Development of a human activity recognition system using inertial measurement unit sensors on a smartphone

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of
Masters of Applied Science

University of Ottawa

DEPARTMENT OF MECHANICAL ENGINEERING
Ottawa-Carleton Institute for Mechanical Engineering
University of Ottawa

ABSTRACT

Monitoring an individual's mobility with a modern smartphone can have a profound impact on rehabilitation in the community.

The thesis objective was to develop and evaluate a third-generation Wearable Mobility Monitoring System (WMMS) that uses features from inertial measurement units to categorize activities and determine user changes-of-state in daily living environments. A custom suite of MATLAB® software tools were developed to assess the previous WMMS iteration and aid in third-generation WMMS algorithm construction and evaluation.

A rotation matrix was developed to orient smartphone accelerometer components to any three-dimensional reference, to improve accelerometer-based activity identification. A quaternion-based rotation matrix was constructed from an axis-angle pair, produced via algebraic manipulations of acceleration components in the device's body-fixed reference frame.

The third-generation WMMS (WMMS3) evaluation was performed on fifteen able-bodied participants. A BlackBerry Z10 smartphone was placed at a participant's pelvis, and the device was corrected in orientation. Acceleration due to gravity and applied linear acceleration signals on a BlackBerry Z10 were then used to calculate features that classify activity states through a decision tree classifier. The software tools were then used for offline data manipulation, feature generation, and activity state prediction.

Three prediction sets were conducted. The first set considered a "phone orientation independent" mobility assessment of a person's mobile state. The second set differentiated immobility as sit, stand, or lie. The third prediction set added walking, climbing stairs, and small standing movements classification. Sensitivities, specificities and F_1-Scores for activity categorization and changes-of-state were calculated.

The mobile versus immobile prediction set had a sensitivity of 93% and specificity of 97%, while the second prediction set had a sensitivity of 86% and specificity of 97%. For the third prediction set, the sensitivity and specificity decreased to 84% and 95%

respectively, which still represented an increase from 56% and 88% found in the previous WMMS.

The third-generation WMMS algorithm was shown to perform better than the previous version in both categorization and change-of-state determination, and can be used for rehabilitation purposes where mobility monitoring is required.

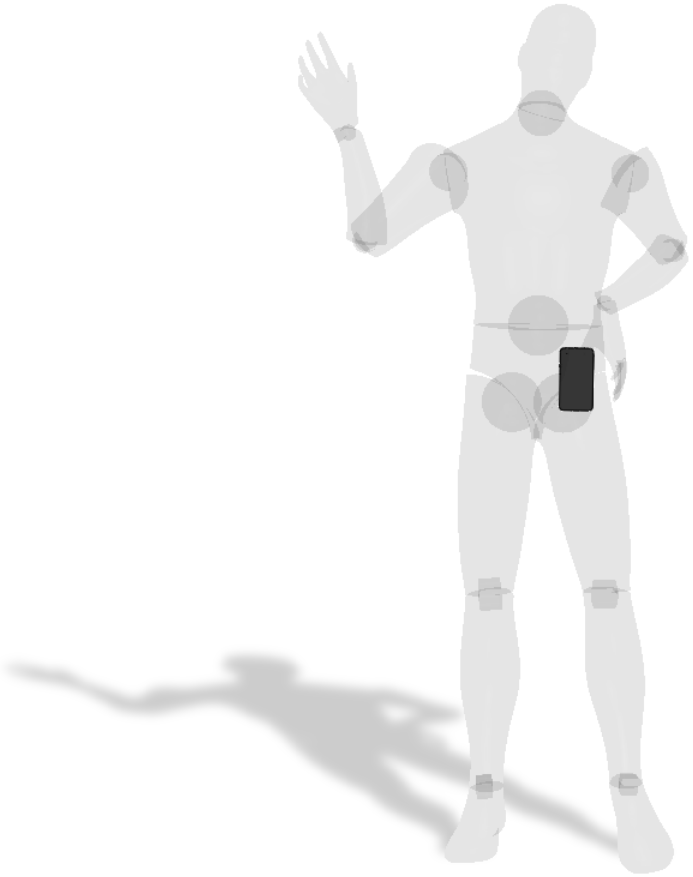
ACKNOWLEDGEMENTS

I would like to thank my supervisors Dr. Natalie Baddour & Dr. Edward Lemaire for their constructive criticism, guidance, and support throughout my research.

I would also like to thank the kind staff at the Ottawa Hospital Rehabilitation Centre for their help and accommodations, as well as BlackBerry Ltd. for their financial support and for their supply of all devices used in the research study.

I would like to thank my family and friends for their understanding, encouragement and patience throughout my journey.

Finally, a special thank you for Ms. Cynthia Lopez, who had to listen to me blabber about quaternions and four-dimensional worlds for two years straight. I love you.



PUBLICATIONS

M. D. Tundo, E. Lemaire, and N. Baddour, "Correcting Smartphone orientation for accelerometer-based analysis," in *2013 IEEE International Symposium on Medical Measurements and Applications Proceedings (MeMeA)*, 2013, pp. 58–62.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

ADG: Acceleration Due to Gravity

ALA: Applied Linear Acceleration

API: Application Programming Interface

BB10: BlackBerry OS 10

BB7: BlackBerry OS 7

CoS: Change-of-State

CPU: Central Processing Unit

FFT: Fast Fourier Transform

FN: False Negative

FP: False Positive

GUI: Graphical User Interface

HAR: Human Activity Recognition

I/O: Input/Output

IMU: Inertial Measurement Unit

kNN: k-Nearest Neighbour

MARG: Magnetic, Angular Rate, Gravity

MEMS: Microelectromechanical Systems

OS: Operating System

PIC: Programmable Interface Controller

RAM: Random Access Memory

TN: True Negative

TOHRC: The Ottawa Hospital Rehabilitation Centre

TOL CAT: Categorization Tolerance

TOL COS: Change-of-State Tolerance

TP: True Positive

WMMS: Wearable Mobility Monitoring System

Chapter 1
INTRODUCTION

Ubiquitous sensing has become an engaging area of research in the past decade due to increasingly powerful, small size, and low cost computing and sensing equipment [1]. Particularly, sensors that recognize human activity have become an interest in the medical, military, and security fields.

In the medical field, people with mobility disabilities or illnesses such as heart disease or diabetes might require mobility pattern monitoring, providing the healthcare professional with an inventory of the person's activity in a daily living environment, outside of a clinical setting. Mobility is a requirement for activities of daily life, including movements around the house and in the community. Disabilities that affect mobility may severely impact an individual's quality of life and can increase dependence on others for completing common daily tasks, such as cooking or getting dressed [2]. Statistics Canada reported that the percentage of elderly (65 years and over) in Canada is drastically increasing, with this percentage doubling by 2061 [3]–[5]. Health monitoring in the home could help deal with health issues for this increasing senior population. Ubiquitous sensing could determine when medical or care intervention would be beneficial. In addition, military-based applications could monitor soldier mobility in real-time, for health status updates and training scenarios. Military sensing can include crawling, kneeling, or situation assessments, which can be critical for military missions. Security fields may use real-time mobility tracking for personnel on foot.

Information on a person's activity level can account for energy expenditure and overall physical activity. This leads to applications that promote physical exercise and can track activity history to provide useful statistics [6]. Elderly individuals could benefit from ubiquitous activity sensing where data is relayed to the physician or clinician for review. This activity information could be an indicator of the person's overall mobility status and cognitive well-being [7].

1.1 Objectives

The thesis objective was to develop and evaluate a third generation Wearable Mobility Monitoring System (WMMS), designed for a smartphone that includes a tri-axis accelerometer, gyroscope, and magnetometer. The new mobility algorithm must include calibration measures to account for differences in body proportions. This new algorithm must also reduce the number of false positives for changes-of-state identification and increase sensitivity in activity categorization. The algorithm must perform better overall in sensitivity and specificity over its predecessor.

1.2 Outline

This thesis is organized into six chapters. The first chapter includes the objectives, outline, and contributions to Human Activity Recognition. The second chapter reviews existing work in the field of mobility monitoring. Both wearable sensors and smartphone use as a mobility monitor are covered. The third chapter describes the previous two WMMS versions and includes an in-depth review of the WMMS2 algorithm. The fourth chapter reports on a new evaluation of WMMS2, including a new rotation correction applied to the accelerometer data. The fifth chapter describes the software developed to evaluate the WMMS2 and WMMS3 algorithms. The sixth chapter introduces the WMMS3 algorithm, which uses the Blackberry Z10 hardware and BlackBerry OS 10 (BB10) operating system, and discusses evaluation results. The seventh chapter includes concluding remarks. A bibliography and appendices are then included.

1.3 Contributions

Several contributions in the field of smartphone-based Human Activity Recognition (HAR) can be noted from this thesis research. First, a calibration correction using a quaternion rotation matrix was shown to decrease device orientation variability between users. The rotation may then be applied to different motion sensors, such as the accelerometer, magnetometer, or gyroscope. This work is shown in Chapter 4.

A fully featured software tool was developed to efficiently process and display evaluation outcomes for a Human Activity Recognizer, namely the WMMS. The tool can parse and organize large motion data sets and provide graphics to assist the user when building motion recognition algorithms. The tool assists the user when investigating quantitative correlations between mathematic manipulations of raw sensor data and human movement. The program's modularity facilitates future development, hence increasing its effectiveness for future work. The tool is described in Chapter 5.

A standardized WMMS evaluation method was used, that includes continuous activity transitions in a realistic daily living environment. Testing with realistic but untargeted activities decreased recognition sensitivity. Therefore, realistic testing circuits are encouraged for valid evaluation of WMMS performance. The standardized evaluation with the third generation WMMS algorithm is reviewed in Chapter 6.

Lastly, features extracted from the applied linear acceleration and acceleration due to gravity signals were used to increase sensitivity and specificity in a WMMS. Modern smartphone technology provides these signals to the developer in real-time, and the recognition of human motion can take advantage of this innovation.

Chapter 2
LITERATURE REVIEW

2.1 Mobility monitoring

Mobility monitoring using wearable sensors has generated a great deal of interest due to the development of consumer microelectronics capable of detecting motion characteristics with a high degree of accuracy [1]. Human Activity Recognition (HAR) using sensors is a fairly recent area of research, with preliminary studies in the 1980's and 1990's [8]–[10]. Wearable hardware can sense and process human movements in real-time and relay the information wirelessly over a variety of data networks.

Mobility monitoring systems use either sensing equipment fixed to stationary objects (external sensing) or sensors fixed to the person (inertial sensing). For both monitoring methods, wireless communication can be used for data transfer (i.e., Wi-Fi, Bluetooth, Zigbee, cellular, etc.) [1], [6].

With the external approach, sensing equipment is attached to predetermined points of interest. Sensing is based on the interaction between users and sensors. For example, intelligent homes may use cameras, radio-frequency identification devices (RFIDs), reed switches, pressure mats, passive infrared sensors (PIR), float sensors, or temperature sensors to achieve a nodal-based sensing system [11]. Video-based HAR systems, where humans are recognized by video analysis software, are also used extensively for security applications [12], [13].

HAR with wearable inertial sensors is less costly than external sensing, since less equipment is required. Wearable inertial sensors can be used to monitor mobility outside the home or laboratory, allowing for a full range of activities in the typical environment. Common HAR wearable sensors include accelerometers, but may also include gyroscopes, magnetometers, altimeters, microphones, and video cameras. This chapter will focus on inertial sensor methods.

2.2 Recognition methods

Many inertial sensor-based methodologies have been used for mobility recognition, termed ‘inertial HAR’ in this section. HAR systems employ three main subsystems: data collection, model parameters, activity recognition [6]. Inertial HAR systems interpret raw sensor data to classify a set of activities. An evaluation procedure, usually modeled as a binary classifier, is used to validate the prediction outcome with a gold-standard data set.

Data collection includes all raw sensor data required for the prediction algorithms, which is then manipulated by algorithm parameters. Algorithm parameters can include pre-calculated training data and threshold values for activity recognition. Considerations in the data sample rate, the amount of sensor drift, and overall signal quality play roles in shaping the inertial HAR algorithm.

Tri-axis accelerometers are effective sensors for activity recognition. The use of accelerometers in data collection greatly increased with the implementation of accelerometers in mobile devices. Initially, accelerometers were used for screen rotation detection and mobile gaming control. Mobile operating systems such as iOS, Android, or BlackBerry allow developers to access accelerometer data through application programming interfaces (APIs). Proximity and light sensors, GPS, and cameras have also been used in HAR applications [6]. Recently, some mobile phone manufacturers have included gyroscope, magnetometer, and altimeter sensors in their devices; however, these sensors are not yet used in mainstream HAR systems.

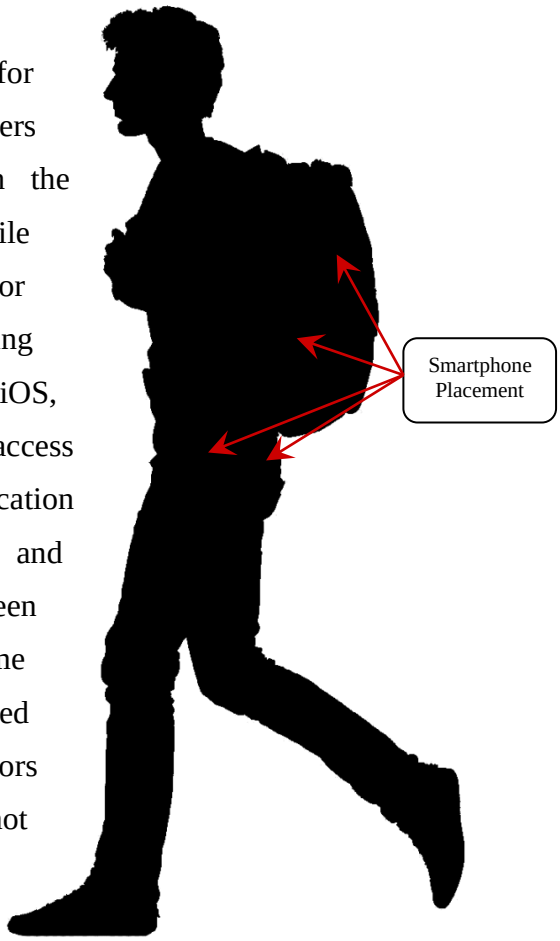


Figure 1: Common smartphone placements on the body.

Sensor placement (Figure 1) also plays a critical role in HAR. Most algorithms rely heavily on sensor position, whereas other methods attempt an orientation and placement-independent approach. However, raw data alone is not sufficient to detect mobility states. Preprocessing, segmentation and feature extraction, in conjunction with supervised learning procedures, can be used in a full HAR algorithm that detects human mobility activities (Figure 2).

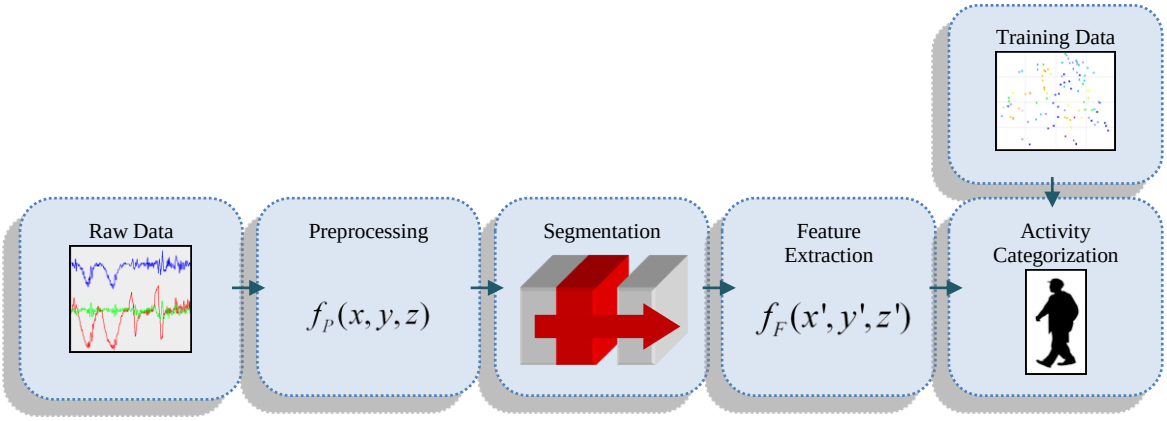


Figure 2: Common architecture of an inertial human activity recognizer.

Generally, the raw data is preprocessed to remove noise or artifacts. Then, segmentation defines a data window, usually as a smaller subset of a continuous data stream. Subsets may overlap in time, known as a sliding window. Features are then calculated from the segmented sensor readings, usually by statistical or frequency analysis of the raw data components. Examples of HAR features are shown in Table 1.

Table 1: HAR features [14].

Feature Type	Examples
Heuristic	Simple moving average, peak-to-peak acceleration, mean rectified value, root mean square
Time-Domain	Mean, median, standard deviation, skewness, kurtosis
Frequency-Domain	Fast Fourier Transform (FFT) before any time-domain feature
Wavelet Analysis	Hybrid of time and frequency analysis

Frequency-domain and wavelet analysis features are sometimes complex and processor intensive. Most algorithms employ some or all of the features described in equations 2.1 to 2.6.

Arithmetic Mean $$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \,, \tag{2.1}$$

Root Mean Square $$RMS(Y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} y_i^2} \,, \tag{2.2}$$

Standard Deviation $$\sigma_y = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} (y_i - \bar{y})^2} \,, \tag{2.3}$$

Variance $$\sigma_y^2 = \frac{1}{n-1}\sum_{i=1}^{n} (y_i - \bar{y})^2 \,, \tag{2.4}$$

Mean Absolute Deviation $$MAD(Y) = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} |y_i - \bar{y}|} \,, \tag{2.5}$$

Energy $$Energy(Y) = \frac{\sum_{i=1}^{n} F_i^2}{n} \,, \tag{2.6}$$

where n is the number of samples, and $y_1, y_2, y_3 ... y_n$ is the sample set.

Learning procedures can then be applied to the extracted features, to classify movement as specific activities. Machine learning tools are often used in HAR systems to describe, analyze, and predict variations in features.

Supervised learning methods can be composed of training and classification phases, and vary in their algorithms. The recognition algorithms can be tailored for the individual, or be flexible enough to work with different users of varying age, gender, weight, etc. Decision tree classifiers are the most commonly used learning procedure [2], [8], [15]–[26]. Bayesian networks are also commonly used as a learning model [16], [17], [21], [23], [24], [26] but, unlike a decision tree, probability factors and assumptions on feature independence are required. Support vector machines, a type of neural network, have been commonly used in modern HAR systems [16], [17], [23], [27]–[30], but at the expense of high computational cost and large training data requirements. A simple machine learning technique, k-Nearest Neighbours (kNN) [16], [17], [21], [23], [24], [31], [32] has also been useful for HAR systems, but at the expense of large training sets. Other learning methods include the multilayer perceptron [18], [26], fuzzy logic [24], [33]–[36], neural network models [37]–[39], and linear regression models [18].

Learning procedures can also follow a subject-dependent or subject-independent evaluation. In the former, classifiers are based on person-specific data; whereas, subject independent training uses a database of human motion and cross validation statistics, but usually leads to a decrease in accuracy.

HAR systems can attempt to categorize several activities (Table 2), depending on the type of sensors used, sensor placement, processing power, and performance.

Table 2: Common activities that HAR systems may predict [1], [6].

Category	Activity Examples
Still/Rest	Standing, sitting, lying down, kneeling, crouching
Ambulation	Walking, running, sprinting, crawling, small movements
Transportation	Bicycling, in a vehicle, public transit
Exercise and sports	Rowing, weight training, skating, soccer, push-ups
Smartphone specific	Texting, making a call, playing a game, browsing the web
Daily activities	Eating, drinking, on computer, watching TV, brushing teeth, combing hair, preparing food, cleaning/vacuuming
Health and rehabilitation	Following routines, using a wheelchair, using a walker, using a cane

Performance measures vary between studies, but binary classification tests are typically used. Binary classifiers outline two potential options, whether the user is performing a specified activity or not performing the activity. Binary classification measures include: accuracy, precision, recall, confusion matrices, and F-measure [6]. Binary classifiers are comprised of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). A summary and overview of these measures can be found in 5.5.3.2 Sensitivity & specificity.

2.3 Wearable sensors for mobility monitoring

HAR studies have used single or multiple wearable sensors, placed on various body locations, to detect human ambulation [40]. Methods for data recording, analysis, and evaluation vary between studies, as shown below (Table 3).

Table 3: Wearable sensor type, placement and evaluation procedures in the literature.

Year	Researcher(s)	Number of Accelerometers	Accelerometer Placement	Evaluation
1996	Veltink et al. [19]	6 x 1D IC sensors 5 g	Sternum, shoulder, thigh, shank	Protocol order with investigator logging
1997	Kiani et al. [38]	1 x 2D Unknown, 2 x 3D Unknown	Chest, thighs	Protocol order with investigator logging
2002	Luinge [41]	3 x 1D ADXL05	Upper back	Vicon™
2003	Mathie et al. [20]	2 x 2D Unknown	Waist	Investigator logging start/stop times
2003	Bao [21]	5 x 2D ADXL2010E	Hip, wrist, arm, ankle, thigh	Subject logging start/stop times
2004	Lyons et al. [22]	2 x 2D ADXL202	Thigh, sternum	Investigator logging start/stop times
2005	Ravi et al. [23]	1 x 3D CDXL04M3	Pelvis	Investigator logging start/stop times
2009	He et al. [30]	1 x 3D ADXL330	Trouser pocket	Unknown
2009	Helmi et al. [35]	1 x 3D LIS3LV02DQ	Waist	Unknown
2010	Hong et al. [42]	3 x 3D MMA7260Q	Thigh, waist, wrist	Investigator logging start/stop times
2010	Pärkkä et al. [25]	4 x 3D Nokia Wireless Motion Bands	Wrists, ankles	Subject logging start/stop times on PDA
2011	Lee et al. [39]	1 x 3D MMA7260Q	Sternum	Subject voice annotations
2012	Chiang et al. [36]	2 x Unknown	Right chest, thigh	Unknown
2013	Zhang et al. [37]	1 x 3D LIS3LV02DQ	Hand	Unknown

Inertial HAR, where multiple sensors are placed on the body (Figure 3), can be quite cumbersome and complex to design. Programmable interface controllers (PICs) are

required to achieve real-time monitoring, and could involve wireless networking modules, along with complex interfacing, programming, calibrating, and testing procedures.

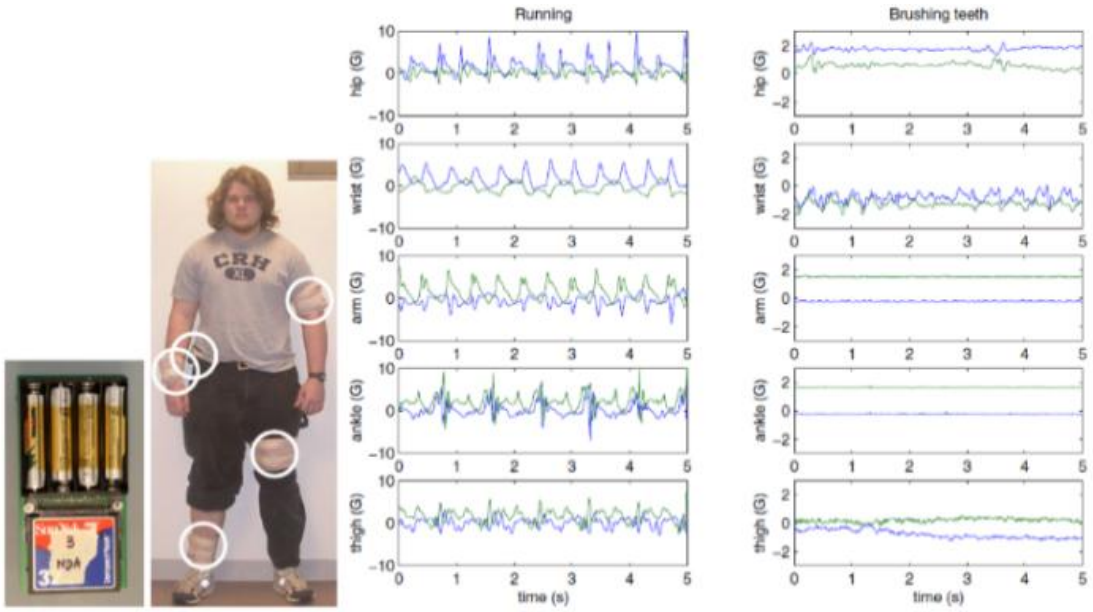


Figure 3: Sensor placement and accelerometer graphs (Bao) [21].

2.4 Smartphone adoption for research

Smartphones contain the necessary sensors and real-time computation capability for mobility activity prediction. Real-time analysis on the device permits quick activity classification and data upload without user or investigator intervention. Unlike wearable sensor methods, a smartphone contains a variety of embedded sensors, including inertial measurement units (IMUs) that can be costly and cumbersome to program and network individually. A smartphone with mobility analysis software could provide fitness tracking, health monitoring, fall detection, home or work automation, and/or, self-managing exercise applications [43]. Mobility assessment on a smartphone is limited by device specifications such as CPU power, sensor sample rate, video recording capabilities, and battery life.

Current smartphone models have performance specifications that rival laptop computers, such as the iPhone 5S A7 microprocessor or the 1.7 Ghz quad-core HTC One. The BlackBerry Z10 (Figure 4) integrates software and hardware to allow its 1.5 Ghz dual-

core CPU to perform at its highest potential, providing a large amount of computing power in the user's pocket. All 2013 smartphones include a large amount of RAM, 1 to 2 GB, which is sufficient for most computation, storage, and multitasking operations. Furthermore, solid-state storage can reach 64 GB, suitable for video, audio, database, or other large file types. Modern smartphone cameras are sufficient for research purposes, with at least 720p resolution video and high megapixel counts for front and rear cameras; however, photo and video quality vary between devices. Nonetheless, modern smartphone cameras take clear and large images. Wireless connectivity over LTE is the norm for 2013, with data transmission rates up to 150 Mbps. It is not expected that every phone can reach these speeds everywhere, but LTE does allow for large data file transmission over cellular networks.

Figure 4: BlackBerry Z10 smartphones.

These advancements in smartphone technologies are not without cost. Battery life has been average if not sub-par in relation to increases in all other hardware specifications. Smartphone lithium-ion batteries typically range from 1800 mAh to 3,100 mAh, and require hours to charge. Long duration applications that use smartphones and involve intensive CPU calculations or continuous video recording can be limited by battery capacity. Future generation battery technologies that allow fast charging with higher capacities will certainly allow long-term mobility research to be realized.

2.5 Smartphone use for mobility monitoring

Several activity recognition studies have used a smartphone accelerometer to detect activity states, as shown in Figure 5 and Table 4.

Figure 5: Real-time Human Activity Recognition application on a smartphone [28], [32], [44].

Table 4: Summary of Human Activity Recognition research studies.

Authors	Device	Placement	Features	Classifier	Activities	Accuracy
Brezmes [31]	Nokia N95	User-dependant	Unknown	k-Nearest Neighbour	Sitting, standing, walking, upstairs, downstairs, falling	20-90%
Berchtold [33]	Unknown	User-dependant	Mean, variance	FIS Mapping, Fuzzy Classification	Sitting, standing, lying, walking, upstairs, cycling, holding phone, talking on phone, typing message	60-98%
Sun [27]	Nokia N97	User-dependant	Mean, variance, correlation, FFT Energy, frequency-domain entropy	Support Vector Machine	Stationary, walking, running, upstairs, downstairs, cycling, vehicle	91-94%
Lau [16]	Nokia N95	Right pocket	Mean and standard deviation in time and frequency domain	Decision tree, Bayesian Network, Naïve Bayes, k-Nearest Neighbour, Support Vector Machine	Sitting, standing, walking, upstairs, downstairs	92-99%
Yang [17]	Nokia N95	User-dependant	Mean, variance, mean crossing rate, spectral energy and peaks, spectral energy ratio	Decision tree, Bayesian Network, Naïve Bayes, k-Nearest Neighbour, Support Vector Machine	Stationary, walking, cycling, running, vehicle	73-92%
Kwapisz [18]	Nexus One, HTC Hero, Motorola Backflip	Front pocket	Mean, standard deviation, average absolute difference, magnitude, peak time, binned distribution	Decision Tree, Logistic Regression, Multilayer Neural Perceptron	Sitting, standing, walking, jogging, upstairs, downstairs,	77-96%
Maruno [45]	Accelerometer mounted to mobile phone	Unknown	Wavelet transform with Singular Value Decomposition	Broyden-Fletcher-Goldfarb-Shanno quasi Newton Neural Network	Standing, walking, running, train	85-93%
Siirtola [32]	Nokia N8	Front pocket	Standard deviation, mean, minimum, maximum, sum, square sum	Quadric Discriminant Analysis, k-Nearest Neighbour	Stationary, walking, running, cycling, vehicle	91-99%
Fahim [34]	Samsung Galaxy S	Front pocket	Root mean square, variance, correlation, energy	Evolutionary Fuzzy Model	Walking, jogging, running, upstairs, downstairs, cycling, hopping	82-100%
Viet Vo [28]	HTC Nexus One	Front pocket	Mean, variance, peak time, energy	Support Vector Machine with k-Medoids Clustering	Walking, jogging, upstairs, downstairs, cycling	72-95%
He [44]	Samsung Nexus S	Chest	Average, mean, standard deviation, skewness, kurtosis, interquartile range, decline	Fisher's Discriminant Ratio	Sitting, lying, standing, walking, upstairs, downstairs, running, jumping, falling, transitions	91-97%

Brezmes et al. [31] used a N95 Nokia smartphone with a real-time classification system for pattern recognition algorithms to classify human movement. Euclidean distance and kNN algorithms were used for activity classification. The accelerometer sample rate was approximately 30 Hz, and required a training data set for activity recognition. Prediction accuracy increased as training sets increased.

