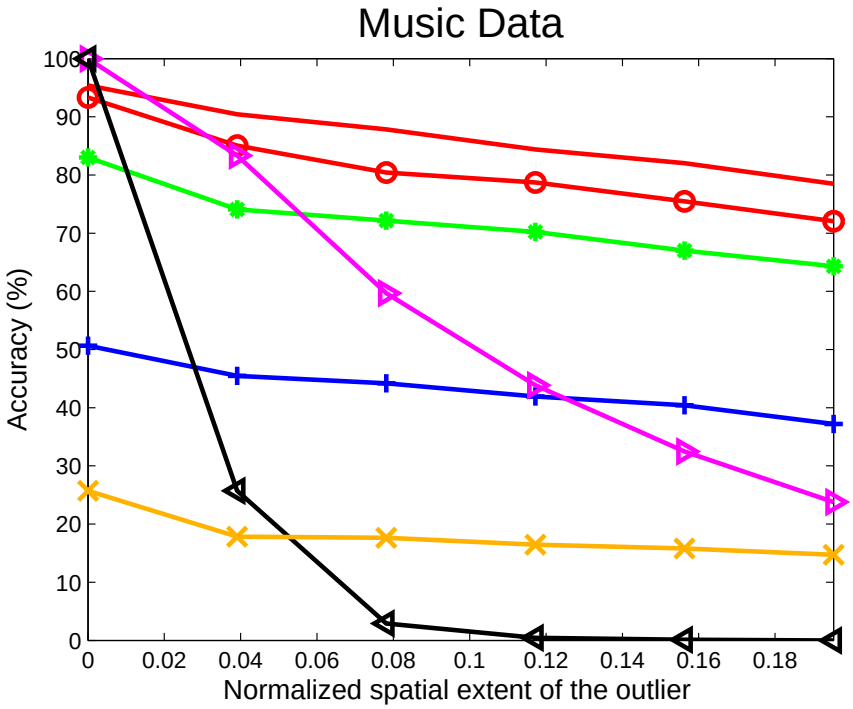

Figure 5.19: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

process, the threshold in the extrema detection step described in section 5.3.2 is adjusted such that the number of extrema from each filter are approximately the same. This process ensures that the computational effort required for the different filtering techniques is also identical. The features are then stored in a data structure (KD-tree) as described before. In the online phase, the features generated from the query signals are matched with the features present in the KD-tree. The matches in the data structure yield different estimates of the location from which the query signal was extracted. These locations are put into a histogram to obtain the most agreed upon location estimate for the time series subsequence as described in section 5.4.3.2. If the location estimate from the matching process

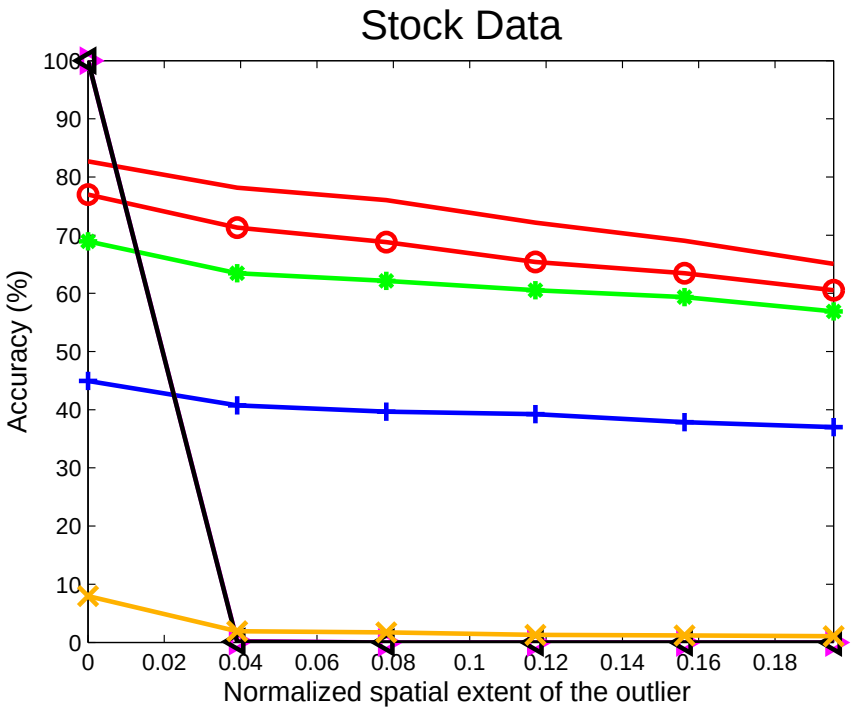

Figure 5.20: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

was within a certain threshold distance (+/- 25 data points) from the true location of extraction then the result is deemed accurate. In the above scheme, the filters described in Figure 5.6 are used to generate extrema which are encoded using the sequential encoding method which results in bias and scale invariance , as described in Figure 5.4, as these distortions are expected in the query signal. The subsequence matching method is also performed by encoding features using the generalized encoding technique described in section 5.3.3 while using the extrema obtained from the optimal filter.

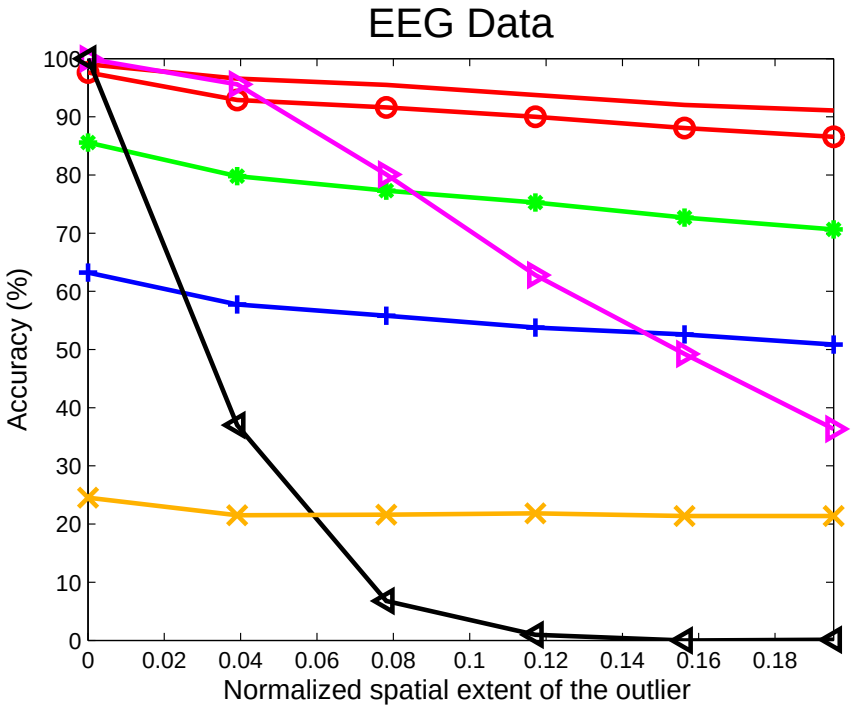

Figure 5.21: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

5.4.3.6 Results and Inferences

Accuracy Benefits in Robust Retrieval of Time-Series: Figures 5.18 - 5.22 shows the accuracy plots for time series subsequence matching when performed using the above methods. In particular, accuracy is plotted with respect to the size of the outlier relative to the size of the query signal (normalized spatial extent of the outlier). The accuracy of all the methods is expected to decrease as the normalized spatial extent of the outlier increases.

The results clearly show that the extrema methods outperform sliding window methods because of their ability to handle complex distortions. Also, the piecemeal nature of the feature matching technique leads to some features which are totally

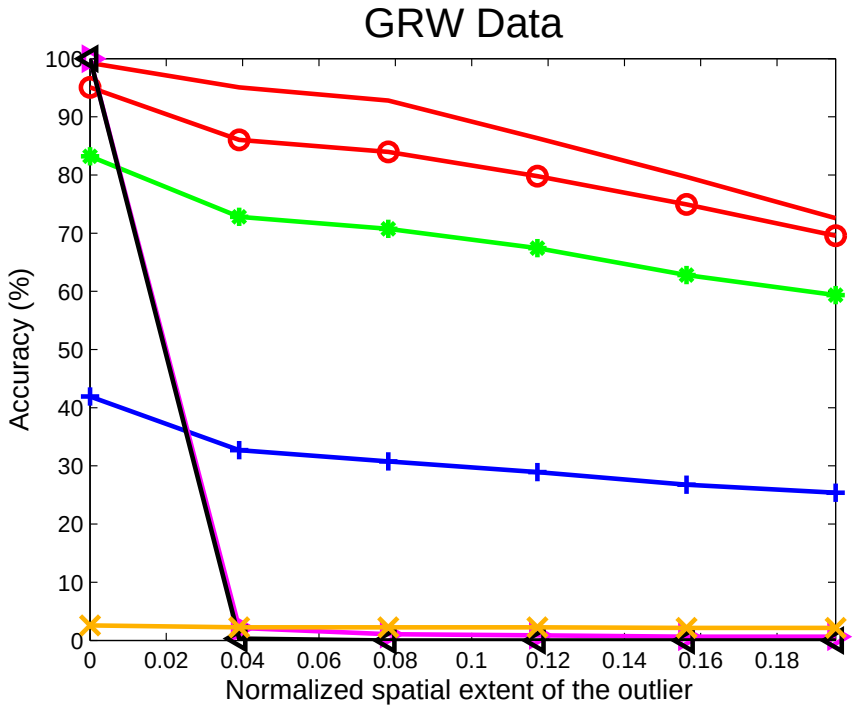

Figure 5.22: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

unaffected by outlier data. These features lead to a correct match in spite of the incorrect cues from the features which are affected by the presence of the outlier. It can also be seen that the generalized encoding based optimal extrema method outperforms all the extrema methods as it is able to favorably increase the uniqueness of the extrema based feature representation. The inability of the euclidean and DTW methods to handle outliers and the inability of the LCSS method to handle scale factor noise severely affects their performance.

In summation, Euclidean, DTW and LCSS methods perform excellently in the presence of Gaussian noise, temporal distortion, and outliers respectively as noted in earlier literature. However, in the presence of complex distortions, such as the

simultaneous presence of bias, scale, and outliers, use of extrema methods leads to improved matching accuracy and the use of the optimally robust extrema as proposed in this chapter leads to further accuracy improvements. It is important to note that the optimal filter need not deliver the best subsequence matching result and it may be possible to find other linear/non-linear filters that deliver better performance. This is mainly because the optimal filter has been obtained by performing optimization over robustness and this may not always translate into a better subsequence matching result. It is also important to understand that their might be other distortions in which the other methods could potentially outperform the extrema based methods. For example, situations in which the query signal is corrupted by just Gaussian random noise, or just temporal distortion or just outliers will result in the Euclidean, DTW and the LCSS method giving the best performance. However, the extrema based methods can be useful in situations such as the present where one encounters severe bias , scale factor and outlier noise simultaneously.

Economy of Representation enabled by the use of Extrema Features:

The number of features utilized by each matching method is shown in Figure 5.23 and it can be seen that the extrema methods use up to 10 times fewer number of features while delivering improved performance in this subsequence task. It should be noted from Figure 5.23 that the generalized encoding technique necessarily incurs a higher feature dimensionality its sequential encoding counterpart,

albeit this increase is mined into performance (accuracy) improvements as seen in Figures 5.18 - 5.22.

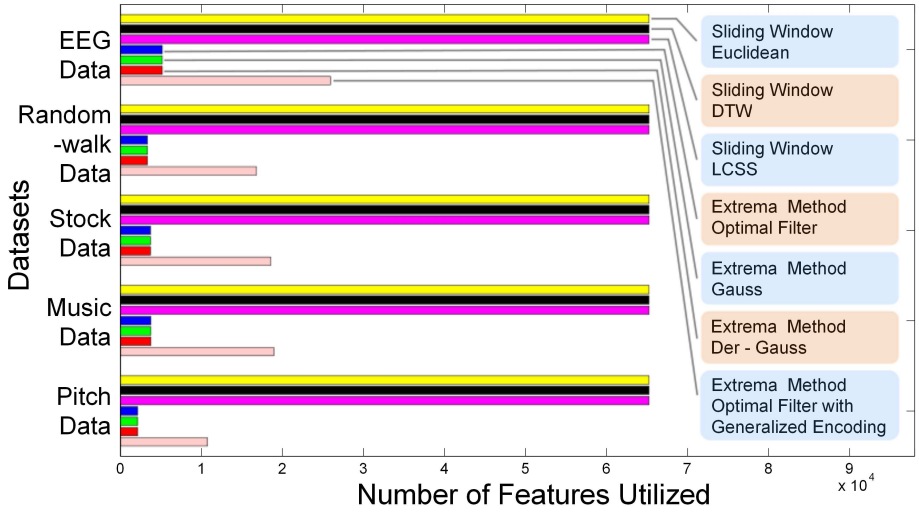

Figure 5.23: The number of features used by the different techniques for the subsequence matching results presented in Figures 5.18 - 5.22. The number of features can be used to gauge the computation involved with each method.

5.5 Conclusions

We consider the problem of extracting adaptive and robust extrema features from time-series data. We enable the extraction of robust extrema by formulating a novel optimization problem which optimizes the coefficients of a pre-processing linear filter that is applied to the raw time-series data prior to extrema detection. This formalizes the intuition shared by a variety of existing methods in the literature which use intuitively motivated smoothing filters. Remarkably, the optimization reduces to a generalized eigenvalue problem and therefore admits a

simple, tractable solution. Our second contribution is a new generalized method for encoding extrema features that can enable superior trade-offs between the properties of robustness, uniqueness and cardinality and help customize/enhance extrema based feature extraction for different pattern recognition tasks. We evaluate the proposed framework by comparing it against the state-of-the art for robust retrieval of time-series also known as subsequence matching in time series databases. Experimental results on real as well as simulated time-series data reveal significant practical benefits in accuracy of retrieval.

Chapter 6

Discrete Sine Transform for Extracting Extrema From Random Walk Data

6.1 Introduction

In the previous chapter, a filter optimization procedure that enables extraction of robust extrema was presented. The entire derivation and the final results showing its application to various datasets was performed in a deterministic sense. This implies that we were able to obtain 'a particular optimal filter' by performing the optimization on 'a particular dataset'. Interestingly, the optimal filters from the different datasets have a rather similar shape as shown in Figure 5.6. The main aim of this chapter is to solve the above problem in a 'stochastic sense' and this

will lead to interesting theoretical results that will provide an intuitive explanation about the similarity in shapes that was observed amongst the different filters shown in Figure 5.6.

The derivation presented in this chapter provides a closed form solution to the optimal filter for a Gaussian random walk time series. It is important to gain an understanding about the usefulness of such a derivation. The stochastic equations that describe a process provide general rules that describe its evolution as a function of time. Because of the randomness of the process equations, a stochastic process can lead to a large number of actual instantiations of the same process. In order to arrive a particular instantiation, one would not only require the stochastic rules or equations at hand but would also need enough extra information that would specify that particular instantiation in a definite or 'deterministic' manner (E.g the seed of the random number generator). The rules that govern a certain stochastic process are usually detailed enough so that a number of global statistical parameters (such as mean, variance etc) could be derived from such rules. In fact, in case of certain special stochastic processes (such as random walks) the stochastic rules can be used to easily calculate all the necessary statistics that are required to obtain the optimal filter described in the previous chapter. The ease of calculation is due to the properties of independence and stationarity of the increments of the random walk process which simplify the calculation of the cross correlation terms used in the computation of the optimal filter. It would be useful to obtain closed-

form solutions for the 'optimal filter' for various commonly used random walk models as any potential time series that is generated from that process can use the same filter for extrema extraction. Even if a certain signal can approximately be modeled as a random walk (Ex: Stock data as a Gaussian Random Walk [91]), then the above closed-form solution would still be useful as it is quite likely that it would be close to the "real optimal filter" that could have been obtained from performing the optimization on the original dataset.

6.2 Optimal Filter for the Case of Gaussian Random Walk Data (Discretely Sampled Weiner Process)

6.2.1 Setting up the matrix for eigen analysis

From equation (5.43) one can see that, if a dataset is given, the eigen vector corresponding to the maximum eigen value of the $[I + J_{2N+1}]^{-1}X_{Data}$ matrix can be used to obtain the optimal extrema filter for that data. In this section, it is assumed that the input data is a Gaussian random walk and a theoretical derivation for the optimal extrema filter is presented. Though the Gaussian random walk is a special case of a random walk process, the implications of this derivation will be generalized to all random walk processes in the next section. This derivation

is important because a number of processes in the real world are modeled as a random walk. Given the broad importance of random walk data in time series analysis, it is useful to have a closed-form solution for the final filter in order to avoid the computation. Starting with the assumption that the data is a Gaussian random walk, the following procedure derives the optimal filter with '2N+1' taps. Let $[x^j_{-N-1}\ x^j_{-N}\ x^j_{-N+1}\ ...\ x^j_0\ ...\ x^j_{N+1}\ ]$ be a $'2N+3'$ long time series extracted from the j^{th} position of the original dataset.

