ROBUST SIGNAL PROCESSING TECHNIQUES FOR WEARABLE INERTIAL MEASUREMENT UNIT (IMU) SENSORS

A Dissertation

by

JIAN WU

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Roozbeh Jafari
Committee Members,	Tracy Hammond
	Sung Il Park
	Radu Stoleru
Head of Department,	Dilma Da Silva

December 2018

Major Subject: Computer Engineering

ABSTRACT

Activity and gesture recognition using wearable motion sensors, also known as inertial measurement units (IMUs), provides important context for many ubiquitous sensing applications including healthcare monitoring, human computer interface and context-aware smart homes and offices. Such systems are gaining popularity due to their minimal cost and ability to provide sensing functionality at any time and place. However, several factors can affect the system performance such as sensor location and orientation displacement, activity and gesture inconsistency, movement speed variation and lack of tiny motion information.

This research is focused on developing signal processing solutions to ensure the system robustness with respect to these factors. Firstly, for existing systems which have already been designed to work with certain sensor orientation/location, this research proposes opportunistic calibration algorithms leveraging camera information from the environment to ensure the system performs correctly despite location or orientation displacement of the sensors. The calibration algorithms do not require extra effort from the users and the calibration is done seamlessly when the users present in front of an environmental camera and perform arbitrary movements. Secondly, an orientation independent and speed independent approach is proposed and studied by exploring a novel orientation independent feature set and by intelligently selecting only the relevant and consistent portions of various activities and gestures. Thirdly, in order to address the challenge that the IMU is not able capture tiny motion which is important to some applications, a sensor fusion framework is proposed to fuse the complementary sensor

modality in order to enhance the system performance and robustness. For example, American Sign Language has a large vocabulary of signs and a recognition system solely based on IMU sensors would not perform very well. In order to demonstrate the feasibility of sensor fusion techniques, a robust real-time American Sign Language recognition approach is developed using wrist worn IMU and surface electromyography (EMG) sensors.

DEDICATION

To my loving and supporting family, especially my parents Guoqing Wu and Maiju Jin, and my wife Hui Xu.

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Jafari, and my committee members, Dr. Hammond, Dr. Park and Dr. Stoleru, for their guidance and support throughout the course of this research.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professor Jafari, [advisor] and Professors Hammond and Stoleru of the Department of Computer Science and Engineering and Professor Park of the Department of Electrical and Computer Engineering.

All work for the dissertation was completed independently by the student.

Funding Sources

Graduate study was supported by a research assistantship from Texas A&M University. This work was also made possible in part by the National Science Foundation, under grants CNS-1150079, CNS-1012975, ECCS-1509063 and EEC-1648451, the National Institute of Health, under grant R15AG037971 and the TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding organizations.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

1.1 Objective

The objective of this dissertation is to develop robust wearable Inertial Measurement Unit (IMU) sensors based activity/gesture recognition systems. The robustness of the systems refers to that the system works with arbitrary sensor orientation, works with different activity/gesture speed, works robustly with sensor location displacement and works for certain applications that may not be realized solely by IMU sensors.

1.2 Significance

Human activity/gesture recognition aims to recognize the actions and body gestures of from a series of observations on the human activities/gestures and the environmental conditions. Since the 1980s, this research topic has attracted the attention of several computer science communities due to its strength in providing useful context for many applications and its connection to many different fields of study such as healthcare, human-computer interaction and sociology [1].

Human activity recognition system is widely employed in healthcare systems which are usually installed in homes, hospitals and rehabilitation centers. It is an important component in rehabilitation centers to monitor the activities of elderly people for chronic disease management and disease prevention [2, 3]. It is used to encourage physical exercises in rehabilitation centers for children with motor disabilities [4], to help patients with motor recovery after stroke [5] and to help patients with dysfunction and psychomotor slowing [6]. The human activity recognition system is also integrated

into homes to monitor the daily activities of patients to aid in obesity prevention and treatment [7]. This technique is also used to monitor other behaviors that are related to human health and help the clinicians perform effective interventions. For example, the monitoring of abnormal behavior of cardiac patients [8], the detection of early signs of illness [9] and estimation of stereotypical motion conditions in children with Autism Spectrum Disorders (ASD) [10]. Thus, human activity recognition is of significant benefit particularly in the United States where in 2015, healthcare expenditure was about $3.2 trillion (*i.e.* $9,900-per-person), which accounted for 17.8% of the GDP [11].

