A REAL-TIME HUMAN POSTURE CLASSIFIER AND FALL-DETECTOR

by

CHIA-HUA LIN

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

CASE WESTERN RESERVE UNIVERSITY

August, 2014
CASE WESTERN RESERVE UNIVERSITY
SCHOOL OF GRADUATE STUDIES

We hereby approve the thesis/dissertation of

CHIA-HUA LIN

candidate for the degree of Doctor of Philosophy*

Committee Chair
Mehran Mehregany

Committee Member
M. Cenk Cavusoglu

Committee Member
Francis Merat

Committee Member
Elizabeth Madigan

Date of Defense
May 16/2014

*We also certify that written approval has been obtained for any proprietary material contained therein.
Table of Contents

Chapter 1  Introduction .................................................................................................. 1

1.1 Motivation and Objectives .................................................................................... 1

1.2 Literature Review on Human Activity Recognition .............................................. 6

1.2.1 Assessment Techniques................................................................................... 6

1.2.2 Wearable Sensors ........................................................................................... 7

1.2.3 Activity Recognition Using Wearable Sensors ................................................ 11

1.2.4 Prior Works in Our Group ............................................................................. 14

Chapter 2  Proposed Solution and Research Method .............................................. 18

2.1 Proposed Solution for Human Activity Classification ........................................ 18

2.2 Inertial Sensor and Fundamental Study ............................................................... 21

2.2.1 VN-100 Inertial Measurement Unit and Attitude Heading Reference System 21

2.2.2 Coordinate Alignment/Transformation .......................................................... 23

2.2.3 Vertical Displacement Estimation ................................................................. 26

2.3 Research Method ................................................................................................ 27

Chapter 3  Classification Algorithm Verification Using MATLAB ................................ 29

3.1 System Introduction ............................................................................................. 29

3.2 Activity Level Estimation ................................................................................... 32

3.3 Offset Control in Vertical Displacement Estimation ......................................... 37
### List of Tables

Table 1.1 Percent U.S. population over age 65 by year, 2000-2050. This table is from "Remote and wireless patient monitoring systems" [2], and the original source is U.S. Census Bureau[1].

Table 3.1 Sample numbers of the activity level experiment. The data points were sampled randomly during the time of each specific activity/posture.

Table 3.2 All pair-wise comparisons (Fisher’s LSD) among different offset estimation methods. The difference is only considered significant when P < 0.05.

Table 3.3 Fuzzy rules

Table 3.4 Follow-up rules

Table 3.5 Experiment results of posture sequences on subject 0

Table 3.6 Experiment results of posture sequences on 5 testing subjects

Table 3.7 Error analysis over 60 sampled sequences on subject 0

Table 3.8 Error analysis over 300 sample sequences on 5 testing subjects

Table 3.9 Results of experiments on subject 0 with DOR activated, expressed in individual postures

Table 3.10 Results of experiments on subject 0 with DOR disabled, expressed in individual postures

Table 3.11 Results of experiments on five testing subjects with DOR activated, expressed in individual postures

Table 3.12 Results of trail experiments on five subjects with DOR disabled, expressed in individual postures
Table 4.1 The Estimated Power Consumed by Each Major Components in the Portable Device................................................................................................................................ 61

Table 4.2 The activity sequences designed for the testing on Case students. The duration of each lasting activity is 10 seconds. ........................................................................................................ 72

Table 4.3 Analytical results of One-Way ANOVA on accuracy grouped by height or weight ............................................................................................................................................... 82

Table 4.4 Averaged sensitivity and specificity of fall detection................................................................................................................................. 83

Table 5.1 Reported Parameters from the Sensors Used in the Testing on elderly .......... 87

Table 5.2 Activity conversion table for human posture classifier performance evaluation on elderly ........................................................................................................................................ 92

Table 5.3 List of correspondences between the activity log and classification results using different algorithms........................................................................................................................................ 94

Table 5.4 Dependences between different attribute groups and correspondences using different classification algorithms ........................................................................................................................................ 94

Table 5.5 One-Way ANOVA comparing the results of different fall risk groups ............ 95
List of Figures

Figure 1.1 The U.S. population from 1980 to 2050. This plot is from "Remote and wireless patient monitoring systems" [2], and the original source is U.S. Census Bureau[1]. ........................................................................................................................... 1

Figure 1.2 Life expectancy at birth: United States, 1940, 1950, 1960, 1970, and 1975-2008[3]. ............................................................................................................................... 2

Figure 1.3 The birth rate in the US, 1980-2010. The sources are Department of Health and Human Services, National Center for Health Statistics[5]................................. 3

Figure 1.4 Number of Hospitals in the US, 1975-2005, Estimated 2015. This plot is from "Remote and wireless patient monitoring systems"[2], and the original source is American Hospital Association[6]....................................................................................................................... 4

Figure 1.5 The system architecture of the preceding posture classifier. ......................... 14

Figure 1.6 The algorithm process flow of the preceding classifier ................................ 15

Figure 1.7 Architecture of the prior fuzzy logic classifier ................................................. 16

Figure 2.1 Data flow of the proposed human posture classifier ........................................ 20

Figure 2.2 Block diagram of VN-100 ............................................................................. 22

Figure 2.3 The user interface of Vectornav Sensor Explorer ............................................ 23

Figure 3.1 Data flow of the MATLAB-based human posture classifier ............................. 29

Figure 3.2 The process flow of the posture classification algorithms ............................ 31

Figure 3.3 Performed sequences of movement. The timeline is in unit of "seconds." .... 34

Figure 3.4 Results of the experiment to compare SMA and maximum SVM .............. 36
Figure 3.5 VD estimation in an offline analysis: (a) double integration without any d.c. offset technique applied; (b) d.c. offset is removed by subtracting the overall mean value from vertical acceleration (only valid when the initial and final accelerations are the same); (c) d.c. offset is removed by subtracting overall mean value from vertical acceleration and conditional integration technique is applied. VD in this figure is double-integrated from the VA of a subject performing sitting and standing in 40 seconds. ..... 38

Figure 3.6 Adopted methods for offset estimation; the acceleration or velocity in the next window is adjusted by subtracting the offset estimated using the data in the current window if the classified posture is static posture or locomotion. The offset estimation methods include: 1) mean of VA, 2) least-square of VA, and 3) linear estimation of VV. ................................................................. 41

Figure 3.7 Experiment result comparison of the three proposed offset estimation methods and reference group ................................................................. 42

Figure 3.8 Membership functions of the fuzzy logic classifier ........................................ 45

Figure 3.9 A picture of the hardware used for the testing and the defined orientation. The orientation of the sensor is defined as the angle between the fixed inertial frame z-axis toward the ground and the body frame x-axis. The $X_b$ values marked in red color are the boundaries of membership functions. ................................................................. 46

Figure 3.10 Expected sequences of movement ................................................................. 50

Figure 4.1 A schematic showing how the portable human activity classification system works ........................................................................................................ 62

Figure 4.2 The exterior view and dimensions of the portable human activity classifier . 64
Figure 4.3 Algorithm 1 for human activity classification: a decision tree structure using only pitch and GCSVM$^2$ as inputs. T1 is set to 3, and T2 is set to 80 in the testing. These values are decided according to the data collected from subject 0. 68

Figure 4.4 Algorithm 2 for human activity classification: a decision tree structure using pitch, GCSVM$^2$, and VD as inputs. T1 is set to 3, and T2 is set to 80 in the testing. These values are decided according to the data collected from subject 0. 70

Figure 4.5 The arrangement of the testing environment used in the testing on Case students (not in scale). 72

Figure 4.6 A sample of output data sequence 1 from subject 1: a) result of classification algorithm 1, b) result of classification algorithm 2, c) the reference sequence. 73

Figure 4.7 Testing result in averaged accuracy analyzed using analytical method 1 and compared by subject. 77

Figure 4.8 Testing result in averaged accuracy analyzed using analytical method 2 and compared by subject. 77

Figure 4.9 Testing result in averaged accuracy analyzed using analytical method 1 and compared by sequence. 79

Figure 4.10 Testing result in averaged accuracy analyzed using analytical method 2 and compared by sequence. 79

Figure 4.11 Testing result of the fall detection function compared by sequence. 81

Figure 5.1 Data flow of the final wearable human activity classifier - the bottom-lined parameters refer to the average value in the corresponding loop. 86
Figure 5.2 A picture of the devices benchmarked in this testing - from left bottom to right bottom: Fitbit Flex, Jawbone Up, and Nike+ Fuelband ..................................................... 88

Figure 5.3 Step counts from the testing on elderly .......................................................... 90

Figure 5.4 Burned calories from the testing on elderly .................................................... 90

Figure 5.5 Estimated distances (km) from the testing on elderly ................................. 91

Figure 5.6 Active time (minutes) from the testing on elderly. The value of algorithm 1 on subject number 6 is 815 minutes and truncated in this plot. This value is obviously misestimated. ................................................................................................................... 91
A Real-Time Human Posture Classifier and Fall-Detector

Abstract

by

CHIA-HUA LIN

There is a growing demand on human activity recognition and remote wellness monitoring in modern societies. Related algorithms and hardware platforms have been intensively researched during the past decade. This work presents a wearable system for real-time human activity classification and fall detection. As the major part of this system, several algorithms are designed for feature extraction, vertical displacement estimation, and posture classification, and implemented in a customized wearable embedded platform. This battery-powered device has been tested on both undergraduate/graduate students and the elderly. The automatically logged activity reports in the testing are compared with video clips, manually logged activity, and the report from three different commercially available activity trackers. The testing results show up to 89.96% of second-by-second accuracy on daily activities, 75.2% fall detection sensitivity, and 99.85% fall detection specificity.
Chapter 1 Introduction

1.1 Motivation and Objectives

According to the data provided by U.S. Census Bureau, the U.S. population has shown steady growth since the year 1980 (0.8% - 1.2% annually) and is expected to reach 341 million by the year 2020 (Fig. 1.1)[1]. The life expectancy at birth has also shown a tendency to increase every year in the U.S. as shown in Fig. 1.2 because of advances in healthcare, medical research, sanitation, and nutrition[2]. A U.S. child born in 2008 is expected to live four years longer than one who was born in 1981[3]. It is expected that the U.S. population over age 65 will be more than 20% over the total U.S. population in year 2050 (Table 1.1) due to increasing life expectancy and decreasing birth rates (Fig. 1.3)[4-5].

Figure 1.1 The U.S. population from 1980 to 2050. This plot is from "Remote and wireless patient monitoring systems" [2], and the original source is U.S. Census Bureau[1].
Figure 1.2 Life expectancy at birth: United States, 1940, 1950, 1960, 1970, and 1975-2008[3].

Table 1.1 Percent U.S. population over age 65 by year, 2000-2050. This table is from "Remote and wireless patient monitoring systems" [2], and the original source is U.S. Census Bureau[1].

<table>
<thead>
<tr>
<th>Year</th>
<th>Over 65 Population (millions)</th>
<th>% of Total U.S. Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>35</td>
<td>12.4%</td>
</tr>
<tr>
<td>2010</td>
<td>40</td>
<td>12.9%</td>
</tr>
<tr>
<td>2020</td>
<td>55</td>
<td>16.1%</td>
</tr>
<tr>
<td>2030</td>
<td>72</td>
<td>19.3%</td>
</tr>
<tr>
<td>2040</td>
<td>81</td>
<td>20.0%</td>
</tr>
<tr>
<td>2050</td>
<td>89</td>
<td>20.3%</td>
</tr>
</tbody>
</table>
On the other hand, there is a declining trend in the number of hospitals in the United States due to the structural change in the medical industry[6]. As shown in Fig. 1.4, the number of hospitals in the U.S. dropped from 7,000 to 5,700 between the year 1975 and 2005[2]. As a result, the cost of medical services has increased for patients and hospitals seek to reduce hospital admissions and the length of stay.
Figure 1.4 Number of Hospitals in the US, 1975-2005, Estimated 2015. This plot is from "Remote and wireless patient monitoring systems"[2], and the original source is American Hospital Association[6].

A method to maintain a high quality of healthcare with limited medical resources is to use remote health monitoring technologies, which results in a decreased length of stay in hospitals for patients, need for fewer personnel, increased coverage by existing personnel, reduction in errors, and thus more efficient use of healthcare resources[2]. Compared with other patient monitoring technologies such as physical measurements[7-10] and video capture[11-12], wearable ambulatory monitor (WAM) is a practical and affordable option to monitor activities of daily living (ADL). Inertial sensor-based WAMs have been proven effective in human activity recognition and fall detection[13-15]. With the advances of integrated circuit(IC) and micro-electromechanical systems(MEMS) technology, the performance of these sensors has increased and their unit prices have dropped dramatically during the past decade.
The objectives of this research include: 1) developing a low-cost hardware platform which contains micro-sensors, micro-processors, on-board memories and wireless transceivers for human activity analysis; 2) designing algorithms to classify sitting, standing, lying, locomotion, and falls in real-time using a single WAM; 3) validating the classifier design by benchmarking the performance along with other algorithms, tools, or commercial products; 4) using this system as a tool to study the gait pattern of groups with high fall risk, such as senior residents in Judson Manor.

This dissertation is organized as follows. Section 1.2 reviews related papers and prior works of our group. Chapter 2 explains the general idea of our posture classification system first, followed by the introduction of the inertial sensor we used and the fundamental principle of coordinate transformation, as well as the research method of this study. Chapter 3 introduces the preliminary testing using a VN-100 development board to stream the raw data to a personal computer for analysis using MATLAB. Chapter 4 introduces the second testing on twenty young Case students using a wearable human posture classification system. This system contains a VN-100 SMD chip and a microcontroller to generate the classification result in real-time. Chapter 5 introduces the testing designed to compare the accuracy and reliability of our wearable system and other commercial products on the elderly in free-living environments. Conclusions and future works are discussed in Chapter 6.
1.2 Literature Review on Human Activity Recognition

Plenty of works have been published to analyze various human activities, using various tools and techniques. This section first reviews publications related to classification of simple activities such as sitting, standing, lying, locomotion, and falls. Techniques using wearable inertial sensors is further reviewed and followed by the introduction of prior works in our lab.

1.2.1 Assessment Techniques

Common techniques for human activity recording can be classified into two major categories: subjective methods and objective methods. Subjective methods, such as interviews, diaries, and questionnaires, are relatively inexpensive tools but usually suffer from inaccuracy and biased interpretation[15]. Objective methods, which use various types of sensors to gather the information, can be further classified into categories using ambient sensors or wearable sensors. Ambient sensors are designed be placed in the living environment of the subject. For example, pressure sensors can be integrated with beds, pillows, or chairs to identify how much time the subject spends on sleeping or sitting[16]. Combined with suitable algorithms, an array of pressure sensors can also be placed under a mattress to analyze the sleeping pattern of the subject[7-8, 10]. Another example of ambient sensing technique is video capture. The activity patterns of a subject in a video image can be analyzed and classified by mathematical models such as Fuzzy Logic[17], Neural Network[12, 18-19], Hidden Markov Model[20-22], Bayesian networks[23], and Dynamic Time Warping[24]. In practical applications, infrared sensors[25], sound sensors[26], optical/ultrasonic sensors[27], and pressure
sensors\cite{16} are also used in combination to create a "smart home" environment\cite{16, 25, 27}. The advantages of such systems include less severe design limitations (power consumption, form factor, calculation power, ...etc) and less intervention from users\cite{28}. However, such systems are usually expensive and cannot monitor the subject outside a limited area. Also, privacy becomes a potential issue when video capture tools or wireless technologies are used.

1.2.2 Wearable Sensors

The other category of sensors used in objective monitoring is wearable sensor. These sensors include accelerometers, gyroscopes, magnetometers, goniometers, pressure sensors, pedometers, actometers\cite{13}, and wearable cameras\cite{29}. Inertial sensors such as accelerometers and gyroscopes are comparatively more common sensors used in the application of human activity monitoring for its low-cost and convenience to set up. However, such devices are usually intrusive to the user and the classification accuracy is usually low due to the relative movement between the sensor and the body \cite{14}. Accelerometers fabricated leveraging MEMS technology are known for their low cost, low power consumption, and high accuracy\cite{30-35}. These sensors are designed according to Hooke's law ($F = k \times x$) and Newton's second law ($F = m \times a$). The proof-mass displacement caused by acceleration can be picked up by capacitance change or physical property changes of piezoelectric or piezoresistive materials. From the application point of view, accelerometers provide information on both the static orientation and the dynamic movement of a subject. For example, Mathie, et al. proposed a accelerometer-based human posture monitoring system which separates
the d.c. signal (gravitational acceleration) and a.c. signal by a low-pass filter[36]. The a.c. signal was used to estimate energy expenditure and identify various dynamic movements[32]. This process later became a common signal processing technique for human posture classification.

Another common wearable sensor type is MEMS gyroscope. These gyroscopes measure the angular rate by taking advantage of Coriolis effect. The vibrating microfabricated proof-mass shows slight displacement when the sensor rotates; thus, the rotation speed/angular velocity can be converted to electrical signals using similar mechanisms as in MEMS accelerometers. Angular velocity can also be integrated into an angle after a period of time[37-39] or differentiated to an instant angular acceleration[39]. Based on the observation that falls of humans normally come with a peak in angular velocity, gyroscopes can be used for fall detection[40-43]. Angular acceleration and sensor orientation have also been researched for human posture classification. For example, in the work published by Bourke and Lyons, a bi-axial gyroscope is placed at the subject's sternum to measure rotations in two directions (trunk pitch and roll)[39]. The angular velocities are processed to resultant angular velocity, resultant angular acceleration, and resultant change in trunk-angle. According to their experimental data, the thresholds to distinguish a fall from ADL are 3.1 rad/s, 0.05 rad/s², and 0.59 rad, respectively.

