# Geometry Aware Compressive Analysis of Human Activities : Application in a Smart Phone Platform

by

Aswin Sivakumar

A Thesis Presented in Partial Fulfillment
of the Requirement for the Degree
Master of Science

Approved April 2014 by the
Graduate Supervisory Committee:

Pavan Turaga, Chair
Andreas Spanias
Antonia Papandreou-Suppappola

ARIZONA STATE UNIVERSITY

May 2014

## ABSTRACT

Continuous monitoring of sensor data from smart phones to identify human activities and gestures, puts a heavy load on the smart phone's power consumption. In this research study, the non-Euclidean geometry of the rich sensor data obtained from the user's smart phone is utilized to perform compressive analysis and efficient classification of human activities by employing machine learning techniques. We are interested in the generalization of classical tools for signal approximation to newer spaces, such as rotation data, which is best studied in a non-Euclidean setting, and its application to activity analysis. Attributing to the non-linear nature of the rotation data space, which involve a heavy overload on the smart phone's processor and memory as opposed to feature extraction on the Euclidean space, indexing and compaction of the acquired sensor data is performed prior to feature extraction, to reduce CPU overhead and thereby increase the lifetime of the battery with a little loss in recognition accuracy of the activities. The sensor data represented as unit quaternions, is a more intrinsic representation of the orientation of smart phone compared to euler angles (which suffers from gimbal lock problem) or the computationally intensive rotation matrices. Classification algorithms are employed to classify these manifold sequences in the non-Euclidean space. By performing customized indexing (using K-means algorithm) of the evolved manifold sequences before feature extraction, considerable energy savings is achieved in terms of smart phone's battery life.

## DEDICATION

*To Mom and Dad.*

## ACKNOWLEDGEMENTS

I would like to express my deep sense of gratitude to my advisor Prof. Pavan Turaga for his consistent support throughout the research study. His critical comments at various stages helped in enhancing the results. I express my sincere thanks to the Graduate students of the Department of Arts, Media and Engineering who offered suggestions and participated in the research study, the authorities which helped me use the Intelli stage lab in the department and provided me with other requisites whenever needed. It would be incomplete if my friends and family are not acknowledged. I sincerely thank them for their support.

TABLE OF CONTENTS

Page

LIST OF TABLES . . . . . v

LIST OF FIGURES . . . . . vi

CHAPTER

1 INTRODUCTION . . . . . 1

2 LITERATURE REVIEW . . . . . 4

3 SYMBOLIC APPROXIMATION ON ACTIVITY RECOGNITION . . . . . 9

3.1 Activity Recognition: Overall Approach . . . . . 9

3.2 Smartphone: Device coordinate frames . . . . . 10

3.3 Signal Acquisition . . . . . 11

3.3.1 Smartphone placements . . . . . 11

3.3.2 Positioning belts . . . . . 12

3.4 Signal Processing . . . . . 12

3.4.1 Windowing . . . . . 13

3.4.2 Features: Time-domain & Frequency-domain . . . . . 13

3.4.3 Feature computation & Storage . . . . . 17

3.4.4 Training: Classifier models . . . . . 17

3.5 Approximation of the sensor information . . . . . 21

3.5.1 Effects of varying the sampling frequency . . . . . 23

3.5.2 Effects of varying the approximation parameters . . . . . 24

4 GEOMETRIC PROPERTIES OF QUATERNIONS . . . . . 26

4.1 Theory of quaternions . . . . . 26

4.1.1 Quaternion rotation . . . . . 27

4.1.2 Advantages of quaternions: Gimbal lock problem . . . . . 27

4.1.3 Representation of a unit quaternion . . . . . 29

CHAPTER Page

4.2 Manifold theory . . . . . . 30

4.2.1 Tangent vectors & Tangent spaces . . . . . . 30

4.2.2 Riemannian metric . . . . . . 31

4.2.3 Exponential and Inverse exponential maps . . . . . . 32

4.2.4 Mean computations . . . . . . 32

4.2.5 Extrinsic Mean computations and Distance metrics with respect to unit quaternions . . . . . . 34

4.2.6 Uniform & Trained symbol set . . . . . . 36

4.2.7 Distance between two manifold sequences . . . . . . 38

5 RESULTS AND DISCUSSION . . . . . . 39

5.1 Effects of varying approximation parameters . . . . . . 39

5.1.1 Battery usage . . . . . . 39

5.1.2 Recognition accuracy . . . . . . 41

5.2 Human gesture recognition experiment . . . . . . 44

6 CONCLUSION AND FUTURE WORK . . . . . . 49

REFERENCES . . . . . . 50

# LIST OF TABLES

| Table | | Page |
|---|---|---|
| 3.1 | Physical Environments for various activities | 13 |
| 3.2 | Sample Feature Vector | 18 |
| 5.1 | Recogntion Accuracy: Varying PAA window size | 42 |
| 5.2 | Recogntion Accuracy: Varying alphabet set size | 43 |
| 5.3 | Recognition Accuracy:Sensor Information | 44 |
| 5.4 | Recognition Accuracy(Rotation Information).Symbol set size=10. | 47 |
| 5.5 | Recognition Accuracy(Acceleration Information).Symbol set size=10. | 47 |
| 5.6 | Recognition Accuracy(Acceleration Information).Symbol set size=30. | 48 |

LIST OF FIGURES

Figure Page

3.1 Overall Block Diagram . . . . . . . . . . 10

3.2 Device Coordinate System & Axis of Rotations . . . . . . . . . . 11

3.3 Setup . . . . . . . . . . 12

3.4 Overlapping / Sliding Windows . . . . . . . . . . 14

3.5 Feature Computation . . . . . . . . . . 17

3.6 A simple neuron. . . . . . . . . . . 18

3.7 A simple Decision tree structure . . . . . . . . . . 19

3.8 Piecewise Aggregation (PAA). $\bar{\mathrm{C}}$ denotes the approximation of the Original signal C . . . . . . . . . . 22

3.9 Symbolic Approximation: Symbols a, b, c are assigned to the piecewise aggregated time series. . . . . . . . . . . 23

3.10 Effect on SAX on CPU Computational Time . . . . . . . . . . 23

3.11 Battery Drain Curves at different Sampling Rates : A 10 % decrease scenario. . . . . . . . . . . 24

3.12 Effects of varying Approximation parameters on CPU Computational Time . . . . . . . . . . 25

4.1 Axis angle Representation . . . . . . . . . . 26

4.2 Rotation of a point by unit Quaternions . . . . . . . . . . 27

4.3 Gimbal Lock Representation . . . . . . . . . . 28

4.4 +q and -q rotation of an arbitrary point [a,b,c]. . . . . . . . . . . 30

4.5 Tangent vectors, Tangent Spaces on a manifold. . . . . . . . . . . 31

4.6 Riemannian Manifold . . . . . . . . . . 33

4.7 Distance between two quaternions q1 and q2 . . . . . . . . . . 35

4.8 Mean Computations: Quaternion Space . . . . . . . . . . 36

Figure Page

4.9 Symbol set . . . . . . . . . . 37

5.1 Android App . . . . . . . . . . 40

5.2 Battery Usage: PAA Window Size variation . . . . . . . . . . 40

5.3 Battery Usage: Alphabet Set Size variation . . . . . . . . . . 41

5.4 Actitracker Dataset and its Class Distribution . . . . . . . . . . 42

5.5 Trained Symbol set . . . . . . . . . . 43

5.6 Gestures: Motion Trace . . . . . . . . . . 45

5.7 Effects of varying Approximation parameters on Battery Usage . . . . . . . . 46

5.8 Pairwise Distance plots: Rotation Information . . . . . . . . . . 46

5.9 Pairwise Distance plots :Acceleration Information . . . . . . . . . . 48

# Chapter 1

# INTRODUCTION

Human activity recognition generally can assume many forms. Among them the principal methods include on-body sensor-based approach and the vision-based approach. The former method involves the use of sensors like accelerometers, on body or body worn sensors, active badge systems etc. which provide multi modal input. The latter approach involves video processing, where the input images from a camera are tracked and processed to identify potential features that best classifies the activity. This technique is chiefly employed in user-interface design and in the field of Human Computer Interaction [21]. Recognition and classification of activities of daily living like walking, climbing stairs, sitting, standing, running or jogging etc. has become quite significant in the areas of mobile and ubiquitous computing. Some of the applications include health monitoring – detecting falls in elderly subjects which could cause potential injury, fitness tracking – identifying total number of calories burnt for every mile of running activity, social networking [? ] – updating current user activity like jogging or running on user profiles of social websites like twitter or facebook and several context driven or context-aware approaches – playing a soundtrack while the user is jogging in leisure etc.

Accelerometers are the most common sensors that have been widely used in sensor based activity recognition over the past few decades. In the past, several biaxial accelerometers (modules that provide acceleration information in the X and Y axes) placed on various sites of the body like forearm, wrist, thigh and legs have been used to classify upto 20 different activities with significant accuracy [3]. With the advent of a tri-axial accelerometer (which provides acceleration in the Z axis in addition to X and Y dimension) , the need for multiple sensor placements is avoided and it became possible to use a single tri-axial accelerometer to classify the ADL activities with good accuracy. Then over the last decade smartphone driven Activity Recognition (AR) applications have become significant as most of the modern mobile devices are equipped with a tri-axial accelerometer together with the rich set of embedded sensors like orientation sensor, gyroscopes, microphones, GPS etc. which enabled better classification results to be obtained from the rich set of sensor information. Further, considering the processing power, storage capabilities and other resources of todays smartphone, we can see that establishing a data mining and classification algorithm on board is highly feasible.

The recent trend in smartphone based activity recognition shows that continuous sampling of sensor data leads to better classification results. However tradeoff lies in terms of drastic consumption of battery power and significant utilization of the CPU resources. To solve the above problem, symbolic representations of the sensor data are considered to reduce the size of the acquired raw sampled data and subsequent feature computation steps are employed on these symbols to reduce the computational overload by almost half thus leading to energy savings by a significant factor. These discretization techniques are widely used in many domains which involve motif discovery in telemedicine time series, DNA sequencing etc.

Also, past research has shown clearly that a single tri-axial accelerometer can classify translational activities like walking, running, climbing and static activities like sitting and standing. However they do not consider the geometry of the rich 3D data & are insufficient to classify orientation related activities like turning, twisting and inter classification between static activities like standing, sitting and lying down. Hence, we consider quaternion-based orientation information obtained from the smartphone and study their representation and utility in activity recognition. However the quaternion space is a non-Euclidean space and the general notions of Euclidean geometry cannot be used. Hence, metrics on the spherical manifold are considered in this study for effective signal matching and feature computation.

Contributions and Organization:

The main contributions made to this Activity Recognition study are as follows:

1. Geometrical consideration of the non-Euclidean group of 3D rotations in the form of unit quaternions towards rich feature computations and better classification of activities.

2. Conservation of smart phone's battery resources through approximation of the rich sensor data sampled for subsequent computations.

The Document is organized into the following parts:

Chapter 3 Identifies the general notion towards human activity recognition with time series accelerometer data. Study on the effects of placing the sensor at various parts of the body on the classifier efficiency is made. Also identifies the implementation of Symbolic Approximation techniques to the time series accelerometer data. Studies on the effects of varying the parameters of SAX technique like window length and the number of symbols or alphabets used for discretization is studied and an optimum pair is identified.

Chapter 4 Discusses the use of orientation information from the smartphone in the form of quaternions for achieving better classification efficiency after computing the features from the non Euclidean manifold sequences.

Chapter 5 Discusses the experimentation performed and results obtained. Finally, the document suggests potential future extension of this work.

# Chapter 2

# LITERATURE REVIEW

In this chapter, we discuss the existing methodologies of human activity recognition from multi-modal sensor platforms.

Tapia et.al (2007) used wireless heart rate monitor and wireless accelerometer to detect not only the type of physical activity performed but also the intensity of such activity [19]. However, at the end of the study it was found that Heart rate has negligible significance in detection of the type of activity as it is found to be altered by the subjects emotional states as well.

Gyllensten et.al (2011) employed multi-sensor input information for activity recognition. Activites such as Walking, Running, Lying, Sitting and Standing were classified and the results are compared with a commercially available activity recognition device IDEEA [9]. Some of the features considered in this study are mean, variance, skewness, Fourier transform & kurtosis etc.

