Activity Recogniton using Singular Value Decomposition

by

Vineet K. Jolly

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Dr. Mark T. Jones, Co-chair
Dr. Thomas L. Martin, Co-chair
Dr. Paul E. Plassmann

August 9, 2006
Blacksburg, Virginia

Keywords: E-Textiles, activity recognition, singular value decomposition

Activity Recognition using Singular Value Decomposition

Vineet K. Jolly

(ABSTRACT)

A wearable device that accurately records a user's daily activities is of substantial value. It can be used to enhance medical monitoring by maintaining a diary that lists what a person was doing and for how long. The design of a wearable system to record context such as activity recognition is influenced by a combination of variables. A flexible yet systematic approach for building a software classification environment according to a set of variables is described. The integral part of the software design is the use of a unique robust classifier that uses principal component analysis (PCA) through singular value decomposition (SVD) to perform real-time activity recognition. The thesis describes the different facets of the SVD-based approach and how the classifier inputs can be modified to better differentiate between activities. This thesis presents the design and implementation of a classification environment used to perform activity detection for a wearable e-textile system.

Acknowledgements

There are a lot of people I want to thank without whose support this thesis would not have been completed. For those who are not mentioned specifically, thank you.

I would like to thank my advisor Dr. Mark Jones for painstakingly proofreading my thesis over and over again. His advice, wisdom, and words of encouragement always came at the right time.

I would like to thank Dr. Tom Martin for his guidance and for serving on my committee.

I would also like to thank Dr. Paul Plassmann for serving on my committee.

I would like to thank the members of the Configurable Computing Lab and E-Textiles group from whom I learned many things, from juggling to solving the rubik cube. I would like to especially thank Josh Edmison, David Lehn, Christopher Einsmann, Braden Sawyer, Christopher Zeh, Justin Rice, Ingrid Burbey, Tingting Meng, and Edward Curley.

Above all I would like to dedicate this thesis to my parents Vijay and Anita Jolly and sister, Smriti, without whose love and support I wouldn't be the person that I am today.

This material is based upon work supported by the National Science Foundation under Grant No. CCR-0219809, CNS-0447741, and CNS-0454195. Any opinions, findings and conclusions or recomendations expressed in this material are those of the author(s) and do

not necessarily reflect the views of the National Science Foundation (NSF).

Contents

1 Introduction **1**

1.1 Motivation 1

1.2 Contributions 3

1.3 Thesis Organization 3

2 Background **5**

2.1 Wearable Devices and Electronic-Textiles 5

2.2 Context Awareness 9

2.3 Activity Recognition Methods 10

2.4 Singular Value Decomposition 12

2.4.1 Latent Semantic Indexing 12

2.5 Principal Component Analysis 13

3 Design Guidelines for the Classification Environment **14**

3.1 Final System Design 15

3.2 Factors Influencing Generation of Activity Inputs 18
3.2.1 Format of Data Files 18
3.2.2 Classification Window Size 19
3.2.3 Sensors Utilization 19
3.2.4 Sampling Rate 20
3.2.5 Pre-processing Data 20
3.2.6 Statistics and Collection Windows 21
3.3 Feature Extraction Algorithm 22
3.4 Factors Influencing the SVD Classifier 24

4 The Classifier - Vector Generation and Classification 25

4.1 Vector Generation Mode 26
4.1.1 Creation of the Data Matrix 26
4.1.2 Singular Value Decomposition of the Data Matrix 26
4.1.3 Activity Vector Generation 27
4.2 Classification Mode 27
4.2.1 Query Vector Generation 27
4.2.2 Query Matching and Activity Recognition 28
4.2.3 Using Different Activity Sets 28
4.3 Real-time Classification 29

4.3.1 Creation of Classification Constant Files 30

4.3.2 Online Activity Recognition 30

5 Results **32**

5.1 Selection of k Over Different Activity Sets 33

5.1.1 An Example for Selecting k 33

5.1.2 Classification Errors over Different Activity Sets 35

5.1.3 Determining an Optimal k A Priori 37

5.2 Change in Classification Error by Varying the Activity Inputs 40

5.3 Modifying the Feature Space to Improve Results 43

5.4 Longer Classification Windows and Activity Transitions 50

5.4.1 Longer Complex Activities: Classification Window of 12 seconds . . . 50

5.4.2 Activity Transitions 54

5.5 Final Online Results for 17 Activities 58

6 Conclusions **62**

6.1 Future Work 63

Bibliography **64**

List of Figures

1.1 The sensing and classification variable sets that represent the design space . 2

2.1 LifeVest wearable Defibrillator 6

2.2 The wearable motherboard used to monitor soldier's vital signs 8

3.1 Flowchart of a typical pattern recognition system 15

3.2 The e-textile pants used to collect data for all experimentation 16

3.3 Flowchart depicting how the software model generates classifier inputs 18

3.4 Sit, stand, and lying back being represented by their 2D feature space 22

4.1 Activity - query matching in 3D space 29

4.2 Flowchart of online activity recognition 31

5.1 The basic five activities plotted in 1-, 2-, and 3-dimensional space 34

5.2 Correct classifications over k for different activity lists 37

5.3 Correct classifications over k as a function of S for smaller activity lists . . . 38

5.4 Correct classifications over k as a function of S for expanded activity lists . . 39

5.5 Minimize errors by choosing optimal k and readjusting sensing variables . . . 46

5.6 The relationship between error percent in activity detection and sampling rate. 48

5.7 The foxtrot, the waltz, and the east coast swing dance steps 51

5.8 Shifting window approach to classify subject transitions (sit-stand-run-stand) 57

5.9 Sliding window approach to classify subject transitions (sit-stand-run-stand) 57

5.10 The subject transitions through a series of activities 58

5.11 List of 26 activity inputs and 17 outputs 60

List of Tables

3.1 Description of the Sensors 17

3.2 Feature space of each activity 23

5.1 Correctly classified activities over different k values 34

5.2 List of activity sets and the activities they contain 35

5.3 Optimal k values for different activity inputs 36

5.4 Classifier accuracy for activity instances collected over the same subject . . . 41

5.5 Confidence Intervals for activity instances collected over the same subject . . 42

5.6 Classifier accuracy for activity instances collected over a set of subjects . . . 44

5.7 Confidence Intervals for activity instances collected over a set of subjects . . 45

5.8 The effect of sampling rate on classification error 49

5.9 Dance classification using activity inputs from a single subject 52

5.10 Dance classification using inputs from a combination of subjects 53

5.11 Confidence Intervals for different dances collected over the same subject . . . 54

5.12 The confidence intervals calculated for inputs generated from a group of subjects 55

5.13 Percent errors of individual activity sets 61

5.14 Various subjects activity files and error percents 61

Chapter 1

Introduction

1.1 Motivation

Activity recognition is the process of determining the different activities performed by a person over a period of time. Correctly classifying activities can be used to create an activity diary, which lists what the person was doing when and for how long. Activity classification can be used to cross-check activity patterns versus measurements of heart activity from implanted sensors, for wearable monitoring of heart patients [1] or activity-driven temperature regulation for *smart* clothing [2]. In general, detecting activity is a valuable component in context-aware systems. Recognizing user activity is not a new concept [3], but has been hampered by the difficulty of placing sensors near the body discretely and comfortably. Another problem is the difficulty in mapping low-level sensor data to higher level abstractions [4].

The implementation of context-aware systems involves the consideration of many design variables, as described in [5]. These variables can be subdivided into two sets of variables,

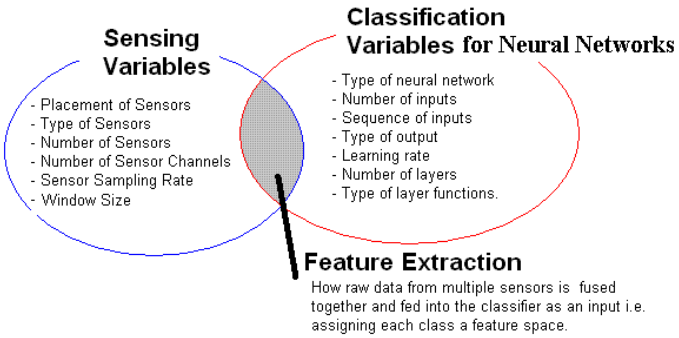

Figure 1.1: The sensing and classification variable sets that represent the design space

namely, sensing and classifier variables. Figure 1.1 shows different variables that could be combined to develop an application. The sensing variable set is usually based on the application and common sense. They are used to generate classifier inputs from the gathered sensor data. The classifier variables depend on the type of classifier used. One of the most popular and robust classification technique for detecting context is neural networks. They can provide an excellent black box solution to discern high dimensional inputs if trained and configured properly. One problem with using neural networks as a classifier is that they introduce a large number of variables that have many combinations [6]. Neural networks tend to require relatively large amounts of storage, processing power, and may be trained prior to use [7]. The primary goal of this thesis is to develop a unique classifier that reduces the percentage of error in the classification of different everyday activities. The classifier introduces a single variable, the value for which is determined such that it reduces classification error. A classification environment is developed to perform rigorous testing on the classifier and showcase its efficiency. This classifier overcomes the central problems

of statistical pattern recognition by generalizing to new patterns while not being overly concerned with optimizing feature extraction [8].

1.2 Contributions

This thesis presents the design and implementation of a classification environment used to perform activity detection for a wearable e-textile system. The classifier developed uses Singular Value Decomposition (SVD) to determine distinguishing features among the high-dimensional classifier inputs and then uses principal component analysis to map these high-dimensional inputs into a coordinate space of reduced dimensionality. The single classification variable is the coordinate space to which the classifier inputs are mapped. This variable is determined by the classification environment, by computing the percentage of error in activity classification over a batch of activity files not recognized by the classifier. The classification environment is very flexible and the overall accuracy can be further improved by modifying the sensing variables. The classification environment is also extended to perform activity detection in real-time using a wearable e-textile prototype. Both the classification environment and the e-textile prototype show high levels of accuracy for activity classification when tested over multiple subjects, in multiple trials.

1.3 Thesis Organization

The thesis is organized as follows. Chapter 2 discusses the background information necessary for understanding the work performed in this research. Chapter 3 discusses the design issues related to sensing variables and extracting distinctive features from raw sensor input. Chapter 4 describes how the system generates activity vectors from classifier inputs and

performs activity recognition on a wearable prototype in real-time. Chapter 5 describes results obtained from the experiments. Chapter 6 summarizes the contributions of this thesis, presents conclusions, and discusses related future work.

Chapter 2

Background

This chapter presents previous work in areas related to this research. A combination of the different concepts discussed were utilized in developing the classification environment presented in this thesis and for the completion of the e-textile prototype used to conduct all the experimentation.

2.1 Wearable Devices and Electronic-Textiles

Wearable devices are being used for a host of applications ranging from medical to military. Many such applications have been developed by academia and commercial vendors. Wearable devices in the medical domain can be used to promote health assessment and feedback, clinical research, and self care for individuals. Existing wearable health monitoring devices have a variety of form factors and capabilities. Wearable systems designed for health assessment and feedback are used by clinicians to assess patients' normal daily activities and provide real time feedback. Smart devices which monitor and transmit physiological information include the Wireless ECG Systems [9] and Remote Heart Monitoring [10]. Wireless

Figure 2.1: LifeVest wearable Defibrillator [13]

tele-medical networks used to monitor the patients' vital signs in real-time and dispatch emergency medical response teams in case of a medical emergency include CodeBlue [11] and Guardian Angel [12]. The LifeVest from LifeCor [13], shown in Figure 2.1, is a wearable defibrillator consisting of backpack-like straps that are worn underneath the clothing. The straps contain wires that carry signals from sensors attached to the vest and defibrillator electrodes. The data from the sensors is collected via a control box that is worn on the hip. The vest induces defibrillation shock if necessary, based on the information collected. Data from the vest can also be uploaded for delivery to a physician.

Clinical research is performed by the collection and retrieval of physiological and other forms of data in an unobtrusive and efficient manner, from the subjects' natural environment. Devices such as the SenseWear Armband developed by BodyMedia [14] have a single point of attachment (the upper arm). The reason for this choice is based on the experiments conducted by [15]. The SenseWear Armband targets wearability, ease of use, and aesthetics making it more marketable to average users. The armband collects data from a two axis accelerometer, heat flux, skin temperature, near body temperature, and galvanic skin

response. The collected data is used to calculate several metrics such as calories burned, energy expenditure, wake time, sleep duration, and physical activity duration. Krause et al. [16] utilized the body media device and the physiological data it collects to help determine the users context using unsupervised recognition algorithms.

As factors such as social isolation and low activity persist in senior citizens, the number of issues in areas such as drug administration and assistance of the elderly are growing [17]. Certain wearable applications can aid patients in taking better care of themselves by providing intelligent reminders for daily drug administration and other activities. IBM Zurich Research Laboratory has developed a mobile health tool kit which prompts patients to take their medication, if the system detects that it is overdue [18]. Other drug dispensing projects include the reconfigurable fabric project, developed at UCLA [19]. This projects primary focus is fault tolerance and drug dispensing in an ambulatory sensing garment. It uses a non-invasive technique to administer drugs through the skin.

Unlike existing wearable health monitoring devices which attempt to combine a suite of sensors into a single monolithic box or multiple box form factor, e-textiles allow sensors to be worn more comfortably and invisibly. Electronic-textiles are the natural integration of clothing and sensors, where usable information about the user's context is extracted without hampering the wearability of the garment. These textiles may or may not be wearable. The distribution of sensors throughout the surface of the body not only enables multi-point measurements such as electrocardiogram and motion classification, but also enhances privacy by blending the health monitoring devices with garments. The wearable motherboard application [20] is an example of a wearable e-textile used to monitor the vital signs of soldiers during combat situations and is shown in Figure 2.2. The tool detects gun wounds in injured soldiers when discontinuity in the optical fiber is observed. Acoustic applications such as speech processing and source separation [21] and large-scale beamforming applications [22]

Figure 2.2: The wearable motherboard used to monitor soldier's vital signs [20]

are examples of e-textiles that are not wearable. The large-scale beamforming application [22] uses acoustic sensors to compute a vehicles direction of arrival.

E-textiles can be used by fitness enthusiasts to monitor health indicators and competitive athletes aiming to maximize their performance and training. The LifeShirt [23] from Vivometrics [24] is a shirt that contains respiratory, ECG (electrocardiogram), and accelerometer sensors. An electrocardiogram is a test that measures the heart's electrical activity through electrodes placed on the skin. Other sensors can be added to the LifeShirt [23] using an additional module that provides ports for plugging in various sensors. The Sensatex athletic SmartShirt provides continuous biometric monitoring, measuring metrics such as, heart rate, respiration rate, caloric burn, and body temperature for athletes [25]. In [26] Van Laerhoven argues that embedding and distributing sensors in clothing is more practical than using strapped-on sensors, assuming many sensors providing usable information could be incorporated and interfaced in the clothing.