In the field of human-computer interaction, activity/gesture recognition has also received increasing attention. First, human activity recognition is widely applied in gaming, such as Kinect [12], Nintendo Wii [13], full body motion based games for adults [14]. Secondly, the hand is extensively used for gesturing compared with other body parts because it is a natural medium for communication between humans and thus the most suitable tool for HCI [15]. As Internet-of-Things (IoT) is gaining attraction, potentially a large number of smart devices are being offered to the consumers, such as smart appliances and entertainment devices in homes. The hand gesture interface will provide the most suitable solution to interfacing with these devices which makes the gesture recognition more significant.

Human activity/recognition can also benefit the field of sociology. One example is the surveillance systems that monitor the public safety and prevent crimes and dangerous activities. A human activity prediction system is introduced to detect the early stage of an activity and to predict the intended activities of the human before the activity

is fully executed [16]. Video surveillance is introduced as an effective tool for today's businesses in security surveillance, production monitoring, and deterring predatory and purloining behaviors [17].

Two main sensing technologies are used for human activity/gesture recognition: camera sensors and wearable IMU sensors [18]. The sensing technology captures the raw sensor information from the human body and the signal processing algorithms process the sensor input and recognize the activities/gestures using intelligent models. The research of human activity/gesture recognition starts with camera-based systems [19-21]. And the camera-based systems have been extensively studied and explored. However, there are some limitations of camera-based systems. The camera-based techniques typically require cameras mounted in the environment which inherently suffer from a limited range of vision. Further, the required infrastructure may not be available at all the desired locations or may be too expensive to implement. Issues associated with users' privacy also limit the utility of vision-based techniques. Moreover, it is very computationally expensive to analyze the image sequence in order to detect a certain activity/gesture. In contrast, the wearable IMU based systems are gaining popularity due to its minimum cost, ubiquitous nature and ability to provide sensing opportunities at any time and place.

The wearable computers are smart electronic devices that can be incorporated into clothing or worn on the body as implants or accessories. The wearable computers have seen a large increase in recent times in different applications, such as healthcare, sports training and entertainment. The global wearable market is expected to exceed

more than US$ 51.60 Billion by 2022 at a CAGR of 15.51% from 2016 [22]. Different types of sensors are embedded into wearable computers which include optical sensors, IMU sensors, and pressure sensors and so on. Among all these sensors, IMU sensors are the most common and popular one due to its low cost and low power consumption. The commodity of wearable IMU sensors makes the wearable IMUs based activity/gesture recognition significantly important. However, there are several practical challenges that affect the recognition accuracy of wearable IMU based systems: sensor location and orientation displacement, activity and gesture inconsistency, movement speed variation and lack of tiny motion information.

As introduced so far, the activity/gesture recognition is significantly important in various applications and the wearable IMU based solutions have obvious advantages and are more popular compared to camera-based solutions. This thesis aims to address the challenges associated with IMU based activity/gesture recognition and developing robust signal processing techniques that will ensure the systems will be applied widely in reality.

1.3 Technical Challenges

In this section, the technical challenges are introduced in detail. There are four major challenges associated with wearable IMU based activity/gesture recognition: sensor orientation displacement, sensor location displacement, speed variation and lack of tiny motion information. In order to develop robust signal processing techniques, these challenges should be addressed carefully.