Ravi et.al (2005) performed analysis between base-level and meta-level classifier performances for different test conditions like gathering data from a single subject over several days, from multiple subjects over different days. Several classifiers were tested, such as Decision trees, SVM , Nave Bayes etc., Plurality voting was found to perform consistently well [18].

Takumi et.al designed an algorithm to cancel out the effects of rotation of a phone on its 3D axis accelerometer data output [13]. The principal feature extracted from the signals in this study is the Fourier transform of the sample window. Invariance to rotations & temporal shifts of the smartphone is provided by the formation of an auto-correlation matrix of the complex fourier features. Due to the unitarity nature of the rotation matrix, the rotational effects are unseen in the auto-correlation matrix, rendering it suitable for rotation invariant feature extraction.

Bagala et al made a comparative study on performance of thirteen different fall-detection algorithms when they are applied to a database of 29 real-world falls [7]. These algorithms apart from detecting a fall also detected parameters like the posture, velocity before/after a fall. The study helped to benchmark the performance of fall detection in terms of sensitivity & specificity.

Yan et al worked towards reducing the energy overhead of continuous sampling of accelerometer data towards activity recognition through the concept of A3R adaptive accelerometer-based activity recognition [23]. It was found that the choice of parameters like accelerometer sampling frequency & classification features affects the energy overhead vs. classification accuracy tradeoff for each activity separately. The principal idea behind A3R approach is to continuously track the current/ongoing activity of the user and then adjust the above two parameters dynamically which makes the choice optimal for detecting that activity efficiently. Time domain features like mean, rms value, variance & covariance etc., or frequency domain features like energy (FFT components) , entropy (FFT histogram) or a combination of both are used as features depending on the type of activity. It was found that time-domain features are computationally less expensive than a combination of both time+frequency domain features. The overall energy savings of the A3R approach applied on Android phones was found to be $20 - 25$ %.

Kose et al compared the performance of naive-Bayes classifier vs. the performance of clustered KNN online classification algorithms. The clustered KNN uses a combination of both minimum distance and k-nearest neighbour classification algorithms. It was found that clustered KNN has an overall average accuracy rate far higher than the naive-Bayes algorithm [14]. Also the study compared the execution times & resource consumptions for both the algorithms on the smartphone platform. Increasing K in clustered KNN considerably increased the execution time. The resource consumption study shows that applications using minimum distance classifier and clustered KNN consume nearly the same amount of CPU resources. On the other hand naive-Bayes algorithm has a considerably higher amount of CPU usage. For benchmarking the CPU usage, studies are carried out by comparing with text to speech and other internet-based applications.

Viet et al performed a comparative study between Support Vector Machine (SVM) based classification and Dynamic Time Warping (DTW) method of classification considering both time and frequency-domain features. Results show that SVM and DTW methods using time-domain features produce higher levels of classification accuracy than using frequency-domain features. It is also found that frequency-domain features have more computational complexity than time domain features & that SVM classifier consumes less average power when compared to DTW technique utilizing the time domain features.

Rachuri et al [17] proposed SociableSense – a smartphone application which captures user behavior in an office setting through activity recognition. The system employs an adaptive sampling mechanism as well as models to perform computation of tasks such as classification algorithms locally on the smartphone device or on the cloud. The algorithm assigns weights to various parameters like energy, latency and amount of data sent over the network for performing a classification task depending on the user needs. For example, if a smartphone user has unlimited data plan, then he or she can send the raw accelerometer data to a server on the cloud to perform further computations and classification if battery life is considered a higher priority. Linear, quadratic & exponential functions are utilized to adaptively adjust the sampling rate based upon the events (missable or unmissable) which are marked from the raw sensor data by a GMM classifier. Missable events are not of interest, hence the sampling rate can be decreased at these events. On the contrary the sampling rate can be increased if an event is deemed unmissable by the GMM classifier. This approach serves to conserve the CPU resources of the smartphone.

Vega et al [5] worked towards unconstrained mobile sensor-based human activity recognition where the activities of daily living are detected irrespective of the user specific mobile sensor position, orientation and body attachment (as loose device attachment limits the accuracy of the sensor measurements by introducing information of the self-displacement of the data). The orientation correction is performed by convolving or multiplying the raw sensor data which is obtained with respect to the sensor frame with a computed quaternion that describes the sensor frame of reference with respect to the earth frame of reference. In this way the measurements become identical for each distinct activity regardless of the smartphone orientation.

Though major research in the past as seen above is focused towards improving the classifier efficiency, not much focus is given to the geometry of the rich data acquired from the sensors and the importance of conserving CPU resources to improve battery life.

A related work in the field of Computer Vision [2] finds the use of symbolic representations of motion sequences (video data) which evolve in non-Euclidean spaces like Kendall's shape space (spherical manifold) & affine shape spaces (Grassmann manifold) to effect fast, efficient indexing & subsequent computations on the data in such complex spaces. Inspired by this work on activity recognition, the following research study is carried out on a signal-based sensor data (unit quaternions) acquired from smartphones. The study takes into consideration the geometry of the spherical manifold of the unit quaternions while extracting the features & performing compressive analysis on the sensor data to classify human activities and gestures. Much focus is given to conserve smart phone's power consumption by implementing optimizations in terms of choosing the right set of parameters of the approximation algorithms. The implementations are done in Java-Android framework to study and observe the improvement in the battery statistics after including the optimizations.

# Chapter 3

# SYMBOLIC APPROXIMATION ON ACTIVITY RECOGNITION

Before considering the concepts of non-Euclidean manifold geometry of the sensor data, we have studied and implemented the importance of sensor data compaction and indexing, focused towards conservation of battery resources of a smartphone. For this analysis study, approximation techniques are initially applied to vector time sequences (which has Euclidean geometry) as obtained by a single tri-axial accelerometer of a smartphone. We have also developed an Android application to study the effects of approximations in terms of battery performance of the smartphone and the efficiency of classification of activities is also observed. Common activities of daily living such as walking, climbing stairs, standing/sitting, bending etc., are considered for classification in this study.

## 3.1 Activity Recognition: Overall Approach

The basic approach of activity recognition from the time series data of a smartphone is shown in Figure 3.1. The Java-Android application running on the smartphone acquires the accelerometer sensor information along the various orthogonal axes of the smartphone (X, Y & Z). The acquired sensor data is transmitted to the server application running on the PC side over bluetooth. The desired bluetooth MAC address of the PC and the com port of the server has to be specified in the application. Suitable overlapping windows of size N are chosen and applied on this acquired time series data and features are extracted along each axis & stored in a separate file for subsequent analysis. Once computation is performed for the entire set of windows, classification is attempted in Weka. Weka is a popular machine learn-

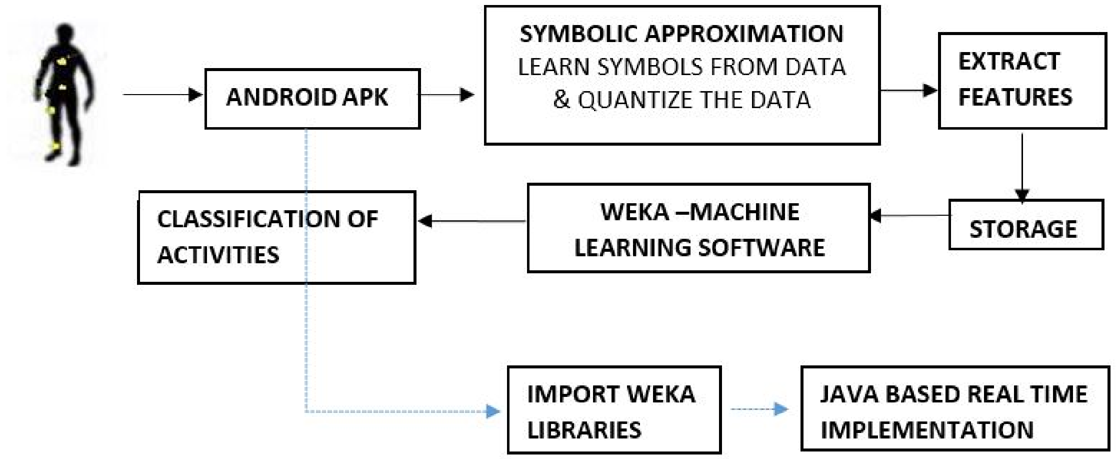


Figure 3.1: Overall Block Diagram

ing software which has a collection of visualization tools and algorithms to perform standard data mining or analysis tasks such as data preprocessing, clustering, regression and classification using the various available inbuilt classifiers. Alternatively, this Java-based weka library packages can be imported into the android platform to create a stand alone real time activity classifier application (as shown by the dotted lines in the figure 3.1).

## 3.2 Smartphone: Device coordinate frames

The native co-ordinate system of the accelerometer with respect to the smartphone is given below:

1. X axis is aligned along the body axis of the smartphone.
2. Z axis points downwards so that it is aligned with the gravity when the smartphone is kept flat on a table.
3. Y axis is orthogonal to both X and Y axes as shown in Figure 3.2.

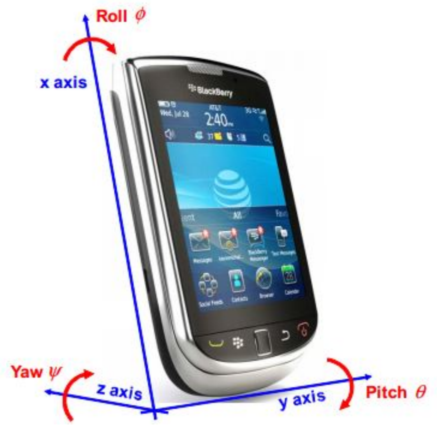


Figure 3.2: Device Coordinate System & Axis of Rotations

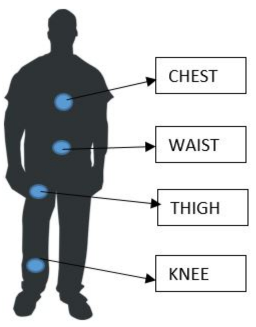
The rotations of the device brought about the axes X, Y and Z are termed **roll, pitch and yaw** respectively.

## 3.3 Signal Acquisition

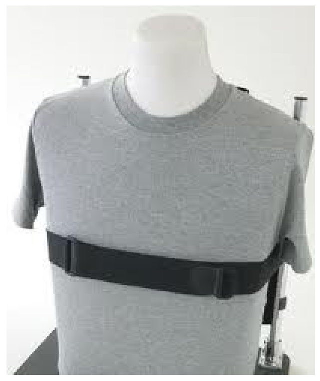
### 3.3.1 Smartphone placements

In addition to power consumption and classification accuracy studies, we also study the effect of sensor placement/location relative to the human body, on the classification results. The following four locations for smartphone placements are considered for this analysis. It is found that smartphone placement near the waist has good classification results.

1. Chest (Center position).
2. Waist (Center position).
3. Thigh (Left or Right).
4. Above ankle joint (Left or Right) (as shown in Figure 3.3a).



(a) Phone Placements



(b) Positioning belts

Figure 3.3: Setup

### 3.3.2 Positioning belts

The main purpose of using positioning belts is to prevent translational movements or slipping of the phone during signal acquisition process which might affect the classification efficiency at the output. Hence, for each activity measurement and for each location, suitably sized belts/wraps are worn by the user before the commencement of the signal acquisition (as shown in Figure 3.3b).

For each smartphone placement location, each specific activity say walking, climbing stairs, sitting/standing, reaching for an object etc., is performed continuously for a period of 100 seconds. Every trial is repeated sufficient number of times to obtain a large feature set and the same activity is repeated for the same period of time but with a different placement location (say waist). The physical environments considered for each activity is discussed in Table 3.1.

## 3.4 Signal Processing

This section deals about the windowing (overlapping windows) and feature extraction operations performed on the acquired 3 channel time series data.