2.2 Context Awareness

Context awareness has been a widely researched topic for ubiquitous computing. Context is defined by [27] as any piece of information used to characterize the situation of a person, place, or object. The context space for what a person is doing can be categorized hierarchically [28]. It contains many attributes such as physiological, environmental, and social conditions. Context-aware applications are developed to intuitively deduce and interpret the context of the current situation and react appropriately. The Active Badge [29] system developed at Olivetti Research Lab is considered to be one of the first context-aware applications. In this system the office personnel wear badges that transmit IR (infra red) signals to a network of sensors placed around the building. The system is used to locate people and forward calls to the closest phone near them. The ParcTab system developed at Xerox PARC is based on palm-sized wireless ParcTab computers with an IR communication system that links them to each other and to desktop computers [30]. Many applications were developed for the ParcTab system, such as determining user location, the presence of other mobile devices, and nearby non-mobile machines. The Forget-Me-Not [31] application used ParcTabs to records where its user is, who they are with, whom they phone, and other autobiographical information, which are stored it in a database for later retrieval. The StartleCam developed by Healey et al. integrated a wearable video camera, computer, and sensing system, which stores images to a remote server whenever it detects events of interest to the wearer [32]. These events are determined by the change in skin conductivity, which can be correlated to arousal level.

This thesis explores a subset of context detection, where everyday activities performed by the user are recognized. Examples of these activities include but are not limited to walking, sitting down, standing up, and running. Activity recognition by itself provides low level information, but can be used in conjunction with other on-body physiological sensors to

enhance medical monitoring. Many health monitoring applications require that the patient keep a log of activities so that when the physiological data is analyzed off-line, the data can be correlated to what the patient was doing before or during the health events of interest. Patients often do a poor job of self-reporting, so it would be desirable to automatically annotate the physiological data with the user's activity. The creation of an activity diary automatically from sensor data eliminates the need for user input, potentially increasing accuracy and detail over a hand-written diary. Also, the log of activities can easily be translated to more useful information such as caloric energy expenditure of the user. The next section discusses the approach taken by different researchers to perform activity recognition.

2.3 Activity Recognition Methods

Some researchers have implemented basic activity recognition using machine learning techniques such as neural networks (Backpropagation, Kohonen Self - Organizing Maps), probabilistic models, and fuzzy logic. In [4], Van Laerhoven and Cakmakci used Kohonen maps to perform adaptive recognition, i.e. the system was not pre-trained but trained specifically by the user. The KSOMs and probabilistic models used to train the system also preprocess data using techniques such as standard deviation, variance, derivatives, and FFTs (Fast Fourier Transforms) before clustering the data to determine the output. The two disadvantages of the KSOM approach described are catastrophic forgetting and the curse of dimensionality. Catastrophic forgetting implies that the system initially has a high learning rate that is exhausted after some time. In order to keep learning previous values are over - written. The curse of dimensionality states that as the number of inputs to the classifier increases, so does system complexity which results in slower learning rates.

In [33], Golding used a probabilistic approach based on Bayes' rule to develop context-

awareness for an indoor navigation system. Several sensors are placed in a utility belt and are used to measure distinctive patterns in the readings to deduce location. The sensors are sampled every 50 msec (20 samples per second) after which Bayes' rule is used to calculate the user's location probability. An important result from this paper is that the error rate in predicting user location is dramatically reduced from 50 percent to two percent by preprocessing the data. The data can be preprocessed in a number of ways such as computing the mean, standard deviation, and variance of the last N samples. There is no general rule of thumb for processing data from different sensors. Also, the system has to be trained to recognize the constraints (maxima, minima) of the body. This training helps solve issues with state transitions, eliminates erroneous data, and takes more accurate absolute and relative measurements.

Randell et al. [34] applied clustering neural network algorithms to determine user context. Their work examined the validity of using a single accelerometer and minimizing power by using a very low sampling rate. The measured accuracy for several motions was roughly 75 %. In contrast to this single accelerometer approach, work by Van Laerhoven [35] incorporated a relatively large number of sensors, the majority of which were accelerometers, to determine user activity. Van Laerhoven et al. developed a harness-like system that was worn on-top of clothing and examined the performance of classification as a function of the number of sensors. The classification accuracy was also monitored as the number of contexts increased. It was determined that the performance heavily depends on the number of sensors and contexts, as well as the nature of the contexts.

In most machine learning techniques the rate of learning slows down as the number of inputs increases. It is extremely difficult to visualize and work with high-dimensional data. Researchers who work with some variation of neural networks often complain about the opacity in its behavior. Due to the opaqueness of the inner workings of neural networks the

selection of the number of inputs, hidden layers, and type of transfer functions is difficult to optimize. In addition to all the choices for neural network design, even the randomization of inputs affects the training model. Non-random selection of inputs can lead to over-training the system for a specific activity [36]. The classifier described in this thesis uses a unique linear algebraic approach to perform activity recognition that overcomes these neural network limitations. This linear algebraic approach is described in the following section.

2.4 Singular Value Decomposition

SVD is a factorization technique for rectangular matrices largely used in signal processing and pattern recognition. A non-square data matrix A of size $m \times n$ with $m > n$ can be factorized into three matrices U, S, and V using singular value decomposition as shown in Equation 2.1. Here U is an $m \times m$ matrix, S is a $m \times n$ matrix and V is an $n \times n$ matrix. S is the diagonal matrix containing all of the non-negative singular values of the original data matrix listed in descending order. U and V are orthogonal square matrices representing the left and right singular vectors for the data matrix. U represents the row space and the transpose of V represents the column space [37].

$$A = U * S * V^T \tag{2.1}$$

2.4.1 Latent Semantic Indexing

Singular Value Decomposition (SVD) is used by search engines to conduct LSI (Latent Semantic Indexing). LSI enhances the efficiency of the search by looking for an associative relationship between keywords and documents. LSI is widely researched and the process of

information retrieval through linear algebraic techniques is discussed in [38] and [39]. The beauty of the LSI approach is that it looks for hidden relationships between the keywords chosen and the documents being searched. The keyword is a word or group of words being used to find and retrieve different documents. The approach used does not just look for exact matches but tries to measure the associativity between the keywords and the documents. For example, when you search for the word *lawyer* the search engine does not just look for exact string matches of the word *lawyer* in different documents but rather looks for a group of words that can be closely associated with the word *lawyer* like attorney, judge, court, and case. The best result could be a document that does not have the string *lawyer* present in it but has many other words closely associated with the keyword *lawyer*. In this paper a variation of the approach described by Berry [38] is used to perform activity recognition. It is important to understand concepts like singular value decomposition and principal component analysis before discussing the actual classifier algorithm. These topics are discussed in the following sections.

2.5 Principal Component Analysis

Principal Component Analysis is used in conjunction with SVD to linearly transform the original high-dimensional data into a new coordinate system of reduced dimensionality [37]. PCA can be described as the orthogonal projection of data onto a vector whose direction is chosen such that the variance of the projected data is maximized. The direction and the scaling constant of the vectors denoting maximum spread in descending order are provided by the singular values and vectors in S and U. The selection of k, the dimensionality of the new coordinate system, is important and further discussed in Chapter 5.

Chapter 3

Design Guidelines for the Classification Environment

A typical recognition system as shown in Figure 3.1 has many components. A large amount of raw data is initially collected for the different activities that are to be classified. Using all of this information for activity classification is costly, complex, and inefficient. Certain features are extracted from the raw data that help best separate the activities being classified. Feature extraction of the raw data can be done through manual and latent (hidden) processes. Manual feature extraction is a means of preprocessing the data to reduce the amount of raw data being provided to the classifier. In this process statistics such as maximums and minimums are collected from the raw data. The manual feature extraction leads to the generation of activity inputs which are later processed by the classifier. Depending on the type of classifier being used, a latent feature extraction is performed by the classifier to separate the activities and to develop a set of rules to differentiate amongst these activities. Once these rules are established unknown activities are processed by the classifier and assigned a class label denoting the activity they best represent [40].

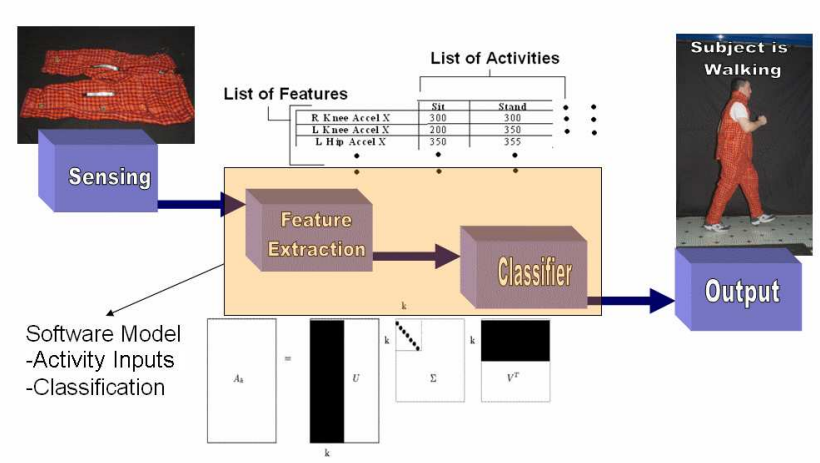

Figure 3.1: Flowchart of a typical pattern recognition system

When developing a system for a particular context application there are many design decisions. This paper is primarily concerned with the activity recognization aspect of the system, while Martin et al [41] discuss a complete design framework for wearable electronic textiles. As in [41] our goal is to generalize the design and perform minimal tuning for individuals. The SVD-based approach is an e-textile solution for which several garment sizes will be constructed. All experiments for this thesis were conducted on a pair of e-textile pants which is shown in Figure 3.2.

3.1 Final System Design

Hardware design - The data is collected from an e-textile garment which is a pair of pants currently hosting eight sensor buttons which return a total of eighteen sensor readings. Table 3.1 provides a description of the sensor button statistics. The gyroscopes return angular velocity, the piezoelectrics return applied pressure on heel strike, and the accelerometers return the accelerations for the two axes for which they are aligned. While discussing the sensor orientation in Table 3.1 it is assumed that the X axis lies in front of the user i.e. it

Figure 3.2: The e-textile pants used to collect data for all experimentation

changes as the user moves forward or backwards, the Y axis lies along the user's height, and the Z axis lies along the user's side i.e. it changes as the user turns left or right. The data collected from each sensor button is relayed over an I^2C bus to a Bluetooth module which transmits the data to a laptop or PC. The data is stored in the form of ASCII text files which are parsed to generate MATLAB m-files. The MATLAB m-files represent the data collected for each activity as a list of arrays for each sensor button.

Table 3.1: Description of the Sensors

Button Name	Button Quantity	Sampling Rate	Sensor Placement	Orientation
ADXL 311 dual-axis accelerometer [42]	2	120	Left and right side of the waist	Y and Z axis
ADXR S300 Gyroscopes [43] along with ADXL 311 [42]	4	120	Left and right side of the body, one set slightly above the knee and the other close to the ankles	X and Z axis
6" LDT-052k laminated piezoelectric film [44]	2	720	Left and right heel	none

Software Design - The software model provides a robust environment for classification of different activities as well as a good aid for exploring the design space. Although most of the experimentation is done based on the data collected from the e-textile pants, the software model can classify activities obtained through alternate data files such as C3D files [45] and data collected using the Cubix [46]. The classification environment generates inputs from the raw sensor data by pre-processing it over each channel of each sensor specified by the user. Figure 3.3 illustrates the approach taken to generate activity inputs for the classifier.

There are two important aspects involved in developing a good recognition system. The first is to obtain a set of desirable inputs by selecting the most appropriate sensing variables and

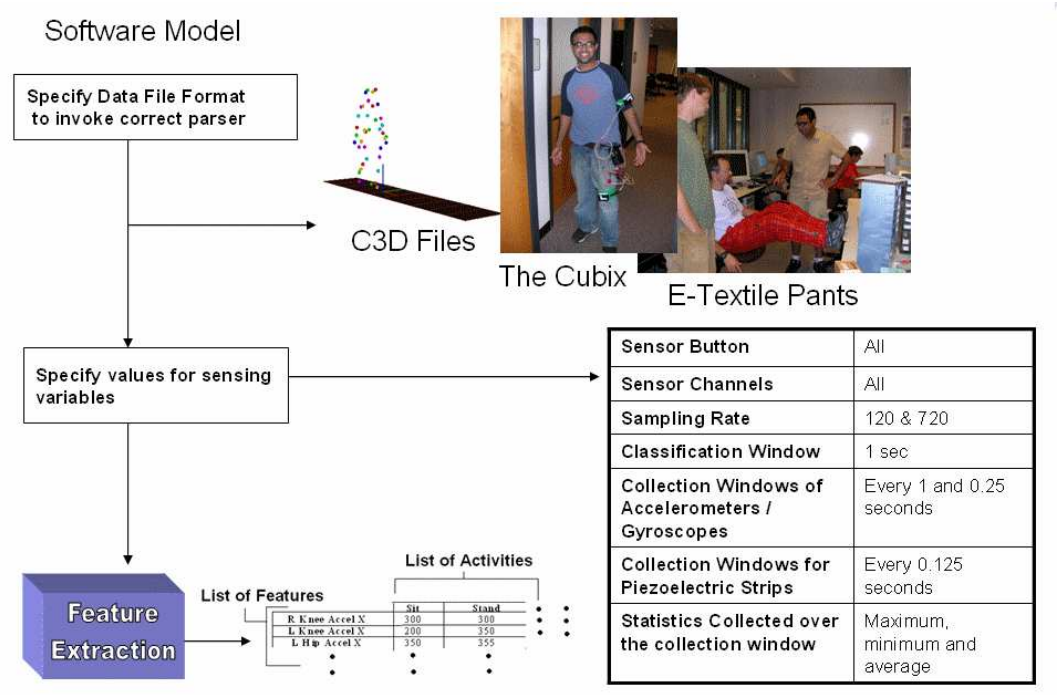

Figure 3.3: Flowchart depicting how the software model generates classifier inputs

the second is to generate rules from these inputs to obtain a high accuracy while classifying unknown inputs. The next section of this paper will examine the list of sensing variables that can be altered to generate activity inputs for the recognition system.

3.2 Factors Influencing Generation of Activity Inputs

3.2.1 Format of Data Files

As mentioned most of the experiments done to substantiate the results are performed using the e-textile pants. The classification environment can also use C3D files or data files from the Cubix for activity classification. The C3D files provide positional information for many body points and as described in [5] can be simulated to represent actual sensor readings. The e-textile pants were developed based on this simulation methodology. A large set of

C3D files needed to classify the activities explored in this paper was not available at [47]. Using the Cubix [46], activities of interest could be generated, but the Cubix could only handle four channels of information. Thus, most of the testing was done using the e-textile pants which could handle many more channels of information.