In practical human activity monitoring systems or Personal Emergency Response Systems (PERS), inertial sensors are embedded with other components such as microcontrollers, batteries, wireless transmitters, antennas, etc. Such embedded
systems usually carry one of three major functions: data logging, data forwarding, or data processing[28]. For practical uses, these functions are sometimes performed by modern smartphones which carry powerful microprocessors, sufficient memory, and the ability to make emergency calls anywhere anytime[44-45].

**Data logger**

Data logging WAMs are designed to collect data from sensors and to preserve the data in the on-board memory[37, 46-48]. The collected data are not analyzed in real-time; instead, the data have to be uploaded to a PC for post-processing and offline analysis. Therefore, such systems consume less power for data processing and their sampling rates are usually higher because they are not restricted by the data uploading speed or the data processing time. Data logging WAMs are suitable for researching human activity patterns because complex data analysis algorithms can be performed in a powerful PC. However, for applications which require an immediate response, such as fall detection, these devices are not suitable because emergent events are not discovered until the next data upload.

**Data forwarder**

Data forwarding WAMs do not perform data analysis or data storage on-board; instead, the collected data are delivered to a base-station immediately (usually via a wireless transceiver)[31, 36, 38-39, 44, 49-53]. Their advantages include real-time response and compatibility with complex analysis algorithms. However, their power consumption is higher than the other two types of WAM due to the continuous wireless
data transmission. Also, their performances are usually limited by the data transmission technology. For systems using wireless transceivers such as Bluetooth, ZigBee, or Wi-Fi, the standard signal coverage range is usually less than 100m and increasing signal strength requires additional power consumption[54]; therefore, the uses of such systems are generally restricted to the building which contains the base-station. Also, the speed of wireless data transmission restricts the refreshing rate of a real-time system. A sampling frequency of 50Hz is generally the upper limit of such systems[31, 38, 49-50].

Data processor

Data processing WAMs analyze the information from on-board sensors using the local micro-processor and generate a real-time report to the user[28, 44, 55-59]. Depending on the design, the report can be uploaded to another device via wireless transceivers or a cable. Unlike data forwarders, the uploaded data of data processors are generally processed and thus more compact than the raw data from the sensors. A certain amount of memory is required for calculation on such devices but not as much as on data loggers. The major drawback of such systems may be the relatively high power consumption for real-time data processing. However, for applications such as fall detection, these systems are sometimes better candidates for the real-time response.
1.2.3 Activity Recognition Using Wearable Sensors

When a human activity classifier is being designed, two major design factors need to be determined in advance: sensor arrangement and the activity recognition method.

Sensor arrangement

The sensor type, amount, and locations on the subject are decided according to the application and activities that need to be recognized. Even though the amount of collectable information is theoretically in proportion to the amount of sensors on different body locations, the trend in developing a system for ADL monitoring is to reduce the number of sensors. For example, lying, sitting, and standing are usually differentiated by examining the accelerometer orientations on both the chest and the thigh[60]. Instead of using two accelerometers on different parts of the body, Najafi, et al. used a chest-mounted sensing device which contains one single-axis gyroscope and one single-axis accelerometer to differentiate sitting, standing, lying, and walking[37]. This group later added another single-axis accelerometer to the device for better sensitivity and specificity[61]. The single sensing unit solution proposed by Najafi, et al. had some limitations. First, it used very complex algorithms such as wavelet transform to analyze the data; thus, the output could not be generated in real-time. Second, the algorithms included a step of double integration to estimate the vertical displacement(VD) but the notorious "drift" issue was not resolved. As a result, the sensitivity and specificity for identifying sitting and standing were relatively low.

Another approach was presented by Karantonis, et al.[33] A waist-mounted, real-time
data processor containing only a tri-axial accelerometer was designed in their work. Since the sensor was worn on the waist, this processor could differentiate lying, sitting, and standing by comparing the z-axis acceleration readout and the gravitational acceleration vector. The reported accuracy was high but the results may suffer from various factors, such as changes in orientation or location of the sensor after continuous motions and the differences between individuals.

Activity recognition method

Besides the sensor arrangement, algorithms used for activity recognition are also an important part when a human posture classifier is being designed. According to the review paper published by Avci, et al. in 2010[62], activity recognition processes include five major steps: preprocessing, segmentation, feature extraction, dimensionality reduction, and classification.

Preprocessing is a step to discriminate useful information and noise from collected data. Common approaches include low-pass median filter[33, 49, 63], Laplacian[64], Butterworth filter[30, 65-66], and reconstruction of wavelet transforms[34, 37, 61, 63].

The second step, segmentation, is a necessary step to break up continuous data in time domain when the computing resource is limited, the data amount is large, or simply a real-time report is desired. One of the most prevalent segmentation methods is using sliding windows[49, 67]. The performance of such window-by-window analysis is affected by the chosen window width[49].
The third step is feature extraction. This transforms the original large input data into a reduced representation set of features[62]. Common extracted features include: time-domain features (mean[48, 68], amplitude peak[69], variance[48], ...etc.), frequency-domain features (spectral energy[70] and spectral entropy[61]), time-frequency domain features (wavelet coefficients[34]), and heuristic features (signal magnitude area[31, 33, 35], signal vector magnitude[33-35], tile angle[33, 35, 39, 42, 69], vertical velocity[71], ...etc.))[62].

The fourth step is dimensionality reduction. The purpose of this step is mainly to reduce the computational effort by removing irrelevant or redundant feature sets[62]. This step is sometimes skipped when it is not necessary.

Classification is the last step in human activity recognition. Various techniques have been applied. These approaches can be divided into two categories: analytical models and machine learning methods. Analytical models, such as decision tree[33, 50, 61, 71-72], decision tables[31, 36, 39], and fuzzy logic[35, 73-75], are usually threshold-based techniques. The advantage of such classifiers is less required calculation power; however, since the thresholds are usually fixed values, the performance of classification varies from subject to subject. The other category, machine learning methods, associates observations of movements to possible movement states in terms of the probability of the observation[15]. A large amount of training data are usually required in order to increase the accuracy of such classifiers. These methods include $k$-nearest neighbor[76], support vector machines[77], Naive Bayes classifier[74, 78], Gaussian mixture model[79], and hidden Markov model[77].
1.2.4 Prior Works in Our Group

Human posture classification has been researched by M. Aloqlah in our group since 2006[63, 73]. In his publication, as shown in Fig. 1.5, a wireless headband was built to transmit the forward signals from a 3-axis accelerometer, a temperature/humidity sensor, and a MEMS microphone to a laptop (the base-station) for offline analysis using MATLAB. As shown in Fig. 1.6, the classification algorithm (written in MATLAB codes) was designed to generate outputs including standing, sitting, bending, lying, walking, running, and posture transitions between sitting and standing. Posture transitions were identified by detecting the peak of the difference between the approximated versions (level 9 and level 1) of the wavelet transformed z-axis acceleration. The other eight static and dynamic postures were differentiated by a Fuzzy logic classifier, which took the mean accelerations over a window of certain period, the energy readings, and tilting angles of all the three axes as inputs (Fig. 1.7).

Figure 1.5 The system architecture of the preceding posture classifier.
Figure 1.6 The algorithm process flow of the preceding classifier
The algorithm shows high accuracy in identifying some static postures such as upright (sitting/standing), lying, and bending. However, some issues have been observed in this classifier. First, the system sometimes misclassifies the postures of sitting and standing because their signatures are almost identical. Initial conditions are required to "track" the change between sitting and standing; for example, if the previous status is "sitting" and a "posture transition" is detected, the next static posture
will be classified as "standing". Such design is impractical for long-term ADL monitoring because the technique to identify posture transitions does not provide any information about the postures before and after. As a result, any misclassified output causes further misclassification in the next window. Second, the classifier shows difficulty responding to slow posture transitions which makes the sitting/standing misclassification issue inevitable in real life. Third, running and walking cannot be successfully identified since wavelet transform technique identifies these two dynamic postures as "posture transition". Fourth, some of the fuzzy logic inputs are redundant; mean of the accelerations alone is sufficient to identify upright, lying, and bending. Last, wavelet transform consumes too much calculation power; thus, implementing such algorithms in a microcontroller for real-time posture monitoring is almost impossible.
Chapter 2 Proposed Solution and Research Method

2.1 Proposed Solution for Human Activity Classification

Real-time activity classification at local wearable processor level is desired for emergent cases such as falling and stumbling. The ultimate goal of this project is to build a system which can generate a report of activity immediately without processing the data using a PC. Therefore, the limitation of local resources such as memory size, sampling frequency, the instruction cycle of the microcontroller, physical dimensions, and power consumption need to be taken into consideration when a real-time system is being designed. Generally speaking, real-time systems are designed based on the following restrictions[33].

1) There is no knowledge of future events.
2) The amount of data that may be buffered is limited.
3) The amount of processing time available is limited.

The human posture classification system in this project is designed to identify sitting, standing, lying, locomotion (walking/running), and falls in real-time using only one WAM on a subject's waist. Compared to the prior classifier, the new design includes the following features: 1) simple real-time algorithms which can be processed in microcontrollers, 2) the capability to differentiate sitting and standing using only one WAM, 3) the capability of auto-calibration, 4) independence from previous data and results, and 5) the capability of fall detection.
Figure 2.1 presents the data flow of the new system. The first stage of the classifier is an analytical model which takes the accumulated activity level, averaged trunk orientation, and averaged VD over a period of time as inputs. This information is collected from inertial sensors and processed in real-time. Signal magnitude area (SMA)[31, 33, 35] and signal vector magnitude (SVM)[33-35] are two of the most commonly used indicators of activity level (static postures, transitions, and intense motion such as falls). In this work, these two parameters are used and compared in the stage of preliminary testing which classifies activities using MATLAB on a PC, and a new parameter is introduced and applied on the final wearable system. Body frame x-axis acceleration alone can provide enough information on the trunk orientation so that upright (sitting/standing) and lying postures can be differentiated. However, in order to compensate the effect of different user and different wearing location, the pitch angle of the device is read directly from VN-100 in the final system. The details of this calibration process is discussed in Section 4.2. Vertical displacement provides information of the elevation/height of the sensor, regardless of the orientation. Such additional information is expected to improve the accuracy of fall detection or daily activity classification comparing to systems using only the physical orientations and the vector magnitude of acceleration. Generally speaking, real-time VD monitoring provides following benefits: 1) enabling the system to differentiate sitting and standing, 2) tracking the user's current level in a 3D environment, 3) detecting falls end in upright position such as sitting, 4) providing additional information about the falling distance to determine the seriousness of falls, and 5) reducing the false-positive rate of falls caused
by bouncing of a mattress or a bed. In order to estimate VD in real-time, a series of algorithms are designed in this work to minimize the effect of the d.c. offset. These techniques are introduced in Section 3.3.

A rule-based classifier is implemented after the analytical model for classification. It combines the output of the first classifier and additional information to differentiate locomotion from posture transition, and fallen from lying. The output of this classifier is used to reset or to stabilize VD in upcoming window as well. Therefore, not only can this system be used to classify human postures but also be used for long term, real-time VD monitoring.

![Figure 2.1 Data flow of the proposed human posture classifier](image)

The major contributions of this work include the following: 1) presenting a new method to remove the d.c. offset from vertical acceleration and to estimate VD; 2) designing a series of algorithms for real-time human posture classification and fall
detection. The algorithms were reliable and simple enough to be implemented and executed in a local microcontroller for long term monitoring; 3) designing a hardware architecture specific to the proposed algorithms; 4) presenting a benchmarking procedure for performance comparison between different human activity classifiers.

2.2 Inertial Sensor and Fundamental Study

This section reviews the technical specification of the major inertial sensor used in this work, as well as the fundamental knowledge of coordinate alignment and VD estimation.

2.2.1 VN-100 Inertial Measurement Unit and Attitude Heading Reference System

Tracking the orientation of the sensor is necessary for VD estimation. In order to estimate VD, measured raw body frame accelerations have to be converted or aligned to the inertial frame first. VN-100 from VectorNav, Richardson, TX is used in this project for its capability to monitor the current orientation (attitude) and to output the orientation in the form of Directional Cosine Matrix, Euler angles, or Quaternion.

As shown in Fig. 2.2, a VN-100 device contains a tri-axis accelerometer, a tri-axis gyroscope, and a tri-axis magnetometer. The readings are calibrated by its Vector Processing Engine (VPE) and updated at a refreshing rate up to 200Hz. The orientation of VN-100 is tracked by the on-board sensors and corrected by a Kalman filter which takes gravitational acceleration (accelerometer), earth magnetic field (magnetometer), and angular velocity (gyro) as inputs. Using the information from these three sensors, the on-board Kalman filter effectively compensates the drawbacks in configurations
where any single sensor is used alone for attitude estimation[80]. For example, estimating the orientation by gravitational acceleration is effective and reliable when the sensor is stationary. However, the gravitational acceleration vector cannot be captured accurately when accelerations other than gravitation are applied to the sensor. Gyro-alone orientation computers are immune to linear acceleration; however, gyros are notorious for the drift over time and temperature. Also, since the rotation angles are integrated from angular velocity, errors accumulate very fast over time. Magnetometers suffer from artificial magnetic fields in our living environment.

![Figure 2.2 Block diagram of VN-100](image)

Figure 2.2 Block diagram of VN-100

Figure 2.3 shows the user interface of VectorNav Sensor Explorer, which is a software developed to setup VN-100 and visualize the collected data. This software is used in the preliminary testing (discussed in Chapter 3) for data acquisition. The USART
baud-rate is set to 921600bps in this project to stream the acceleration readings and quaternions at the maximum speed (200Hz).

![Figure 2.3 The user interface of Vectornav Sensor Explorer](image.png)

2.2.2 Coordinate Alignment/Transformation

In order to estimate the VD of the sensor, inertial frame z-axis acceleration has to be calculated from body frame accelerations. Such coordinate transformation/alignment can be expressed in three different forms: Direction cosines, Euler angles, and Quaternions. In *Strapdown inertial navigation technology*[81], these three representations are defined as follows:
**Direction cosines**

"The direction cosine matrix is a 3 × 3 matrix, the columns of which represent unit vectors in body axes projected along the reference axes."[81]

**Euler angles**

"A transformation from one co-ordinate frame to another is defined by three successive rotations about different axes taken in turn."[81]

**Quaternions**

"The quaternion attitude representation allows a transformation from one co-ordinate frame to another to be effected by a single rotation about a vector defined in the reference frame."[81]

In this project, quaternions are used to represent the sensor orientation for higher accuracy of attitude computation, less system burden, less memory requirements (four quaternions as opposed nine directional cosines), and to avoid Gimbal lock (singularity points when the pitch is approaching ±90°) in Euler angles[81].

The mathematical expression of quaternions is defined as

\[
q = \begin{bmatrix}
a \\
b \\
c \\
d
\end{bmatrix} = \begin{bmatrix}
\cos\left(\frac{\theta}{2}\right) \\
\left(\frac{\mu_x}{\mu}\right)\sin\left(\frac{\theta}{2}\right) \\
\left(\frac{\mu_y}{\mu}\right)\sin\left(\frac{\theta}{2}\right) \\
\left(\frac{\mu_z}{\mu}\right)\sin\left(\frac{\theta}{2}\right)
\end{bmatrix} = a + ib + jc + kd,
\]

(1)
where $\mu_x$, $\mu_y$, and $\mu_z$ are the components of the rotation vector $\mu$, $\mu$ is the magnitude of $\mu$, and $\Theta$ is the rotation angle.[81]. The mathematical rules for quaternion computation is expressed in Equation (2).

$$i \cdot i = -1 \quad i \cdot j = k \quad j \cdot i = -k \quad ... \text{etc.}$$ (2)

When quaternions are used to represent a vector without rotation, such as the measured body frame accelerations $r^b$ in Equation (3), the magnitude of rotation becomes zero and the quaternion can be simplified as a three-element vector.

$$r^{b'} = \begin{bmatrix} 0 \\ x \\ y \\ z \end{bmatrix} \rightarrow r^b = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \quad (3)$$

A vector rotation from the body frame ($r^b$) to the inertial frame ($r^i$) can be expressed as Equation (4).

$$r^{i'} = q \ r^{b'} q^* \quad (4)$$

$$q^* = (a -ib -jc -kd) \quad (5)$$

where $q^*$ is the complex conjugate of $q$. Equation (4) can be understood as two consecutive rotations about two different vectors and the second rotation, $q^*$, is to make sure the first quaternion parameter, $a$, of $r^{i'}$ is zero. Due to this reason, the angle appears in Equation (1) is only defined as half of the actual rotation angle. Equation (4) can be further simplified to the form of Equation (6) and (7).

$$r^i = Cr^b \quad (6)$$
where \( r' \) is the inertial frame accelerations including the gravitational acceleration vector

\[
g = \begin{bmatrix} 0 \\ 0 \\ 9.818 \end{bmatrix}.
\]  

### 2.2.3 Vertical Displacement Estimation

Theoretically, VD can be derived straight from double integration of g-corrected vertical acceleration. However, in reality, errors from various sources, such as d.c. offset coupled in the accelerometer output or errors from coordinate alignment[82], are also integrated during this process[83]. Therefore, the actual VD may be buried in accumulated errors after few seconds of process. A common method to compensate for this effect is applying a band-pass filter to the acceleration data before double integration or other processes[30, 65, 71, 84-85]. However, implementing high order band-pass filters consumes too much calculation power to make it practical for real-time microcontroller-based applications. This section introduces the algorithms used in this project for VD estimation.