From equation (5.33)

$$X_{Data} = \sum_{j=N+1}^{M} [y_j y_j^T + z_j z_j^T] \tag{6.1}$$

Where y_j denotes a $2N+1$ vector whose i^{th} element is denoted by $(x_{-N-1+i+j} - x_{N+1+j})$ and z_j denotes a $2N+1$ vector whose i^{th} element is denoted by $(x_{-N-1+i+j-1} - x_{N+j})$. The element on the α^{th} row and β^{th} column of the $y_j y_j^T$ matrix is given by

$$y_j y_j^T(\alpha, \beta) = \sum_{j=N+1}^{M} (x_{\alpha+j-N-1} - x_{j+N+1}) * (x_{\beta+j-N-1} - x_{j+N+1}) \tag{6.2}$$

Given a particular time series, x_n, one could express it in increments w_n where

$$x_{n+1} = x_n + w_n \tag{6.3}$$

In this particular case, as the time series is from a Gaussian random walk the

increments w_n are from a normal distribution with zero mean. The terms in equation (6.2) can now be expressed in terms of w_n as shown in the below equations.

$$\sum_{k=\alpha}^{2N+1}(w_{k+j-(N+1)}) = x_{\alpha+j-N-1} - x_{j+N+1} \tag{6.4}$$

$$\sum_{k=\beta}^{2N+1}(w_{k+j-(N+1)}) = x_{\beta+j-N-1} - x_{j+N+1} \tag{6.5}$$

Substituting equations (6.4) and (6.5) into equation (6.2) we obtain

$$y_j y_j^T(\alpha,\beta) = \sum_{j=N+1}^{M}((\sum_{k=\alpha}^{2N+1} w_{k+j-(N+1)}) * (\sum_{k=\beta}^{2N+1} w_{k+j-(N+1)})) \tag{6.6}$$

$$y_j y_j^T(\alpha,\beta) = \sum_{j=N+1}^{M}((\sum_{k=max(\alpha,\beta)}^{2N+1}(w_{k+j-(N+1)})^2)+(\sum_{k=\alpha}^{2N+1}\sum_{i=\beta,k\neq i}^{2N+1} w_{k+j-(N+1)}w_{i+j-(N+1)})) \tag{6.7}$$

$$\begin{aligned} y_j y_j^T(\alpha,\beta) = &\sum_{k=max(\alpha,\beta)}^{2N+1}\sum_{j=N+1}^{M}(w_{k+j-(N+1)})^2+ \\ &\sum_{k=\alpha}^{2N+1}\sum_{i=\beta,k\neq i}^{2N+1}\sum_{j=N+1}^{M} w_{k+j-(N+1)} * w_{i+j-(N+1)} \end{aligned} \tag{6.8}$$

$$\begin{aligned} y_j y_j^T(\alpha,\beta) = &((2N+1)-(max(\alpha,\beta)-1)) * (M-N) * E(w^2)+ \\ &\sum_{k=\alpha}^{2N+1}\sum_{i=\beta,k\neq i}^{2N+1}\Big((M-N) * E(w_{k+j-(N+1)} * w_{i+j-(N+1)})\Big) \end{aligned} \tag{6.9}$$

Given the information that the increments w_i are from a standard normal distribution and are independent, the quantity $E(w_{k+j-(N+1)}w_{i+j-(N+1)})$ is zero.

The next section elaborates on the situation in which $E(w_{k+j-(N+1)}w_{i+j-(N+1)})$ is a constant, which holds for all random walk processes.

$$y_j y_j^T(\alpha,\beta) = ((2N+1) - (max(\alpha,\beta)-1)) * (M-N) * E(w^2) \tag{6.10}$$

One can derive a similar equation for $z_j z_j^T$ and it results in an identical form. This finally leads to the following form for the X_{data} matrix. The constant K in equation (6.11) denotes $2*(M-N)*E(w^2)$ and is obtained by combining the contributions of $y_j y_j^T$ and $z_j z_j^T$. The constant will not be shown in further equations as it's only effect would be to scale the final result by K times.

$$X_{data} = K \begin{bmatrix} 2N+1 & 2N & 2N-1 & ... & 2 & 1 \\ 2N & 2N & 2N-1 & ... & 2 & 1 \\ 2N-1 & 2N-1 & 2N-1 & ... & 2 & 1 \\ ... & ... & ... & ... & ... & ... \\ 2 & 2 & 2 & ... & 2 & 1 \\ 1 & 1 & 1 & ... & 1 & 1 \end{bmatrix} \tag{6.11}$$

Given the above form for the X_{data} matrix, the aim of this derivation is to find the eigen vector corresponding to the maximum eigen value of the $[I + J_{2N+1}]^{-1}X_{data}$ matrix as shown in Section 5.3.1.3. For a given filter of length $'2N+1'$, the

$[I + J_{2N+1}]^{-1}$ matrix is given by

$$[I + J_{2N+1}]^{-1} = 1/(2N+2) \begin{bmatrix} 2N+1 & -1 & ... & -1 & -1 \\ -1 & 2N+1 & ... & -1 & -1 \\ ... & ... & ... & ... & ... \\ -1 & -1 & ... & 2N+1 & -1 \\ -1 & -1 & ... & -1 & 2N+1 \end{bmatrix} \tag{6.12}$$

Let

$$F = [I + J_{2N+1}]^{-1} * X_{data} \tag{6.13}$$

By performing straightforward algebraic manipulation one can calculate $F(\gamma, \delta)$ where γ and δ denote the row and column locations.

$$F(\gamma,\delta) = \begin{cases} \dfrac{(2N+2-\delta)*(2N+3-\delta)}{2}, & when\ \gamma \leq \delta \\ (2N+2)*(2N+2-\gamma) - \dfrac{(2N+2-\delta)*(2N+1+\delta)}{2}, & when\ \gamma \geq \delta \end{cases} \tag{6.14}$$

Therefore, the F matrix has the form

$$F = \begin{bmatrix} \frac{(2N+1)*(2N+2)}{2} & \frac{(2N)*(2N+1)}{2} & \frac{(2N-1)*(2N)}{2} & ... & 1 \\ (2N+2)*(2N) - \frac{(2N+1)*(2N+2)}{2} & \frac{(2N)*(2N+1)}{2} & \frac{(2N-1)*(2N)}{2} & ... & 1 \\ (2N+2)*(2N-1) - \frac{(2N+1)*(2N+2)}{2} & ... & \frac{(2N-1)*(2N)}{2} & ... & 1 \\ ... & ... & ... & ... & ... \\ ... & ... & ... & ... & ... \\ (2N+2)*(1) - \frac{(2N+1)*(2N+2)}{2} & ... & ... & ... & 1 \end{bmatrix} \tag{6.15}$$

The definitions and the lemmas that are required for the proof relating to the eigen vectors of matrix F are given below. Given the F matrix, Lemma 1 can be gleaned from observation and Lemma 2 can be obtained from simple mathematical manipulation.

Lemma 6.2.1. $F(\gamma+1,\delta) = F(\gamma,\delta)$ *when* $\gamma < \delta$

Lemma 6.2.2. $F(\gamma+1,\delta) = F(\gamma,\delta) - (2N+2)$ *when* $\gamma \geq \delta$

Definition $F(i,:)$ denotes the i^{th}row of the F matrix

Definition $F(:,j)$ denotes the j^{th}column of the F matrix

Definition $F(i,j)$ denotes the element in the i^{th} and the j^{th} column of the F matrix

Definition w_m *and* λ_m represent the m^{th} eigen vector and the corresponding eigen value respectively

Theorem 6.2.3. *Given a square matrix of size* $'2N+1'$*, whose elements in row*(γ) *and the column*(δ) *are given by*

$$F(\gamma,\delta) = \begin{cases} \dfrac{(2N+2-\delta)*(2N+3-\delta)}{2}, & when\ \gamma < \delta \\ (2N+2)*(2N+2-\gamma) - \dfrac{(2N+2-\delta)*(2N+1+\delta)}{2}, & when\ \gamma \geq \delta \end{cases}$$

Then the eigen vectors of such a matrix are given by

$$w_m(a) = sin(m\pi\frac{a}{2N+2}) - sin(m\pi\frac{a-1}{2N+2})$$

where $w_m(a)$ *is the* a^{th} *element of the* m^{th} *eigen vector and* $a \in [1, 2N+1]$. *The corresponding eigen values are given by*

$$\lambda_m = \frac{n+1}{1 - cos(m\pi \frac{1}{2N+2})}$$

Proof: From the definition of the eigen value we know that

$$w_m(1) * \lambda_m = F(1,:) * w_m \tag{6.16}$$

Similarly,

$$w_m(2) * \lambda_m = F(2,:) * w_m \tag{6.17}$$

From Lemmas 6.2.1 and 6.2.2,

$$F(1,:) * w_m = F(2,:) * w_m + (2N+2) * w_m(1) \tag{6.18}$$

Substituting equations (6.16) and (6.17) into the above equation

$$w_m(1) * \lambda_m - w_m(2) * \lambda_m = (2N+2) * w_m(1) \tag{6.19}$$

Therefore,

$$\lambda_m = (2N+2) * \frac{w_m(1)}{w_m(1) - w_m(2)} \tag{6.20}$$

if the eigen vectors are given by $w_m(a) = sin(m * \pi * \frac{a}{2N+2}) - sin(m * \pi * \frac{a-1}{2N+2})$,

then the eigen values

$$\lambda_m = (2N+2) * \frac{sin(m*\pi*\frac{1}{2N+2})}{2*sin(m*\pi*\frac{1}{2N+2}) - sin(m*\pi*\frac{2}{2N+2})} \tag{6.21}$$

$$\lambda_m = \frac{(2N+2)*sin(m*\pi*\frac{1}{2N+2})}{2*sin(m*\pi*\frac{1}{2N+2}) - 2*sin(m*\pi*\frac{1}{2N+2})*cos(m*\pi*\frac{1}{2N+2})} \tag{6.22}$$

Therefore,

$$\lambda_m = \frac{N+1}{1-cos(m*\pi*\frac{1}{2N+2})} \tag{6.23}$$

In order to prove that $w_m(a)$ is the eigen vector for the F matrix, one must show that the eigen value obtained from each row, using equations similar to (6.16) and (6.17), must be the same as equation (6.23) where

$$w_m(a) = sin(m*\pi*\frac{a}{2N+2}) - sin(m*\pi*\frac{a-1}{2N+2}) \tag{6.24}$$

From the definition of equation (6.16), starting from the first row $F(1,:)$, it is required to show that

$$F(1,:) * w_m = w_m(1) * \lambda_m \tag{6.25}$$

$$\iff \sum_{\delta=1}^{2N+1} \left(\frac{(2N+2-\delta)*(2N+3-\delta)}{2} * \right.$$

$$\left(sin(m*\pi*\frac{\delta}{2N+2}) - sin(m*\pi*\frac{\delta-1}{2N+2})\right)\Big) = w_m(1)*\lambda_m \tag{6.26}$$

Using the following mathematical identity

$$\sum_{i=1}^{M}\left(\frac{(M+1-i)*(M+2-i)}{2}*(sin(b*i) - sin(b*(i-1)))\right) = \frac{sin(b(M+1)) - (M+1)*sin(b)}{2cos(b)-2} \tag{6.27}$$

Substituting $i = \delta$, $M = 2N+1$, and $b = m\pi/(2N+2)$ into the above equation

$$\sum_{\delta=1}^{2N+1}\left(\frac{(2N+2-\delta)*(2N+3-\delta)}{2}* \left(sin(m*\pi*\frac{\delta}{2N+2}) - sin(m*\pi*\frac{\delta-1}{2N+2})\right)\right) = \frac{sin(b(N+1)) - (N+1)*sin(b)}{2cos(b)-2} \tag{6.28}$$

The L.H.S of the above equation is same as that of equation (6.26)

$$R.H.S = \frac{sin(\pi) - (2N+2)*sin(m\pi/(2N+2))}{2cos(\pi/(2N+2))-2} \tag{6.29}$$

$$R.H.S = (N+1)\frac{sin(m\pi/(2N+2))}{1-cos(\frac{m\pi}{2N+2})} \tag{6.30}$$

$$R.H.S = \lambda_m * w_1(1) \tag{6.31}$$

From the above derivation we have shown that the eigen value given in equation

(6.23) satisfies the condition for the first row (6.25). For all the subsequent rows, the proof is obtained by mathematical induction. Suppose the equation $w_m(n) * \lambda_m = F(n,:) * w_m$ is true for row $'n'$ in the F-matrix, then we need to show that

$$w_m(n+1) * \lambda_m = F(n+1,:) * w_m \tag{6.32}$$

From Lemmas 6.2.1 and 6.2.2, we obtain

$$F(n,:) * w_m = F(n+1,:) * w_m + \sum_{i=1}^{n}(2n+2) * w_m(i) \tag{6.33}$$

Substituting equation (6.33) into equation (6.32), we need to show that

$$\lambda_m(w_m(n) - w_m(n+1)) = \sum_{i=1}^{n}(2n+2) * w_m(i) \tag{6.34}$$

Substituting $w_m(a) = sin(m\pi * \frac{a}{2N+2}) - sin(m\pi * \frac{a-1}{2N+2})$, we obtain

$$\begin{aligned} \iff (2N+2) * \frac{1}{2 - 2cos(m * \pi * \frac{1}{2N+2})} = \\ (2n+2) * \frac{sin(m\pi * \frac{n}{2N+2})}{2 * sin(m\pi * \frac{n}{2N+2}) - sin(m\pi * \frac{n-1}{2N+2}) - sin(m\pi * \frac{n+1}{2N+2})} \end{aligned} \tag{6.35}$$

substituting $b = m\pi/(2N+2)$

$$\iff \quad \frac{1}{2-2cos(b)} = \frac{sin(nb)}{2*sin(nb) - sin((n+1)b) - sin((n-1)b)} \tag{6.36}$$

$$\iff \quad \frac{1}{2-2cos(b)} = \frac{sin(nb)}{2*sin(nb) - 2cos(b)sin(nb)} \tag{6.37}$$

$$\iff \quad \frac{1}{2-2cos(b)} = \frac{1}{2-2cos(b)} \tag{6.38}$$

Hence the relation $w_m(n+1) * \lambda_m = F(n+1,:) * w_m$ has been shown to be true for all $'2N+1'$ rows where $w_m(a)$ and λ_m are given by equation (6.23) and (6.24).

6.2.2 Computing the optimal filter from the eigen vectors

Given the eigen vector w_m the optimal filter is given by the cumulative sum of the eigen vector and this implies that the filter (f_m) corresponding to the m^{th} eigen vector is given by

$$f_m(a) = sin(m\pi * \frac{a}{2N+2}) \; where \; a \in [1, 2N+1] \tag{6.39}$$

Equation (5.49) showed that the optimal filter is obtained from the eigen vector corresponding to the maximum eigen value. Given the eigen values are $\lambda_m = ((n+1)/(1-cos(m\pi * \frac{1}{2N+2})))$ where $m \in [1, 2N+1]$, we proceed to find the maximum eigen value. One can clearly see that λ_m is maximum when $cos(m\pi * \frac{1}{2N+2})$ is closest to one. This happens when $m = 1$. Thus the filter corresponding to the

maximum eigen value is given by the below equation and the corresponding plot of the filter is shown in Figure 6.1. In fact, the optimal filters that are obtained for the case of the Gaussian Random Walk are identical to the filters used as a part of the Discrete Sine Transform. This shows that the Discrete Sine Transform could potentially be used in the context of robust extrema.

$$f_m(a) = sin(\pi * \frac{a}{2N+2}) \; where \; a \in [1, 2N+1] \tag{6.40}$$

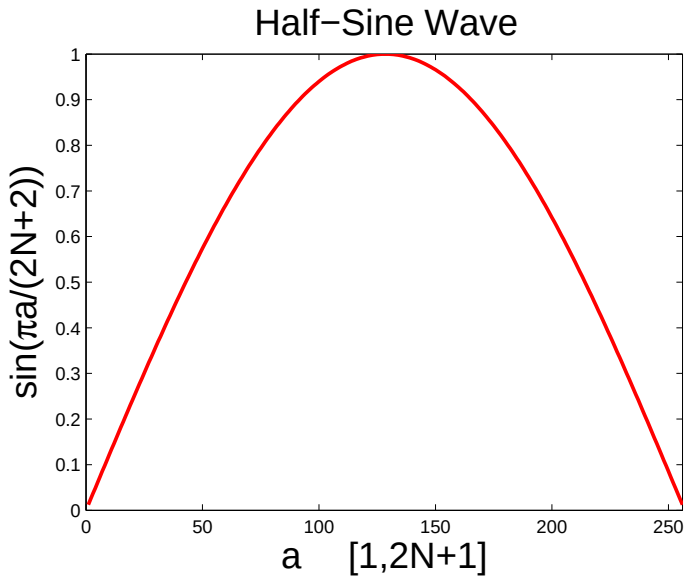

Figure 6.1: The half-sine filter.

6.3 Optimal Filter for the Case of General Random Walk Data (Discretely Sampled Levy Processes)

In the previous section, it was shown that the half sine filter is the optimal filter for Gaussian random walk data. In this section, these results are extrapolated for the case of all random walk data. Starting from equation (6.9) (shown below) in the previous section until which point no assumption was made about the nature of the time series data.

$$y_j y_j^T(\alpha,\beta) = ((2N+1) - (max(\alpha,\beta)-1)) * (M-N) * E(w^2) + \sum_{k=\alpha}^{2N+1} \sum_{i=\beta, k\neq i}^{2N+1} \Big((M-N) * E(w_{k+j-(N+1)} * w_{i+j-(N+1)}) \Big) \tag{6.41}$$

Given the information that the time series is a random walk, the increments must be independent and stationary and this implies that

From the independence property

$$E(w_{k+j-(N+1)} w_{i+j-(N+1)}) = E(w_{k+j-(N+1)}) * E(w_{i+j-(N+1)}) \tag{6.42}$$

From the stationarity property

$$E(w_{k+j-(N+1)} w_{i+j-(N+1)}) = E(w_{k+j-(N+1)})^2 = Constant(\sigma^2) \tag{6.43}$$

For the special case of a Gaussian random walk process,$E(w_{k+j-(N+1)})^2 = 0$ but for a generalized random walk process $E(w_{k+j-(N+1)})^2 = Constant(\sigma^2)$. Substituting equation (6.43) into equation (6.41) we obtain

$$y_j y_j^T(\alpha,\beta) = ((2N+1)-(max(\alpha,\beta)-1))*(M-N)*E(w^2)+\sum_{k=\alpha}^{2N+1}\sum_{i=\beta,k\neq i}^{2N+1}\Big((M-N)*\sigma^2\Big) \tag{6.44}$$

$$y_j y_j^T(\alpha,\beta) = ((2N+1)-(max(\alpha,\beta)-1))*(M-N)*E(w^2)$$
$$+((M-N)*E(w)^2)(2N+1-(max(\alpha,\beta)-1))(2N+1-(min(\alpha,\beta)-1)-1) \tag{6.45}$$

One can derive a similar equation for $z_j z_j^T$ and it results in an identical form. This finally leads to the following form for the X_{data} matrix which is as follows

$$X_{data} = K_1 \begin{bmatrix} 2N+1 & 2N & 2N-1 & ... & 2 & 1 \\ 2N & 2N & 2N-1 & ... & 2 & 1 \\ 2N-1 & 2N-1 & 2N-1 & ... & 2 & 1 \\ ... & ... & ... & ... & ... & ... \\ 2 & 2 & 2 & ... & 2 & 1 \\ 1 & 1 & 1 & ... & 1 & 1 \end{bmatrix}$$

$$+ K_2 \begin{bmatrix} (2N+1)(2N) & (2N)(2N) & (2N-1)(2N) & ... & (1)(2N) \\ (2N)(2N) & (2N)(2N-1) & (2N-1)(2N-1) & ... & (1)(2N-1) \\ (2N-1)(2N) & (2N-1)(2N-1) & (2N-1)(2N-2) & ... & (1)(2N-2) \\ ... & ... & ... & ... & ... \\ (2)(2N) & (2)(2N-1) & (2)(2N-2) & ... & (1)(1) \\ (1)(2N) & (1)(2N-1) & (1)(2N-2) & ... & 0 \end{bmatrix} \tag{6.46}$$

where K_1 & K_2 are a multiplication constants such that $K_1 > K_2$. A complete closed-form solution for the eigen vector corresponding to the maximum eigen value for the above matrix could not be obtained. Hence a numerical analysis was performed. In this method the parameter ratio K_2/K_1 was varied and the resulting optimal filter for different values of the parameter was plotted. Figure 6.2 shows that the optimal filter for the case of general random walk data is very similar to that of the Gaussian random walk case and one could approximately use the half-sine wave filter for the case of all random walk data. Note that under some circumstances the optimal filter from the highest eigen value may not result in any extrema and in these cases one must use eigen vectors corresponding to subsequent eigen values. The same simulation as above was performed to obtain the filters corresponding to the second and third eigen values and the results are shown in Figure 6.3 and Figure 6.4. It is interesting to note that the filter corresponding to

the second highest eigen vector is given by Sine wave in all the cases. A formal proof to ascertain this simulation result is shown in Appendix C. It is also interesting to see that the optimal filter for the third eigen value tends to take the form of a sombrero, which is a commonly used filter.