In 2010, Sun et al. [27] proposed activity recognition using a smartphone's accelerometer, independent of position and orientation. The user could use varying pocket locations for the phone, including left and right sides of the body, trouser back pockets, or the front jacket pocket. Sliding window frames were used for feature extraction, which included mean, variance, energy, frequency-domain entropy, and correlation. These features were normalized within each window and vector classification, regression, and distribution estimation approaches were applied to convey mobility states. Predicted mobility states included stationary, walking, running, bicycling, ascending stairs, descending stairs, and driving. The results were presented as an F-Score, which is a weighted combination of sensitivity and precision of a correct prediction. An overall average F-Score of 93 % was achieved for unknown pocket positions. Much like Brezmes, a training algorithm was required.

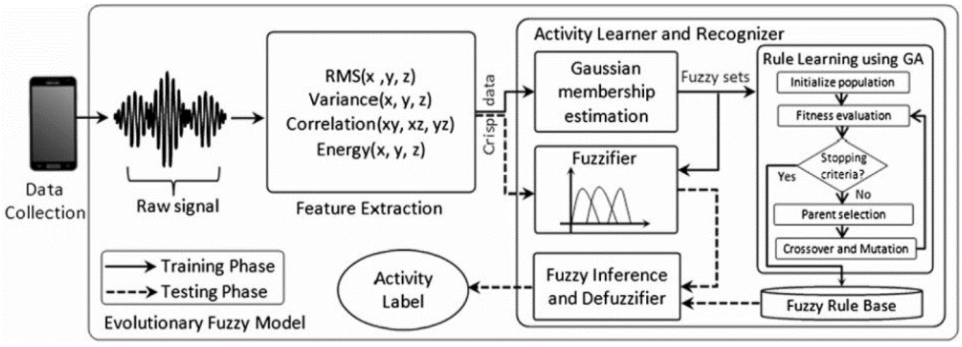


Figure 6: EFM (Evolutionary Fuzzy Model) proposed architecture [34].

Fahim et al. [34] applied a fuzzy logic approach to smartphone accelerometer readings. Fuzzy classification techniques in mobility monitoring are not new, as seen in [35] in 2009, and [36] in 2012. The coined *Evolutionary Fuzzy Model* contains a fuzzy rule base that acts as a main training set (Figure 6). Estimation through a fuzzy network of pre-

determined training data was used to maximize the likelihood of predicting the expected activity. Features were constructed from raw accelerometer signals and then inputted into an activity learner. Each three-second window included root mean square (RMS), variance, correlation, and energy. Recognized activities included walking, jogging, running, cycling, downstairs, hopping, and upstairs. The overall F-Score result was 95%.

2.6 Mobility monitoring in mobile OS APIs

As an update to Google Play Services for Android devices, new APIs for developers were released in 2013, including an activity recognition API. This allowed developers to query the Android system for activity states; such as, bike, foot, car, still, tilting, and unknown. The software returned a confidence level (0-100%) on each mobility state predicted.

A quick experiment was conducted on the Carmesh blog to validate the Android activity recognizer [45]. Walking was the most accurate, followed by driving, but bicycling had large confidence levels of 'tilt' and 'unknown'.

On September 10, 2013, Apple Inc. announced the iPhone 5S, with a new microchip called the M7. This coprocessor works alongside the A7 microprocessor to analyze motion for sensor data collection and processing [46]. The A7 CPU does not need to be powered for the M7 to perform motion calculations. This provides a more efficient method for motion capture and analysis. Apple's new *CoreMotion* API allows developers to identify human movement to aid in contextual app design, where activities such as being stationary, walking, running, or driving can be predicted. There is currently no research that evaluates the M7 efficiency or efficacy for mobility prediction.

Chapter 3
WMMS1 & WMMS2 OVERVIEW

The *Wearable Mobility Monitoring System* (WMMS) is an ongoing project at The Ottawa Hospital Rehabilitation Centre, in Ottawa, Ontario, Canada. This research started in 2007 and led to a thesis by Gaëtanne Haché in 2010 [2][47][48] and was continued by Hui Hsien Wu's thesis in 2012 [15][49][50].

A WMMS should categorize mobility and detect user changes-of-state. A mobility categorization can be whether the user is currently standing or sitting, while a change-of-state is the transition from one state to another, for example between standing to walking or sitting to walking, as shown in Figure 7. Most systems achieve mobility categorization by recording data at a moment in time and deducing a mobility state prediction. This method can be regarded as a context-independent approach.

Over periods of continuous activity monitoring, detecting changes-of-state and classifying activities become separate problems, since algorithms tailored for classification alone may produce large amounts of false positives in changes-of-state. The decision-based structure for current WMMS algorithms minimizes the sensitivity gap between these two measures.

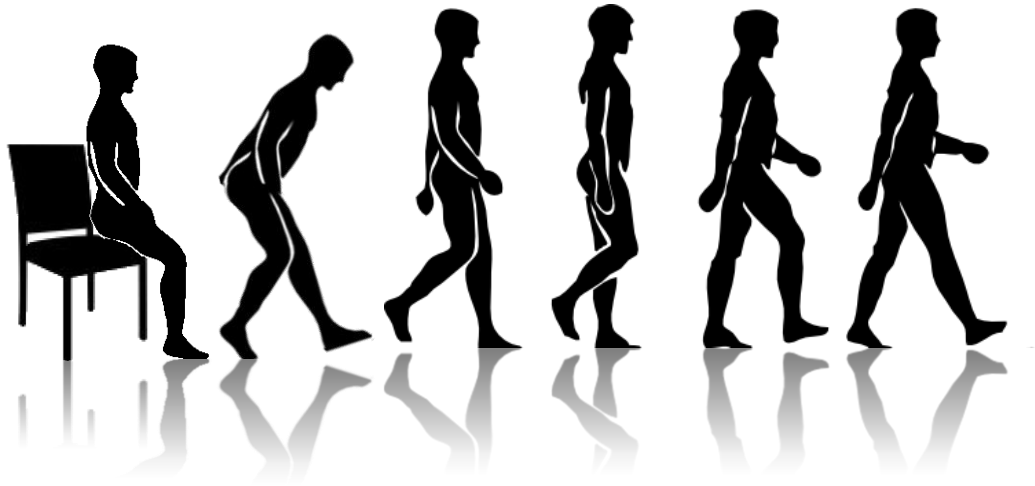


Figure 7: Transition from sit to walk.

3.1 WMMS1

The WMMS1, or the first iteration of the WMMS project, was developed and tested by Gaëtanne Haché, for a MASc. in Mechanical Engineering at the University of Ottawa. It was comprised of a 'smart-holster', consisting of an accelerometer, Bluetooth radio, temperature sensor, humidity sensor, and light sensor (Figure 8). The system's central node was a BlackBerry Bold 9000 smartphone. An independent, external microcontroller allowed for the sampling and packet control of all onboard sensors, and included a buffer until the data were sent to the BlackBerry handset. The BlackBerry 9000 did not have an onboard accelerometer and therefore acquired the accelerometer data from the smart-holster. The handset was used for data storage, GPS, current time, and camera functions as well as all algorithm calculations to detect changes of state. The handset was also used to take a photograph whenever a change of state was detected by the WMMS algorithm.

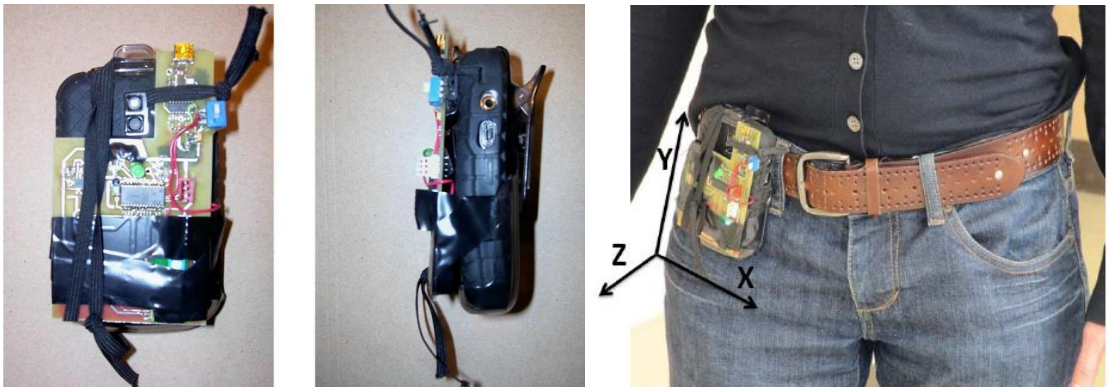


Figure 8: WMMS1 hardware and placement [2].

3.1.1 *Features*

The features calculated in the WMMS1 algorithm included,

Inclination Angle $$\phi = \arctan 2(A_z, A_y), \tag{3.1}$$

Standard Deviation $$\sigma_y = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2}\ , \tag{3.2}$$

Skewness $$s_y = \frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\left(\frac{y_i - \bar{y}}{\sigma}\right), \tag{3.3}$$

Signal Magnitude Area $$SMA = \frac{1}{t}\left(\int_0^t |A_x(t)|\,dt + \int_0^t |A_y(t)|\,dt + \int_0^t |A_z(t)|\,dt\right), \tag{3.4}$$

where n is the number of samples in the feature window, A_x, A_y, A_z are the accelerations in the *x, y* and *z* directions, and $y_1, y_2, y_3 ... y_n$ is the sample set for the above equations.

3.1.2 *Algorithm*

Thresholds were applied to the features via a decision tree classifier to determine the user state. Alongside accelerometer time-domain features, GPS and light sensors were used to classify vehicle riding and inside or outside environments. The full set of activities included standing, sitting, walking, climbing stairs, and ramp navigation.

Unlike the papers cited in the literature review, Haché determined changes-of-state between mobility activities. State transitions included stand-to-sit, sit-to-stand, walk-to-upstairs, walk-to-downstairs, stand-to-lie, lie-to-stand, start of car ride, stop of car ride, and indoor-outdoor. Changes-of-state were used to trigger still image capture in real-time. This introduced context-identification to the system, where user mobility is monitored with the context of motion, identifiable from a photo.

Additionally, change-of-state images helped classify more complex tasks and reduced changes-of-state false positives. For example, differentiating an elevator ride from standing or the vehicle transportation method (i.e., car, bus, train etc.). Furthermore, identifying the walking surface could be useful for older adults or people with a physical

disability since knowing if a person walked on a non-leveled or icy surface may change a physician's perspective or help understand fall risk.

Haché achieved overall results of 77.2% (± 2.5%) sensitivity and 96.4% (± 2.2%) specificity. Deficiencies in her algorithm included the detection of stairs and indoor/outdoor transitions.

3.2 WMMS2

The second WMMS iteration (WMMS2) continued the project by only using internal sensors in the BlackBerry 9550 or BlackBerry Bold 9900 smartphones. The research was conducted by Hui Hsien Wu for a MASc. in Mechanical Engineering at the University of Ottawa.

The WMMS2 objectives were to detect, in real-time, a user's mobility change-of-state using only the BlackBerry's onboard sensors, to record a three-second video clip when a change-of-state was identified, and to classify the activities into static and dynamic states, walking, sitting, lying, standing, upstairs, downstairs, and inclined/declined walking. The video clips allowed for future development of video tracking/detection software for further mobility categorization. This thesis proposed a video post-processing method performed by an assistant for recorded clip categorization. The BlackBerry device was placed on the right front pelvis (Figure 9).

The WMMS2 algorithm was a modified WMMS1 version, with a more complex decision tree for categorization and change-of-state determination. The accelerometer data rate averaged 7.88 ± 1 Hz on the BlackBerry devices tested. The algorithm was evaluated for change-of-state (CoS) determination and activity classification.

The WMMS2 algorithm is summarized in Figure 10. Like section 2.2, the WMMS2 used a decision-tree classifier with features generated from an accelerometer signal. The full algorithm was depicted as a flow chart in the thesis and split between activity classification (categorization) and change-of-state determination.

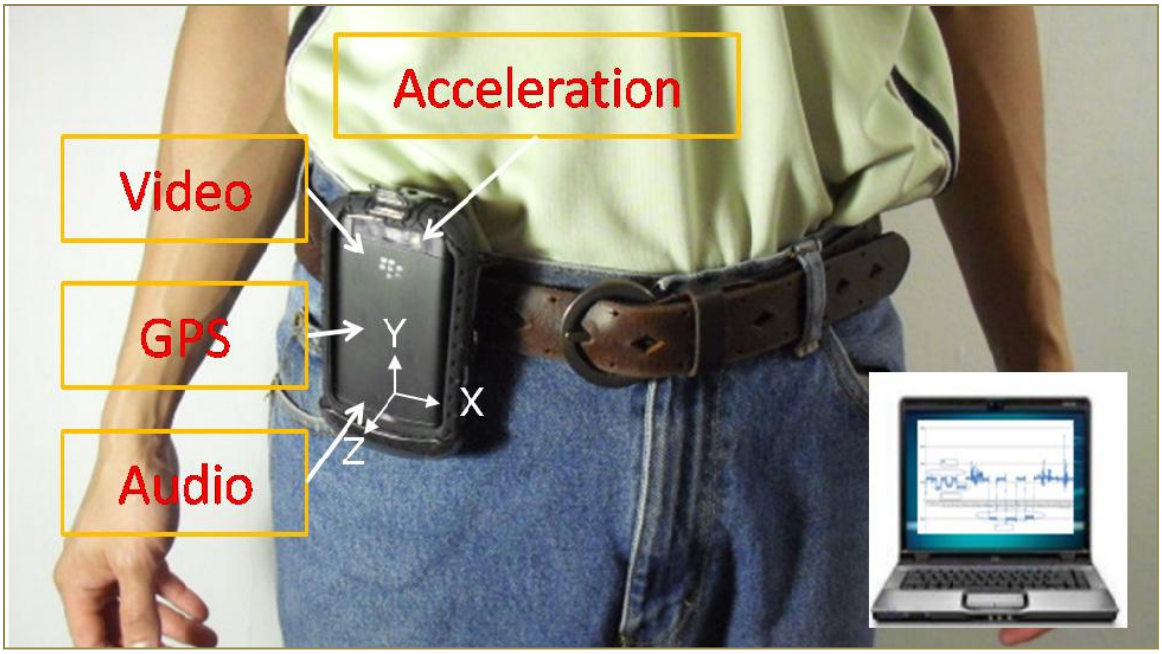


Figure 9: WMMS2 hardware and placement [15].

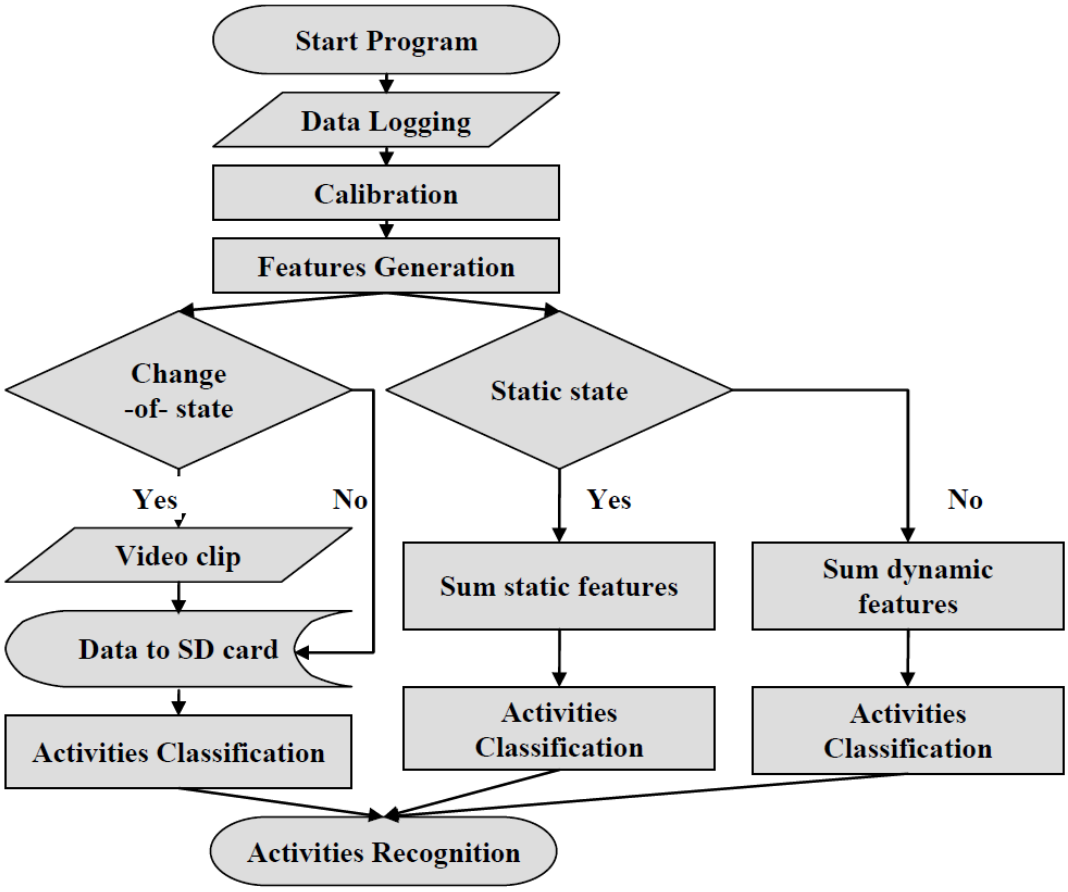


Figure 10: Simplified WMMS2 algorithm and signal processing flow chart [15].

3.2.1 *Features*

WMMS2 used a fixed one-second window size for feature calculation. However, accelerometer samples per window varied due to variable data rate. The entire data set can be represented as,

$$x_i, y_i, z_i \;{}_{i=1}^{n}\;{}_{j=1}^{N}, \tag{3.5}$$

where the sequence of i accelerations is inside a feature window of size n, with j windows for data size N.

The features were defined as,

Standard Deviation
$$\sigma_{(x,y,z),j} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} \left(x, y, z \right)_i - \left(\bar{x}, \bar{y}, \bar{z} \right)_j^{\,2}}, \tag{3.6}$$

Tilt Angle
$$\phi_j = \arctan 2(\bar{z}_j, \bar{y}_j), \tag{3.7}$$

Range Y
$$range_{y,j} = \max_j(\mathrm{y}_{i \in j}) - \min_j(\mathrm{y}_{i \in j}), \tag{3.8}$$

Range XZ
$$range_{xz,j} = range_{x,j} + range_{z,j}, \tag{3.9}$$

Sum of Ranges
$$SoR_j = range_{x,j} + range_{y,j} + range_{z,j}, \tag{3.10}$$

Average Y
$$avg_{y,j} = \frac{1}{n}\sum_{i=1}^{n} y_i, \tag{3.11}$$

Signal Magnitude Area
$$SMA_j = \begin{cases} \frac{1}{8}\sum_{k=j}^{k+7} \sum \max_{-k}(x, y, z)_{i \in -k} & \text{for } j > 8 \\ 0 & \text{for } j \le 8 \end{cases}, \tag{3.12}$$

Difference for Sum of Ranges
$$DIFFSR_j = \begin{cases} SoR_j - SoR_{j-1} & \text{for } j > 1 \\ 0 & \text{for } j \le 1 \end{cases}. \tag{3.13}$$

3.2.2 *Algorithm*

The WMMS2 algorithm is described from Table 5 to Table 8. Features (generated by the raw data) and thresholds (predetermined) are used to calculate state variables, which lead to the expect act variable, which dictates the outcome of the algorithm within a single second activity frame.

Table 5: WMMS2 features, thresholds, and state variables.

Features	Variable Name
Standard deviation X	STDX
Standard deviation Y	STDY
Standard deviation Z	STDZ
Tilt angle	TiltZY
Sum of ranges	SoR
Difference for sum of ranges	DIFFSR
Signal magnitude area	SMASR
Range XZ	RangeXZ
Range Y	RangeY
Average Y	AvgY
ChangeofState3	Sub_state[12]
Expected Act	Sub_state[13]

Category of threshold	# of thresh -olds	Variable name
Dynamic	3	Dyn1… Dyn3
Static	1	Sta1
Downstairs	3	DwnS1… DwnS3
Not downstairs	8	NDwn1… NDwn8
Upstairs	3	UpS1… UpS3
Ramp	3	Rmp1… Rmp3
Elevator	6	Elv1… Ele6
Move static	2	MveSt1, MveSt2
Lie	2	Lie1, Lie2
Stand	2	Stnd1, Stnd2

State variable	Variable Name
LyorSt	Sub_state[2]
StorDy	Sub_state[3]
MoveStatic	Sub_state[4]
Downstair	Sub_state[5]
Upstair	Sub_state[6]
Ramp	Sub_state[7]
Elevator	Sub_state[8]
State	Sub_state[9]
CoS 1	Sub_state[10]
CoS 2	Sub_state[11]
CoS 3	Sub_state[12]
Expected act	Sub_state[13]

Table 6: WMMS2 state variable calculations.

		Algorithm			
	State Variable	Thresholds	Features	Code	Output
	sub_states	sub_thresholds	sub_features		sub_states
Dictates how the phone is positioned	LyorSt	Stnd1, Stnd2, Lie1, Lie2	TiltZY	TiltZY>Stnd1 & TiltZY<Stnd2	32 (sit)
				TiltZY>Lie1 & TiltZY<Lie2	16 (lie)
				Else…	8 (stand)
Dictates whether the user is static or dynamic	StorDy	Dyn1, Dyn2, Dyn3, Sta1	STDX, STDY, STDZ, SOR, RangeY	STDX>Dyn1 OR STDY>Dyn1 OR STDZ>Dyn1 OR (SOR>Dyn2 &RangeY>Dyn3)	1 (dynamic)
				STDY<Sta1	0 (static)
				Else…	Previous Value
Dictates whether a motion was perceived when StorDy registers as static	MoveStatic	MveSt1, MveSt2	SofR, StorDy*	StorDy=0 & SofR>MveSt2	2048 (perceived)
				StorDy=0 & SofR>MveSt1	Previous Value
				Else…	0 (not perceived)
Dictates a downstairs activity	Downstair	DwnS1, DwnS2, DwnS3, NDwnS1, NDwnS2, NDwnS4, NDwnS5, NDwnS6, NDwnS7, NDwnS8	RangeY, SofR, SMASR, DIFFSR, RangeXZ,	See WMMS2 Code [15]	64 (perceived) 0 (not perceived) Previous Value
Dictates an upstairs activity	Upstair	UpS2, UpS3, NDwnS6, NDwnS8	RangeY, SMASR, RangeXZ	See WMMS2 Code [15]	64 (perceived) 0 (not perceived) Previous Value
Dictates a ramp activity	Ramp	Rmp1, Rmp2, Rmp3	RangeY, SofR	See WMMS2 Code [15]	64 (perceived) 0 (not perceived) Previous Value
Dictates an elevator activity	Elevator	Elv1, Elv2, Elv3, Elv4, Elv5, Elv6, Dyn1, Dyn2	STDX, STDY, STDZ, RangeY, AvgY, SofR	See WMMS2 Code [15]	256 (perceived) 0 (not perceived) Previous Value

**StorDy state variable used as a threshold to solve for MoveStatic.*
Not used in algorithm: MaxX, MaxY, MaxZ, MinX, MinY, MinZ, RangeX, RangeZ, AvgX, AvgZ, DiffSR, NDwnS (3)

Table 7: WMMS2 intermediate state variables.

	State variable	Algorithm/Output
	sub_states	sub_states
Sum of all state variables calculated	State*	StorDy+MoveStatic+Downstair +Upstair+Ramp+Elevator
Difference between current state and last state	Change of state 1*	State(n)-State(n-1)
Difference between current state and 2 states ago	Change of state 2*	State(n)-State(n-2)
Difference between current state and 3 states ago	Change of state 3*	State(n)-State(n-3)

Table 8: Solving for the expected act variable for WMMS2.

	Algorithm/Output for expected act value		Expected act	
	Code	Value	sub_states[13]	
(a)	If (StorDy & MoveStatic=0)	LyorSt + Elv	8,16,32,264,272,288	StorDy = 0
(b)	elseIf {MoveStatic(n-2,n-1,n)>0, StorDy=0, LyorSt(n)=LyorSt(n-1)}	LyorSt + MoveStatic	2056,2064,2080	
(c)	elseIf (Downstair & Upstair & Ramp=0, StorDy>0, LyorSt>8)		1024	StorDy = 1
(d)	elseIf (StorDy=1)	Downstair+Upstair+ Ramp+48	48,112,176,240	
(e)	Else…		Previous Value	

a) Minimal movement/static state detected. The output can be a stand, sit, lie or elevator,
b) If a static state was detected, and the last three feature windows detected a Movestatic, and the phone position was the same as the previous, the output is always the previous state (>1023),
c) If no stairs or ramp were detected, dynamic movement was detected, and the phone position is not a stand, then the previous state will be the output,
d) A walk is perceived if StorDy is dynamic, and the above are not true,
e) If none are true, the previous act is presumed.

Expected Act	State
8	Stand
32	Sit
16	Lie
48-240	Walk
If Ramp State = 64:	
	Ramp
If Upstair State = 64:	
	Stairs
If Downstair State = 64:	
	Stairs
264-288	Elevator
>1023	Prev Action

The WMMS2 algorithm contained a few redundancies. In Step 4b, LyorSt (8, 16, or 32) summed with Movestatic (2048) is redundant because anything greater than 1024 always registers as last. In Step 4d, there is no need to add downstairs, upstairs, and ramp since the algorithm always checks the state variables for ramp, upstairs and downstairs.

For evaluation, the algorithm included a car-ride classifier, which used the BlackBerry 9550's GPS module in addition to the accelerometer. The algorithm was tuned to the subject being tested (i.e., all thresholds were chosen to ensure their effectiveness in categorization and CoS prediction).

The WMMS2 thesis was split into its two main evaluations. The 'Change-of-State Determination' evaluation involved a single able-bodied participant performing a series of consecutive movements across three trials. Changes-of-state were analyzed and compared with gold-standard measures from a video recording of the trials. Sensitivities for change-of-state detection ranged from 0% to 100%.

The algorithm performed poorly when detecting changes-of-state for entering or exiting an elevator (0%), going up or down a ramp (33%) and going up or down stairs (66%). However, transitions for stand, sit, and lie were detected as a change-of-state every time. Transitions to walk were quite high (98%) but walking tended to produce many false positives.

For the 'Activity Classification' evaluation, five able-bodied participants moved through a predetermined circuit, including a daily-living environment. The circuit included walking, taking an elevator, various kitchen and bathroom tasks (brushing teeth, washing dishes etc.), lying on a bed, and climbing stairs, and walking on a ramp. For changes-of-state, sensitivities and specificities were 13%-100% and 88%-100%, respectively. Static to dynamic or dynamic to static transitions were easier to identify. Stand to walk, or lie to walk changes-of-state were detected using the standard deviation feature.

Human intervention with video clip reviewing was added, resulting in an increase in specificity. Activities such as washing dishes, combing hair, toasting bread and brushing teeth were visible in the video clips taken by the BlackBerry device, thereby allowing classification of these activities.

Chapter 4
WMMS2 CORRECTION & EVALUATION

This section focuses on device orientation. The content is from a conference paper titled "Correcting Smartphone orientation for accelerometer-based analysis" [51] and represents the first part of the WMMS3 research. Evaluation was with a BlackBerry 9900 phone running BlackBerry OS 7.

4.1 Introduction

A common source of error with a waist-mounted smartphone WMMS is the resting device angle due to waist girth, clothing, or the device's holster (Figure 11). Certain users require a standing offset calibration to orient the accelerometer axis to a world reference since several activity classification algorithms rely heavily on the standing state. For this class of users, inaccuracies in activity determination occur since the features and state evaluations are adversely affected by the orientation offset.

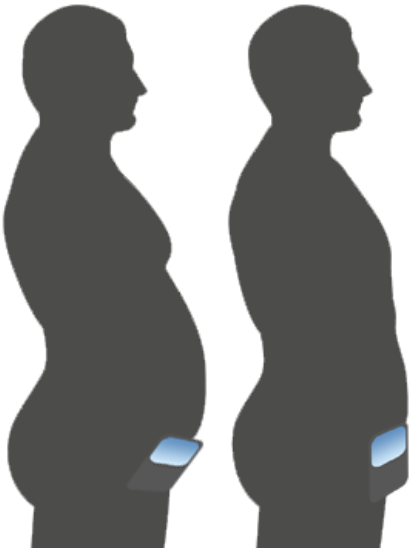


Figure 11: Smartphone orientation variability in hip placement during standing.

4.2 Method

A calibration method was proposed that adjusts accelerometer component orientation to natural device stance for a mobility monitoring system. This method corrects accelerometer axis orientation by applying a quaternion rotation transformation to the device's accelerometer data. This rotation compensates for device alignment variability when standing naturally. The standing coordinate system for the WMMS BlackBerry device is shown in Figure 12.

In three-dimensional space, mutually perpendicular axes passing through a fixed origin conveniently describe the location of a point or direction and magnitude of a vector from

the specified origin [52]. Rotating the three axes in R^3 while retaining their orthogonality is called a frame rotation.

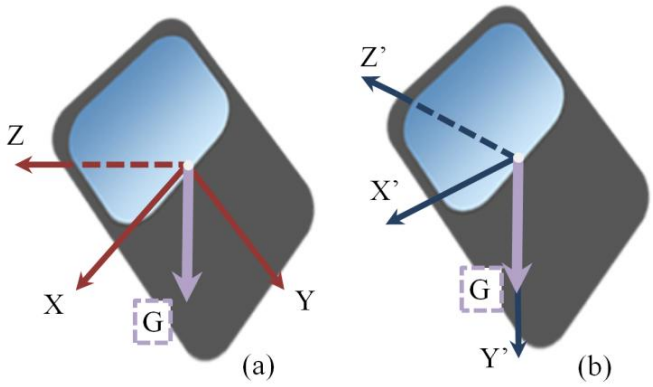


Figure 12: (a) The non-calibrated standing position (*XYZ*) is the device's position in the holster. The gravitational force vector (G) is split into components along *X*, *Y* and *Z*. (b) The virtual calibrated standing position (*X'Y'Z'*) after the transformation. The gravitational force vector (G) is now along the positive Y' axis, representing an upright device position.