In this project, VD takes the height of the subject's waist while standing as the reference level and is defined in the same direction of gravitational acceleration, which increases toward the ground. The integration is calculated by trapezoidal rule:

\[
VV(n + 1) = VV(n) + \frac{1}{2} \times t \times (VA(n) + VA(n + 1)) 
\]  

(9)
\[ VD(n + 1) = VD(n) + \frac{1}{2} \times t \times (VV(n) + VV(n + 1)) \]  

where \( VA \) is vertical acceleration derived from the VN-100 output readings (g-corrected body-frame accelerations and quaternions) using the rotation matrix introduced in Section 2.1.2, \( VV \) is vertical velocity, \( VD \) is vertical displacement, \( n \) is the number of the current sample, and \( t \) is the sampling period. The parameter \( t \) is set to 1/200 second in the preliminary testing (Chapter 3) and 1/100 second in the final wearable system (Chapter 4 and 5) due to the nature of calculation power difference between PC and microcontrollers. In order to accurately estimate \( VD \), offset control is important and needs to be taken care of in different systems. In this work, two different algorithm sets for offset control are implemented in the preliminary testing; the details will be discussed in Chapter 3.

2.3 Research Method

In order to design a system which meets the requirements as described in Section 2.1, this research is proceeded in three consecutive major stages:

1) implementing the classification algorithms in MATLAB on a PC and performing a preliminary testing for design concept validation; 2) designing an independent, wearable, and microcontroller-based system to carry out the C language version of the classification algorithms; 3) testing the system on Case students in laboratory environments and on the elderly in independent environments. Additional health information is also collected for gait analysis and fall risk score estimation in the testing
on elderly. Data collected from each testing are used to improve the performance of the system.
Chapter 3 Classification Algorithm Verification Using MATLAB

3.1 System Introduction

Chapter 3 introduces the first classification system designed in this research. Based on the general idea introduced in Chapter 2, this system used a VN-100 development board as the inertial sensor. The data collected from VN-100 were sent to and stored in a PC for offline analysis by classification algorithms implemented in MATLAB. Figure 3.1 shows the data flow of this system. Fuzzy logic was adopted as the core of the classifier. SMA was chosen to represent the activity level for reasons discussed in Section 3.2 and body frame x-axis acceleration was used as an indicator of sensor orientation.

![Figure 3.1 Data flow of the MATLAB-based human posture classifier](image-url)
Figure 3.2 shows the process flow of the classification algorithms implemented in MATLAB for the preliminary testing. Designed to generate output posture types in real-time, the system collects data from the sensor over a sliding, non-overlapping, 1-second window. For each 1-second window, SMA, mean of VD, and mean of body frame x-axis acceleration are sent to the fuzzy logic core. The fuzzy logic classifier then generates a preliminary output (sitting, standing, lying, transition, and intense motion). This preliminary output is further classified by a rule-based classifier. VD and vertical velocity may be reset according to the final output posture type. For the convenience of algorithm development, during the classifier design stage the data are stored in a personal computer (PC) as .txt files for offline analysis.
Figure 3.2 The process flow of the posture classification algorithms


3.2 Activity Level Estimation

Signal magnitude area (SMA), defined as "the sum of the areas under the moduli of the integrals, normalized to the length of the signal" by Karantonis, et al., is reported to be effective to distinguish between periods of user activity and rest[33]. The mathematical form of SMA is

\[
SMA = \frac{1}{t} \times \left( \int_t \left| x_i(t) \right| dt + \int_t \left| y_i(t) \right| dt + \int_t \left| z_i(t) \right| dt \right)
\]

(11)

where \( x_i, y_i, \) and \( z_i \) represent the inertial frame accelerations corrected for the gravitational acceleration \( g \), and \( t \) is the integration period. In the same paper, the authors also used signal vector magnitude (SVM) to detect possible falls. SVM is defined as

\[
SVM = \sqrt{x_i^2 + y_i^2 + z_i^2}
\]

(12)

where \( x_i \) is the \( i \)th sample of the x-axis signal (similarly for \( y_i \) and \( z_i \)). Theoretically, both SMA and SVM can be used to distinguish the activity levels of static posture (sitting, standing, and lying), locomotion, posture transition, and falling.

In order to decide which parameter to use in our work, a 30-year-old, male subject was asked to perform twelve different tasks and repeat each task five times. These tasks were sequences of actions including sitting in a chair, sitting on a 17cm-thick mattress, standing, lying on the mattress, walking at different paces, and simulated falls. These sequences are visualized in Fig. 3.3. The data collected by VN-100 from this
subject were stored in a laptop for SMA and SVM calculation. The collected data were also used to tune the membership functions of the fuzzy logic core. The sampled postures were categorized as follows: static postures (sitting, standing, and lying), falling, transitions (between sitting, standing, and lying), slow walking (25 to 31 paces in 20 seconds), normal walking (37 to 38 paces in 20 seconds), and fast walking (45 to 47 paces in 20 seconds). The total sample numbers are listed in Table 3.1.
Figure 3.3 Performed sequences of movement. The timeline is in unit of "seconds."
Table 3.1 Sample numbers of the activity level experiment. The data points were sampled randomly during the time of each specific activity/posture.

<table>
<thead>
<tr>
<th>Posture Types</th>
<th>No. Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>90</td>
</tr>
<tr>
<td>Falling</td>
<td>20</td>
</tr>
<tr>
<td>Transition</td>
<td>55</td>
</tr>
<tr>
<td>Walking(slow)</td>
<td>25</td>
</tr>
<tr>
<td>Walking(normal)</td>
<td>25</td>
</tr>
<tr>
<td>Walking(fast)</td>
<td>25</td>
</tr>
</tbody>
</table>

The results are visualized in Fig. 3.4. The SMA is normalized to 200 samples at a 200Hz sampling rate. The SVM is the maximum value over a 1-second window (200 samples). The first observation from the results is the outliers in the SVM data. Outliers may cause error to activity level classification and, thus, lead to poor posture classification performance. Second, regardless of the outliers in the SVM data, both SMA and SVM show good separation of "static postures," "transition," and "falling." Third, to differentiate the three walking speeds, the boundaries of SMA are clearer (no overlapping between each other) than those of SVM. Considering the outlier issue and imprecise boundaries of SVM, SMA is chosen to be one of the extracted features in this project to distinguish activity level. However, according to the results, the SMA of "walking" covers the range of both "falling" and "transition." Fuzzy logic with the information of SMA for activity level classification alone cannot differentiate "walking" from "transition" or "falling." Therefore, an additional rule-based classifier is designed to solve this problem.
Figure 3.4 Results of the experiment to compare SMA and maximum SVM
3.3 Offset Control in Vertical Displacement Estimation

This section first introduces the algorithms designed for d.c. offset control in VD estimation. With properly selected parameters, these algorithms minimize the effect of d.c. offset so that the system can maintain its accuracy for a long term application. The second part of this section discusses the result of an experiment to evaluate three different methods for d.c. offset estimation.

3.3.1 Algorithms for Offset Control

In order to suppress the d.c. offset, three algorithms are implemented as following:

**Algorithm 1:**

```plaintext
if max(abs(VA(window))) > threshold
   Do Integration by Trapezoidal rule
else
   Reset VV and pass through current VD to the next window
end
```

The first algorithm is designed to remove the offset when the subject is in stationary. It is based on the observation that humans cannot maintain moving at a constant speed without any acceleration. In other words, whenever the observed acceleration is close to zero, the velocity must be zero as well, and thus, no VD change should happen. The vertical acceleration(VA) in a window contributes to vertical velocity(VV) and VD only when the maximum VA of the window is above a pre-defined threshold, which is set to 1 m/s^2 in this work. The window sliding over the input data is non-overlapping and set to 1-second wide to conserve the system memory while
preserving enough details. In case the maximum acceleration is less than the threshold, the velocity of the whole window is set to zero and this window does not contribute any displacement change to the total VD. The effect of this algorithm is demonstrated in Fig. 3.5.

Figure 3.5 VD estimation in an offline analysis: (a) double integration without any d.c. offset technique applied; (b) d.c. offset is removed by subtracting the overall mean value from vertical acceleration (only valid when the initial and final accelerations are the same); (c) d.c. offset is removed by subtracting overall mean value from vertical acceleration and conditional integration technique is applied. VD in this figure is double-integrated from the VA of a subject performing sitting and standing in 40 seconds.
The second algorithm is designed for offset control during the integration process. The training data is collected from a 175 cm-tall subject, thus the maximum possible VD is about one meter (from the waist to the ground). In order to make sure the estimated VD falls in a reasonable range, the generated VD value is confined in the range between -0.2m and 1.2m.

Algorithm 2:

After integration,
if VD > 1.2
    Set VD to 1.2
if VD < -1.2
    Set VD to -0.2
end

According to our observation, d.c. offset is not a constant value but varies slowly over time. This effect is more obvious when the subject in dynamic motion such as walking or running. Thus, the third algorithm is designed to compensate offset in constant dynamic motion. This approach uses the classified posture output of the current window to estimate the offset in the next window based on the assumption that the variation of offset is much slower than the output refreshing rate, which is set to 1

Algorithm 3:

if Posture is “transition” or “intense motion”
    Reset DC_offset
else
    DC_offset of the next window is estimated using one of the three proposed methods.
end
The estimated offset will be compensated from the VA if the output posture type is static posture or locomotion (running/walking). Three methods are presented for this offset estimation: mean of VA, least-square of VA [86], and linear estimation of VV [87] as shown in Fig. 3.6. Their performance will be discussed and compared in the next section. In case the output is "posture transition" or "intense motion", the offset compensation will not take effect; thus, all information, including the offset, will be preserved and integrated in the next window. Even though VD estimation may not take advantage from the offset compensation techniques when the user is moving constantly, the algorithms are expected to take effect in fall detection when the user falls from the bed or during posture transition. A possible issue of this technique is that the first window of each persistent posture is not (or mistakenly) compensated. Therefore, a classification mechanism with high input error tolerance, such as fuzzy logic, is required.
Figure 3.6 Adopted methods for offset estimation; the acceleration or velocity in the next window is adjusted by subtracting the offset estimated using the data in the current window if the classified posture is static posture or locomotion. The offset estimation methods include: 1) mean of VA, 2) least-square of VA, and 3) linear estimation of VV.

3.3.2 An Experiment to Evaluate Methods for D. C. Offset Estimation

An experiment was conducted to benchmark the three methods for d.c. offset estimation in the next window: mean of VA(#1), least-square of VA(#2), and linear estimation of VV(#3). Three subjects were asked to wear a VN-100 on the chest and repeat a 30-second walk at a regular pace for six times. Since the subjects did not change their vertical height, the expected net VD change was zero and the measured maximum VD values were considered as uncompensated errors. The errors derived
using the three methods are compared with the error without any offset compensation, which is marked as #0 in Fig. 3.7 and Table 3.2.

![Main Effects Plot for Max VD](image)

**Figure 3.7** Experiment result comparison of the three proposed offset estimation methods and reference group

<table>
<thead>
<tr>
<th>Pair</th>
<th>Mean difference</th>
<th>SE of difference</th>
<th>T-value</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 vs 0</td>
<td>-4.378</td>
<td>7.924</td>
<td>-0.553</td>
<td>0.9455</td>
</tr>
<tr>
<td>2 vs 0</td>
<td>66.398</td>
<td>7.924</td>
<td>8.38</td>
<td>0</td>
</tr>
<tr>
<td>3 vs 0</td>
<td>-8.347</td>
<td>7.924</td>
<td>-1.053</td>
<td>0.7188</td>
</tr>
<tr>
<td>2 vs 1</td>
<td>70.776</td>
<td>7.924</td>
<td>8.9322</td>
<td>0</td>
</tr>
<tr>
<td>3 vs 1</td>
<td>-3.968</td>
<td>7.924</td>
<td>-0.5008</td>
<td>0.9586</td>
</tr>
<tr>
<td>3 vs 2</td>
<td>-74.74</td>
<td>7.924</td>
<td>-9.433</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 3.2** All pair-wise comparisons (Fisher's LSD) among different offset estimation methods. The difference is only considered significant when P < 0.05.

The results of this experiment, visualized in Fig. 3.7, are further analyzed by pair-wise t-tests (Table 3.2). It is observed that the second method, least-square of VA
showed significant worse result than the other two methods and caused more error than the reference group. The other two method, mean of VA and linear estimation of VV, showed improved offset suppression over the reference group. However, as shown in table 3.2, the result of analysis of variance (ANOVA) indicated that there was no significant evidence to prove that such improvement was because of the different offset estimation methods used.

3.4 The Core Classification Algorithms

This section introduces the algorithms implemented in Matlab for offline posture analysis. These algorithms include the Fuzzy logic classifier and the rule-based classifier.

3.4.1 Fuzzy Logic

Fuzzy logic is a many-valued logic based on the concept of "Computing with words" presented by L. A. Zadeh[88]. Unlike classical "binary" logic where the output (0 or 1) is determined by a specific threshold, the variables in fuzzy logic can be mapped into any value between 0 and 1 according to the membership functions. In practical uses, fuzzy logic usually converts numerical input variables into linguistic forms (fuzzification) by membership functions. The results are next evaluated by "rules of inference" (fuzzy implication). These linguistic rules are similar to human reasoning; for example, "if time (input 1) is between 11:30am to 1:00pm (condition 1) and status (input 2) is hungry (condition 2), then action (output 1) is go out for lunch (outcome 1)". Next, the strength of each outcome is calculated. This step is called "aggregation." According to the chosen aggregation method (additive or maximum), an output fuzzy set is generated. The
element number of the output fuzzy set is the same as the number of possible outcomes. The last stage is called "defuzzification," which evaluates the output fuzzy sets with the pre-defined output membership functions and generates a crisp value as the final output of the fuzzy logic system. Common defuzzification methods include "center of area" and "mean of maximum." Fuzzy logic is used in this work for its robustness against the error accumulated over time from the sensor, signal processing, coordinate transformation, and integration.

The fuzzy logic core of this classifier is implemented using Fuzzy Logic toolbox in MATLAB R2009a. It takes the mean of the x-axis body frame acceleration, the mean of VD, and SMA over a 1-second window as inputs and generates a preliminary output posture type. The membership functions of the x-axis body frame acceleration in Fig. 3.8 is designed according to the angle between the inertial frame z-axis and the body frame x-axis as shown in Fig. 3.9. The range of the acceleration is expanded from [-9.8m/s², 9.8m/s²] to [-12m/s², 12m/s²] in order to cover additional acceleration caused by dynamic motions.
Figure 3.8 Membership functions of the fuzzy logic classifier
Figure 3.9 A picture of the hardware used for the testing and the defined orientation. The orientation of the sensor is defined as the angle between the fixed inertial frame z-axis toward the ground and the body frame x-axis. The $X_b$ values marked in red color are the boundaries of membership functions.

The membership functions of VD are arbitrarily designed to divide the whole range (-0.4m to 1.4m) of possible VD into three even parts: high, mid, and low. Note that the membership functions are named according to the height of the sensor but not the value of VD. The membership functions of SMA are designed as "low," "mid," and "high" based on the collected training data. The SMA values of static postures such as sitting, standing, lying, and fallen are categorized as "low". Posture transitions and fallings are categorized as "mid" and "high" postures, respectively. The overall range of SMA is from 0 to $3000\text{m/s}^2$. The membership functions cover the corresponding SMA values from the training data and are extended so that all possible values are covered.
Thirteen fuzzy rules, as listed in Table 3.3, are defined to differentiate sitting, standing, lying, posture transition, intense motion (possible falling), and error caused by loosely attached sensors.

### Table 3.3 Fuzzy rules

<table>
<thead>
<tr>
<th>mean of $X_b$ is</th>
<th>mean of $VD$ is</th>
<th>SMA is</th>
<th>output is</th>
</tr>
</thead>
<tbody>
<tr>
<td>vertical</td>
<td>low</td>
<td>low</td>
<td>sitting</td>
</tr>
<tr>
<td>vertical</td>
<td>mid</td>
<td>low</td>
<td>sitting</td>
</tr>
<tr>
<td>vertical</td>
<td>high</td>
<td>low</td>
<td>standing</td>
</tr>
<tr>
<td>horizontal</td>
<td>low</td>
<td>low</td>
<td>lying</td>
</tr>
<tr>
<td>horizontal</td>
<td>mid</td>
<td>low</td>
<td>lying</td>
</tr>
<tr>
<td>horizontal</td>
<td>high</td>
<td>low</td>
<td>lying</td>
</tr>
<tr>
<td>horizontal</td>
<td>low</td>
<td>mid</td>
<td>lying</td>
</tr>
<tr>
<td>horizontal</td>
<td>mid</td>
<td>mid</td>
<td>lying</td>
</tr>
<tr>
<td>horizontal</td>
<td>high</td>
<td>mid</td>
<td>lying</td>
</tr>
<tr>
<td>vertical</td>
<td>high</td>
<td>mid</td>
<td>transition</td>
</tr>
<tr>
<td>vertical</td>
<td>mid</td>
<td>mid</td>
<td>transition</td>
</tr>
<tr>
<td>inverted</td>
<td></td>
<td>high</td>
<td>intense motion</td>
</tr>
<tr>
<td>inverted</td>
<td></td>
<td></td>
<td>error</td>
</tr>
</tbody>
</table>

#### 3.4.2 Rule-Based Classifier

The fuzzy logic classifier is followed up by a rule-based classifier for more detailed activity classification. This technique is a common approach in fall detection[89].

At this level, a falling-flag and a locomotion-counter are introduced to differentiate falling and locomotion from posture transitions. The falling-flag is set when "intense motion" is detected in the previous fuzzy logic stage. When falling-flag is set to 1, any "lying" posture detected is classified as "fallen". The falling-flag is reset to 0 when the output posture becomes "standing." The second variable introduced is the locomotion-counter. Since the SMA of "locomotion" covers the range of "transition" and "intense motion"
motion," these postures can only be differentiated by counting the elapsed time. It is assumed that a regular posture transition does not last longer than six seconds. Therefore, any continuous "transition" or "intense motion" is considered as "locomotion" when this counter is larger than six (seconds). This locomotion-counter is reset to 0 when a posture other than "transition" or "intense motion" is indentified.