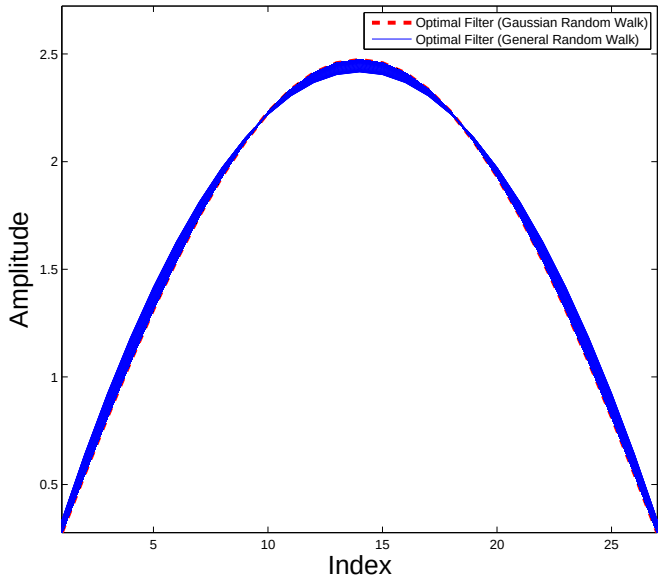

Figure 6.2: Optimal filter (Highest Eigen Value) for Random Walk Data

6.4 A Statistical Test to Check for the Applicability of the Half-Sine Filter

It is important to understand that being a random walk time series is a sufficient condition for the applicability of the half-sine wave filter as shown in sections 5.6 but is not a necessary condition for its applicability. A more general sufficient

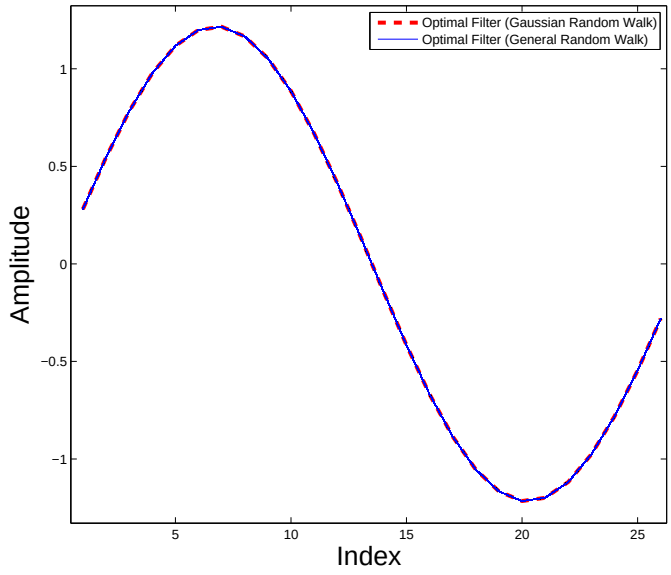

Figure 6.3: Optimal filter (Second Highest Eigen Value) for Random Walk Data

condition for the applicability of the half-sine wave is

$$E(w_{k+j-(N+1)}w_{i+j-(N+1)}) = Constant,\ Where\ k\ \&\ i \in [1, 2N+1] \tag{6.47}$$

Given that both k and i can take up to $'2N+1'$ values each. There are a total of $'(2N+1)*(2N+1)'$ different combinations of terms for which equation (6.47) needs to be verified. The task is simplified by the fact that for any particular value of k and i , say k_α and i_α and a constant β, one can show that

$$E(w_{k_\alpha+j-(N+1)}w_{i_\alpha+j-(N+1)}) = E(w_{k_\alpha+j+\beta-(N+1)}w_{i_\alpha+j+\beta-(N+1)}) \tag{6.48}$$

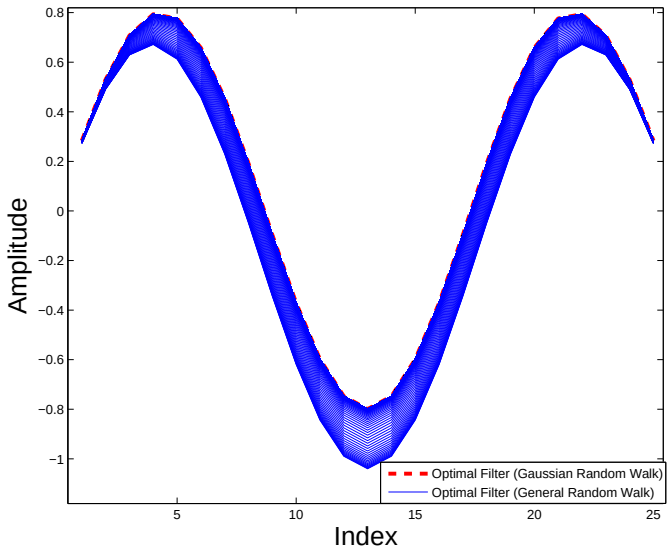

Figure 6.4: Optimal filter (Third Highest Eigen Value) for Random Walk Data

Substituting $j^{'} = j + \beta$, we obtain

$$E(w_{k_\alpha+j-(N+1)}w_{i_\alpha+j-(N+1)}) = E(w_{k_\alpha+j^{'}-(N+1)}w_{i_\alpha+j^{'}-(N+1)}) \tag{6.49}$$

Equation (6.49) is true as the expectation for the terms is computed over the variable j or $j^{'}$. This reduces the total number of different combinations for which a statistical check for equality has to be performed to $'2N+1'$ values. Therefore from equation (6.47), keeping k constant at value of 1, one has to vary i and show that the following holds true:

$$\begin{aligned} &E(w_{1+j-(N+1)}w_{1+j-(N+1)}) = E(w_{1+j-(N+1)}w_{2+j-(N+1)})... \\ &...E(w_{1+j-(N+1)}w_{2N+j-(N+1)}) = E(w_{1+j-(N+1)}w_{2N+1+j-(N+1)}) \end{aligned} \tag{6.50}$$

The above condition can be easily verified by performing a one-way anova test or non-parametric tests like the Kruskal-Wallis test. It is important to note that equation (6.47) provides a more general sufficient condition but not the necessary condition for the applicability of the half-sine filter. Through experimentation it was found that some real time series datasets violate this sufficiency condition, but yield an approximate half-sine filter as their optimal filter.

6.5 Experiemental Results : Extrema Stability Test

The "Extrema Stability" test described in Section 5.4.2 was previously performed on five different datasets and the results validated the optimization method for robustness. The same test was repeated while using the 'Half-Sine' filter and the results are compared in Figures 6.5 - 6.9. It can be seen that while the performance of the 'Half-Sine' filter is close to that of the optimal filter , the optimal filter outperforms the 'Half-Sine' filter in all the cases as it is specifically optimized for that particular dataset. The 'Half-Sine' filter is compared to the other optimal filters from the different datasets in Figure 6.10. The results comparing the performance of the 'Half-Sine' filter with the Gaussian filter and the optimal filter using surface plots are presented in Appendix-B.

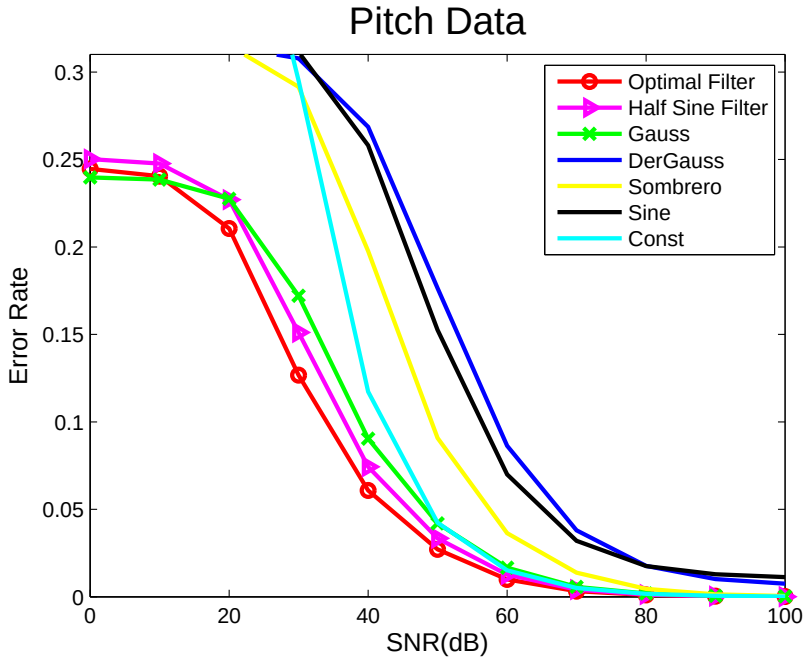

Figure 6.5: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

6.6 Experimental Results : Subsequence Matching Test

The subsequence matching test that was previously described in Section 5.4.3 was performed using the 'Half-Sine' filter in order to compare its performance to the optimal filter. The results from these tests are shown in Figures 6.11 - 6.15. It can be seen that apart from the case of the Pitch data (Figure 6.11), the optimal filter outperforms the 'Half-sine' filter for all the other datasets. The better performance of the optimal filter in those cases could be attributed to its higher robustness as seen in Figures 6.5- 6.9 but this may be only partially true as we are not considering the effect of these filters on the properties of uniqueness and cardinality. In the

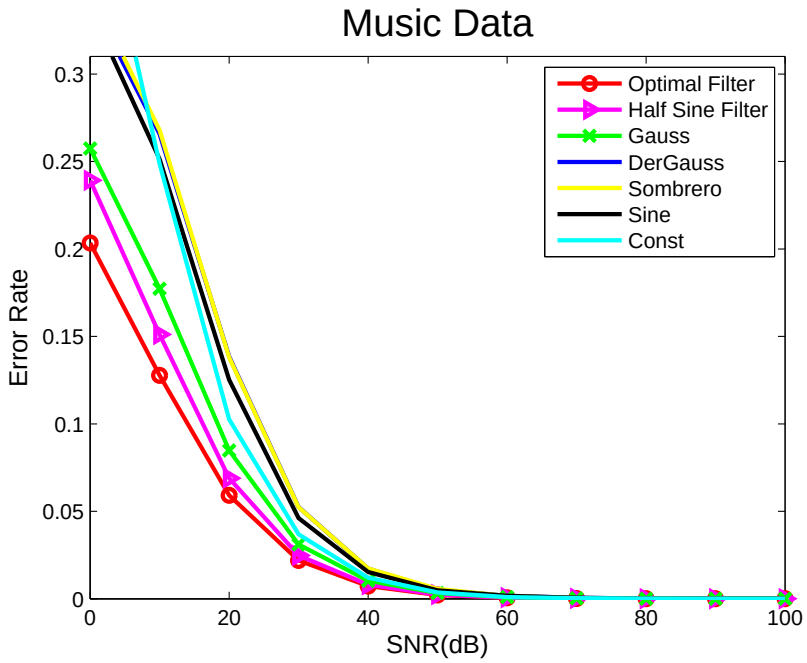

Figure 6.6: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

same way the better performance of the 'Half-Sine' filter in the case of the Pitch dataset could be possibly attributed to improvements due to changes in uniqueness and cardinality in spite of its reduced robustness as shown in Figure 6.5.

6.7 Conclusion

The above derivation provides a rigorous theoretical framework for evaluating filters for the purposes of extrema identification. The outcome of this theoretical framework was a methodology for the derivation of an optimal filter for obtaining robust extrema. This methodology led to the discovery that the half sine filter (and the Discrete Sine Transform in general) is optimal in the case of Gaussian

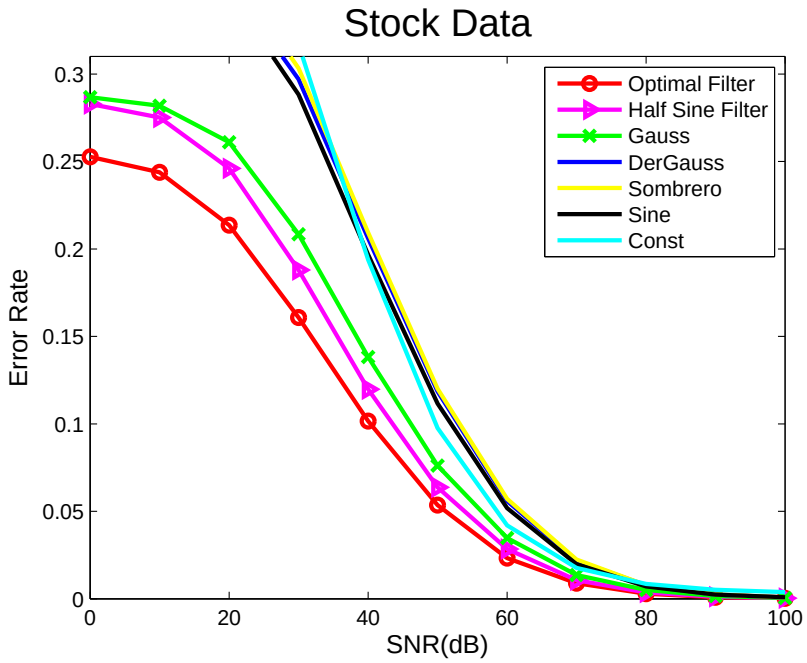

Figure 6.7: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

random walk data and approximately optimal for general random walk data. The two experimental tests not only verify these predictions but also provide a clear example of the practical applicability of these theoretical derivations.

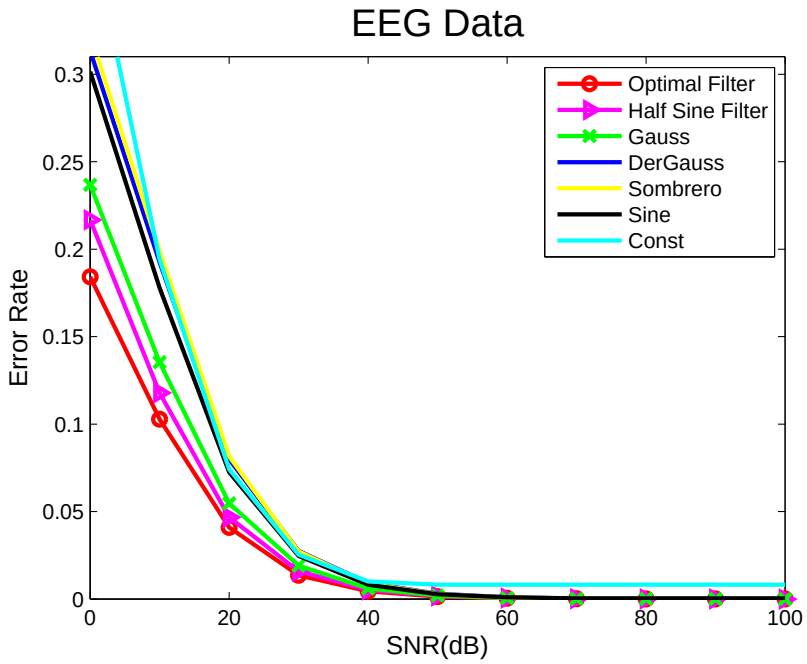

Figure 6.8: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

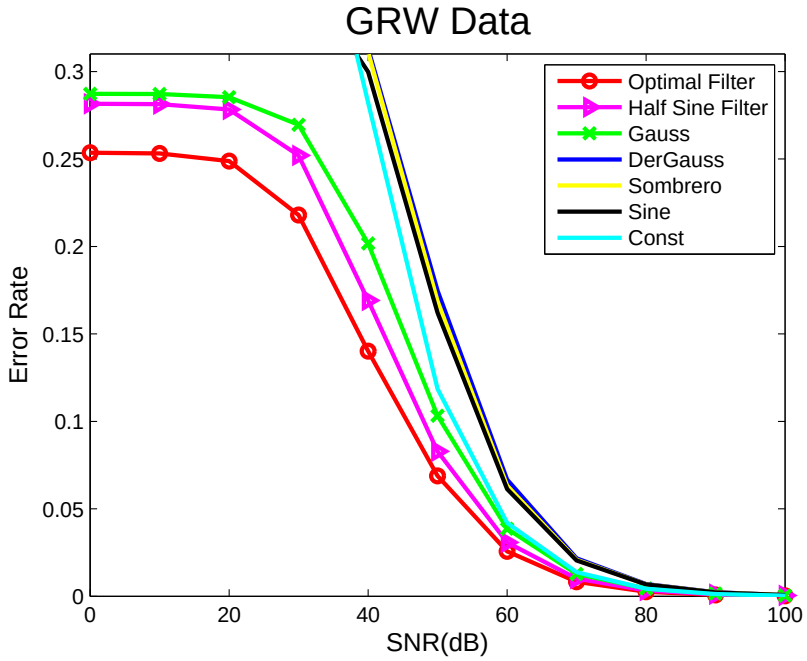

Figure 6.9: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

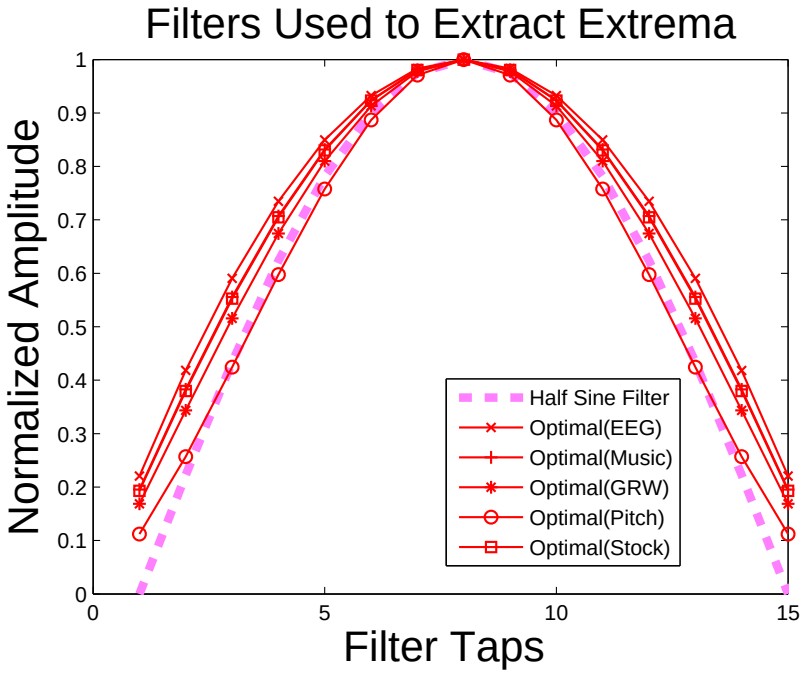

Figure 6.10: Accuracy curves show the stability of the extrema obtained by using different filters on different datasets.