3.2.2 Classification Window Size

Clearly defining the objectives of the application is the first step towards developing a robust system. This helps in determining the required sensors for the application and the appropriate window size over which the system should collect sensor data. The classification window is the window over which data is captured for the purposes of activity input generation and classification. The window size should be as large as the activity or type of motion it is trying to recognize. For example, an elaborate dance may require a one to two minute window but a transition from the sitting to the standing position would require a very short window of less than a second to capture that specific motion. If the window size is five seconds and the sensors sampling rate is set at 120 samples per second then at least 600 samples from each sensor must be collected.

3.2.3 Sensors Utilization

The location and type of sensors are chosen based on the application requirements. In [4], Van Laerhoven performed sensor placement testing using accelerometers placed on a Velcro strap and determined that the outside of the upper-leg, just above the knee, was the best position to collect data for basic everyday activities such as walking and running. Simulations could be run to determine the best combination of sensors and locations for a particular set of activities but that is beyond the scope of this thesis [41]. Concentrating on just the number

of sensors and not on the classification algorithm or robustness of the system, one could argue that including just the sensors that pick up the characteristic aspects of a context will be sufficient and adding sensors will not improve the recognition. Sensor fusion theory [48] on the other hand shows that recognition often becomes faster and more accurate as sensors get added.

3.2.4 Sampling Rate

The sampling rate for these sensors also depends on the application. For detecting a sudden transitional change, a high sampling rate is required. A sampling rate between 20 and 120 samples per second provides a good accuracy for detection of most everyday activities (lying down, sitting, standing, walking, running). This result is further discussed in chapter 5.

3.2.5 Pre-processing Data

Most machine learning techniques usually provide a higher accuracy when handling processed data instead of raw sensor data. Pre-processing data to reduce the percentage of error in context detection is explained in [33]. Many statistics such as averages, zero crossings, and standard deviation of different window sizes within the classification window can be obtained. Raw data from different sensors at different locations has to be analyzed systematically to determine distinguishing features. In [49], Farringdon uses the difference between maximum and minimum horizontal and vertical acceleration obtained at the waist to differentiate between walking and running while [33] states the variance of Z acceleration can be used to determine if an individual is walking upstairs or downstairs. Raw data has to be analyzed to determine which pre-processing technique will create a pattern. This can be quite a cumbersome task depending on the dimensionality of the data analyzed. The other question

that arises is: *Will these data processing techniques work reliably for different people and for different sensor locations ?*

3.2.6 Statistics and Collection Windows

The processed usable information collected from the raw data is a statistic of that data set. Many different statistics can be collected over the same data set like maximums, minimums, and averages. The window (number of samples) over which these statistics are collected is the collection window. The collection window can be less than or equal to the size of the classification window. For example, if the collection window is equal to the classification window then one set of statistics will be collected over the entire classification window. If the collection window is 0.25 times the size of the classification window then four sets of statistics will be collected.

The goal of this thesis is not to describe how to find the optimal sensor selection, placement, or data processing techniques, but rather to provide the tools and software model needed to minimize erroneous output in a systematic fashion. The SVD-based approach collects data from a host of sensors and pre-processes the raw data to generate activity inputs. Distinguishable traits amongst these activity inputs are automatically found by the latent feature extraction of the classifier. The SVD algorithm not only provides the best classification for the given input, but also provides feedback on which activities are harder to separate based on the core inputs. The modification of the core sensing variables or appending new features to better distinguish between similar activities have to be determined by examining the data sets for those activities. These changes can easily be incorporated into the system. An in depth explanation of how this is done is presented in chapter 5. It should also be noted that the generation of inputs is independent from the classifier used; once the activity inputs are

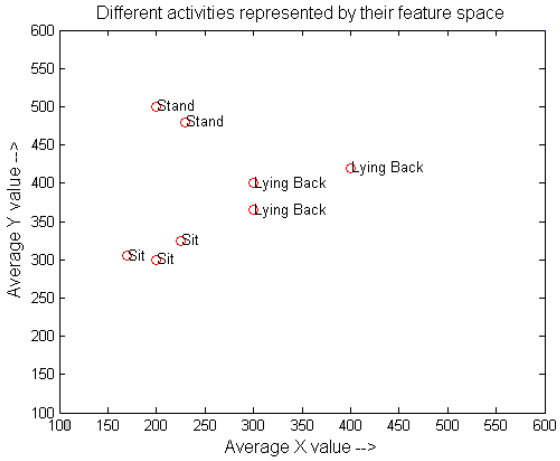

Figure 3.4: Sit, stand, and lying back being represented by their 2D feature space

generated they could be passed along to any classifier including but not limited to neural networks and SVD.

3.3 Feature Extraction Algorithm

Once the factors that influence the generation of inputs are decided an algorithm can be run to extract features for each activity. It is best to visualize each activity as a d-dimensional column vector or a d-dimensional point in space representing that activity. These d features being used to represent the activity are its feature space. The features are the list of statistics collected over different collection windows for each channel of each sensor for each activity. For example, if each activity is represented by the average X and average Y value of a sensor placed on the knee, Figure 3.4 shows its representation in a 2D coordinate system. Unfortunately most data is highly dimensional and difficult to visualize. For instance, one basic configuration of the system shown in Figure 3.3 uses 288 features (or dimensions).

Let us examine this configuration where each activity is represented by 288 features. The total feature space of an activity is the sum of the features provided by each set of sensors,

namely, the piezoelectrics, accelerometers, and the gyroscope with accelerometers. As mentioned in Figure 3.3 there are three statistics of information extracted over the classification window. When the features are collected over the classification window then only a single set of three readings (maximum, minimum, and average value) will be written to the table for each channel of each sensor. But if these pieces of information are collected over the different window sizes within the classification window then the number of readings collected will be $\sum_{i=1}^{n} \frac{Numberofsamplesperchannel}{WindowSize(i)}$, where WindowSize is an array of different collection window sizes over which the information is being collected and n is the length of the array WindowSize. The number of features provided by each sensor set can be determined by: *Classification Window * Number of sensors * Number of channels * Number of statistics * $\sum_{i=1}^{n} \frac{Numberofsamplesperchannel}{WindowSize(i)}$*.

Table 3.2 uses the above equation to calculate the number of features generated by every activity. Here the activity inputs are generated over a one second window with three statistics being collected (maximum, minimum, and average)

Table 3.2: Feature space of each activity

Sensor Set	Number of Sensors	Number of channels	Sampling Rate	Collection Windows	Features Collected
Piezoelectric	2	1	720 sps	0.125 sec	1*2*1*3*(720/90) = 48
Accelerometer	2	2	120 sps	1 and 0.25 sec	1*2*2*3*(120/120 + 120/30) = 60
Gyroscope with Accelerometer	4	3	120 sps	1 and 0.25 sec	1*4*3*3*(120/120 + 120/30) = 180
Total Feature Space of each activity					48 + 60 + 180 = 288

The size of the feature space is dependent on the sensing variables selected. These include the number of sensors, the number of channels on the sensor, the sampling rate of the sensor, the number of activity files, the number of statistics collected, the classification window size,

and the window sizes over which the statistics are collected.

3.4 Factors Influencing the SVD Classifier

The only classification variable in the SVD approach is the selection of the dimensionality k. The number of values for k that can be chosen is equivalent to the number of singular values in S. This is also equivalent to the number of activity inputs processed by the classifier. The value of k is a real positive integer between one and the number of activity inputs processed by the system. The optimal value of k is determined by running the different values of k for an activity input set against a large test set of different activities to determine which k provides minimal classification error. This test is run again using different activity input sets to determine the optimal value of k over many cases. The different tests run to determine k and the experimental results are discussed in chapter 5.

Chapter 4

The Classifier - Vector Generation and Classification

This Chapter describes the actual design of the SVD-based model. The initial offline model has two modes, vector generation and classification. Once the model determines the activity vectors for a set of activities it generates classification constants, which are used during activity classification.

The data files used for testing and activity input generation are collected by having a test subject wear the e-textile pants and perform different activities each lasting at least the size of the classification window. When the subject is performing an activity the raw data from the different sensor buttons is sent from the pants via Bluetooth to a PC which logs this information as ASCII text files. As already described this data is parsed to generate MATLAB m-files. These m-files are processed by the feature extraction algorithm once the desired sensing variable values are defined to generate activity inputs.

4.1 Vector Generation Mode

4.1.1 Creation of the Data Matrix

The data matrix columns are the list of different activities and the rows represent the statistics collected over different collection windows for each activity. Each column of the data matrix represents a single activity's feature space. The feature space for each activity is extracted from the raw data using the feature extraction algorithm as described in the previous chapter. It should be noted that the data extracted from the sensors fall within the same dynamic range and there is no need to scale the inputs. However, if the data is collected over a hetrogeneous set of sensors the inputs need to be scaled before being placed in the data matrix.

4.1.2 Singular Value Decomposition of the Data Matrix

The data matrix A_k, is an $m \times n$ matrix where m is the feature space for each activity and n is the number of activities the system is able to recognize. In equation 4.1, the matrices shown have the following sizes: A_k is $m \times n$, U_k is $m \times k$, S_k is $k \times k$, and V_k^T is $k \times n$.

$$A_k = U_k * S_k * V_k^T \tag{4.1}$$

The only free variable while factorizing the data matrix is k, which represents the number of singular values taken to reconstruct the original matrix. The variable k can be defined as the dimensional space to which the original data matrix column vectors are projected. The diagonal matrix S provides a descending list of singular values having corresponding singular vectors in U, which provide the direction of the maximum spread of the projected data. The

value of k can be chosen based on a function of the trace percent of the diagonal matrix S. The value of k can also be selected based on the classification errors resulting from running different values of k against a large set of test data.

4.1.3 Activity Vector Generation

Activity vector generation is done by taking all the column vectors Q_t of size $m \times 1$ from the original data matrix and transforming them into k-space. The transpose of each column vector is multiplied by the row space in k^{th} space (U_k) and inverse of S_k to get AV_k, a $1 \times k$ vector. This result is the projection of the original activity column vector from m-dimensional space to k-dimensional space. Equation 4.2 is used to generate activity vectors from classifier inputs.

$$AV_k = Q_t * U_k * inv(S_k) \tag{4.2}$$

Once the activity vectors are created from the set of activity inputs, unknown queries are generated in a subsequent fashion and compared to the activity vectors to correctly classify the unknown query.

4.2 Classification Mode

4.2.1 Query Vector Generation

Once the classifier projects all the activities it can recognize in k-dimensional space it is ready to accept unknown inputs and classify them. These unknown inputs are called queries. A query is a single column vector Q_t of size $m \times 1$ which matches the number and sequence of

features for any column of the original data matrix. Query vector generation is exactly the same as activity vector generation where the transpose of the query, Q_t is multiplied by the row space in k^{th} space (U_k) and inverse of S_k to get Q_k, a $1 \times k$ vector. Equation 4.3 is used to generate query vectors from unknown classifier inputs. Queries are generated from data files belonging to subjects not used to generate activity inputs.

$$QV_k = Q_t * U_k * inv(S_k) \tag{4.3}$$

4.2.2 Query Matching and Activity Recognition

The position of the query vector is compared with all the positions of the activity vectors in k-space. There are n comparisons, one for each activity. The comparison is done by finding the cosine between the query and activity vectors to determine the activity vector most similar to the generated query vector. The classifier assigns a label to the query vector denoting the activity it best represents based on the closest cosines. Figure 4.1 shows five activities (lying, sitting, standing, running, and walking) that are projected into a three-dimensional space by the classifier. A query vector for a person running (Subject1RunS) is projected into three-dimensional space by the system. The cosines between this query vector and all the activity vectors is as follows, 0.0681 (lying), 0.9998 (running), 0.1196 (sitting), 0.2172 (standing), and 0.2737 (walking). The query is determined to be the running because the angle between the query and the running activity vector is the smallest.

4.2.3 Using Different Activity Sets

A set of over 400 m-files containing data for different subjects and activities have been collected. Each activity file has a specific naming convention of $< SubjectName > - <$

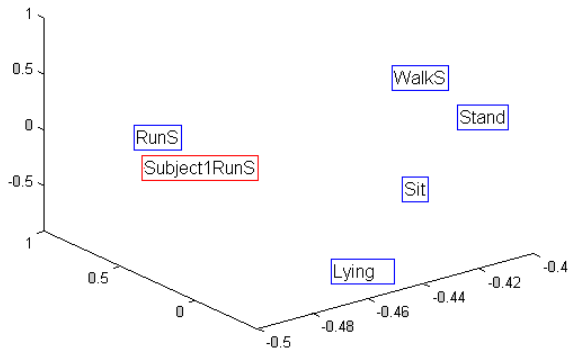

Figure 4.1: Activity - query matching in 3D space

$Typeofmotion > - < TrialNumber > . < FileExtension >$, making retrieval of files easier. Generating new activity inputs, appending inputs, or testing the list of detectable activities just involves running a script which places the desired file(s) in their respective input generation and testing folders. Different combinations of sensing variables for different activity input sets can be easily examined, resulting in an efficient means to develop and test a more robust application.

4.3 Real-time Classification

The offline model is used to create classification constant files that can be used to perform real-time activity recognition. Currently the e-textile pants broadcast information over Bluetooth which is processed by a connecting end host to determine the current user context. The activity notification has a lag of approximately the size of the classification window. Eventually a button would be designed to perform the activity detection on the wearable itself. This is discussed in Chapter 6.

4.3.1 Creation of Classification Constant Files

The classification constant files needed for online activity recognition are the list of activity vectors in k space, AQ_{vk}, and the matrix generated by multiplying the row space with the inverse of the diagonal matrix in k space, resulting in a $m \times k$ matrix defined through equation 4.4.

$$C_{us} = U_k * inv(S_k) \tag{4.4}$$

4.3.2 Online Activity Recognition

The user performs everyday activities wearing the e-textile garment. The data collected by the PC over Bluetooth is used to create a query by preprocessing the raw data and multiplying it with C_{us} every second. The query generated in real-time is compared against the activity vector list which was generated off-line to determine user activity. It can be seen that the classification algorithm is not computationally intensive after the offline vector generation process. While performing activity recognition the only actions performed are array manipulation, matrix multiplication, and array comparison. Figure 4.2 shows a flowchart describing the online activity recognition process. The process is broken into three sequential stages, data gathering, query generation, and query matching.