A rotation matrix describes rotation of a coordinate system orientation with respect to another. A vector $\vec{V}$ in the initial reference frame F can be transformed into a vector $\vec{V}'$ in the rotated frame F' by multiplication of $\vec{V}$ with the rotation matrix between F and F' [53]. Rotation matrices are characterized as orthogonal, square, and invertible:

$$R^T = R^{-1},\ R^T R = RR^T = I,\ \det R = 1. \tag{4.1}$$

In three dimensions, rotations around the three principal axes produce an orientation shift of any vector from the frame F to F' (Figure 13). The two static vectors $\vec{V}$ and $\vec{V}'$ can be used to describe the orientation adjustment between the current reference frame and desired frame, producing a rotation matrix R. Multiplying R by $\vec{V}$ produces $\vec{V}'$, while multiplying the transpose rotation matrix R^T with $\vec{V}'$, produces the initial vector $\vec{V}$.

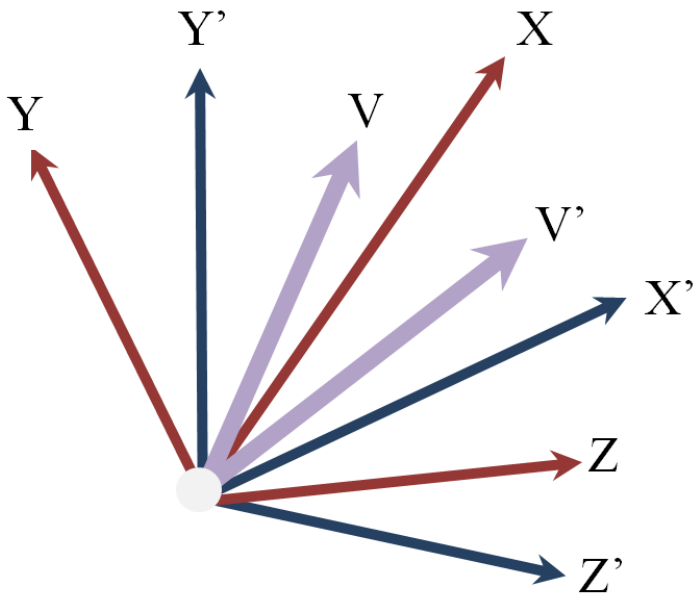


Figure 13: Vector $\vec{V}$ in reference frame *F* rotated to become vector $\vec{V}'$ in reference frame F'.

4.2.1 *Quaternions*

Hamilton quaternions are a four-dimensional hyper-complex system of numbers that describe object orientation and rotation in three-dimensions. A matrix describing a rotation in R^3 can be constructed with a four-dimensional linear description through quaternions. Each quaternion consists of four scalar independent axes, with one real dimension, and three imaginary dimensions. The unit values for the imaginary dimensions are mutually perpendicular square roots of -1, while the real dimension is an orthogonal axis to the imaginary ones, with a unit value of 1. A quaternion can be represented as,

$$\vec{q} = q_0 + iq_1 + jq_2 + kq_3, \tag{4.2}$$

where the complex unit vectors in a quaternion space are,

$$i^2 = j^2 = k^2 = ijk = -1. \tag{4.3}$$

The mathematical properties of quaternions are described in [52]. Quaternions are convenient in representing three-dimensional rotations and orientations because issues with non-linearity, singularities, and difficulties with Euler angle representations are

avoided. A rotation described by quaternions is also efficient since only four variables are required to represent the rotation, whereas Euler angles require more variables and consideration of the operation order [54].

The unit quaternion $\vec{q} = q_0 + q$ exists on the unit sphere in the four-dimensional Hamilton space H^4, where q_0 is the called the 'scalar part' and q is called the 'vector part'. This philosophy is analogous to real and imaginary parts of a complex number. The Hamilton space is described further in [55]. The unit quaternion becomes,

$$\vec{q} = \cos\theta + \hat{e}\sin\theta, \tag{4.4}$$

$$\hat{e} = \frac{\vec{q}}{|\vec{q}|}, \qquad \tan\theta = \frac{|\vec{q}|}{q_0}, \tag{4.5}$$

where vector $\hat{e}$ is the oriented unit vector axis of rotation while $2\cdot\theta$ is the angle of rotation in R^3 and $\vec{q} \in \mathrm{H}^4$.

Input vectors $\{\vec{V}, \vec{V}'\} \in \mathrm{R}^3$ can construct a rotation matrix after applying the action of the triple product operator $\vec{V}' = L_q\ \vec{V}\ = \vec{q}\vec{V}_H\vec{q}^*$ described by [52], where q^* is the complex conjugate of q. Vector $\vec{V}_H$ is simply treated as vector $\vec{V}$ but as a pure quaternion, $q \in \mathrm{H}^4$, whose real part is zero, that is,

$$\vec{V} \in \mathrm{R}^3 \leftrightarrow \vec{V}_H = 0 + \vec{V} \in Q_0 \subset Q,$$

$$\vec{V}_H = 0 + iV_x + jV_y + kV_z.$$

Due to the correspondence between a pure unit quaternion (where the scalar part $q_0 = 0$) and a vector in R^3 global coordinates, a transformation using the $L_q\ \vec{V}$ operator allows for the transformation from $\vec{V} \to \vec{V}_H \to \vec{V}'$ (Figure 14).

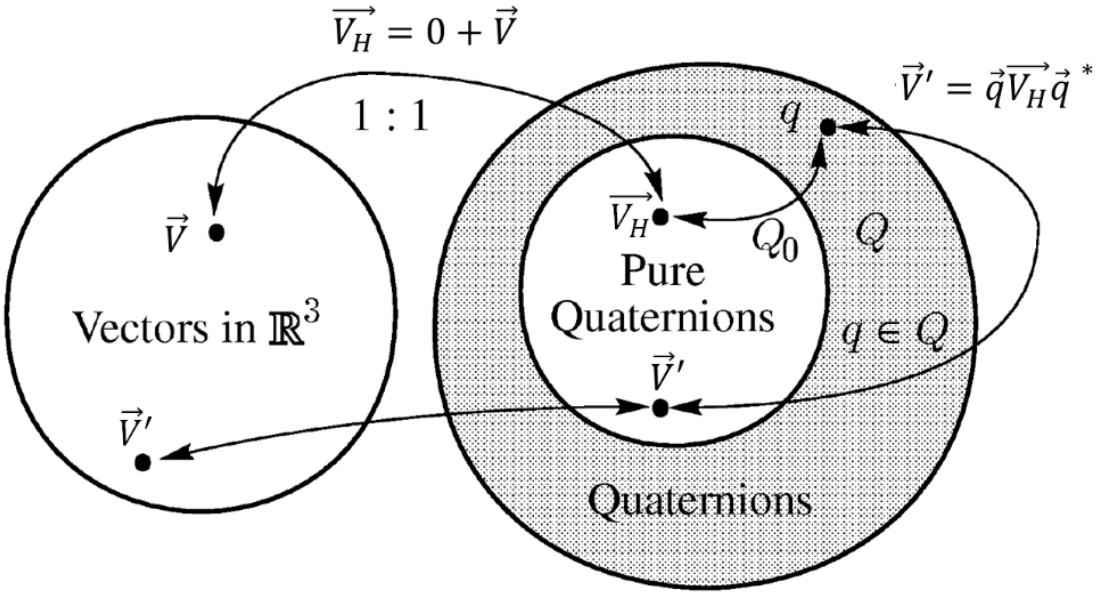


Figure 14: The correspondence between vectors $\leftrightarrow$ quaternions. Vector $\vec{V}$ transforms to $\vec{V}_H$ and then is further manipulated by the rotation operator returning back to R^3 as $\vec{V}'$. This process can be reversed from $\vec{V}'$ to $\vec{V}$ [52].

Geometrically, the unit quaternion and operator construct a rotation about angle α, on the orthogonal two-dimensional plane in R^3 around an axis in the direction of a normalized vector $\hat{e}$, given by the cross product of $\vec{V}$ and $\vec{V}'$. This relationship is called the axis-angle representation of a rotation [54]. Since $\vec{V}$, and $\vec{V}'$ are known, the quaternion that represents the rotation between them can be deduced, resulting in a unique solution that shifts the reference frame for all preceding $\vec{V}$ vectors to the new frame, F'. Considering rotation angle α and the normalized axis vector $\hat{e}$, the quaternion unit vector is:

$$\vec{q} = \cos(\alpha/2) + i\sin(\alpha/2)\hat{e}_{norm,x} + j\sin(\alpha/2)\hat{e}_{norm,y} + k\sin(\alpha/2)\hat{e}_{norm,z}. \qquad (4.6)$$

The triple vector product operator algorithm $L_q\ \vec{V}_H$ is defined as,

$$\vec{V}' = \vec{q}\vec{V}_H\vec{q}^{*} = \ q_0 + \vec{q}\quad 0 + \vec{V}\quad q_0 - \vec{q} \qquad (4.7)$$

$$\vec{V}' = \ 2q_0^2 - 1\ \vec{V} + 2\ \vec{q}\square\vec{V}\ \ \vec{q} + 2q_0\ \ \vec{q}\times\vec{V}\ . \qquad (4.8)$$

Expanding to matrix format gives

$$2q_0^2-1\ \vec{V}=\begin{bmatrix}2q_0^2-1 & 0 & 0\\ 0 & 2q_0^2-1 & 0\\ 0 & 0 & 2q_0^2-1\end{bmatrix}\begin{bmatrix}V_1\\ V_2\\ V_3\end{bmatrix}, \tag{4.9}$$

$$2\ \vec{q}\vec{V}\ q=\begin{bmatrix}2q_1^2 & 2q_1q_2 & 2q_1q_3\\ 2q_1q_2 & 2q_2^2 & 2q_2q_3\\ 2q_1q_3 & 2q_2q_3 & 2q_3^2\end{bmatrix}\begin{bmatrix}V_1\\ V_2\\ V_3\end{bmatrix}, \tag{4.10}$$

$$2\ \vec{q}\times\vec{V}\ q=\begin{bmatrix}0 & -2q_0q_3 & 2q_0q_2\\ 2q_0q_3 & 0 & -2q_0q_1\\ -2q_0q_2 & 2q_0q_1 & 0\end{bmatrix}\begin{bmatrix}V_1\\ V_2\\ V_3\end{bmatrix}. \tag{4.11}$$

Applying the single constraint,

$$q_1^2+q_2^2+q_3^2+q_4^2=0, \tag{4.12}$$

the matrices can be combined to construct the resulting rotation matrix,

$$R\ q_0,q_1,q_2,q_3\ =R=\begin{bmatrix}1-2(q_2^2+q_3^2) & 2(q_1q_2-q_0q_3) & 2(q_0q_2+q_1q_3)\\ 2(q_1q_2+q_0q_3) & 1-2(q_1^2+q_3^2) & 2(q_2q_3-q_0q_1)\\ 2(q_1q_3-q_0q_2) & 2(q_0q_1+q_2q_3) & 1-2(q_1^2+q_2^2)\end{bmatrix}, \tag{4.13}$$

where $\vec{V}'=L_q\ \vec{V}_H\ =\vec{q}\vec{V}_H\vec{q}^*=R\vec{V}$.

Solving for $\hat{e}$ and α between $\vec{V}$ and $\vec{V}'$ provides the quaternion unit vector required to build the rotation matrix that describes a rotation between $\vec{V}$ and $\vec{V}'$. This rotation matrix can then be used to rotate frame F to F'.

4.3 Application

The calibration method involves constructing a quaternion rotation matrix, via the axis-angle representation. First, the axis vector is produced by taking the initial orientation gravitational vector's cross product with the desired orientation gravitational vector. Then, the dot product is used to solve for the rotation angle α, which is then used

with the axis vector to build the quaternion rotation matrix. Finally, all raw accelerometer data are multiplied by the rotation matrix to achieve the corrected orientation.

The gravitational force applied to the Smartphone accelerometer is used as the initial orientation vector. The desired device position is upright; therefore, the final orientation vector is gravity acting only in the A_y' direction in the second reference frame. The gravitational acceleration vector for the user's natural stance is referred to as $\vec{A}_i = \vec{A}$ and the desired gravitational vector, which passes only through the A_y' axis, $\vec{A}_f = A_y'$

Considering the initial acceleration vector $\vec{A}_i$ and the desired acceleration vector $\vec{A}_f$,

$$\vec{A}_f = R\vec{A}_i,$$

where,

$$\begin{Bmatrix} A_x' \\ A_y' \\ A_z' \end{Bmatrix} = R \begin{Bmatrix} A_x \\ A_y \\ A_z \end{Bmatrix}. \tag{4.14}$$

The rotation matrix from equation 4.13 is applied to all accelerometer readings in real time using equation 4.14 thus correcting for the device slant or tilt.

The initial gravity vector $\vec{A}_i$ is obtained from a 10 second sampling of accelerometer data with the user naturally standing still. This is averaged to yield $\vec{A}_i = A_x\hat{i} + A_i\hat{j} + A_z\hat{k}$, while the desired gravity vector is $\vec{A}_f = 0\hat{i} + 9.81\hat{j} + 0\hat{k}$.

To construct the set of quaternions that yield the required rotation, an axis-angle pair must first be produced. The axis vector $\hat{e}$ can be found from the cross-product between the initial and final gravity vectors,

$$\hat{e} = \vec{A}_i \times \vec{A}_f = (-9.81 \cdot A_z)\hat{i} + (9.81 \cdot A_x)\hat{k}. \tag{4.15}$$

The axis vector $\hat{e}$ from 4.15 can then be normalized by dividing by the magnitude of $\hat{e}$,

$$\hat{e}_{norm} = -\left(\frac{A_z}{\sqrt{A_x^2 + A_z^2}}\right)\hat{i} + \left(\frac{A_x}{\sqrt{A_x^2 + A_z^2}}\right)\hat{k}. \tag{4.16}$$

The angle between vectors can be expressed as the cosine angle from the dot product. Using initial vector $\vec{A}_i$ and end vector $\vec{A}_f$, it follows that

$$\vec{A}_i \cdot \vec{A}_f = \left\|\vec{A}_i\right\| \cdot \left\|\vec{A}_f\right\| \cdot \cos\ \alpha\ ,$$

$$\vec{A}_i \cdot \vec{A}_f = A_x A_x^{'} + A_y A_y^{'} + A_z A_z^{'}\ .$$

Since $A_x', A_z' = 0$, and $A_y' = \left\|\vec{A}_f\right\|$, the equation becomes

$$\alpha = \arccos\left(\frac{A_y}{\left\|\vec{A}_i\right\|}\right). \tag{4.17}$$

The axis-angle pair described in equations 4.16 and 4.17 can now be used in the quaternion rotation equations, as described in [56].

The quaternion rotation matrix can now be constructed since $\hat{e}$ and α have been calculated based on the ten-second standing time window for calibration. The matrix has nine scalar entrants that are static during the WMMS use. The matrix allows for raw acceleration rotation components as desired,

$$A_x' = q_{11}A_x + q_{12}A_y + q_{13}A_z\ , \tag{4.18}$$

$$A_y' = q_{21}A_x + q_{22}A_y + q_{23}A_z\ , \tag{4.19}$$

$$A_z' = q_{31}A_x + q_{32}A_y + q_{33}A_z\ . \tag{4.20}$$

4.4 Validation

A controlled set of trials were conducted to verify the method for correcting accelerometer data orientation. A BlackBerry 9900 device was securely attached to a rotating mechanical arm that spun 360° clockwise (Figure 15, Figure 16). Four trials were conducted, each with different smartphone orientations based on how the device may rest on the user's hips. Each trial included 20 seconds of sampling without mechanical arm movement, followed by 20 arm revolutions at a constant angular velocity. Accelerometer data were recorded using the TOHRC Data Logger

application [57] at a variable sample rate between 17 and 20 Hz. The testing protocol was repeated with an XSens MTi sensor in the upright orientation as a comparator to the smartphone accelerometer output. Accelerations were recorded at 100 Hz. For trial 1, the device was positioned upright on the rotating arm. This reflects the natural standing position for an average user, where the gravity vector passes only through the accelerometer's *y*-axis. Trials 2-4 represent offsets that can occur when the device is positioned on the user's pelvis (Figure 15). All BlackBerry accelerometer data were imported into Microsoft Excel for analysis. A 10-second sample of static data was used to produce the $\vec{A}_i$ vector and a rotation matrix from equation 4.13 was generated for each trial. Raw accelerometer data were converted to $\vec{A}_f$ using equation 4.14.

Figure 15: Rotation matrix experimental setup.

Figure 16: Device experimental setup showing three of four orientations.

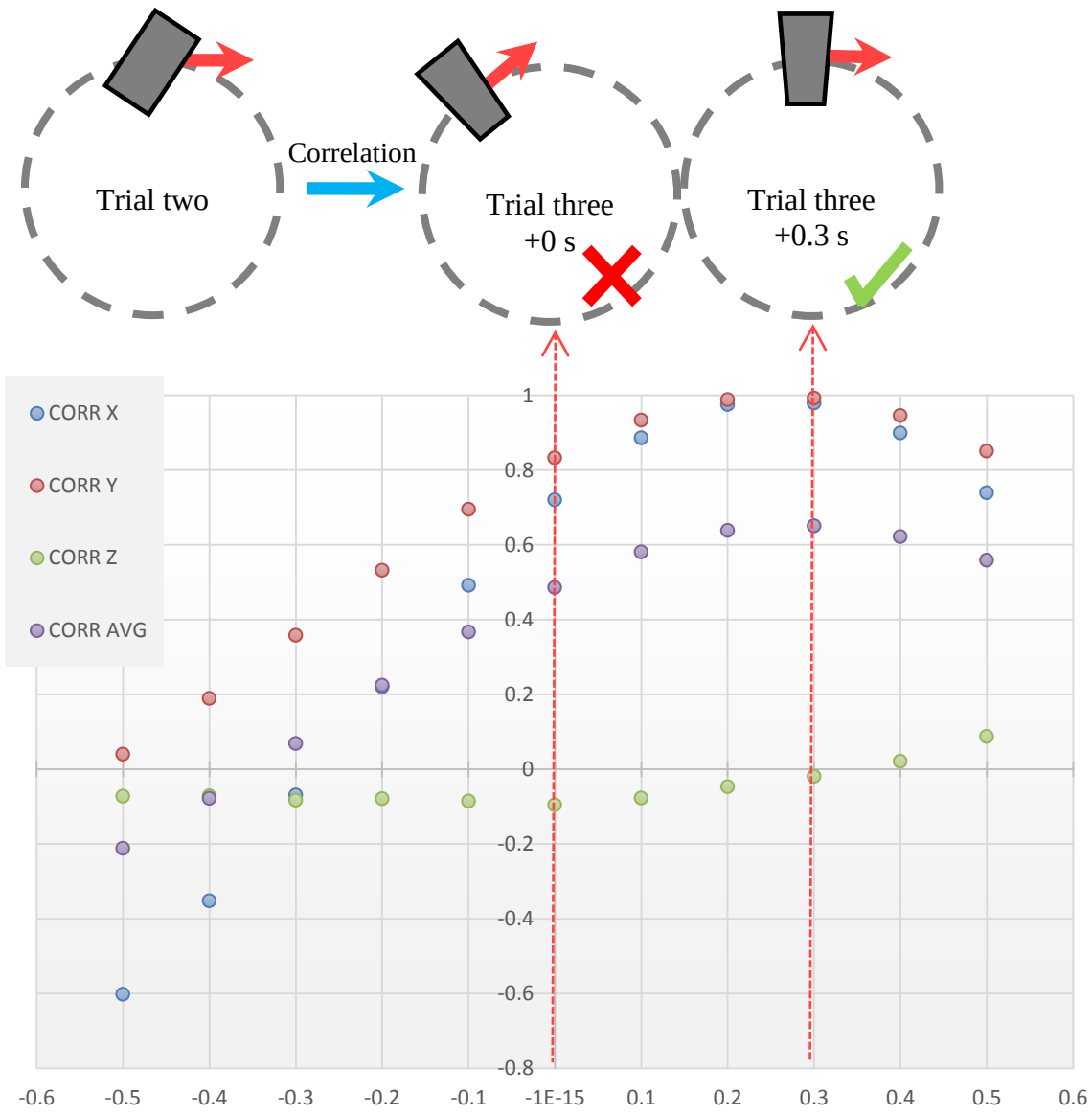


Figure 17: Correlation values between trial two and trial three with a time-shift of -0.5 to 0.5.

Accelerometer data from trials 1 to 4 (Figure 15) should correlate after the rotation matrix is applied, due to the shift to an upright position coordinate system. Gravitational, normal, and tangential accelerations were applied to the smartphone as the mechanical arm rotated. Different phone orientations produced different sets of A_x, A_y, and A_z data. Table 9 shows the static 10 second sampling averages used to construct their respective rotation matrices. $A_y^{'}$ values did not reach the full 9.81 m/s^2, possibly due to consumer grade smartphone accelerometer inaccuracy. The BlackBerry device did not have a constant sampling rate; therefore, each data set was interpolated from 10 s (non-static

data set start) to 67.55 s, in 0.05-second intervals (20 Hz) to match time intervals prior to correlation. Furthermore, the non-static data set start was shifted within a range of -0.5 s to +0.5 s to minimize experimental timing errors (Figure 17) where the *x*-axis is the time shift between trials and the *y*-axis is the correlation value. The maximum correlation was found where the correlation average was the highest within the 1-second time shift. Correlations for rotated and non-rotated values mainly follow a sinusoidal-type plot, which was expected since the experiment was cyclic in nature, where one revolution spanned approximately 1.5 s.

Table 9: Static sampling before and after transformation.

	Raw Accelerometer Averages (m/s^2)			Rotated Accelerometer Averages (m/s^2)		
Trial	A_x	A_y	A_z	A_x'	A_y'	A_z'
1	-0.010	9.74	0.21	0.00	9.74	0.00
2	-6.17	7.43	0.18	0.00	9.66	0.00
3	-0.18	9.19	-3.20	0.00	9.73	0.00
4	-7.49	5.63	-1.97	0.00	9.58	0.00

For trial 1, the quaternion rotation matrix fixed slight offsets on the *x* and *z* axes caused by experimental setup error (Table 9). The accelerometer reference frame was corrected and gravitational, tangential, and normal accelerations were applied only on the *xy* plane. Consequently, the orthogonal *z* component only experienced accelerations due to vibrational noise, with an average of 0.151 m/s^2 and standard deviation of 0.627 m/s^2.

In trial 2, the phone was slanted to the right, resulting in acceleration due to gravity along the *x* component (Table 9). Rotation matrix results were as expected since the *y* component comprised the full magnitude of acceleration due to gravity before movement and the sinusoidal nature of the *x* and *y* accelerations due to the spinning arm reflected trial 1. *Z*-axis noise for trial 2 was similar to trial 1, with an average value of 0.131 m/s^2 and standard deviation of 0.255 m/s^2.

Trial 3's forward tilt moved some of the gravitational acceleration to the *z* component (Table 9). As a result, the arm's spinning motion caused the z component acceleration to fluctuate as the arm rotated. The *z* acceleration's sinusoidal pattern was greatly reduced once the rotation matrix was applied. Much like the other trials, the *z* component experienced vibrational noise that averaged 0.066 m/s^2 with a standard deviation of 0.480 m/s^2. The larger standard deviation in *z* for this trial was expected, since the rotation matrix was unable to fully negate the cyclic nature of the gravitational component along *z*.

For trial 4, a forward and slanted position was applied, splitting the gravitational vector along all components. Rotating the reference frame corrected the offset; however, much like the other trials, *z* experienced vibration noise, with an average of 0.034 m/s^2 and standard deviation of 1.118 m/s^2.

The quaternion rotation performed as expected in all trials, by rotating $\{A_x, A_y, A_z\}$ into A_x', A_y', A_z' to achieve the upright orientation required for the WMMS. Correlation results between data sets are shown in Table 10.

Table 10: Correlations between accelerometer data after interpolation and time-shift.

	Rotated x	*Rotated y*	*Rotated z*
Trial 1 & 2	0.95	0.97	0.16
Trial 1 & 3	0.94	0.97	0.12
Trial 1 & 4	0.92	0.96	0.15
Trial 2 & 3	0.98	0.99	0.02
Trial 2 & 4	0.98	1.00	0.14
Trial 3 & 4	0.98	0.99	0.02

The *x* and *y* values demonstrated strong correlations between the rotated data sets, with an average difference to perfect correlation of 4.2% and 2.1%, respectively. Experimental errors were a result of small vibrations in the rotating arm. The *z*-axis, which had very low accelerometer readings, was most affected by the vibration artifacts, resulting in poor

correlations (Table 10). However, rotated *z* component correlation values for all trials were near zero, thus demonstrating uncorrelated noise.

Vibration noise was further investigated with the XSens sensor on the rotating arm. The *z* component data averaged 0.059 m/s^2, with a standard deviation of 0.298 m/s^2. These values were similar to those generated from the Smartphone, further confirming the vibrational noise caused by the rotating arm.

4.5 Evaluation

Twelve able-bodied participants were recruited from The Ottawa Hospital Rehabilitation Centre to evaluate the orientation correction method. A BlackBerry smartphone was positioned in a belt-holster on the right front pelvis. The smartphones recorded and logged acceleration data with the WMMS2 application, starting with a ten-second standing period. The participants then walked through a predetermined circuit, described in Appendix A. The circuit included walking, sitting and standing, stair climbing and common household tasks. An assistant recorded video footage for each participant through the trial, to provide gold-standard timing data for analyses. The phone was shaken at the start of each trial to synchronize between the video and time-stamped phone acceleration signals. The evaluation protocol mimicked that of Wu's thesis [15].

Accelerometer data retrieved from the BlackBerry devices were processed with the WMMS Tool, described in Chapter 5. Sensitivity and specificity for activity categorization and change-of-state categorization were calculated.

Two evaluation procedures were performed with the WMMS Tool. *Evaluation A* was the unaltered WMMS2 method. *Evaluation B* was the WMMS2 method with a rotation matrix transformation applied to the raw acceleration data. The WMMS2 threshold values used in this study were set as those in Wu's original work, and are shown in Table 11. The *Stnd1* and *Stnd2* thresholds were modified from Wu's original work from 130 to 121 and 188 to 165, respectively. This was necessary for full participant calibration, rather than the initial participants from the original study.

Table 11: WMMS2 threshold values.

Category	Threshold	Value
Dynamic	Dyn1	1.67
	Dyn2	10.78
	Dyn3	3.43
Static	Sta1	1.57
Downstairs	DwnS1	12.74
	DwnS2	31.36
	DwnS3	176.40
Not downstairs	NDwn1	-5.88
	NDwn2	19.60
	NDwn3	29.00
	NDwn4	4.00
	NDwn5	23.50
	NDwn6	157.00
	NDwn7	8.13
	NDwn8	4.90
Upstairs	UpS1	24.50
	UpS2	9.80
	UpS3	127.40

Category	Threshold	Value
Ramp	Rmp1	11.76
	Rmp2	24.50
	Rmp3	5.88
Elevator	Elv1	0.39
	Elv2	1.76
	Elv3	9.41
	Elv4	11.76
	Elv5	1.55
	Elv6	0.49
Move static	MveSt1	0.69
	MveSt2	0.98
Lie	Lie1	50.00
	Lie2	120.00
Stand	Stnd1	121.00
	Stnd2	165.00

A one second feature window, three-window change-of-state tolerance, two-window categorization tolerance were used for all analyses. Sensitivities and specificities were extracted from the WMMS Tool output files and averaged for all participants and activities. The interval between the device's first shake and walk to the first chair in the experimental procedure was used to aid in manually setting the start and end times for the calculation of vector $\vec{A}_i$ from equation 4.14. The WMMS Tool constructed a rotation matrix from this quiet standing period by averaging the *x*, *y*, and *z* accelerometer components, which produced the $\vec{A}_i$ vector. The rotation matrix was then applied to all accelerometer data and the WMMS2 algorithm was re-calculated. The same gold-standard file was used, with the same initial shake time correction, for all analyses.

4.6 Results

Figure 18 shows participant three results from 45 to 90 seconds in the trial. At 45 seconds, the participant shook the smartphone, stood, walked to a chair, and took a seat at 77 seconds. Before the rotation matrix was applied, the device experienced a slant forward, as seen with the A_z component. This resulted in an offset with the sit orientation, which was corrected once the rotation matrix was applied. Ramp instances were false positives, and were not alleviated after applying the rotation matrix.