1.3.1 Sensor Orientation Displacement

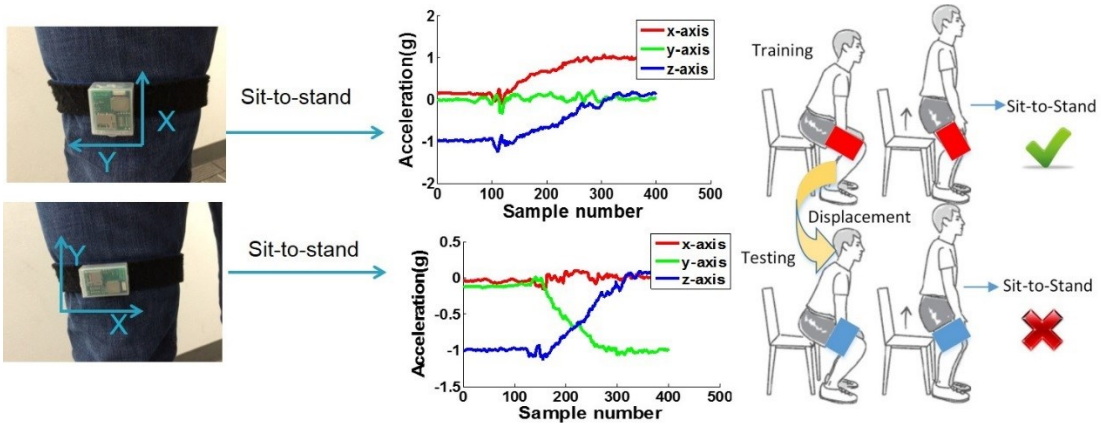


Figure 1.1 - 3-axis accelerometer signals for sit-to-stand with different sensor orientation.

When sensors are attached to the human body, the sensor orientation impacts sensor readings. If the sensors are attached at different orientations, the directions of all three axes will be different. As a result, for the same movement, the signals along each axis will be different. Therefore, the accuracy of the classification algorithms which use the features derived from each axis will be dramatically affected. Figure 1.1 shows the 3-axis acceleration for the same movement (sit-to-stand) with different sensor orientations. The signals are totally different for the different sensor orientations. If the system is trained with the sensor orientation as shown in the top half of the figure and is tested with the sensor attached at a different orientation as shown in the bottom half of the figure, the system is not going to recognize the same movement sit-to-stand.

The impact of the sensor orientation displacement on a sample activity recognition algorithm called dynamic time warping is discussed in [23]. Their results show that if the orientation displacement increase from 0 degrees to 20 degrees, the accuracies for several

daily activities decrease from 100% to 0%. This challenge should be carefully considered when we develop the activity recognition algorithms since accidental orientation displacement will happen during the user's movement or the users may not wear the sensor in the same orientation every time.

1.3.2 Sensor Location Displacement

Similar to sensor orientation displacement, users may wear the sensor at any location on the body due to their preference or the location of the sensor may change due to human motion. There are two different types of sensor location displacement. The first type is the displacement between different body segments. For example, the sensor should be worn at the wrist as required by the algorithm but is attached to the forearm by the user. Since different body segments have different movement patterns, the activity/gesture recognition algorithms will not perform correctly if this type of sensor location displacement happens. The second type of location displacement is the location displacement on the same body link. For example, when the sensor is supposed to be attached to the lower arm, it is considered as a displacement when the sensor is attached close to the wrist or close to the elbow. The angular velocity (gyroscope readings) is the same for sensors at different locations while the accelerations (accelerometer readings) are different for sensors at different locations on the same body link. This is because the same rigid body link will have the same rotation velocity for the same movement regardless of the muscle movement. However, for the acceleration, since the sensors move different distances in the same time duration, the acceleration will be different. This challenge should also be considered during the design of robust recognition algorithms.

1.3.3 Movement Speed Variation

Speed variation is another challenge when the activity recognition algorithm is developed. In real life, people will perform the same movement at different speeds in different scenarios. For example, if the user is in a hurry, he will stand fast and walk fast. On the other hand, if the user feels tired or is in a very relaxed state, he tends to act slowly. The speed variation is extensively studied by vision-based approaches [24-26]. However, it is not widely studied in wearable IMU based activity/gesture recognition systems. For wearable IMUs, different speed will result in a difference for the measured motion signals. In Figure 1.2, we show the magnitudes of acceleration and angular velocity for different speeds of the sit-to-stand movement. The magnitude is the root square sum of the readings along the three axes. The figure on the left side shows the magnitude of the acceleration and the one at the right side shows the magnitude of the angular velocity. We can see a clear difference between the two signals in each plot due to the different speeds. Thus, the signal processing algorithm should be able to handle these speed variations.