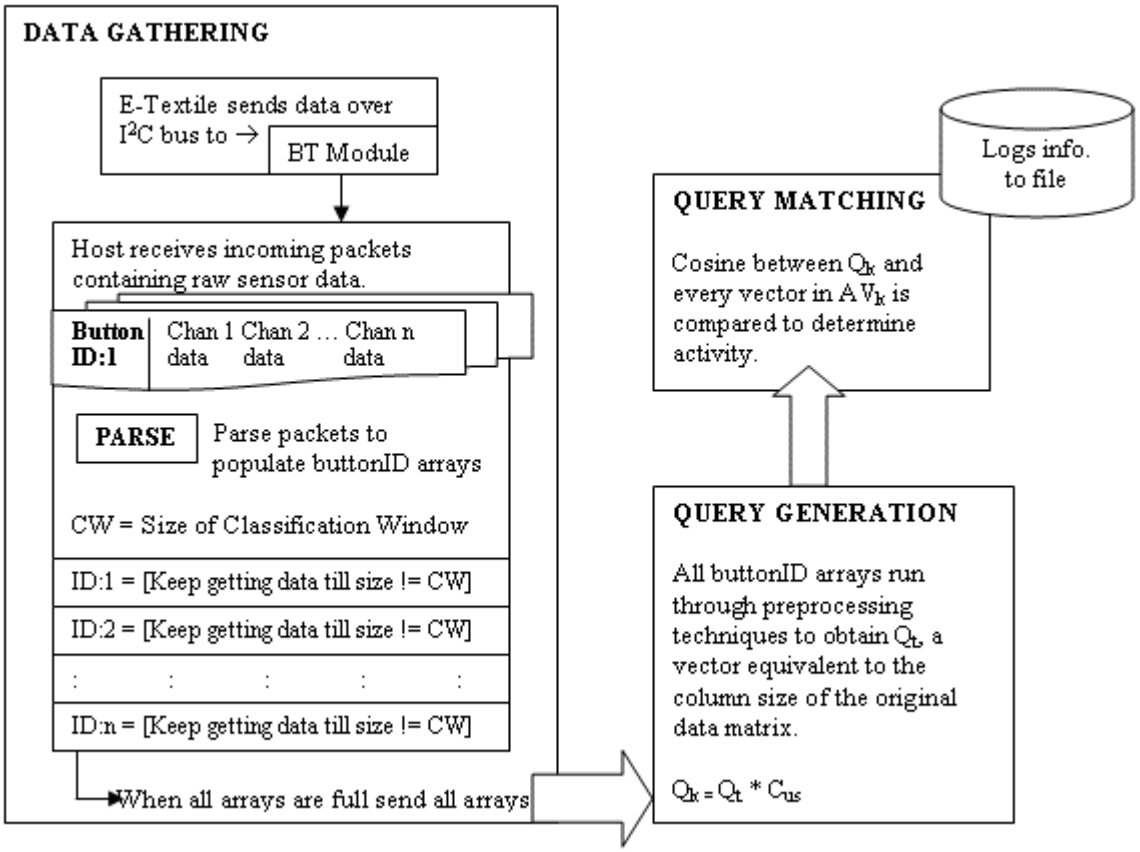

Figure 4.2: Flowchart of online activity recognition

Chapter 5

Results

This chapter summarizes all of the experimentation done to attain the minimum classification error during activity recognition. The process through which k is picked for the classifier is described. The optimal k-value is the one which results in minimum classification errors over a given set of activity inputs. Once k has been determined, the accuracy can be further improved by adjusting the classifier inputs. Based on the selection of the classification variable k and the input settings for the classifier, the optimal results for recognizing different activity sets are listed. The chapter ends by incorporating all of the knowledge gained during offline testing into a real-time system that performs activity recognition. The final version of the real-time classifier can recognize 17 different activities with a high success rate.

5.1 Selection of k Over Different Activity Sets

5.1.1 An Example for Selecting k

The classifier has only one variable, k. This variable determines the dimension of the coordinate space to which the high dimensional activity inputs will be projected. The value of k has to be selected such that it is high enough to distinguish between the different activities yet low enough to generalize well to activities that have not been presented to the classifier. The variable k can be selected from among n choices, where n is the number of activity inputs processed by the classifier. Because k can be picked from n choices, the system generates activity vectors for $k = 1$ through n and the classification errors resulting from each case are noted.

The following example shows the process of selecting k over the five root activities of sitting, standing, lying, walking, and running. Because the system receives five activity inputs, k can take on values from one through five. The recognition scheme used by the classifier assigns class labels to unknown activities based on closest cosines, i.e. an unknown query vector is projected into k-dimensional space and the cosines between the unknown query and all the projected activity inputs are calculated. The closest match is the activity having the greatest cosine or smallest angle with respect to the query vector in k-dimensional space. Figure 5.1 shows the activity vectors projected in one-, two-, and three-dimensional space. Table 5.1 shows the classification error for different values of k. The variable k can be expressed as a percentage for the trace of diagonal matrix, S. The column, *Trace percent, S(k)*, lists the corresponding trace percents for the different k values. The column *Correct* lists the number of unknown activities correctly identified by the classifier. The lowest classification errors are recorded when $k = 2$ or 3.

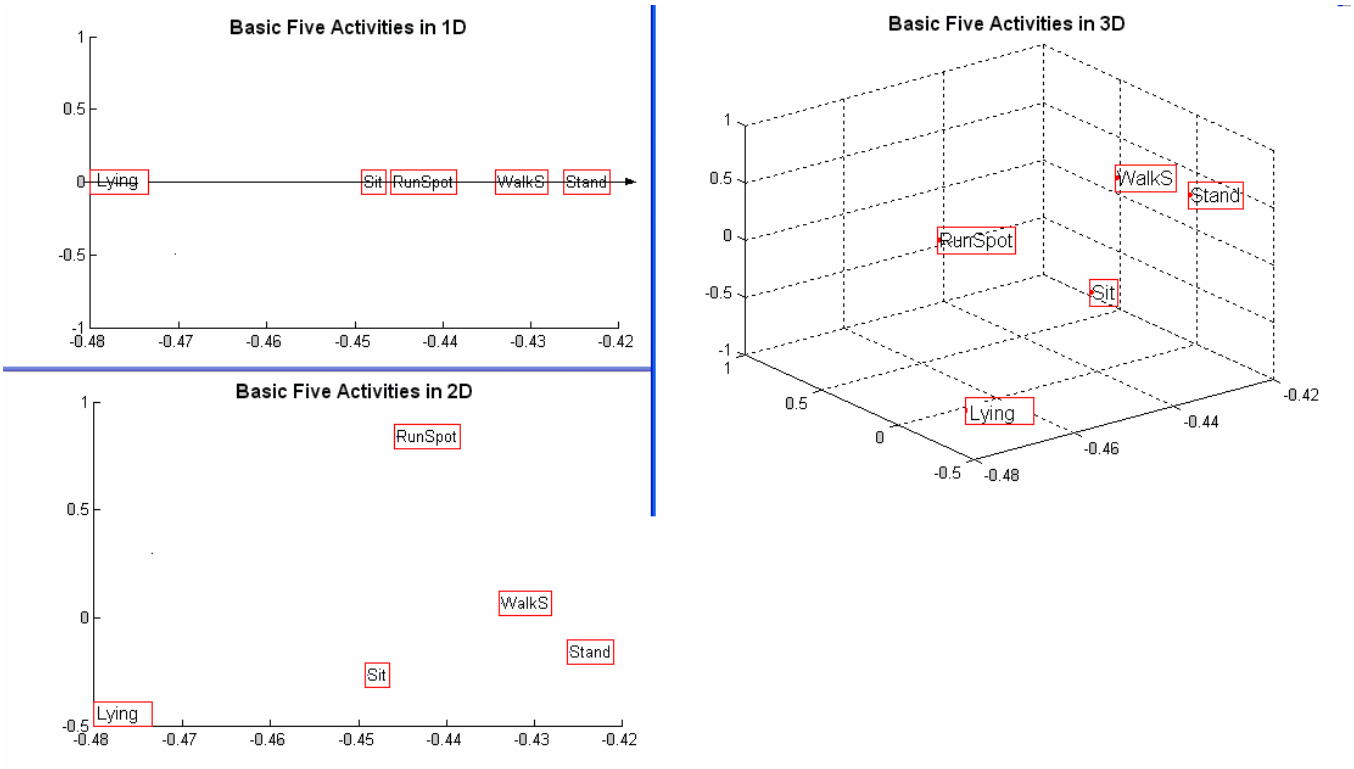

Figure 5.1: The basic five activities plotted in 1-, 2-, and 3-dimensional space

Table 5.1: Correctly classified activities over different k values, using 101 test files and a 3 second classification window

k	Trace percent, S(k)	Correct	Total correct percent
1	74.14 %	23	22.8 %
2	85.92 %	101	100 %
3	92.60 %	101	100 %
4	97.88 %	88	85.1 %
5	100 %	85	84.2 %

Table 5.2: List of activity sets and the activities they contain

Set name	List of activities
Root	Sit, stand, lie, walk and run
Sitting	Sitting tapping right foot, sitting tapping left foot, sitting with right leg over left leg, sitting with left leg over right and squatting
Lying	Lying back, lying front, lying facing the left side and lying facing the right side
Standing	Stand with left leg around right, stand leaning back, stand leaning forward, stand tapping left foot, stand with left leg raised, stand with right leg around left, stand with right leg raised and stand tapping right foot
Walking	Walking upstairs, walking downstairs, walk at a slow, medium and fast pace
Running	Jogging in place, running at a slow, medium and fast pace

5.1.2 Classification Errors over Different Activity Sets

The ability of the classifier to handle a variety of activity inputs is tested. This experiment is conducted to determine the classification errors that result over the different activity inputs presented to the classifier. The system is tested using five different subjects. The activity inputs generated from each of these subjects is tested over a number of activity file unknown to the classifier. The different sets of activities processed by the classifier are shown in Table 5.2. The final results can be seen in Table 5.3, where the system processes 5 (root set), 10 (root set + sitting subset), 13 (root set + sitting subset + lying subset), and 23 (root set + all other subsets) activities over a three second classification window. It is observed that the classification error increases as the number of recognizable activity inputs increase.

Table 5.3 lists the k values that provide the least classification error for the different activity inputs, but a record of all of the k values and corresponding classification errors is maintained. By analyzing this data it is observed that while running different activity sets over a number of activities the correct classification of activities gradually increases with k up to a point after which the accuracy falls.

Table 5.3: Optimal k values for different activity inputs

Activity input set	Number of recognizable activities	Percent Error	k	S(k)
1	5	0 %	2	85.92 %
2	5	0 %	2	86.81 %
3	5	0 %	3	93.14 %
4	5	0 %	2	86.57 %
5	5	1 %	2	87.49 %
1	10	8.2 %	6	94.60 %
2	10	3.5 %	4	89.55 %
3	10	9.12 %	7	95.57 %
4	10	3.2 %	4	87.29 %
5	10	15.5 %	5	92.31 %
1	13	5.4 %	5	89.22 %
2	13	6.5 %	6	88.14 %
3	13	10.2 %	6	87.92 %
4	13	7.2 %	7	93.16 %
5	13	12.7 %	9	96.67 %
1	23	12.3 %	13	93.76 %
2	23	11.9 %	14	94.15 %
3	23	13.7 %	16	96.82 %
4	23	8.7 %	13	93.13 %
5	23	20.4 %	16	96.63 %

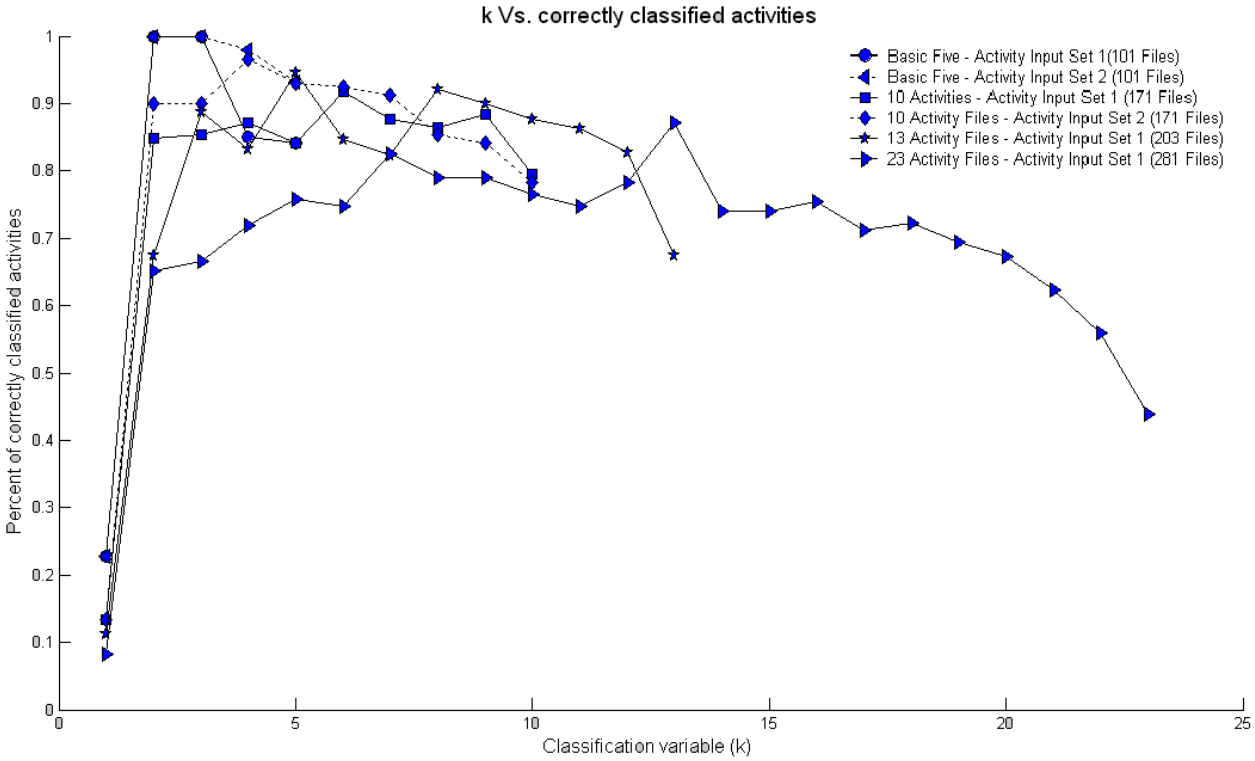

Figure 5.2: Correct classifications over k for different activity lists

5.1.3 Determining an Optimal k A Priori

The objective of the experiments conducted in this subsection is to determine whether an optimal value for k can be determined a priori without having to explore every value of k. The first approach explored is determining an optimal value of k based on the number of activity inputs processed by the classifier. As no consistent trends are observed for k as a function of n, this approach can not be used to determine k. Figure 5.2 shows the classification errors and k selection for different activity input sets.