Table 3.1: Physical Environments for various activities

| **Activity** | **Environment** |
|---|---|
| Walking | Subject is made to walk in both rough and even terrain. |
| Climbing | Normal pace climbing (Upstairs or Downstairs). |
| Standing/Sitting | Subject is made to periodically sit/stand in a chair. |
| Reaching for an object (Bending) | Subject is asked to grab an object in the ground from the standing position. |

### 3.4.1 Windowing

Windowing operation is performed on the data obtained from each axis. A suitable window size of about 5 seconds (500 samples if the sampling frequency is 100 Hz) is chosen. The periodic amount of time required to perform each activity is taken into account while choosing the window length (N samples) such that, each window has sufficient information about an activity performed. After extracting the samples in a particular window, the window is then time shifted by an order of 2.5 s and the signal sample of size N is obtained again. Thus, window is time shifted in such a way that, each window overlaps by half amount (50%) with its subsequent window (as shown in Figure 3.4). This process is continued for the entire signal length (100 seconds). Thus a total of 36 windows are obtained for each activity performed with the phone placed in one of above discussed placements.

### 3.4.2 Features: Time-domain & Frequency-domain

For each window of sample size N, features are extracted. The various features which are considered are discussed below.

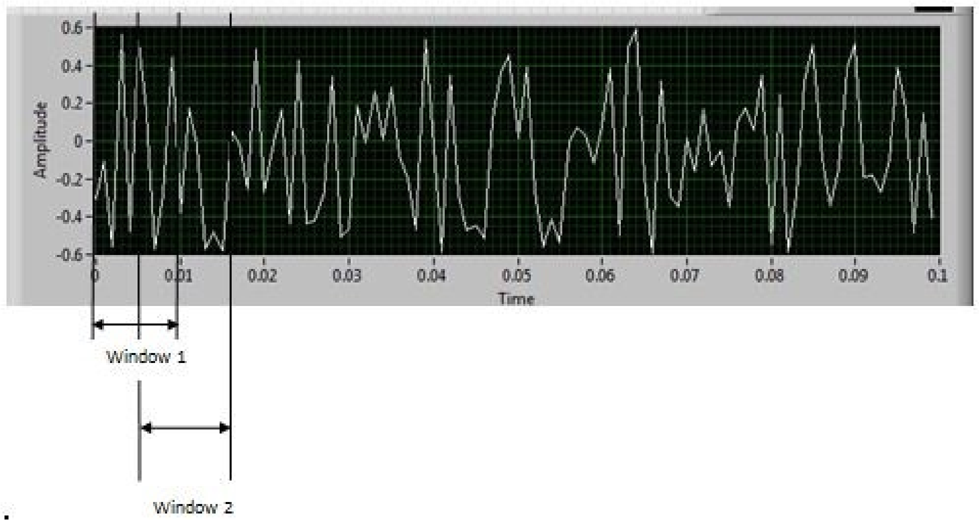


Figure 3.4: Overlapping / Sliding Windows

Time-domain features:

*Mean:*

The mean over the window of sample size N can be obtained with little computational cost and it requires minimal memory requirements. This metric is primarily used to differentiate between activities which are static and activities that are dynamic and also used to recognize user posture.

*Standard Deviation:*

Standard deviation metric is used to define the stability of the signal. For example, it was expected that when the accelerometer is worn on the waist and if the subject walks, they sway to a greater extent than when sitting/bending down. Thus the standard deviation parameter helps to achieve classification between these activities [10].

*Range:*

This metric is defined as the difference between the minimum and maximum sample value in any given window and they are primarily used to differentiate between activities such as walking and running.

*RMS (Root Mean Square)*

The root mean square of a signal $S_i$ is defined as

$$RMS = \frac{1}{n}\sum_{i=1}^{n}\sqrt{x^2 + y^2 + z^2} \tag{3.1}$$

Here x,y & z denotes the resolved components of the signal along X,Y & Z axes respectively. The RMS value of a signal as shown by 3.1 is primarily used to identify falls and to distinguish behavior patterns in walking.

*Correlation Coefficient:*

The strength and the direction of a linear relationship between two correlations is given by signal correlation. This metric is useful for distinguishing activities that involve translation in a single dimension. The correlation coefficient (See Equation 3.2) is evaluated between any two axis (say X and Y) at a time. Correlation is quite helpful in classifying dynamic activities that involve multi-dimensional movements. For example walking and sitting can be distinguished from climbing upstairs or downstairs using the correlation coefficient.

$$CorrelationCoefficient(X,Y) = \frac{Covariance(X,Y)}{Std(X)Std(Y)} \tag{3.2}$$

*Zero Crossings:*

The number of times the given signal in a window crossing the mean value (or typically half the signal range) is termed as *Zero Crossing count.* This metric is found to be useful to classify activities that involve periodic movements like *walking and running* and also can detect more complex human gestures.

*Angular Velocity:*

This metric when combined with other sensor data is used to identify the user orientation.

*Signal Magnitude Area:*

Signal Magnitude Area or SMA as given in Equation 3.3 is principally used to discriminate between dynamic and resting physical activities. It is defined as the magnitude sum of the areas under each axial curve over the given window size N

$$SMA = \frac{1}{N}(\int_0^N |x(t)|\,dt + \int_0^N |y(t)|\,dt + \int_0^N |z(t)|\,dt) \tag{3.3}$$

Frequency-domain features:

In terms of frequency-domain, the fourier transform of each window can also be computed. Activities such as walking, running are cyclic in the sense that they can be well distinguished from non cyclic events using their frequency distribution. ***Spectral Energy*** (squared sum of the spectral coefficients normalized by the length of the sample window) metric is used to identify the mode of transport like walking, running, cycling and driving. Similarly ***Entropy*** metric can be used to identify the patterns in similar activities like walking and jogging. It is the normalized entropy of

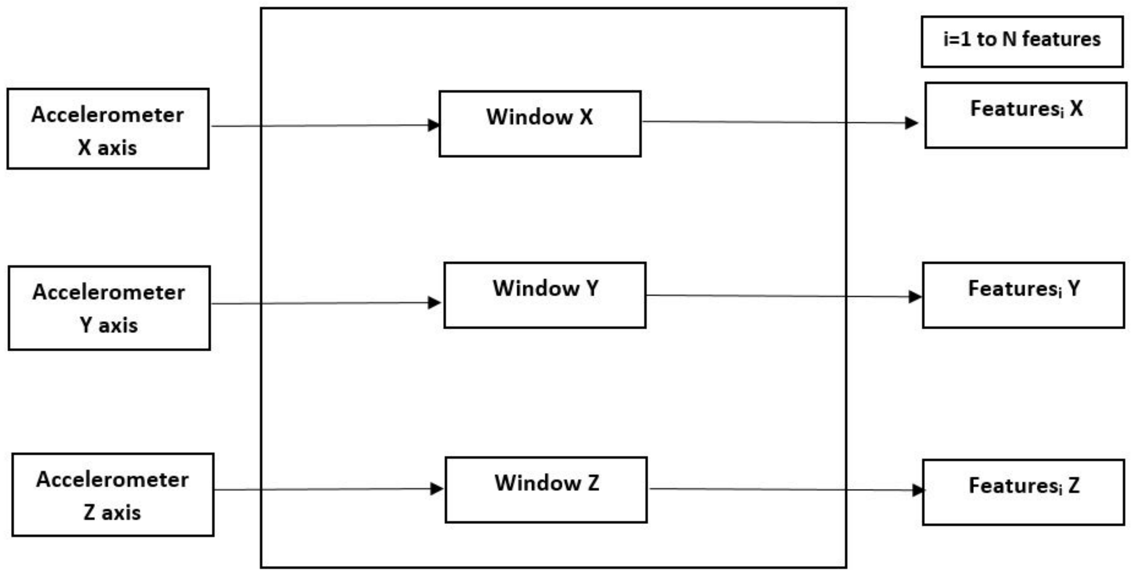


Figure 3.5: Feature Computation

the FFT coefficient magnitudes excluding the DC component in the FFT spectrum. However, since frequency-domain features computations are more expensive than the time-domain features [8] they are disregarded in this study.

### 3.4.3 Feature computation & Storage

The computation of the above features are done in MATLAB for each window & for each axis (X, Y & Z) as shown in Figure 3.5 .Thus procedure is iterated for the total set of 36 overlapping windows (total period of 100 seconds).The computed features are stored in either csv or arff format and the class label for a given computed feature vector is appended to the end of the feature vector as nominal values. A sample Feature vector is shown in Table 3.2

### 3.4.4 Training: Classifier models

Neural Networks are models that are built for learning and optimization. More closely they resemble the biological neuron. A neural network consists of large num-

Table 3.2: Sample Feature Vector

| Mean X | Std X | Mean Y | Std Y | Mean Z | Std Z | Corr (X,Y) | Corr (X,Z) | Corr (Y,Z) | ..... | **Class** |
|---|---|---|---|---|---|---|---|---|---|---|
| 0.23 | 0.34 | 0.22 | 0.1 | -0.23 | -0.01 | 1.23 | 0.59 | -0.89 | ..... | **1** |

ber of simple processing elements called neurons or nodes. Each neuron or node is connected to an adjacent node through links which has an associated weight $w_i$. with it as shown in Figure 3.6. Training a neural network model means adapting these weights on the links so that the input X is matched with its correct target value Y. Some few neural network models considered in this study are discussed here under.

*Decision Trees*

A decision tree is a way of splitting a large data set into branch like segments arranged in an inverted fashion that reflects a tree like appearance with the root node on the top of the tree. The main purpose of building a decision tree is to arrive at a target value for an end variable, based upon the input from several variables. Decision trees can predict the values for unseen observations for which input is known but target value is unknown [6]. Figure 3.7 shows a simple decision tree model for classification

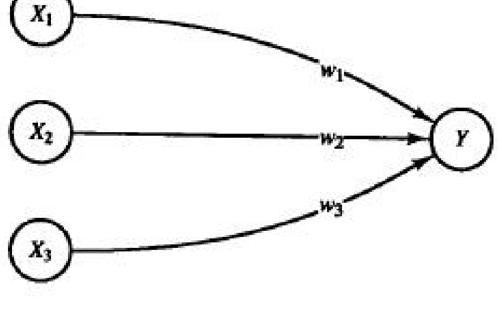


Figure 3.6: A simple neuron.

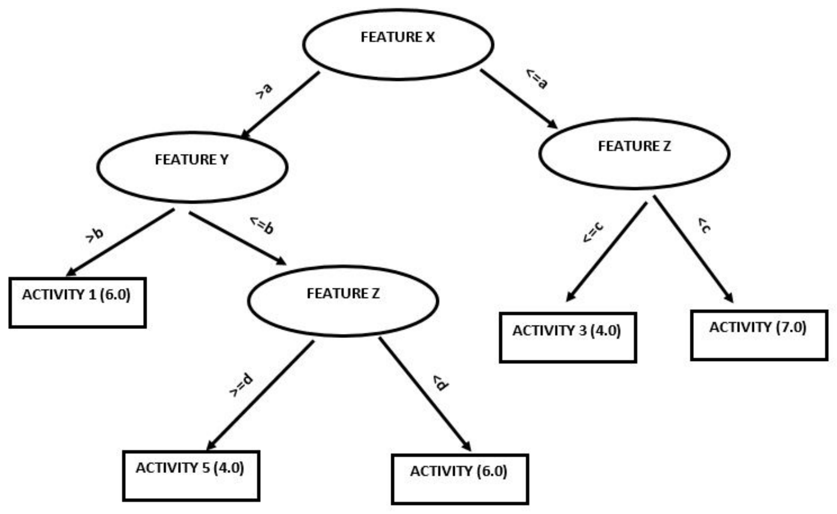


Figure 3.7: A simple Decision tree structure

of 5 different activities based on three computed features (as discussed in the previous sections) from the time series data. The numbers in parenthesis next to the activity, signifies the number of observations of that activity class.

*Naive-Bayes Classifier*

Bayesian classifiers are particularly used when the number of input feature vectors to the model are very high. The algorithm is based on the **Bayes rule**. They can be built with real valued inputs. A naive-Bayes Classifier performs decision making based on **prior experience or information** which is given (Probability of occurrence of event A given B) i.e. it can predict the outcome, without the need for them to actually occur. For this purpose, two parameters are evaluated- **Prior probability and Posterior probability**.