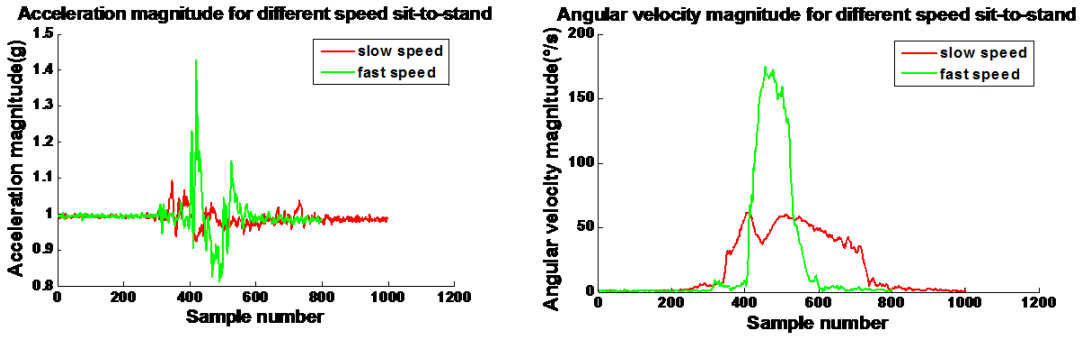


Figure 1.2 - Magnitude of acceleration (left) and angular velocity (right) for different speeds of sit-to-stand.

1.3.4 Lack of Tiny Movement Information

The IMU sensors are good at capturing visible movement and are not suitable to detect and recognize some tiny movements (*e.g.* muscle activity). However, in some applications, the tiny moments provide very important information to distinguish different activities/gestures. For example, in American Sign Language recognition, the muscle activity is also important since there are some signs that have exactly the same hand/arm motion pattern but have different muscle activity. For sign language recognition systems, the wrist-worn IMU sensor is good at capturing hand orientations and hand and arm movements while sEMG does well in distinguishing different hand shapes and finger movements when the sensors are placed on the forearm. Thus, they each have their own advantages in capturing different information about a sign. The fusion of these two complementary modalities will enhance the performance of an SLR system and thus enable the recognition of a large number of signs. The comparison of accelerometer based and sEMG based gesture recognition systems is discussed [27]. It is suggested that accelerometer and sEMG sensors are good at capturing different information of gestures and the performance enhancement combining these two modalities has been studied. The experiments show that 5% - 10% accuracy improvement is obtained by fusing these two modalities [28]. This challenge should be carefully considered and addressed when designing a recognition system for such applications.

1.4 Proposed Approaches

In order to address the technical challenges, this dissertation proposes four different approaches to solve the challenges from different perspectives. Firstly, the design and development of most of the existing systems do not consider the sensor orientation displacement and they are not able to function correctly when a sensor orientation displacement happens. In order to help these systems to work correctly despite orientation displacement of sensors, an opportunistic sensor orientation displacement calibration algorithm is proposed leveraging the environmental camera information. The sensor orientation displacement is calibrated seamlessly when the user presents in front of an environmental camera and perform arbitrary movements. Secondly, similar to the first approach, for the existing systems that have been designed to work with fixed sensor location, an opportunistic sensor location displacement calibration algorithm is discussed leveraging the environmental camera information. Thirdly, in order to address the sensor orientation displacement and speed variation challenges, a robust orientation independent and speed independent activity/gesture recognition algorithm is proposed that works with arbitrary sensor orientation and different movement speed. This general algorithm could be adopted as a reference design for future wearable IMUs based activity/gesture recognition. Fourthly, in order to address the challenge of lack of tiny movement information, a sensor fusion enhanced framework is proposed for an example application: American Sign Language (ASL) recognition. In this application, IMU is good at capturing hand and arm gestures but is not able to capture the hand shape and finger movements. Surface Electromyography

(EMG) provides the ability to capture this information by detecting muscle activities as a complementary modality. The fusion of these two modalities enhances the system performance significantly.