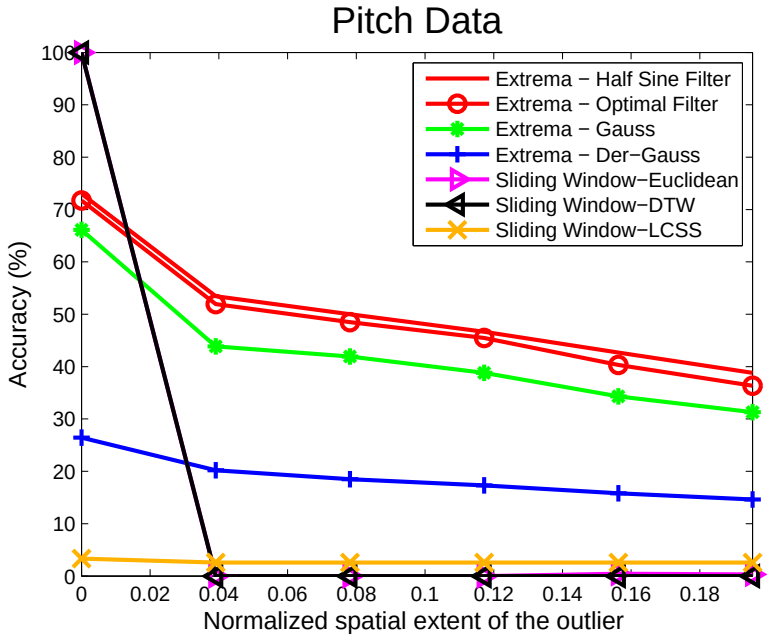

Figure 6.11: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

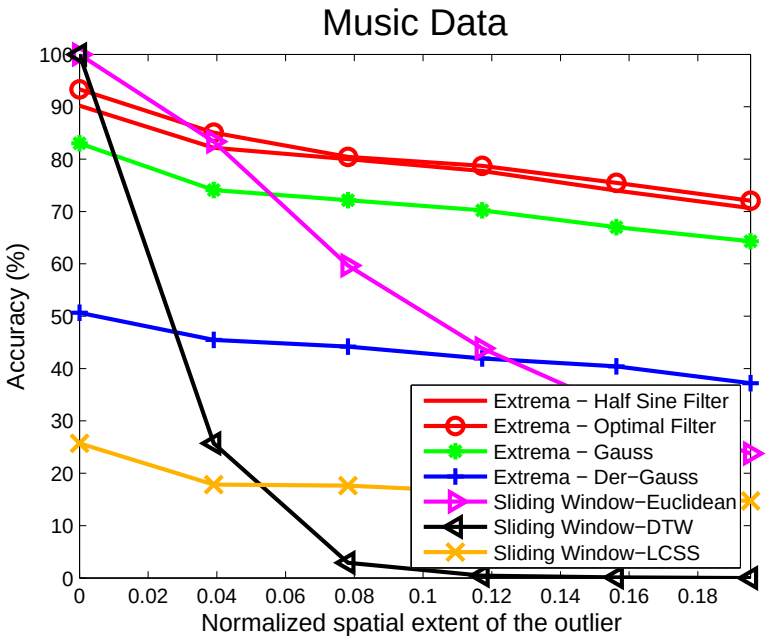

Figure 6.12: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

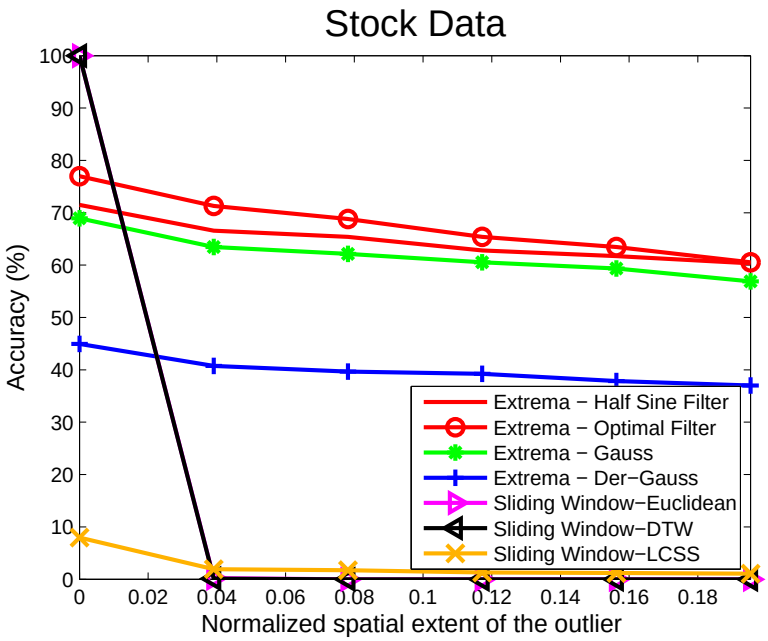

Figure 6.13: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

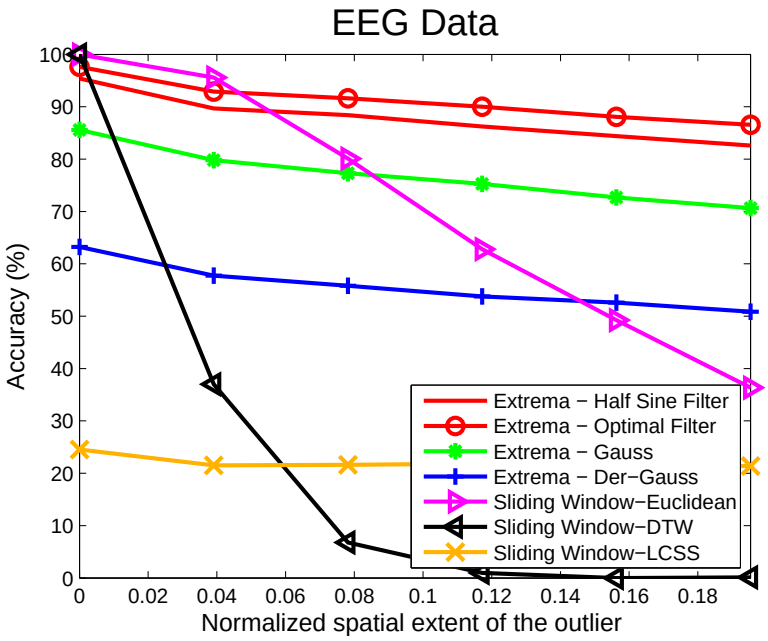

Figure 6.14: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

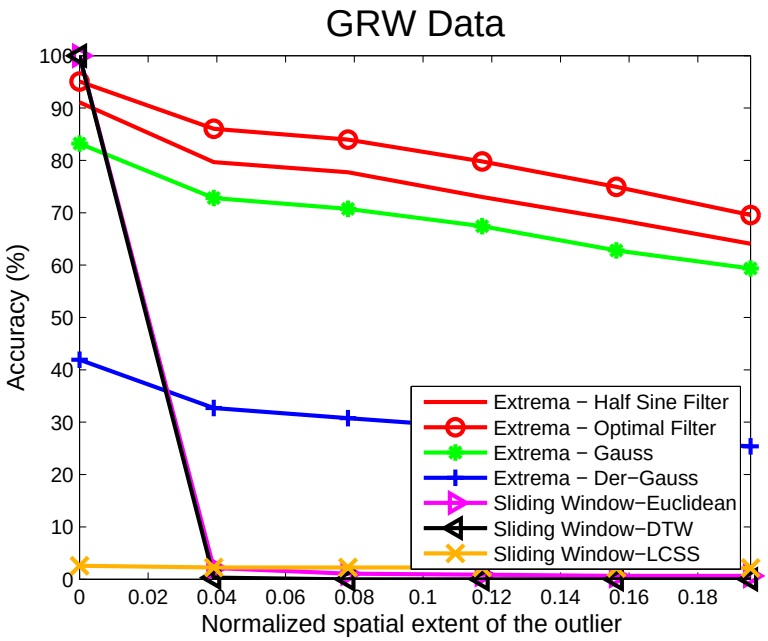

Figure 6.15: The subsequence matching results for different time series datasets while using different sliding window methods and extrema techniques.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The overall contributions of this dissertation can be broken down into three distinct categories.

1. The first category consists of the application oriented research work which focused on utilizing extrema features for performing localization using pitch and acceleration data. In this work, the standard 'Wavelet Modulus Maxima' were used as the extrema for encoding feature vectors as described in Chapter 2.

   Chapter 3 presented an algorithm that achieved global localization within very large road networks using pitch information. The result was a viable alternative to GPS localization. The proposed approach was also advantageous as it utilized inertial sensors that are already often utilized in intelli-

gent and/or autonomous systems. Theoretical analysis and simulations were performed to analyze the performance of the Multi-Scale Extrema Features. The algorithm's results in localizing a vehicle's position without initialization within a road network spanning 6000 km were also presented.

Chapter 4 presented a novel approach to find patterns in vehicle x-y-z acceleration data for use in prognostics and diagnostics. In this problem, vehicles are assumed to travel on the same routes and often times as a part of convoys but their GPS and other position information has been removed for privacy reasons. The goal of the pattern matching scheme was to identify the route or convoy associations within vehicles by using the acceleration data collected onboard those vehicles. A crucial step in solving this problem was to choose the right feature vector, as raw matching of acceleration signals is inappropriate due to different velocities, driving behaviors, vehicle loading, etc. The research work demonstrated the feasibility of using 'Multi-Scale Extrema Features' for acceleration data. The work also addressed implementation details to enhance performance for in-vehicle acceleration data, corrupted by different sources of noise. A novel 'Multi-Scale Encoding' method was also proposed for the above feature vector and it lead to a significant improvement in the performance over traditional pattern matching methods. While the main focus of the chapter was towards identifying feature vectors that effectively describe in-vehicle acceleration data, the feature vector could

potentially be used with acceleration data obtained from other applications.

2. The second category consists of the tools and techniques developed to control various properties of the extrema features. Invariably, extrema methods involve filtering of the time-series with an intuitively motivated filter (e.g. for smoothing), and subsequent thresholding to identify robust extrema. Chapter 5 delineated the feature generation process from raw time-series data, while using extrema, into three different steps 1.) filtering step, 2.) extrema detection step, and 3.) feature encoding step. Subsequently the work defined and utilized the properties of robustness, uniqueness and cardinality as a framework to identify the design choices available in each of the above steps. Unlike most methods from existing literature which utilize filters "inspired" from either domain knowledge or intuition, the proposed approach explicitly optimized the filter based on training time-series to optimize robustness of the extracted extrema features. It was demonstrated that the underlying filter optimization problem reduces to a generalized eigenvalue problem and has a tractable solution. An encoding technique that enhances the control over cardinality and uniqueness was also presented. Experimental results obtained for the problem of time-series subsequence matching established the merits of the proposed framework.

3. The final category focuses on obtaining theoretical results that help better understand the implications of the optimization technique presented in

Chapter 5. While Chapter 5 approached the problem of creating more robust extrema features in a 'deterministic sense', Chapter 6 proposed to solve the problem in a 'stochastic sense'. This stochastic approach was fruitful in delivering interesting theoretical results in situations where the underlying time-series was a Gaussian random walk or a general random walk. In case of a Gaussian random walk, the optimal filters using our proposed framework were proven to be the filters of the Discrete Sine Transform. In case of a general random walk, it was shown that the filters corresponding the highest even eigen values $(2^{nd}, 4^{th}, 6^{th}, etc)$ remained the same as those for the Gaussian Random Walk while the odd highest eigen vectors $(1^{st}, 3^{rd}, 5^{th}, etc)$ resulted in a family of curves for the optimal filters. This chapter provided closed-form solutions which could be directly utilized without having to perform the otpimization procedure incase certain stochastic characteristics of the time-series are known beforehand.

In summation, the dissertation presents three interesting facets of utilizing and developing extrema features for their application in global-localization and pattern matching of time-series data.

7.2 Future Work

There are wide range of different avenues towards which this work could be extended and some of these directions are discussed below.

7.2.1 Unifying Extrema Methods and Sliding Window Methods for Feature Extraction

Chapters 3 and 4 have shown the application of extrema features for pitch and acceleration data. However, the features that were utilized were constructed solely out of extrema and were contrasted against the sliding window methods. While such a constrast is definetly interesting, the work clearly showed that it might be possible to combine these two approaches such that one is able to incorporate the computational and memory efficiencies that extrema provide with the wide range of encoding possibilities that have been designed for sliding window methods. In this new technique, the window extraction is not performed by "sliding the window" but by using the extrema as seed points for extracting windows. These windows could now be encoded using traditional means that have been suggested for sliding window methods. The merits and the implementation details of such a scheme should be carefully studied to check to see if the promised advantages materialize in real world pattern matching tasks. A schematic that compares the proposed approach to the standard sliding window method is shown in Figure 7.1.

7.2.2 Adding the uniqueness criterion to the optimization

The optimization procedure presented in Chapter 2 found the filter that would yield the maximum robustness to the extracted extrema. It is important to remember that while robustness to noise is critical, another property that is highly desirable

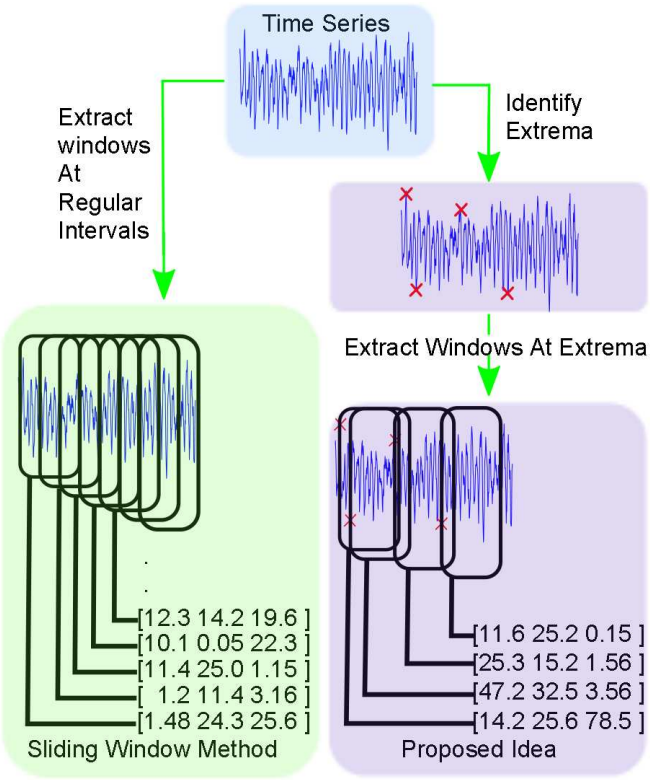

Figure 7.1: A comparision of the proposed extrema based window selection method to the standard sliding window method.

is the uniqueness of the features produced from the data. It is therefore important to optimize for both robustness and uniqueness in the optimization scheme. A drawback of optimizing for just robustness is shown in Figure 7.2. where the filters are optimally robust, but their variation along the direction of the filter is small. A technique to mitigate situations such as those in Figure 7.2 is to possibly add a uniqueness term to the optimization function in Eqn (5.34). The uniqueness term should measure the variation in the data along the direction of the filter.