Table 3.4 lists all the actions done for each fuzzy logic output type.

### Table 3.4 Follow-up rules

<table>
<thead>
<tr>
<th>Fuzzy logic output</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>sitting</td>
<td>Reset vertical velocity and locomotion-counter. Change the output to &quot;fallen&quot; if a &quot;falling&quot; event is detected previously (falling-flag is set).</td>
</tr>
<tr>
<td>standing</td>
<td>Reset vertical velocity, vertical displacement, and locomotion-counter. Reset the falling-flag if the falling-flag is set.</td>
</tr>
<tr>
<td>lying</td>
<td>Reset locomotion-counter and vertical velocity. Change the output to &quot;fallen&quot; if falling-flag is set.</td>
</tr>
<tr>
<td>transition</td>
<td>Add 1 to locomotion-counter. Change the output to &quot;locomotion&quot; and reset the vertical displacement if locomotion-counter &gt; 6.</td>
</tr>
<tr>
<td>intense motion</td>
<td>Set falling-flag. Add 1 to locomotion-counter. Change the output to &quot;locomotion&quot; and reset the vertical displacement if locomotion-counter &gt; 6.</td>
</tr>
<tr>
<td>error</td>
<td>Reset locomotion-counter.</td>
</tr>
</tbody>
</table>

### 3.5 Testing Setup

The evaluation process of the proposed classifier was inspired by the work published by Karantonis, et al.[33] The testing was conducted in a lab with a 18 inch-high chair and a 7 inch-thick mattress. Six volunteered testing subjects were asked to wear the sensor (a VN-100 Development board) on the waist using a Velcro strap or a
belt as shown in Fig. 3.9 and to repeat the posture sequences as shown in Fig. 3.3 five times. The VN-100 development board streamed raw data including body-frame acceleration and quaternions to a PC for data storage using Vecternav Sensor Explorer, the data acquisition software provided by VectorNav. The actual posture classification occurred in an offline analysis using Matlab. The data process flow was the same as introduced in Section 3.1. The exact Matlab code for the classification and Fuzzy Logic parameters can be found in Appendix 1 and 2, respectively. Data from the first testing subject (subject 0) were first analyzed and used as the "training" data to optimize the parameters in the classifier, such as the threshold defined in Algorithm 1 (see Section 3.3.1) and the Membership Functions in the Fuzzy Logic core. Then the optimized parameters were further used to analyze data from the other five subjects.

3.6 Data Analysis and Discussion

In order to evaluate the report generated by the classification algorithms, the expected movement sequences were defined in advance as in Fig. 3.10. The output sequences of the classifier could then be compared with the expected sequences. Data from the first testing subject (subject 0) and data from the other five subjects were analyzed separately since some parameters used in the classification algorithms were optimized to the data from the first subject. Parameters used to evaluate the performance of the classifier include accuracy, sensitivity, and specificity. Algorithms 1 and 2 introduced in Section 3.3.1 were implemented in the classifier. The effect of offset estimation algorithm 3, dynamic offset removal (DOR), was evaluated as well. When DOR was activated, the system used the mean vertical acceleration from the previous "non-
transition" window (method #1 discussed in Section 3.3.2) to estimate the offset dynamically.

Figure 3.10 Expected sequences of movement

The performance was first represented by the accuracy of each activity sequence performed. Table 3.5 and Table 3.6 show the testing results on the first and the other five subjects, respectively, in "accuracy," which was defined as the rate between the classified sequences matching the expected sequences and the total testing / sample numbers of each sequence type.
In this analysis, only the order of detected postures was taken into consideration. In other words, a classified posture sequence was determined as "correct" as long as its order matched the expected sequence, regardless of the timing offset mismatch. Such analysis method eliminated the influence of the response time/delay of the testing subject and simplified the testing setup as well. However, the response time of the system could not be measured in this case. To address this issue, the experiment setting in the upcoming testing (Chapter 4) included a video recorder so that the period of each posture could be recorded precisely.

Table 3.5 Experiment results of posture sequences on subject 0

<table>
<thead>
<tr>
<th>Sequence number</th>
<th>Movement class</th>
<th>Total sequences</th>
<th>Overall correct (DOR)</th>
<th>Overall correct (no DOR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQ1</td>
<td>Static posture</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td>25</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>SQ6</td>
<td>Walking</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ8</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>SQ9</td>
<td>Falls and recovery</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ10</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>SQ11</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>SQ12</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Sequence number</td>
<td>Movement class</td>
<td>Overall incorrect (DOR)</td>
<td>Overall incorrect (no DOR)</td>
<td>Accuracy (DOR)</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
<td>-------------------------</td>
<td>-----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>SQ1</td>
<td>Static posture</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ2</td>
<td>Static posture</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ3</td>
<td>Static posture</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ4</td>
<td>Static posture</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ5</td>
<td>Static posture</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ6</td>
<td>Walking</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ7</td>
<td>Walking</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ8</td>
<td>Walking</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ9</td>
<td>Falls and recovery</td>
<td>0</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>SQ10</td>
<td>Falls and recovery</td>
<td>3</td>
<td>0</td>
<td>40%</td>
</tr>
<tr>
<td>SQ11</td>
<td>Falls and recovery</td>
<td>0</td>
<td>1</td>
<td>100%</td>
</tr>
<tr>
<td>SQ12</td>
<td>Falls and recovery</td>
<td>1</td>
<td>0</td>
<td>80%</td>
</tr>
<tr>
<td>Subtotal</td>
<td></td>
<td>4</td>
<td>1</td>
<td>80%</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>4</td>
<td>1</td>
<td>93.3%</td>
</tr>
</tbody>
</table>

Table 3.6 Experiment results of posture sequences on 5 testing subjects
The experimental results of the training data reached 100% accuracy for the first eight sequences, which included orientation changes and locomotion. This result was not surprising because the classifier was optimized to the training data. However, misclassified posture sequences were observed when the tested sequences included falling and recovery such as sequences 10 to 12. The causes of misclassification were analyzed and shown in Table 3.7. It was observed that VD was sometimes mistakenly estimated when the subject fell if DOR was activated. A possible explanation was that the classifier might not be able to identify the posture change in the first one or two windows when falls happened. As a result, the vertical acceleration change was
considered as the offset and removed in the next window; therefore, the estimated VD became smaller than it should be. The overall accuracy was improved when DOR was disabled; however, a misclassified sequence sample due to SMA error was observed in this case. This sample was correctly classified when DOR was activated. A further inspection of this misclassified sample indicated that the SMA level fell in the "high" range when the subject was actually "fallen;" thus, the posture was misclassified as "intense motion". A possible cause of this phenomenon was that the SMA ranges of posture transition and intense motion were slightly overlapped in the training data and the membership functions were designed according to the training data with DOR disabled. Therefore, the calculated SMA in the misclassified sample with DOR disabled was slightly above the boundary between "mid" and "high." When DOR was activated, the offset in accelerations used to estimate SMA was removed, and, thus, the SMA was reduced to the level of "mid." This misclassification can be corrected by redesigning the membership functions so that the SMA does not fall in the range of "high." However, doing so may reduce the sensitivity to falls.

Table 3.7 Error analysis over 60 sampled sequences on subject 0

<table>
<thead>
<tr>
<th>Error description</th>
<th>Samples (percentage), DOR</th>
<th>Samples (percentage), no DOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct</td>
<td>56 (93.3%)</td>
<td>59 (98.3%)</td>
</tr>
<tr>
<td>VD error</td>
<td>4 (6.7%)</td>
<td>0</td>
</tr>
<tr>
<td>SMA level error</td>
<td>0</td>
<td>1 (1.7%)</td>
</tr>
<tr>
<td>Orientation error</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SMA over falling threshold</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Transition overtime</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Multiple errors</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
As expected, the testing results of the data from the other five subjects were lower in accuracy than the results of the training data. The overall accuracy of samples with and without DOR dropped from 93.3% and 98.3% to 85% and 86.3%, respectively. As observed in the training data, classification algorithms with DOR reduced the overall accuracy. Besides errors caused by wrong VD estimation and SMA level, orientation error, SMA thresholding error, and transition overtime were also observed in the general testing as shown in Table 3.8. Among all these errors, SMA thresholding error became the majority of the reasons causing misclassification. This was due to the different falling patterns of each individual. There are two possible methods to reduce such error. The first method is simply expending the training data size and collect the training data from more subjects so that a more generalized data base can be established. These subjects should better include people in different age, gender, and body size. The second way is categorizing the testing subject/user first and using corresponding training data to optimize the result.

**Table 3.8 Error analysis over 300 sample sequences on 5 testing subjects**

<table>
<thead>
<tr>
<th>Error description</th>
<th>Samples (percentage), DOR</th>
<th>Samples (percentage), no DOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct</td>
<td>255 (85%)</td>
<td>259 (86.3%)</td>
</tr>
<tr>
<td>VD error</td>
<td>15 (5%)</td>
<td>10 (3.3%)</td>
</tr>
<tr>
<td>SMA level error</td>
<td>0 (0%)</td>
<td>1 (0.3%)</td>
</tr>
<tr>
<td>Orientation error</td>
<td>6 (2%)</td>
<td>6 (2%)</td>
</tr>
<tr>
<td>SMA over falling threshold</td>
<td>17 (5.7%)</td>
<td>19 (6.3%)</td>
</tr>
<tr>
<td>Transition overtime</td>
<td>3 (1%)</td>
<td>3 (1%)</td>
</tr>
<tr>
<td>Multiple errors</td>
<td>4 (1.3%)</td>
<td>2 (0.7%)</td>
</tr>
</tbody>
</table>
The second way to represent the performance was by extracting individual posture types from the recorded posture sequences and comparing them with each other using "sensitivity" and "specificity"[90]. These two parameters were defined as follows:

\[
\text{Sensitivity} = \frac{\text{True positives}}{\text{True positives} + \text{False negatives}} \times 100\% \\
\text{Specificity} = \frac{\text{True negatives}}{\text{True negatives} + \text{False positives}} \times 100\%
\]

In these equations, sensitivity was the rate of successfully identifying a specific posture when it really happened and specificity was the rate of successfully identifying a posture that did not happen when it really did not. The samples of postures were extracted from the posture sequences and analyzed in Table 3.9 - Table 3.12. In the results of training data, both sensitivity and specificity of each posture were higher than 90% because the classifier was well-optimized to the training data. It was also observed that enabling DOR sometimes caused wrong estimation of VD and changed the SMA value; thus, the sensitivity of some postures such as standing and lying was reduced. In the general testing on the five other subjects, the results of most postures were above 85%, except for the sensitivity of sitting and lying, and specificity of standing. This was due to the errors described in Table 3.8.

<p>| Table 3.9 Results of experiments on subject 0 with DOR activated, expressed in individual postures |
|---|---|---|---|---|
| Performed | Output | Sitting | Standing | Lying | Locomotion |
| Sitting | 20 | 0 | 0 | 0 |
| Standing | 0 | 86 | 0 | 0 |</p>
<table>
<thead>
<tr>
<th></th>
<th>Lying</th>
<th>0</th>
<th>0</th>
<th>41</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locomotion</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Fallen</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Fallen</td>
<td>Transition /Error</td>
<td>Sensitivity</td>
<td>Specificity</td>
</tr>
<tr>
<td>Sitting</td>
<td>0</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Standing</td>
<td>4</td>
<td>0</td>
<td><strong>95.6%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Lying</td>
<td>4</td>
<td>0</td>
<td><strong>91.1%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Locomotion</td>
<td>0</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Fallen</td>
<td>20</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>95.3%</strong></td>
</tr>
</tbody>
</table>

**Table 3.10** Results of experiments on subject 0 with DOR disabled, expressed in individual postures

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Sitting</td>
<td>Standing</td>
<td>Lying</td>
<td>Locomotion</td>
</tr>
<tr>
<td>Sitting</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Standing</td>
<td>0</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Lying</td>
<td>0</td>
<td>0</td>
<td>45</td>
<td>0</td>
</tr>
<tr>
<td>Locomotion</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>Fallen</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Fallen</td>
<td>Transition /Error</td>
<td>Sensitivity</td>
<td>Specificity</td>
</tr>
<tr>
<td>Sitting</td>
<td>0</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Standing</td>
<td>0</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Lying</td>
<td>0</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Locomotion</td>
<td>0</td>
<td>0</td>
<td><strong>100%</strong></td>
<td><strong>100%</strong></td>
</tr>
<tr>
<td>Fallen</td>
<td>19</td>
<td>1</td>
<td><strong>95%</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

**Table 3.11** Results of experiments on five testing subjects with DOR activated, expressed in individual postures

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Sitting</td>
<td>Standing</td>
<td>Lying</td>
<td>Locomotion</td>
</tr>
<tr>
<td>Sitting</td>
<td>81</td>
<td>5</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>Standing</td>
<td>1</td>
<td>433</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>Lying</td>
<td>0</td>
<td>0</td>
<td>183</td>
<td>0</td>
</tr>
<tr>
<td>Locomotion</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>75</td>
</tr>
</tbody>
</table>
Table 3.12 Results of trail experiments on five subjects with DOR disabled, expressed in individual postures

<table>
<thead>
<tr>
<th>Performed</th>
<th>Output</th>
<th>Sitting</th>
<th>Standing</th>
<th>Lying</th>
<th>Locomotion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sitting</td>
<td>76</td>
<td>10</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Standing</td>
<td>0</td>
<td>446</td>
<td>0</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Lying</td>
<td>0</td>
<td>0</td>
<td>193</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Locomotion</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>Fallen</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

The testing results of this experiment are concluded as the three following observations:

1) fuzzy logic combining with rule-based classifiers is a valid technique for offline analysis of human posture;
2) sensor orientation (on the waist), SMA level, and VD can be combined to indicate the current activity including sitting, standing, lying, locomotion, and falls, of the user;

3) d. c. offset in VD estimation can be effectively reduced to an acceptable level by algorithms 1 and 2. However, algorithm 3 cannot be proven useful in most of cases.

The design concept has been validated in this chapter. Since our final goal is to design a portable and wearable device which generates real-time reports of the user activity, some changes need to be made in the design. Issues such as power consumption, calculation power, and timing control also have to be taken into consideration. Experiences from the experiment discussed in this chapter also help improving the experiment design in the future. Chapter 4 will discuss the next stage of this project.
Chapter 4 An Embedded Solution for Real-Time Human Activity Classification

4.1 System Overview

A PC-based human posture classifier taking the body-frame X-axis acceleration, SMA, and VD as inputs has been introduced in Chapter 3. The next phase of this work was to design a wearable system to carry out the same function in real-time. The whole system contained two parts: a battery-powered wearable device and a PC. Unlike the PC-based classifier, in this new design, real-time activity reports were generated locally in the wearable device, which worked as a data processor (see Section 1.2.2), and stored in the local memory. The stored activity reports were wirelessly uploaded to a base-station, a laptop in this case, for visualization or further offline analysis. The data transmission could basically be any type of protocols. Considering the convenience and the relatively small data amount, wireless transceivers based on ZigBee (IEEE 802.15.4) technology were used in this work.

As shown in Fig. 4.1, the device could be broken down to five major parts according to the functionality. First, all electrical components were powered by a 1000mAh polymer lithium ion battery (PRT-00339 provided by Sparkfun Electronics, Inc., Colorado) which was regulated from 3.7V to 3.3V. Theoretically, a fully charged battery can sustain at least 9 hours of continuous activity monitoring. This value was calculated using the estimated power consumption of each major components as listed in Table 4.1.
Table 4.1 The Estimated Power Consumed by Each Major Components in the Portable Device

<table>
<thead>
<tr>
<th>Components</th>
<th>Description</th>
<th>Power Consumption(mA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VN-100</td>
<td>200 mW (listed in spec.); Proximately 91mA when powered by 3.3V</td>
<td>91</td>
</tr>
<tr>
<td>EEPROM 25AA1024</td>
<td>Reading current: 5 to 7mA depending on the driving voltage</td>
<td>6</td>
</tr>
<tr>
<td>PIC18LF2620</td>
<td>Rough estimation. The power can be controlled by switching between normal and sleep modes</td>
<td>10</td>
</tr>
<tr>
<td>Red LED</td>
<td>Indicating on/off status. Lowered to half from 3.3mA by setting to blink</td>
<td>1.6</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>108.6</strong></td>
</tr>
</tbody>
</table>
The second part was the inertial sensor. The sensor orientation (quaternions and pitch) and body-frame accelerations were sampled and digitized by a VN-100 SMD chip. It was factory calibrated and set to update the output reading every 10ms to the microcontroller via the Serial Peripheral Interface (SPI) serial bus.

The third part was the processing unit. A PIC18LF2620 microcontroller collected data from the VN-100 SMD sensor and generated the report of the current activity/posture. It was chosen for its low cost (about 6.14 USD on www.digikey.com), high operating frequency (up to 40 MHz), sufficient amount of program and data memory (65535 bytes and 3968 bytes, respectively), and low driving voltage (2 V to 5.5
V). The microcontroller was set to run at 16 MHz, which was the maximum frequency at 3.3 V supplied. It was programmed to operate at sleep mode to save power while not being used and to wake up when the inertial data from the VN-100 SMD were updated (using interrupts).