1.5 Innovations

In this section, the overview of the current research which is related to our proposed approaches is discussed. The unmet needs are pointed out for each category of our work and the innovations of each approach are listed.

1.5.1 Sensor Displacement Calibration

The first category of work that addresses the sensor orientation/location displacement challenges is to calibrate the sensor displacement time by time so that the system will function correctly even if the displacement happens. However, most of the works require the user to perform certain activities in order to perform the calibration [5, [29, 30]. For example, in one of the representative existing works, the authors defined a global frame by asking the user to perform symmetric forward/backward movements (*e.g.,* walking) [30]. The accelerometer signals are projected to this global frame such that the sensor orientation is calibrated to this global frame. The disadvantage of this type of work is that it requires the effort from the user and it limits the ubiquitous sensing ability of the activity/gesture recognition system. In order to fill this unmet need, we propose zero-effort opportunistic calibration algorithms to calibrate the sensor orientation/location displacement in this thesis. The proposed method does not require zero effort from the user and the calibration is done seamlessly when the user presents in front of the camera and performs arbitrary movements. The innovations of the

opportunistic sensor orientation and location displacement calibrations are listed as follows:

1. No user effort is required and the calibration is done seamless leveraging environmental sensors.
2. The calibration can be performed by arbitrary movement of the user.
3. The calibration algorithms can be easily plugged into the existing systems to ensure the existing systems will still work if sensor orientation/location displacement happens.
4. Extraction of useful matching information from two different modalities considering noise from both modalities is described.
5. A two-step search algorithm is proposed in sensor orientation displacement work.
6. A cascade classifier is proposed to determine the location of sensors considering effect of noise from both IMU and camera.

1.5.2 Orientation Independent Activity/Gesture Recognition

In order to address the sensor orientation displacement and to make the system work with arbitrary sensor orientation, several research works have been studied [31, 32]. The relative energy distribution over five different parts of the body is investigated to classify four different activities [31]. This approach performs well to distinguish dynamic activities that have different energy distribution on different body parts. However, for those activities that have similar energy distribution on different parts, (*e.g.*, walking and running), it may not exhibit acceptable performance. In another study, features in frequency domain are used to distinguish the periodic dynamic activities by

analyzing the periodicity between different movements [32]. However, the frequency resolution is a concern when identifying the difference between activities that have similar periodicity. Moreover, these features are not suitable to detect short term transitional movements (*e.g.*, sit-to-stand, stand-to-sit). Thus, the unmet need is that a general algorithm is needed to work with arbitrary sensor orientation and for different types of activities. In this thesis, we bridge this gap by proposing an orientation-independent speed independent activity/gesture recognition algorithm which is described in Section 6. The innovations of this approach are listed as follows:

1. Propose an orientation independent, speed independent activity/gesture recognition algorithm.
2. Propose a novel feature set which is orientation-independent and speed-independent.
3. Propose one general algorithm works with different types of activities, works with arbitrary sensor orientation and works for both gesture and activity recognition.
4. A template refinement technique is applied to determine the consistent segments of a movement and the inconsistent segments are eliminated using a variation of DTW called star-padding.

1.5.3 Sensor Fusion Enhanced System

IMU is good at capturing visible motion using accelerometer and gyroscope, however, it is not capable of capturing tiny movement (*e.g.* muscle activities). For certain application, both visible gesture/activity and tiny movement are very important.

One example is the Sign Language recognition. There are some signs that have exactly the same motion signature while the muscle activities are different. In order to recognize such gesture, the complementary sensor modality is required, and the sensor fusion framework is required. There are works that explore the fusion of IMU and sEMG for Sign Language recognition [33, 34]. However, their technique is very power consuming and is lack of auto-segmentation. This prevent applying the fusion technique in practical use for wearable system. In order to bridge this gap, the first American Sign Language recognition system is proposed fusing IMU and sEMG sensors in this thesis. The innovations of this work are described as follows:

1. The first time a sensor fusion framework is proposed for American Sign Language (ASL) recognition fusing wearable IMU sensors and surface EMG sensors.
2. An auto-segmentation technique based on sEMG is proposed and performance improvement of ASL recognition compared to using only the IMUs is demonstrated.
3. The best feature subset is selected from a broad range of features using information gain criterion and the selected features from different modalities (*e.g.* accelerometer, gyroscope and 4-channel sEMG) are discussed.