Given that α in Eqn (5.34) denotes a $2N+1$ vector whose i^{th} element is denoted by α_{-N-1+i}. The filter taps are given by $T_{mat}\alpha$ where T_{mat} is a $2N+1$

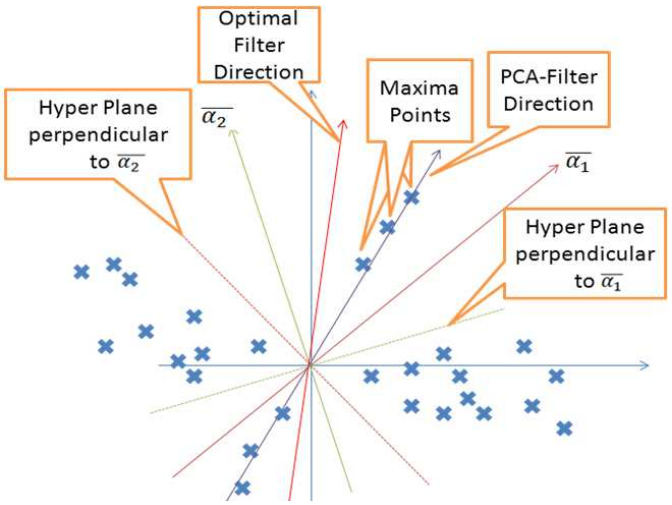

Figure 7.2: An example case in which the maxima are optimally robust are far away from the separating hyperplanes but the variation of the maxima along filter hyperplane is small and would therefore not lead to very unique features.

lower triangular matrix with 1 for all non-zero elements as shown below

$$T_{mat} = \begin{bmatrix} 1 & 0 & 0 & ... & 0 \\ 1 & 1 & 0 & ... & 0 \\ 1 & 1 & 1 & ... & 0 \\ ... & ... & ... & ... & ... \\ 1 & 1 & 1 & ... & 1 \end{bmatrix} \tag{7.1}$$

and T_{mat} is obtained as a consequence of Eqn (5.11). Let β_j denote a $2N+1$ vector whose i^{th} element is denoted by x_{-N-1+i} (In a manner similar to y_j and z_j as shown before Eqn (5.31)). Then the variation of the data when projected onto

the filter is given by $U_{uniqueness}$ or U_u.

$$U_u = \sum_{j=N+1}^{M} [(T_{mat}\alpha)^T(\beta_j - \mu)(\beta_j - \mu)^T(T_{mat}\alpha)] \tag{7.2}$$

The term μ is the mean of all the β_j vectors. Note that this measure of uniqueness of the data is heuristic in nature and may not capture the true extent of variation in the feature vector coefficients. A true measure would be the variation of the filter coefficients for a particular set of maxima but creating this term would make the problem analytically intractable and would not result in a closed form solution to the filter. The above equation can be further simplified to

$$U_u = \alpha^T[\sum_{j=N+1}^{M} [(T_{mat})^T(\beta_j - \mu)(\beta_j - \mu)^T(T_{mat})]]\alpha \tag{7.3}$$

Let $X_{uni} = [\sum_{j=N+1}^{M} [(T_{mat})^T(\beta_j - \mu)(\beta_j - \mu)^T(T_{mat})]]$. Thus, the optimization term for the uniqueness can be written as

$$U_u = \alpha^T X_{uni}\alpha \tag{7.4}$$

The modified optimization function that included robustness and uniqueness terms is given by

$$U_r + U_u = \alpha^T X_{Data}\alpha + k\alpha^T X_{uni}\alpha \tag{7.5}$$

where k is a weighting constant. Thus

$$U_{total} = U_r + U_u = \alpha^T (X_{Data} + kX_{uni})\alpha \tag{7.6}$$

Note: $(X_{Data} + kX_{uni})$ is positive definite for most time-series. The subsequent solution follows the same pattern as that in the previous section and the final solution for the matrix can be obtained obtained from the Eigen vector corresponding to the largest Eigen value of $(X_{Data} + kX_{uni})^{1/2}[I + J_{2N+1}]^{-1}(X_{Data} + kX_{uni})^{1/2}$. The benefits of the above formulation in terms of the pattern matching results mus tbe quanitified and a procedure to select the weighting constant 'k' must be delineated.

7.2.3 Representation of Extrema Detection as Unsupervised Clustering

The fundamental interpretation that makes the derivations in Chapter 2 possible is the morphing of the extrema detection technique as a geometric problem of finding two hyper planes that separate data in a higher dimensional space. The problem of separating unlabelled data is in fact the unsupervised clustering problem. Therefore one could adapt the diverse techniques that have developed to solve the clustering problem, to the optimal extrema detection method. This concept of how clustering, which has often been utilized to extract bounded regions in a

hyperspace, be utilized for extrema detection can be further demonstrated via an example. Instead of the conditions given in the equations (5.3) and (5.4) where the maxima are defined in the usual manner, let us consider the situation in which the maxima are defined as

$$\sum_{i=-N-1}^{N+1}(b_i - b_{i+1})x[n_0 - i] > t_1 \tag{7.7}$$

$$\sum_{i=-N-1}^{N+1}(b_i - b_{i+1})x[n_0 - i] < t_2 \tag{7.8}$$

$$\sum_{i=-N-1}^{N+1}(b_i - b_{i-1})x[n_0 - i] > t_1 \tag{7.9}$$

$$\sum_{i=-N-1}^{N+1}(b_i - b_{i-1})x[n_0 - i] < t_2 \tag{7.10}$$

where t_1 and t_2 are thresholds such that $t_1 > t_2 > 0$. While conditions (7.7)

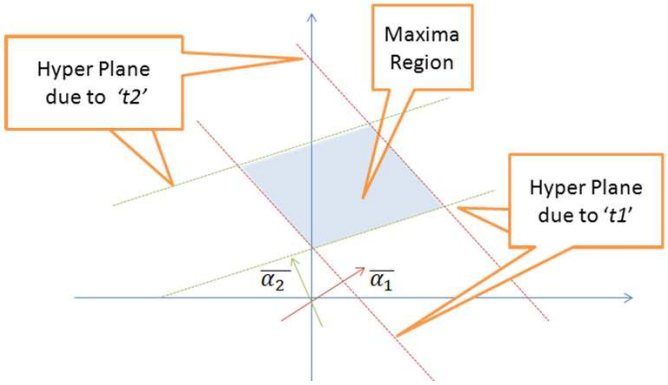

Figure 7.3: A 2D projection of the hyperspace that illustrates the boundary of the maxima region for conditions given in Eqns (7.7)-(7.10).

and (7.9) make intuitive sense, as one would want the extrema to be substantial and not be artifacts of noise, conditions (5.3) and (5.4) donot have the same

intuitive meaning. However, the above conditions lead to the maxima regions which are quadrilaterals when projected on to a two dimensional space are shown in Figure 7.3. From this illustration, we can easily see that by definition any polyhedra in higher dimensional data hyperspace can be interpreted as a set of filters resulting from conditions such as those given in Eqns (7.7)-(7.10). In fact from the supporting hyperplane theorem, we can conclude that any convex set can be interpreted as being formed from a set of such filter conditions. Of course, some of these convex regions will have infinite such filter conditions to satisfy. Thus, the problem of extrema detection is equivalent to the unsupervised clustering problem. It is important to note that as described before, all the infinite number of convex regions that can be defined over the hyperspace, many not make intuitive or make practical sense in being considered as candidates for extrema detection and optimization routines must be used to fish out those with desirable properties as was done in Chapter 2.

7.2.4 Closed-form Solutions to the General Random Walk Problem

Chapter 6 and Appendix C provided only half the solution to the problem of identifying the closed-form solution for the optimal filter in the case of random walk time-series data. The other half of the solution which corresponds to obtaining the closed-form equations for the odd highest eigen vectors $(1^{st}, 3^{rd}, 5^{th}, etc)$ is

very challenging and would lead to an interesting family of filters shown in Figure 6.2 and Figure 6.4. The mathematical properties of these family of filters would be interesting to study and and may provide greater insight into the current formulation of the optimization problem and its implications for various stochastic time-series data.

Appendix A

Global Localization using Bias and Scale Invariant Features

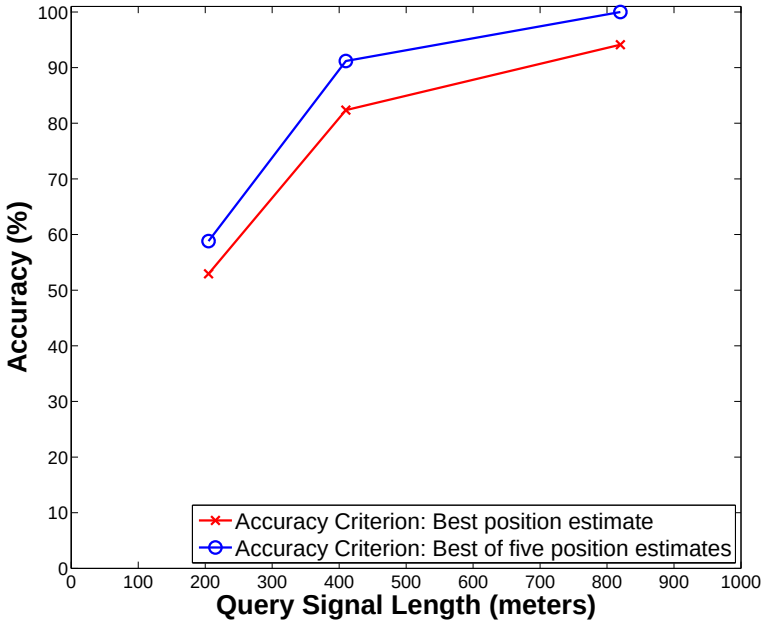

Figure A.1: The localization accuracy of the 'amplitude bias' feature vector is immune to the bias noise present in the sensor.

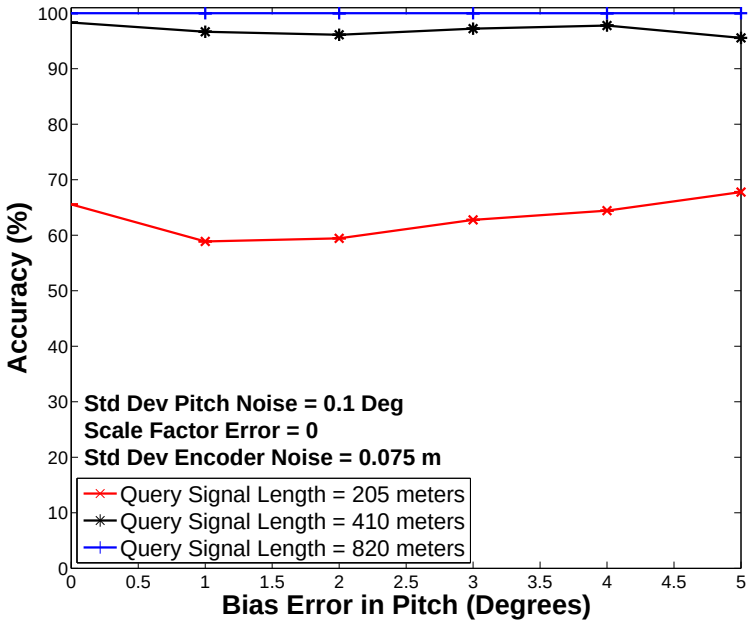

Figure A.2: The localization accuracy of the 'Amplitude Bias Amplitude Scale Time Scale' feature vector is immune to the bias noise present in the sensor.

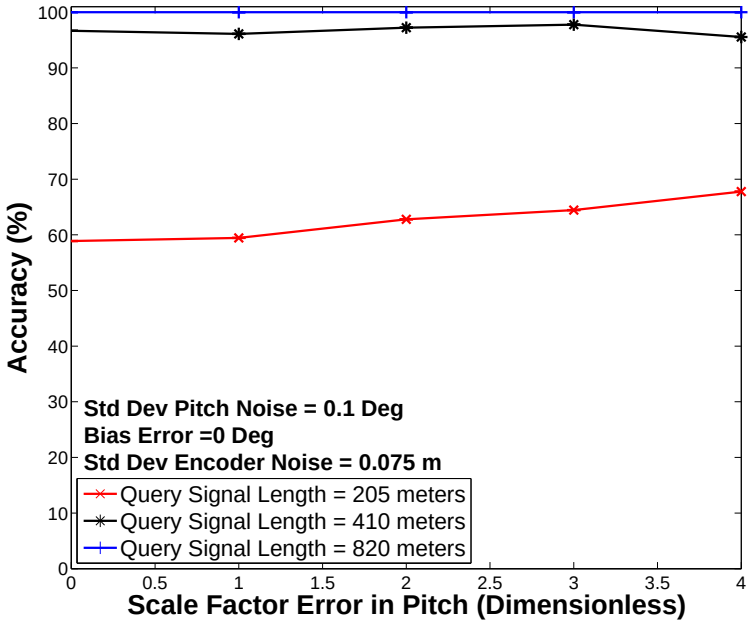

Figure A.3: The localization accuracy of the 'Amplitude Bias Amplitude Scale Time Scale' feature vector is immune to the scale noise present in the sensor.

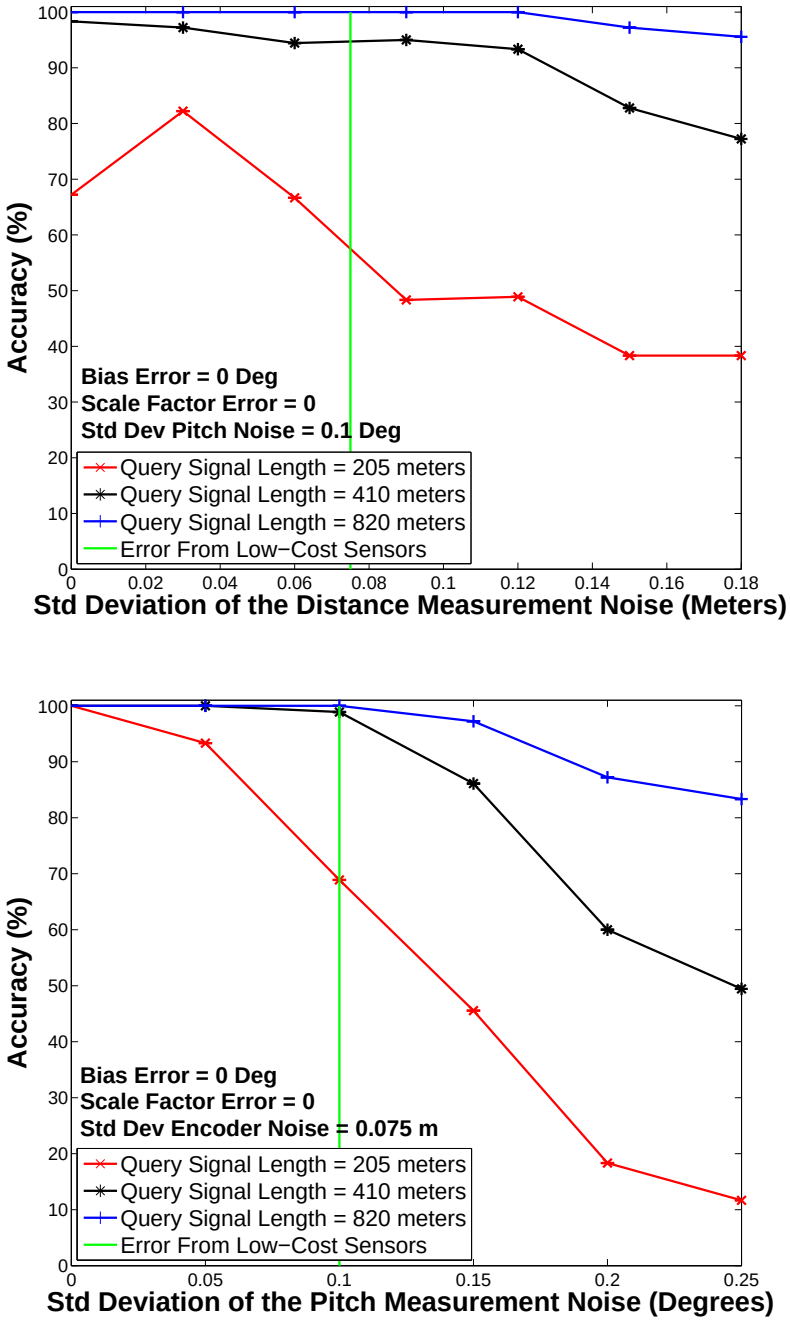

Figure A.4: The plots examine the effects of band-limited white noise in the encoder and pitch measurements on localization accuracy.

Appendix B

Extrema Stability Surface Plots

The plots comparing the 'Half-sine' filter with the optimal filter and the Gaussian filter while changing the variance parameter of the Gaussian filter are shown in Figures B.1 - B.5. The procedure for conducting the tests and the datasets that were used are described in Section 5.4.2 and Section 5.4.1.1 respectively. Analysis of the below plots reveals that the performance of the 'Half-Sine' filters isvery similar to that of the optimal filter. If one were to exclude very high SNR situations where the results could be effected by numerical errors , one can clearly see that the optimal filter delivers better performance than the 'Half-Sine' filter as it specifically customized for that dataset.