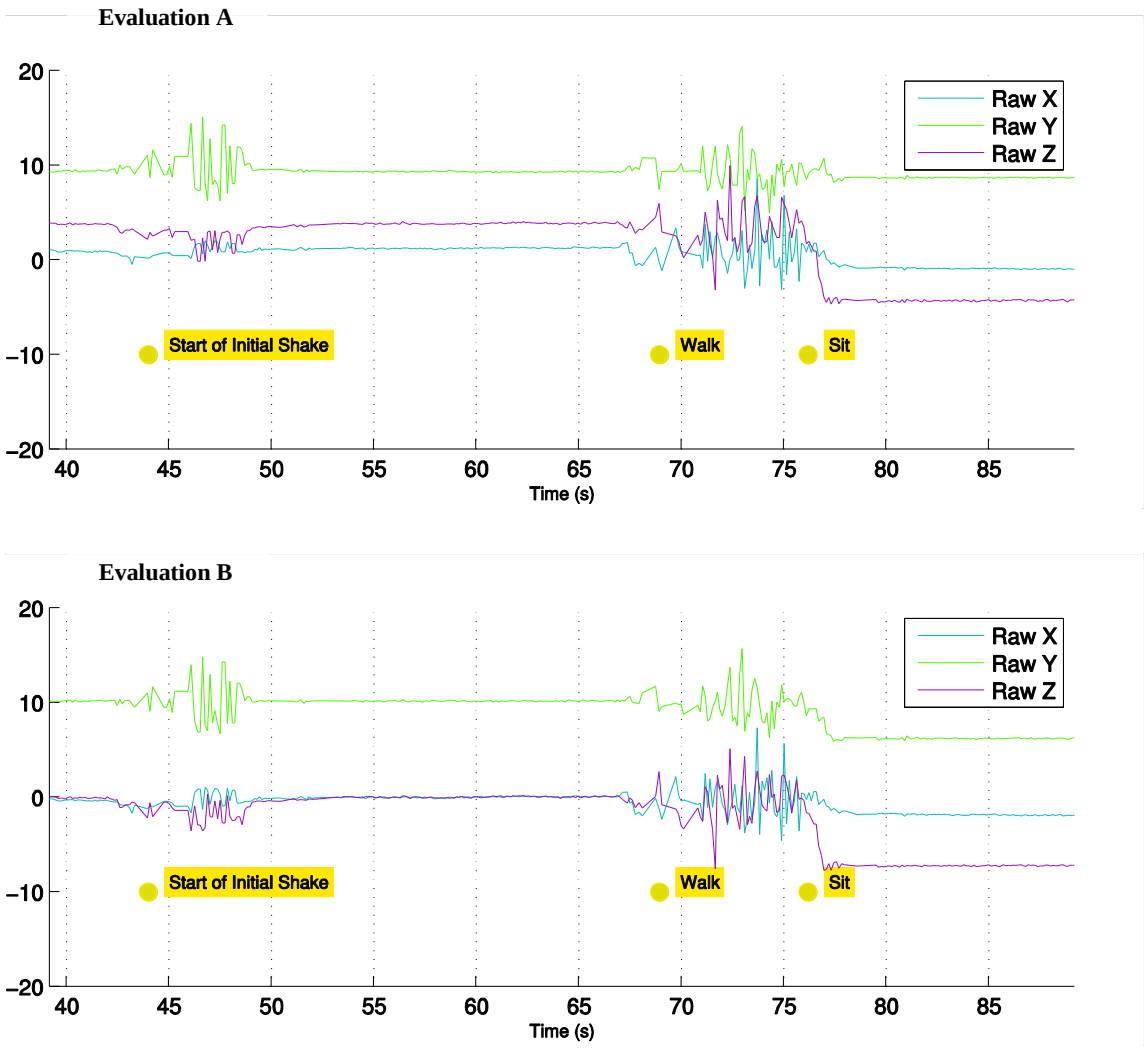


Figure 18: Example of non-rotated (top) vs. rotated data (bottom).

The averaged sensitivities and specificities for *Evaluation A* and *Evaluation B* for all twelve participants are listed in Table 12.

Table 12: Sensitivity and specificity for evaluations A (non-rotated) and B (rotated).

Sensitivity (%)			
Evaluation	A	B	+/-
Stand	65.3	63.0	-2.3
Sit	37.6	62.6	25.0
Lie	85.5	83.6	-1.8
Walk	30.3	30.2	-0.1
Ramp	32.4	33.7	1.3
Upstairs	36.7	38.2	1.4
Downstairs	3.1	3.1	0.0
Elevator	0.0	0.0	0.0
CoS	60.8	59.9	-1.0
Average	39.1	41.6	2.5
Std. Dev.	25.0	25.3	7.6

Specificity (%)			
Evaluation	A	B	+/-
Stand	82.4	84.5	2.1
Sit	100.0	100.0	0.0
Lie	99.1	99.1	0.0
Walk	64.9	64.8	-0.1
Ramp	73.6	73.2	-0.4
Upstairs	91.2	91.2	0.0
Downstairs	98.6	98.7	0.1
Elevator	99.9	99.8	-0.1
CoS	90.1	90.5	0.4
Average	88.9	89.1	0.2
Std. Dev.	11.4	11.4	0.7

4.7 Discussion

Considering all possible categorizations for WMMS2, the sensitivity after applying a rotation matrix showed an average increase from 39.1% to 41.6%. Specificity increased from 88.9% to 89.1%. The largest increase in sensitivity and true positives was for sitting, which increased from 37.6% to 62.6% for the twelve subjects, a difference of 25% (Table 12). An increase sitting sensitivity after applying a rotation matrix was expected.

Other categorizations result in little to no change, in comparison to the sensitivity increase achieved with sitting. Sensitivity and specificity for standing and lying were also expected to increase, but instead demonstrated little to no change. This lack of change could be due to the choice of thresholds used on the rotated data. Outcomes may improve if the thresholds were recalibrated after applying a rotation matrix to the data.

4.8 Conclusion

A quaternion based rotation matrix was applied to accelerometer data to orientate the accelerometer's Cartesian mapping to a gravity-based position. This approach was validated using a revolving arm mechanism. Acceleration components correlated well after applying the rotation transformation, regardless of initial device orientation. This validated the quaternion rotation matrix method for smartphone orientation correction.

The rotation matrix method was then applied to a smartphone WMMS to correct for user variability in phone orientation. A window of standing data was used to construct the rotation matrix, and all accelerations thereafter were calibrated to this new virtual device orientation. Sitting true positives increased substantially once the device was re-oriented. However, other activities did not show improvement in classification, possibly due to the thresholds used in the study.

The rotation matrix method standardizes device orientation. Applications involving mobility and movement will benefit by using this method since a device's accelerometer components can be orientated in any direction around its origin, allowing accelerometer coordinate system control on the device.

Chapter 5
WMMS TOOL DEVELOPMENT

5.1 Introduction

A difficulty in HAR system design is the supervised learning approach. Depending on the learning technique, raw data characteristics and potential features must be assessed to discover relationships between activities and signals. Evaluation procedure standardization is also important for valid HAR system comparison.

Software was developed to help create and evaluate the third generation WMMS (WMMS3), allowing investigation of features and variable interactions and comparison with the WMMS2 algorithm. This WMMS Tool streamlined the process of calculating data windows, features, predictions and results that were previously calculated via multiple Excel® spreadsheets. The objective was to construct an aid for analyzing and parsing BlackBerry OS10 sensor data for WMMS2 and WMM3 design and testing.

5.2 Programming environment

MATLAB® was chosen as the WMMS Tool development environment. MATLAB is a universal computing toolset with built-in graphical functions and graphical user interface (GUI) capabilities [58]. GUI construction using MATLAB benefits from high-level scripting and libraries of efficient functions and scripts. High-level scripting allows for quick development without the need for compilation. MATLAB libraries allow for efficient and rapid feature calculations, especially when 10 minutes of BB10 data exceeds 30,000 records.

The new application synchronizes the gold-standard data with the recorded accelerometer readings. This involves human intervention and, therefore, a visual aid helps with time shifting. Furthermore, plotting features aid in visually analyzing and detecting new and useful artifacts in data, leading to new features and threshold discoveries.

5.3 Protocol

The WMMS Tool aids in standardizing the evaluation protocol for the WMMS. To comply with previous WMMS iterations, an independent video recording device is used by an investigator to capture the participants performing their specified movements. This is done while the raw motion data is being recorded on a smartphone located at the participant's hip. The independent video file is then reviewed, and all changes-of-states are time-annotated with the corresponding participant activity. This allows investigators to establish a gold-standard data set where the algorithm predictions could be evaluated. The gold-standard time is synchronized with the WMMS output by an initial shake of the smartphone, thereby providing an easily recognizable accelerometer signal and video event, allowing for the cross correlation of time-annotated change-of-state events.

The evaluation protocol can be modified by altering the gold-standard file. The protocol used in this research incorporated walking activities, a daily living environment, and various terrains and transitions into activity recognition and detection. All actions were done consecutively while the smartphone recorded raw data signals. The raw data file produced by the smartphone, as well as the time-annotated gold-standard spreadsheet file are used as main inputs to the WMMS Tool. A sample gold-standard file can be viewed in Appendix B.

5.4 Layout

This section outlines the overall WMMS Tool layout, with brief descriptions of its components. Figure 19 shows the initial launch screen. Most visual elements are hidden on launch so choosing raw file becomes intuitive at the start of an analysis session. The WMMS Tool can import and read several raw data file formats, for past WMMS versions. Once a raw file is loaded, the other user interface elements are displayed, as seen in Figure 20. Table 13 then briefly describes the GUI sections.

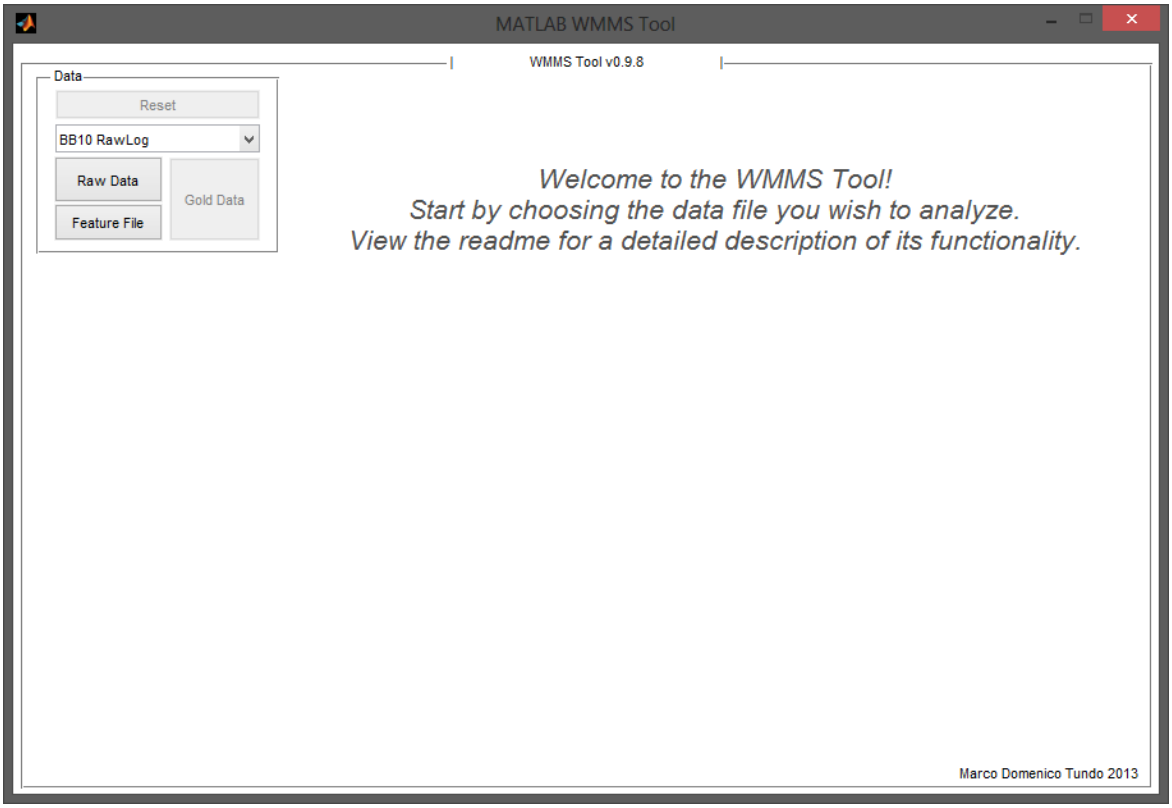


Figure 19: Launch screen for the WMMS Tool.

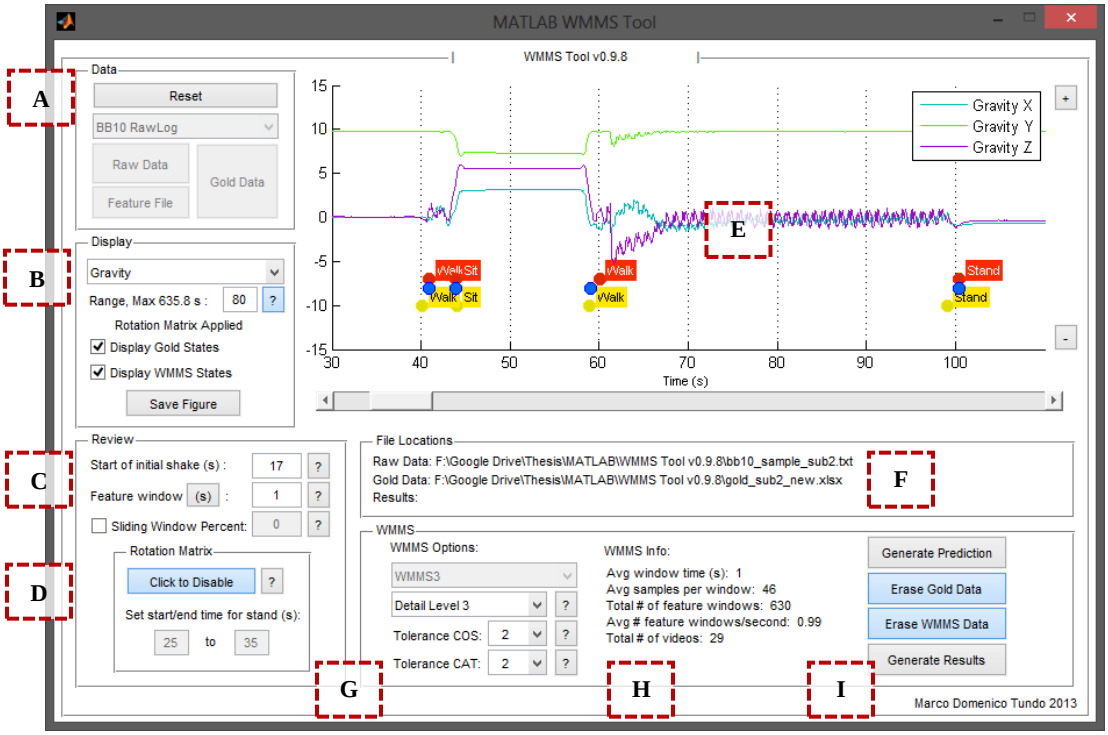


Figure 20: WMMS Tool GUI with loaded data.

Table 13: WMMS Tool GUI description.

Element	Content	Description
A	Data	Raw data and gold data input with reset button
B	Display	Display options, including raw data and features, toggles for state text on plot and save figure button
C	Review	Gold-standard timing, feature window type, and sliding window input
D	Rotation matrix	Toggle with start and end time for standing sample data
E	Plot	Plot area with visible change-of-states and plot slider underneath
F	File locations	File locations, including raw data file, gold data file and results file once generated
G	WMMS options	WMMS type, detail level and tolerance selection
H	WMMS info	Info for prediction, including averages, windows and video count
I	Prediction	Prediction generation, plotting and result file generation

5.4.1 *Data*

Raw data files can be generated from the BlackBerry 7 Data Logger [59], BlackBerry 10 Data Logger [57], or BlackBerry WMMS applications (Table 14). BlackBerry 7 WMMS2 features files can also be loaded for analysis.

Table 14: Supported functions based on input file for the WMMS Tool.

	BlackBerry 7 Raw Data	BlackBerry 7 Feature Files	BlackBerry 10 Raw Data
Display	Only Accelerometer & State Text	Only Range & State Text	All
Review	All	Only Initial Shake	All
WMMS	WMMS2 & WMMS3 detail level one	Only WMMS2	Only WMMS3

The *Gold Data* button loads the specialized gold-standard spreadsheet file used for correlation with the prediction result, as described in 5.3. The expected input to the WMMS Tool is an Excel *.xlsx* sheet where columns D and E are the time and new state code (Table 15), respectively.

The activity order in the gold-standard file can be easily modified since the state variables are based on a VLOOKUP Excel function that matches the state entered with a pre-defined state code. A template for BB7 and BB10 gold files are included in Appendix B.

Table 15: WMMS Tool state codes for gold files.

WMMS3 state codes			
States predicted		States not predicted	
State	Gold State	State	Gold State
Stand	1	Brush Teeth	21
Sit	2	Comb Hair	22
Lie	3	Wash Hands	23
Walk	4	Dry Hands	24
Stairs	5	Move Dishes	25
Small Move	6	Fill Kettle	26
		Toast Bread	27
		Wash Dishes	28

WMMS2 state codes			
States predicted		States not predicted	
State	Gold State	State	Gold State
Stand	1	Brush Teeth	9
Sit	2	Comb Hair	10
Lie	3	Wash Hands	11
Walk	4	Dry Hands	12
Ramp	5	Move Dishes	13
UpStairs	6	Fill Kettle	14
DownStairs	7	Toast Bread	15
Elevator	8	Wash Dishes	16

5.4.2 *Display*

The WMMS Tool has several display options to help the user visualize raw data plots, features, and WMMS predictions. Upon loading a BB10 raw data file, accelerometer, linear acceleration, and gravity can be plotted. Once features are generated, the end user can select individual features to plot. Gold and WMMS activity states can be displayed on the plot using text/annotations. The visible plot can be saved as a vector-based *.eps* or *.pdf* file or *.png* or *.jpg* image, using the *export_fig* functions retrieved from MATLAB File Exchange [60].

5.4.3 *Review*

The review section allows the user to modify WMMS-independent settings. This includes initial start time (determined by shaking the phone and used to synchronize the gold data), unit or type of feature window (seconds or Hertz), and sliding window activation.

The initial shake can be preloaded from the gold-standard file found in Appendix B. If the initial shake time is blank, the software will prompt the user to enter an initial shake time. Once set, WMMS data is clipped to the initial and final shake times, lining up the gold and sensor data and removing extraneous data. Gold data is also checked for extending past raw data timing. The feature window size can be set to a number of samples or time interval. Due to the BlackBerry's variable data rate, the number of samples per window will vary.

A sliding window can be enabled. The percentage used in the software is the amount the window 'slides' over the previous window. A sliding window of 25% will start the next window while the previous window is 75% complete, allowing window overlap. The value can range from 0% to 99%, since a 100% sliding window is erroneous. The sliding window is independent of window size or measurement unit (seconds or Hertz). The benefit of a sliding window is that features can be viewed with more detail when plotted, but with the processing drawback of an increase in the number of windows per trial. This outcome is shown in Figure 21 for Kurtosis sampling.

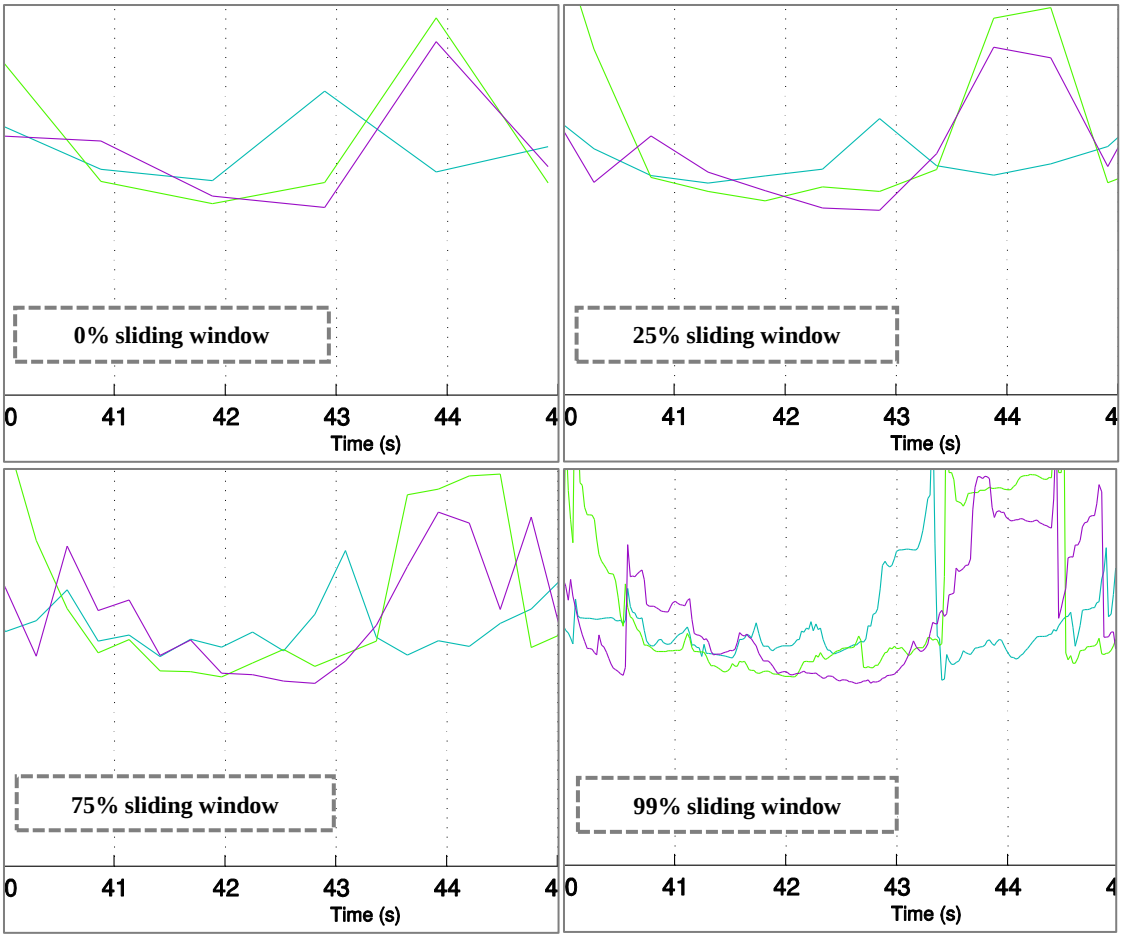


Figure 21: Kurtosis sampling as the sliding window percentage increases.

5.4.4 *Rotation matrix*

A rotation matrix can be applied to the raw data, to correct for phone positioning differences between people. The operator must locate a section where the phone is not accelerating and the participant is in a natural standing position. Start and end times for this standing period are entered into the GUI and the average acceleration vector is calculated for *x*, *y* and *z* components. The mathematical approach for calculating the rotation matrix is described in Chapter 4.

5.4.5 *Plot*

The plot area displays the raw and feature data as a function of time. The *y*-axis range can be modified, while the numerical range input box specifies the visible *x*-axis range in the plot. The slider bar in the GUI can be used to pan the plot area. These functionalities allow for full plot control and manipulation in the GUI.

Changes-of-state are depicted by red points on the plot, with their respective annotations placed on the right. Gold-standard changes-of-state can also be plotted as yellow/gold points with annotations. For WMMS3 on BB10, blue points signify a capture video command. These annotations are depicted in Figure 22.

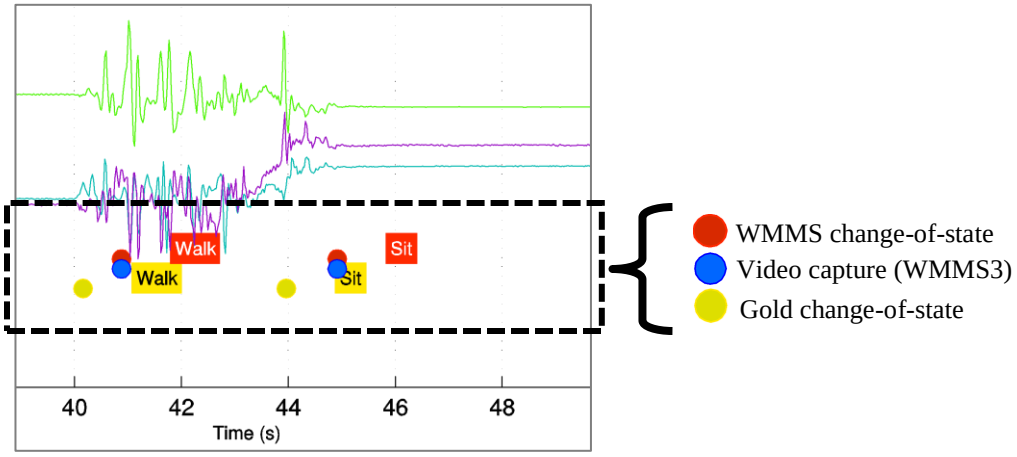


Figure 22: Example of annotated changes-of-state.

Figure 23 shows a data set with different GUI displays. The ability to plot features with annotations allows the operator to investigate the data visually.

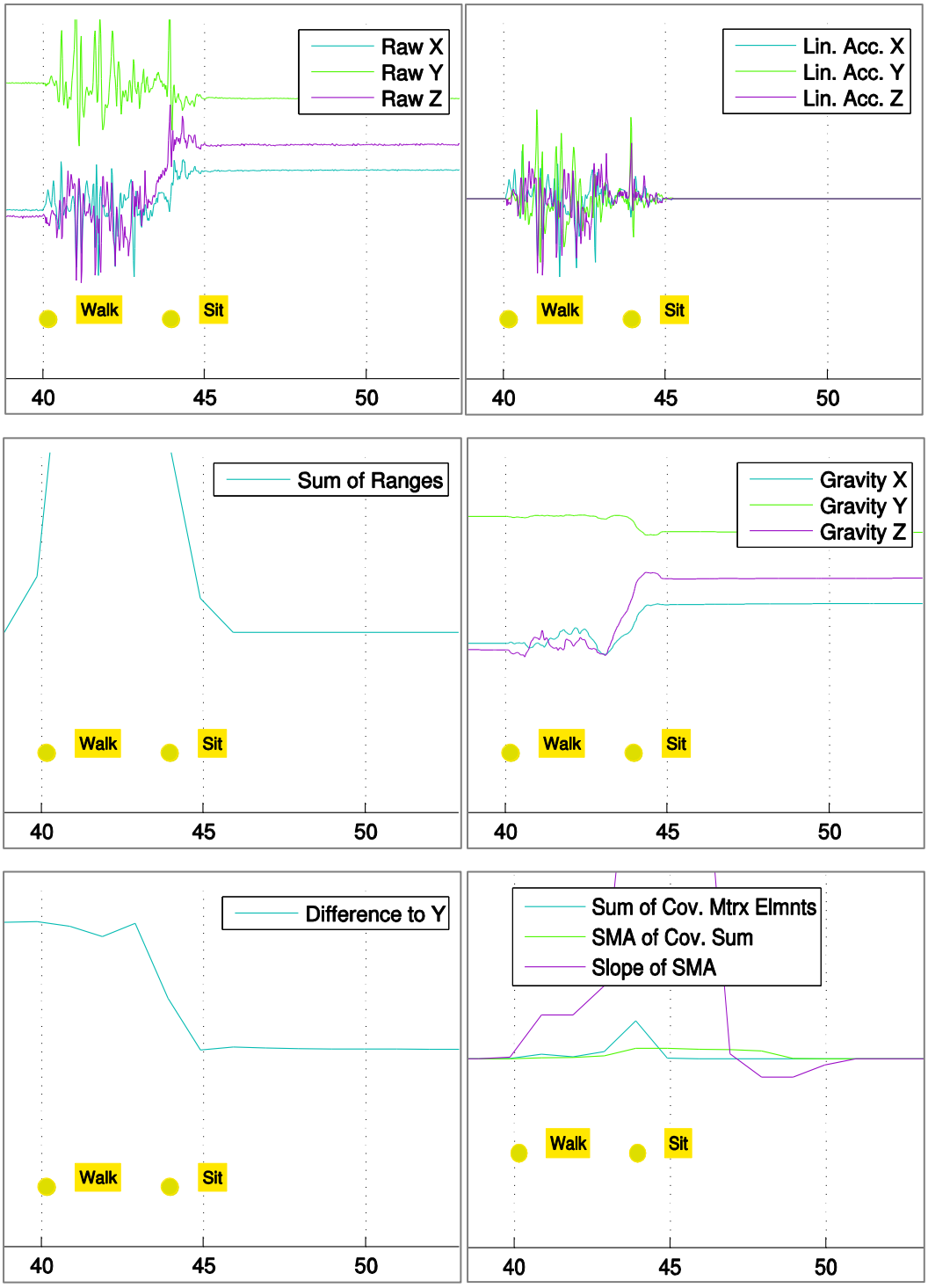


Figure 23: Plot examples of the same data sample with different feature processing.

5.4.6 *File locations*

The current raw data file, gold data file, and result file paths are displayed on the GUI for referencing purposes. The file paths for raw and gold data are also recorded in the results file.

5.4.7 *WMMS options*

The user can interchange between WMMS2 and WMMS3 prediction data, set the prediction's detail level, and set the tolerance for activities and changes-of-state for the results calculation. The detail level (Table 16) controls which activities are predicted.

Table 16: Prediction level modifications for WMMS2 and WMMS3.

Detail Level	WMMS2 predictions	WMMS3 predictions	Gold file changes
One	Mobile, Immobile	Mobile, Immobile	Elevator & Daily Living Actions become Immobile Walk & Stairs become Mobile
Two	Stand, Sit, Lie, Walk	Stand, Sit, Lie, Walk	Elevator & Daily Living Actions become Stand Stairs become Walk
Three	Stand, Sit, Lie, Walk, Upstairs, Downstairs, Elevator	Stand, Sit, Lie, Walk, Stairs, Small Movements	No Change

The change-of-state tolerance (*TOL COS)* and categorization tolerance (*TOL CAT)* are variables that modify sensitivity and specificity calculations. *TOL COS* provides a search distance tolerance for gold-standard changes-of-state around the time where the WMMS change-of-state occurs. *TOL CAT* aids in reducing false-positives in categorizing a state around state transitions. WMMS Tool tolerance is further described in section 5.5.3.1.

5.4.8 *WMMS info*

Upon classification, trial information is displayed on screen and saved in the results file. The information displayed is shown in Table 17.

Table 17: The WMMS info displayed on the WMMS Tool GUI.

Information item	Description
Average window time	The average time in seconds of every window in the trial. Due to sensor data rate variability, this value can be slightly higher or lower than the entered feature window time.
Average samples per window	The average number of sensor samples per window. Selecting a feature window size in Hertz fixes this value, with variability due to the remainder at the data set's end.
Total number of feature windows	The total number of feature windows over the entire data set.
Average number of feature windows per second	The average number of windows per second. This value increases with an increasing sliding window.
Total number of videos	The total number of video trigger events (WMMS3 with BB10 only).

5.4.9 *Prediction*

Prediction descriptions are listed in Table 18.

Table 18: The WMMS prediction and result button functionality.