1.6 Organization of this Dissertation

The rest of this dissertation document is organized as follows: Section 2 details the necessary background information for the theoretical concepts and hardware systems

used during the course of this research. Section 3 lays out the various published related works to give a sense of the current status of research contributions to the problems being studied and how it contrasts with the contributions of this dissertation. Section 4 describes the research looking into the opportunistic sensor orientation calibration, whereas Section 5 describes opportunistic sensor location calibration. Section 6 describes the research that develops a robust orientation-independent speed-independent activity/gesture recognition algorithm, whereas Section 7 introduces the framework that fuses wearable IMUs and sEMG for ASL recognition. Finally, Section 8 summarizes and highlights the primary conclusions of this work.

2. PRELIMINARIES

This section expands on some background topics to better contextualize the research described in subsequent sections. We first introduce the IMU sensor hardware which is used for all the sections followed by the introduction of EMG acquisition system which will be used for American Sign Language recognition research. Then, a 3-D depth camera Kinect is introduced and it is used in opportunistic sensor orientation and sensor location calibration work.

2.1 IMU Sensor Platform

Figure.2.1 shows the 9-axis motion sensor with dimensions 1"x1.5" that was designed and developed in our laboratory [35]. An InvenSense MPU9150 9-axis MEMS sensor is used to measure the 3-axis acceleration, 3-axis angular velocity and 3-axis magnetic strength. A Texas Instrument 16-bit low power microcontroller MSP430F5528 is used as the central processor. Both a dual mode Bluetooth module and a microSD card unit are available on the board. The user can choose to stream the data to a PC/tablet for real-time processing or log all the data to the microSD card for long-term movement monitoring. A charging circuit is included for the battery.

Figure 2.1 - Inertial sensors.

The magnetometer measures the magnetic field strength and is used as e-compass in some mobile phone applications. However, it suffers from severe magnetic interference from environment (*e.g. the existence of metal pipes*) and requires extensive calibration for different locations [36]. Thus, it is not used in any part of this research. The 3-axis gyroscope measures angular velocity of the motion. The 3-axis accelerometer measures the gravitational acceleration and non-gravitational acceleration caused by motions (known as force specific acceleration). The force specific acceleration is also called dynamic acceleration in our paper. Figure. 2.2 shows an example of how gravitational acceleration and force specific acceleration are measured by an accelerometer during an arm lifting movement of the user. The accelerometer measures acceleration a_x, a_y and a_z in its local frame *Xs*, *Ys* and *Zs*, where

$$a_x = a_{xg} + a_{xsf} \tag{2.1}$$

$$a_y = a_{yg} + a_{ysf} \tag{2.2}$$

$$a_z = a_{zg} + a_{zsf} \tag{2.3}$$

a_{xg}, a_{yg} and a_{zg} are decomposed values of gravitational vector **g** along sensor local frame *Xs*, *Ys* and *Zs*. a_{xsf}, a_{ysf} and a_{zsf} are decomposed values of specific force vector $\boldsymbol{a_{sf}}$ along *X-axis*, *Y-axis* and *Z-axis* of sensor local frame. The gravitational acceleration **g** is caused by earth gravity and points to earth surface. The specific force is caused by motion and in this example, it is caused by horizontal arm lifting movement. More details are explain in these two articles [37, 38].