Considering the event of classifying an object X between A and B, with prior information given as M > N where M is the number of objects in A and likewise N in B, the **Posterior probability** measures the likelihood of the unknown object X in the vicinity of A/B as follows

**Posterior probability of X being A/B = Prior probability A/B * Likelihood of X given A/B**.

The object X is classified to the class A or B which assumes the **largest Posterior probability value**.

Advantages of naive-Bayes Classifier:

1. Reduced computational complexity as compared with decision trees.
2. Bayesian models easily adapts to the environment and they are resistant to irrelevant attributes [1].

The main disadvantage of Bayes classifier is that they are limited to only simplified models.

### *Support Vector Machines*

Support Vector Machines is a classification technique introduced in 1992 by Boser, Guyon and Vapnik. Due to its high accuracy and ability to accept a high dimensionality of input data, it is widely used in the area of Bio-Informatics. Support Vector Machines are unique in the sense that they can efficiently create non linear decision boundaries based on kernel algorithms especially when the feature input space has non linearities ( i.e. when a linear decision boundary is difficult to construct). Before training an SVM model care has to be taken while pre processing the data, choosing the kernel and while setting the parameters of SVM. Otherwise, they would lead to reduced classification efficiencies. In cases where the two classes are not linearly separable, a nonlinear mapping function $\varphi$ is used which non linearly transforms the input space into a separate feature space where the inputs are linearly separable. Once the above step is done, the algorithm now follows Linear SVM where a linear decision boundary or hyperplane is created in the feature space between the classes.

## 3.5 Approximation of the sensor information

Classification of activities of daily living using a single tri-axial accelerometer of the mobile phone will help in a long way towards activity driven context aware applications. Though the accuracy of detecting these day to day activities has been reported high in various research work, only a few contribution has been done towards optimizing the battery life of the smartphones.

In this research, we adopt and extend the signal approximation strategy first proposed in [16], who applied it to scalar-valued signals. The sensor data is subjected to preprocessing where the dimensionality of the signal is reduced by many folds. Symbolic representation allows a time series data which is a series of continuous values to be compressed and represented by a discrete set of symbols. The time series data $S_j$ of length $n$ is first subjected to Piecewise Aggregate Approximation (PAA) as shown in figure 3.8 which simply means that the original data series $S_j$ is divided into chunks/frames as determined by a window size (n/w) (where w denotes the number of frames), the mean of the samples in these equal sized frames are computed and the vector of these values becomes the approximation of the original time series data $S_i$ Equation 3.8. Here for a frame, the mean parameter is chosen as opposed to minimum, maximum or median values to mitigate the adverse effects of noise (if any) in the signal data from affecting the quality of approximation. Also it has to be noted that the original time series has to be normalized to have zero mean and unit variance.

$$S_i = \frac{w}{n} \sum_{\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} S_j \tag{3.4}$$

The next step called Symbolic Approximation involves discretization of the PAA sequence by quantizing each value in the sequence to the nearest symbol [4] in the discrete set of symbols as shown in figure 3.9. The discrete set of symbols are usually formed by determining the breakpoints in the Gaussian distribution of the time series data (it is assumed Gaussian as the data is normalized).

Considering an instance, say the activity – *Sitting* class consists of 59,939 acquired float samples of data. It requires a total storage size of about (59,939 * 4 bytes) 2,39,756 bytes or 240 KB. If the above time series data is subject to symbolic approximation with a window size of 4 samples, then the approximated data will have a total size of about 60KB (240KB/4) . Thus we have a reduction factor of 4 in the total storage size. In addition, the computational overload / feature computational time will be reduced by means of working with the approximated data. This is clearly reflected in Figure 3.10 which shows a simulated CPU computational time for feature

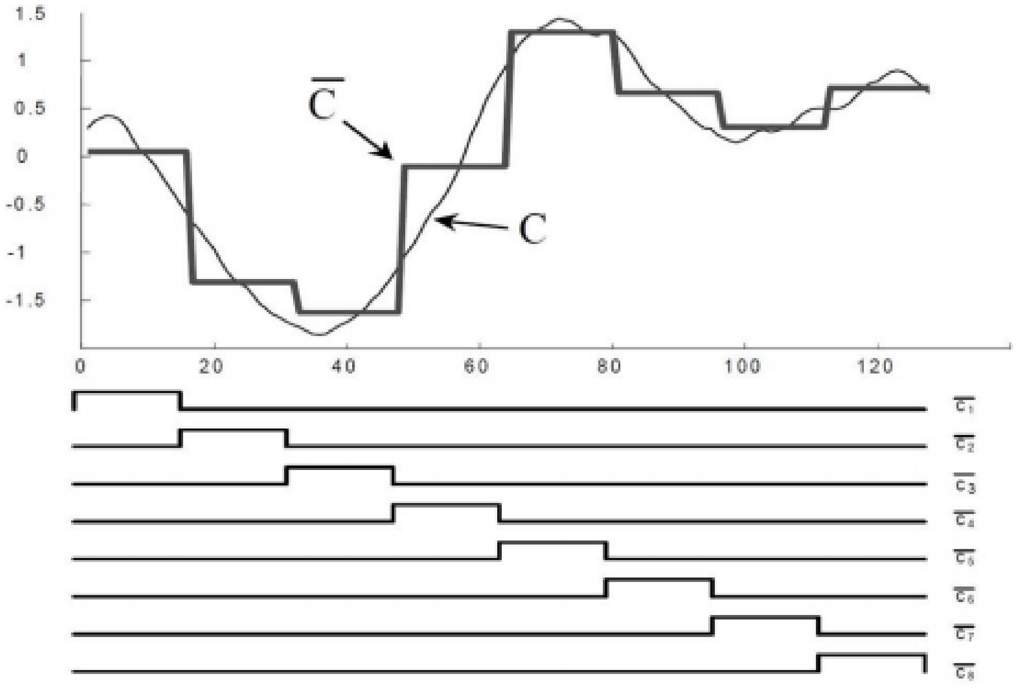


Figure 3.8: Piecewise Aggregation (PAA). $\bar{\mathrm{C}}$ denotes the approximation of the Original signal C

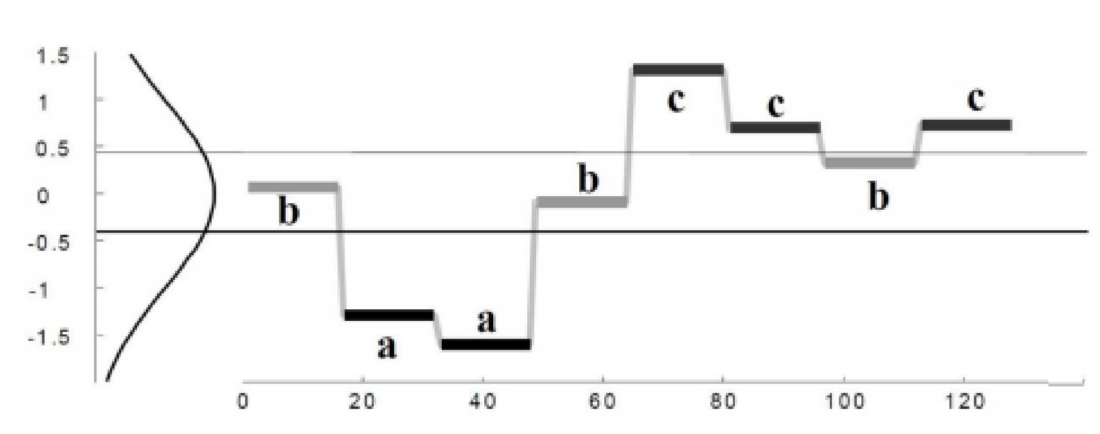


Figure 3.9: Symbolic Approximation: Symbols a, b, c are assigned to the piecewise aggregated time series.

extraction from a Compressed vs Uncompressed version of the data.

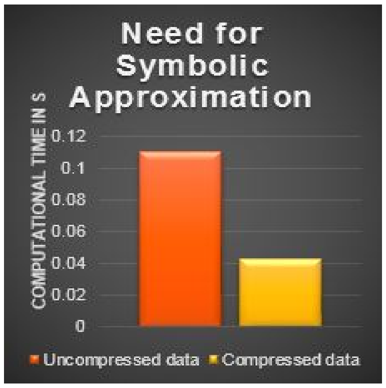


Figure 3.10: Effect on SAX on CPU Computational Time

### 3.5.1 Effects of varying the sampling frequency

Studies have shown that all the common activities of daily living can be classified with good accuracy when the sensor is sampled at 20Hz. Increasing the sampling frequency will have no further improvement in the accuracy but will have negative effects on the battery performance as shown in Figure 3.11. which indicates the

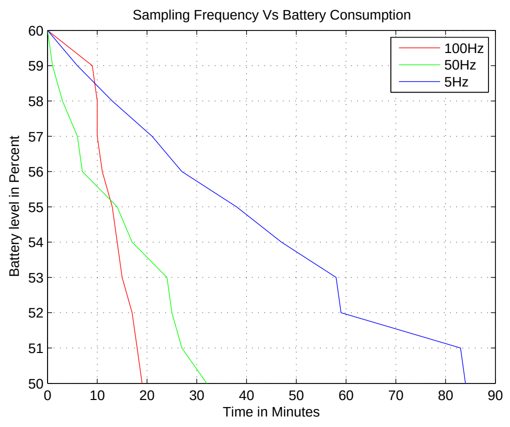


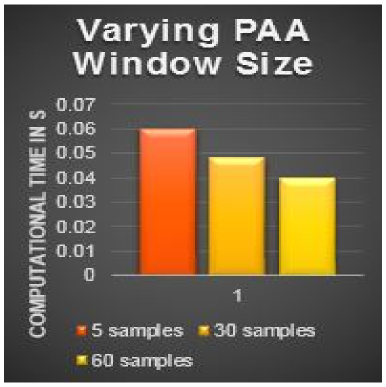
Figure 3.11: Battery Drain Curves at different Sampling Rates : A 10 % decrease scenario.

battery drain at different sampling rates.The sample study was implemented in an Android-based smartphone.

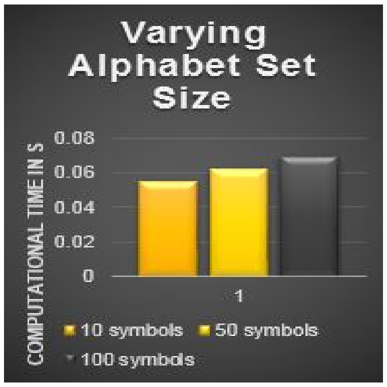
### 3.5.2 Effects of varying the approximation parameters

Having identified the need for SAX, in this section, variation of the individual parameters of SAX and their collective influence on recognition accuracy is analyzed. The PAA window size parameter indicates that lower the window size better will be the approximation representation (Figure 3.8). with respect to the true time series. However, there is a trade off in terms of increased number of samples which in turn would lead to increased computation and thus increased CPU Usage (due to frequent PAA operation in SAX). This is clearly depicted by comparing the window sizes 5 and 30 in Figure 3.12a However, on the other hand, a further higher window size will have bad effects on the recognition accuracy due to loss in resolution of the signal. Hence, an optimum value of window size of around 30 samples is chosen.

Also on the other hand a tradeoff lies in terms of choosing the alphabet or symbol set size. Choosing a large alphabet set size will lead to more accurate representation



(a) PAA Window size



(b) Alphabet set size

Figure 3.12: Effects of varying Approximation parameters on CPU Computational Time

of original time series but, a large number of comparisons has to be made to associate the aggregated output to a symbol which is depicted in Figure 3.12b.