Action	Description
Generate prediction	Calculate features and the WMMS prediction and then refreshes the plot.
Plot gold data	Toggles the gold changes-of-state on the plot. The gold data also trims the WMMS data, where anything before the initial shake and anything after the final shake is removed from the plot.
Plot WMMS data	Toggles the WMMS changes-of-state on the plot.
Generate results	Generates the *results_WMMSTOOL.xls* file.

5.5 Functions

A modular framework of functions and system variables was used for the WMMS Tool. Modularity would minimize system architecture difficulties in future development of the software, as well as future development in WMMS-type algorithms.

5.5.1 *Features*

An important WMMS Tool ability is manipulating the features and the feature windows. Separate WMMS2 and WMMS3 functions calculate activity features. These features and their column numbers in the features array are listed in Appendix C.

Feature calculation inputs include the raw data, window size, window unit type (Hertz or seconds), and sliding window percentage. Feature function outputs include the feature array and a vector containing the feature window properties; including, average window size, average window standard deviation, average window samples, average window samples standard deviation, and number of windows per second.

5.5.2 *WMMS functions*

WMMS functions are called once the feature array has been constructed. These functions run the algorithms described in 3.2.2 and 6.2.2. The output is a continuous array of timestamps and activity codes for the expected activities (Figure 24).

…	1	1	1	1	4	4	4	4	4	4	4	2	2	2	2	…

Figure 24: Segment of a WMMS expected act array with single-second time stamped windows and their respective state codes.

5.5.3 *Results*

After calculating the results, an output file is generated for participant or WMMS algorithm evaluation (Appendix D). This file includes the outputs, file locations for the reviewed subject, feature information, and overall WMMS Tool setting information.

5.5.3.1 Tolerance

A gold-standard data set array is generated to become a time-specific reference to the predicted WMMS result. The array is time-shifted so the raw data correlates in time with the gold-standard set, using the start of the initial smartphone shake. Then, the gold data is stretched to span the feature vector length, synchronizing with the feature window timing. Figure 25 demonstrates how a gold-standard set of data is stretched. A sample excerpt from a gold-standard file shows a participant sitting, then standing, and then transitioning to a walk at 51 seconds, and back to a stand at 58 seconds. The state code data is fixed into window sizes that correspond to what is selected in the Tool. In the example below, the window size selected is 1 second.

(a)

New Action	Time (s)	State Code
Sit	30	2
Stand	45	1
Walk	51	4
Stand	58	1

(b)

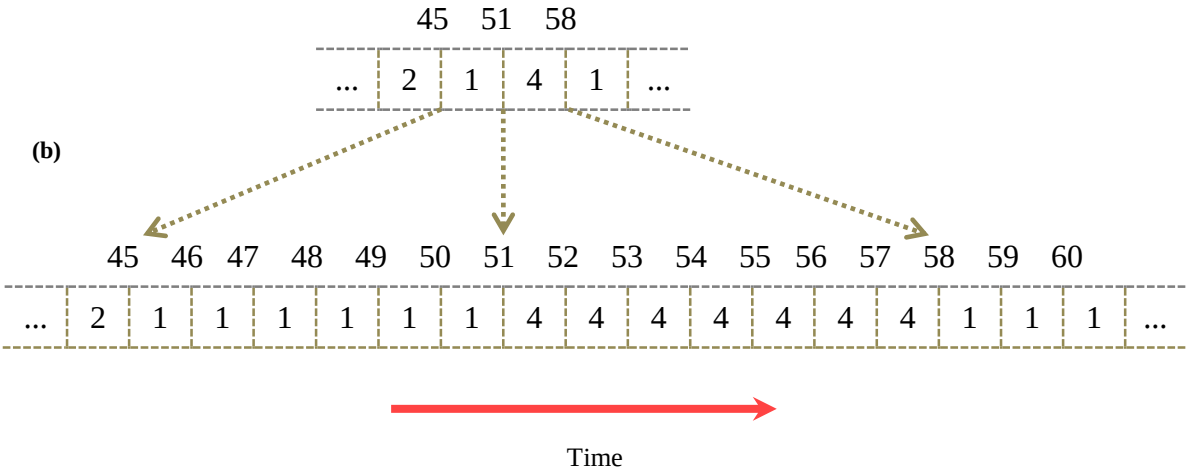


Figure 25: (a) Raw gold data retrieved from file (b) Stretched gold data synchronized with a feature window timing of one second.

The gold data and WMMS prediction arrays are then correlated with each other. The results are affected by two tolerance variables.

Firstly, *TOL COS* allows for the search of a gold change-of-state around a predicted change-of-state. Figure 26 demonstrates two separate change-of-state tolerance values. In groups (a) and (b), the top horizontal line of boxes signifies prediction state windows, while the bottom boxes signify gold stretched data. Condition (a) demonstrates a *TOL COS* of one (one behind, one ahead), where no change-of-state is discovered in the gold set, while condition (b) is a *TOL COS* of three (three behind, three ahead), and a change-of-state is positively discovered.

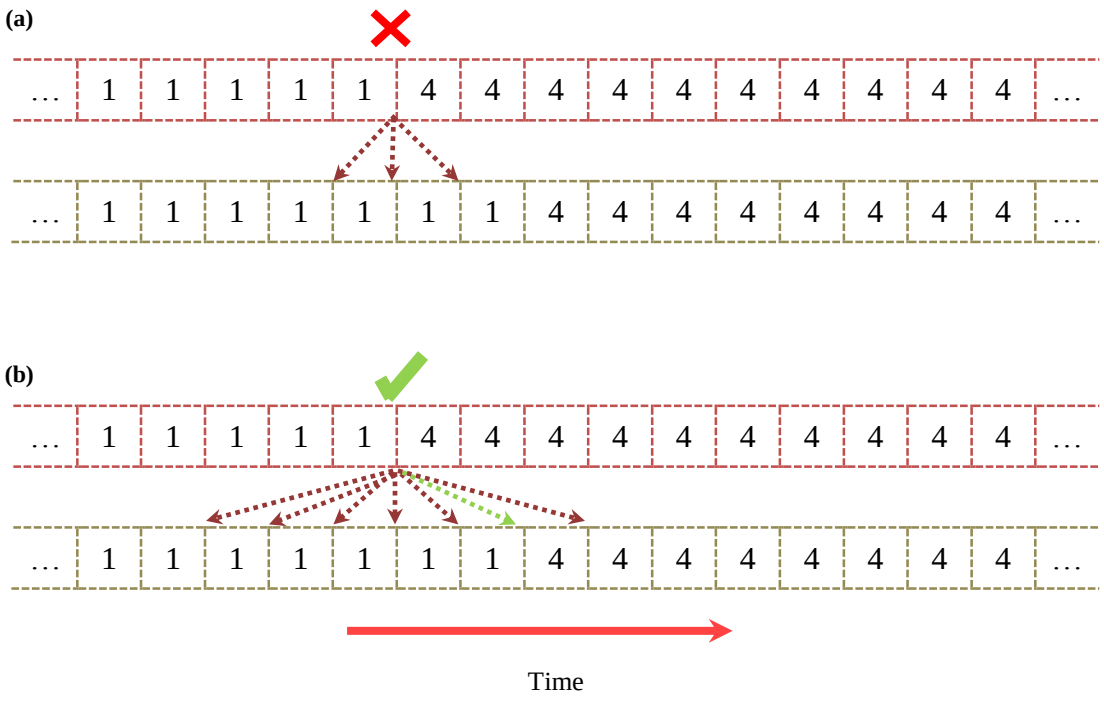


Figure 26: Prediction (top array) and gold-standard (bottom array) for (a) change-of-state tolerance of one (false positive since no change-of-state is found) and (b) change-of-state tolerance of three (true positive since a gold change-of-state is within the desired search area).

When a gold change-of-state is discovered, it is flagged and therefore cannot be discovered by the tolerance algorithm again. Increasing this tolerance for matching a prediction change-of-state near a gold change-of-state results in an overall increase in change-of-state true positives, thereby increasing the sensitivity for changes-of-state.

Secondly, the *TOL CAT* algorithm discovers changes-of-state in gold data and removes the specified amount of surrounding windows from categorization results. Figure 27 demonstrates two categorization tolerance values. In groups (a) and (b), the top horizontal line of boxes signifies gold stretched data, while the bottom boxes signify the prediction. Condition (a) is a *TOL CAT* of zero, which results in two false positives occurring. Condition (b) is a *TOL CAT* of two (remove two behind, remove two ahead), which results in those false positives being removed from the results, but also two true positives being removed as well.

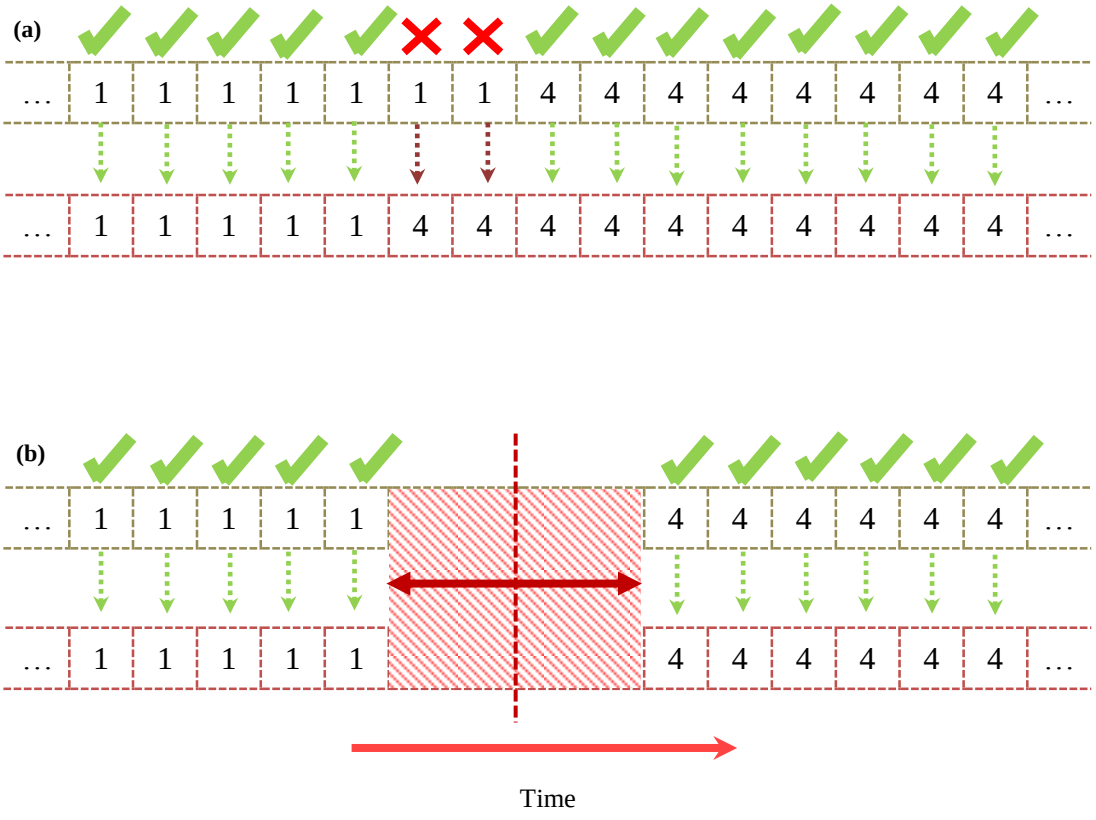


Figure 27: Prediction (top array) and gold-standard (bottom array) for (a) categorization tolerance of zero (two false positives) and (b) categorization tolerance of two (false positives are removed from the results, but two true positives are removed as well).

5.5.3.2 Sensitivity & specificity

Sensitivity (also known as recall rate) and specificity are evaluations of a binomial (binary) classification. They test the ability to determine if a certain element has a certain trait. Sensitivity measures the proportion of gold-standard positives that are correctly identified as positive. Specificity measures the proportion of gold-standard negatives that are correctly identified as negative.

A common example used when describing specificity and sensitivity is testing whether someone is ill [61]. In this circumstance, the patient having the illness is a positive (Table 19).

		Truth	
		The patient is ill.	The patient is healthy.
Prediction	The patient is ill.	True positive	False negative
	The patient is healthy.	False positive	True negative

Table 19: Confusion matrix for sensitivity and specificity

where,

1) True positive: correctly identified the illness,
2) False positive: incorrectly identified the illness,
3) True negative: correctly rejected the illness,
4) False negative: incorrectly rejected the illness.

Sensitivity and specificity can then be written as,

$$\text{sensitivity} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}\ , \tag{5.1}$$

$$\text{specificity} = \frac{\text{true negatives}}{\text{true negatives} + \text{false positives}}\ . \tag{5.2}$$

Precision, the test of how well positive outcomes are predicted, can be written as,

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}\ . \tag{5.3}$$

The balanced F-Score (F_1) can then be used to measure the test's overall accuracy, calculated as the harmonic mean between sensitivity and precision with equal weighting:

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{sensitivty}}{\text{precision} + \text{sensitivity}}. \tag{5.4}$$

Emphasis can be placed on either precision or sensitivity in calculating the F-Score. Using equations 5.1, 5.2, and 5.3, a non-balanced F-Score weighting is described as,

$$F_\beta = \frac{(1+\beta) \cdot \text{true positives}}{(1+\beta^2) \cdot \text{true positives} + \beta^2 \cdot \text{false negatives} + \text{false positives}}, \tag{5.5}$$

where $\beta = 0.5$ puts emphasis on precision than recall, and $\beta = 2$ places emphasis on recall over precision.

5.6 Conclusion

The WMMS Tool facilitates WMMS development and evaluation. Several raw data formats can be loaded, with many display options and plot settings available for reviewing and evaluating a trial. Its framework allows for the development of new features, options, or WMMS iterations.

Chapter 6
WMMS3 DEVELOPMENT & EVALUATION

6.1 Introduction

WMMS3 requires hardware that supports inertial measurement unit sensors. A BlackBerry Z10 smartphone was used for development, providing a variety of outputs based on tri-axis accelerometer, gyroscope, and magnetometer sensors. The WMMS3 algorithm was programmed and developed entirely in the MATLAB environment, using the WMMS Tool for feature design and evaluation. Data collection was performed on the BlackBerry Z10 smartphone using the TOHRC Data Logger for BB10 application [57].

6.1.1 *Annotations*

This chapter includes plots generated by the WMMS Tool for sensor data and generated features relative to time, as shown in Figure 28. A legend describes the plotted data, where the *y*-axis is the signal magnitude. The device orientation (smartphone on the mannequin) and changes-of-state (coloured dots) are overlaid on the plot area. The predicted changes-of-states are shown as red dots at their respective time-interval, while the video triggers (a trigger to capture a 3-second video clip) are shown as blue dots. The gold-standard data, as described in the previous chapter, are annotated gold dots.

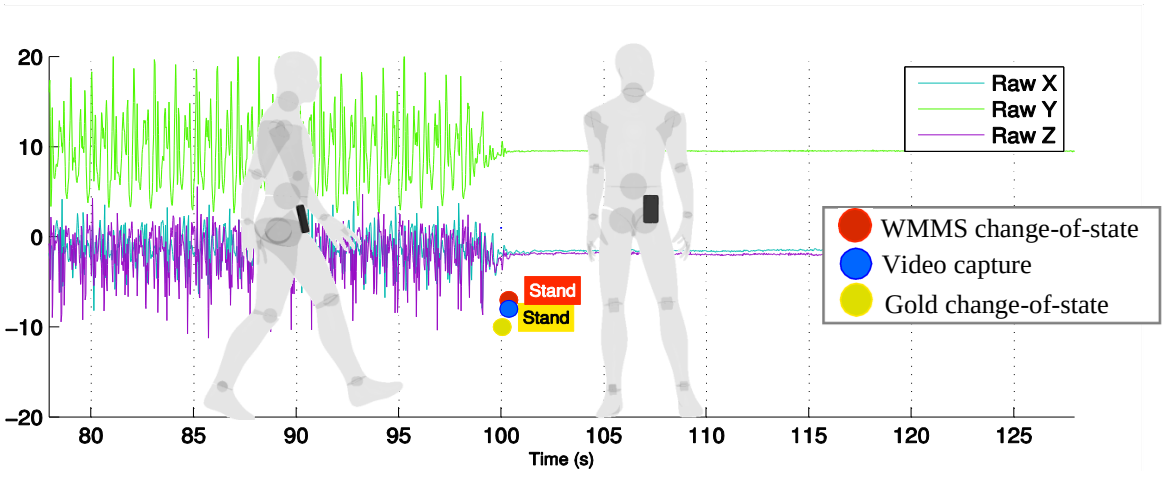


Figure 28: Example of manikin and CoS annotations demonstrating device orientation.

6.1.2 *BlackBerry 10 devices*

WMMS3 runs on BB10 devices (Table 20), and its dependency on new hardware sensors restricts the ability for the algorithm to run on earlier BlackBerry handsets.

Table 20: BlackBerry 10 models and specifications (2013) [62].

			
	BlackBerry Z10	BlackBerry Q10	BlackBerry Q5
Display	4.2”, 1280 x 768 resolution, 356 ppi	3.1”, 720 x 720 resolution, 329 ppi	3.1”, 720 x 720 resolution, 329 ppi
CPU	Qualcomm Snapdragon S4, Dual-core 1.5 Ghz Krait CPU	Qualcomm Snapdragon S4, Dual-core 1.5 Ghz Krait CPU (LTE model)	Qualcomm Snapdragon S4, Dual-core 1.2 Ghz CPU
Storage	Up to 64 GB storage	Up to 64 GB Storage	Up to 32 GB Storage
RAM	2 GB RAM	2 GB RAM	2 GB RAM
Network	4G LTE Ready	4G LTE Ready	4G LTE Ready
Standby	Up to 13 days standby time (3G)	Up to 15 days standby time (3G)	Up to 14 days standby time (3G)
Sensors	Accelerometer, gyro, compass	Accelerometer, gyro, compass	Accelerometer, gyro, compass
Battery	Li-Ion 2100 mAh battery	Li-Ion 1800 mAh battery	Li-Ion 2180 mAh battery (non-remove)

6.1.3 *Sensors*

Devices running BB10 include new hardware sensors not seen in previous BlackBerry smartphones. Previous devices included a tri-axis accelerometer, but with limitations on sample rates. The accelerometer included with the BlackBerry Z10 allows for sample rates of up to 50 Hz. New sensors include a tri-axis gyroscope and magnetometer. These sensors are fully accessible through API commands in Native SDK.

6.1.3.1 Accelerometer

A tri-axis accelerometer senses applied linear acceleration and the acceleration due to gravity (Figure 29). On a smartphone, accelerometers are aligned to the device's body axis and measure the total acceleration acting on the device. If the device is resting, the component magnitudes sum to 9.81 m/s^2 . Once the device experiences a user-applied force, this magnitude increases, making it difficult to estimate the device's orientation relative to the Earth. The acceleration due to gravity is the only accelerometer measurement that indicates how the phone is positioned in space relative to the ground.

Figure 29: BlackBerry Z10 accelerometer components.

Accelerometer data can be represented as,

$$\vec{A} = \frac{d\vec{v}}{dt} = (\vec{g} + \vec{l}), \tag{6.1}$$

$$\begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} = \begin{pmatrix} g_x + l_x \\ g_y + l_y \\ g_z + l_z \end{pmatrix}, \tag{6.2}$$

where $\vec{A}$ (acceleration), $\vec{g}$ (acceleration due to gravity) and $\vec{l}$ (applied linear acceleration) are in m/s^2.

6.1.3.2 Gyroscope

A gyroscope measures angular velocity around an axis on a body-fixed coordinate system (Figure 30). A tri-axis gyroscope is able to measure rotation around the *x*, *y*, and *z* axes, also known as pitch, roll, and yaw.

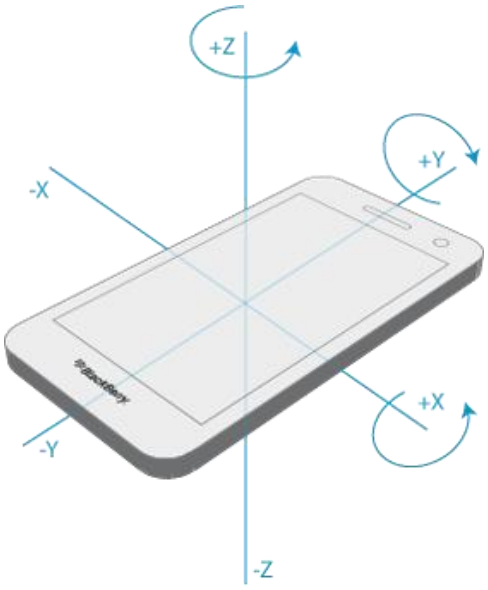


Figure 30: BlackBerry Z10 gyroscope.

Gyroscope data can be represented as,

$$\vec{\omega} = \frac{d\phi}{dt} = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}, \qquad (6.3)$$

where $\vec{\omega}$ (angular velocity) is in radians per second.

6.1.3.3 Magnetometer

A magnetometer measures the Earth's magnetic field (Figure 31). A tri-axis magnetometer can be used to estimate magnetic north by reading the Earth's magnetic flux density ($\vec{T}$), allowing for the quantification of device heading.

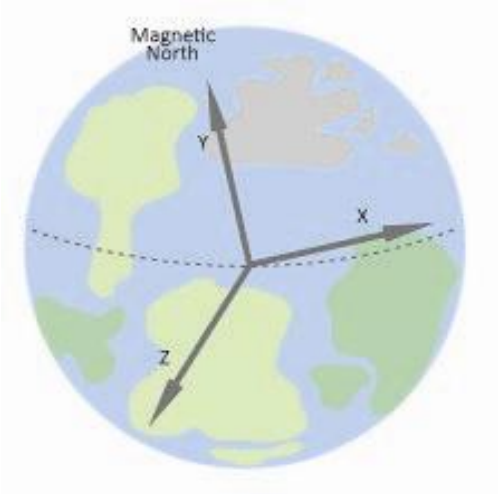


Figure 31: BlackBerry Z10 diagram for magnetic north.

Magnetometer data can be represented as,

$$\vec{T} = \begin{pmatrix} T_x \\ T_y \\ T_z \end{pmatrix}, \qquad (6.4)$$

where $\vec{T}$ is in $\mathrm{Wb/m^2}$ or $\mathrm{N \cdot s/C \cdot m}$ and T_x, T_y, T_z are the *x*, *y*, and *z* components of the magnetometer measurement.

6.1.3.4 Sensor fusion

An IMU is a collection of microelectromechanical systems (MEMS) and sensor fusion algorithms. On the BlackBerry Z10, the IMU fuses the accelerometer, gyroscope and magnetometer sensors, and splits acceleration components into applied linear acceleration (ALA) and acceleration due to gravity (ADG), although the device manufacturer does not indicate how this is accomplished [63]. There exist open-source algorithms capable of extracting the linear acceleration from the accelerometer signal through software-based sensor fusion algorithms [64]. Gyroscope, accelerometer, and magnetometer sensor fusion reduces drift, noise, and error in output signals.

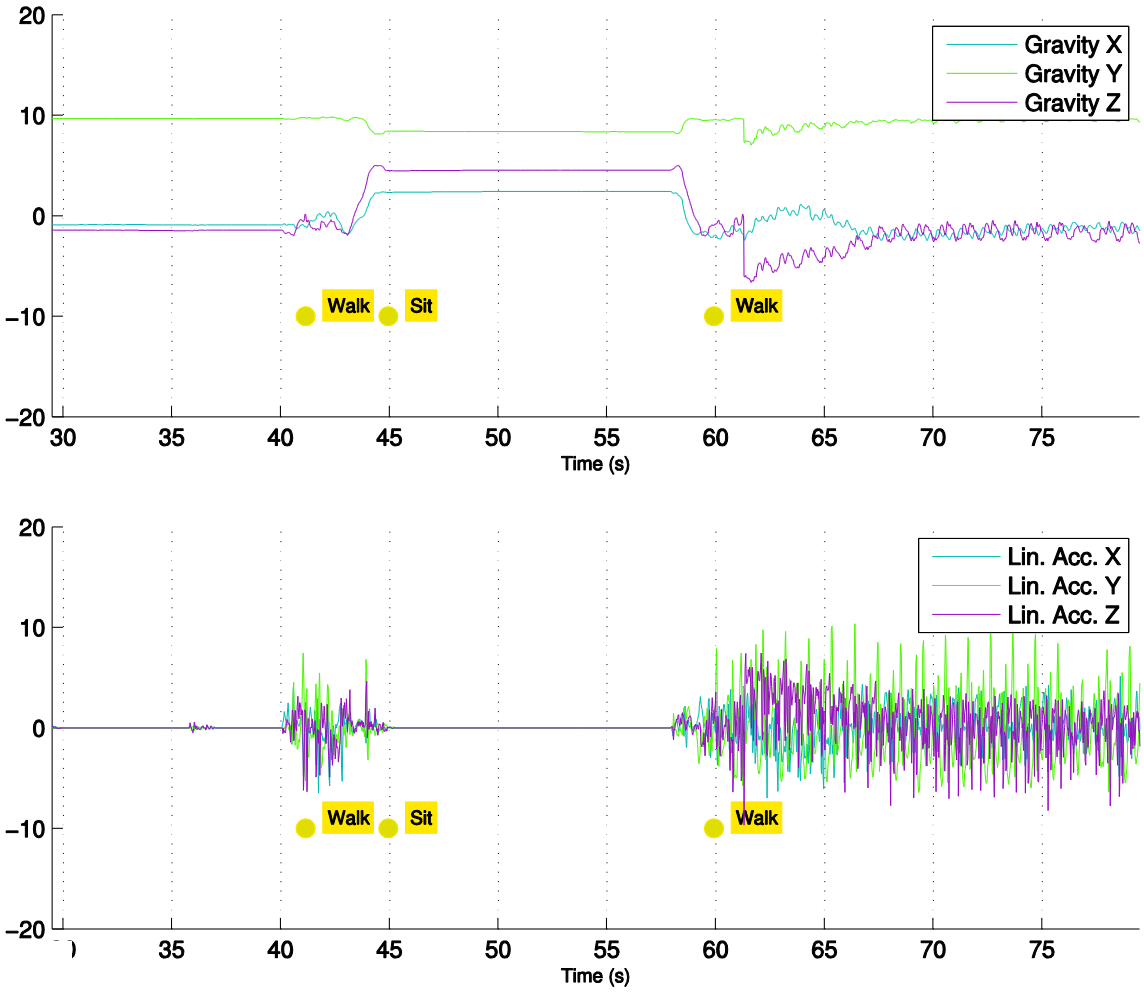


Figure 32: ADG (top) and ALA (bottom) output for standing to walking to sitting to walking.

The WMMS3 algorithm uses the ALA and ADG components exclusively (Figure 32). Therefore, only devices that output the gravity and linear acceleration vectors (or equivalently devices that contain an accelerometer, gyroscope and magnetometer) can run the algorithm.

6.2 Architecture

The WMMS3 consists of features and a decision tree for categorizing mobility activities. Features are variables calculated from raw sensor data, which are used in the decision-making. The decision tree also employs variables that count the length of time spent in a mobile or immobile state, which is then used to determine if a new state is active, and to aid in removing false positives in changes-of-state. The algorithm is divided into two stages, with separate orientation independent predictions (stage one) and orientation dependant predictions (stage two).

6.2.1 *Features*

WMMS3 features are derived from both the ALA and ADG vectors, which are produced by the BlackBerry itself. Three features are calculated using the ALA vector (Sum of ranges, simple moving average and sum of standard deviation) and two features are based on the ADG vector (difference to *y*-axis, variance sum difference). The features presented in this section are structured as,

$$x_i, y_i, z_i \;{}_{i=1}^{n}\;{}_{j=1}^{N}, \tag{6.5}$$

where the vector of i component accelerations is inside a feature window of size n, with j windows for data size N. The components *x*, *y* and *z* are for ALA or ADG where noted.

6.2.1.1 Difference to y-axis

With the ADG, a full description of device orientation can be constructed, regardless of device movement. This 'difference to *y*-axis' feature is described as:

$$Difftoy_j = \frac{1}{n}\sum_{i=1}^{n} \; y_i - x_i - z_i \quad , \tag{6.6}$$

where n is the size of the feature window. The ADG component in the y direction is subtracted by the x and z components. Since the device is initially corrected to an upright position, $Difftoy_j$ remains close to 10 m/s^2. Sitting, standing and lying down can be detected with this single feature, since the x and z components becoming larger and y becomes smaller when the device is orientated in any orientation other than upright.

The gravity vector signal from a BlackBerry 10 device is sufficiently clean so that no spikes in magnitude occur with any gravity component, making it suitable for this purpose. Figure 33 demonstrates sitting to walking, which is clearly shown by a change in $Difftoy_j$. Standing, walking and performing other actions show minimal change in this feature.