The fourth part was the memory for data storage. In PIC18LF2620, the built-in EEPROM (for data storage only) was 1024 bytes, which was not sufficient for either the designed testing on Case student introduced in this chapter or the field testing introduced in the next chapter. Therefore, an external EEPROM unit (25AA1024 from Microchip Technology, Inc., Arizona) was implemented in the scheme. This 1 Mbit (131072 bytes) EEPROM unit was accessed via the SPI serial bus. In the testing discussed in Section 4.3, there were seven parameters (the mean of pitch, the mean of G-Corrected SVM square(GCSVM²), VD, result 1 and 2, and falling-flag 1 and 2) needed to be recorded for offline analysis and visualization. Each parameter was stored as floating-point numbers, which occupied 4 bytes in the memory. Therefore, 25AA1024 can hold up to 78 minutes of continuous data recording if the data are logged every second. (Note that the report was generated every 10 ms, but the data/reports did not have to be logged at the same speed. In case the logging speed was slower than 10 ms, unsaved data were overwritten, even though the falling-flag can still be triggered.) In case only the results from the two classification algorithms were needed and the data were logged every minute, 25AA1024 were able to monitor the testing subject continuously for more than ten days without releasing the memory space by uploading the data (regardless of the battery life).
The last major part of the wearable device was the XBee2 wireless transceiver from Digi International, Inc., Minnesota. It communicated with another XBee2 attached to the base-station at a speed of 250 kbps. The setting parameter of these two XBee2 devices can be found in Appendix 3. The base-station was a laptop PC. Data received from the XBee2 dongle were analyzed and visualized using MATLAB and stored in the hard drive. The detailed PCB layout of the wearable device can be found in Appendix 4. Figure 4.2 shows picture of a wearable device in the final package and a XBee2 wireless transceiver mounted on a USB dongle. The enclosure was 2 5/8” by 2 1/8” and designed to be worn on the waist with a belt.

Figure 4.2 The exterior view and dimensions of the portable human activity classifier
4.2 The Classification Algorithms

The concept of using analytical models and VD to classify human activities has been implemented on a PC in MATLAB codes and validated in Chapter 3. While being converted to C language-based codes, which can be compiled and executed on PIC microcontrollers, three major changes were made to improve the performance or to accommodate the hardware limitation. The first change was the parameter used to represent the physical orientation of the sensor. In the MATLAB-based algorithm, the x-axis acceleration in the body frame was read directly from the inertial sensor and sent to the Fuzzy Logic classifier as one of the three input parameters. A 2% error (see Table 3.8) due to mistakenly estimated trunk orientation was observed when applying the thresholds from the training data on different subjects. This error was believed from different body shape of the testing subjects or different sensor-wearing location (even the sensors were supposed to be worn on the same location). Also, the x-axis acceleration was not linearly proportional to the rotation angle and, thus, any offset in the location/orientation might affect the sensitivity of rotation. As a solution, the pitch-angle of the device was read directly from VN-100 and the device was calibrated in the beginning of any recording period. The effect of different user and wearing location was expected to be reduced. The second change of the algorithm was the parameter used to represent the activity level. Even though SMA has been validated in differentiating static postures and various dynamic activities, its calculation involved g-corrected accelerations in the inertial frame. Reading these values from VN-100 took additional time and delayed the process. Considering the limited calculation power (speed) of the
PIC microcontroller, this step was avoided and, instead, a new parameter, the mean value of G-corrected Signal Vector Magnitude Square (GCSVM$^2$) over a half second window (at 100Hz sampling rate) was used to represent the activity level. The mean value of GCSVM$^2$ only involved the body-frame acceleration, which had to be read from VN-100 anyway. It was defined as following:

$$mean \ of \ GCSVM^2 = \frac{\sum_{i=0}^{51} |X_{b,i}^2 + Y_{b,i}^2 + Z_{b,i}^2 - 9.8^2|}{51}$$

The last part being changed in the code was the analytical model for classification. Resources for data processing can be considered as limitless in a PC for offline data processing. However, in microcontroller-based embedded systems, floating-point calculation in Fuzzy Logic required too much system resources to maintain the sampling frequency at 100Hz. A simpler structure, decision tree, was therefore implemented to replace the Fuzzy Logic classifier.

In order to see if a decision tree-based classifier can take advantage from real-time VD estimation and, thus, increase the classification accuracy, two different algorithms were implemented in the code and generated reports concurrently for comparison. The first algorithm used only the information of sensor orientation (mean of pitch) and activity level (mean of GCSVM$^2$) as shown in the flow chart in Fig. 4.3. With properly chosen T1 and T2 (from subject 0), the mean of GCSVM$^2$ might fall in the range less than T1, between T1 and T2, or larger than T2. In case of the value less than T1, the pitch value alone could determine the posture to be sitting, standing, or lying. A parameter called "falling flag" was implemented to indicate if a falling event has been
detected. The output posture was classified as *fallen* if the falling flag was set while the status was *lying*. In case the mean of GCSVM$^2$ was between T1 and T2, a counter(C1) was activated to monitor the time that the value stays in this range. If C1 was less than 500 (5 seconds), the output posture was classified as *transition/walking*; otherwise, it was determined as *walking*. In some scenarios that the testing subject stumbled but recovered immediately, a later lying posture might be classified as *fallen*. Therefore, a second counter(C2) was used here to rule out such situations by resetting the falling flag to 0 if C2 reached 300 (3 seconds). The falling flag was also reset when *fallen* has been detected and the subject recovered from the *fallen* status. In case the mean of GCSVM$^2$ was larger than T2, the algorithm simply determined the activity as *falling* and set the falling flag to 1.
Figure 4.3 Algorithm 1 for human activity classification: a decision tree structure using only pitch and GCSVM$^2$ as inputs. T1 is set to 3, and T2 is set to 80 in the testing. These values are decided according to the data collected from subject 0.
The second algorithm was designed based on the same concept as algorithm 1 but VD was added as the third input parameter. Figure 4.4 shows the process flow chart. The error in VD estimation were compensated by methods introduced in Section 3.3.1. However, it was observed that the error control was only reliable in static condition or dynamic activity lasting less than 5 seconds. In other words, VD was only helpful in differentiating postures if both the last and the current mean of GCSVM$^2$ were less than T1 or the mean of GCSVM$^2$ was at the falling edge after an active period less than 500 counts. In other situations, the algorithm basically followed the same rules as the first algorithm. In order to maintain the long-term stability of the VD estimation, VD was reset to pre-defined height levels after generating each successfully classified output, if possible. The pre-defined height of standing, sitting, and lying were 0, 0.5m, and 0.07m, respectively. These values should be altered according to the testing environment. In addition, the vertical velocity value was reset to zero when a static posture was detected so that the error in vertical velocity did not accumulate. The C code for activity classification can be found in Appendix 5.
Figure 4.4 Algorithm 2 for human activity classification: a decision tree structure using pitch, GCSVM, and VD as inputs. T1 is set to 3, and T2 is set to 80 in the testing. These values are decided according to the data collected from subject 0.

4.3 A Benchmarking Protocol for Performance Evaluation on Young Subjects

A benchmarking protocol was designed and used to test the performance of the wearable human posture classification system. Twenty undergraduate/graduate students, from 18 to 35 years old, were recruited from Case Western Reserve University as the testing subjects. Among these volunteers, the male to female ratio was 13:7 (not controlled in the recruiting process). These subjects were first interviewed for their willingness of participation and their health condition. The subjects were next asked to
provide information including gender, age, height, and weight for further data analysis and to sign off the informed consent document (ICD) for Case IRB review before the testing. Case IRB approved this testing protocol with the protocol number: IRB-2013-456. The testing took place in a laboratory arranged as shown in Fig. 4.5. A camera was used to record the activity performed by the subjects. The testing environment included a 7\"-thick sleeping mattress, a 18\"-high chair, a 30\"-high desk, and sufficient space for walking and simulated falls. The subject were asked to wear the device on the waist as shown in Fig. 4.1 and to repeat the activity sequences listed in Table 4.2 five times. The recorded sequences were later compared with the reference sequences, which were classified by the researcher according to the video clip and manually logged into a file in the same format as the output from the sensor. Figure 4.6 shows a sample of the comparison between the output result and the reference result.
Figure 4.5 The arrangement of the testing environment used in the testing on Case students (not in scale)

Table 4.2 The activity sequences designed for the testing on Case students. The duration of each lasting activity is 10 seconds.

<table>
<thead>
<tr>
<th></th>
<th>Activity Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>standing - sitting - standing - lying(face up) - lying(face left) - lying(face right) - standing - walking - standing - walking - sitting - walking - lying - walking - stamping - walking - lying</td>
</tr>
<tr>
<td>2</td>
<td>standing - sitting - falling while standing up - fallen - walking - lying</td>
</tr>
<tr>
<td>3</td>
<td>standing - falling(fainted) - fallen - walking - lying</td>
</tr>
<tr>
<td>4</td>
<td>standing - walking - falling(stumbled) - fallen - walking - lying</td>
</tr>
<tr>
<td>5</td>
<td>standing - lying - falling from the desk to the mattress - fallen - walking - lying</td>
</tr>
</tbody>
</table>
Figure 4.6 A sample of output data sequence 1 from subject 1: a) result of classification algorithm 1, b) result of classification algorithm 2, c) the reference sequence.
The data/reports collected in this testing were stored in a PC for further analysis. Four different evaluation tools were used to benchmark these two proposed classification algorithms.

**Evaluation Tool 1**

The first tool was designed to evaluate the performance of the classifier using second-by-second accuracy. This value can be derived by summing up the total counts that the classified results match the reference activity. This method was simple but the value of accuracy was low and did not represent the performance very well due to the mismatch of time stamp in both the recorded and the reference activity sequences. Such mismatch was unavoidable because it happened during the process of manual data alignment for MATLAB analysis. Therefore, a second method to estimate the classification accuracy was also introduced. It took the classified activity as a successful count as long as the classified posture type matches any reference data in the previous second, the current second, or the next second. The MATLAB code generating these two accuracy values can be found in Appendix 6. The results of these two classification algorithms were first expressed in terms of the testing subject, so that the variety among the testing subjects was able to be reviewed. The results were also expressed in terms of activity sequences. See the results in the next section.

**Evaluation Tool 2**

The second tool was designed to evaluate the performance of fall detection by accuracy. The fall detection accuracy was derived by summing up all counts of detected
falls (regardless of the time and duration) and dividing the number by the total expected falls, which was 20. The averaged fall detection accuracy using the two classification algorithms was compared with each other.

**Evaluation Tool 3**

The third tool used One-Way Analysis of Variance (ANOVA) to evaluate the performance of posture classification on different testing groups. The testing subjects were first divided into three groups according to their height: under 170cm (6 people), between 170cm and 178cm (7 people), and above 178cm (7 people). A confidence level of 95% was used in the One-Way ANOVA test in Minitab 16 to analyze the overall activity accuracy and fall detection accuracy. The null hypothesis, that there was no significant difference in these three groups, was validated if the P-value was greater than 0.05. The similar process was also used to analyze the relationship between the weight of testing subject and the output classification accuracy. In this analysis, the testing subjects were divided into three groups according to their weight: under 60kg (6 people), between 60kg and 70kg (7 people), and above 70kg (6 people). The testing results will be shown and discussed in the next section.

**Evaluation Tool 4**

The fourth tool further evaluated the fall detection performance by second-by-second sensitivity and specificity. These two parameters followed the same definition as in Section 3.6 and represented the performances on positive and negative cases; a higher value of sensitivity means the system was more likely to trigger the alarm when a
fall happened and a higher value of specificity meant the system was less likely to generate a false-alarm on falls.

4.4 Testing Result and Discussion

The testing results analyzed by the four evaluation tools mentioned previously are discussed in this section.

Result Using Evaluation Tool 1

As the result analyzed using the first tool, Fig. 4.7 and Fig. 4.8 compare the accuracy of human activity classification on different testing subjects using analytical method 1 and 2, respectively. At first, it was observed that the second accuracy calculation method generated values higher than the first method did, as expected. It increased the overall classification accuracy using classification algorithm 1 from 81.33% to 89.96% and the same parameter using classification algorithm 2 from 80.07% to 88.43%. It was also observed that classification algorithm 1, which used only physical orientation and activity level as inputs, showed a higher overall classification accuracy (and the averaged accuracy on most of the subjects) than classification algorithm 2, which used VD in additional to the other two inputs. In other words, VD did not help improve the overall classification accuracy. One possible reason was that, even though VD was designed to be automatically calibrated and the error was compensated, the estimated value was still not able to represent the actual VD very well; thus, a 1.5% accuracy drop occurred.
Figure 4.7 Testing result in averaged accuracy analyzed using analytical method 1 and compared by subject

Figure 4.8 Testing result in averaged accuracy analyzed using analytical method 2 and compared by subject
The classification accuracy was also reviewed by averaging the accuracy on all the testing subjects and expressed in terms of the five pre-defined activity sequences. These five sequences simulated different situations that may occur in daily life. The results were illustrated in Fig. 4.9 and Fig. 4.10. A worth-mentioning observation in these two figures was that even though the overall accuracy of algorithm 1 was higher than the value of algorithm 2, the results of accuracy on sequence 1 and 2 were otherwise. The accuracy of algorithm 1 (using analytical method 2) on sequence 1 was 0.95% lower than the accuracy of algorithm 2. Similarly, the accuracy of algorithm 1 (using analytical method 2) on sequence 2 was 0.7% lower than the accuracy of algorithm 2. Therefore, it was proved that the algorithm using VD indeed improved the classification accuracy in certain situations that sitting and standing were involved, such as the activities performed in sequence 1 and 2. Such result was not surprising because the algorithm without VD used the pitch angle only to differentiate sitting and standing. The pitch angle of sitting varied with the habit and the body shape of each subject, as well as the shape of the chair. Adding VD as the second input decreased the dependency on the pitch angle and, thus, generated a more reliable output activity.
Figure 4.9 Testing result in averaged accuracy analyzed using analytical method 1 and compared by sequence

Figure 4.10 Testing result in averaged accuracy analyzed using analytical method 2 and compared by sequence
**Result Using Evaluation Tool 2**

The second evaluation tool inspected the accuracy of fall detection. The results were compared in terms of individual subjects as shown in Fig. 4.11. It was observed that, in most cases, classification algorithm 1 caught more fall events than algorithm 2 (92.5% vs. 82.5%), except for the cases on subject 1 and 13. Such result might be caused by the wrong VD estimation due to the fast rotation and the impact when falls happened. The data from subject 1 were used to decide the threshold of activity level (GCSVM^2) and, thus, the accuracy values from both algorithms were 100%. Subject 13, on the other hand, showed a unusually low accuracy in all cases (35% fall detection using both algorithms and 10% lower than other subjects using the first evaluation tool). Such low accuracy might be caused by the following two reasons. First, by inspecting the recorded video clip, we found that this subject talked and moved a lot during the period of static activities. Second, the sensor might not be calibrated correctly in the beginning of the testing.
Figure 4.11 Testing result of the fall detection function compared by sequence

Result Using Evaluation Tool 3

The third evaluation tool was a follow-up analysis from the result of the first and the second evaluation tools. The averaged classification accuracy of each subject was first categorized into three height groups (separated by 170 cm and 178 cm), three weight groups (separated by 60 kg and 70 kg), two age groups (separated by 30 years old), and 2 gender groups (male and female) as described in the previous section. Then, the grouped accuracies were analyzed using One-Way ANOVA under the assumption that these accuracies were normally distributed in order to see if there was any significant differences among these groups. The results are shown in Table 4.3. Since the P-value was larger than 0.05 in all cases, it can be concluded that the height and weight of the subjects did not affect the accuracy of either algorithm 1 or algorithm 2.
Table 4.3 Analytical results of One-Way ANOVA on accuracy grouped by height or weight

<table>
<thead>
<tr>
<th>Response</th>
<th>Factor</th>
<th>P-value</th>
<th>Is $H_0$ valid? (P&gt;0.05)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (Alg.1)</td>
<td>Height Group</td>
<td>0.598</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.2)</td>
<td>Height Group</td>
<td>0.845</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 1)</td>
<td>Height Group</td>
<td>0.396</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 2)</td>
<td>Height Group</td>
<td>0.415</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.1)</td>
<td>Weight Group</td>
<td>0.615</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.2)</td>
<td>Weight Group</td>
<td>0.956</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 1)</td>
<td>Weight Group</td>
<td>0.479</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 2)</td>
<td>Weight Group</td>
<td>0.954</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.1)</td>
<td>Age Group</td>
<td>0.265</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.2)</td>
<td>Age Group</td>
<td>0.435</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 1)</td>
<td>Age Group</td>
<td>0.095</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 2)</td>
<td>Age Group</td>
<td>0.268</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.1)</td>
<td>Gender Group</td>
<td>0.634</td>
<td>Yes</td>
</tr>
<tr>
<td>Accuracy (Alg.2)</td>
<td>Gender Group</td>
<td>0.883</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 1)</td>
<td>Gender Group</td>
<td>0.334</td>
<td>Yes</td>
</tr>
<tr>
<td>Fall Detection Accuracy (Alg. 2)</td>
<td>Gender Group</td>
<td>0.734</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Result Using Evaluation Tool 4**

The fourth tool evaluated the performance on fall detection by sensitivity and specificity. The results are shown in Table 4.4. The first activity sequence did not include any kind of falls; therefore, the sensitivity value was not available for calculation. Three observations could be concluded according to the results. First, The specificity values of algorithm 1 and 2 were both very high (~99.8%). In other words, the system was unlikely to trigger the alarm of falling mistakenly. Second, the sensitivity values of falling were lower than the corresponding accuracy values in the third tool. The reason was that second-by-second sensitivity calculation took time and duration into consideration; misalignment of the time stamp affected the results and was unavoidable in most cases.
It was also observed that, even though in some cases algorithm 2 showed a very close sensitivity to algorithm 1, algorithm 1 outperformed algorithm 2 in fall detection in most cases (overall sensitivity: 75.2% vs. 66.88%). The error might be caused by mistakenly estimated VD. Last, both of the two algorithms showed much lower fall detection sensitivity on Sequence 5, which involved falling from a bed(lying) to the ground. A possible explanation was that the impact cause by such falls was usually not as drastic as those caused by falls from upright position. Adjusting the threshold of activity level(GCSVM$^3$) might be able to improve the sensitivity.