Chapter 4

# GEOMETRIC PROPERTIES OF QUATERNIONS

## 4.1 Theory of quaternions

Quaternions are efficient ways of representing 3D rotations finding its much use in mechanics, aerospace, 3D game development and other graphical applications. They consists of 4 elements – 3 of which are vector components (x,y,z) which defines the three dimensional axis about which the rotation will occur and one scalar component $w$ which defines the magnitude of the rotation about that axis. The general representation of quaternions is given by Equation 4.1 where i,j,k denotes the pure imaginaries/orthogonal vectors which obeys Hamilton's Rule (Equation 4.2). Thus quaternions are a kind of axis angle representation as shown in Figure 4.1.

$$q = w + x * i + y * j + w * k \tag{4.1}$$

$$i^2 = j^2 = k^2 = ijk = -1 \tag{4.2}$$

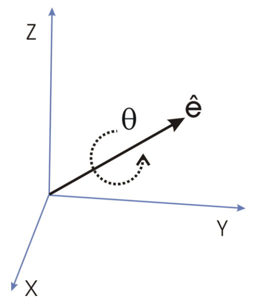


Figure 4.1: Axis angle Representation

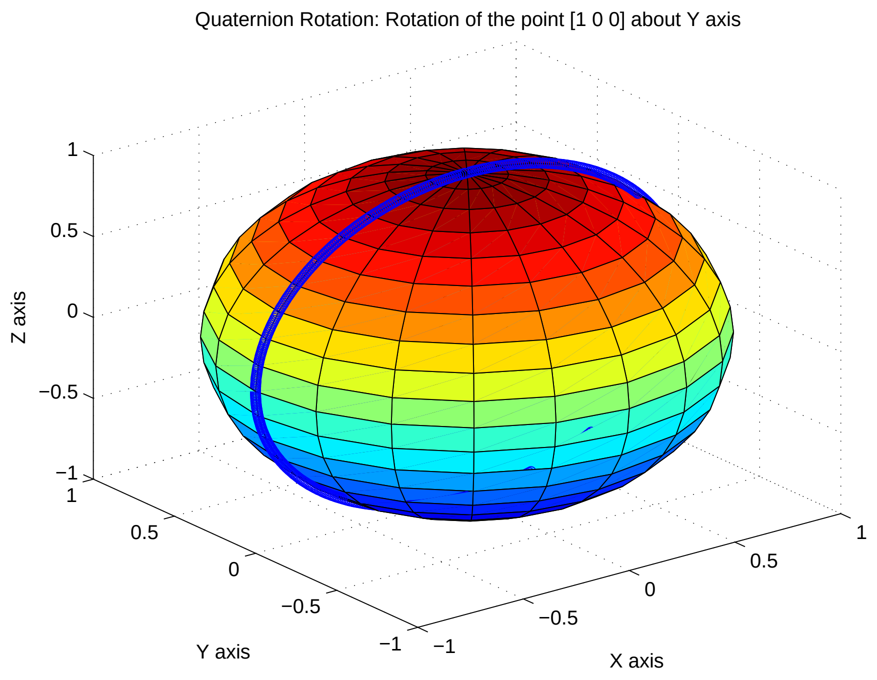


Figure 4.2: Rotation of a point by unit Quaternions

### 4.1.1 Quaternion rotation

Let P(0,x,y,z) be a pure vector on a unit sphere. Rotation of this vector by a unit quaternion q to a new point $P^{'}$ is obtained by Equation 4.3

$$P^{'} = q^{'} * P * q \tag{4.3}$$

q' denotes the conjugate of the unit quaternion q & * denotes quaternion multiplication. Figure 4.2 shows the rotation of the pure vector [0 0 1 0] on a unit sphere about the y axis (change in pitch) by unit quaternions.

### 4.1.2 Advantages of quaternions: Gimbal lock problem

Gimbal Lock is a condition which signifies the degeneration of a rotation in three dimension into a two dimensional space, when two of the three gimbals (also called

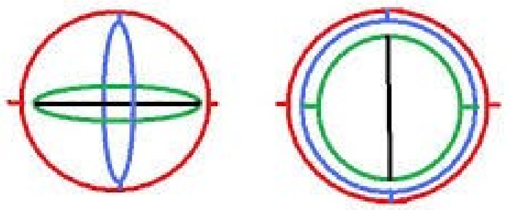


Figure 4.3: Gimbal Lock Representation

rotation axis) gets aligned in a parallel plane in a such a way that applied rotation on one axis is same as the applied rotation in another axis. This condition is depicted in Figure 4.3.

Figure in the left describes the three gimbals (red, blue and green denoting row, pitch and yaw) orthogonal to each other. Figure in the right describes two among the three gimbals (blue and green) getting aligned parallel to each other making the rotation applied on red gimbal same as rotation applied on blue gimbal, i.e, degenerating rotation in three dimension to rotation in two dimension.

In euler angles representation, if $\alpha$, $\beta$ and $\gamma$ denotes successive rotations along X, Y and Z axis, then a three dimensional rotation represented through a rotation matrix is given by

$$[\begin{pmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\beta & -sin\beta \\ 0 & sin\beta & cos\beta \end{pmatrix} \begin{pmatrix} cos\gamma & -sin\gamma & 0 \\ sin\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}]$$

Now when $\beta$=0,

$$[\begin{pmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} cos\gamma & -sin\gamma & 0 \\ sin\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix}]$$

Carrying out the Matrix Multiplication, [ $\begin{pmatrix} cos(\alpha+\gamma) & -sin(\alpha+\gamma) & 0 \\ sin(\alpha+\gamma) & cos(\alpha+\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix}$ ]

From the above resultant rotation matrix we find that the axis is now fixed in the Z direction. Irrespective of the fact that rotation is applied in the X or Y direction (changing $\alpha$ or $\gamma$) there is loss in one degree of freedom.

Since traditionally rotations are represented through euler angles gimbal lock is a main problem. On the other hand since quaternions are axis angle representations, orientations are represented as a single value rather than resolving into three separate axes.

### 4.1.3 Representation of a unit quaternion

Like unit vectors, unit quaternions are required to perform math like quaternion multiplication, interpolation etc. A unit quaternion is obtained by normalizing the quaternion vector Equation 4.4.

$$magnitude = \sqrt{w^2 + x^2 + y^2 + z^2}; q_{unit} = \frac{q}{magnitude} \tag{4.4}$$

Unit quaternions denote the *"space of rotations"* – the unit sphere $S^3$ in 4-space (Hypersphere H) with antipodal points +q and -q representing the same rotation as shown in Figure 4.4.

$$S^3 = q\epsilon H : \|q\| = 1. \tag{4.5}$$

With unit quaternions, it becomes easier to visualize 3D rotations and make use of natural and elegant distance metrics on the $S^3$ manifold.

## 4.2 Manifold theory

A set of points in the set $R^n$ which satisfies an equation f(x)=0 (with conditions defined on f()) is called a **manifold** [20]. Manifold theory study is of prime importance when it comes to feature spaces with non- Euclidean geometry. Here $R^n$ is a differential manifold, which means that it is locally Euclidean. That is to say, for a given point $p$ in a topological space $\boldsymbol{M}$, $p \; \varepsilon \; M$, there exists a mapping $\emptyset$ : $U \rightarrow R^n$ such that $\emptyset(U)$ is open in $R^n$, where U is the open neighborhood of $p$.Using this mapping chart called the coordinate chart (U,$\emptyset$), one can move between U and $\emptyset(U)$ and perform computations in a more convenient Euclidean space.

### 4.2.1 Tangent vectors & Tangent spaces

Understanding the tangent space structure of a manifold is required to perform differential calculus i.e. functional derivatives, critical points etc., on these manifolds. To observe the general notions of tangent spaces, let us consider an $n$ dimensional manifold $M$ as shown in Figure 4.5. For any point $p\varepsilon$ $M$, let us consider a differentiable curve $\gamma$ : (-$\varepsilon$,$\varepsilon$) $\rightarrow$ $M$ such that $\gamma$(0)=p. The **tangent vector** is the velocity vector

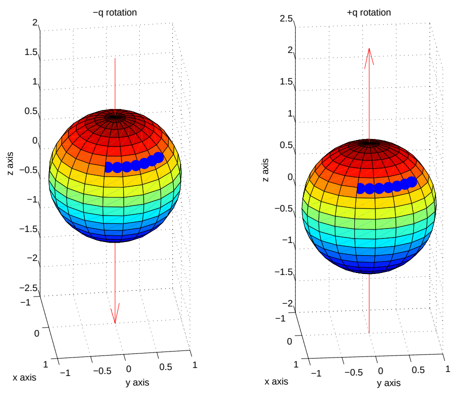


Figure 4.4: +q and -q rotation of an arbitrary point [a,b,c].

$\gamma(0)$ to manifold M at point p. The set of all such tangent vectors is called the **tangent Space**. Thus this tangent space is always linear when compared to the manifold allowing us to perform easier computations in the linear space and projecting back the results to the manifold space.

### 4.2.2 Riemannian metric

Riemannian metric is a mapping $\langle .,. \rangle$, that allows one to smoothly transform points $p \ \varepsilon \ M$ from the manifold space to the tangent space $T_p(M)$. A manifold with a Reimannian metric defined on it is called a Riemannian manifold and the metric is defined as shown in Equation 4.6.

$$< v_1, v_2 >= v_1^T v_2 \tag{4.6}$$

For any two given points *a,b* on a Riemannian manifold, it is possible to define the path length between them in terms of summation of the length of tangent vectors for all the points on the curve between a and b. That is to say, for a path $\alpha$ :[0,1] on the Riemannian manifold M that is differentiable everywhere in the interval [0,1], then the length of the velocity vector at a point t on the curve is given

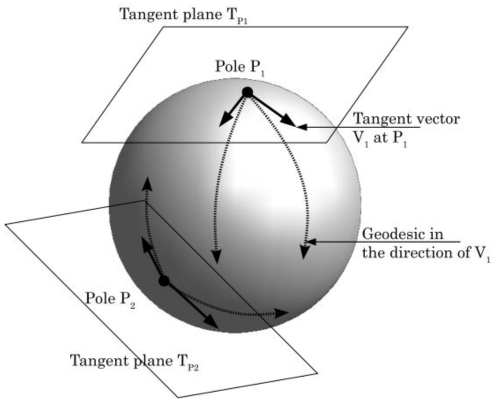


Figure 4.5: Tangent vectors, Tangent Spaces on a manifold.

by $\sqrt{(<\frac{d\alpha(t)}{dt},\frac{d\alpha(t)}{dt}>)dt}$. Therefore, the length of the entire path $\alpha$ is given by: Equation 4.7

$$L[\alpha] = \int_1^0 (\sqrt{(<\frac{d\alpha(t)}{dt},\frac{d\alpha(t)}{dt}>)dt}) \tag{4.7}$$

Among all possible lengths from a to b, the shortest possible length is defined as a geodesic [11] (see Figure 4.6b) which is given by Equation 4.8 . It is the infimum of the lengths of all smooth paths on M which start at a and end at b.

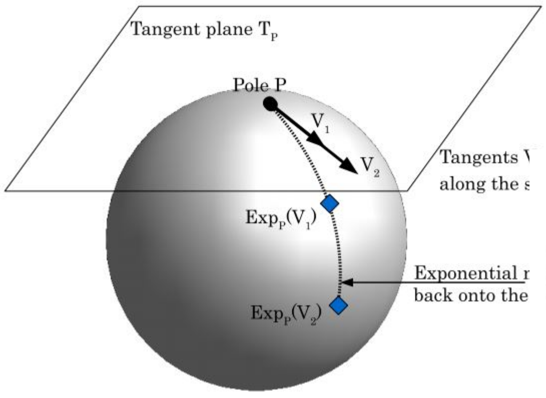
$$d(a,b) = inf L[\alpha]; \alpha(0) = a, \alpha(1) = b \tag{4.8}$$

### 4.2.3 Exponential and Inverse exponential maps

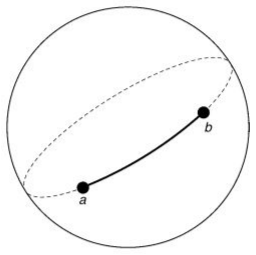
For a given Riemannian manifold , an exponential map is used to transform points from the tangent space to the manifold space, ie. for a point $p \varepsilon M$ : $T_p$M $\longrightarrow$ M mapping is given by $exp_p$(v) =$\alpha_v(1)$ where v denotes the velocity vector at p and $\alpha_v(1)$ is a constant speed geodesic. On the other hand an inverse exponential map helps to project points from the non-Euclidean manifold to the tangent space as shown in Figure 4.6a.