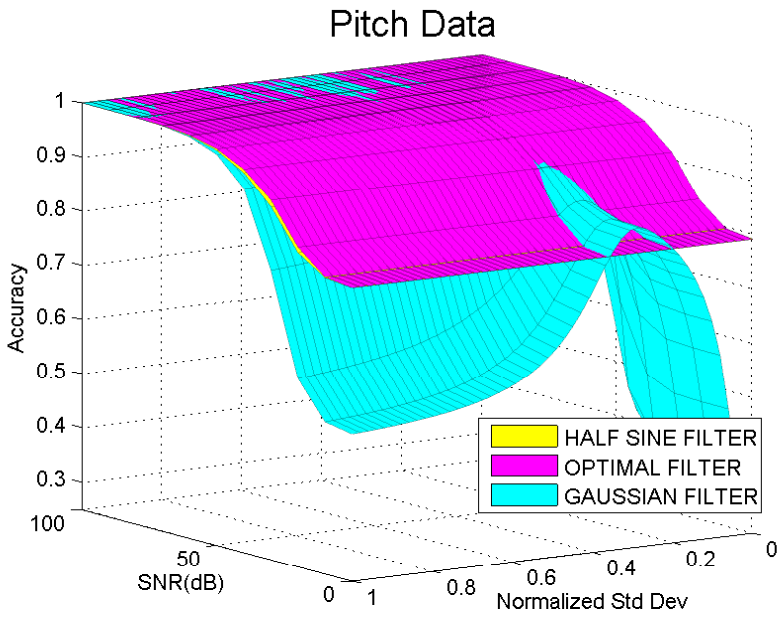

Figure B.1: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

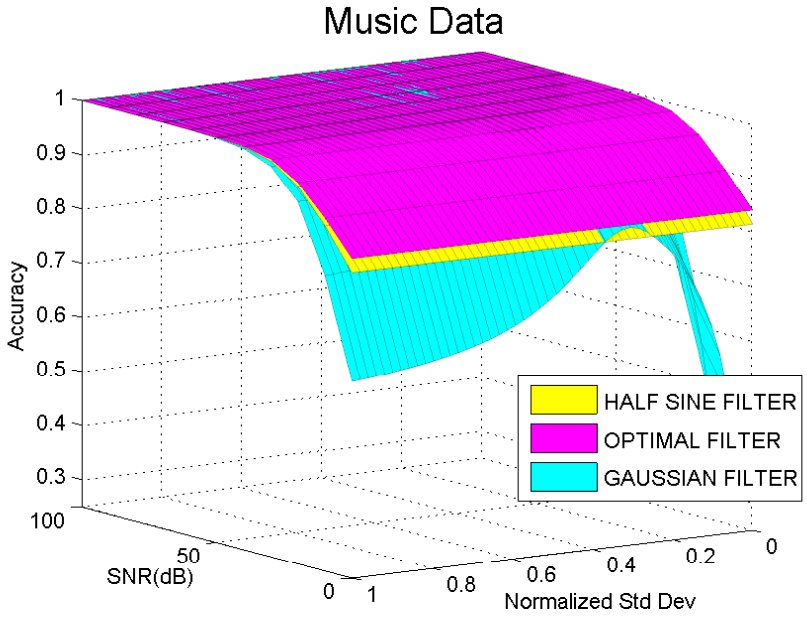

Figure B.2: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

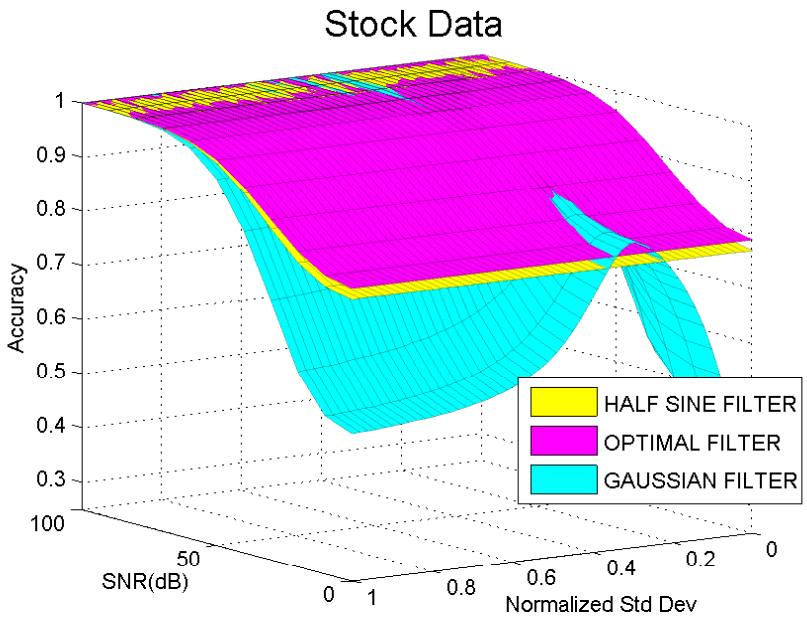

Figure B.3: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

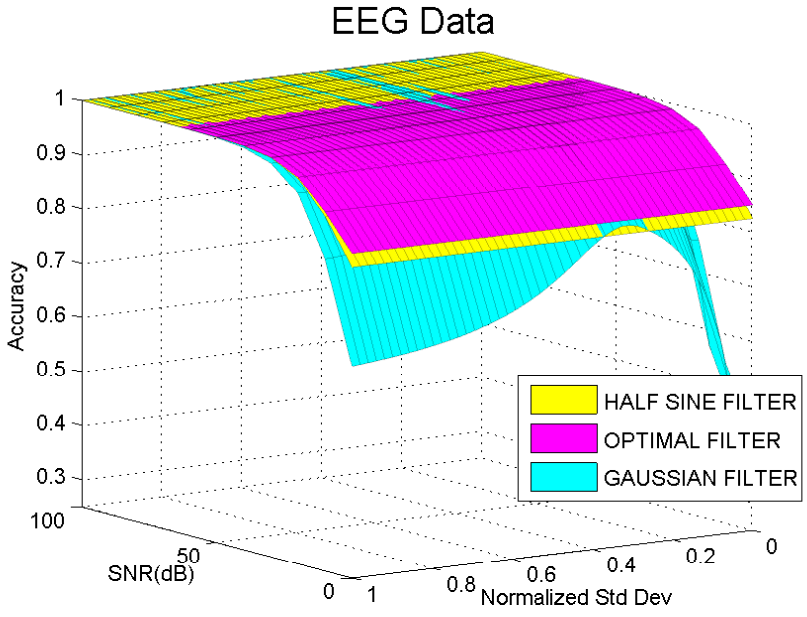

Figure B.4: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

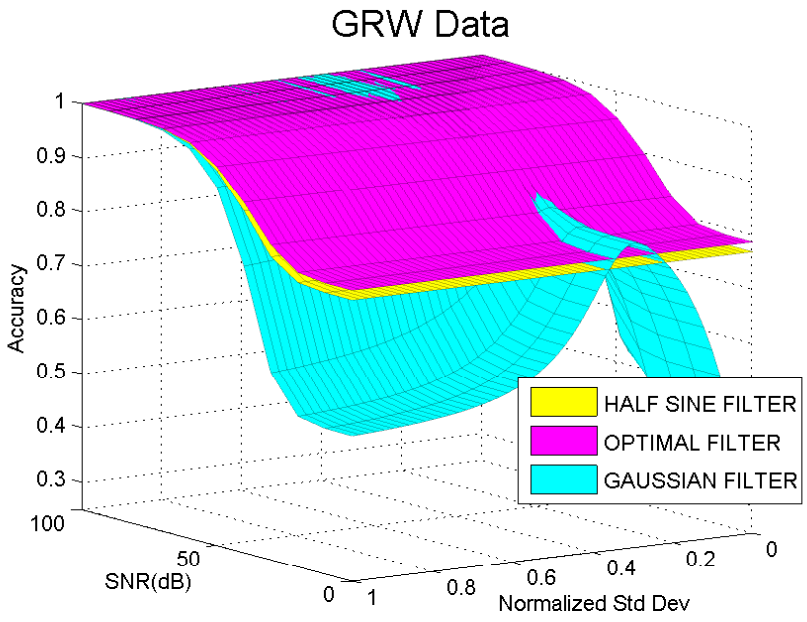

Figure B.5: Comparison of the extrema stability of the optimal filter against Gaussian filters with different standard deviations on different time series datasets.

Appendix C

Even Eigen Vectors for the Robust Extrema Formulation in the Case of General Random Walk Data (Discretely Sampled Levy Processes)

Starting with formulation for the X_{data} matrix that was given in Equation (6.46) for the case of General Random Walk data (Discretely sampled Levy process).

$$X_{data} = K_1 \begin{bmatrix} 2N+1 & 2N & 2N-1 & ... & 2 & 1 \\ 2N & 2N & 2N-1 & ... & 2 & 1 \\ 2N-1 & 2N-1 & 2N-1 & ... & 2 & 1 \\ ... & ... & ... & ... & ... & ... \\ 2 & 2 & 2 & ... & 2 & 1 \\ 1 & 1 & 1 & ... & 1 & 1 \end{bmatrix}$$

$$+ K_2 \begin{bmatrix} (2N+1)(2N) & (2N)(2N) & (2N-1)(2N) & ... & (1)(2N) \\ (2N)(2N) & (2N)(2N-1) & (2N-1)(2N-1) & ... & (1)(2N-1) \\ (2N-1)(2N) & (2N-1)(2N-1) & (2N-1)(2N-2) & ... & (1)(2N-2) \\ ... & ... & ... & ... & ... \\ (2)(2N) & (2)(2N-1) & (2)(2N-2) & ... & (1)(1) \\ (1)(2N) & (1)(2N-1) & (1)(2N-2) & ... & 0 \end{bmatrix} \tag{C.1}$$

where K_1 & K_2 are multiplication constants such that $K_1 > K_2$. Let

$$X_{org} = \begin{bmatrix} 2N+1 & 2N & 2N-1 & ... & 2 & 1 \\ 2N & 2N & 2N-1 & ... & 2 & 1 \\ 2N-1 & 2N-1 & 2N-1 & ... & 2 & 1 \\ ... & ... & ... & ... & ... & ... \\ 2 & 2 & 2 & ... & 2 & 1 \\ 1 & 1 & 1 & ... & 1 & 1 \end{bmatrix} \tag{C.2}$$

The X_{data} matrix can be written as follows.

$$X_{data} =$$

$$(K_1 + 2NK_2) * \begin{bmatrix} X_{org}(1,1) & X_{org}(1,2) & X_{org}(1,3) & ... \\ X_{org}(2,1) & X_{org}(2,2) & X_{org}(2,3) & ... \\ X_{org}(3,1) & X_{org}(3,2) & X_{org}(3,3) & ... \\ ... & ... & ... & ... \\ X_{org}(2N+1,1) & X_{org}(2N+1,2) & X_{org}(2N+1,3) & ... \end{bmatrix}$$

$$- K_2 \begin{bmatrix} 0 & X_{org}(1,2) & 2X_{org}(1,3) & 3X_{org}(1,3) & ... \\ 0 & X_{org}(2,2) & 2X_{org}(2,3) & 3X_{org}(2,3) & ... \\ 0 & X_{org}(3,2) & 2X_{org}(3,3) & 3X_{org}(3,4) & ... \\ ... & ... & ... & ... & ... \\ 0 & X_{org}(2N+1,2) & 2X_{org}(2N+1,3) & 3X_{org}(2N+1,4) & ... \end{bmatrix}$$

$$+ K_2 \begin{bmatrix} 0 & 2N & 2(2N-1) & 3(2N-2) & 4(2N-3) & ... \\ 0 & 0 & 2N-1 & 2(2N-2) & 3(2N-3) & ... \\ 0 & 0 & 0 & (2N-2) & 2(2N-3) & ... \\ 0 & 0 & 0 & 0 & 2N-3 & ... \\ ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & 0 & 0 & ... \end{bmatrix} \tag{C.3}$$

Given the above form for the X_{data} matrix, the aim of this derivation is to find the eigen vectors corresponding to even eigen values ($2^{nd}, 4^{th}, 6^{th}, etc$) of the $[I + J_{2N+1}]^{-1} X_{data}$ matrix. For a given filter of length $'2N+1'$, the $[I+J_{2N+1}]^{-1}$ matrix is given as follows.

$$[I + J_{2N+1}]^{-1} = 1/(2N+2) \begin{bmatrix} 2N+1 & -1 & ... & -1 & -1 \\ -1 & 2N+1 & ... & -1 & -1 \\ ... & ... & ... & ... & ... \\ -1 & -1 & ... & 2N+1 & -1 \\ -1 & -1 & ... & -1 & 2N+1 \end{bmatrix} \tag{C.4}$$

The matrices F_{data}, F_{org},F_{ext} and F_{rem} are defined as given below. Let

$$F_{data} = [I + J_{2N+1}]^{(-1)} X_{data} \tag{C.5}$$

Let

$$F_{org} = [I + J_{2N+1}]^{(-1)} \begin{bmatrix} X_{org}(1,1) & X_{org}(1,2) & X_{org}(1,3) & ... \\ X_{org}(2,1) & X_{org}(2,2) & X_{org}(2,3) & ... \\ X_{org}(3,1) & X_{org}(3,2) & X_{org}(3,3) & ... \\ ... & ... & ... & ... \\ X_{org}(2N+1,1) & X_{org}(2N+1,2) & X_{org}(2N+1,3) & ... \end{bmatrix} \tag{C.6}$$

Let

$$F_{ext} = [I + J_{2N+1}]^{(-1)} \begin{bmatrix} 0 & X_{org}(1,2) & 2X_{org}(1,3) & ... & ... \\ 0 & X_{org}(2,2) & 2X_{org}(2,3) & ... & ... \\ 0 & X_{org}(3,2) & 2X_{org}(3,3) & ... & ... \\ ... & ... & ... & ... & ... \\ 0 & X_{org}(2N+1,2) & 2X_{org}(2N+1,3) & ... & ... \end{bmatrix} \tag{C.7}$$

Let

$$F_{rem} = [I + J_{2N+1}]^{(-1)} \begin{bmatrix} 0 & 2N & 2(2N-1) & 3(2N-2) & 4(2N-3) & ... \\ 0 & 0 & 2N-1 & 2(2N-2) & 3(2N-3) & ... \\ 0 & 0 & 0 & (2N-2) & 2(2N-3) & ... \\ 0 & 0 & 0 & 0 & 2N-3 & ... \\ ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & 0 & 0 & ... \end{bmatrix} \tag{C.8}$$

Therefore,

$$F_{data} = (K_1 + 2NK_2)F_{org} - K_2F_{ext} + K_2F_{rem} \tag{C.9}$$

An extended form of F_{data} that highlights the relationship between F_{org} and F_{ext} is shown in equation (C.10).

Lemma C.0.1. $\sum_{i=2}^{2N+1}((2N+1)-(i-1))*(w_m(i)) = -(2N+1)sin(m*\pi/(2N+2))$ *where* m *is even and* $w_m(a) = sin(m\pi\frac{a}{2N+2}) - sin(m\pi\frac{a-1}{2N+2})$

Proof:

$$R.T.P \sum_{i=2}^{2N+1}((2N+1)-(i-1))*(w_m(i)) = -(2N+1)sin(m*\pi/(2N+2)) \tag{C.11}$$

Using the identity

$$\sum_{i=2}^{2N+1}((2N+1)-(i-1))*(cos(b*(2i-1)/2)) =$$

$$(1/4)cosec(b/2)(-2(2N+1)sin(b) + sin((2N+1)b)$$

$$+ (cos(b/2)(1-cos((2N+1)b))/sin(b/2))) \tag{C.12}$$

$$F_{data} = (K_1 + 2NK_2)\begin{bmatrix} F_{org}(1,1) & F_{org}(1,2) & F_{org}(1,3) & F_{org}(1,4) & ... \\ F_{org}(2,1) & F_{org}(2,2) & F_{org}(2,3) & F_{org}(2,4) & ... \\ F_{org}(3,1) & F_{org}(3,2) & F_{org}(3,3) & F_{org}(4,3) & ... \\ ... & ... & ... & ... & ... \\ F_{org}(2N+1,1) & F_{org}(2N+1,2) & F_{org}(2N+1,3) & F_{org}(2N+1,4) & ... \end{bmatrix}$$

$$-K_2\begin{bmatrix} 0 & F_{org}(1,2) & 2F_{org}(1,3) & 3F_{org}(1,4) & ... \\ 0 & F_{org}(2,2) & 2F_{org}(2,3) & 3F_{org}(2,4) & ... \\ 0 & F_{org}(3,2) & 2F_{org}(3,3) & 3F_{org}(3,4) & ... \\ ... & ... & ... & ... & ... \\ 0 & F_{org}(2N+1,2) & 2F_{org}(2N+1,3) & 3F_{org}(2N+1,4) & ... \end{bmatrix}$$

$$+\frac{K_2}{2N+2}\begin{bmatrix} 0 & (2N+1)(2N)-2N & (2N+1)(2(2N-1))-(2N-1)(3*2/2) & (2N+1)(3(2N-2))-(2N-2)(4*3/2) & ... \\ 0 & -2N & (2N+1)(1(2N-1))-(2N-1)(3*2/2) & (2N+1)(2(2N-2))-(2N-2)(4*3/2) & ... \\ 0 & -2N & -(2N-1)(3*2/2) & (2N+1)(1(2N-2))-(2N-2)(4*3/2) & ... \\ 0 & -2N & -(2N-1)(3*2/2) & -(2N-2)(4*3/2) & ... \\ ... & ... & ... & ... & ... \\ 0 & -2N & -(2N-1)(3*2/2) & -(2N-2)(4*3/2) & ... \end{bmatrix}$$

(C.10)

Assuming $b = m\pi/(2N+2)$ and m is even we obtain the identities $sin((2N+1)b) = -sin(b)$ and $cos((2N+1)b) = cos(b)$. Substituting these identities into the above identity we obtain

$$\iff \sum_{i=2}^{2N+1}((2N+1)-(i-1)) * (cos(b*(2i-1)/2)) =$$

$$(1/4)cosec(b/2)(-2(2N+1)sin(b) - sin(b) + (cos(b/2)(1-cos(b))/sin(b/2))) \tag{C.13}$$

$$\iff \sum_{i=2}^{2N+1}((2N+1)-(i-1)) * (cos(b*(2i-1)/2)) =$$

$$(1/4)cosec(b/2)(-2(2N+1)sin(b) - sin(b) + (cos(b/2)*2*sin(b/2))) \tag{C.14}$$

$$\iff \sum_{i=2}^{2N+1}((2N+1)-(i-1)) * (cos(b*(2i-1)/2)) =$$

$$(-1/2)cosec(b/2)(2N+1)sin(b) \tag{C.15}$$

$$\iff \sum_{i=2}^{2N+1}((2N+1)-(i-1)) * (cos(b*(2i-1)/2)) =$$

$$-(2N+1)cos(b/2) \tag{C.16}$$

Substituting $b = m\pi/(2N+2)$ we obtain

$$\iff \sum_{i=2}^{2N+1}((2N+1)-(i-1)) * (cos(m*\pi*(2i-1)/(4N+4))) =$$

$$-(2N+1)cos(m\pi/(4N+4)) \tag{C.17}$$

Starting with the L.H.S of equation (C.11) $\sum_{i=2}^{2N+1}((2N+1)-(i-1))*(sin(m\pi\frac{a}{2N+2})-sin(m\pi\frac{a-1}{2N+2}))$

$$\iff \sum_{i=2}^{2N+1}((2N+1)-(i-1))*(2*sin(m*\pi/(4N+4))cos(m*\pi*(2i-1)/(4N+4))) \tag{C.18}$$

$$\iff 2*sin(m*\pi/(4N+4))\sum_{i=2}^{2N+1}((2N+1)-(i-1))*cos(m*\pi*(2i-1)/(4N+4)) \tag{C.19}$$

Using equation (C.17)

$$\iff 2 * sin(m * \pi/(4N+4))(-(2N+1)cos(m\pi/(4N+4))) \tag{C.20}$$

$$\iff -(2N+1)sin(m * \pi/(2N+2)) \tag{C.21}$$

Hence proved.