### Table 4.4 Averaged sensitivity and specificity of fall detection

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sensitivity</strong> Algorithm 1</td>
<td>N.A.</td>
<td>77.66%</td>
<td>77.09%</td>
<td>76.42%</td>
<td>68.05%</td>
<td>75.2%</td>
</tr>
<tr>
<td><strong>Specificity</strong> Algorithm 1</td>
<td>99.5%</td>
<td>99.97%</td>
<td>99.94%</td>
<td>99.92%</td>
<td>99.94%</td>
<td>99.85%</td>
</tr>
<tr>
<td><strong>Sensitivity</strong> Algorithm 2</td>
<td>N.A.</td>
<td>75.22%</td>
<td>66.64%</td>
<td>76.44%</td>
<td>50.28%</td>
<td>66.83%</td>
</tr>
<tr>
<td><strong>Specificity</strong> Algorithm 2</td>
<td>99.64%</td>
<td>99.94%</td>
<td>99.86%</td>
<td>99.9%</td>
<td>99.69%</td>
<td>99.81%</td>
</tr>
</tbody>
</table>

According to the testing results, this chapter can be wrapped up by the following conclusions: 1) algorithms using physical orientation of the waist, GCSVM, and VD provides a certain level of capability classifying activities; 2) the algorithm with a feedback loop gives stable VD information and helps improve the accuracy classifying sitting and standing; however, accuracy of detecting some activities (such as falling) using such algorithm may be lower than the algorithm using only the physical orientation and activity level (GCSVM$^3$); 3) other than statistic postures, walking,
posture transitions, and falls can also be detected by both algorithms; 4) the performances of the proposed algorithms are not affected by the height of weight of the user; 5) the system rarely triggers false-alarm on fall detection; and 6) the algorithms can be carried out on low-cost microcontrollers with reasonable building price (below 500USD).

The capability of the system to classify human activities has been validated and benchmarked in this chapter; however, its long-term stability has not been tested yet. Also, its performance has only been tested on young college/graduate students. Tests on users in different age group are necessary. In the next chapter, the same system will be tested on elderly and the long-term performance will be compared with some commercial products, such as Jawbone UP, Fitbit Flex, and Nike+ Fuelband.
Chapter 5 Verification of the Activity Classifier on Elderly

5.1 Objectives

The experiment introduced in this chapter was designed for three major purposes: validation of human posture classification algorithms on elderly, data collection for algorithm optimization, and commercial product benchmarking. As an additional study, feedback from the elderly was also collected and quantified to show a general idea of the acceptance of such wearable sensors among senior users.

5.2 Testing Setup

In order to achieve long-term activity monitoring, the process for both data acquisition and data storage had to be modified to accommodate the limited memory, regardless of the battery life. The EEPROM implemented in the system was 1 Mbit (131072 bytes). The information needed to be stored included the output results from classification algorithm 1 and algorithm 2 (unsigned character, 1 byte each), and the mean of GCSVM^2 (floating-point, 4 bytes). Adding 2 bytes for memory alignment, the total size of each stored information, or so-called a "word", was 8 bytes. As a result, only 16384 "words" could be stored. In case of logging the results every second as implemented in Chapter 4, the memory was only enough for four and half hours of continuous human activity monitoring and data logging. In order to extend the working time, the process was redesigned to generate and store the report every minute. For each stored report, only the major output posture type of the 60 classified activity
outputs in a minute and the sum of the GCSVM² mean were logged. The extended working time was up to 11 days. The modified process flow is illustrated in Fig. 5.1.

The testing was designed to be conducted in the living environment of 15 recruited volunteers from Judson Manor, Judson Park, and South Franklin Circle. The ages of these volunteers were from 72 to 92 years old. The volunteers were instructed to wear our device on the waist as shown in Fig. 4.1 and three commercial sensors (Fitbit Flex, Jawbone Up, and Nike+ Fuelband as shown in Fig. 5.2) on the wrist for a day, from the morning to the evening until they went to bed for sleep. These commercial sensors were based on similar technologies which measures acceleration from MEMS accelerometers and generates reports of step counts, walking distance, total active time, longest continuous active time, and calories burned as listed in Table 5.1. Basic information such as gender, height, weight, and age was provided by the volunteers before the testing, as well as recent history of illness and medication. The fall risk of each testing subject was also evaluated using the MAHC-10 fall risk assessment tool.

![Figure 5.1 Data flow of the final wearable human activity classifier - the bottom-lined parameters refer to the average value in the corresponding loop.](image-url)
which was developed by Missouri Alliance for Home Case. This tool is attached in
Appendix 7. The subjects were requested to log their activities in a provided activity log
during the day of testing. A survey using System Usability Scale(SUS) was also conducted
at the end of the testing for each participant. The testing protocol was approved by Case
IRB with number IRB-2013-637.

**Table 5.1 Reported Parameters from the Sensors Used in the Testing on elderly**

<table>
<thead>
<tr>
<th>Reported Parameters</th>
<th>Jawbone UP</th>
<th>Fitbit Flex</th>
<th>Nike+ Fuelband</th>
<th>Our Classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calories</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Step Counts</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Wearing Hours</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Active Time</td>
<td>√</td>
<td>√</td>
<td></td>
<td>√*</td>
</tr>
<tr>
<td>Posture Type</td>
<td></td>
<td></td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Fall Alarm</td>
<td></td>
<td></td>
<td></td>
<td>√</td>
</tr>
</tbody>
</table>

* calculated by summing up the minutes of all active posture types
5.3 Data Analysis and Discussion

The data collected using three smart wristbands and our device are attached in Appendix 8. Two testing subjects quitted for personal reasons before the testing; therefore, the recorded data only included 13 subjects. One of the 13 testing subjects reported an event of falling during the day of testing. However, our activity sensor did not catch this event because the system shut down after the fall. The exposed battery connector and wire might be vulnerable to impacts and any short period of power disconnect caused the microcontroller reboot itself. Three other testing subjects reported device power off before the end of the testing day. According to the hardware
inspection, the on/off button were accidentally pressed. The front panel has been redesigned and strengthened to avoid such accidents in testing on the last nine subjects. The averaged battery life from the nine testing in which the battery fully depleted without any intervention was 1125 minutes, or 19 hours approximately. This number was about twice than the estimated battery life (9 hours).

The collected data was first analyzed by comparing the reported parameters: step counts, burned calories, estimated distances, and active time. The reported step counts are shown in Fig. 5.3. In most cases, Fitbit Flex reported the highest step counts and Nike+ Fuelband generated the least. In other words, Fitbit Flex was the most sensitive device in counting steps. Since the burned calories and moved distances were calculated from the estimated step counts, the same order was also observed in the burned calories (Fig. 5.4) and distance (Fig. 5.5); Flex and UP reported higher number than Fuelband did. Figure 5.6 compares the active minutes during the testing period. The active minutes in our classification algorithms were calculated by summing up the time of all "active" activities. Obviously, the threshold between "active" and "inactive" could be defined arbitrarily by different manufacturers. Therefore, such comparison was only meaningful in showing the correspondence of the reports from different devices. An unlikely high value of active time (815 minutes) was reported on subject number 6 by our classification algorithm 1. Since the result from algorithm 2 on the same subject was in a normal range and the thresholds of "active" in these two algorithms were the same, such discrepancy can only be explained as an "bug" in the hardware implementation.
Figure 5.3 Step counts from the testing on elderly

Figure 5.4 Burned calories from the testing on elderly
Figure 5.5 Estimated distances (km) from the testing on elderly

Figure 5.6 Active time (minutes) from the testing on elderly. The value of algorithm 1 on subject number 6 is 815 minutes and truncated in this plot. This value is obviously misestimated.
The second method to analyze the data was comparing the result activities generated by our algorithms with the activity log recorded manually by the test subjects. In order to derive their correspondence, the activity log was first converted into a sequence of "known" activities as listed in the first column of Table 5.2.

**Table 5.2 Activity conversion table for human posture classifier performance evaluation on elderly**

<table>
<thead>
<tr>
<th>Logged activity</th>
<th>Possible activity type</th>
<th>Assigned code in the classification algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleeping (bed)</td>
<td>Lying</td>
<td>A</td>
</tr>
<tr>
<td>Paperwork, Eating, Meeting, Using computer</td>
<td>Sitting, Transition</td>
<td>B</td>
</tr>
<tr>
<td>Watching TV, Reading, Listening to music/radio, Resting, Seeing dentist</td>
<td>Sitting, Lying</td>
<td>C</td>
</tr>
<tr>
<td>Doing exercise, Walking dog, Traveling, House work, Shopping</td>
<td>Standing, Transition, Walking</td>
<td>D</td>
</tr>
<tr>
<td>Driving, Taking bus</td>
<td>Sitting, Transition, Walking</td>
<td>E</td>
</tr>
<tr>
<td>Device taken off</td>
<td>Sitting, Standing, Lying</td>
<td>F</td>
</tr>
<tr>
<td>Partying, Mixed activity, Painting</td>
<td>Sitting, Standing, Transition, Walking</td>
<td>G</td>
</tr>
<tr>
<td>Fallen</td>
<td>Fallen</td>
<td>H</td>
</tr>
</tbody>
</table>

Then the classification results using algorithm 1 and algorithm 2 were compared with the "logged activity" sequences minute by minute until the last available data where either the subject took off the sensor or the battery was depleted. A successful count was determined when the classification result matched the "possible activity". The correspondence between the manual log and the classification result was calculated using the following equation:
Correspondence = \[ \frac{\sum_{N=1}^{Data\ length} Counter(N)}{Data\ length}, \]

where Counter(N) is 1 if the Nth classification result matches any of the possible activity type in the second column of Table 5.2, otherwise Counter(N) is 0.

The result correspondence of the activity log from the 13 testing subjects and the classification results using algorithm 1 and algorithm 2 was listed in Table 5.3. The average correspondences of Algorithm 1 and Algorithm 2 were 60% and 62.3%, respectively. The overall range was from 35.7% to 84.5%. Note that even though the equation to calculate "Correspondence" was similar to the one used to calculate "Accuracy" in Chapter 3 and Chapter 4, these two parameters should not be compared with each other. The parameter "Accuracy" was used to represent the correspondence between the "actual" activity recorded by a camera and the classification result. The parameter "Correspondence" used here, on the other hand, represented how close were the "manually" logged activity and the classification results. Since the manually logged activity and the corresponding time stamp were not as accurate as the video captured by camera, the "Correspondence" was expected to be much lower than the "Accuracy".
Table 5.3 List of correspondences between the activity log and classification results using different algorithms

<table>
<thead>
<tr>
<th>Subject Number</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log versus Alg. 1(%)</td>
<td>43.3</td>
<td>76.4</td>
<td>73.3</td>
<td>36.4</td>
<td>53.7</td>
<td>84.5</td>
<td>47.7</td>
<td>48.8</td>
</tr>
<tr>
<td>Log versus Alg. 2(%)</td>
<td>59.2</td>
<td>76.2</td>
<td>73.5</td>
<td>35.7</td>
<td>53.1</td>
<td>84.5</td>
<td>48.1</td>
<td>66.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subject Number</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log versus Alg. 1(%)</td>
<td>64.7</td>
<td>50.1</td>
<td>75.7</td>
<td>67.0</td>
<td>58.1</td>
<td>60.0</td>
</tr>
<tr>
<td>Log versus Alg. 2(%)</td>
<td>64.6</td>
<td>50.0</td>
<td>71.5</td>
<td>66.9</td>
<td>61.2</td>
<td>62.3</td>
</tr>
</tbody>
</table>

The correspondences listed in Table 5.3 were further analyzed by comparing these values with the basic information and MAHC score of the testing subjects (attached in Appendix 9). Table 5.4 shows the dependences (Pearson Correlation) between the correspondences using different algorithms and some attributes of the testing subjects, such as age, weight, and height. According to this table, it can be concluded that the performances of the two classification algorithms implemented in this work were independent from these three attributes since the correlations were all close to 0.

Table 5.4 Dependences between different attribute groups and correspondences using different classification algorithms

<table>
<thead>
<tr>
<th>Variable 1</th>
<th>Variable 2</th>
<th>Pearson Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Age</td>
<td>Corr. Alg. 1</td>
<td>0.165</td>
</tr>
</tbody>
</table>
According to the description of the MAHC 10 Fall Assessment Tool, a score of 4 or more is considered at risk for falling. Therefore, the testing subjects were divided to two groups: high fall risk group and low risk group. Among all 13 testing subjects, only 3 of them were in the low risk group. The analysis results are listed in Table 5.5. Even though the mean correlation using Algorithm 1 was higher than the same parameter derived using Algorithm 2, their P-value were both larger than 0.05. In other words, there was no significant evidence saying different fall risk group may affect the performance of the classification algorithms.

<table>
<thead>
<tr>
<th>Risk Group</th>
<th>Algorithm</th>
<th>Mean correlation between the log and classification results</th>
<th>P-value</th>
<th>Std</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>1</td>
<td>63.26</td>
<td>0.15</td>
<td>15.26</td>
</tr>
<tr>
<td>Low</td>
<td>1</td>
<td>49.03</td>
<td></td>
<td>5.28</td>
</tr>
<tr>
<td>High</td>
<td>2</td>
<td>64.82</td>
<td>0.23</td>
<td>14.08</td>
</tr>
<tr>
<td>Low</td>
<td>2</td>
<td>54.03</td>
<td></td>
<td>4.77</td>
</tr>
</tbody>
</table>

Table 5.5 One-Way ANOVA comparing the results of different fall risk groups
As an additional research, feedbacks from the testing subjects were collected at the end of each individual testing using a System Usability Scale (SUS) survey tool. This tool consisted of a 12 item questionnaire as showing below:

1. I think that I would like to use Jawbone UP or any similar system frequently.

2. I think that I would need the support of a technical person to be able to use these systems.

3. I found the various functions in these systems were well integrated.

4. I would imagine that most people would learn to use these systems very quickly.

5. I found these systems very cumbersome to use.

6. I needed to learn a lot of things before I could get going with these systems.

7. I would like to know detailed activity information such as sitting, standing, lying, and walking rather than just active/inactive at a time of a day.

8. I prefer wearing a sensor on the wrist more than wearing it on the waist.

9. I think automatic real-time fall detection is an important function of such devices.

10. I would not consider an activity logger which requires charging the battery every two days even if it provides additional functionalities such as differentiating various postures and fall detection.

11. What is your maximum acceptable price for an activity monitor with automatic fall detection functionality?
12. Any other comments?

The first 10 questions came with five response options for respondents, from Strongly agree to Strongly disagree, and the last two questions were open-ended. The first 10 questions were designed in a way that positive and negative questions were distributed evenly and, thus, a final score adjusted to the range between 0 to 100 can be derived using the following equation:

\[ \text{SUS score} = 2.5 \times (20 + Q_1 - Q_2 + Q_3 - Q_4 + Q_5 - Q_6 + Q_7 - Q_8 + Q_9 - Q_{10}) \]

The average SUS score collected from the 13 testing subjects was 66.3. Only one SUS score from the subjects was lower than 50. In other words, most testing subjects were positive to such smart systems with fall detection functionality. Several observations from the result of SUS are listed as below:

1. Acceptable price (one-time payment) of such smart systems with fall detection functionality ranged from 0 to 500 USD and the average acceptable price was 89.23 USD.

2. Even though most of commercial activity sensors are wrist-band type, 54% of the testing subjects did not have any preferred wearing location (between the wrist or the waist) to wear such devices. 30% of the subjects preferred wearing the sensors on the waist and only 15% people preferred the wrist.

3. Over 92% of testing subjects agreed that automatic real-time fall detection is important.
4. Without the fall detection functionality, only 15% of the testing subjects were willing to use activity trackers such as Jawbone UP or Fitbit Flex to monitor their daily activity. The complete results collected from the 13 testing subjects are listed in Appendix 10.

5.4 Conclusion

This chapter described the testing process of our human activity classification algorithms on elderly. An updated firmware utilizing the same algorithms as used in Chapter 4 was installed in our wearable system and tested along with three commercially available smart wristbands. The active/inactive time from the report generated by our system was compared with the value from these wristbands and the minute by minute correspondence between our report and the activity log was also calculated for performance evaluation. The correspondences of Algorithm 2(62.3%) was 2.3% higher than the value of algorithm 1(60.0%). The improvement was contributed by the additional information of VD. The reported activity time from our algorithm was proportional to the value generated by the commercial activity trackers. These products were designed for athletes and, thus, the thresholds to determine "active" motion were set higher than those for daily activities, such as our system. Besides, a SUS tool was also used to collect feedbacks from the testing subjects. Useful information such as preferred locations to wear the device and the acceptable pricing range can be used as a reference in the future system design.
Chapter 6 Conclusion and Future Work

6.1 Conclusion

In this research project, we presented the structure of a Fuzzy logic-based human posture classifier and fall detector. Following the defined structure, we first implemented a classification algorithm in Matlab, as well as a feedback-loop to stabilize d.c. offset in the vertical acceleration in order to estimate vertical displacement. This classification algorithm was further refined and modified to accommodate the limitations of embedded systems. Implemented in a customized wearable inertial sensing device, our algorithm was able to classify sitting, standing, lying, posture transition, walking, intense activity, falling, and fallen in real-time. The performances of the system with algorithm 1 (using tilt angle and activity level only, for comparison) and algorithm 2 (using tilt angle, activity level, and VD) were benchmarked in our lab on 20 Case students. The accuracy of posture classification was 89.96% using algorithm 1 (without VD) and 88.43% using algorithm 2 (with VD). The fall detection accuracy was 92.5% and 82.5% using algorithm 1 and 2, respectively. These algorithms also showed a 99.8% specificity on fall detection. In other words, these algorithms can effectively differentiate various daily activities and falls while avoiding false-alarm in fall detection. However, while providing additional information of vertical displacement, algorithm 2 sacrificed 1.5% of accuracy in daily activity classification and almost 10% in fall detection. Such performance drop was believed due to occasional error observed in VD estimation. A refined d.c. offset removal algorithm may be needed to resolve this problem.
In order to validate the classification algorithms, the wearable system was also
tested on 13 senior volunteers in their living environment (independent living
apartments, such as Judson Park/Manor and South Franklin Circle). The performance
was evaluated by comparing the classification results with three other commercial
products and the activity log provided by each testing subject. It was observed that the
total active minutes estimated by our algorithms was generally higher but proportional
to the reports from the commercial smart wristbands. The minute-by-minute
correspondence between our classification result and the activity log was about 60%,
which was not satisfying but reasonable for the inaccuracy of the manually logged
activities.