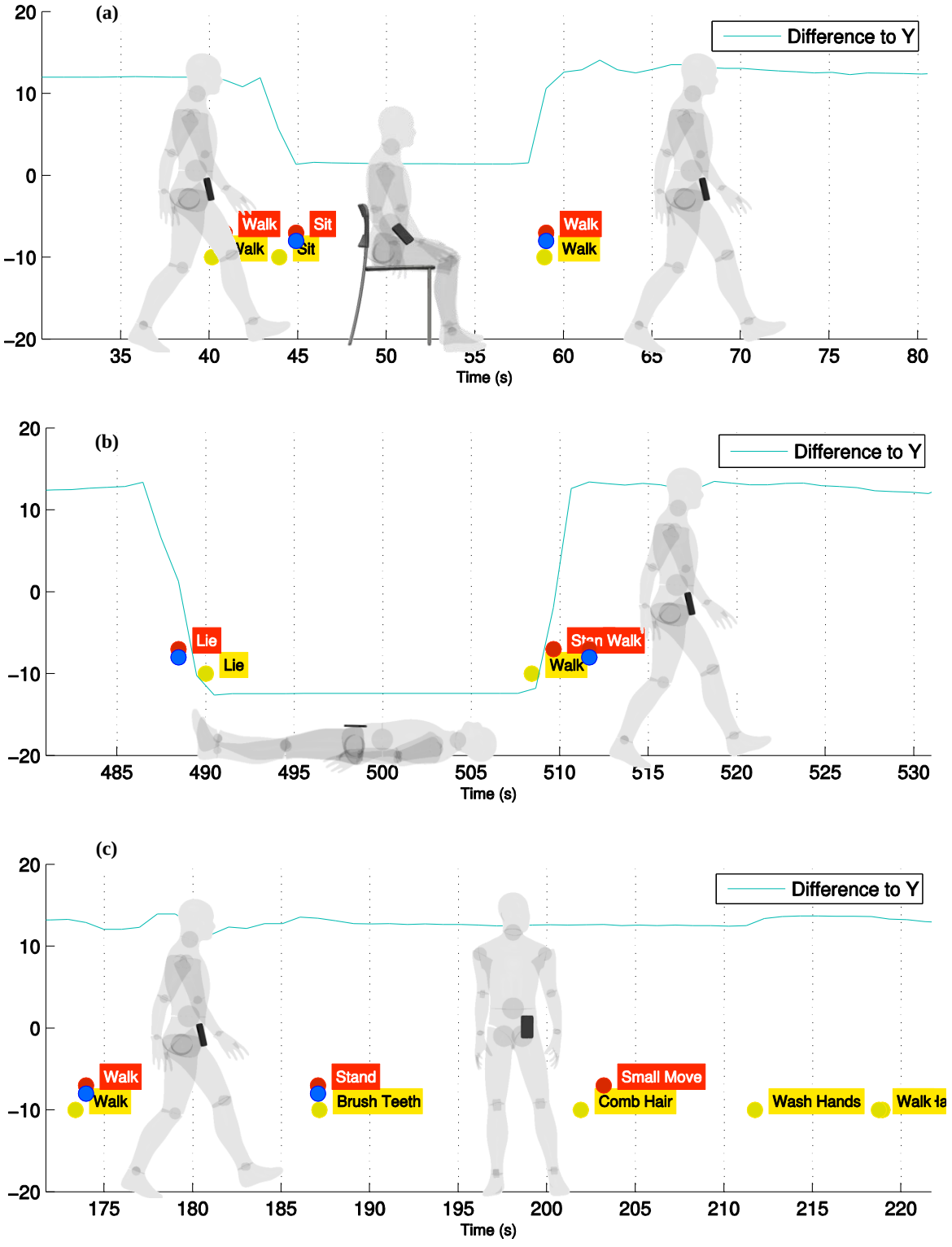


Figure 33: Difference to y-axis for, (a) sit-to-walk, (b) lie-to-walk, (c) walk-stand.

6.2.1.2 *Sum of ranges*

Sum of ranges is as a useful feature for mobility change-of-state detection [50]. With the linear acceleration signal provided by the BB10 API, the sum of ranges in expected to increase the accuracy in detecting smartphone movement compared with previous BlackBerry phones (Figure 34). The sum of ranges is calculated as,

$$SoR_j = range_{x,j} + range_{y,j} + range_{z,j}, \tag{6.7}$$

where $range_{a,j} = \max_j a_{i \in j} - \min_j a_{i \in j}$ and a is the x, y, or z linear acceleration component.

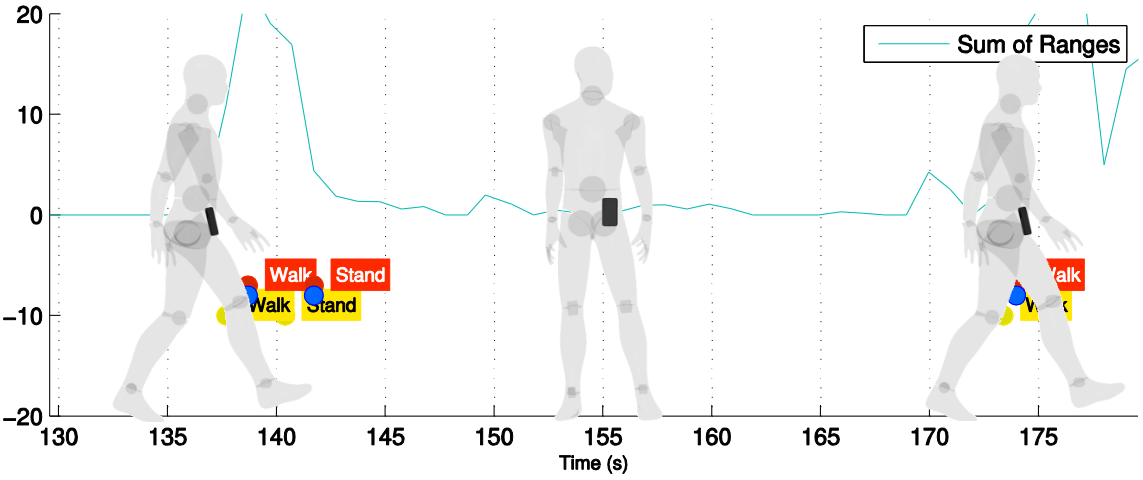


Figure 34: Sum of ranges for walk-to-stand-to-walk.

6.2.1.3 *Simple moving average*

Simple moving average (SMA) is similar to the signal magnitude area feature proposed by Khan [14] and Wu [15]. These studies take the sum of maximum accelerations for a number of previous windows. The average SoR_j range over the current and previous three windows was more effective in detecting mobility versus immobility than the linear acceleration alone. This simple moving average (Figure 35) is given by,

$$SMA_j = \begin{cases} \frac{1}{4}\sum_{k=j}^{k+3} \sum range_{-k}(x,y,z)_{i\in -k} & \text{for } j > 4 \\ 0 & \text{for } j \le 4 \end{cases}, \quad (6.8)$$

where *x*, *y*, and *z* are linear acceleration components for windows *j*, *j-1*, *j-2* and *j-3*.

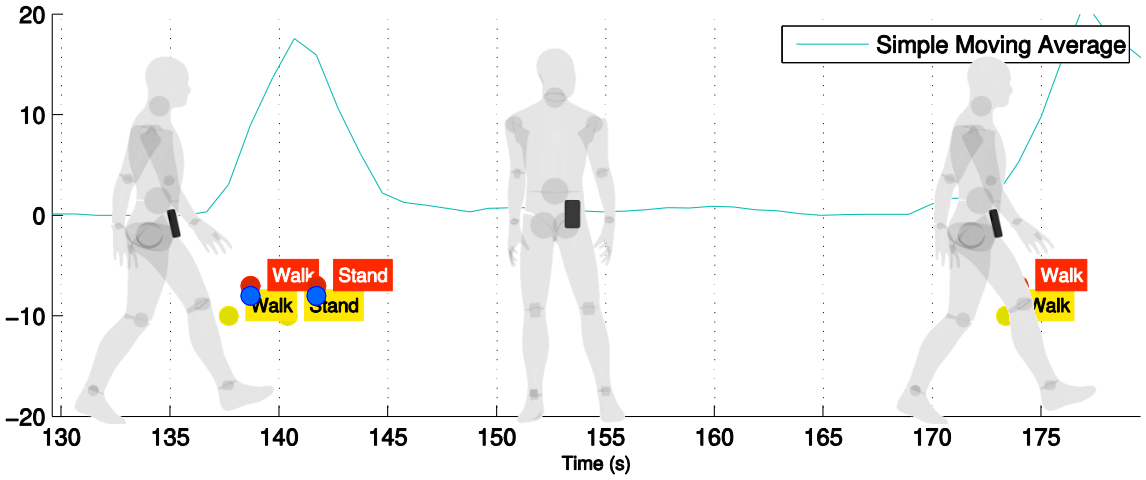


Figure 35: Simple moving average for walk-to-stand-to-walk.

6.2.1.4 Sum of standard deviation

A standard deviation feature is commonly used in HAR studies [16], [18], [32], [44]. By summing the standard deviation of components of ALA, device orientation was found not to affect mobility assessment. For the WMMS3, the sum of standard deviations (Figure 36) is calculated as,

$$SStD_j = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} x_i - \bar{x}_j^{\,2}} + \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} y_i - \bar{y}_j^{\,2}} + \sqrt{\frac{1}{n-1}\sum_{i=1}^{n} z_i - \bar{z}_j^{\,2}}, \quad (6.9)$$

where x, y, and z are linear acceleration components for windows j and the overbar indicates an average.

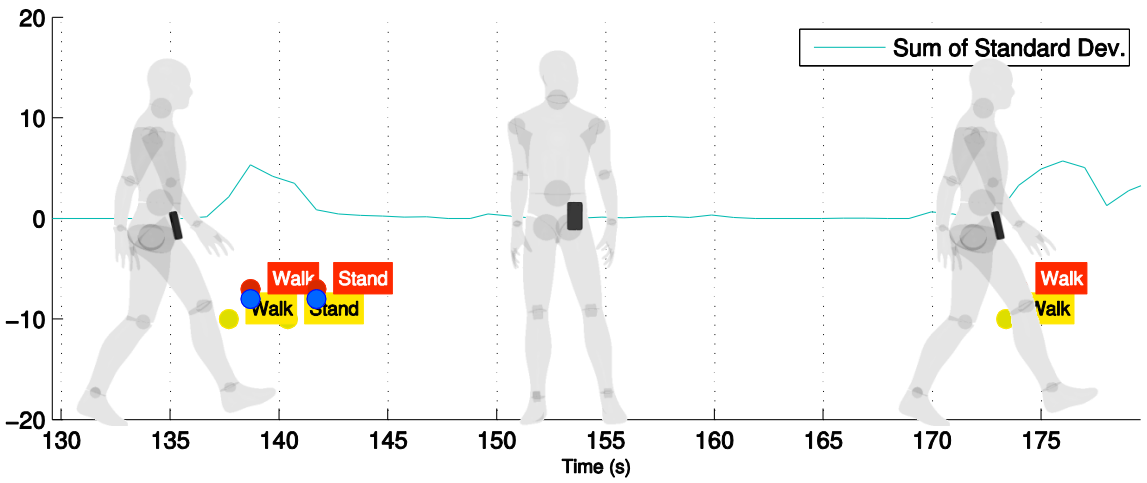


Figure 36: Sum of standard deviations for walk-to-stand-to-walk.

6.2.1.5 Variance sum difference

Variance and covariance features from accelerometer data have been used to detect stair climbing [17], [27], [28], [33], [34]. The WMMS3 algorithm calculates a variance-covariance matrix defined as,

$$\mathrm{cov}_{ab,j} = \frac{1}{n}\sum_{i=1}^{n}(a_i - \bar{a}_j)(b_i - \bar{b}_j), \tag{6.10}$$

$$V_j = \begin{bmatrix} \mathrm{cov}_{xx,j} & \mathrm{cov}_{xy,j} & \mathrm{cov}_{xz,j} \\ \mathrm{cov}_{yx,j} & \mathrm{cov}_{yy,j} & \mathrm{cov}_{yz,j} \\ \mathrm{cov}_{zx,j} & \mathrm{cov}_{zy,j} & \mathrm{cov}_{zz,j} \end{bmatrix}. \tag{6.11}$$

where *x*, *y*, and *z* are the gravity vector components for window *j*. On the main diagonal, the variance-covariance matrix, V_j , demonstrates variances of each acceleration component, while the off-diagonals represent covariances between the acceleration components. The sum of the diagonal variances are calculated as,

$$\mathrm{SUMcov}_j = \mathrm{cov}_{xx,j} + \mathrm{cov}_{yy,j} + \mathrm{cov}_{zz,j}, \tag{6.12}$$

which can be written as,

$$\mathrm{SUMcov}_j = \frac{1}{n}\sum_{i=1}^{n}\left[(x_i - \bar{x}_j)^2 + (y_i - \bar{y}_j)^2 + (z_i - \bar{z}_j)^2\right]. \tag{6.13}$$

The simple moving average SUMcov_j for the current and four previous windows is then calculated as,

$$\text{SMAcov}_j = \begin{cases} \frac{1}{5}\sum_{k=j}^{k+4} \text{SUMcov}_{-k} & \text{for } j > 5 \\ 0 & \text{for } j \le 5 \end{cases}. \qquad (6.14)$$

SMA of SUMcov_j tends to increase during stair ascent (labelled as 'Slope of SMA' in Figure 37). To account for an increase or decrease in this value, the difference (as a proxy for slope of SMA) between successive values of SMAcov_j was calculated over four feature windows. Taking only the maximum of the set,

$$\text{MAXDIFFcov}_j = \begin{cases} \max_{k\,:\,(j-4)\to j} |\text{SMAcov}_k - \text{SMAcov}_{k-1}| & \text{for } j > 5 \\ 0 & \text{for } j \le 5 \end{cases}. \qquad (6.15)$$

where $k = (j-4),\ (j-3),\ (j-2),\ (j-1),\ j$.

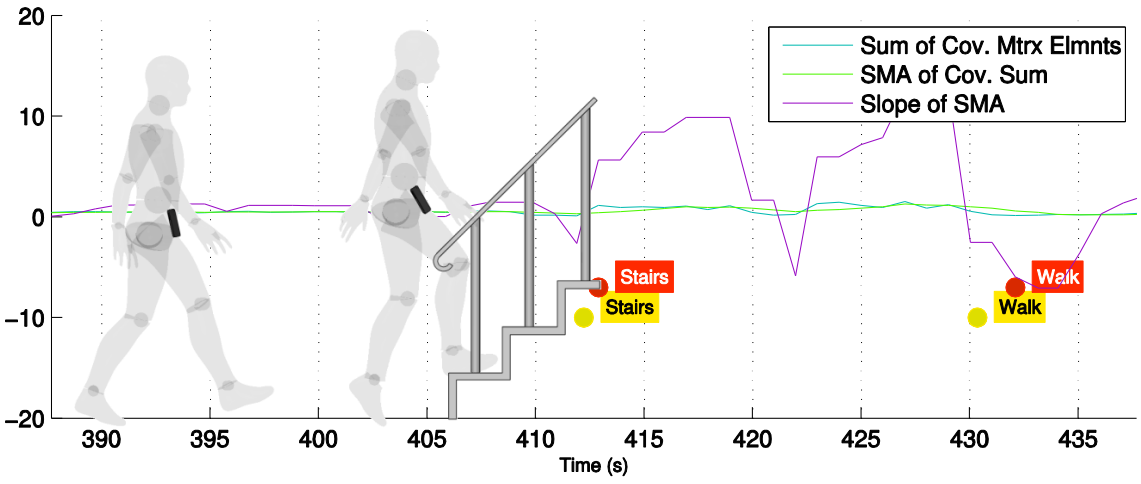


Figure 37: Variance sum difference for walk-to-stairs.

6.2.2 *Algorithm*

The WMMS3 algorithm is separated into two main stages, where the first stage is orientation independent, and the second is orientation dependant.

The first stage assesses whether the user is mobile or immobile. This stage is phone orientation independent, meaning that the device may be located anywhere on pelvis for the algorithm to work correctly. The second stage then takes output from the first stage and refines the detail of classification to detail level two and detail level three. Detail level two splits immobility into sitting, standing, or lying. Mobility in detail level two is categorized strictly as walking. Detail level three may then split mobility into walking, stairs, or small movements. Table 21 summarizes the algorithm detail levels and predicted states.

Table 21: WMMS3 algorithm detail level and prediction.

Stage	Detail Level	Possible States	Features	Thresholds
One	One	Immobile Mobile	Sum of range (Corrected Linear) Simple moving average (Corrected Linear) Sum of standard deviation (Corrected Linear)	Predefined for all users
Two	Two	Standing Sitting Walking Lying	Difference to Y (Gravity)	Predefined for all users
	Three	Stairs Small Movements	Variance sum average difference (Gravity)	S1 S2

6.2.2.1 Stage one

The first stage assesses mobility versus immobility using features and false-positive correction measures. The possible categorizations are binary: mobile-immobile and immobile-mobile. Validation tests and corrections in categorization are made to minimize change-of-state false positives, while still providing the ability to trigger a video event. These corrections are designed to alleviate common issues that arise in detecting natural human mobility. Figure 38 shows the decision tree for stage one. The variables used in the state include:

1) *Mobile Strength:* A count of thresholds surpassed incurring a mobile state,
2) *Active Count:* A counter that increases while in a mobile state,
3) *Inactive Count:* A counter that increases while in an immobile state,
4) *Video Wait:* A temporary variable that flags or cancels a video trigger,
5) *Take Video:* A video trigger.

Sum of ranges (SoR_j), simple moving average ($SStD_j$), and sum of standard deviation (SMA_j,) features - each with their specific threshold - are used to determine whether the user is mobile or not. The three thresholds are set as constants for all users. The three features used in stage one must each exceed their specific threshold in order to register a mobile state.

At every feature window, *Mobile Strength* is initialized to null. Each time a single threshold is surpassed by its respective feature, *Mobile Strength* for that window is increased by 1. If a *Mobile Strength* value of 3 is reached for a single window (all stage one features surpassed their respective thresholds), then current window is categorized as a mobile state. Thresholds for stage one are user independent and static. Sum of ranges (SoR_j), and sum of standard deviation ($SStD_j$) thresholds are set to 1, while the simple moving average (SMA_j) threshold is set to 5.

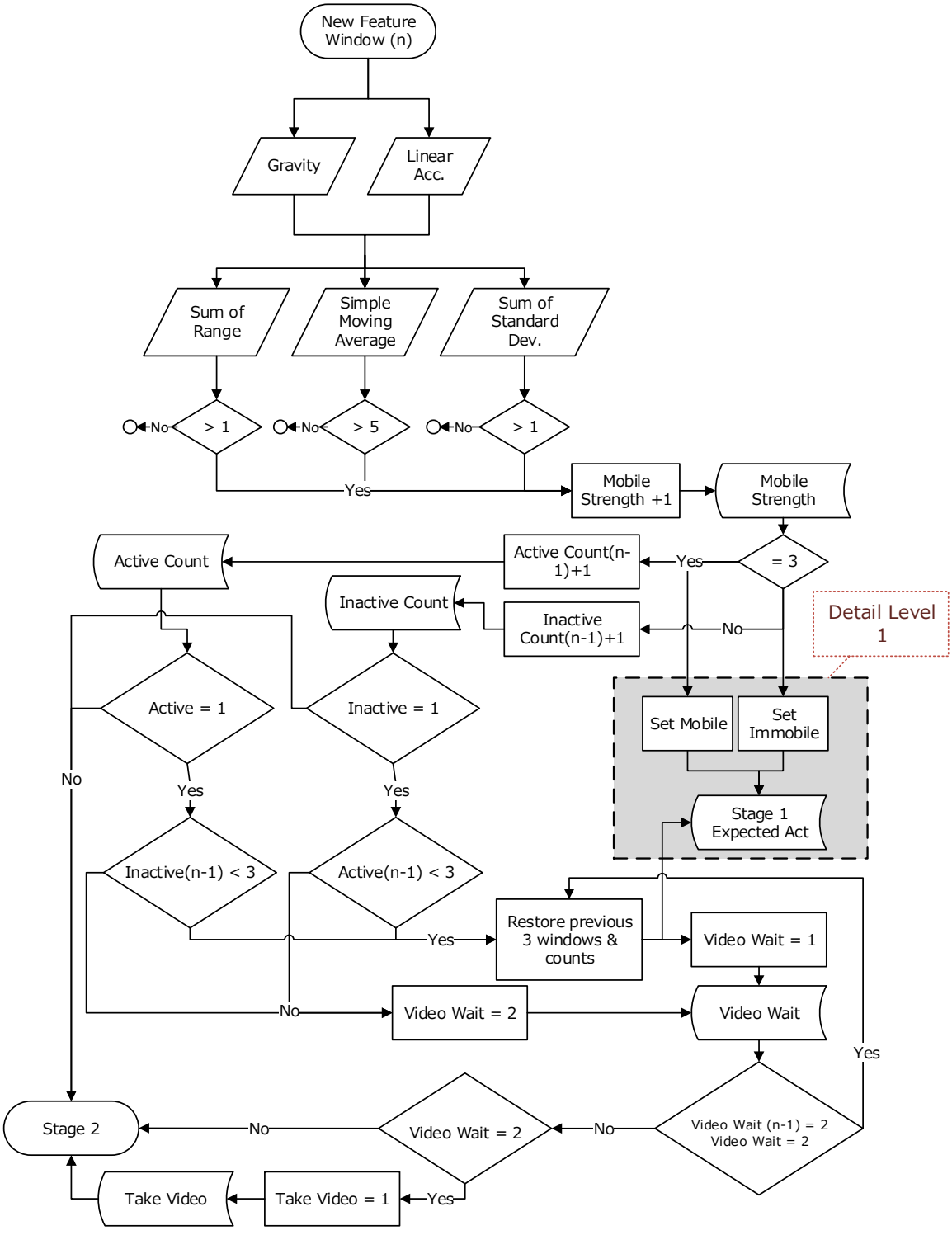


Figure 38: WMMS3 algorithm stage one.

Once all three thresholds have been surpassed in a window, the *Active Count* variable increases by one from its previous window value, while *Inactive Count* is set to zero. If *Mobile Strength* is not three, then *Inactive Count* will increase by one from its previous value, and *Active Count* is set to zero.

Two validation tests are used to reduce the number of change-of-state false positives. The first test accounts for consecutive changes-of-state that occur in a small amount of time. This can happen if the person takes longer to transition between a mobile and immobile state or if the person shakes or pivots while in an immobile state. If two consecutive changes-of-states are found, the algorithm restores the previous three windows and continues the *Active Count* or *Inactive Count*, depending on state (Figure 39).

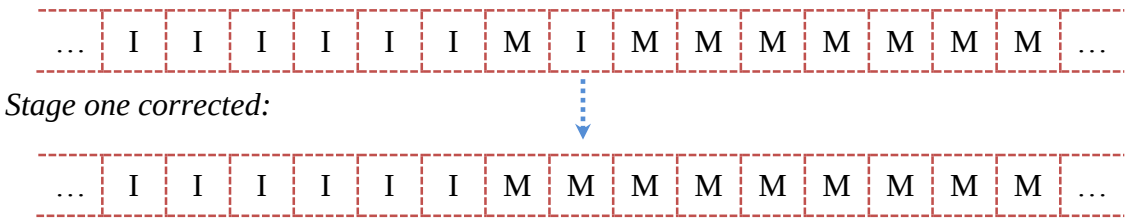


Figure 39: Correction for two consecutive changes-of-state.

The second correction restores previous windows if a short-lived state is discovered. These can occur if a person readjusts footing while standing or takes longer than expected to walk through a door. If *Active Count* or *Inactive Count* is equal to one (new change-of-state) and the previous window's count is less than three, a short-lived state is detected. Figure 40 demonstrates a short immobile action in a continuous mobile state. The algorithm removes the immobile action.

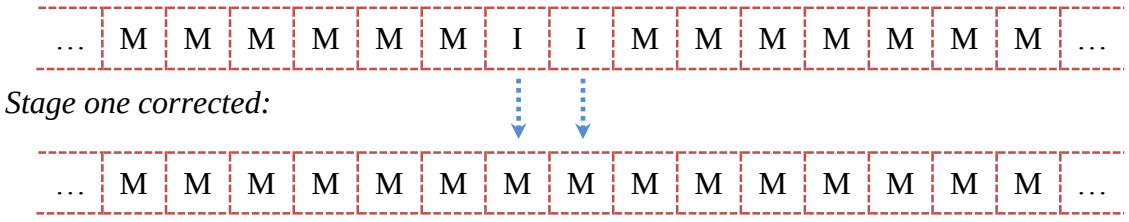


Figure 40: Correction for change-of-state false positives.

These corrections work together in stage one to produce a clean output in real-time. Missed change-of-states may occur if the change is short, (Figure 41); however, video triggers can still be activated. A video trigger starts recording a 3-second video clip from the smartphone, which is used to improve categorization and provide movement context. Video triggers were used in the WMMS1 and WMMS2 algorithms as well, but the WMMS3 analysis separates changes-of-states from video triggers, allowing for the two corrections described above, as well as video clips to contextualize both changes-of-state, and potential changes-of-state.

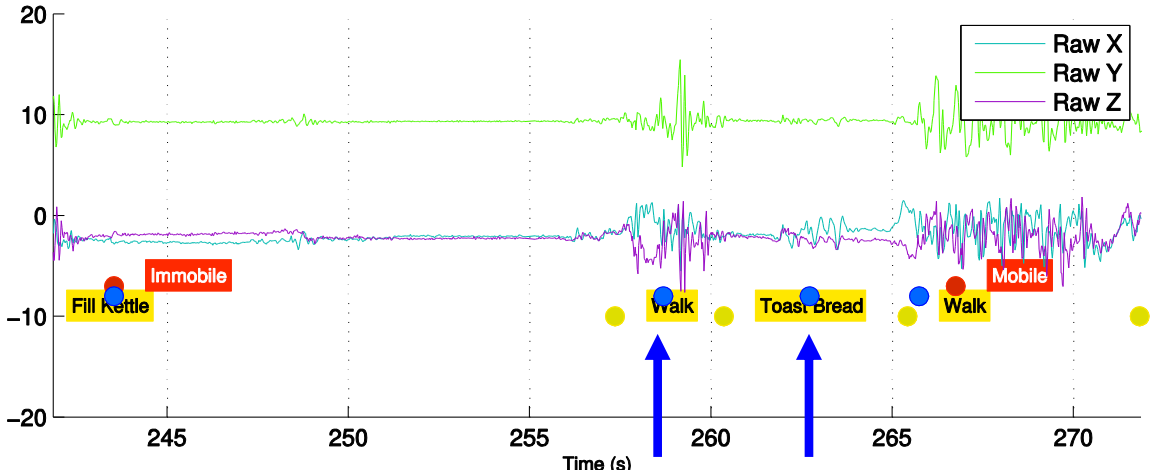


Figure 41: Missed changes-of-state with video triggers.

Video Wait determines whether a video recording should be initiated. If a change-of-state is determined, the *Video Wait* variable equals two. If a correction must be applied, *Video Wait* becomes one, which cancels the trigger to initiate video recording.

Once stage one complete, the expected act values are copied to stage two and the process continues.

6.2.2.2 Stage two

Stage two increases the detail level for categorizing mobility states. Mobility is differentiated between stairs and walking, while immobility states are differentiated between sitting, standing, walking, lying down, and performing small movements. Figure 42 shows the decision tree for stage two. Variables in this stage include:

1) *Small Move Count:* A counter that increases while in a small move state,
2) *Stair Count:* A counter that increases while in a stair state.

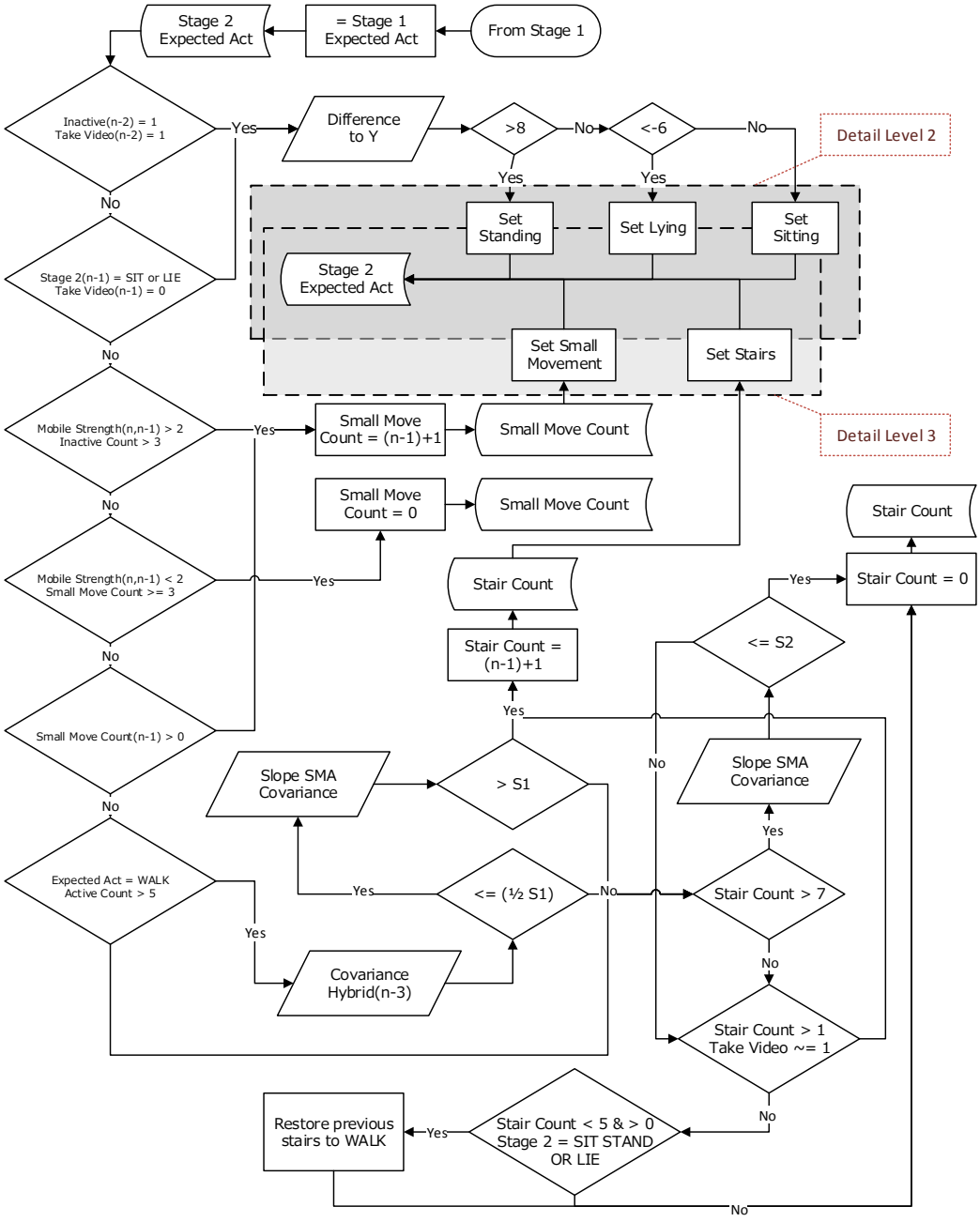


Figure 42: WMMS3 algorithm stage two.