### 4.2.4 Mean computations

Since we perform Piecewise Aggregation (PAA) while approximating sensor data to symbols as discussed in chapter 3, computations of mean in the non-Euclidean manifold space has to be studied. Generally there are two significant ways to compute mean statistics. They are **Intrinsic and Extrinsic Statistics**



(a) Mapping



(b) Geodesic Arc

Figure 4.6: Riemannian Manifold

*Intrinsic Statistics*

Mean on a Riemannian manifold is given by karcher mean. The karcher mean of a set of independent random samples q1, q2, q3..qk is given by the local minimizer of the function (Equation 4.9)

$$\rho(p) = \frac{1}{k}\Sigma_i = \frac{1}{k}d(p, q_i)^2 \tag{4.9}$$

An iterative algorithm to minimize the above function (Equation 4.9) from i=1,2...k yields the karcher mean.

*Extrinsic Statistics*

Extrinsic statistics simply involves statistics computation on the vector space to simplify calculations and projecting back the final results to manifold space M. The projection map from manifold M space to vector space V is given by Equation 4.10

$$\pi(v) = argmin_{p\varepsilon M}\|v - \varepsilon(p)\|^2 \tag{4.10}$$

where $\epsilon$ is the embedding factor used for proper projection. So now after projection mean is calculated as in the vector space domain as shown in Equation 4.11

$$\mu = \Pi(x); x = \int_V vf(v)dv \tag{4.11}$$

Here f(v) denotes the p.d.f (Probability density function of v). Since extrinsic mean computations involve less computational overload than intrinsic mean computations as seen above, they are used for mean computations in case of quaternion sample data from the smartphone.

### 4.2.5 Extrinsic Mean computations and Distance metrics with respect to unit quaternions

In case of 3D rotations, the most common and essential task that will be encountered is the measurement of the distance between two 3D rotations (between two unit quaternions). As unit quaternions +q and -q denote the same rotation, the distance function should take into account this ambiguity in quaternion representation. That is to say, for any two quaternions $q_1$ and $q_2$, the function should yield the minimum of the d($q_1$,$q_2$)-$\theta$1 or d($-q_1$,$q_2$)-$\theta$2 as shown in Figure 4.7. Hence, the metric is given by the absolute value of the inner dot product of the two unit quaternions $q_1$ , $q_2$ representing two different rotations (Equation 4.12) [12].

$$\phi(q_1, q_2) = arccos(|q_1.q_2|) \tag{4.12}$$

The above operation when implemented with unit quaternions involve 4 multiplications, 1 arccos operation and 1 square root operation. The usage of quaternions to represent 3 D rotations is computationally more efficient than using rotation matrices where the product of two rotation matrices itself will involve 27 multiplications. Also in terms of storage, rotation matrices requires space for 9 floating elements.

Rounding off error is one such common problem with rotation matrices, where successive multiplications can result in rotation matrices that are no longer orthogonal. Though round-off errors are also found to occur in unit quaternion multiplication, it is more easy to renormalize the quaternion product vector than to reorthogonalize a distorted matrix [12].

To compute the average of the set of unit quaternions, as required by Piecewise Aggregate Approximation, we implement the following extrinsic mean computations:

The mean of a set of unit quaternions q1, q2, q3...qn is given by the rank-1 singular value decomposition (SVD) of the arithmetic mean of the matrix representations of the unit quaternions given by $q_i q_i^T$.(Equation 4.13)

$$\overline{M} = \frac{1}{n}\Sigma_{i=1}^{n} q_i q_i^T \tag{4.13}$$

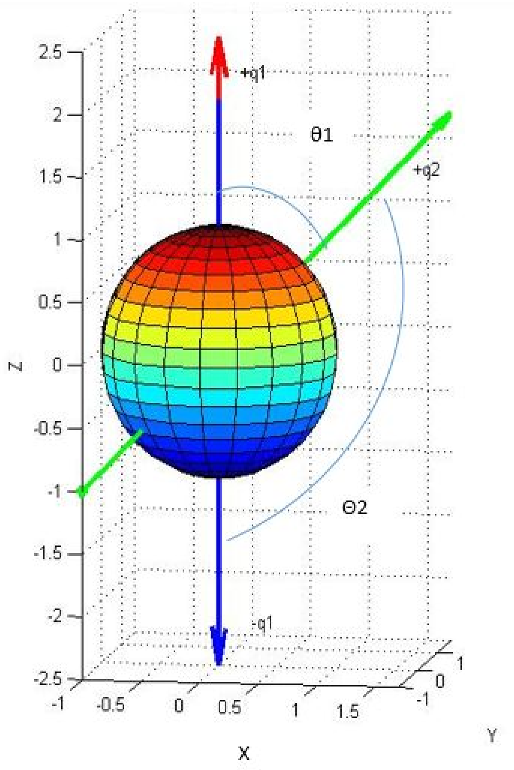


Figure 4.7: Distance between two quaternions q1 and q2

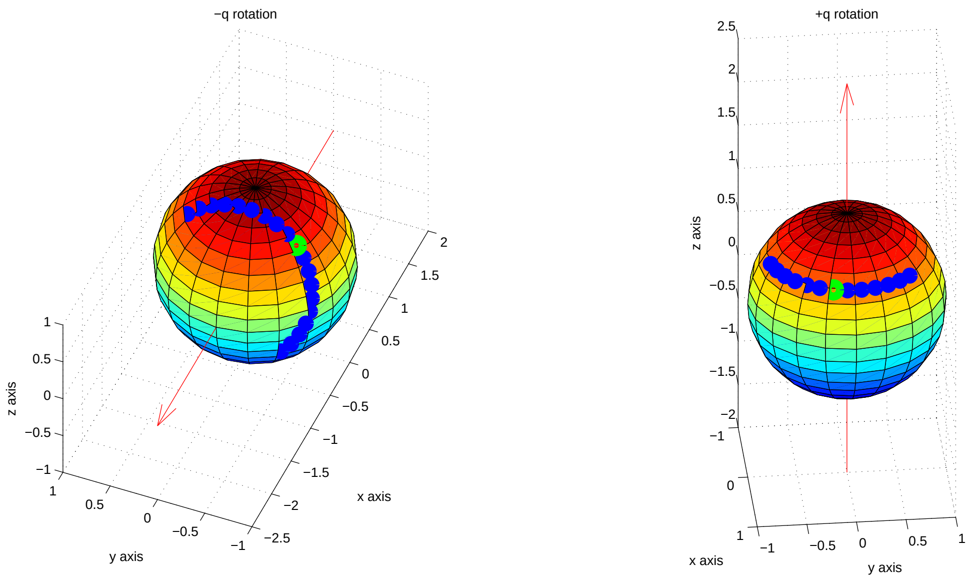


Figure 4.8: Mean Computations: Quaternion Space

$$\overline{M} = U\Sigma V^T \tag{4.14}$$

$U$ which is a vector representing the singular values of the decomposition, yields the computed average quaternion.

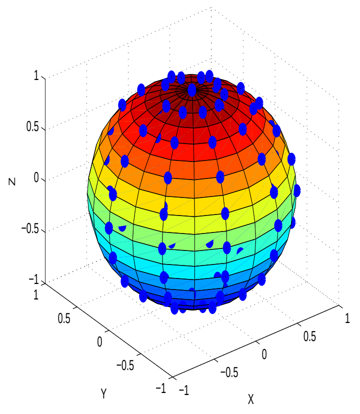
Figure 4.8 shows the mean computations (denoted by green marker) of the quaternions, which operate on an arbitrary point [a b c], producing rotations about X and Z axes.

### 4.2.6 Uniform & Trained symbol set

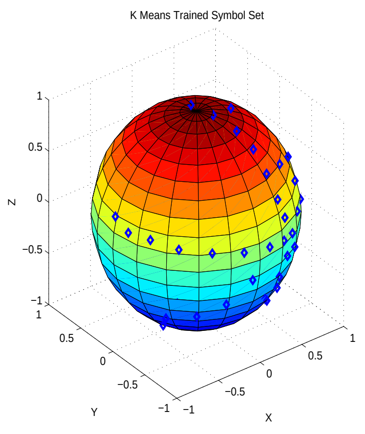
As discussed in chapter 3, to define symbols/alphabets (to compress the sampled data) on a manifold space, two different approaches are considered.

1. Uniformly Distributed Symbol set

2. Trained Symbol set



(a) Uniform



(b) Trained Symbol set

Figure 4.9: Symbol set

*Uniformly Distributed Symbol set*

Considering the quaternion space, levels or symbols are chosen so as to get them uniformly distributed about the entire sphere as shown in Figure 4.9a. As shown in the Figure 4.9a , the 100 levels are uniformly spread across the entire manifold. The prime advantage of this method is that there is no need for a training data to form the levels as compared to K-means clustering of levels.

*Trained Symbol set*

In this case, the symbol or alphabet set is customized according to the sensor data pertaining to activities from an initial training set. As a result, the symbols may be concentrated only on a particular region of the manifold (see Figure 4.9b) depending upon the domain application. The main advantage of such trained symbol set is to represent the non-Euclidean manifold time series with minimum most symbol set length L with a little trade off in terms of training time in the initial phase.

Generally the pairwise distances as given by Equation 4.12 are computed between all possible combinations of symbols in the alphabet set to generate a look up table. This look up table is referenced while obtaining the distance between the approximated sequences, thereby leading to less burden on CPU computations.

### 4.2.7 Distance between two manifold sequences

The closeness of a manifold sequence (corresponding to a particular activity) with another can be obtained through a nearest neighbor search where the distance metric is given by the sum of minimum distances function $D_{min}$ (Equation 4.15 ). The function takes into account distances between each point $e$ in one manifold sequence $S_1$ and the entire other manifold sequence set $S_2$.

$$D_{min}(S_1, S_2) = \frac{1}{2}(\sum_{e\varepsilon S_1} \Delta_m(e, S_2) + \sum_{e\varepsilon S_2} \Delta_m(e, S_1)) \tag{4.15}$$

where $\Delta_m(x, S)$ returns the distance of the closest point of S for x.

Chapter 5

# RESULTS AND DISCUSSION

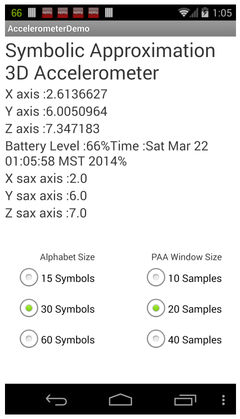
In this section, we conduct experiments on available datasets to prove the potential savings in battery life of the smartphone that can be acheived through signal approximation and the consideration of the geometric constraints of the rich sensor data for efficient classification results.

## 5.1 Effects of varying approximation parameters

### 5.1.1 Battery usage

Before the activity recognition experiments, studies on battery usage were conducted for the choice of pairs of signal approximation parameters like PAA window size and alphabet set size. The screen shot of the Java-Android app developed for this purpose is shown in Figure 5.1. The developed algorithm samples the tri-axial accelerometer values at 20 Hz (X,Y & Z axes), applies symbolic approximations as discussed in chapter 3 with user defined PAA window size and alphabet set size. Basic time-domain features like mean, standard deviation, minimum & maximum values etc., are computed and the app keeps track of the smartphones battery level and records it against time in an external sd card.

The plot as shown in Figure 5.2 shows the battery usage (drain curves) for various window sizes chosen for Piecewise Aggregate Approximation. The symbol set size is kept constant at 20 symbols. It can be inferred from the figure that, smaller the PAA window size (say 5 samples), more frequently features are computed resulting in an increased CPU/battery usage than when compared to a slightly larger window



Symbolic Approximation
3D Accelerometer
X axis :2.6136627
Y axis :6.0050964
Z axis :7.347183
Battery Level :66%Time :Sat Mar 22
01:05:58 MST 2014%
X sax axis :2.0
Y sax axis :6.0
Z sax axis :7.0

| Alphabet Size | PAA Window Size |
|---|---|
| 15 Symbols | 10 Samples |
| [✓ 30 Symbols] | [✓ 20 Samples] |
| 60 Symbols | 40 Samples |

Figure 5.1: Android App

size (20 samples). However, on other hand, increasing the window size further (say 60 samples) will lead to loss in classification accuracy as features are computed less frequently.

Hence, an optimum PAA window size of about 20 samples is chosen for the approximation.