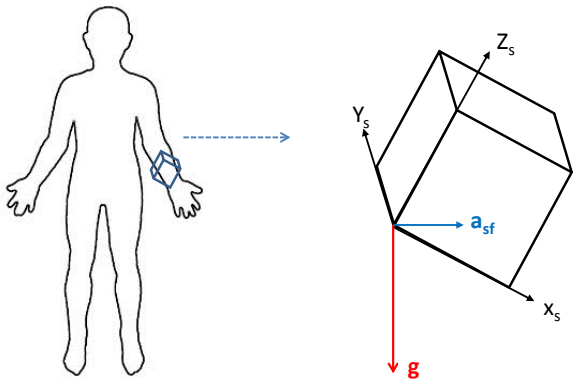


Figure 2.2 - Example of accelerometer measurement during horizontal arm lifting movement.

2.2 Kinect Sensor

Kinect is a low-cost RGB-Depth motion sensing device developed by Microsoft as shown in Figure. 2.3. It is widely used in applications of motion tracking [39, 40] and rehabilitation [4, 41]. Microsoft provides an API to obtain the joint positions of the human body as captured by the Kinect, enabling real time skeleton tracking. In this paper, the 3-D joint positions and joint tracking states are used in our algorithm. The body segment vectors are constructed from positions of every two adjacent joints.

Figure 2.3 - Kinect sensor.

2.3 Surface EMG Acquisition System

sEMG measures the electrical activity generated by skeletal muscle. Figure. 2.4 shows a customized 16-channel Bluetooth-enabled physiological signal acquisition system. It can be used for ECG, sEMG and EEG data acquisition. The system is used as a four channel sEMG acquisition system in this study. A TI low power analog front end, the ADS1299, is used to capture four channel sEMG signals and a TI MSP430 microcontroller is responsible for forwarding data to a PC via Bluetooth. A resolution of 0.4 μV is achieved setting a gain of 1 on the ADS1299. Covidien Kendall disposable surface EMG patches are attached to skin and the same electrodes are used as introduced in our previous work [42].

Generally, sEMG signals are in the frequency range of 0Hz -500 Hz depending on the space between electrodes and muscle type [43]. To meet the Nyquist criterion, the sampling rate is chosen as 1K Hz, which is usually used in surface EMG based pattern recognition tasks [44].

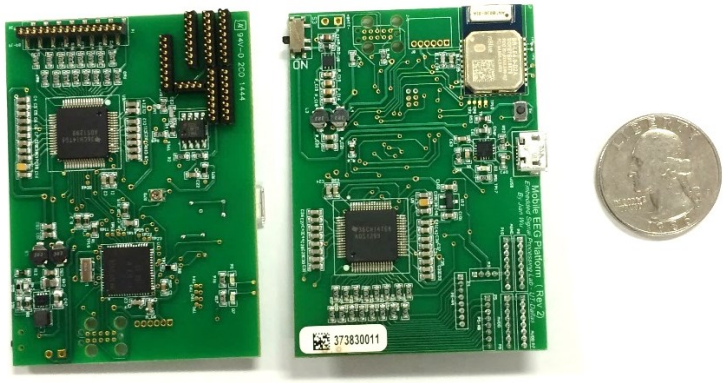


Figure 2.4 - sEMG acquisition system.

3. RELATED WORKS

3.1 Opportunistic Sensor Orientation Calibration

It is known that sensor displacement affects the accuracy of activity recognition algorithms. The impact of the sensor translation and rotation on a sample activity recognition algorithm called dynamic time warping is discussed in [23]. In [45], the authors explore how the rotation and translation displacement affect the results of recognition algorithms and provide recommendations about how to deal with sensor displacement.

Several approaches have been proposed to address the issue associated with sensor displacement and their impact on recognition algorithms. A solution to find the displacement invariant features has been proposed. In [46], device orientation independent features are used for step detection and direction estimation for the applications of dead reckoning. Another approach to study the statistical distribution of the features, and adjust the features adaptively has been suggested. The possibility of system self-calibration through the adjustment of the classifier decision boundaries is proposed in [47]. Similarly in [48], the authors propose a method to compensate for the data distribution shift caused by sensor displacements using an expectation-maximization algorithm and covariance shift analysis. These approaches adjust the feature space in case of a minor displacement. However, they cannot calibrate the presence of substantial displacements. If a major displacement occurs, the recognition algorithm will function with poor accuracy.