Stage two assesses whether the user is sitting, standing, or lying down, if *Inactive Count* and *Take Video* have been triggered. Standing, sitting and lying down are distinguished

by using the $Difftoy_j$ feature. A $Difftoy_j$ greater than 8 will define standing and a value less than -6 indicates lying down. Values between these thresholds classify sitting. These threshold values are constants across users.

Sit, stand, and lie categorization is based on the phone orientation at the user's pelvis, and the algorithm will not work at other locations on the body. The check for the immobile state to register a sit or lie is through the two previous windows. This ensures that the transition between standing and a new immobile state does not register prematurely and falsify the new state. For example, in Figure 43, the time interval between 650 seconds and 655 seconds is a sit but is corrected to a lie.

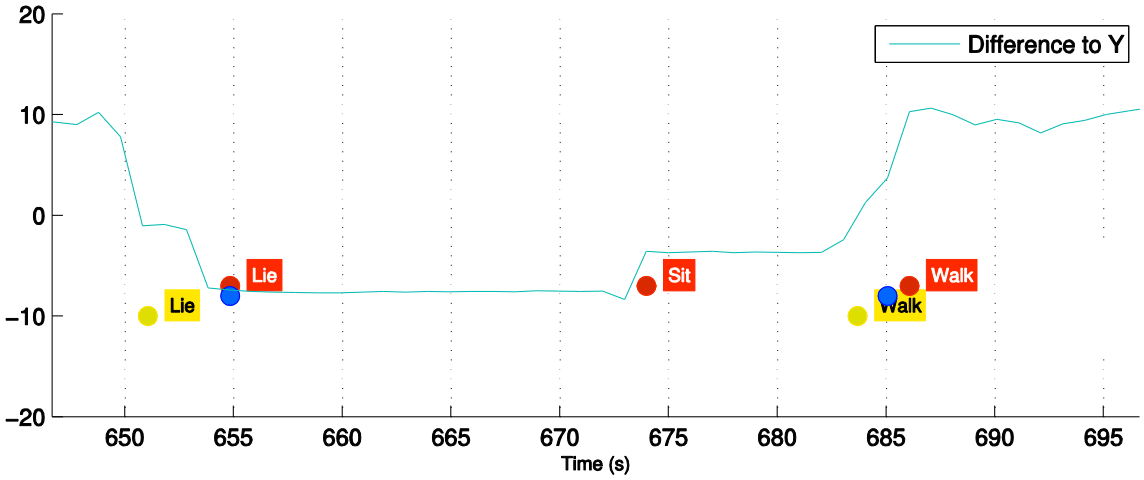


Figure 43: Difference to *y*-axis transition and feature prediction versus gold data.

The immobile state will continue as long as a mobile state is not identified. However, a transition from a lie to a stand often results in an intermediate sit. For detail level two, the sit, stand, and lie position overwrite the immobile state prediction, while the walk overwrites the mobile state prediction.

Detail level three in stage two categorizes small movements while standing; for example, brushing teeth, combing hair, washing hands, etc. The small movement state is triggered if and only if the user is in an immobile standing position. Small movements are a minimum of three windows in length, and will trigger if the *Mobile Strength* is equal to two (a *Mobile Strength* of three results in a full mobile state).

Level three in stage two also includes stair navigation categorization. Stairs are difficult to identify due to high user variability, as well as difficulties with finding correlations between a feature distinction and a stairs activity. After experimental testing, the MAXDIFFcov_j feature was used for defining the stair climb, along with two thresholds chosen to distinguish stairs from walking. These thresholds, named *Stair1 (S1)* and *Stair2 (S2)* were used for detecting the start of a stair climb and the end of a stair climb respectively. In addition, at least five previous walking windows are required before the stair threshold trigger.

Figure 44 and Figure 45 demonstrate relatively different sized 'humps' where the participants climbed stairs. MAXDIFFcov_j is labeled as Slope of SMA.

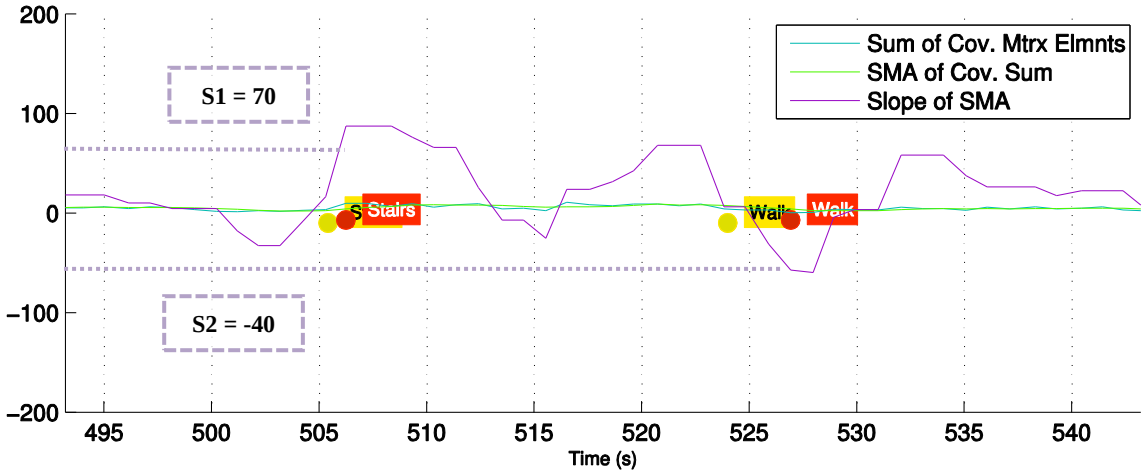


Figure 44: Participant where S1 and S2 are relatively large (-70 and -40).

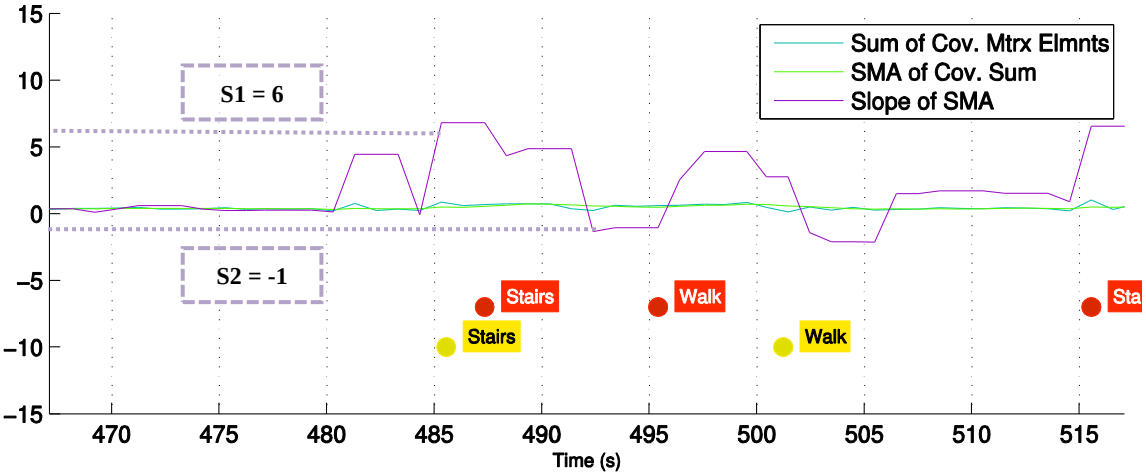


Figure 45: Participants where S1 and S2 are relatively small (6 and -1).

The drop at 515 seconds in Figure 44and 492 seconds in Figure 45 are caused by the presence of a landing that the users walked around between the two flights of stairs. To remove a false change-of-state, the stairs must be present for a minimum of eight windows before a new change-of-state is allowed.

A small correction was included to remove false positives from stair ascent at a transition point from mobile to immobile. The MAXDIFFcov$_j$ feature spikes when transitioning from a walk to a stand, sit or lie. A check for a mobile-to-immobile change during a stairs period is conducted to restore the previous windows from stairs back to walk if an immobile state is detected. For certain participants, this correction was insufficient to prevent a stair activity change-of-state before an immobile state, as demonstrated in Figure 46.

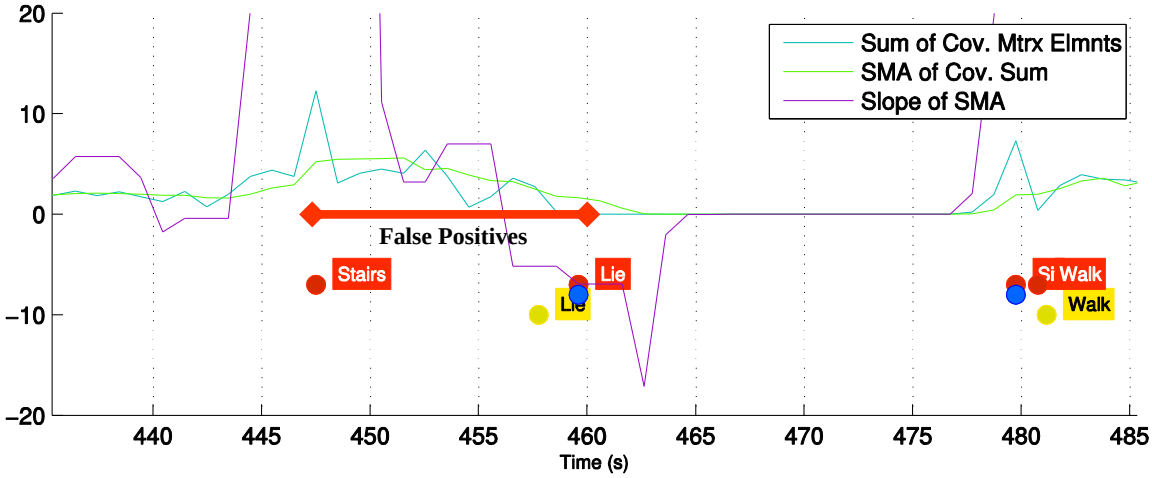


Figure 46: Stairs false-positive before an immobile activity.

6.3 Evaluation

Fifteen able-bodied participants (ten males, five females) were recruited from staff and students at The Ottawa Hospital Rehabilitation Centre (TOHRC) for evaluating the WMMS3 algorithm (Table 22).

All participants provided informed consent and signed a consent form, to be video recorded by an assistant while following a predetermined circuit around TOHRC. The

circuit was the same as the WMMS2 evaluation (Appendix A). All data were collected using the TOHRC Data Logger for BlackBerry 10 application [57] on a BlackBerry Z10 smartphone (model STL100-3, firmware 10.1.0.273). For all trials, the Z10 was positioned in a holster that was attached to the person's waist belt, at the right-front pelvis, with the screen facing outwards.

The objective was to evaluate the WMMS3 through the different prediction detail levels, as well as compare the WMMS3 algorithm results with the WMMS2.

Table 22: WMM3 evaluation participants.

	Average	Standard Deviation
Age (years)	26	8.9
Height (cm)	173.9	11.4
Weight (kg)	68.9	11.1
Waist (cm)	81.0	10.6
Leg height (cm)	90.6	8.8

The device sensor sampling rate was set to 50 Hz, however small fluctuations in data timing caused a standard deviation of ±3 Hz. The phone was shaken at the start of each trial to synchronize the video and time-stamped phone ALA and ADG signals. One trial was collected for each participant.

The WMMS Tool was used for all data processing and to calculate outcome variables including sensitivity, specificity, F_1-Score, true positives, false positives, true negatives, and false negatives. Results were tabulated for all detail levels and activities.

Three analyses were performed at each detail level for each participant, totalling forty-five evaluations and fifteen trials. Analyses were performed with a one-second feature window, three-window change-of-state tolerance, and two-window categorization tolerance. These options were consistent with the WMMS2 evaluation described in Chapter 4.

6.3.1 *Thresholds & fitting*

WMMS3 was developed by visually inspecting features generated by two of the fifteen participants. Features and thresholds used in the WMMS3 algorithm were based on these persons. Since this participant data were also used for evaluation, the results could be artificially better; although, these participants accounted for only13% of the sample size (Table 23).

Table 23: Threshold summary for the WMMS3.

	Threshold
Sum of Range	1
Simple Moving Average	5
Sum of Standard Deviation	1
Difference to Y 1	8
Difference to Y 2	-6
Stairs 1	User Specific
Stairs 2	User Specific

The WMMS3 stair thresholds (S1, S2) were set for each participant by quantitatively assessing the slope of the simple moving average for the sum of variances (MAXDIFFcov_j). Specific stair thresholds for each participant were required due to the high variability in the stair features, as shown in Figure 44 and Figure 45.

6.4 Results

Three detail levels were processed for the fifteen participants. The proceeding tables includes overall sums of true positives, false negatives, true negatives, and false positives for activity categorization (i.e. during standing, during sitting, etc.) and changes-of-state (i.e. stand to sit, stand to walk, etc.).

For detail level one (Table 24, Table 25), categorization sensitivities were 93.8% for mobile and 98.3% for immobile. A change-of-state sensitivity of 87.3% was achieved, with a 9.0% overestimation in the number of changes-of-state detected (443 estimated changes and 403 actual changes for all trials). Most change-of-state false positives occurred during a mobile activity (71 FP and 3898 TN), while most missed changes-of-state occurred from a mobile to immobile transition (160 TP and 34 FN).

Table 24: WMMS3 true positive (TP), false negative (FN), true negative (TN), false positive (FP), estimated and actual count for detail level one.

	TP	FN	TN	FP	Estimated	Actual
Mobile	2998	197	3898	71	4041	4305
Immobile	3898	71	2998	197	5157	4893
Change of State	355	51	8703	89	443	403
During Mobile			3993	70		
During Immobile			4710	19		
Immobile to Mobile	166	17				
Mobile to Immobile	160	34				

Table 25: WMMS3 sensitivity (SE), specificity (SP) and F-Score (F_1) average and standard deviation for detail level one.

	SE Avg. (%)	SP Avg. (%)	F1 Avg. (%)
Mobile	93.8 ± 3.8	98.3 ± 1.0	95.7 ± 2.3
Immobile	98.3 ± 1.0	93.8 ± 3.8	96.7 ± 1.6
Change of State	87.3 ± 7.6	99.0 ± 0.6	88.1 ± 6.9

Overall F_1-scores of 95.7%, 96.7% and 88.1% for mobile, immobile and change-of-state detection were achieved, respectively.

At detail level two (Table 26, Table 27), the immobile state was subdivided into sit, stand and lie. Stand and lie sensitivities were 86.9% and 94.5%, respectively, while sit was at 63.9% with a standard deviation of 39.0%. Walking achieved a 98.3% sensitivity. Four out of the fifteen participants had nearly 0% sensitivity in the sit category. Change-of-state sensitivity was 88.9% with 98.4% specificity. 499 estimated changes-of-state occurred out of a possible 404 actual state changes over all fifteen participants, a difference of 19.0%.

Stand, sit, and lie had 123, 181, and 117 false positives respectively, totalling 421 in immobile, an increase from the 195 in detail level one. This was caused by a correct choice of immobile state, but with an incorrect choice in the type of immobile state.

Table 26: WMMS3 true positive (TP), false negative (FN), true negative (TN), false positive (FP), estimated and actual count for detail level two.

	TP	FN	TN	FP	Estimated	Actual
Stand	1955	294	4790	122	2811	3137
Sit	346	192	6476	147	667	688
Lie	382	23	6639	117	563	480
Walk	3898	71	2998	194	5157	4893
Change of State	359	47	8660	131	489	404
During Stand			2869	74		
During Sit			642	8		
During Lie			449	7		
During Walk			4693	49		
Stand to Walk	135	16				
Walk to Stand	139	24				
Walk to Sit	28	1				
Sit to Walk	30	0				
Walk to Lie	11	4				
Lie to Walk	15	0				

Table 27: WMMS3 sensitivity (SE), specificity (SP) and F-Score (F_1) average and standard deviation for detail level two.

	SE Avg. (%)	SP Avg. (%)	F1 Avg. (%)
Stand	86.9 ± 11.4	97.6 ± 2.5	89.9 ± 8.3
Sit	63.9 ± 39.0	97.8 ± 3.6	66.1 ± 38.2
Lie	94.5 ± 7.2	98.4 ± 3.0	87.6 ± 16.4
Walk	98.3 ± 1.0	93.9 ± 3.7	96.7 ± 1.5
Change of State	88.9 ± 8.1	98.4 ± 0.6	79.9 ± 6.6

Overall F_1-scores of 89.9%, 66.1%, 87.6%, 96.7% and 79.9% for stand, sit, lie, walk, and change-of-state detection were achieved, respectively (Table 27).

Detail level three added climbing stairs and small movements to the classification algorithm (Table 28, Table 29). Small movements scored quite poorly in sensitivity, at 9.2%, with the highest single participant score at 28.6%.

Table 28: WMMS3 true positive (TP), false negative (FN), true negative (TN), false positive (FP), estimated and actual count for detail level three.

	TP	FN	TN	FP	Est.	Act.
Stand	1155	152	4923	694	2582	1716
Sit	346	192	6248	138	667	688
Lie	382	23	6402	117	563	480
Walk	3371	326	3023	204	4640	4636
Stairs	142	40	6483	259	517	257
Small Movement	77	718	6090	39	229	1421
Change of State	423	76	8532	167	587	493
During Stand			1583	26		
During Sit			641	9		
During Lie			450	11		
During Walk			4394	66		
During Stairs			243	3		
During Brush Teeth			190	12		
During Comb Hair			158	3		
During Wash Hands			150	6		
During Dry Hands			119	4		
During Move Dishes			91	6		
During Fill Kettle			191	3		
During Toast Bread			71	1		
During Wash Dishes			251	17		
Stand to Walk	71	6				
Walk to Stand	76	12				
Walk to Sit	26	1				
Sit to Walk	28	0				
Walk to Lie	14	0				
Lie to Walk	14	0				
Walk to Small Move	63	7				
Small Move to Walk	56	13				
Walk to Stairs	12	2				
Stairs to Walk	11	3				
Small Move <>	35	31				

Table 29: WMMS3 sensitivity (SE), specificity (SP) and F-Score (F_1) average and standard deviation for detail level three.

	SE Avg. (%)	SP Avg. (%)	F1 Avg. (%)
Stand	88.6 ± 14.8	87.7 ± 3.2	72.1 ± 11.3
Sit	63.9 ± 39.0	97.9 ± 3.6	66.3 ± 38.3
Lie	94.5 ± 7.2	98.3 ± 3.1	87.6 ± 16.4
Walk	91.2 ± 8.8	93.7 ± 4.2	92.5 ± 4.5
Stairs	78.5 ± 38.4	96.0 ± 5.0	62.3 ± 30.4
Small Movement	9.2 ± 10.5	99.4 ± 1.1	22.5 ± 15.4
Change of State	84.8 ± 5.8	98.1 ± 0.6	77.8 ± 6.1

A confusion matrix was produced for the sum of all categorizations in detail level three (Table 30). Most missed small movements occurred while the participant was standing or walking. Missed stairs occurred while the participant was walking and missed walking occurred while on stairs, as expected.

Table 30: Confusion matrix for detail level three.

		Truth					
		Stand	Sit	Lie	Walk	Stairs	Small Move
Prediction	Stand	1155	58	0	52	0	584
	Sit	86	346	12	14	0	26
	Lie	0	116	382	1	0	0
	Walk	39	10	7	3371	40	108
	Stairs	0	0	4	255	142	0
	Small Move	27	8	0	4	0	77

The WMMS3 results were compared with WMMS2. Table 31 shows greater sensitivity for WMMS3 in every category. The average number of false positives drastically decreased for walking, thereby increasing the specificity from 64.8% to 93.7%. The

number of true positives for walking also increased, resulting in a sensitivity increase from 30.2% to 91.2%. The least changed evaluation is sit, with an increase of 1.3% in sensitivity and a decrease of 0.8% in specificity. The standard deviation between categorizations in sensitivity and specificity decreased.

Table 31: Difference in sensitivity and specificity between WMMS2 and WMMS3.

Sensitivity			
Evaluation	WMMS2 (%)	WMMS3 (%)	Difference (%)
Stand	63.0%	88.6%	25.6%
Sit	62.6%	63.9%	1.3%
Lie	83.6%	94.5%	10.8%
Walk	30.2%	91.2%	61.0%
Stairs	38.2%	78.5%	40.3%
CoS	59.9%	84.8%	24.9%
Average	56.2%	83.6%	27.3%
Std. Dev.	17.6%	10.1%	

Specificity			
Evaluation (%)	WMMS2 (%)	WMMS3 (%)	Difference (%)
Stand	84.5%	87.7%	3.2%
Sit	100.0%	97.9%	2.1%
Lie	99.1%	98.3%	0.8%
Walk	64.8%	93.7%	28.9%
Stairs	91.2%	96.0%	4.9%
CoS	90.5%	98.1%	7.6%
Average	88.3%	95.3%	7.0%
Std. Dev.	11.8%	3.8%	

Table 32: Student's t-test of WMMS2 vs. WMMS3 results p-value.

Student's t-test			
Evaluation	Sensitivity	Specificity	F_1-Score
Stand	0.014	0.065	0.000
Sit	0.164	0.043	0.386
Lie	0.113	0.449	0.868
Walk	0.000	0.000	0.000
Upstairs	0.010	0.032	0.007
Change of State	0.000	0.000	0.000

A Student's t-test was performed for each activity (two-tailed distribution, two-sample heteroscedastic unequal variance, $p<0.05$). Stand, walk, stairs, and change-of-state results were significantly different between WMMS2 and WMMS3, while sit and lie were not

significantly different at $p<0.05$. This is expected since sit and lie are calculated similarly in both systems.

6.5 Discussion

Three WMMS3 detail levels were evaluated on fifteen able-bodied participants. Each detail level involved more predicted activities for the Human Activity Recognizer. Increasing the number of predicted activities decreased overall sensitivity and specificity for activity and change-of-state recognition. The first evaluation (detail level one) included mobility and immobility detection. The second evaluation (detail level two) expanded the previous recognition by detecting standing, sitting, lying down and walking. The final evaluation (detail level three) added small movement detection and stair detection with the previously mentioned activities.

Reduction in sensitivity and specificity was commonly caused by:

1) Device orientation variability between subjects,
2) Unpredictable movement of the device in the hostler throughout the trial,
3) The gold-standard evaluation protocol for small movements,
4) Increasing the amount of prediction categories (detail level),
5) Challenges in the detection of stairs using decision tree logic.

Firstly, detecting device orientation through the $Difftoy_j$ feature to differentiate a sit, stand or lie would flag false positives and false negatives for certain participants depending on their natural sit posture. An upper threshold of 8 for $Difftoy_j$ activates a stand, while below -6 signifies lying down. Between -6 and 8 categorizes the mobility state as sit. This method for immobile categorization did not work for all participants, since device orientation can sometimes issue a lie while the user is sitting (Figure 47).

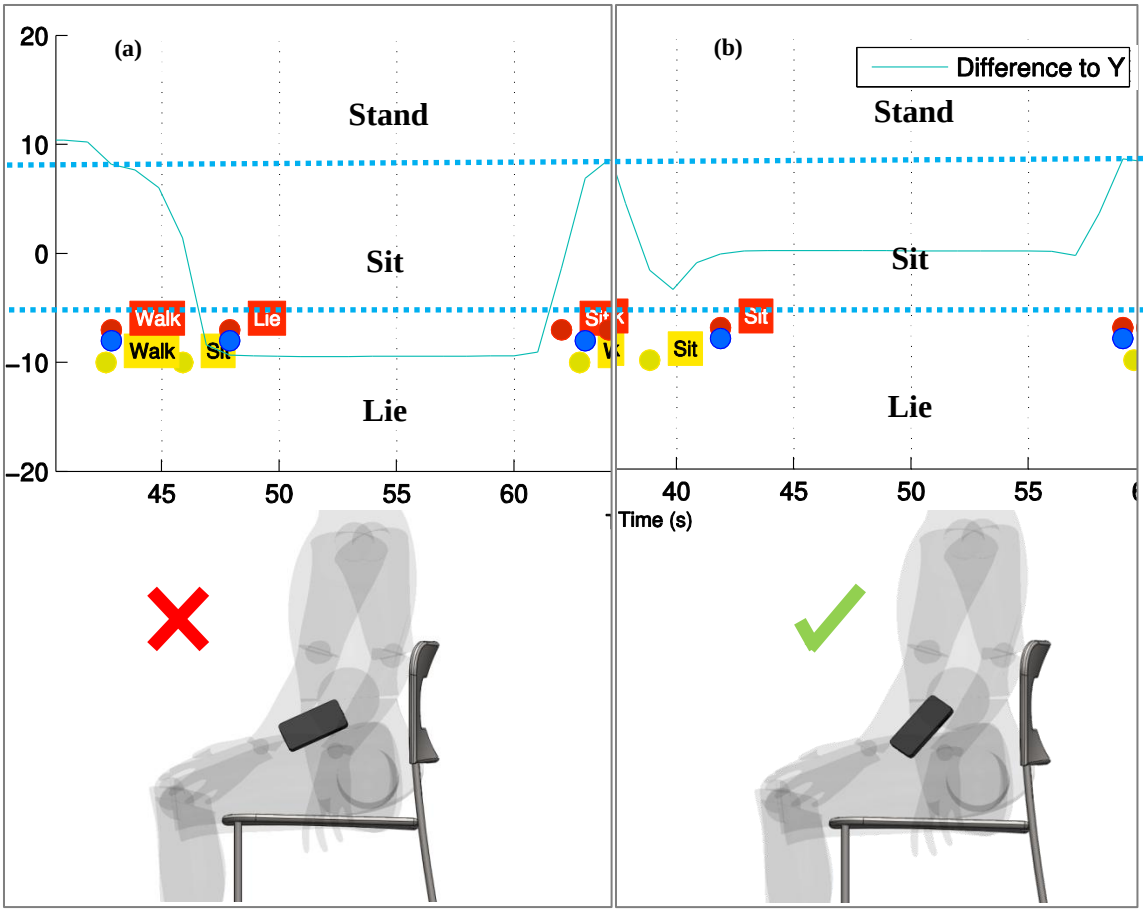


Figure 47: (a) Incorrect device sit orientation (b) Correct device sitting orientation.

Secondly, several participants had different device orientations between the beginning stand and end stand for their trial. Indirect, unintentional pivoting of the holster holding the BlackBerry device was caused by the user's leg, usually after standing from a sitting position. An orientation correction was applied at the beginning of the trial, but was not alleviated if the device shifts from this correction during the trial. The offset results in a skew of the $Difftoy_j$ feature, reducing sensitivity and specificity for sit stand and lie in detail level two and three. An error associated with device orientation caused by holster movement is shown in Figure 48.

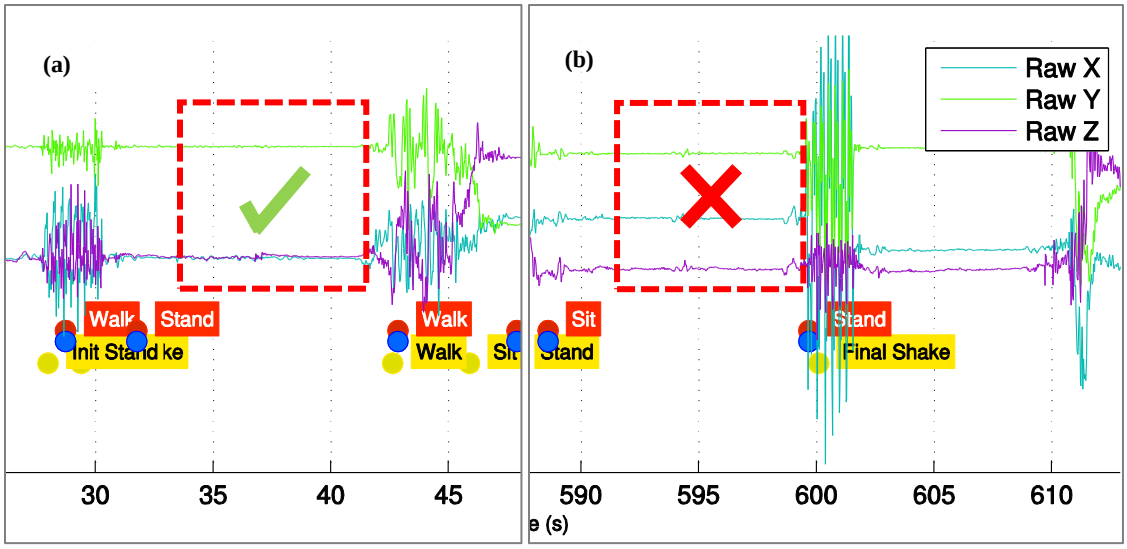


Figure 48: Device orientation difference at the start (a) and end (b) of a participant's trial.

Four out of the fifteen participants tested experienced one or both of the two orientation issues mentioned above. By removing the four subjects with substantial changes in device orientation, a sensitivity of 90.5%, 91.3%, and 86.9% was achieved for stand, sit and lie respectively (Table 33).

Table 33: Sensitivity, specificity, and F_1-score for detail level two with four participants with poor results removed.

	SE (%)	SP (%)	F1 (%)
Stand	90.9	98.1	93.2
Sit	89.9	99.5	90.6
Lie	93.7	100.0	96.5
Walk	98.7	93.5	96.7

If the user ensures that the device is properly upright during standing, the prediction rate for immobile states will increase. The user can also ensure that the holster has not pivoted, which would aid in correcting the orientation while in use. Combining sitting and lying into a universal rest activity would minimize errors but decrease WMMS capability. Thirdly, the average sensitivity for small movements was 9% with a standard deviation of 10%. The main cause of low sensitivity was due to the matching of small

movement actions with a gold set of small movements. If the user performed any of: brush teeth, comb hair, wash hands, dry hands, move dishes, fill kettle, toast bread, or wash dishes, the gold-standard set would say the user is performing a small movement action. Most users do not pivot or perform hip movements while standing and using their hands. Therefore, most of these daily living activities registered as a stand instead of a small movement (Figure 49).