6.2 Future Work

The wearable system for human posture classification and fall detection
designed in this work has been tested and validated on both young and senior testing
subjects. For the future work, this research can continue on four major topics: algorithm
improvement, hardware redesign, long term field testing, and application. Even though
the accuracy of our algorithms are comparative to other works (~90%), there are still
something we can do to make them better. For example, some parameters, such as
thresholds for different activity levels and the boundary tilt angle of upright and lying
positions in our algorithms, are decided according to the data collected from a small
amount of samples. Parameters decided using a large amount of data from testing
subjects in different age, weight, or height group are expected to improve the
performance in posture classification and fall detection dramatically. Besides, the same
decision tree or Fuzzy logic structure can be customized to include more input parameters for different uses. For example, in case the living environment of the user/subject is known and limited in a certain range, adding a digital compass (magnetometer) will provide additional information on the heading direction and potentially help identifying his/hers activity.

The second category of future work is the improvement on the hardware. As reported by some of the testing subjects, our hardware prototype is bulky regarding its physical dimensions and weight. Electrical components can be replaced by smaller or lighter parts; however, the rechargeable battery becomes the bottleneck of current technology. According to the feedback from senior testing subjects, charging the battery every two days is the minimum acceptable period. A battery capacity of 2000 mAh is required for such long term monitoring but the size and weight of such battery is doubled from the battery used in our prototype. Without the advance of battery technology, future system designers can only extend the battery life by preserving the calculation power such as reducing the sampling/update frequency or using less power-consuming components.

The third topic of the future work in this research project is to validate the system in a long term field testing. The longest testing period in this work is about 20 hours (due to the limited battery life). With a refined firmware and a redesigned hardware platform, the system should ideally be able monitor testing subjects for a complete week. Also, combining with ambient sensors such as pressure sensors-integrated carpets, chairs or mattresses, the actual activity of the testing subjects can be
recorded with fidelity. So that a more credible and meaningful comparison between the classification result and reference data can be achieved.

Last, as the ultimate goal of this research project, this posture classification/fall detection system should be made beneficial to general consumer. There are some brilliant applications already presented by various researchers or manufacturers. For example, Tamura et al. introduced a wearable airbag incorporating a fall-detection system[91]. The idea was phenomenal but their fall detection algorithm suffered from high false-alarm rate. A more reliable fall detection algorithm such as ours may be integrated with such airbag or vest for guaranteed protection on senior users. Also, as a matter of fact, there are more and more commercialized smart activity trackers coming out in 2014. While current smart fitness device manufacturers such as Nike, Jawbone, and Fitbit are updating their old models to provide more functionalities and better performance, there are also many new companies trying to bring up new ideas for wearable systems. These novel platforms should be able to take advantage of the concept and algorithms introduced in this work.
Appendix 1 Matlab Code for the Preliminary Testing

clc;
clear all;
fis = readfis('logic engine.fis');
data = load('Test_data\TJ\walk(slow31)_5.txt');
data = data';
[m, n] = size(data);
axis = 1 : n;
a = data(4, 1 : n);
b = data(1, 1 : n);
c = data(2, 1 : n);
d = data(3, 1 : n);
acc_b = data(5 : 7, 1 : n);
x_b = acc_b(1, 1 : n);
y_b = acc_b(2, 1 : n);
z_b = acc_b(3, 1 : n);
acc_i = zeros(3, n);
x_i = zeros(1, n);
y_i = zeros(1, n);
z_i = zeros(1, n);
z_i_filtered = zeros(1, n);
MAX_SVM = zeros(1, n);
SVM = zeros(1, n);
SP = 1/200;
V_pre = 0;
H_pre = 0;
loco_counter = 0;
position = 2;
nw = 1 / SP;
threshold = 1;
VD_stand = 0;
V = zeros(1, n);
H = zeros(1, n);
posture = zeros(1, n);
SMA = zeros(1, n);
window_mark = zeros(1, n);
DC_offset = 0;
fall_flag = 0;
while (position + nw <= n)
    for i = position : position + nw - 1
        C(:, :, i) = [(a(i)^2 + b(i)^2 - c(i)^2 - d(i)^2) (2 * (b(i) * c(i) - a(i) * d(i))) (2 * (b(i) * d(i) + a(i) * c(i))); (2 * (b(i) * c(i) + a(i) * d(i))) (a(i)^2 - b(i)^2 + c(i)^2 - d(i)^2) (2 * (c(i) * d(i) - a(i) * b(i))); (2 * (b(i) * d(i) - a(i) * c(i))) (2 * (c(i) * d(i) + a(i) * b(i))) (a(i)^2 - b(i)^2 - c(i)^2 + d(i)^2)];
        acc_i(1 : 3, i) = C(:, :, i) * acc_b(1 : 3, i);
        x_i(i) = acc_i(1, i);
        y_i(i) = acc_i(2, i);
        z_i(i) = acc_i(3, i) - (-9.818);
        z_i_filtered(i) = z_i(i) - DC_offset;
        SVM(i) = sqrt(x_i(i)^2 + y_i(i)^2 + z_i_filtered(i)^2);
    end
MAX_SVM(position : position + nw - 1) = max(SVM(position : position + nw - 1));
SMA(position : position + nw - 1) = (sum(abs(x_i(position : position + nw - 1))) + sum(abs(y_i(position : position + nw - 1))) + sum(abs(z_i_filtered(position : position + nw - 1)))) / (nw * SP);
if (max(abs(z_i_filtered(position : position + nw - 1))) > threshold)
  for i = position : position + nw - 1
    if i == position
      V(i) = V_pre + 0.5 * SP * (z_i_filtered(i) + z_i_filtered(i - 1));
      H(i) = H_pre + 0.5 * SP * (V(i) + V(i - 1));
    else
      V(i) = V(i - 1) + 0.5 * SP * (z_i_filtered(i) + z_i_filtered(i - 1));
      H(i) = H(i - 1) + 0.5 * SP * (V(i) + V(i - 1));
      if H(i) > 1.2
        H(i) = 1.2;
      end
      if H(i) < -0.2
        H(i) = -0.2;
      end
    end
  end
V_pre = V(position + nw - 1);
H_pre = H(position + nw - 1);
else
  V_pre = 0;
  V(position : position + nw - 1) = V_pre;
  H(position : position + nw - 1) = H_pre;
end
posture(position : position + nw - 1) = round(evalfis([mean(x_b(position : position + nw - 1))
mean(H(position : position + nw - 1)) SMA(position)], fis));
if posture(position) == 1
  V_pre = 0;
  if fall_flag == 1
    posture(position : position + nw - 1) = 8;
    %V_pre = 0;
  end
  loco_counter = 0;
elseif posture(position) == 2
  V_pre = 0;
  H_pre = VD_stand;
  fall_flag = 0;
  loco_counter = 0;
elseif posture(position) == 3
  if fall_flag == 1
    posture(position : position + nw - 1) = 8;
  else
    V_pre = 0;
  end
  loco_counter = 0;
elseif posture(position) == 4
  loco_counter = loco_counter + 1;
  if loco_counter > 6
    posture(position : position + nw - 1) = 7;
    H_pre = VD_stand;
  end
elseif posture(position) == 5
    fall_flag = 1;
    loco_counter = loco_counter + 1;
    if loco_counter > 6
        posture(position : position + nw - 1) = 7;
        H_pre = VD_stand;
    end
elseif posture(position) == 6
    loco_counter = 0;
end
position = position + nw;
end
Appendix 2 Fuzzy Logic Core Using Matlab Fuzzy Logic Toolbox

[System]
Name='logic engine'
Type='mamdani'
Version=2.0
NumInputs=3
NumOutputs=1
NumRules=13
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='mom'

[Input1]
Name='MeanX_b'
Range=[-12 12]
NumMFs=3
MF1='Inverted':'trapmf',[-12 -11.6 -5.3 -4.9]
MF2='Horizontal':'trapmf',[-4.9 -4.5 4.5 4.9]
MF3='Vertical':'trapmf',[4.9 5.3 11.6 12]

[Input2]
Name='MeanH'
Range=[-0.4 1.4]
NumMFs=3
MF1='low':'trapmf',[0.8 0.85 1.35 1.4]
MF2='mid':'trapmf',[0.2 0.25 0.75 0.8]
MF3='high':'trapmf',[-0.4 -0.35 0.15 0.2]

[Input3]
Name='SMA'
Range=[0 3000]
NumMFs=3
MF1='low':'trapmf',[0 10 110 120]
MF2='mid':'trapmf',[120 130 1060 1070]
MF3='high':'trapmf',[1070 1080 2990 3000]

[Output1]
Name='Posture'
Range=[0 7]
NumMFs=6
MF1='sit': trimf, [0 1 2]
MF2='stand': trimf, [1 2 3]
MF3='lie': trimf, [2 3 4]
MF4='transition': trimf, [3 4 5]
MF5='Intense': trimf, [4 5 6]
MF6='loose': trimf, [5 6 7]

[Rules]
3 1 1, 1 (1) : 1
3 2 1, 1 (1) : 1
3 3 1, 2 (1) : 1
2 1 1, 3 (1) : 1
2 3 1, 3 (1) : 1
2 2 1, 3 (1) : 1
3 2 2, 4 (1) : 1
3 3 2, 4 (1) : 1
2 1 2, 3 (1) : 1
2 2 2, 3 (1) : 1
0 0 3, 5 (1) : 1
1 0 0, 6 (1) : 1
2 3 2, 3 (1) : 1
## Appendix 3 XBee 2 Wireless Transceiver Setting

<table>
<thead>
<tr>
<th></th>
<th>PC-end</th>
<th>Remote-end</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Modem type</strong></td>
<td>XB24-ZB</td>
<td>XB24-ZB</td>
</tr>
<tr>
<td><strong>Firmware</strong></td>
<td>Coordinator 20A0</td>
<td>Router AT 22A0</td>
</tr>
<tr>
<td><strong>ID</strong></td>
<td>1111</td>
<td>1111</td>
</tr>
<tr>
<td><strong>DH</strong></td>
<td>0013A200</td>
<td>0013A200</td>
</tr>
<tr>
<td><strong>DL</strong></td>
<td>407C456C</td>
<td>407A3936</td>
</tr>
<tr>
<td><strong>BD</strong></td>
<td>7(115200 bps)</td>
<td>5(38400 bps)</td>
</tr>
<tr>
<td><strong>JV</strong></td>
<td>n.a.</td>
<td>1</td>
</tr>
</tbody>
</table>
Appendix 4 Hardware Schematic of the Wearable System
Appendix 5 C Code for Posture Classification and Fall Detection for the Wearable System

//classification algorithm 1/
if (RMS_sq.f < Threshold1){
    counter = 0;
    counter_f = 0;
    if ((pitch_mean.f < 20) && (pitch_mean.f > -30)){
        Result_1.f = 2; //standing
    }    
    if ((pitch_mean.f > 20) && (pitch_mean.f < 65)){
        Result_1.f = 1; //sitting
    }    
    if ((pitch_mean.f < -30) || (pitch_mean.f > 65)){
        if (flag.f == 1){
            Result_1.f = 7; //fallen
            fall_mark = 1;
        } else{
            Result_1.f = 3; //lying
        }
    }    
} else{
    if ((pitch_mean.f < 20) && (pitch_mean.f > -30) && (fall_mark == 1)){
        flag.f = 0;
        fall_mark = 0;
    }    
    if (counter < Loco){
        counter = counter + 1;
    }
    if (RMS_sq.f > Threshold2){
        Result_1.f = 5; //falling
        flag.f = 1;
        counter_f = 1;
    } else{
        if ((counter_f > 0) && (counter_f < 300)){
            counter_f = counter_f + 1;
        } if (counter_f == 300){
            counter_f = 0;
        }
    }
}

110
flag.f = 0;
}
if (counter == Loco){
    Result_1.f = 6; //walking
}
else{
    Result_1.f = 4; //transition or walking
}
}

//classification algorithm 2/
if (RMS_sq.f < Threshold1){
    counter_f2 = 0;
    if (RMS_sq_last > Threshold1){
        if (counter_2 < Loco){
            V_pre = V.f;
            H_pre = H.f;
            V.f = V_pre + 0.5 * (Zi[26].f + Zi[25].f) / SP_freq;
            H2 = H_pre + 0.5 * (V.f + V_pre) / SP_freq;
            H.f = H2 - 0.5 * counter_2 * V.f / SP_freq;
            if (H.f > 1){
                H.f = 1;
            }
            if (H.f < 0){
                H.f = 0;
            }
        }
        if ((pitch_mean.f < 65) && (pitch_mean.f > -30)){
            if (H.f < 0.3){
                if (pitch_mean.f > 20){
                    Result_2.f = 1; //sitting
                    V.f = 0;
                }
                else{
                    Result_2.f = 2; //standing
                    H.f = 0;
                    V.f = 0;
                }
            }
            if ((H.f > 0.3) && (H.f < 0.7)){
                Result_2.f = 1; //sitting
                V.f = 0;
            }
            if (H.f > 0.7){
                if (flag_2.f == 1){

111
Result_2.f = 7; //fallen
fall_mark2 = 1;
H.f = Floor;
V.f = 0;
}
else{
    Result_2.f = 1; //sitting
    H.f = Floor;
    V.f = 0;
}
}
}
else{
    if ((H.f > 0.45) && (flag_2.f == 1)){
        Result_2.f = 7; //fallen
        fall_mark2 = 1;
        H.f = Floor;
        V.f = 0;
    }else{
        Result_2.f = 3; //lying
        H.f = Bed_H;
        V.f = 0;
    }
}
}
else{
    if (((pitch_mean.f < 20) && (pitch_mean.f > -30))){
        Result_2.f = 2; //standing
        H.f = 0;
        V.f = 0;
    }
    if (((pitch_mean.f > 20) && (pitch_mean.f < 65)){
        Result_2.f = 1; //sitting
        V.f = 0;
        H.f = Chair_H;
    }
    if (((pitch_mean.f < -30) || (pitch_mean.f > 65)){
        if (flag_2.f == 1){
            Result_2.f = 7; //fallen
            fall_mark2 = 1;
            V.f = 0;
            H.f = Floor;
        }
    }
}
else{
    Result_2.f = 3; //lying
    H.f = Bed_H;
    V.f = 0;
}
}
}
}
else{
    if ((pitch_mean.f < 65) && (pitch_mean.f > -30)){
        if (H.f < 0.3){
            if (pitch_mean.f > 20){
                Result_2.f = 1; //sitting
                V.f = 0;
            }else{
                Result_2.f = 2; //standing
                H.f = 0;
                V.f = 0;
            }
        }
        if ((H.f > 0.3) && (H.f < 0.7)){
            Result_2.f = 1; //sitting
            V.f = 0;
        }
        if (H.f > 0.7){
            if (flag_2.f == 1){
                Result_2.f = 7; //fallen
                fall_mark2 = 1;
                H.f = Floor;
                V.f = 0;
            }else{
                Result_2.f = 1; //sitting
                H.f = Floor;
                V.f = 0;
            }
        }
    }
    else{
        if ((H.f > 0.45) && (flag_2.f == 1)){
            Result_2.f = 7; //fallen
            fall_mark2 = 1;
            V.f = 0;
        }
    }
}
113
H.f = Floor;
}
else{
    Result_2.f = 3; //lying
    H.f = Bed_H;
    V.f = 0;
}
}

else{
    if ((pitch_mean.f < 20) && (pitch_mean.f > -30) && (fall_mark2 == 1)){
        flag_2.f = 0;
        fall_mark2 = 0;
    }
    V_pre = V.f;
    H_pre = H.f;
    V.f = V_pre + 0.5 * (Zi[26].f + Zi[25].f) / SP_freq;
    H.f = H_pre + 0.5 * (V.f + V_pre) / SP_freq;
    if (RMS_sq_last > Threshold1){
        if (counter_2 < Loco){
            counter_2 = counter_2 + 1;
        }
    }
    else{
        counter_2 = 1;
    }
}
if (RMS_sq.f > Threshold2){
    Result_2.f = 5; //falling
    flag_2.f = 1;
    counter_f2 = 1;
}
else{
    if ((counter_f2 > 0) && (counter_f2 < 300)){
        counter_f2 = counter_f2 + 1;
    }
    if (counter_f2 == 300){
        counter_f2 = 0;
        flag_2.f = 0;
    }
}
if (counter_2 == Loco){
    Result_2.f = 6; //walking
}
else{
Result_2.f = 4; //transition or walking
Appendix 6 Matlab Code for the Testing on Case Students

clc;
clear all;
load('20\20_5.mat');
load('20\20_ref_5.mat');
Total = 0;
Correct1 = 0;
Correct2 = 0;
[m,n]=size(ref);
Correct1_c = 0;
Correct2_c = 0;

if ref(1) ~= 0
    if ref(1) == Result1(1)
        Correct1 = Correct1 + 1;
    end
    if ref(1) == Result2(1)
        Correct2 = Correct2 + 1;
    end
    if (ref(1) == Result1(1)) || (ref(1) == Result1(2))
        Correct1_c = Correct1_c + 1;
    end
    if (ref(1) == Result2(1)) || (ref(1) == Result2(2))
        Correct2_c = Correct2_c + 1;
    end
    Total = Total + 1;
end
for i = 2 : n
    if ref(i) ~= 0
        if ref(i) == Result1(i)
            Correct1 = Correct1 + 1;
        end
        if ref(i) == Result2(i)
            Correct2 = Correct2 + 1;
        end
        if (ref(i) == Result1(i)) || (ref(i) == Result1(i - 1)) || (ref(i) == Result1(i + 1))
            Correct1_c = Correct1_c + 1;
        end
        if (ref(i) == Result2(i)) || (ref(i) == Result2(i - 1)) || (ref(i) == Result2(i + 1))
            Correct2_c = Correct2_c + 1;
        end
        Total = Total + 1;
    end
end

116
end
end

Accuracy1 = Correct1 / Total;
Accuracy2 = Correct2 / Total;
Accuracy1_corrected = Correct1_c / Total;
Accuracy2_corrected = Correct2_c / Total;
Appendix 7 Multi-Factorial Fall Risk Assessment Tool (MAHC-10)

**MAHC 10 - Fall Risk Assessment Tool**

Click [here](#) to review the Validation Study of the Missouri Alliance for Home Care’s fall risk assessment tool.