Another approach explored is by choosing k as a percentage of the trace of S. The last column of Table 5.3, $S(k)$, lists the corresponding trace percents for the k values that provide the least classification error. Figures 5.3 and 5.4 show the correct classification percentage of activities for k as a function of S. By examining the different cases, it can be seen that the

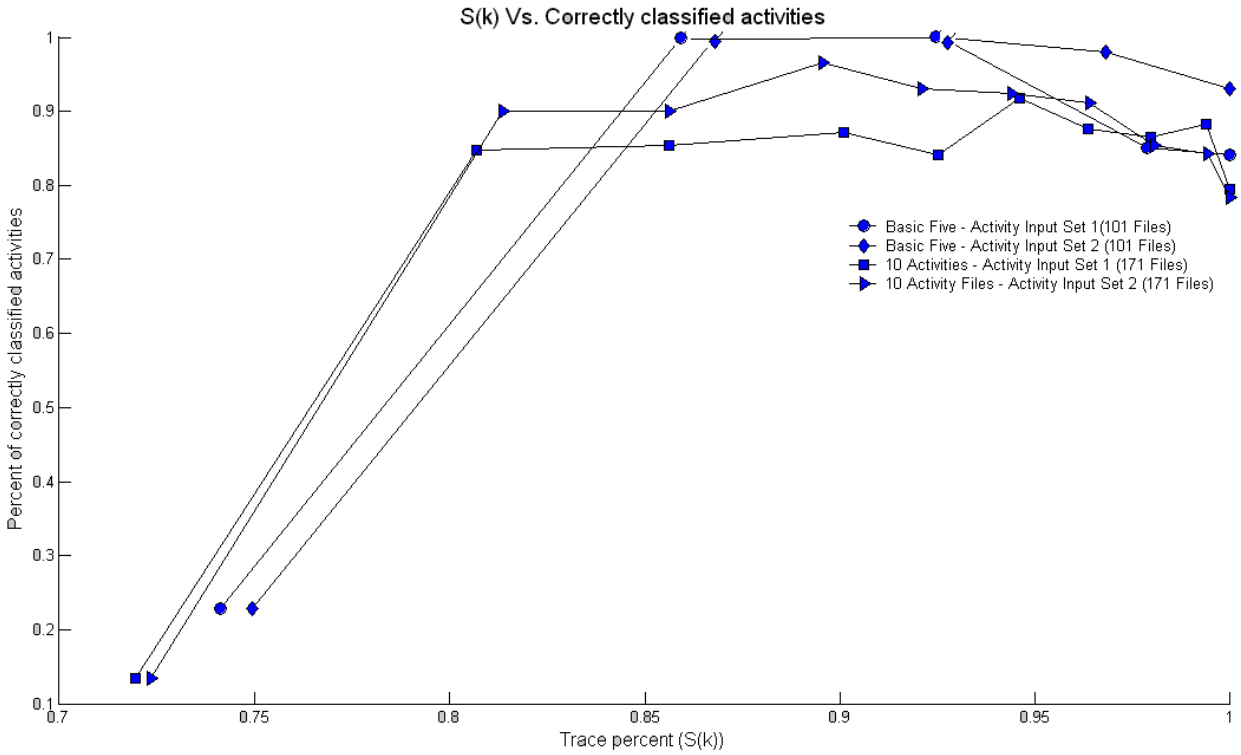

Figure 5.3: Correct classifications over k as a function of S for smaller activity lists

optimal value for k lies between 0.86 and 0.97 of the diagonal matrices trace for the different test cases.

In conclusion, it is difficult to determine the optimal value of k a priori but a range of values where the optimal k lies can be determined based on the trace of the diagonal matrix, S. Currently, the system determines the value of k by running every k to obtain which value provides the fewest number of classification errors.

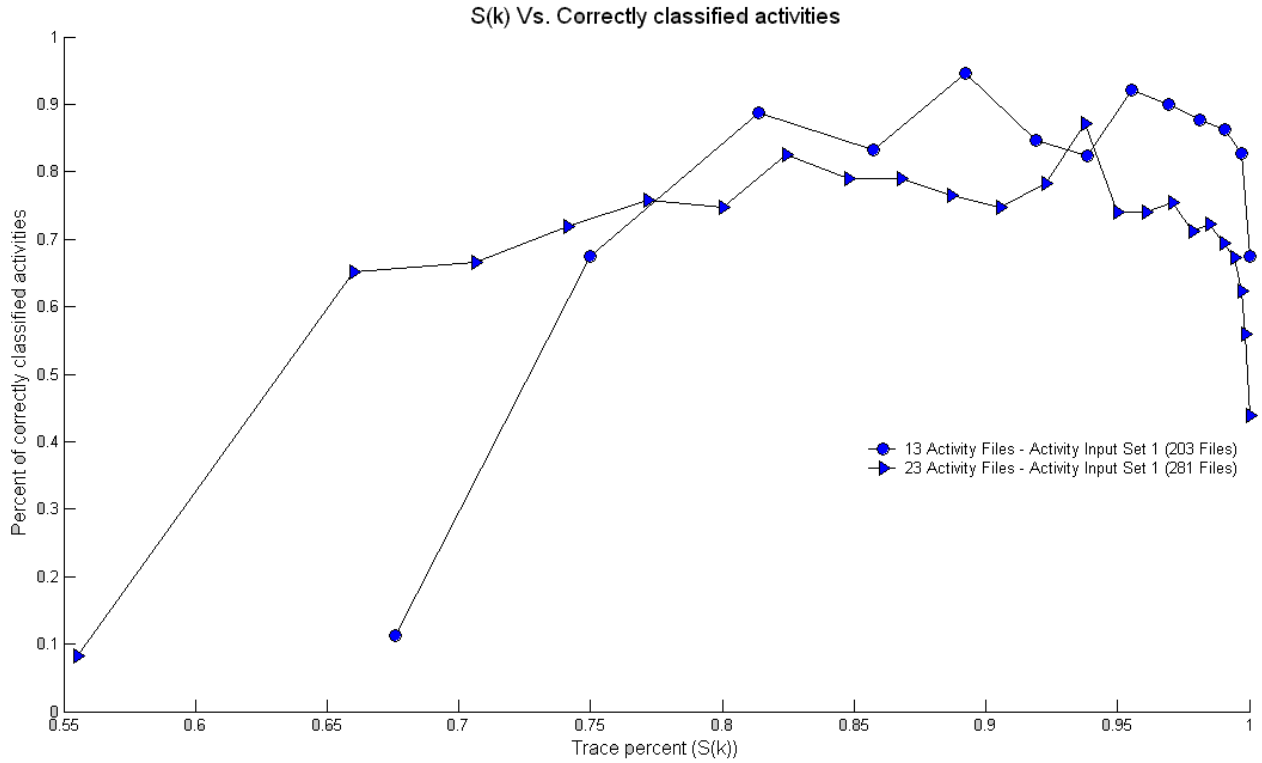

Figure 5.4: Correct classifications over k as a function of S for expanded activity lists

5.2 Change in Classification Error by Varying the Activity Inputs

The objective of this experiment was to observe the effect of different activity inputs on the classification accuracy. A number of activity files were collected over ten subjects performing the same list of activities. The subjects were separated into two disjoint sets. One set of subjects were used to generate activity inputs to the classifier and the other set of subjects were used to generate queries. The activity inputs provided to the classifier enable it to recognize fourteen activities. The list of activities include standing, standing leaning forward, standing leaning back, walking, running, and all the activities mentioned in the lying and sitting subset described in Table 5.2 except squatting. All the different input variations were tested on the same set of 158 queries.

In Table 5.4 the activity inputs were generated from a single subject using one and two instances of each activity, i.e. 14 and 28 activity inputs were presented to the classifier. When multiple activity instances from the same or different subjects are used, the classifier determines the query using two approaches, greatest cosine match and average cosine match. The greatest cosine match finds the greatest cosine or smallest angle between the query and the list of activity vectors. Thus, the activity vector resulting in the highest cosine value (closest to one) with the query, is the label assigned to the query. The average cosine method compares the average cosine value for the different instances of the same activity to the query. Table 5.5 shows the mean error and standard deviation of classifying activities over different subjects. These values are used to calculate the 90 % confidence intervals for classifying an activity. In Table 5.6 one instance of each activity to be recognized was taken from a combination of subjects, i.e. 28 and 42 activity inputs were generated. Table 5.7 shows the confidence intervals for classifying an activity when the inputs are generated from a set of

Table 5.4: Classifier accuracy for activity instances collected over the same subject

Subject Number	Misclassification over different subjects							Total Error
	Sub 4	Sub 5	Sub 6	Sub 7	Sub 8	Sub 9	Sub 10	
Subject 1								
-One Instance	4/22 (18.18%)	3/17 (17.64%)	0/16 (0%)	3/18 (16.67%)	6/53 (11.32%)	0/16 (0%)	2/16 (12.5%)	18/158 (11.39%)
-Two Instances	3/22 (13.63%)	1/17 (5.88%)	1/16 (6.25%)	1/18 (5.56%)	5/53 (9.43%)	2/16 (12.5%)	1/16 (6.25%)	14/158 (8.86%)
-Two Instances (Average Cosine Method)	2/22 (9.09%)	1/17 (5.88%)	0/16 (0%)	1/18 (5.56%)	4/53 (7.55%)	1/16 (6.25%)	1/16 (6.25%)	10/158 (6.33%)
Subject 2								
-One Instance	5/22 (22.73%)	2/17 (11.76%)	1/16 (6.25%)	4/18 (22.22%)	6/53 (11.32%)	3/16 (18.75%)	2/16 (12.5%)	23/158 (14.56%)
-Two Instances	3/22 (13.63%)	2/17 (11.76%)	1/16 (6.25%)	0/18 (0%)	6/53 (11.32%)	1/16 (6.25%)	3/16 (18.75%)	16/158 (10.13%)
-Two Instances (Average Cosine Method)	2/22 (9.09%)	2/17 (11.76%)	1/16 (6.25%)	1/18 (5.56%)	4/53 (7.55%)	2/16 (12.5%)	2/16 (12.5%)	14/158 (8.86%)
Subject 3								
-One Instance	2/22 (9.09%)	0/17 (0%)	1/16 (6.25%)	3/18 (16.67%)	3/53 (5.66%)	1/16 (6.25%)	1/16 (6.25%)	11/158 (6.96%)
-Two Instances	2/22 (9.09%)	1/17 (5.88%)	0/16 (0%)	1/18 (5.56%)	3/53 (5.66%)	1/16 (6.25%)	0/16 (0%)	8/158 (5.06%)
-Two Instances (Average Cosine Method)	2/22 (9.09%)	1/17 (5.88%)	1/16 (6.25%)	1/18 (5.56%)	4/53 (7.55%)	1/16 (6.25%)	0/16 (0%)	10/158 (6.33%)

Table 5.5: Confidence Intervals for activity instances collected over the same subject

Subject Number	Mean Error	Std. dev. of Error	90 % Confidence Intervals for correctly classifying activity
Subject 1			
-One Instance	10.9%	7.87%	(83.32% , 94.88%)
-Two Instances	8.5%	3.39%	(89.01% , 93.99%)
-Two Instances (Average Cosine Method)	5.797%	2.83%	(92.13% , 96.27%)
Subject 2			
-One Instance	15.08%	6.23%	(78.69% , 91.15%)
-Two Instances	9.71%	6.08%	(84.21% , 96.37%)
-Two Instances (Average Cosine Method)	9.32%	2.97%	(87.71% , 93.65%)
Subject 3			
-One Instance	7.17%	5%	(89.16% , 96.5%)
-Two Instances	4.63%	3.39%	(92.88% , 97.86%)
-Two Instances (Average Cosine Method)	5.8%	2.83%	(92.13% , 96.27%)

subjects instead of just one subject.

From Table 5.5 and Table 5.7 it can be observed that providing multiple instance to the classifier from a single or multiple subjects lowers the mean error. The average cosine method yields slightly lower mean errors and also lowers the standard deviation of error over the different test subjects. As the parent activity set and its variants lie closer together, the average cosine method provides better results by examining the average cosine between the multiple activity instances and query. For example, a query may have the greatest cosine match with the sitting activity vector but taking the average cosine over the multiple instances it is determined that the greatest average cosine match is actually sitting tapping the left foot. Thus, the average cosine method provides better results as the multiple instances of parent activities and their variants are clustered closer together. The classifier provides high accuracy using inputs from both multiple activity instances for a single subject and from a combination of different subjects. The best overall result is obtained by generating inputs from three subjects and classifying the query using the average cosine method. This results in a mean error of 5.77 % over the seven test subjects with a standard deviation of just 0.61 %. Thus, the classifier can accurately recognize these fourteen activities with 90 % confidence between (93.78 %, 94.68 %).

5.3 Modifying the Feature Space to Improve Results

The classifier will determine a value for k to provide optimal results for a particular set of activity inputs. To further improve classification results, the inputs being provided to the classifier can be changed by modifying the sensing variables. At this point all of the activities do not have to be examined, only the pair of activities which result in higher overall errors. Usually these activities are closely placed together in k-dimensional space i.e., the cosine

Table 5.6: Classifier accuracy for activity instances collected over a set of subjects

Subject Set	Misclassification over different subjects							Total Error
	Sub 4	Sub 5	Sub 6	Sub 7	Sub 8	Sub 9	Sub 10	
Subject 1 and 2								
-Two Instances	2/22 (9.09%)	1/17 (5.88%)	1/16 (6.25%)	0/18 (0%)	4/53 (7.55%)	3/16 (18.75%)	2/16 (12.5%)	13/158 (8.23%)
-Two Instances (Average Cosine Method)	2/22 (9.09%)	2/17 (11.64%)	1/16 (6.25%)	1/18 (5.56%)	3/53 (5.66%)	1/16 (6.25%)	2/16 (12.5%)	12/158 (7.59%)
Subject 2 and 3								
-Two Instances	2/22 (9.09%)	2/17 (11.64%)	0/16 (0%)	1/18 (5.56%)	4/53 (7.55%)	2/16 (12.5%)	3/16 (18.75%)	14/158 (8.86%)
-Two Instances (Average Cosine Method)	2/22 (9.09%)	2/17 (11.64%)	0/16 (0%)	1/18 (5.56%)	3/53 (5.66%)	2/16 (12.5%)	2/16 (12.5%)	12/158 (7.59%)
Subject 3 and 1								
-Two Instances	2/22 (9.09%)	1/17 (5.88%)	0/16 (0%)	1/18 (5.56%)	4/53 (7.55%)	2/16 (12.5%)	1/16 (6.25%)	11/158 (6.96%)
-Two Instances (Average Cosine Method)	2/22 (9.09%)	1/17 (5.88%)	0/16 (0%)	0/18 (0%)	3/53 (5.66%)	2/16 (12.5%)	1/16 (6.25%)	9/158 (5.7%)
Subject 1, 2, and 3								
-Three Instances	2/22 (9.09%)	1/17 (5.88%)	0/16 (0%)	1/18 (5.56%)	3/53 (5.66%)	2/16 (12.5%)	1/16 (6.25%)	10/158 (6.33%)
-Three Instances (Average Cosine Method)	1/22 (4.55%)	1/17 (5.88%)	1/16 (6.25%)	1/18 (5.56%)	3/53 (5.66%)	1/16 (6.25%)	1/16 (6.25%)	9/158 (5.7%)