The effect of varying the alphabet set size is shown in Figure 5.3 . For a given PAA window size (20 samples) if the alphabet set size is increased, the battery usage will be more as more number of comparisons has to be made to associate a sample

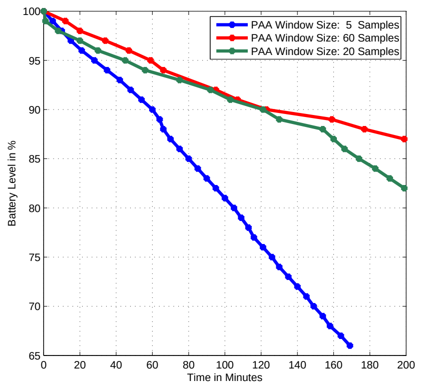


Figure 5.2: Battery Usage: PAA Window Size variation

with a symbol from the symbol set.

The optimum choice of the approximation parameters goes in par with the recognition accuracy studies conducted below.

### 5.1.2 Recognition accuracy

To support the studies on approximation parameters, we have considered the WISDM Actitracker dataset [15] based time series data for performance evaluation in terms of recognition accuracy. The statistics of the dataset are given in the Figure 5.4. The basic activities considered in this dataset include walking, sitting, standing, jogging & climbing. The sensor data include tri-axial accelerometer sampled at about 20 Hz. Subjects carried the smartphone in their front pant's leg pocket.

Table 5.1 shows the per class recognition accuracy under the various test conditions. Class accuracies (hereafter in this document) are given by the area under the Region of Convergence (ROC) curve. Firstly, the classifier efficiency with a J48 Tree classifier implemented on the raw dataset (i.e without implementing approximations) was found to be 92.98%.The time series data is then subject to symbolic approxima-

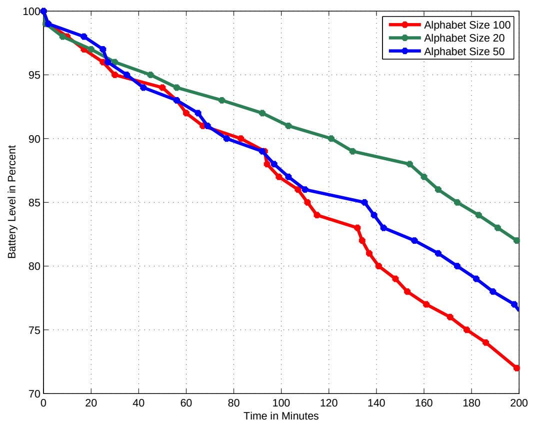


Figure 5.3: Battery Usage: Alphabet Set Size variation

Table 5.1: Recogntion Accuracy: Varying PAA window size

| Series | Walking | Jogging | Stairs | Sitting | Standing | **Overall** |
|---|---|---|---|---|---|---|
| Original Time Series | 95.7% | 98.7% | 90.7% | 98.9% | 98.3% | **92.98%** |
| Win Size: 5 samples | 92.5% | 95.2% | 84.4% | 98.7% | 98.4% | **88.008%** |
| Win Size: 20 samples | 87.8% | 93.3% | 78.7% | 97.8% | 89.6% | **81.37%** |
| Win Size: 40 samples | 86.9% | 90.1% | 76.3% | 98.1% | 84.6% | **77.58%** |

tion. We can find that the drop in recognition accuracy is insignificant when the data is subject to approximation. However, it can be seen that for a constant alphabet set size of 20 symbols, the recognition accuracy drops as expected as PAA window size increases. To maintain a balance between battery life and classifications results, we chose an optimum value of window size to be 20 samples.

Now to choose the next approximation parameter the alphabet size, the same experiment is conducted as above for various alphabet set sizes. Symbols are learned

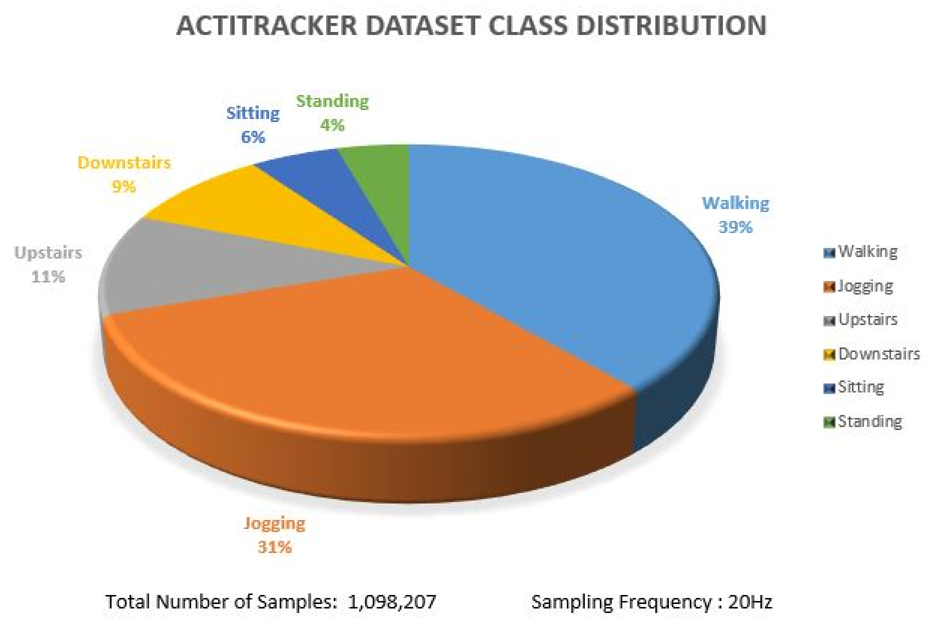


Figure 5.4: Actitracker Dataset and its Class Distribution

Table 5.2: Recogntion Accuracy: Varying alphabet set size

| Series | Walking | Jogging | Stairs | Sitting | Standing | **Overall** |
|---|---|---|---|---|---|---|
| Original Time Series | 95.7% | 98.7% | 90.7% | 98.9% | 98.3% | **92.98%** |
| Alphabet Size: 15 symbols | 81.6% | 86.9% | 59.5% | 93.5% | 66.3% | **78.96%** |
| Alphabet Size: 30 Symbols | 87.8% | 93.3% | 78.7% | 97.8% | 89.6% | **81.37%** |
| Alphabet Size: 60 symbols | 88.6% | 93.7% | 80.3% | 98.1% | 90.4% | **82.6%** |

from a subset of the data through K-means clustering. Figure 5.5 shows a sample set of 15 learned symbols. From Table 5.2 we can find that larger the symbol set size better will be the classification results as the time series data will be best approximated. However, tradeoff lies interms of battery life as seen from Figure 5.3. Hence, again an optimum alphabet set size of about 30 symbols is chosen.

#### 5.1.2.1 Sensor information

In order to demonstrate the use of orientation sensor to identify orientation related activities like walk forward, walk left, walk right etc., we have applied again the time series approximation as before with the chosen approximation parameters to another dataset (USC-Human Activity Dataset). The dataset [24] typically include most

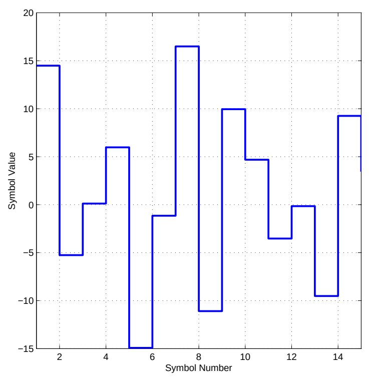


Figure 5.5: Trained Symbol set

Table 5.3: Recognition Accuracy:Sensor Information

| Sensors | **Walk-L** | **Walk R** | **Walk-up** | Run | **Walk-S** | Jump | Sit | **Overall** |
|---|---|---|---|---|---|---|---|---|
| Acc+ Gyro | **94%** | **94.7%** | **90.9%** | 99.4% | **91.3%** | 100% | 98.8% | **84.3%** |
| Acc | **88.2%** | **89.8%** | **88.2%** | 99.4% | **88.9%** | 99.3% | 99.7% | **81.62%** |

common low-level daily activities of humans. The data is captured by an interial sensing device called motion node which integrates a 3 axis accelerometer and a 3 axis gyroscope both of which are sampled at 100 Hz. The sensor device is worn on the front right hip of the subject & the activities include: walk forward, walk left, walk right, go upstairs, go downstairs, run forward, jump up and down, sit and fidget, stand, sleep, elevator up, and elevator down and with a total of 5 trials per activity. The detailed analysis as shown in table 5.3 is performed with and without taking into consideration information from gyroscope sensor.

From Table 5.3, we find that orientation related activities like walking (left-L, right-R, upstairs-up, straight-S) require gyroscope information inorder to be classified better. Hence, among the orientation sensing methods, we chose to use unit quaternions as a measure to identify true orientation of the smartphone as opposed to other likely measures like euler angles, rotation matrices which suffers from gimbal lock & computational complexities.

## 5.2 Human gesture recognition experiment

To demonstrate the idea of considering the non-Euclidean geometry while processing unit quaternions , we have designed a smartphone application which would transmit via bluetooth the quaternion information at about 20Hz to the server PC. The quaternions are initially normalized to turn them to unit quaternions & further processed offline using Matlab.

Five different gestural activities as shown in Figure 5.6 are performed with each activity repeated for a period of 150 seconds (3000 samples). To trace the movement of a point on the spherical manifold as manipulated by the quaternions we have assumed an arbitrary point [a,b,c] on the manifold & rotated this point using the sampled quaternions as given by Equation 4.4. The traces of different activities can be found in Figure 5.6. Symbolic approximations are applied on this evolving manifold sequences where the moving window size is taken to be 150 samples and thus a total of 70 instances (5 activities * 14(instances per activity) is obtained. (The smartphone CPU usage as per the different choice of the approximation parameters is shown in Figure 5.7). While doing PAA, extrinsic statistics are used as they are computationally less expensive. Approximation of the aggregated manifold sequences as shown in Figure 5.6(middle row) is then performed using both learned symbols (through K-means) and uniformly distributed levels on the manifold to study their implications . The distance metric considered in this case(Equation 4.12) preserves the non-Euclidean geometry.

The approximated manifold sequences are classified using nearest neighbor algo-

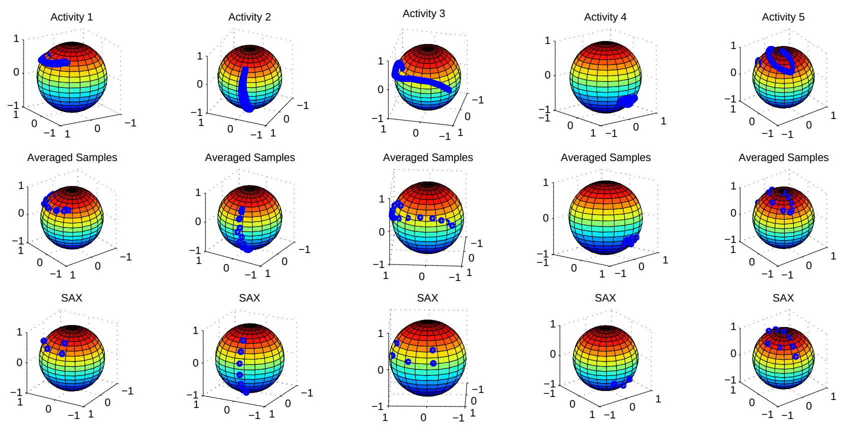
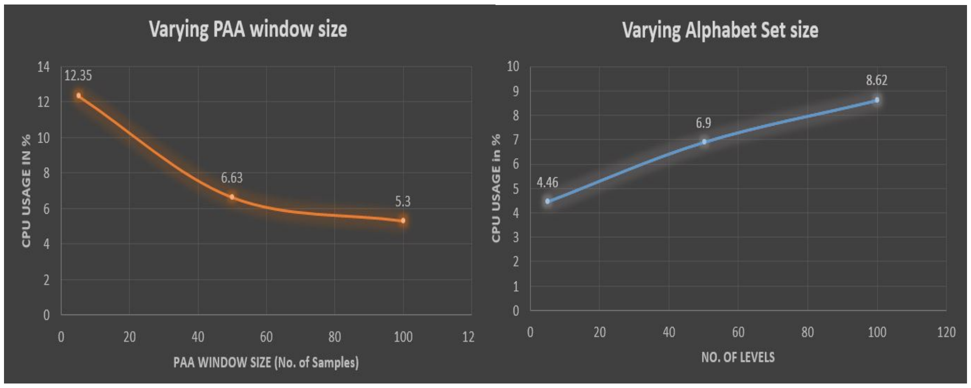
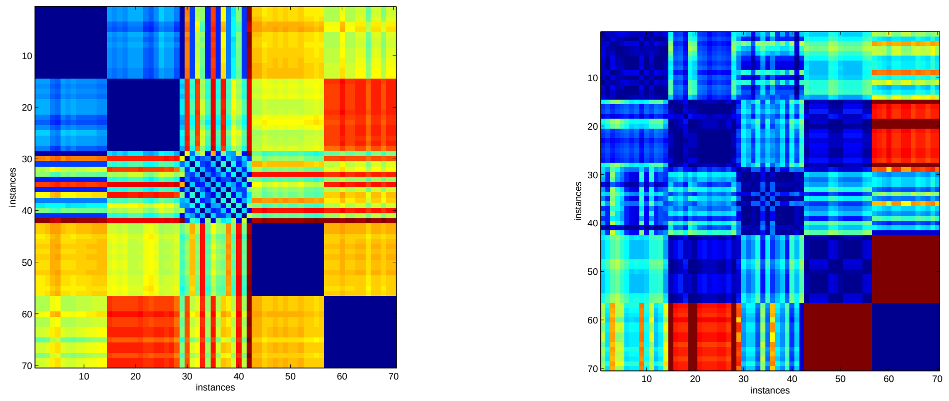