Theorem C.0.2. *Given a square matrix F_{data} of size $'2N+1'$, (as shown in equation(C.9))*
The eigen vectors corresponding to the highest even eigen values ($2^{nd}, 4^{th}, 6^{th}, etc$) of such a matrix are given by
$w_m(a) = sin(m\pi\frac{a}{2N+2}) - sin(m\pi\frac{a-1}{2N+2})$
where $w_m(a)$ is the a^{th} element of the m^{th} eigen vector, m is even, and $a \in [1, 2N+1]$. The corresponding eigen values are given by
$\lambda_m = \frac{K_1 - K_2}{2(1 - cos(m\pi/(2N+2)))}$

From the definition of the eigen value we know that

$$(K_1 + 2NK_2)F_{org}(1,:)w_m - (K_2)F_{ext}(1,:)w_m + (K_2)F_{rem}(1,:)w_m = w_m(1) * \lambda_m \tag{C.22}$$

Similarly,

$$(K_1 + 2NK_2)F_{org}(2,:)w_m - (K_2)F_{ext}(2,:)w_m + (K_2)F_{rem}(2,:)w_m = w_m(2) * \lambda_m \tag{C.23}$$

From Lemma 6.2.2,

$$F_{org}(1,:) * w_m = F_{org}(2,:) * w_m + (2N+2) * w_m(1) \tag{C.24}$$

Subtracting equations (C.22) and (C.23)

$$(K_1+2NK_2)(2N+2)*w_m(1)/(2N+2)+(2N+2)K_2\sum_{i=2}^{2N+1}((2N+1)-(i-1))*w_m(i)/(2N+2)$$

$$= (w_m(1) - w_m(2)) * \lambda_m \tag{C.25}$$

Using Lemma C.0.1,

$$(sin(m*\pi/(2N+2)))(K_1+2NK_2-(2N+1)K_2) = (w_m(1)-w_m(2))*\lambda_m \quad (C.26)$$

$$(sin(m*\pi/(2N+2)))(K_1-K_2) = (w_m(1)-w_m(2))*\lambda_m \quad (C.27)$$

$$(sin(m*\pi/(2N+2)))(K_1-K_2) =$$

$$2sin(m*\pi/(2N+2))(1-cos(m*\pi/(2N+2)))*\lambda_m \quad (C.28)$$

$$(K_1-K_2) = 2(1-cos(m*\pi/(2N+2)))*\lambda_m \quad (C.29)$$

$$Therefore,\ \lambda_m = \frac{(K_1-K_2)}{2*(1-cos(m*\pi/(2N+2)))} \quad (C.30)$$

In order to prove that w_m is the eigen vector for the F_{data} matrix, one must show that the eigen value obtained from each row, using equations similar to (C.22) and (C.23), must be the same as equation (C.30).

$$Where,\ w_m(a) = sin(m*\pi*\frac{a}{2N+2}) - sin(m*\pi*\frac{a-1}{2N+2})\ and\ m\ is\ even \quad (C.31)$$

Starting from the last row $F_{data}(2N+1,:)$, it is required to show that

$$\sum_{\delta=1}^{2N+1} F_{data}(2N+1,\delta)w_m(\delta) = \lambda_m * w_m(2N+1) \quad (C.32)$$

Using equations (6.14),(C.10),(C.31), and (C.30)

$$\begin{aligned}
\iff & \sum_{\delta=1}^{2N+1}\Bigg(((2N+2)-\frac{(2N+2-\delta)(2N+1+\delta)}{2})(K_1+2NK_2) \\
& -((2N+2)-\frac{(2N+2-\delta)(2N+1+\delta)}{2})(K_2)(\delta-1) \\
& -(\frac{(2N+2-\delta)(\delta)(\delta-1)}{2})(K_2)\Bigg)* \\
& \left(sin(m*\pi*\frac{\delta}{2N+2}) - sin(m*\pi*\frac{\delta-1}{2N+2})\right) = \\
& \left(\frac{(2N+2)(K_1-K_2)}{2*(1-cos(m*\pi/(2N+2)))}(sin(\frac{(2N+1)m\pi}{2N+2}) - sin(\frac{(2N)m\pi}{2N+2}))\right)
\end{aligned} \quad (C.33)$$

$$L.H.S = \sum_{\delta=1}^{2N+1} \Bigg((2N+2)(K_1+2NK_2) - (\frac{(K_1+2NK_2)(2N+2-\delta)(2N+1+\delta)}{2}) - ((2N+2)(K_2)(\delta-1)) + (\frac{(2N+2-\delta)(\delta-1)(K_2)(2N+1)}{2}) \Bigg) * \Bigg(sin(m*\pi*\frac{\delta}{2N+2}) - sin(m*\pi*\frac{\delta-1}{2N+2}) \Bigg) \tag{C.34}$$

Using the identities given in equations (C.35),(C.36),(C.37) and (C.38) and substituting $n = 2N+1$ and later substituting $b = m\pi/(2N+2)$.

$$\sum_{\delta=1}^{n} w_m(\delta) = sin(nb) \tag{C.35}$$

$$\sum_{\delta=1}^{n} (n+\delta)(n+1-\delta) w_m(\delta) = (1/(2*sin^2(b/2)))((n+1)sin(bn) - nsin(b(n+1))) \tag{C.36}$$

$$\sum_{\delta=1}^{n} (\delta-1) w_m(\delta) = (-sin(\frac{bn}{2}))((-2n+1)cos(\frac{bn}{2}) + cot(\frac{b}{2})sin(\frac{b*n}{2})) \tag{C.37}$$

$$\sum_{\delta=1}^{n} (n+1-\delta)(\delta-1) w_m(\delta) = (cosec(\frac{b}{2}))(-cos(\frac{b(n+1)}{2}))((n)cos(\frac{bn}{2}) - cot(\frac{b}{2})sin(\frac{b*n}{2})) \tag{C.38}$$

$$\begin{aligned} L.H.S \; = & ((2N+2)(K_1+2NK_2)(sin((2N+1)b))) \\ & - \left(\frac{K_1+2NK_2}{4*sin^2(b/2)} ((2N+2)sin((2N+1)b) - (2N+1)sin(b(2N+2))) \right) \\ & + \left(\frac{2*(2N+2)*K_2*sin((2n+1)b/2)}{2} * \right. \\ & \left. \left(-(4N+1)cos(b(2n+1)/2) + \frac{cos(b/2)sin(b(2N+1)/2)}{sin(b/2)} \right) \right) \\ & + \left(\frac{-(2N+1)*K_2*cos((2N+2)b/2)}{2sin(b/2)} * \right. \\ & \left. \left((2N+1)cos(b(2N+1)/2) - \frac{cos(b/2)sin(b(2N+1)/2)}{sin(b/2)} \right) \right) \end{aligned} \tag{C.39}$$

Using $b = m\pi/(2N+2)$ and m is even we obtain the identities $sin((2N+1)b) =$

$-sin(b)$, $sin((2N+2)b) = 0$, and $cos((2N+1)b) = cos(b)$ .Substituting these identities into the above identity we obtain

$$\begin{aligned}
L.H.S \ =&(-(2N+2)(K_1)(sin(b))) \\
&+\left(\frac{(K_1+2NK_2)(2N+2)sin(b)}{4sin^2(b/2)}\right) \\
&-((2N+2)(2NK_2)(-sin(b))) \\
&+\left(\frac{(4N+1)(2N+2)(K_2)(sin(b))}{2}\right) \\
&+\left(\frac{(cos(b/2))(2N+2)(K_2)(1-cos(b))}{2sin(b/2)}\right) \\
&+\left(\frac{(-1)((2N+1)^2)(K_2)(2cos((2N+2)b/2)cos((2N+1)b/2))}{4sin(b/2)}\right) \\
&+\left(\frac{(cos(b/2))(2N+1)(K_2)(2cos((2N+2)b/2)sin((2N+1)b/2))}{4sin^2(b/2)}\right)
\end{aligned} \tag{C.40}$$

$$\begin{aligned}
L.H.S \ =&\left(-(2N+2)(K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) \\
&+\left(\frac{(2NK_2)(2N+2)sin(b)}{4sin^2(b/2)}\right) \\
&-((2N+2)*(2NK_2)(sin(b))) \\
&+\left(\frac{(4N+1)(2N+2)(K_2)(sin(b))}{2}\right) \\
&+\left(\frac{(2N+2)(K_2)(sin(b))}{2}\right) \\
&-\left(\frac{((2N+1)^2)(K_2)(cos(b(4N+3)/2)+cos(b/2))}{4sin(b/2)}\right) \\
&+\left(\frac{(2N+1)(K_2)(cos(b/2))(sin(b(4N+3)/2)-sin(b/2))}{4sin^2(b/2)}\right)
\end{aligned} \tag{C.41}$$

$$
\begin{aligned}
L.H.S \ = & \left(-(2N+2)(K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) \\
& + \left(\frac{(2NK_2)(2N+2)sin(b)}{4sin^2(b/2)}\right) \\
& - ((2N+2)*(2NK_2)(sin(b))) \\
& + \left(\frac{(4N+1)(2N+2)(K_2)(sin(b))}{2}\right) \\
& + \left(\frac{(2N+2)(K_2)(sin(b))}{2}\right) \\
& - \left(\frac{((2N+1)^2)(K_2)(cos(b(4N+3)/2)+cos(b/2))}{4sin(b/2)}\right) \\
& + \left(\frac{(2N+1)(K_2)(cos(b/2))(sin(b(4N+3)/2)-sin(b/2))}{4sin^2(b/2)}\right)
\end{aligned} \tag{C.42}
$$

Using $b = m\pi/(2N+2)$ and m is even we obtain the identities $cos((4N+3)b/2) = cos(b/2)$, and $sin((4N+3)b/2) = -sin(b/2)$. Substituting these identities into the above identity we obtain

$$
\begin{aligned}
L.H.S \ = & \left(-(2N+2)(K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) \\
& + \left(\frac{K_2 sin(b)}{2}((2N+2)+(4N+1)(2N+2)-2(2N)(2N+2))\right) \\
& + \left(\frac{K_2 sin(b)}{4sin^2(b/2)}((2N)(2N+2)-(2N+1)-(2N+1)^2)\right)
\end{aligned} \tag{C.43}
$$

$$
\begin{aligned}
L.H.S \ = & \left(-(2N+2)(K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) \\
& + \left(\frac{K_2 sin(b)}{2}(4N+4)\right) \\
& + \left(\frac{K_2 sin(b)}{4sin^2(b/2)}(2N+2)(-1)\right)
\end{aligned} \tag{C.44}
$$

$$
\begin{aligned}
L.H.S \ = & \left(-(2N+2)(K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) \\
& + \left((K_2 sin(b))(2N+2)(1-\frac{1}{4sin^2(b/2)})\right)
\end{aligned} \tag{C.45}
$$

$$L.H.S = \left(-(2N+2)(K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) + \left((K_2 sin(b))(2N+2)(1-\frac{1}{4sin^2(b/2)})\right) \tag{C.46}$$

$$L.H.S = \left((2N+2)(K_2-K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)})\right) \tag{C.47}$$

$$L.H.S = (2N+2)(K_2-K_1)(sin(b))(1-\frac{1}{4sin^2(b/2)}) \tag{C.48}$$

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(4sin^2(b/2)-1)}{(1-cos(b))} \tag{C.49}$$

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(1-2cos(b))}{(1-cos(b))} \tag{C.50}$$

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(1-2cos(b))(sin((2N+1)b)-sin((2N)b))}{(sin((2N+1)b)-sin((2N)b))(1-cos(b))} \tag{C.51}$$

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(1-2cos(b))(sin((2N+1)b)-sin((2N)b))}{2sin(b/2)cos((4N+1)b/2)(1-cos(b))} \tag{C.52}$$

Using $b = m\pi/(2N+2)$ and m is even we obtain the identity $cos((4N+1)b/2) = cos(3b/2)$.

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(1-2cos(b))(sin((2N+1)b)-sin((2N)b))}{2sin(b/2)cos(3b/2)(1-cos(b))} \tag{C.53}$$

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(1-2cos(b))(sin((2N+1)b)-sin((2N)b))}{(sin(2b)-sin(b))(1-cos(b))} \tag{C.54}$$

$$L.H.S = \frac{(N+1)(K_2-K_1)(sin(b))(1-2cos(b))(sin((2N+1)b)-sin((2N)b))}{(-sin(b))(1-2cos(b))(1-cos(b))} \tag{C.55}$$

$$L.H.S = \frac{(N+1)(K_1-K_2)(sin((2N+1)b)-sin((2N)b))}{(1-cos(b))} \tag{C.56}$$

$$L.H.S = \frac{(2N+2)(K_1-K_2)(sin((2N+1)b)-sin((2N)b))}{2(1-cos(b))} \tag{C.57}$$

Substituting $b = m\pi/(2N+2)$ we obtain

$$L.H.S = \frac{(2N+2)(K_1-K_2)}{2*(1-cos(m*\pi/(2N+2)))}(sin(\frac{(2N+1)m\pi}{2N+2}) - sin(\frac{(2N)m\pi}{2N+2})) \tag{C.58}$$

L.H.S is equal to R.H.S in equation (C.33). From the above derivation we have shown that the eigen value given in equation (C.30) satisfies the condition for the $(2N+1)^{th}$ row given by equation (C.32). For all the subsequent rows, the proof is obtained by mathematical induction. Suppose the equation $w_m(n+1)*\lambda_m = F_{data}(n+1,:)*w_m$ is true for row $'n+1'$ in the F_{data} matrix, we need to show that

$$w_m(n)*\lambda_m = F_{data}(n,:)*w_m \tag{C.59}$$

Substituting equation (C.9), and condition for row $'n+1'$ into equation (C.59) we obtain

$$\begin{aligned} \iff & ((K_1+2NK_2)(F_{org}(n,:)-F_{org}(n+1,:)))*w_m \\ & -((K_2)(F_{ext}(n,:)-F_{ext}(n+1,:)))*w_m \\ & +((K_2)(F_{rem}(n,:)-F_{rem}(n+1,:)))*w_m \\ & =((w_m(n)-w_m(n+1))*\lambda_m) \end{aligned} \tag{C.60}$$

Using Lemmas (6.2.1) and (6.2.2)

$$\begin{aligned} \iff & \left((K_1+2NK_2)(\sum_{i=1}^{n}(2N+2)w_m(i))\right) \\ & -\left((K_2)(\sum_{i=1}^{n}(2N+2)(i-1)w_m(i))\right) \\ & +\left((K_2)(\sum_{i=n+1}^{2N+1}(2N+2)(2N+1-(i-1))w_m(i))\right) \\ & =((2N+2)(w_m(n)-w_m(n+1))*\lambda_m) \end{aligned} \tag{C.61}$$

$$\begin{aligned}\iff & \left((K_1+2NK_2)(\sum_{i=1}^{n} w_m(i))\right) \\ & -\left((K_2)(\sum_{i=2}^{n}(2N+1)w_m(i))\right) \\ & +\left((K_2)(\sum_{i=2}^{2N+1}(2N+1-(i-1))w_m(i))\right) \\ & =((w_m(n)-w_m(n+1))*\lambda_m)\end{aligned} \tag{C.62}$$

Let $b = m\pi/(2N+2)$ and using Lemma C.0.1 and equation (C.31)

$$\begin{aligned}\iff & \left((K_1+2NK_2-K_2(2N+1))(\sum_{i=1}^{n} w_m(i))\right) \\ & +((K_2)(2N+1)w_m(1)) \\ & +((K_2)(-(2N+1)sin(b))) \\ & =((2sin(nb)-(sin((n+1)b)-sin((n-1)b)))*\lambda_m)\end{aligned} \tag{C.63}$$

Using equation (C.31)

$$\begin{aligned}\iff & ((K_1+2NK_2-K_2(2N+1))(sin(nb))) \\ & +((K_2)(2N+1)sin(b)) \\ & +((K_2)(-(2N+1)sin(b))) \\ & =((2sin(nb)-(sin((n+1)b)-sin((n-1)b)))*\lambda_m)\end{aligned} \tag{C.64}$$

$$\begin{aligned}\iff & ((K_1+2NK_2-K_2(2N+1))(sin(nb))) \\ & =((2sin(nb)-(sin((n+1)b)-sin((n-1)b)))*\lambda_m)\end{aligned} \tag{C.65}$$

$$\begin{aligned}\iff & ((K_1-K_2)(sin(nb))) \\ & =((2sin(nb)-2sin(nb)cos(b))*\lambda_m)\end{aligned} \tag{C.66}$$

$$\iff (K_1-K_2)(sin(nb)) = (2sin(nb)-2sin(nb)cos(b))*\lambda_m \tag{C.67}$$

$$\iff \frac{(K_1-K_2)(sin(nb))}{(2sin(nb)-2sin(nb)cos(b))} = \lambda_m \tag{C.68}$$

$$\iff \frac{(K_1-K_2)}{2(1-cos(b))} = \lambda_m \tag{C.69}$$

Using equation (C.30) and substituting $b = m\pi/(2N+2)$ we obtain

$$\iff \quad \frac{(K_1 - K_2)}{2 * (1 - cos(m * \pi/(2N+2)))} = \frac{(K_1 - K_2)}{2 * (1 - cos(m * \pi/(2N+2)))} \tag{C.70}$$

Hence Proved.