Table 5.7: Confidence Intervals for activity instances collected over a set of subjects

Subject Number	Mean Error	Std. dev. of Error	90 % Confidence Interval for correctly classifying activity
Subject 1 and 2			
-Two Instances	8.57%	5.87%	(87.12% , 95.74%)
-Two Instances (Average Cosine Method)	8.14%	2.95%	(89.69% , 94.03%)
Subject 2 and 3			
-Two Instances	9.3%	5.9%	(86.37% , 95.03%)
-Two Instances (Average Cosine Method)	8.14%	4.66%	(88.44% , 95.28%)
Subject 3 and 1			
-Two Instances	6.69%	3.81%	(90.51% , 96.11%)
-Two Instances (Average Cosine Method)	5.63%	4.53%	(91.05% , 97.69%)
Subject 1, 2, and 3			
-Three Instances	6.42%	3.81%	(90.78% , 96.38%)
-Three Instances (Average Cosine Method)	5.77%	0.61%	(93.78% , 94.68%)

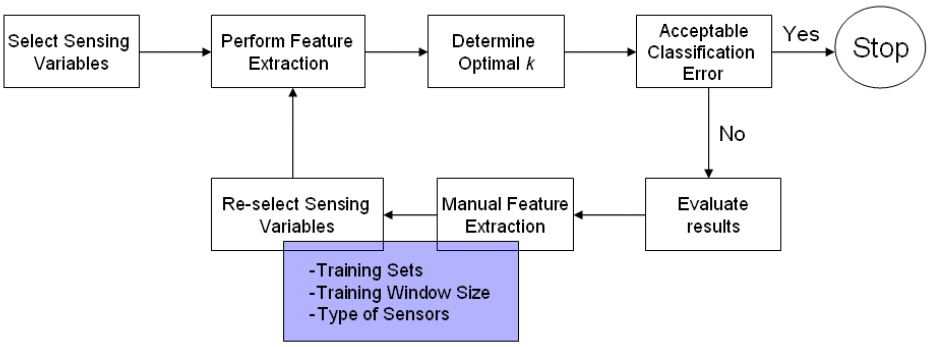

Figure 5.5: Minimize errors by choosing optimal k and readjusting sensing variables

between these activities is very high (approaching 1). Figure 5.5 illustrates the approach taken to improve results. The approach taken to collect data and arrive at these results is as follows, a set of activity inputs are passed to the classifier. These inputs are generated based on which activities are to be recognized and the choice of sensing variables. After the classifier projects these activities into k-dimensional space a large set of similar activities unknown to the classifier but known to the programmer are projected into k-dimensional space and assigned labels pertaining to which activity they best represent based on closest cosines. After each unknown activity is assigned a class label the system calculates the percentage of unknown activities that were correctly classified by the system. If the classification error is acceptable the system has achieved its goal otherwise the following provides a brief list of parameters that could be altered to improve results.

Changing the original activity inputs: It can be seen through Table 5.3 that the classification errors for different activities are dependent on the set of activity inputs. For example, enabling the system to recognize ten activities using activity input sets two and five it can be seen that input set two provides a better accuracy of 96.5 % for $k = 4$ and $S(k) = 0.8955$ whereas, the fifth input set provides an accuracy of 84.5 % for $k = 5$ and $S(k) = 0.9231$. The first way to improve accuracy is by trying a different combination of activity inputs. All the

original inputs or the inputs causing a significant error could be changed. This is primarily done to make sure the classifier receives a good set of inputs.

Changing the classification window size: Using the set of sensing variables listed in Figure 3.3 the optimal value of k resulted in a 94.06 % success rate in classifying the five root activities, where 6 of the 101 test files were misidentified. Upon further examination it was noted that all six errors resulted by misidentifying walking as running or running as walking. As mentioned only the pair of activities resulting in errors need improvement, hence, features extracted from walking and running were compared. The activity inputs generated for these activities were largely similar over a one second classification window. Increasing the classification window size from one to three seconds resulted in more distinguishable features. By regenerating the activity inputs over a three second window and running the same test set, the system could now correctly identify all 101 test files. This result corroborates the fact that it is important to set the classification window size large enough to capture enough discerning information for the activity that is to be recognized.

Providing multiple activity inputs to recognize a single activity: Multiple inputs were incorporated into the system when the subject performed the same activity at a different pace or style. While collecting the test data the subjects were asked to walk and run at what they believed to be slow, medium, and fast speeds. Because a person's gait is a unique biometric quality, activities such as walking and running were performed differently by different people. When the classifier received single activity inputs for walking and running; some errors mistaking medium to fast walking as running were observed. When the classifier processed five activity inputs (root set) and had to recognize 154 test files it resulted in a 84.42 % accuracy with 17 of the 24 misclassified activities being variants of walking and running. To improve results, the classifier received more inputs for different walks and runs at varying speeds. When the classifier was made to recognize the same test set of 154 files with nine activity

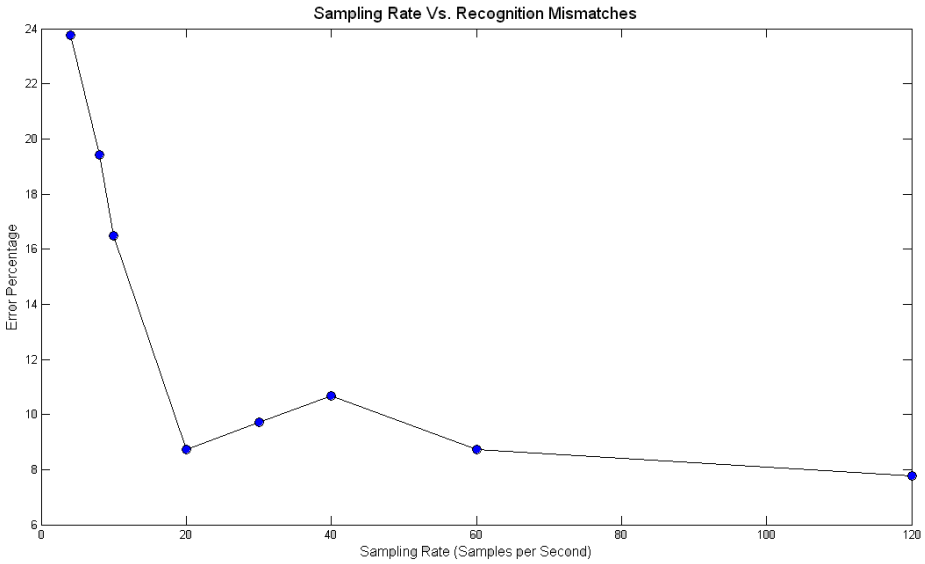

Figure 5.6: The relationship between error percent in activity detection and sampling rate.

inputs (root set + walk medium, walk fast, run medium, and run fast files) it resulted in an accuracy of 94.81 % with two of the eight misclassified activities being variants of walking and running. Currently the classifier is unable to accurately distinguish between pace and inclination of the walk and run, but it is very successful in identifying the different variants belonging to the walk and run set. The SVD classifier can recognize subtle variations in activities over a range of subjects and even some variation of the activity itself. However, as these results show once this tolerance is exceeded more inputs are needed by the classifier to maintain accuracy levels.

Sampling rate: Experiments were conducted to determine if changing the sampling rate had any affect on overall accuracy. The sampling rate for activity inputs was varied keeping all other sensing variables constant to determine classification error. The original test data collected 720 sps for the piezos and 120 sps for the gyroscopes and accelerometers. This data was down sampled to different sampling rates as shown in Table 5.8 to determine the classification error from each. The system had to classify 103 test files after receiving five

Table 5.8: The effect of sampling rate on classification error

Samples per second (sps) for all sensors	Number of misclassified activities	Percent Error
4	24	23.76 %
8	20	19.42 %
10	17	16.5 %
20	9	8.74 %
30	10	9.71 %
40	11	10.68 %
60	9	8.74 %
120	8	7.77 %

root activity inputs generated over a three second classification window with $k = 2$. It was observed that the system provided similar classification error percentages when the sampling rate of the sensors was varied between 20 to 120 sps. Figure 5.6 plots the results of Table 5.8. It can be seen that the percentage of error is not changed by more than 3 % between 20 and 120 samples per second. Changing the sampling rate between 20 and 120 sps effects the overall accuracy in a very minimal way as the statistics (maximum, minimum, average, and standard deviation) collected over the classification window to generate the activity inputs barely change.

Varying the amount of sensor information: Certain experiments were performed to determine how the classification error changes with the amount of sensor data utilized to generate inputs. Instead of generating the activity inputs from all the sensors and channels on the e-textile garment, the system receives the five root activity inputs from a few sensor buttons. At first, one sensor button on the left ankle and the other on the left hip is used which results in a classification error of 6.93 %, i.e. seven misidentified activities from among 101 test files [$k = 2$, $S(k) = 0.8598$]. Taking the sensor buttons from the right ankle and hip a 5 % classification error results, i.e. misidentifying five activities from among 101 test files [$k = 3$, $S(k) = 0.9323$]. However, using all four of these sensor buttons, i.e. left and right ankles

and hips, no errors are recorded [$k = 3$, $S(k) = 0.9298$]. Furthermore, using all eight sensor buttons on the pants to generate inputs for this test set also, results in a 100 % accuracy. Thus, classification error can be improved by generating activity inputs from more sensors only if they provide information to better represent the erroneous queries.

5.4 Longer Classification Windows and Activity Transitions

5.4.1 Longer Complex Activities: Classification Window of 12 seconds

The SVD approach has shown good results working with small classification window sizes between one and three seconds applied to simple activity motions. To check the classifier's ability to recognize longer more complex motions the classifier was presented with inputs for basic dance patterns of the waltz, the foxtrot, and the east coast swing. The steps were performed using a metronome [50] set to 60 beats a minute and 6 beats a cycle. The subjects were shown the steps as can be seen in Figure 5.7 [51] and instructed to perform each dance for two cycles of 6 beats each. Data files were collected from ten subjects that were separated into two disjoint sets. Three subjects were used to generate activity inputs and the remaining seven were used to generate queries to the classifier. The classifier was provided inputs for the fox trot, the waltz, and the east coast swing and tested over the same set of 126 queries generated from the seven test subjects. As described in Section 5.2 the classifier was tested over multiple dance instances taken from a single subject or a combination of different subjects. Thus, six and nine activity inputs were provided to the classifier and tested using both greatest cosine and average cosine match methods. Table 5.9 lists the classification

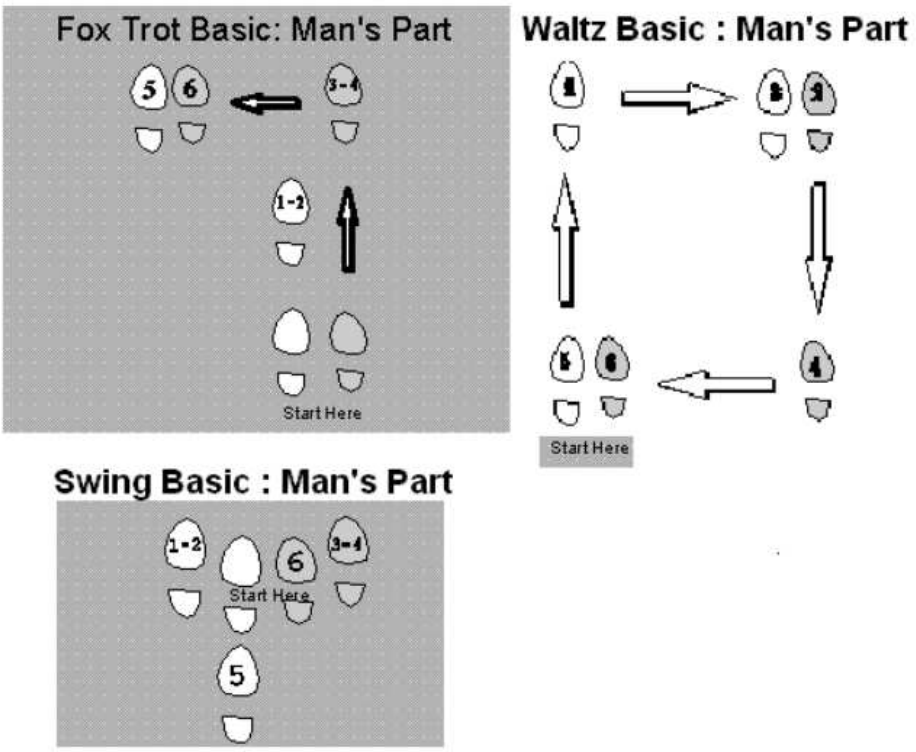

Figure 5.7: The foxtrot, the waltz, and the east coast swing dance steps [51]

error for detecting the different dances over the test subjects using multiple dance instances generated from the same subject and Table 5.10 lists the classification error over the test subjects using multiple dance instances generated from a combination of different subjects.