Figure 5.6: Gestures: Motion Trace



(a) PAA Window size (b) Alphabet set size

Figure 5.7: Effects of varying Approximation parameters on Battery Usage



(a) Trained Symbols (b) Uniformly Distributed Symbols

Figure 5.8: Pairwise Distance plots: Rotation Information

rithm which has *sum of minimum distances* between the sequences compared as the distance metric (Equation 4.15). A final pairwise distance measure plot as shown in Figure 5.8 is constructed for a 10 symbol case.

From Recognition Accuracy Table 5.4, it is obvious that for a given fixed alphabet set size (10 symbols), symbols learned through K-means better classifies the activites than uniformly distributed symbols on the manifold. Figure 5.8 further emphasizes

Table 5.4: Recognition Accuracy(Rotation Information).Symbol set size=10.

| Symbols (10 Nos) | Activity 1 | Activity 2 | Activity 3 | Activity 4 | Activity 5 | **Overall** |
|---|---|---|---|---|---|---|
| Trained | 97.6744% | 97.6744% | 64.78% | 97.6744% | 97.6744% | **91.09%** |
| Uniformly Distributed | 85.71% | 76.7% | 67.7% | 59.8% | 97.67% | **77.54%** |

Table 5.5: Recognition Accuracy(Acceleration Information).Symbol set size=10.

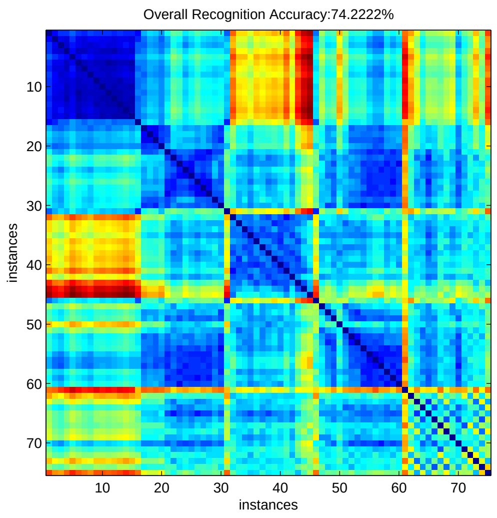
| Symbols (10 Nos) | Activity 1 | Activity 2 | Activity 3 | Activity 4 | Activity 5 | **Overall** |
|---|---|---|---|---|---|---|
| Trained | 91.2% | 81.1% | 65.3% | 75.6% | 70.8% | **62.16%** |
| Uniformly Distributed | 81.9% | 49.3% | 69.2% | 61.2% | 56.4% | **41.44%** |

the same. This is because K-means better distributes the symbols, based on the learned data on the manifold. Whereas in the case of uniform distribution, a major set of symbols in a region of the manifold might go unused while representing the evolving manifold sequences, thereby leading to a chance that two closely similar sequences being assigned the same set of symbols, leading to a drop in classifier accuracy.

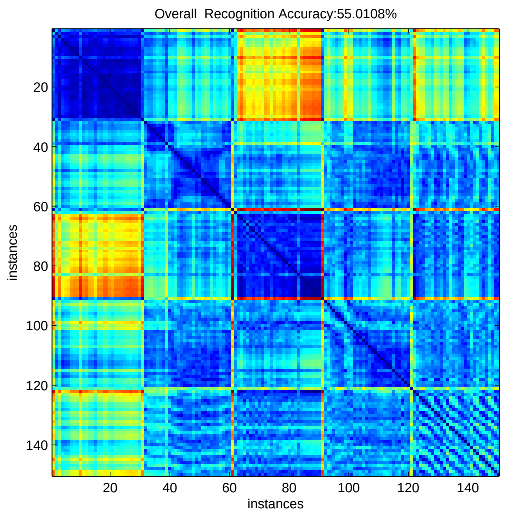
To emphasize the importance of the features from rotational data for better classification of the gestures, we conducted the classification again using the same Nearest Neighbor algorithm after extracting 1D features (as discussed in Chapter 3) from the magnitude component of the accelerometer data. The obtained results (for Symbol set sizes 10 & 30) are shown in Table 5.5 & 5.6. Figure 5.9 indicates the pairwise distance plots, where the approximated time series sequences are classified using nearest neighbor algorithm. The distance metric is the summation of the absolute distances between the pairs of points of the sequences. Thus evaluating the classification using the same classifier, we find that features obtained from quaternions have much more significance in gesture recognition than merely considering the acceleration data.

Table 5.6: Recognition Accuracy(Acceleration Information).Symbol set size=30.

| Symbols (30 Nos) | Activity 1 | Activity 2 | Activity 3 | Activity 4 | Activity 5 | **Overall** |
|---|---|---|---|---|---|---|
| Trained | 96.8% | 94.6% | 92.7% | 88.80% | 79.4% | **84.73%** |
| Uniformly Distributed | 95.2% | 92.4% | 77.8% | 83.2% | 73.6% | **73.28%** |



(a) Trained Symbols



(b) Uniformly Distributed Symbols

Figure 5.9: Pairwise Distance plots :Acceleration Information

# Chapter 6

# CONCLUSION AND FUTURE WORK

In this thesis, while focusing towards smartphone based human activity recognition, we have identified the importance & the need for symbolic representation of signals with better choice of approximation parameters to reduce computational overhead and battery usage. The same is demonstrated through the developed android application. In addition, the non-Euclidean manifold geometry of the sensor signals is considered to extract a better feature set for classification rather than considering them to be a single dimensional time series data. As an instance, unit quaternions are considered in this case for a human gesture recognition experiment and better results are reported. We have also analyzed the significance of a trained symbol set over an uniformly distributed symbol set on the manifold to better approximate the evolving manifold sequences for each activity performed and thereby achieve better classification results.

As a future work, apart from considering the better choice of approximation parameters like PAA window size and symbol set size, instead of representing symbols as points in an N dimensional manifold, they can be better represented through line or sparse representations [22] which accounts for most of the information on the original signal & represents the same in a compact sense suitable for high performance computing. These representations can be constructed from an over-complete dictionary of symbols, by which we mean, the number of symbols in the dictionary exceeds the maximum dimension of the signal so that the signal is efficiently approximated. These representations will definitely help in further enhancing the recognition accuracy results.

## REFERENCES

[1] *Naive Bayes Classifier.* StatSoft Electronic Statistics Textbook, .

[2] R. Anirudh. Low complexity differential geometric computations with applications to human activity analysis. Master's thesis, Arizona State University, 2012.

[3] Ling Bao and Stephen S. Intille. Activity recognition from user-annotated acceleration data. pages 1–17. Springer, 2004.

[4] Eugen Berlin and Kristof Van Laerhoven. Detecting leisure activities with dense motif discovery. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 250–259, New York, NY, USA, 2012. ACM.

[5] Luis Gerardo Mojica de la Vega, Suraj Raghuraman, Arvind Balasubramanian, and Balakrishnan Prabhakaran. Exploring unconstrained mobile sensor based human activity recognition, 2011.

[6] B. De Ville. *Decision Trees for Business Intelligence and Data Mining: Using SAS Enterprise Miner.* SAS Press series. SAS Institute, 2006.

[7] Angelo Cappello Lorenzo Chiari Kamiar Aminian Jeffrey M. Hausdorff Wiebren Zijlstra Jochen Klenk Fabio Bagal, Clemens Becker. Evaluation of accelerometer-based fall detection algorithms on real-world falls, 2012.

[8] Davide Figo, Pedro C. Diniz, Diogo R. Ferreira, and João M. Cardoso. Preprocessing techniques for context recognition from accelerometer data. *Personal Ubiquitous Comput.*, 14(7):645–662, October 2010.

[9] I.C. Gyllensten and A.G. Bonomi. Identifying types of physical activity with a single accelerometer: Evaluating laboratory-trained algorithms in daily life. *Biomedical Engineering, IEEE Transactions on*, 58(9):2656–2663, Sept 2011.

[10] Illapha Cuba Gyllensten. *Physical activity recognition in daily life using a tri-axial accelerometer.* PhD thesis, Masters thesis, Royal Institute of Technology, Stockholm, Sweden, 2010.

[11] Andrew J. Hanson. Visualizing quaternions. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[12] Du Q. Huynh. Metrics for 3d rotations: Comparison and analysis. *J. Math. Imaging Vis.*, 35(2):155–164, October 2009.

[13] T. Kobayashi, K. Hasida, and N. Otsu. Rotation invariant feature extraction from 3-d acceleration signals. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 3684–3687, May 2011.

[14] M. Kose, O.D. Incel, and C. Ersoy. Performance evaluation of classification methods for online activity recognition on smart phones. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pages 1–4, April 2012.

[15] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.*, 12(2):74–82, March 2011.

[16] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[17] Kiran K. Rachuri, Cecilia Mascolo, Mirco Musolesi, and Peter J. Rentfrow. Sociablesense: Exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, pages 73–84, New York, NY, USA, 2011. ACM.

[18] Nishkam Ravi, Nikhil D, Preetham Mysore, and Michael L. Littman. Activity recognition from accelerometer data. In *In Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence(IAAI*, pages 1541–1546. AAAI Press, 2005.

[19] Emmanuel Munguia Tapia, Stephen S. Intille, William Haskell, Kent Larson, Julie Wright, Abby King, and Robert Friedman. Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart monitor. In *IN: PROC. INT. SYMP. ON WEARABLE COMP*, 2007.

[20] Pavan Turaga, Ashok Veeraraghavan, Anuj Srivastava, and Rama Chellappa. Statistical analysis on manifolds and its applications to video analysis. In *Video Search and Mining*, pages 115–144. Springer, 2010.

[21] Silas Wan and Hung T. Nguyen. Human computer interaction using hand gesture. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 2357–2360, Aug 2008.

[22] J. Wright, Yi Ma, J. Mairal, G. Sapiro, T.S. Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, June 2010.

[23] Zhixian Yan, V. Subbaraju, D. Chakraborty, A. Misra, and K. Aberer. Energy-efficient continuous activity recognition on mobile phones: An activity-adaptive approach. In *Wearable Computers (ISWC), 2012 16th International Symposium on*, pages 17–24, June 2012.

[24] Mi Zhang and Alexander A. Sawchuk. Usc-had: A daily activity dataset for ubiquitous activity recognition using wearable sensors. In *ACM International Conference on Ubiquitous Computing (Ubicomp) Workshop on Situation, Activity and Goal Awareness (SAGAware)*, Pittsburgh, Pennsylvania, USA, September 2012.