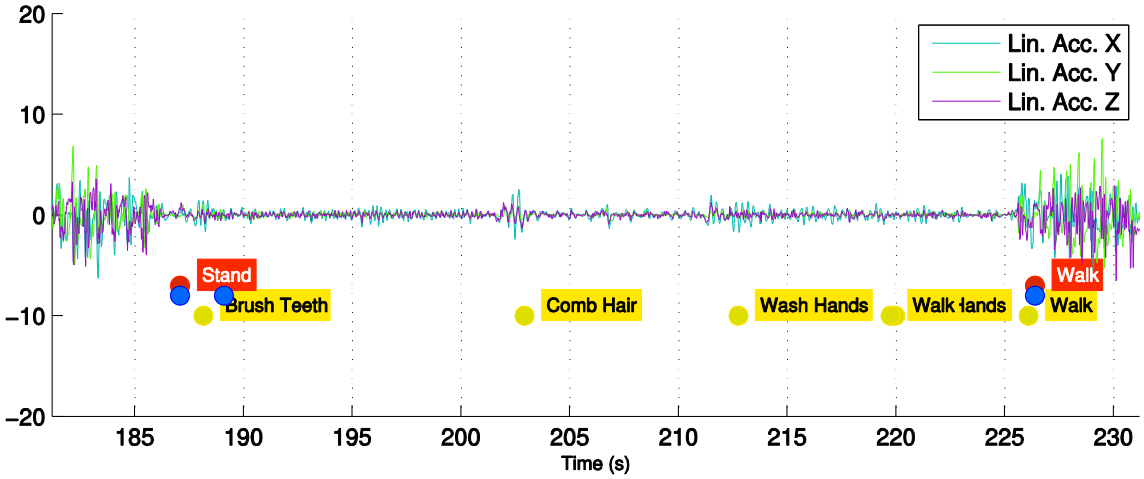


Figure 49: Small movements not registering during bathroom activities.

Certain participants registered a small movement activity partially during these actions (Figure 50).

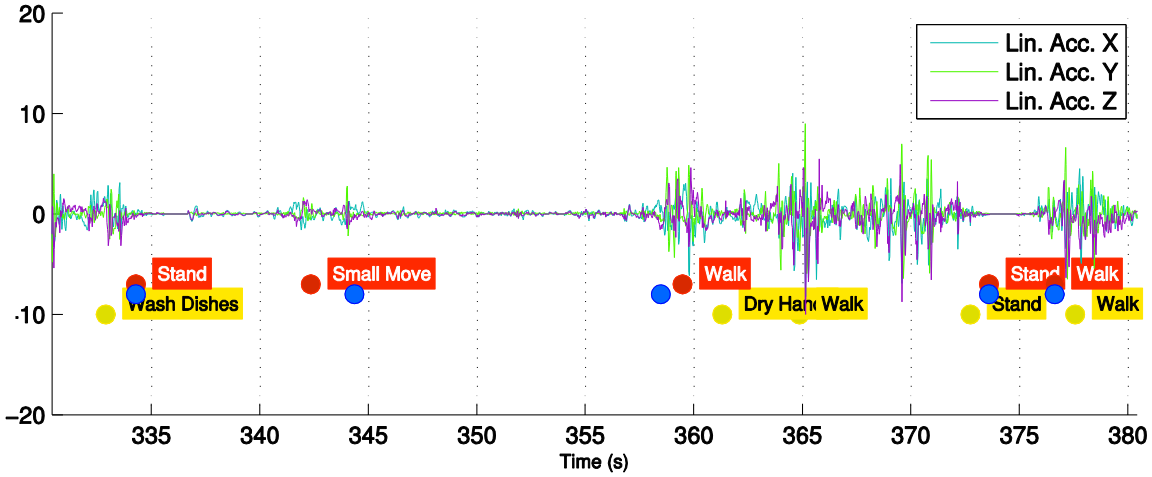


Figure 50: Small movement detection for daily living activities.

An increase in detail level reduced the average sensitivity, specificity and F_1-Score. Increasing the amount of predictable activities increases the chances for false positives and false negatives to occur. An increase in false negatives reduces the score for sensitivity in mobility categorization, while false positives decreases the specificity. An increase in false positives with categorization also produces unwanted change-of-state false positives, reducing the ability to predict mobility changes-of-state.

Lastly, challenges arose when creating a baseline for stair detection using a feature-threshold method. Due to high user variability, participant-specific thresholds were used that corresponded to the participant's variance sum difference values, described in 6.2.1.5. The features used in this thesis are limited in their ability to detect walking and stair climbing.

Overall, the F_1-Scores decreased as detail level increased. Having to choose between more actions reduced the sensitivity of the other action classifications.

Notably, the change-of-state determination increased dramatically in both sensitivity and specificity between the WMMS2 and WMMS3. Minimizing false positives and increasing true positives for changes-of-state reduces the number of inappropriate video clips being captured in the smartphone application, saving memory, battery life, and bandwidth for file transfers.

6.6 Conclusion

A third-generation WMMS algorithm (WMMS3) was developed and evaluated. The IMU gravitational vector (ADG) as well as applied linear acceleration (ALA) were used successfully to distinguish activities such as determining whether the participant was in motion, whether he or she was standing, sitting, or lying as well as detecting a stair climb and small standing movements. Five main features were calculated within single-second time windows.

The WMMS3 was shown to perform better than its predecessor. An average increase of 27% in sensitivity and 7% in specificity on average was found between the WMMS2 and WMMS3 versions.

Increasing the amount of predictable activities decreases the overall efficacy of the software. The average F_1-Score decreased from 93% to 69% while more activities were added to the prediction algorithm.

Chapter 7
CONCLUSION

Smartphone-based Human Activity Recognition is a practical way to monitor mobility in a non-controlled environment. The current computational power, in conjunction with sensitive MEMS sensors, allow for real-time signal manipulation to detect changes in human orientation and in human mobility.

A third-generation WMMS algorithm (WMMS3) has been shown to detect walking, sitting, standing, lying down, climbing stairs and small movements operating on a device is positioned in a holster at the user's hip. The proposed algorithm combines inertial measurement unit sensors, namely the accelerometer, gyroscope and magnetometer, to detect mobility. A BlackBerry Z10 smartphone was used as the hardware for the WMMS3 application.

Fifteen able-bodied participants were recruited to evaluate the algorithm, and the results were compared to the previous generation through a standardized method constructed in the MATLAB environment with the WMMS Tool. A substantial increase in sensitivity for all recognized activities was found between WMMS2 and WMMS3. Change-of-state recognition, where the user is transitioning from one action to another, was also implemented and evaluated.

Overall, the third generation WMMS was shown to perform substantially better than the previous version in both categorization and change-of-state determination. The new WMMS has potential for real-time monitoring and can be used for rehabilitation purposes where mobility monitoring is required.

7.1 Future Work

Future work can include further detail into mobility states, such as differentiating walking from running or jogging, as well as bicycling or exercising, common tasks with rehabilitation procedures. Long-term testing can be useful in the evaluation of battery expenditure; however, a new approach to gold-standard data for evaluation will be

required. Sleep modes, and detecting whether the user has removed the device from the hostler can be implemented, and can aid in battery life.

The use of a more complex classification system, such as fuzzy logic, could be applied to differentiate more difficult activities performed. This can standardize the algorithm to work with all users, without the need for user specific thresholds. Furthermore, detailed daily reports with estimates on caloric consumption and time spent in each activity state may prove useful for medical caregiver.

BIBLIOGRAPHY

[1] O. D. Lara and M. A. Labrador, "A Survey on Human Activity Recognition using Wearable Sensors," *IEEE Commun. Surv. Tutor.*, vol. 15, no. 3, pp. 1192–1209, 2013.

[2] G. Haché, "Development of a wearable mobility monitoring system," University of Ottawa, 2010.

[3] H. Echenberg, *Canada's Aging Population and Public Policy: The Effects on Community Planning*. Library of Parliament, 2012.

[4] S. D. C. Government of Canada, "Canadians in Context - Aging Population / Indicators of Well-being in Canada," 12-Sep-2006. [Online]. Available: http://www4.hrsdc.gc.ca/.3ndic.1t.4r@-eng.jsp?iid=33. [Accessed: 09-Sep-2013].

[5] "Population projections: Canada, the provinces and territories," *Statistics Canada*. [Online]. Available: http://www.statcan.gc.ca/daily-quotidien/100526/dq100526b-eng.htm. [Accessed: 09-Sep-2013].

[6] O. D. Incel, M. Kose, and C. Ersoy, "A Review and Taxonomy of Activity Recognition on Mobile Phones," *BioNanoScience*, vol. 3, no. 2, pp. 145–171, Jun. 2013.

[7] H. Godman, E. Editor, and H. H. Letter, "Two questions can reveal mobility problems in seniors," *Harvard Health Blog*. [Online]. Available: http://www.health.harvard.edu/blog/two-questions-can-reveal-mobility-problems-in-seniors-201309186682. [Accessed: 29-Nov-2013].

[8] F. Foerster, M. Smeja, and J. Fahrenberg, "Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring," *Comput. Hum. Behav.*, vol. 15, no. 5, pp. 571–583, Sep. 1999.

[9] T. F. Elsmore and P. Naitoh, "Monitoring Activity With a Wrist-Worn Actigraph: Effects of Amplifier Passband and threshold Variations," DTIC Document, 1993.

[10] D. F. Kripke, J. B. Webster, D. J. Mullaney, S. Messin, and W. Mason, "Measuring sleep by wrist actigraph," DTIC Document, 1981.

[11] T. L. M. van Kasteren, G. Englebienne, and B. J. A. Kröse, "An activity monitoring system for elderly care using generative and discriminative models," *Pers. Ubiquitous Comput.*, vol. 14, no. 6, pp. 489–498, Sep. 2010.

[12] J. Hernández, R. Cabido, A. S. Montemayor, and J. j. Pantrigo, "Human activity recognition based on kinematic features," *Expert Syst.*, p. n/a–n/a, 2013.

[13] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine Recognition of Human Activities: A Survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1473–1488, 2008.

[14] A. M. Khan, "Human Activity Recognition Using A Single Tri-axial Accelerometer," *Dep. Comput. Eng. Grad. Sch. Kyung Hee Univ. Seoul Korea*, 2011.

[15] H. H. Wu, "Development and Evaluation of a BlackBerry-based Wearable Mobility Monitoring System," University of Ottawa, 2012.

[16] S. L. Lau and K. David, "Movement recognition using the accelerometer in smartphones," in *Future Network and Mobile Summit, 2010*, 2010, pp. 1–9.

[17] J. Yang, H. Lu, Z. Liu, and P. P. Boda, "Physical Activity Recognition with Mobile Phones: Challenges, Methods, and Applications," in *Multimedia Interaction and Intelligent User Interfaces*, L. Shao, C. Shan, J. Luo, and M. Etoh, Eds. Springer London, 2010, pp. 185–213.
[18] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *SIGKDD Explor Newsl*, vol. 12, no. 2, pp. 74–82, Mar. 2011.
[19] P. H. Veltink, H. B. Bussmann, W. de Vries, W. L. Martens, and R. C. Van Lummel, "Detection of static and dynamic activities using uniaxial accelerometers," *IEEE Trans. Rehabil. Eng. Publ. IEEE Eng. Med. Biol. Soc.*, vol. 4, no. 4, pp. 375–385, Dec. 1996.
[20] M. J. Mathie, A. C. F. Coster, D. N. H. Lovell, and B. G. Celler, "Detection of daily physical activities using a triaxial accelerometer," *Med. Biol. Eng. Comput.*, vol. 41, no. 3, pp. 296–301, May 2003.
[21] L. Bao and S. S. Intille, "Activity Recognition from User-Annotated Acceleration Data," in *Pervasive Computing*, A. Ferscha and F. Mattern, Eds. Springer Berlin Heidelberg, 2004, pp. 1–17.
[22] G. M. Lyons, K. M. Culhane, D. Hilton, P. A. Grace, and D. Lyons, "A description of an accelerometer-based mobility monitoring technique," *Med. Eng. Phys.*, vol. 27, no. 6, pp. 497–504, Jul. 2005.
[23] N. Ravi, N. D, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," in *In Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence(IAAI*, 2005, pp. 1541–1546.
[24] L. C. Jatobá, U. Grossmann, C. Kunze, J. Ottenbacher, and W. Stork, "Context-aware mobile health monitoring: evaluation of different pattern recognition methods for classification of physical activity," *Conf. Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. IEEE Eng. Med. Biol. Soc. Conf.*, vol. 2008, pp. 5250–5253, 2008.
[25] J. Pärkkä, L. Cluitmans, and M. Ermes, "Personalization Algorithm for Real-Time Activity Recognition Using PDA, Wireless Motion Bands, and Binary Decision Tree," *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 5, pp. 1211–1215, 2010.
[26] S. Dernbach, B. Das, N. C. Krishnan, B. L. Thomas, and D. J. Cook, "Simple and Complex Activity Recognition through Smart Phones," in *2012 8th International Conference on Intelligent Environments (IE)*, 2012, pp. 214–221.
[27] L. Sun, D. Zhang, B. Li, B. Guo, and S. Li, "Activity Recognition on an Accelerometer Embedded Mobile Phone with Varying Positions and Orientations," in *Ubiquitous Intelligence and Computing*, Z. Yu, R. Liscano, G. Chen, D. Zhang, and X. Zhou, Eds. Springer Berlin Heidelberg, 2010, pp. 548–562.
[28] V. Q. Viet, H. M. Thang, and D. Choi, "Personalization in Mobile Activity Recognition System using K-medoids Clustering Algorithm."
[29] D. John, S. Liu, J. E. Sasaki, C. A. Howe, J. Staudenmayer, R. X. Gao, and P. S. Freedson, "Calibrating a novel multi-sensor physical activity measurement system," *Physiol. Meas.*, vol. 32, no. 9, pp. 1473–1489, Sep. 2011.
[30] Z. He and L. JIN, "Activity recognition from acceleration data based on discrete consine transform and SVM," in *IEEE International Conference on Systems, Man and Cybernetics, 2009. SMC 2009*, 2009, pp. 5041–5044.

[31] T. Brezmes, J.-L. Gorricho, and J. Cotrina, “Activity Recognition from Accelerometer Data on a Mobile Phone,” in *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, S. Omatu, M. P. Rocha, J. Bravo, F. Fernández, E. Corchado, A. Bustillo, and J. M. Corchado, Eds. Springer Berlin Heidelberg, 2009, pp. 796–799.

[32] P. Siirtola and J. Röning, “User-Independent Human Activity Recognition Using a Mobile Phone: Offline Recognition vs. Real-Time on Device Recognition,” in *Distributed Computing and Artificial Intelligence*, S. Omatu, J. F. D. P. Santana, S. R. González, J. M. Molina, A. M. Bernardos, and J. M. C. Rodríguez, Eds. Springer Berlin Heidelberg, 2012, pp. 617–627.

[33] M. Berchtold, M. Budde, D. Gordon, H. R. Schmidtke, and M. Beigl, “ActiServ: Activity Recognition Service for mobile phones,” in *2010 International Symposium on Wearable Computers (ISWC)*, 2010, pp. 1–8.

[34] M. Fahim, I. Fatima, S. Lee, and Y.-T. Park, “EFM: evolutionary fuzzy model for dynamic activities recognition using a smartphone accelerometer,” *Appl. Intell.*, pp. 1–14.

[35] M. Helmi and S. M. T. AlModarresi, “Human activity recognition using a fuzzy inference system,” in *IEEE International Conference on Fuzzy Systems, 2009. FUZZ-IEEE 2009*, 2009, pp. 1897–1902.

[36] S.-Y. Chiang, Y.-C. Kan, Y.-C. Tu, and H.-C. Lin, “Activity Recognition by Fuzzy Logic System in Wireless Sensor Network for Physical Therapy,” in *Intelligent Decision Technologies*, J. Watada, T. Watanabe, G. Phillips-Wren, R. J. Howlett, and L. C. Jain, Eds. Springer Berlin Heidelberg, 2012, pp. 191–200.

[37] Y. Zhang, H. Li, S. Chen, and L. Ma, “Recognition Approach of Human Motion with Micro-accelerometer Based on PCA-BP Neural Network Algorithm,” in *Advances in Neural Networks – ISNN 2013*, C. Guo, Z.-G. Hou, and Z. Zeng, Eds. Springer Berlin Heidelberg, 2013, pp. 635–641.

[38] K. Kiani, C. J. Snijders, and E. S. Gelsema, “Computerized analysis of daily life motor activity for ambulatory monitoring,” *Technol. Health Care Off. J. Eur. Soc. Eng. Med.*, vol. 5, no. 4, pp. 307–318, Oct. 1997.

[39] M.-W. Lee, A. M. Khan, and T.-S. Kim, “A single tri-axial accelerometer-based real-time personal life log system capable of human activity recognition and exercise information generation,” *Pers. Ubiquitous Comput.*, vol. 15, no. 8, pp. 887–898, Dec. 2011.

[40] J. Howcroft, J. Kofman, and E. D. Lemaire, “Review of fall risk assessment in geriatric populations using inertial sensors,” *J. Neuroengineering Rehabil.*, vol. 10, no. 1, p. 91, 2013.

[41] H. J. Luinge, *Inertial sensing of human movement*. Twente University Press, 2002.

[42] Y.-J. Hong, I.-J. Kim, S. C. Ahn, and H.-G. Kim, “Mobile health monitoring system based on activity recognition using accelerometer,” *Simul. Model. Pract. Theory*, vol. 18, no. 4, pp. 446–455, 2010.

[43] J. W. Lockhart, T. Pulickal, and G. M. Weiss, “Applications of mobile activity recognition,” in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, New York, NY, USA, 2012, pp. 1054–1058.

[44] Y. He and Y. Li, “Physical Activity Recognition Utilizing the Built-In Kinematic Sensors of a Smartphone,” *Int. J. Distrib. Sens. Netw.*, vol. 2013, Apr. 2013.

[45] “Experimenting with Google Play Services - Activity Recognition,” *FP7 Carmesh: The Blog*. [Online]. Available: http://blog.carmesh.eu/2013/05/31/experimenting-with-google-play-services-activity-recognition/. [Accessed: 23-Aug-2013].
[46] “Apple - Apple Events - Special Event September 2013.” [Online]. Available: http://www.apple.com/apple-events/september-2013/. [Accessed: 12-Sep-2013].
[47] G. Hache, E. D. Lemaire, and N. Baddour, “Mobility change-of-state detection using a smartphone-based approach,” in *2010 IEEE International Workshop on Medical Measurements and Applications Proceedings (MeMeA)*, 2010, pp. 43–46.
[48] G. Hache, E. D. Lemaire, and N. Baddour, “Wearable Mobility Monitoring Using a Multimedia Smartphone Platform,” *IEEE Trans. Instrum. Meas.*, vol. 60, no. 9, pp. 3153–3161, 2011.
[49] H. H. Wu, E. D. Lemaire, and N. Baddour, “Change-of-state determination to recognize mobility activities using a BlackBerry smartphone,” in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society,EMBC*, 2011, pp. 5252–5255.
[50] H. Wu, E. D. Lemaire, and N. Baddour, “Activity Change-of-state Identification Using a Blackberry Smartphone,” *J. Med. Biol. Eng.*, vol. 32, no. 4, pp. 265–271, 2012.
[51] M. D. Tundo, E. Lemaire, and N. Baddour, “Correcting Smartphone orientation for accelerometer-based analysis,” in *2013 IEEE International Symposium on Medical Measurements and Applications Proceedings (MeMeA)*, 2013, pp. 58–62.
[52] J. B. Kuipers, *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*. Princeton, N.J.: Princeton University Press, 1999.
[53] W. Durham, *Aircraft Flight Dynamics and Control*. 2013.
[54] A. Lerios, *Rotations and Quaternions*. .
[55] J. Bures, R. Láviicka, and V. Soucek, *Elements of Quaternionic Analysis and Radon Transform*. Departamento de Matemática da Universidade de Coimbra, 2009.
[56] J. M. Cooke, “NPSNET: Flight simulation dynamic modeling using quaternions,” Monterey, California. Naval Postgraduate School, 1992.
[57] “BlackBerry World - TOHRC Data Logger for BlackBerry 10.” [Online]. Available: http://appworld.blackberry.com/webstore/content/32013891/?countrycode=CA.
[58] S. T. Smith, *MATLAB: Advanced GUI Development*. Dog Ear Publishing, 2006.
[59] “BlackBerry World - TOHRC Data Logger.” [Online]. Available: http://appworld.blackberry.com/webstore/content/60743/?lang=en&countrycode=CA. [Accessed: 12-Oct-2013].
[60] “export_fig - File Exchange - MATLAB Central.” [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/file_infos/23629-exportfig. [Accessed: 12-Oct-2013].
[61] “Sensitivity and specificity,” *Wikipedia, the free encyclopedia*. 13-Dec-2013.
[62] “BlackBerry Devices - Smartphones & BlackBerry PlayBook Devices - Canada.” [Online]. Available: http://ca.blackberry.com/devices.html.
[63] “Supported sensors - BlackBerry Native.” [Online]. Available: http://developer.blackberry.com/native/documentation/core/supported_sensors.html. [Accessed: 21-Oct-2013].

[64] S. O. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” *Rep. X-Io Univ. Bristol UK*, 2010.

APPENDIX A: EXPERIMENTAL PROCEDURE

The procedure for the BlackBerry 10 and BlackBerry 7 studies in this paper, performed at the Ottawa Hospital Rehabilitation Centre.

1. Turn on the recognizer, and place in hostler on hip.
2. From a standing position, shake the smartphone.
3. Continue standing, then walk close by chair and take a seat.
4. Stand up and walk 60 meters to the elevator.
5. Stand and wait for the elevator and then walk into the elevator.
6. Turn around and then take the elevator to the second floor.
7. Turn left and walk into the independent living unit.
8. Walk into the bathroom, and commence brushing teeth in front of the mirror.
9. Continue standing and start combing hair.
10. Start washing hands.
11. Dry hands using the paper towels next to the sink.
12. Walk into the kitchen.
13. Take dishes from the rack, and place them on the counter.
14. Grab the kettle, and fill it with water from the kitchen sink.
15. Place the kettle on the element.
16. Place the bread in the toaster.
17. Walk a few meters into the dining room.
18. Take a seat at the dining room table.
19. Mimic eating a meal at the table.
20. Stand, and walk back to the kitchen sink.
21. Rinse off the dishes and place them in the rack.
22. Walk from the kitchen back to the elevator.
23. Stand and wait for the elevator and then walk into the elevator.
24. Take the elevator to the first floor.
25. Walk 50 meters to the stairwell.
26. Open the door and enter the stairwell.
27. Walk up stairs (13 steps, around landing, 13 steps).
28. Open the stairwell door into the hallway.
29. Turn right and walk down the hall for 15 meters.
30. Turn around and walk 15 meters back to the stairwell.
31. Open the door and enter the stairwell.
32. Walk down stairs (13 steps, around landing, 13 steps).
33. Exit the stairwell and walk forward into the laboratory.
34. Lie on the bed.
35. Get back up and walk 10 meters to the ramp
36. Walk up the ramp, then down the ramp (20 meters)
37. Continue walking into the hall and open the door to outside.
38. Walk 100 meters on the paved pathway.
39. Turn around and walk back to the laboratory.
40. Walk into the laboratory and stand at the starting point.
41. Continue standing, and shake the smartphone.

APPENDIX B: WMMS TOOL GOLD FILE

Below is a sample of a video evaluation (gold-standard) file used in the *WMMS Tool*. An evaluator would video tape the trial, and enter the change-of-state times manually in this file while watching the video file on a PC. Variables *Shake, S1* and *S2* are written after a subject's initial evaluation, and once entered into the gold file, the *WMMS Tool* will auto-load those values into the GUI to save time and maintain constancy.

Video Evaluation for WMMS Tool

Start of STATE		Time (s)	State Start
Initial Shake		13.27	4
Stand		15.78	1
Walk		36.43	4
Sit		40.23	2
Walk		55.2	4
Stand		95.33	1
Walk		133.97	4
Stand	(Elev)	136.67	1
Walk		169.67	4
Brush Teeth		183.43	21
Comb Hair		198.2	22
Wash Hands		208.03	23
Dry Hands		215.23	24
Walk		221.37	4
Move Dishes		227	25
Walk		232.63	4
Fill Kettle		237.63	26
Walk		253.63	4
Toast Bread		256.63	27
Walk		261.7	4
Sit		268.1	2
Walk		286.17	4
Wash Dishes		292.3	28
Dry Hands		312.87	24
Walk		215.03	4
Stand		321.8	1
Walk		345.23	4
Stand	(Elev)	347.6	1
Walk		373.27	4
Stairs		408.5	5
Walk		426.63	4
Walk	(Down)	455.57	5
Walk		467.83	4
Lie		486.3	3
Walk		504.7	4
Walk	(Ramp)	514.7	5
Walk		528.53	4
Stand		605.07	1
Final Shake		619.07	4

State	Gold State
Stand	1
Sit	2
Lie	3
Walk	4
Stairs	5
Small Move	6

Variables	
Shake	29
S1	20
S2	5

Small Movements

Brush Teeth	21
Comb Hair	22
Wash Hands	23
Dry Hands	24
Move Dishes	25
Fill Kettle	26
Toast Bread	27
Wash Dishes	28

Instructions:

Please fill in the 'Time (s)' column using the camcorder footage.
The time should be when the user has **STARTED** the new task.

APPENDIX C: WMMS TOOL VARIABLES

Array	Description	Columns		
BlackBerry 7 & WMMS2				
sub_rawdata (4)	Retrieved from input file.	(1) Raw Time (2) X Acc	(3) Y Acc (4) Z Acc	
sub_features (21)	Features generated through feature window construction and calculation.	(1) Feature Time (2) Max X (3) Max Y (4) Max Z (5) Min X (6) Min Y (7) Min Z	(8) Range X (9) Range Y (10) Range Z (11) Range XZ (12) Avg X (13) Avg Y (14) Avg Z	(15) StD X (16) StD Y (17) StD Z (18) Tilt ZY (19) Sum Range (20) Diff SR (21) SMA-SR
sub_states (15)	WMMS2 algorithm based on features generated.	(1) Feature Time (2) LyorSt (3) StorDy (4) MoveStatic (5) Downstair	(6) Upstair (7) Ramp (8) Elevator (10) Change of State 1 (11) Change of State 2	(12) Change of State 3 (13) ExpectedAct (14) TakeVideo (15) VideoCount

Array	Description	Columns		
BlackBerry 10 & WMMS3				
sub_rawdata (16)	Retrieved from input file.	(1) Raw Time (2) Acc X (3) Acc Y (4) Acc Z (5) Gravity X (6) Gravity Y	(7) Gravity Z (8) World X (9) World Y (10) World Z (11) Gyro X	(12) Gyro Y (13) Gyro Z (14) LinX (15) LinY (16) LinZ
sub_features (23+)	Features generated through feature window construction and calculation.	(1) Feature Time (2) Sum of Range (3) SumofSTD (4) Simple Moving Avg (5) Diff to Y (6) Range X (grav) (7) Range Y (grav) (8) RangeZ (grav)	(9) Mean X (grav) (10) Mean Y (grav) (11) Mean Z (grav) (12) Range X (lin) (13) Range Y (lin) (14) Range Z (lin) (15) Mean X (lin) (16) Mean Y (lin)	(17) Mean Z (lin) (18) Kurtosis X (lin) (19) Kurtosis Y (lin) (20) Kurtosis Z (lin) (21) Kurtosis SUM (lin) (22) StD Sum Sq (lin) (23+) Testing
sub_states (11)	WMMS3 algorithm based on features generated.	(1) Feature Time (2) Stage1 (3) Stage1Raw (4) MobileStrength	(5) ActiveCount (6) InactiveCount (7) VideoWait (8) Stage1Video	(9) Stage2 (10) SmallMoveCount (11) StairCount

APPENDIX D: WMMS TOOL RESULTS FILE

Results of WMMS Tool v0.9.9:										
Raw Data: *bb10_sample_sub2.txt*										
Gold Data: *gold_sub2_new.xlsx*										
Options			*Value*		*Info*			*Value*	*Stn.Dv.*	
Feature Window (s):			1		Avg window time (s):			1.00819	0.002183	
End of Shake:			17		Avg samples per window:			45.88571	1.088874	
Tolerance COS (windows):			2		Feature windows/second:			0.99094		
Tolerance CAT (windows):			2		Feature windows:			630		
Sliding Window (%):			0		Time (s):			635.76		
Rotation Matrix:			OFF		Total # of videos:			28		
Detail Level:			3		Total # of changes of state:			42		
WMMS Version:			3		Total gold changes of state:			36		
		TP	*FN*	*TN*	*FP*	*Est Total*	*Act Total*	*Sensitivity*	*Specificity*	*F1-Score*
Stand		86	38	297	10	138	156	0.693548	0.967427	0.781818
Sit		21	0	410	0	34	31	1	1	1
Lie		13	0	416	2	24	18	1	0.995215	0.928571
Walk		190	9	214	18	274	268	0.954774	0.922414	0.933661
Stairs		12	17	402	0	15	44	0.413793	1	0.585366
Small Movement		42	3	349	37	115	83	0.933333	0.904145	0.677419
Change of State		23	20	256	17	40	36	0.534884	0.937729	0.554217
During Stand				84	4					
During Sit				29	0					
During Lie				0	0					
During Walk				120	9					
During Stairs				1	2					
During Brush Teeth				12	0					
During Comb Hair				0	0					
During Wash Hands				0	0					
During Dry Hands				0	0					
During Move Dishes				3	0					
During Fill Kettle				3	1					
During Toast Bread				1	0					
During Wash Dishes				3	1					
Stand to Walk		3	2							
Walk to Stand		5	3							
Walk to Sit		1	0							
Sit to Walk		2	1							
Walk to Lie		1	1							
Lie to Walk		0	1							
Walk to Small Move		0	0							
Small Move to Walk		0	0							
Stand to Small Move		0	0							
Small Move to Stand		0	0							
Walk to Stairs		0	2							
Stairs to Walk		0	2							
Other		11	8							