From Table 5.11 and Table 5.12 it can be observed that the average cosine method has a much higher mean error than the greatest cosine method. This is because unlike the previous case the multiple instances of the different dances aren't clustered close together. Also, the classifier provides higher mean error and standard deviation for multiple instances taken over the same subject. Using inputs from a combination of different subjects yields lower mean error and deviation. The best overall result is obtained by generating inputs from three subjects and classifying the query using the greatest cosine method. This results in a mean error of 10.63 % over the seven test subjects with a standard deviation of 5.59 %. Thus, the

Table 5.9: Dance classification using activity inputs from a single subject

Subject Number	Misclassification over different subjects							Total Error
	Sub 4	Sub 5	Sub 6	Sub 7	Sub 8	Sub 9	Sub 10	
Subject 1 -Two Instances	2/15 (13.33%)	4/18 (22.22%)	7/15 (46.67%)	12/30 (40%)	5/15 (33.33%)	2/15 (13.33%)	5/18 (27.78%)	37/126 (29.36%)
-Two Instances (Average Cosine Method)	5/15 (33.33%)	5/18 (27.78%)	3/15 (20%)	15/30 (50%)	2/15 (13.33%)	3/15 (20%)	6/18 (33.33%)	39/126 (30.95%)
Subject 2 -Two Instances	7/15 (46.67%)	10/18 (55.56%)	8/15 (53.33%)	5/30 (16.67%)	9/15 (60%)	7/15 (46.67%)	6/18 (33.33%)	52/126 (41.27%)
-Two Instances (Average Cosine Method)	12/15 (80%)	15/18 (83.33%)	10/15 (66.67%)	19/30 (63.33%)	9/15 (60%)	11/15 (73.33%)	10/18 (55.56%)	86/126 (68.25%)
Subject 3 -Two Instances	4/15 (26.67%)	10/18 (55.56%)	9/15 (60%)	13/30 (43.33%)	4/15 (26.67%)	9/15 (60%)	4/18 (22.22%)	53/126 (42.06%)
-Two Instances (Average Cosine Method)	11/15 (73.33%)	10/18 (55.56%)	14/15 (93.33%)	27/30 (90%)	14/15 (93.33%)	8/15 (53.33%)	12/18 (66.67%)	96/126 (76.19%)

Table 5.10: Dance classification using inputs from a combination of subjects

Subject Set	Misclassification over different subjects							Total Error
	Sub 4	Sub 5	Sub 6	Sub 7	Sub 8	Sub 9	Sub 10	
Subject 1 and 2								
-Two Instances	2/15 (13.33%)	2/18 (11.11%)	4/15 (26.67%)	7/30 (23.33%)	3/15 (20%)	2/15 (13.33%)	3/18 (16.67%)	23/126 (18.25%)
-Two Instances (Average Cosine Method)	11/15 (73.33%)	17/18 (94.44%)	14/15 (93.33%)	3/30 (10%)	2/15 (13.33%)	10/15 (66.67%)	2/18 (11.11%)	61/126 (48.41%)
Subject 2 and 3								
-Two Instances	4/15 (26.67%)	7/18 (38.89%)	5/15 (33.33%)	9/30 (30%)	3/15 (20%)	5/15 (33.33%)	3/18 (16.67%)	36/126 (28.57%)
-Two Instances (Average Cosine Method)	12/15 (80%)	16/18 (88.89%)	10/15 (66.67%)	27/30 (90%)	12/15 (80%)	12/15 (80%)	11/18 (61.11%)	100/126 (79.37%)
Subject 3 and 1								
-Two Instances	2/15 (13.33%)	5/18 (27.77%)	4/15 (26.67%)	6/30 (20%)	3/15 (20%)	5/15 (33.33%)	6/18 (33.33%)	31/126 (24.60%)
-Two Instances (Average Cosine Method)	6/15 (40%)	4/18 (22.22%)	11/15 (73.33%)	22/30 (73.33%)	12/15 (80%)	6/15 (40%)	14/18 (77.78%)	75/126 (59.52%)
Subject 1, 2, and 3								
-Three Instances	2/15 (13.33%)	3/18 (16.67%)	1/15 (6.67%)	4/30 (13.33%)	2/15 (13.33%)	0/15 (0%)	2/18 (11.11%)	14/126 (11.11%)
-Three Instances (Average Cosine Method)	10/15 (66.67%)	10/18 (55.56%)	13/15 (86.67%)	24/30 (80%)	12/15 (80%)	12/15 (80%)	12/18 (66.67%)	93/126 (73.81%)

Table 5.11: Confidence Intervals for different dances collected over the same subject

Subject Number	Mean Error	Std. dev. of Error	90 % Confidence Intervals for correctly classifying activity
Subject 1			
-Two Instances	28.09%	12.81%	(62.5% , 81.32%)
-Two Instances (Average Cosine Method)	28.25%	12.14%	(62.83% , 80.67%)
Subject 2			
-Two Instances	44.60%	15%	(44.38% , 66.42%)
-Two Instances (Average Cosine Method)	68.89%	10.36%	(23.5% , 38.72%)
Subject 3			
-Two Instances	42.06%	16.81%	(45.60% , 70.28%)
-Two Instances (Average Cosine Method)	75.08%	10.92%	(18.33% , 34.37%)

classifier can accurately recognize the foxtrot, the waltz, and the swing with 90 % confidence between (85.27 %, 93.47 %). Although these results for longer complex activities have been successful, the system has to be tested for a greater number of more complex activities to validate the robustness of the SVD approach for such activities.

5.4.2 Activity Transitions

Most data files collected for testing and activity input generation contained a classification window worth of information for only a single activity. Some data files were collected having the subject perform a range of different activities over a long period of time. Initially the classification algorithm used a shifting window approach to classify a test file containing a number of activities. This was done by sequentially processing a new classification window worth of data everytime while classifying the activity. For example, using this approach

Table 5.12: The confidence intervals calculated for inputs generated from a group of subjects

Subject Number	Mean Error	Std. dev. of Error	90 % Confidence Interval for correctly classifying activity
Subject 1 and 2			
-Two Instances	17.78%	5.77%	(77.98% , 86.46%)
-Two Instances (Average Cosine Method)	51.74%	38.97%	(19.64% , 76.88%)
Subject 2 and 3			
-Two Instances	28.41%	7.88%	(65.81% , 77.37%)
-Two Instances (Average Cosine Method)	78.1%	10.71%	(14.03% , 29.77%)
Subject 3 and 1			
-Two Instances	24.92%	7.48%	(69.59% , 80.57%)
-Two Instances (Average Cosine Method)	58.09%	23.36%	(24.76% , 59.06%)
Subject 1, 2, and 3			
-Three Instances	10.63%	5.59%	(85.27% , 93.47%)
-Three Instances (Average Cosine Method)	73.65%	10.92%	(18.33% , 34.37%)

with a three second classification window over a 27 second test file would result in nine classifications. Figure 5.8 shows a subject transitioning through a series of activities which are classified using the shifting window approach. The problem with this approach is that multiple activities could overlap within the same classification window and the predominate activity would be the label assigned to the unknown vector. Thus, the classifier tries to make the best match based on the information it receives rather than depicting the true nature of activity transitions. To get a better idea of when a subject starts transitioning in and out of different activities a sliding window approach is used. Here the classifier does not process a new classification window worth of information everytime but moves the window every $1/10^{th}$ of a second after processing the first classification window. Using this approach with a three second classification window over a 27 second test file would result in 241 classifications. Figure 5.9 depicts the same activity transitions as Figure 5.8 using the sliding window approach. This approach is more fine grained in terms of determining when a subject transitions between different activities. In Figure 5.9 when a subject transitions from sitting to standing, the query vector has strong elements of both activities; as one activity ends and the other begins the query vector starts to develop a stronger match to the activity it is transitioning.

In Figure 5.10 the plot captures a 40 second window in which the subject starts out standing and then transitions to sitting. After sitting for a few seconds the subject transitions back to standing and then sits again. While the subject is sitting he taps his left leg stops and then taps his right leg. Next, the subject gets up and walks around the lab stopping intermittently. The end of the plot is particularly interesting as the shifting window approach shows the subject as predominately walking but the sliding window approach shown in Figure 5.10 shows a more accurate record. The plots generated using the sliding window approach are interesting on many levels as they show the exact cross over points when subject transition in

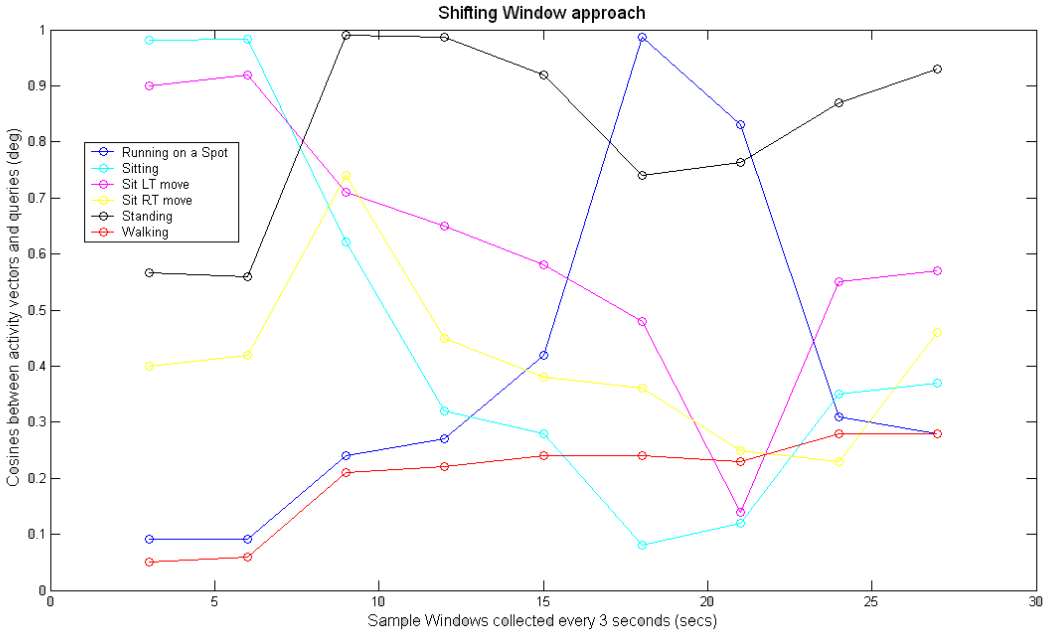

Figure 5.8: Shifting window approach to classify subject transitions (sit-stand-run-stand)

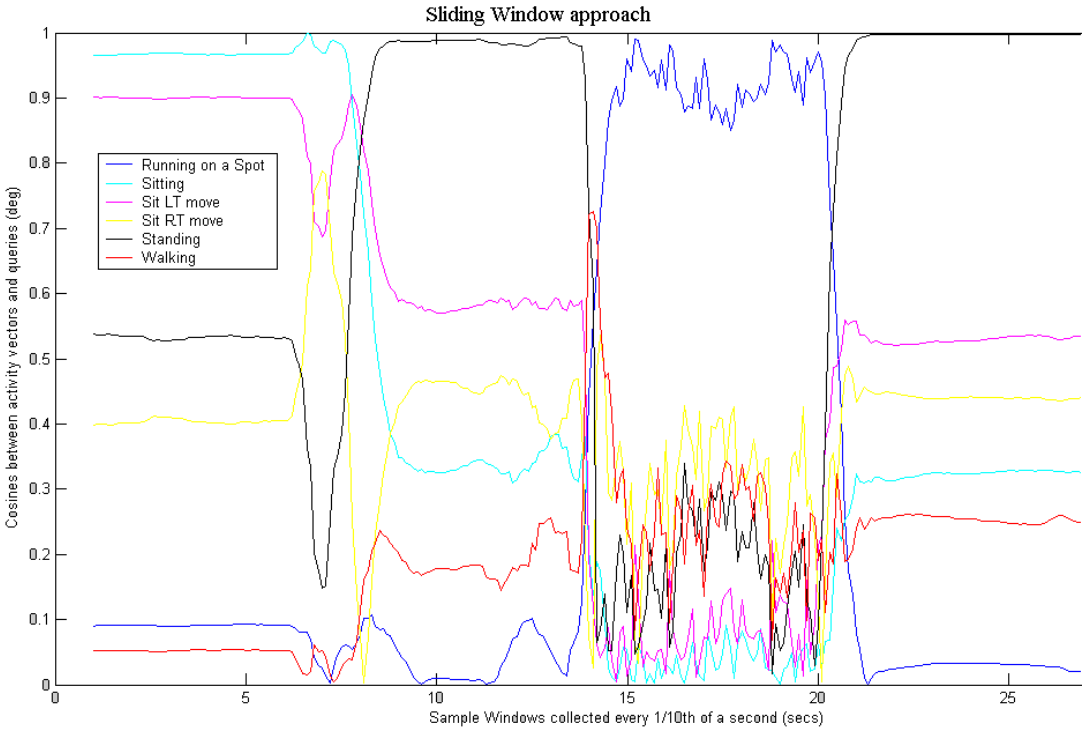

Figure 5.9: Sliding window approach to classify subject transitions (sit-stand-run-stand)

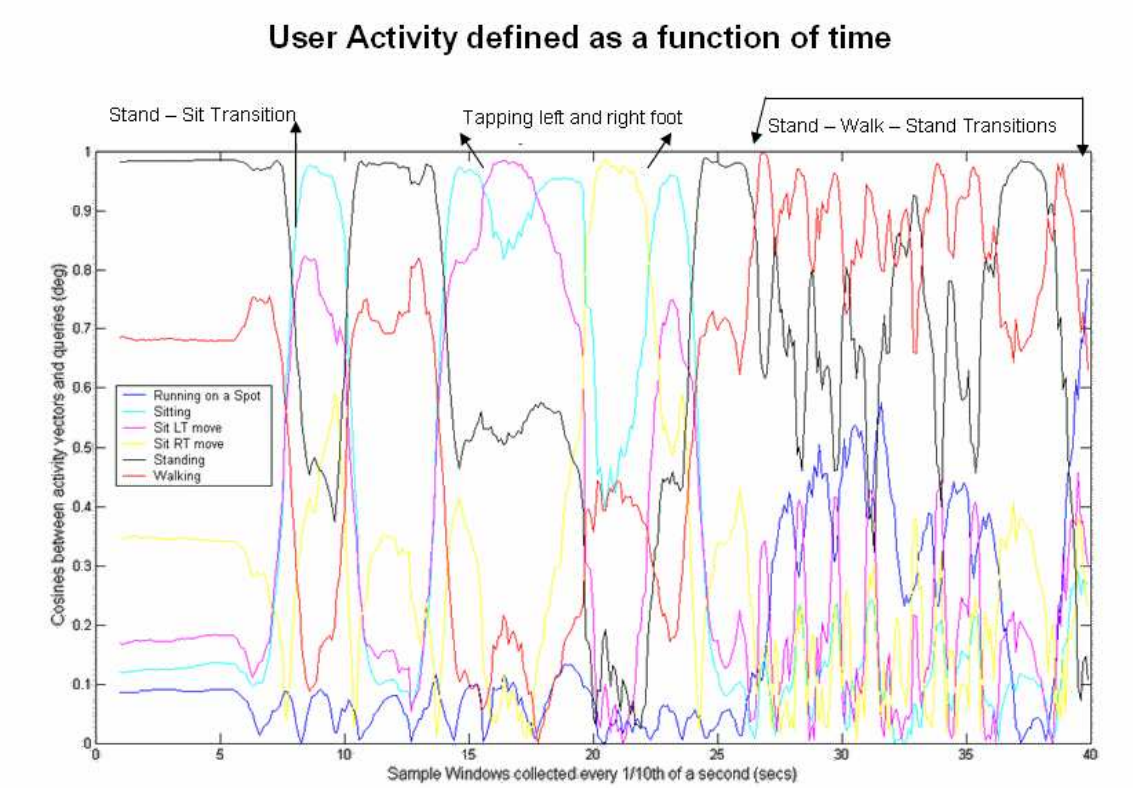

Figure 5.10: The subject transitions between these activities: stand-sit-stand-sit-sit tap left leg-sit-sit tap right leg-sit-stand-walk-stand

and out of different activities, as well as the effect one activity has on the others. The shifting window approach is a more coarse approach in determining activity transition whereas the sliding window approach is more fine grained.

5.5 Final Online Results for 17 Activities

After investigating the offline results it was desired to see how the SVD classifier would perform while detecting user activity in real-time over a three second classification window. The test data was collected over a range of people who could comfortably fit into the pants. The height range of these people was between 5'4" and 6'2", and the waist size varied between

30 to 40 inches. The pants themselves measured 35 inches in length with an elastic waist. Test data was collected for 10 subjects (8 male, 2 female) over a range of different activities. The subjects were asked to perform a series of different activities and the response of the classifier was noted every three seconds, i.e. whether the classifier was successful in identifying the activity performed by the subject or not. A time-stamped log was maintained that recorded what activity the subjects performed and for how long. The data files generated from these sessions were also analyzed offline for any anomalies but none were detected. Hence, as expected the online results were identical to the offline analysis for the same test files.