Conduct a fall risk assessment on each patient at start of care and re-certification.

<table>
<thead>
<tr>
<th>Patient Name:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Circle one) SOC or Re-certification</td>
<td>Date:</td>
</tr>
</tbody>
</table>

### Required Core Elements

Assess one point for each core element "yes".

Information may be gathered from medical record, assessment and if applicable, the patient/caregiver. Beyond protocols listed below, scoring should be based on your clinical judgment.

<table>
<thead>
<tr>
<th>Points</th>
<th></th>
</tr>
</thead>
</table>

| Age 65+ |  |
| Diagnosis (3 or more co-existing) Includes only documented medical diagnosis |  |
| Prior history of falls within 3 months An unintentional change in position resulting in coming to rest on the ground or at a lower level |  |
| Incontinence Inability to make it to the bathroom or commode in timely manner Includes frequency, urgency, and/or nocturia. |  |
| Visual Impairment Includes but not limited to, macular degeneration, diabetic retinopathy, visual field loss, age related changes, decline in visual acuity, accommodation, glare tolerance, depth perception, and night vision or not wearing prescribed glasses or having the correct prescription. |  |
| Impaired functional mobility May include patients who need help with IADLS or ADLS or have gait or transfers problems, arthritis, pain, fear of falling, foot problems, impaired sensation, impaired coordination or improper use of assistive devices. |  |
| Environmental hazards May include but not limited to, poor illumination, equipment tubing, inappropriate footwear, pets, hard to reach items, floor surfaces that are uneven or cluttered, or outdoor entry and exits. |  |
| Poly Pharmacy (4 or more prescriptions – any type) All PRESCRIPTIONS including prescriptions for OTC meds. Drugs highly associated with fall risk include but not limited to, sedatives, anti-depressants, tranquilizers, narcotics, antihypertensives, cardiac meds, corticosteroids, anti-anxiety drugs, anticholinergic drugs, and hypoglycemic drugs. |  |
| Pain affecting level of function Pain often affects an individual’s desire or ability to move or pain can be a factor in depression or compliance with safety recommendations. |  |
| Cognitive impairment Could include patients with dementia, Alzheimer’s or stroke patients or patients who are confused, use poor judgment, have decreased comprehension, impulsivity, memory deficits. Consider patients ability to adhere to the plan of care. |  |

A score of 4 or more is considered at risk for falling

<table>
<thead>
<tr>
<th>Total</th>
<th></th>
</tr>
</thead>
</table>

Clinician’s signature

---

**Missouri Alliance for HOME CARE**

2420 Hyde Park, Suite A,St. Louis City, MO 63105-4731 • (733) 634-7772 • (733) 634-4374 Fax

Want resources to reduce your falls rate & compare yourself with other home care agencies? Join MAHC's Falls Reduction Benchmark Project – contact us today for more information!
## Appendix 8 Collected Data from Senior Subjects

<table>
<thead>
<tr>
<th>Subject</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Steps</strong></td>
<td>1742</td>
<td>3352</td>
<td>518</td>
<td>4134</td>
<td>7740</td>
<td>357</td>
<td>4725</td>
</tr>
<tr>
<td><strong>Distance (km)</strong></td>
<td>1.38</td>
<td>2.79</td>
<td>0.41</td>
<td>3.35</td>
<td>6.3</td>
<td>0.27</td>
<td>3.77</td>
</tr>
<tr>
<td><strong>Active Time (min)</strong></td>
<td>16</td>
<td>29</td>
<td>4</td>
<td>37</td>
<td>69</td>
<td>3</td>
<td>43</td>
</tr>
<tr>
<td><strong>Total Burn (cal)</strong></td>
<td>2004</td>
<td>2088</td>
<td>1944</td>
<td>2124</td>
<td>2291</td>
<td>1935</td>
<td>2143</td>
</tr>
<tr>
<td><strong>Longest Active (min)</strong></td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>19</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td><strong>Active Burn (min)</strong></td>
<td>106</td>
<td>208</td>
<td>31</td>
<td>255</td>
<td>465</td>
<td>21</td>
<td>282</td>
</tr>
<tr>
<td><strong>Longest Idle (min)</strong></td>
<td>91</td>
<td>108</td>
<td>278</td>
<td>72</td>
<td>71</td>
<td>196</td>
<td>68</td>
</tr>
<tr>
<td><strong>Resting Burn (cal)</strong></td>
<td>1898</td>
<td>1880</td>
<td>1913</td>
<td>1869</td>
<td>1826</td>
<td>1914</td>
<td>1861</td>
</tr>
<tr>
<td><strong>Calories</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Steps</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Hours Worn</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Steps</strong></td>
<td>3538</td>
<td>4321</td>
<td>1230</td>
<td>6067</td>
<td>10039</td>
<td>1800</td>
<td>5886</td>
</tr>
<tr>
<td><strong>Calories</strong></td>
<td>2314</td>
<td>2395</td>
<td>2665</td>
<td>2500</td>
<td>2872</td>
<td>2665</td>
<td>2455</td>
</tr>
<tr>
<td><strong>Distance (miles)</strong></td>
<td>1.67</td>
<td>2.04</td>
<td>0.58</td>
<td>2.86</td>
<td>4.73</td>
<td>0.85</td>
<td>2.78</td>
</tr>
<tr>
<td><strong>Distance (km)</strong></td>
<td>2.672</td>
<td>3.264</td>
<td>0.928</td>
<td>4.576</td>
<td>7.568</td>
<td>1.36</td>
<td>4.448</td>
</tr>
<tr>
<td><strong>Very Active Minutes</strong></td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>26</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td><strong>Recording Minutes</strong></td>
<td>1216</td>
<td>428</td>
<td>1256</td>
<td>140</td>
<td>1280</td>
<td>524</td>
<td>1283</td>
</tr>
<tr>
<td><strong>Wearing Minutes</strong></td>
<td>780</td>
<td>600</td>
<td>720</td>
<td>825</td>
<td>870</td>
<td>675</td>
<td>840</td>
</tr>
<tr>
<td><strong>Total Active Alg. 1</strong></td>
<td>43</td>
<td>22</td>
<td>49</td>
<td>2</td>
<td>815</td>
<td>16</td>
<td>96</td>
</tr>
<tr>
<td><strong>Total Inactive Alg. 1</strong></td>
<td>737</td>
<td>406</td>
<td>671</td>
<td>138</td>
<td>55</td>
<td>508</td>
<td>744</td>
</tr>
<tr>
<td><strong>Total Active Alg. 2</strong></td>
<td>42</td>
<td>20</td>
<td>49</td>
<td>2</td>
<td>101</td>
<td>16</td>
<td>96</td>
</tr>
<tr>
<td><strong>Total Inactive Alg. 2</strong></td>
<td>738</td>
<td>408</td>
<td>671</td>
<td>138</td>
<td>769</td>
<td>508</td>
<td>744</td>
</tr>
<tr>
<td>Subject</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td><strong>Jawbone UP</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Steps</td>
<td>3089</td>
<td>6354</td>
<td>4170</td>
<td>11020</td>
<td>2970</td>
<td>462</td>
<td></td>
</tr>
<tr>
<td>Distance(km)</td>
<td>2.26</td>
<td>5.26</td>
<td>3.38</td>
<td>9.05</td>
<td>2.3</td>
<td>0.38</td>
<td></td>
</tr>
<tr>
<td>Active Time(min)</td>
<td>32</td>
<td>55</td>
<td>37</td>
<td>97</td>
<td>29</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Total Burn(cal)</td>
<td>2057</td>
<td>2240</td>
<td>2112</td>
<td>2465</td>
<td>2057</td>
<td>1943</td>
<td></td>
</tr>
<tr>
<td>Longest Active(min)</td>
<td>14</td>
<td>16</td>
<td>16</td>
<td>29</td>
<td>11</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Active Burn(min)</td>
<td>181</td>
<td>395</td>
<td>243</td>
<td>676</td>
<td>177</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>Longest Idle(min)</td>
<td>119</td>
<td>128</td>
<td>167</td>
<td>75</td>
<td>199</td>
<td>225</td>
<td></td>
</tr>
<tr>
<td>Resting Burn(cal)</td>
<td>1876</td>
<td>1845</td>
<td>1869</td>
<td>1789</td>
<td>1880</td>
<td>1914</td>
<td></td>
</tr>
<tr>
<td><strong>Nike+ Fuelband</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calories</td>
<td>273</td>
<td>220</td>
<td>297</td>
<td>633</td>
<td>187</td>
<td>242</td>
<td></td>
</tr>
<tr>
<td>Steps</td>
<td>1832</td>
<td>2709</td>
<td>3105</td>
<td>5998</td>
<td>925</td>
<td>604</td>
<td></td>
</tr>
<tr>
<td>Hours Worn</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>9</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td><strong>Fitbit Flex</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Steps</td>
<td>4041</td>
<td>5137</td>
<td>5319</td>
<td>12102</td>
<td>4047</td>
<td>1450</td>
<td></td>
</tr>
<tr>
<td>Calories</td>
<td>2249</td>
<td>2424</td>
<td>2349</td>
<td>2886</td>
<td>2216</td>
<td>2665</td>
<td></td>
</tr>
<tr>
<td>Distance(miles)</td>
<td>1.91</td>
<td>2.42</td>
<td>2.51</td>
<td>5.71</td>
<td>1.91</td>
<td>0.68</td>
<td></td>
</tr>
<tr>
<td>Distance(km)</td>
<td>3.056</td>
<td>3.872</td>
<td>4.016</td>
<td>9.136</td>
<td>3.056</td>
<td>1.088</td>
<td></td>
</tr>
<tr>
<td>Very Active Minutes</td>
<td>10</td>
<td>6</td>
<td>26</td>
<td>61</td>
<td>19</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td><strong>Our Activity Classifiers</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recording Minutes</td>
<td>752</td>
<td>1247</td>
<td>1228</td>
<td>144</td>
<td>685</td>
<td>1181</td>
<td></td>
</tr>
<tr>
<td>Wearing Minutes</td>
<td>705</td>
<td>570</td>
<td>765</td>
<td>780</td>
<td>700</td>
<td>810</td>
<td></td>
</tr>
<tr>
<td>Total Active Alg. 1</td>
<td>53</td>
<td>52</td>
<td>77</td>
<td>40</td>
<td>44</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>Total Inactive Alg. 1</td>
<td>652</td>
<td>518</td>
<td>688</td>
<td>104</td>
<td>641</td>
<td>776</td>
<td></td>
</tr>
<tr>
<td>Total Active Alg. 2</td>
<td>54</td>
<td>51</td>
<td>77</td>
<td>42</td>
<td>44</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>Total Inactive Alg. 2</td>
<td>651</td>
<td>519</td>
<td>688</td>
<td>102</td>
<td>641</td>
<td>776</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix 9 Collected Data of MAHC Score and Basic Information from Senior Subjects

<table>
<thead>
<tr>
<th>Subject#</th>
<th>Age</th>
<th>Height (cm)</th>
<th>Weight (kg)</th>
<th>MAHC score</th>
<th>Fall Risk</th>
<th>Alg. 1 (%)</th>
<th>Alg. 2 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>72</td>
<td>160.0</td>
<td>63.5</td>
<td>2</td>
<td>Low</td>
<td>43.3</td>
<td>59.2</td>
</tr>
<tr>
<td>3</td>
<td>78</td>
<td>147.0</td>
<td>63.5</td>
<td>4</td>
<td>High</td>
<td>76.4</td>
<td>76.2</td>
</tr>
<tr>
<td>4</td>
<td>86</td>
<td>175.0</td>
<td>77.1</td>
<td>4</td>
<td>High</td>
<td>73.3</td>
<td>73.5</td>
</tr>
<tr>
<td>5</td>
<td>85</td>
<td>143.0</td>
<td>60.3</td>
<td>4</td>
<td>High</td>
<td>36.4</td>
<td>35.7</td>
</tr>
<tr>
<td>6</td>
<td>85</td>
<td>174.0</td>
<td>74.8</td>
<td>2</td>
<td>Low</td>
<td>53.7</td>
<td>53.1</td>
</tr>
<tr>
<td>7</td>
<td>86</td>
<td>155.0</td>
<td>59.0</td>
<td>4</td>
<td>High</td>
<td>84.5</td>
<td>84.5</td>
</tr>
<tr>
<td>8</td>
<td>79</td>
<td>162.5</td>
<td>65.8</td>
<td>4</td>
<td>High</td>
<td>47.7</td>
<td>48.1</td>
</tr>
<tr>
<td>9</td>
<td>89</td>
<td>188.0</td>
<td>77.1</td>
<td>4</td>
<td>High</td>
<td>48.8</td>
<td>66.0</td>
</tr>
<tr>
<td>10</td>
<td>85</td>
<td>144.5</td>
<td>61.2</td>
<td>4</td>
<td>High</td>
<td>64.7</td>
<td>64.6</td>
</tr>
<tr>
<td>11</td>
<td>77</td>
<td>162.5</td>
<td>53.1</td>
<td>3</td>
<td>Low</td>
<td>50.1</td>
<td>49.8</td>
</tr>
<tr>
<td>12</td>
<td>79</td>
<td>162.5</td>
<td>62.1</td>
<td>4</td>
<td>High</td>
<td>75.7</td>
<td>71.5</td>
</tr>
<tr>
<td>13</td>
<td>89</td>
<td>170.0</td>
<td>74.8</td>
<td>4</td>
<td>High</td>
<td>67.0</td>
<td>66.9</td>
</tr>
<tr>
<td>14</td>
<td>92</td>
<td>183.0</td>
<td>93.0</td>
<td>7</td>
<td>High</td>
<td>58.1</td>
<td>61.2</td>
</tr>
</tbody>
</table>
## Appendix 10 Collected Data of SUS from Senior Subjects

<table>
<thead>
<tr>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
<th>Q10</th>
<th>Score</th>
<th>Q11</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td></td>
<td>72.5</td>
<td>150</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td></td>
<td>70</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td>67.5</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
<td>60</td>
<td>100</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>37.5</td>
<td>30</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td>60</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td></td>
<td>37.5</td>
<td>100</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td></td>
<td>60</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
<td>60</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td></td>
<td>37.5</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Score</th>
<th>Q11 (USD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>72.5</td>
<td>150 100 0 100 0 30 0 100 75</td>
</tr>
<tr>
<td>70</td>
<td>100 0 100 0 30 0 100 75</td>
</tr>
<tr>
<td>67.5</td>
<td>500 500 500 500 500 500 500 500</td>
</tr>
<tr>
<td>37.5</td>
<td>60 60 60 60 60 60 60 60</td>
</tr>
<tr>
<td>60</td>
<td>80 80 80 80 80 80 80 80</td>
</tr>
<tr>
<td>72.5</td>
<td>77.5 77.5</td>
</tr>
</tbody>
</table>

### Subject Data

<table>
<thead>
<tr>
<th>Subject</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2.1</td>
</tr>
<tr>
<td>Q2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2.0</td>
</tr>
<tr>
<td>Q3</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3.3</td>
</tr>
<tr>
<td>Q4</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4.4</td>
</tr>
<tr>
<td>Q5</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1.7</td>
</tr>
<tr>
<td>Q6</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1.9</td>
</tr>
<tr>
<td>Q7</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2.6</td>
</tr>
<tr>
<td>Q8</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>3.3</td>
</tr>
<tr>
<td>Q9</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4.6</td>
</tr>
<tr>
<td>Q10</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>2.8</td>
</tr>
<tr>
<td>Score</td>
<td>55</td>
<td>60</td>
<td>65</td>
<td>82.5</td>
<td>66.4</td>
</tr>
<tr>
<td>Q11 (USD)</td>
<td>40</td>
<td>40</td>
<td>500</td>
<td>25</td>
<td>89.2</td>
</tr>
</tbody>
</table>
References


[64] N. Bidargaddi, A. Sarela, L. Klingbeil, and M. Karunanithi, "Detecting walking activity in cardiac rehabilitation by using accelerometer," in Intelligent Sensors,