Bibliography

[1] Duda, R., P. Hart, and D. Stork (2007) "Pattern Classification," *Journal of Classification*, **24**(2), pp. 305–307. 1

[2] E.Bellman, R. (1957) *Dynamic Programming*, Princeton University Press. 4, 29, 122

[3] Gersho, A. and R. M. Gray (1992) *Vector Quantization and Signal Compression*, Kluwer Academic Publishers. 6

[4] Faloutsos, C., M. Ranganathan, and Y. Manolopoulos (1994) "Fast Subsequence Matching in Time-Series Databases," *ACM SIGMOD Record*, pp. 419–429. 6, 17, 42, 74, 95, 96, 99, 133, 134, 137, 142

[5] Korn, F., H. V. Jagadish, and C. Faloutsos (1997) "Efficiently Supporting Ad Hoc Queries in Large Datasets of Time Sequences," *In SIGMOD*, pp. 289–300. 6, 96

[6] Chan, K. and A. Fu (1999) "Efficient Time Series Matching by Wavelets," *Proceedings of the 15th International Conference on Data Engineering*, pp. 126–133. 6, 18, 96

[7] shing Perng, C., H. Wang, S. R. Zhang, and D. S. Parker (2000) "Landmarks: a new model for similarity-based pattern querying in time series databases," *In ICDE*, pp. 33–42. 6, 15, 16, 22, 31, 32, 97, 98, 99, 102, 104, 119, 120, 122, 123, 133, 134, 136, 137

[8] Berndt, D. J. and J. Clifford (1996) "Advances in knowledge discovery and data mining," chap. Finding patterns in time series: a dynamic programming approach, American Association for Artificial Intelligence, Menlo Park, CA, USA, pp. 229–248. 13, 17, 91

[9] Vlachos, M., D. Gunopoulos, and G. Kollios (2002) "Discovering Similar Multidimensional Trajectories," *Data Engineering, International Conference on*, p. 0673. 13, 18, 43, 77, 96, 137, 138, 142, 143

[10] Chen, L. and R. Ng (2004) "On the marriage of Lp-norms and edit distance," *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*, pp. 792–803. 13, 96

[11] Ding, H., G. Trajcevski, P. Scheuermann, and X. Wang (2008) "Querying and mining of time series data: experimental comparison of representations and distance," *Proceedings of the VLDB Endowment archive*. 13

[12] Cheng, K. and M. L. Spetch (1998) *Mechanisms of landmark use in mammals and birds*, Oxford University Press, pp. 1–17. 15, 31

[13] Chen, Y., M. A. Nascimento, B. C. Ooi, and A. K. H. Tung (2007) "SpADe: On Shape-based Pattern Detection in Streaming Time Series," *2007 IEEE 23rd International Conference on Data Engineering*, pp. 786–795. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4221727 16, 77

[14] Kristjánsson, A. and P. U. Tse (2001) "Curvature discontinuities are cues for rapid shape analysis." *Perception And Psychophysics*, **63**(3), pp. 390–403. 16, 32

[15] Shevelev, I. A., V. M. Kamenkovich, and G. A. Sharaev (2003) "The role of lines and corners of geometric figures in recognition performance." *Acta neurobiologiae experimentalis*, **63**(4), pp. 361–368.
URL http://www.ncbi.nlm.nih.gov/pubmed/15053259 16, 32

[16] Chan, K.-P. and A. W. chee Fu (1999) "Efficient Time Series Matching by Wavelets," *In ICDE*, pp. 126–133. 17, 42

[17] Keogh, E. J., K. Chakrabarti, M. J. Pazzani, and S. Mehrotra (2001) "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," *Knowl. Inf. Syst*, pp. 263–286. 17, 96

[18] Chakrabarti, K., E. Keogh, S. Mehrotra, and M. Pazzani (2002) "Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases," *Proc. ACM SIGMOD*, pp. 151–162. 17, 18, 96

[19] Keogh, E. (2002) "Exact indexing of dynamic time warping," *Proceedings of the 28th international conference on Very Large Data Bases*, pp. 406–417. 17, 18, 42, 77, 133, 137, 138, 142

[20] Park, S., W. W. Chu, J. Yoon, and C. Hsu (2000) "Efficient searches for similar subsequences of different lengths in sequence databases," *IEEE*, pp. 23–32.
URL http://portal.acm.org/citation.cfm?id=847377&dl=GUIDE 17

[21] Chen, L., M. T. Zsu, and V. Oria (2004) "Symbolic representation and retrieval of moving object trajectories," *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval MIR 04*, p. 227. URL http://portal.acm.org/citation.cfm?doid=1026711.1026749 18

[22] Goldin, D. and P. Kanellakis (1995) "On Similarity Queries for Time-Series Data," *In Proceedings of CP 95 Cassis France*. 18

[23] Chu, K. K. W. and M. H. Wong (1999) "Fast time-series searching with scaling and shifting," *Proceedings of the eighteenth ACM SIGMODSIGACTSIGART symposium on Principles of database systems PODS 99*, pp. 237–248. URL http://portal.acm.org/citation.cfm?doid=303976.304000 18

[24] Das, G., D. Gunopulos, and H. Mannila (1997) "Finding similar time series," in *Principles of Data Mining and Knowledge Discovery*, vol. 1263 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 88–100. 18

[25] Se, S., D. Lowe, and J. Little (2002) "Global localization using distinctive visual features," *IEEERSJ International Conference on Intelligent Robots and System*, **1**(October), pp. 226–231. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1041393 19, 21, 37

[26] Kosecka, J. (2004) "Global localization and relative pose estimation based on scale-invariant features," *Proceedings of the 17th International Conference on Pattern Recognition 2004 ICPR 2004*, **4**, pp. 319–322. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1333767 19

[27] Monga, V. M. V., A. Banerjee, and B. L. Evans (2006), "A clustering based approach to perceptual image hashing," . URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1597136 19

[28] Monga, V., D. Vats, and B. Evans (2005) "Image Authentication Under Geometric Attacks Via Structure Matching," in *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pp. 229 –232. 19, 21

[29] Bhattacharjee, S. and M. Kutter (1998) "Compression tolerant image authentication," *Proceedings 1998 International Conference on Image Processing ICIP98 Cat No98CB36269*, **1**, pp. 435–439. 19, 21

[30] Lowe, D. G. (2004) "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, **60**, pp. 91–110. 20, 21, 37, 101, 125

[31] Mikolajzyk, K., T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. Van Gool, and L. Van Gool (2005) "A comparison of affine region detectors," *International Journal of Computer Vision*, **65**(1-2), pp. 43–72. 20

[32] Sturm, B. L. (2007) "Stéphane Mallat: A Wavelet Tour of Signal Processing, 2nd Edition," *Computer Music Journal*, **31**(3), pp. 83–85.
URL http://www.mitpressjournals.org/doi/abs/10.1162/comj.2007.31.3.83 21, 26, 27

[33] Mallat, S. and S. Zhong (1992) "Characterization of signals from multiscale edges," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **14**(7), pp. 710–732.
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=142909 21

[34] Kicey, C. and C. Lennard (1997) "Unique reconstruction of band-limited signals by a Mallat-Zhong wavelet transform algorithm," *Journal of Fourier Analysis and Applications*, **3**, pp. 63–82. 21, 101, 104

[35] Baluja, S. and M. Covell (2007) "Audio fingerprinting: Combining computer vision & data stream processing," in *In International Conference on Acoustics, Speech, and Signal Processing (ICASSP*, pp. 213–216. 21

[36] Haitsma, J. and T. Kalker (2002) "A Highly Robust Audio Fingerprinting System," in *ISMIR*. 21

[37] Wang, A. (2006) "The Shazam music recognition service," *Commun. ACM*, **49**, pp. 44–48.
URL http://doi.acm.org/10.1145/1145287.1145312 22, 105

[38] Ellis, D. (2009) "Robust Landmark-Based Audio Fingerprinting @ONLINE," .
URL http://labrosa.ee.columbia.edu/~dpwe/resources/matlab/ 22, 105, 125, 126

[39] chung Fu, T., F. lai Chung, R. Luk, and C. man Ng (2007) "Stock time series pattern matching: Template-based vs. rule-based approaches," *Engineering Applications of Artificial Intelligence*, **20**(3), pp. 347 – 364. 22, 78, 119

[40] Zlot, R. and M. Bosse (2009) "Place Recognition using Keypoint Similarities in 2D Lidar Maps," *Dimension Contemporary German Arts And Letters*, **54**, pp. 363–372. 30, 44, 68, 69

[41] Indyk, P. (2001) "Algorithms for Nearest Neighbor Search," .
URL http://dimacs.rutgers.edu/Workshops/MiningTutorial/pindyk-slides.ppt 30, 44, 134

[42] Nister, D. and H. Stewenius (2006) "Scalable Recognition with a Vocabulary Tree," in *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, IEEE Computer Society, Washington, DC, USA, pp. 2161–2168. 30, 40, 44

[43] Chen, Y., M. A. Nascimento, B. C. Ooi, and A. K. H. Tung (2007) "SpADe: On Shape-based Pattern Detection in Streaming Time Series," *Data Engineering, International Conference on*, **0**, pp. 786–795. 32, 138

[44] Grant, A., P. Williams, N. Ward, and S. Basker (2009) "GPS Jamming and the Impact on Maritime Navigation," *Journal of Navigation*, **62**(02), p. 173. 35

[45] Li, K., H.-S. Tan, and J. K. Hedrick (2009) "Map-Aided GPS/INS Localization Using a Low-Order Constrained Unscented Kalman Filter," *Transform*, pp. 4607–4612.
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5400446 35

[46] Bosse, M. and R. Zlot (2009) "Keypoint design and evaluation for place recognition in 2D lidar maps," *Robotics and Autonomous Systems*, **57**(12), pp. 1211–1224.
URL http://linkinghub.elsevier.com/retrieve/pii/S0921889009000992 35, 36, 38, 39, 40, 74

[47] Schindler, G., M. Brown, and R. Szeliski (2007) "City-Scale Location Recognition," *2007 IEEE Conference on Computer Vision and Pattern Recognition*, **0**, pp. 1–7.
URL http://dx.doi.org/10.1109/CVPR.2007.383150 35, 36, 38, 39, 40, 68, 69, 74

[48] Vu, A., A. Ramanandan, A. Chen, J. A. Farrell, and M. Barth (2012) "Real-Time Computer Vision/DGPS-Aided Inertial Navigation System for Lane-Level Vehicle Navigation," *Intelligent Transportation Systems, IEEE Transactions on*, **13**(2), pp. 899 –913. 36

[49] Fang, H. F. H., C. W. C. Wang, M. Y. M. Yang, and R. Y. R. Yang, "Ground-Texture-Based Localization for Intelligent Vehicles," . 36

[50] Dellaert, F., D. Fox, W. Burgard, and S. Thrun (1999) "Monte Carlo localization for mobile robots," *Proceedings 1999 IEEE International*

Conference on Robotics and Automation Cat No99CH36288C, **2**(May), pp. 1322–1328.
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=772544 36, 38

[51] Dean, A., P. Vemulapalli, and S. Brennan (2008) "Highway Evaluation of Terrain-Aided Localization Using Particle Filters," *Proceedings of the 2008 ASME Dynamic Systems and Control Conference*. 36, 38, 40, 74

[52] Dean, A. (2008) "Terrain-based Road Vehicle Localization using Attitude Measurements," *The Pennsylvania State University, State College, Pennsylvania*. 36, 38, 40, 63, 68

[53] Fox, D., S. Thrun, W. Burgard, and F. Dellaert (2001) *Particle filters for mobile robot localization*, Springer-Verlag, pp. 470–498.
URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.1.9914 36

[54] Fox, D. (2001) "KLD-sampling: Adaptive particle filters and mobile robot localization," *Advances in Neural Information Processing Systems NIPS*, p. 26Ű32.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.21.5786&rep=rep1&type=pdf 36

[55] Dellaert, F., W. Burgard, D. Fox, and S. Thrun (1999) "Using the condensation algorithm for robust, vision-based mobile robot localization," *Proceedings 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Cat No PR00149*, **2**, pp. 588–594.
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=784976 37

[56] Bosse, M. and R. Zlot (2008) "Map Matching and Data Association for Large-Scale Two-dimensional Laser Scan-based SLAM," *The International Journal of Robotics Research*, **27**(6), pp. 667–691.
URL http://ijr.sagepub.com/cgi/doi/10.1177/0278364908091366 37

[57] Murillo, A. and J. Kosecka (2009) "Experiments in place recognition using gist panoramas," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pp. 2196 –2203. 37

[58] David, P. and S. Ho (2011), "Orientation descriptors for localization in urban environments," .
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6048164 37

[59] Tabatabaei, S. A. H., M. Fleury, N. N. Qadri, S. Member, and M. Ghanbari (2011) "Improving Propagation Modeling in Urban Environments for Vehicular Ad Hoc Networks," *Transportation*, **12**(3), pp. 705–716. 38

[60] Drawil, N. M. and O. Basir (2010), "Intervehicle-Communication-Assisted Localization," .
URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5464357 38

[61] Ledwich, L. and S. Williams (2006) "Reduced SIFT Features For Image Retrieval and Indoor Localisation," *Australian Conference on Robotics and Automation*, **322**. 38, 40

[62] Vemulapalli, P., A. Dean, and S. Brennan (2011) "Pitch Based Vehicle Localization Using Time Series Subsequence Matching with Multi-Scale Extrema Features," *American Control Conference*. 40, 74, 97, 104, 122, 123, 126

[63] Kadetotad, S., P. Vemulapalli, S. Brennan, and C. Lagoa (2011) "Terrain-Aided Localization Using Feature-Based Particle Filtering," *Proceedings of the ASME DSCC*. 40, 74, 97, 98, 122

[64] Ye, L. and E. Keogh (2010) "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification," *Data Mining and Knowledge Discovery*, **22**(1-2).
URL http://www.springerlink.com/index/10.1007/s10618-010-0179-5 42, 78

[65] Elkaim, G. H., M. Lizarraga, and L. Pedersen (2008) "Comparison of Low-Cost GPS / INS Sensors for Autonomous Vehicle Applications," *Sensors Peterborough NH*, pp. 1133–1144.
URL http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4570000 63, 68

[66] Sadhukhan, D., C. Moore, and E. Collins (2004) "Terrain estimation using internal sensors," , pp. 195–199. 72

[67] Naito, A., C. Miyajima, T. Nishino, N. Kitaoka, and K. Takeda (2009) "Driver evaluation based on classification of rapid decelerating patterns," in *Vehicular Electronics and Safety (ICVES), 2009 IEEE International Conference on*, pp. 108 –112. 72

[68] Laerhoven, K. V., E. Berlin, and B. Schiele (2009) "Enabling Efficient Time Series Analysis for Wearable Activity Data," , pp. 392–397.

URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5381499 72, 95

[69] Chen, L. and M. T. Özsu (2005) "Robust and fast similarity search for moving object trajectories," in *In SIGMOD*, pp. 491–502. 77

[70] Huang, N. E., Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N. C. Yen, C. C. Tung, and H. H. Liu (1998) "The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis," *Proceedings of the Royal Society A Mathematical Physical and Engineering Sciences*, **454**(1971), pp. 903–995.
URL http://rspa.royalsocietypublishing.org/cgi/doi/10.1098/rspa.1998.0193 94

[71] Marteau, P.-F. (2009) "Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **31**(2), pp. 306 –318. 95, 96

[72] Keogh, E. (2003) "Welcome to the UCR Time Series Classification/Clustering Page," .
URL http://www.cs.ucr.edu/$\sim$eamonn/time_series_data/ 95

[73] Keogh, E., S. Lonardi, and B. Y.-c. Chiu (2002) "Finding surprising patterns in a time series database in linear time and space," *8th ACM SIGKDD*, pp. 550–556. 95

[74] Chen, Q., L. Chen, X. Lian, and Y. Liu (2007) "Indexable PLA for Efficient Similarity Search," *Proc. VLDB*, pp. 435–446. 96

[75] Cai, Y. (2004) "Indexing Spatio-Temporal Trajectories with Chebyshev Polynomials," *Proc. ACM SIGMOD*, pp. 599–610. 96

[76] Lin, J., L. Wei, and et al. (2007) "Experiencing SAX: a Novel Symbolic Representation of Time Series," *Data Mining and Knowledge Discovery*, pp. 107–144. 96

[77] Rath, T. M. and R. Manmatha (2003) "Word image matching using dynamic time warping," *2003 IEEE Proceedings CVPR*, **2**. 96

[78] Vemulapalli, P., V. Monga, and S. Brennan (2012) "Optimally Robust Extrema Filters for Time Series Data," *to appear in American Control Conference*. 97

[79] Wang, A. (2003) "An Industrial-Strength Audio Search Algorithm," *4th Symposium Conference on Music Information Retrieval*, pp. 7–13. 97, 98, 122

[80] Tuytelaars, T. and K. Mikolajczyk (2008) "Local invariant feature detectors: a survey," *Found. Trends. Comput. Graph. Vis.*, **3**, pp. 177–280. 101

[81] Canny, J. (1986) "A Computational Approach to Edge Detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **PAMI-8**(6), pp. 679 –698. 101, 119, 120, 125

[82] Bourennane, E., P. Gouton, M. Paindavoine, and F. Truchetet (2002) "Generalization of Canny-Deriche filter for detection of noisy exponential edge," *Signal Processing*, **82**(10), pp. 1317 – 1328. 101, 120

[83] McAteer, R., P. Kestener, A. Arneodo, and A. Khalil (2010) "Automated Detection of Coronal Loops Using a Wavelet Transform Modulus Maxima Method," *Solar Physics*, **262**, pp. 387–397. 104

[84] Boyd, S. and L. Vandenberghe (2004) *Convex Optimization*, Cambridge University Press. 115

[85] Stern, R. J. and H. Wolkowicz (1995) "Indefinite Trust Region Subproblems and Nonsymmetric Eigenvalue Perturbations," *SIAM J.Optim.*, pp. 286–313. 116, 117

[86] Anstreicher, K., X. Chen, H. Wolkowicz, and Y.-X. Yuan (1999) "Strong duality for a trust-region type relaxation of the quadratic assignment problem," *Linear Algebra and its Applications*, **301**(1-3), pp. 121 – 136. 116, 118

[87] Petrou, M. and J. Kittler (1991) "Optimal edge detectors for ramp edges," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **13**(5), pp. 483 –491. 120

[88] Fink, H. S., Eugene; Gandhi (2010) "Compression of time series by extracting major extrema," *Journal of Experimental and Theoretical Artificial Intelligence*, pp. 89–106. 120

[89] Andrzejak, R., K. Lehnertz, C. Rieke, F. Mormann, P. David, and C. lger (2001) "Indications of nonlinear deterministic and finite dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state," *Phys. Rev. E.* 127

[90] Vlachos, M., M. Hadjieleftheriou, D. Gunopulos, and E. Keogh (2006) "Indexing Multidimensional Time-Series," *The VLDB Journal*, **15**, pp. 1–20. 133, 142

[91] Fama, E. F. (1965) "Random Walks in Stock Market Prices," *Financial Analysts Journal*, **21**(5), pp. pp. 55–59. 153

Vita

Pramod K. Vemulapalli