Twenty six activity inputs collected from a random subject and were used by the classifier to determine 17 activities (output states) as shown in Figure 5.11. The random subject selected was subject number five. The five basic hierarchical subsets of the activity list, lying, sitting, standing, walking, and running were determined along with most of their variants. Table 5.2 shows the list of root activity variants. The different variants of the lying, standing, and sitting subsets are identified among the 17 output states. The different variants of walking, and running are identified as their parent class. This is due to the fact that during testing the subjects are asked to run or walk at the paces they perceive to be slow, medium, or fast. To successfully distinguish between walk and run variants a more controlled environment for testing is needed.

The aggregate error percent is just 6 % for all mistakes and 2.5 % for major mistakes. The mistakes in activity detection are classified into major and minor mistakes, where minor mistakes involves misidentifying the activity but correctly identifying the activity subset. Major mistakes involve misidentifying the activity as well as the activity set. For example, a minor mistake is detecting the user as sitting tapping the right foot, when the left foot is really being tapped. A major mistake is detecting the user as lying on the front when the user is actually walking. Table 5.13 and 5.14 show the subjects and error percents involved

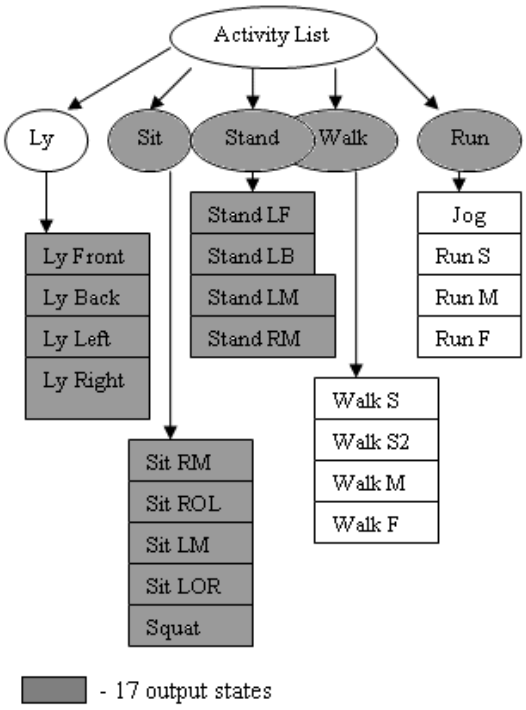

Figure 5.11: List of 26 activity inputs and 17 outputs

for the different activity sets. The classifier could recognize activities based on information collected from subject five, who was five feet eight inches with a waist size of 32 inches and could comfortably fit into the pants. It was seen that the major mistakes that occurred where for people with a large waist size (40 inches) resulting in actual sitting queries to be labeled as lying. Also, people who were five feet four inches had some walking queries labelled as standing. However, the overall results show that the high accuracy obtained during offline testing and the approaches used to obtain the k value and modify the inputs hold in a real-time setting. This approach is used to generate a time-stamped log that records what the subject was doing and for how long.

Table 5.13: Percent errors of individual activity sets

Number of activity files checked	242
Number of mismatches for 18 activities	14
Lying Subset Percent error	0.83 %
Sitting Subset Percent error	1.24 %
Standing Subset Percent error	2.07 %
Walking Percent error	1.65 %
Running Percent error	0 %
Aggregate percent error	5.78 %

Table 5.14: Various subjects activity files and error percents

Subject No.	Subject height	Subject waist	Major Errors	Minor Errors	Total files
1 (Female)	5”4’	30 inches	1	1	18
2 (Male)	5”10’	32 inches	0	0	18
3 (Male)	5”9’	32 inches	0	1	18
4 (Female)	5”4’	32 inches	2	0	18
5 (Male)	5”8’	32 inches	0	1	37
6 (Male)	5”10’	32 inches	0	2	42
7 (Male)	6”1’	40 inches	2	2	16
8 (Male)	6”2’	34 inches	0	1	22
9 (Male)	5”11’	32 inches	0	0	38
10 (Male)	5”9’	34 inches	0	1	15

Chapter 6

Conclusions

The goal of this thesis in developing a robust classifier and classification environment was achieved. As discussed, the SVD approach reduced the design space by eliminating variables in the classification model. The SVD classifier has a sole variable k which was determined through an exhaustive search and would provide optimal solutions for a given set of activity inputs over a set of test data. Working with raw sensor data causes high error rates so basic pre-processing techniques such as finding the maximums, minimums, averages, and standard deviation is performed on the raw data to generate inputs for the classifier. The classifier itself has the ability to perform latent feature extraction where it automatically finds distinguishing characteristics between the different activities during matrix factorization.

The classification environment is set up in such a way that the sensing variables can be easily modified or appended to change inputs going into the classifier. The inputs are examined and modified only when the classifier fails to meet an acceptable classification accuracy. The SVD approach works well not only for the subject used to generate the activity inputs but exhibits good results when tested on other users. The SVD approach has been extensively tested over multiple subjects, for long durations of time, and over a large set of activities. The classifier

has shown high accuracy throughout the trials. The classifier is used to perform activity detection in real-time as well as process information offline. The real-time classification is used to generate a time-stamped log that records what the subject was doing and for how long. All experiments and results obtained through the classification environment and online testing are exactly repeatable.

While examining the activity sets it was noted that a single root activity can have different variants. When recognizing an activity it should be known whether the different variants of an activity are to be recognized or if all the variants are to be recognized as the same root activity. The SVD approach works for both of these scenarios. After rigorously testing the classifier over many scenarios it can be stated that the classification environment and associated results show that the goals stated for this thesis were satisfied.

6.1 Future Work

It is recommended that the future work done in this area begin by expanding the activity list while still retaining accuracy, by adding more sensors to a matching e-textile vest along with the pants. By doing this the system can learn to recognize a more complex set of motions. Physiological sensors are to be integrated into the e-textile to test the wearable as an aid to medical monitoring. Integrating information provided by different kinds of sensors will expand the list of activities that can be classified by the system and make it more aware of user context rather than just being limited to activity recognition. Once the SVD factorization has been done offline to generate the classification constants, the classifier utilizes these constants to perform activity detection. The classifier can be implemented on the wearable itself as the actual classification of activities requires simple array manipulation and comparison which can be performed by an adequate microprocessor.

Bibliography

[1] Ubimon, "Ubiquitous Monitoring Environment for Wearable and Implantable Sensors," 2005. http://www.ubicare.org/projects-ubimon.shtml.

[2] K. Kukkonen, T. Vuorela, J. Rantanen, O. Ryynnen, A. Siili, and J. Vanhala, "The design and implementation of electrically heated clothing," in *Proceedings of the Fifth International Symposium on Wearable Computers*, pp. 77–83, ISWC 2001.

[3] B. Clarkson, K. Mase, and A. Pentland, "Recognizing user context via wearable sensors," in *Proceedings of the Fourth International Symposium on Wearable Computers*, pp. 69–75, ISWC 2000.

[4] K. VanLaerhoven and O. Cakmakci, "What shall we teach our pants?," in *Proceedings of the Fourth International Symposium on Wearable Computers*, pp. 77–83, ISWC 2000.

[5] J. N. Edmison, "Electronic Textiles for Motion Analysis," Master's thesis, Virginia Polytechnic Institue and State University, 2004.

[6] R. Andrews, J. Diederich, and A. Tickle, "A survey and critique of techniques for extracting rules from trained artificial neural networks," in *Knowledge Based Systems*, pp. 187–202, 1995.

[7] Mathworks, "MATLAB Neural Network Toolbox," 2006. http://www.mathworks.com/products/neuralnet/.

[8] S. Salzberg, "On comparing classifiers: Pitfalls to avoid and a recommended approach, data mining and knowledge discovery," in *Data Mining and Knowledge Discovery*, pp. 317–328, Kluwer Academic Publishers, Boston, 1997.

[9] GMP Wireless Medicine Inc, 2005. http://www.wirelessecg.com/.

[10] Health Frontier Inc, 2005. http://www.healthfrontier.com/.

[11] D. Malan et al, "Codeblue: An ad hoc sensor network infrastructure for emergency medical care," in *Proceedings of the MobiSys 2004 Workshop on Applications of Mobile Embedded Systems (WAMES 2004)*, pp. 12–14.

[12] G. Bell, "The body electric," *Commun. ACM.*, vol. 40(2), pp. 31–32, 1997.

[13] LifeCor Inc, "Life Vest," 2006. http://www.lifecor.com/.

[14] Body Media, "SenseWear Armband," 2005. http://www.bodymedia.com/index.jsp.

[15] S. Gemperle, C. Kasabach et al, "Design for wearability," in *Proceedings of the Second International Symposium on Wearable Computers*, ISWC 1998.

[16] A. Krause, A. Smailagic, D. Siewiorek, and J. Farringdon, "Unsupervised, dynamic identification of physiological and activity context in wearable computing," in *Proceedings of the Seventh International Symposium on Wearable Computers*, pp. 88–97, ISWC 2003.

[17] Intel, "Wireless Research," 2005. http://www.intel.com/research/exploratory/.

[18] IBM Zurich Research Lab, "Mobile Health Toolkit," 2005. http://www.zurich.ibm.com/mobilehealth/.

[19] F.-C. Chan, F. Dabiri, R. Jafari, E. Kursun, V. Raghunathan, T. Schoellhammer,D. Sievers, D. Estrin, G. Reinman, M. Sarrafzadeh, M. Srivastava, B. Wu, and Y. Yang,

"Reconfigurable fabric: An enabling technology for pervasive medical monitoring," *In Proceedings of Communication Networks and Distributed Systems: Modeling and Simulation*, pp. 24–30, January 2004.

[20] S. Park, C. Gopalsamy, R. Rajamanickam, and S. Jayaraman, "The wearable motherboard: An information infrastructure or sensate liner for medical applications," *Studies in Health Technology and Informatics, IOS Press*, vol. 62, pp. 252–258, 1999.

[21] R. Shenoy, "Design of e-textiles for acoustic applications," Master's thesis, Virginia Polytechnic Institue and State University, 2003.

[22] Z. S. Nakad, "Architectures for e-Textiles." Ph.D. Dissertation, Virginia Polytechnic Institue and State University, 2003.

[23] P. Grossman, "The lifeshirt: A multifunctional ambulatory system that monitors health, disease, and medical intervention in the real world," in *International Workshop on New Generation of Wearable Systems for eHealth*, pp. 73–80, December 2003.

[24] Vivometrics Inc., "Life Shirt," 2005. http://www.vivometrics.com/.

[25] Sensatex, "Smart Shirt," 2005. http://www.sensatex.com/smartshirt/index.html.

[26] K. VanLaerhoven and H. W. Gellersen, "Spine versus porcupine: a study in distributed wearable activity recognition," in *Proceedings of the Eighth International Symposium on Wearable Computers*, pp. 142–150, ISWC 2004.

[27] A. Dey and G. Abowd, "Towards a better understanding of context and context-awareness," Tech. Rep. GIT-GVU-99-22, VU Technical Report, 1995.

[28] A. Schmidt, M. Beigl, and H. W. Gellersen, "There is more to context than location," in *Interactive Applications of Mobile Computing (IMC)*, Nov 1998. Reprinted in Computer and Graphics Journal, Elsevier, Vol. 23, No. 6, Dec 1999, pp. 893-902.

[29] R. Want, A. Hopper, and J. Gibbons, "The active badge location system," vol. 10(1), pp. 91–102, 1992.

[30] R. W. et al., "An overview of the parctab ubiquitous computing environment," vol. 2(6), pp. 28–43, 1995.

[31] B. J. Rhodes, "The wearable remembrance agent: a system for augmented memory," in *Proceedings of the First International Symposium on Wearable Computers*, pp. 123–128, ISWC 1997.

[32] J. Healey and R. W. Picard, "Startlecam: A cybernetic wearable camera," in *Proceedings of the Second International Symposium on Wearable Computers*, pp. 42–49, ISWC 1998.

[33] A. Golding and N. Lesh, "Indoor Navigation Using a Diverse Set of Cheap Wearable Sensors," in *Proceedings of the Third International Symposium on Wearable Computers*, pp. 29–36, ISWC 1999.

[34] C. Randell and H. Muller, "Context awareness by analysing accelerometer data," in *Proceedings of the Fourth International Symposium on Wearable Computers*, pp. 175–176, ISWC 2000.

[35] K. VanLaerhoven, H. Gellersen, and A. Schmidt, "Multi-sensor context aware clothing," in *Proceedings of the Sixth International Symposium on Wearable Computers*, pp. 49–57, ISWC 2002.

[36] Mathworks, "MATLAB Help," 2006. http://www.mathworks.com/.

[37] D. C. Lay, *Linear Algebra and its appications, second edition update.* Addison-Wesley, 1996.

[38] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," in *SIAM Review*, pp. 573–595, 1995.

[39] M. W. Berry and R. M. Fierro, "Low-Rank Orthogonal Decompositions for Information Retrieval Applications," in *Numerical Linear Algebra with Applications*, vol. 3:4, pp. 301–328, 1996.

[40] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, Second Edition*. John Wiley and Sons, 2001.

[41] T. Martin, M. Jones, J. Edmison, and R. Shenoy, "Towards a design framework for wearable electronic textiles," in *Proceedings of the Seventh International Symposium on Wearable Computers*, pp. 190–199, ISWC 2003.

[42] Analog Devices Inc., "ADXL311 Dual-Axis Accelerometer," 2006. http://www.analog.com/.

[43] Analog Devices Inc, "ADXRS300 Gyroscope," 2006. http://www.analog.com/.

[44] Measurement Specialties Inc., "LDT-052K Piezoelectric Film Sensor," 2006. http://www.msiusa.com/.

[45] Motion Labs Inc., "C3D.org," 2005. http://www.c3d.org/.

[46] National Instruments, "The Cubix," 2005. http://www.ni.com/.

[47] CMU Graphics Lab Motion Capture Database, 2005. http://mocap.cs.cmu.edu/search.html.

[48] R. Joshi and A. C. Sanderson, "Multi Sensor Fusion: A Minimal Representation Framework," in *Series in Intelligent Control and Intelligent Automation*.

[49] J. Farringdon, A. Moore, N. Tilbury, J. Church, and P. Biemond, "Wearable sensor badge and sensor jacket for context awareness," in *Proceedings of the Third International Symposium on Wearable Computers*, pp. 107–113, ISWC 1999.

[50] Webmetronome, "Java applet of free online metronome," 2006. http://www.webmetronome.com/.

[51] Dance TV, "Dance TV: Online tutorial for ballroom basics," 2006. http://dancetv.com/tutorial/.