Other researchers focus on identifying the exact orientation and translation change during the movement by asking the user to perform certain movements. For instance, some have analyzed the acceleration data during walking to determine the sensor placement [29] and orientation [30]. In [5], the authors defined a global frame by asking the user to perform symmetric forward/backward movements (*e.g.,* walking). The accelerometer signals are projected to this global frame such that the sensor orientation is calibrated to this global frame. The disadvantage of this approach is that the user would be required to perform certain specific movements and if the global frame is altered, the algorithm will likely fail.

3.2 Opportunistic Sensor Location Calibration

Several prior investigations have been proposed to localize the on-body locations of the wearable sensors. Most techniques attempt to recognize the walking activity as the first step and after the walking is detected, the on-body sensor location is classified according to the training models. In one work, a sensor location and orientation independent algorithm is used to detect walking and then the specific motion characteristics of walking is leveraged to identify the on-body sensor locations [29]. Their algorithms achieved a 100% accuracy for 4 sensor placements. Another proposed approach can obtain an average of 89% accuracy in estimating 10 different sensor placements with extensive experiments on 25 subjects [49]. In contrast to [29], the latter work uses an unsupervised technique to detect the walking activity such that the effort to define the models and patterns in the setup phase is not required and the walking model is defined during the runtime. However, due to the symmetry between the left arm and

right arm or the right leg and left leg during walking, these algorithms may not be able to distinguish the location between right leg and left leg or between right arm and left arm.

Another technique is proposed to recognize the locations of wearable sensors with a full body motion capture configuration (17 sensors) which was validated with 10 healthy subjects and 7 patients [50]. This work calculates a global coordinate for each sensor using the 6 seconds of walking at the beginning of each trial. The entire sensor data stream would be projected to this global frame such that the orientation information of the sensor is not required to detect the walking. A decision tree based on the C4.5 algorithm is developed to determine the sensor positions.

In all the above works, the walking activity has to first be detected and the on-body sensor localization is achieved by the classification algorithms specifically working with the walking activity. In reality, walking activities may not always be present. It will be more useful if the on-body location can be inferred from arbitrary activities. An approach to determine 5 different on-body sensor placements from arbitrary activities is proposed [51]. This approach achieves up to 82% accuracy while it classifies the locations with a 6 minutes window. However, there will be a large overhead for training the HMM model which needs to capture many arbitrary activities. In addition, the classification for 6 minutes of data will be computationally expensive.

Our proposed approach calibrates the on-body sensor locations from the arbitrary activities by leveraging the information from an environmental camera and requires little to no training effort from the user at the setup phase. The work that comes closest to ours is described in [52]. In their work, the vertical angle change features are extracted for

both five Kinect body segments and five inertial sensors, and dynamic time warping (DTW) [53] is used to align the signal from two modalities to eliminate any time synchronization issues. Meanwhile, the Kinect segment which gives smallest DTW distance is chosen for further consideration. The correlation between inertial and Kinect signals according to the time stamps is calculated which serves as a confidence measure. The same procedure is repeated five times and the body segment which offers the largest average confidence is determined as the final location. Unlike the work in [52], we formulate the problem as a 3-D frame calibration problem, called Wahba's problem, and our method determines the on-body sensor localization by solving this problem. The solution to Wahba's problem will also calibrate the sensor frame to the Kinect frame which is an important step when these two modalities are used together for robust skeleton tracking applications [54, 55]. Moreover, our approach only uses the accelerometers to recognize eight sensor locations instead of using the combination of accelerometers and gyroscopes to recognize five body locations in [52].

3.3 Orientation Independent Activity/Gesture Recognition

As discussed in Section 3.1, sensor displacement affects the accuracy of the activity/gesture recognition algorithms and its impact on activity/gesture recognition are studies in prior research.

Three different approaches are proposed to address the issue when sensor rotational displacement will affect the result of the recognition algorithm. The first approach is to study the statistical distribution of the features and adjust the features adaptively. The possibility of system self-calibration through the adjustment of the

classifier decision boundaries is proposed [47]. Similarly, a method to compensate for the data distribution shift caused by sensor displacements using an expectation-maximization algorithm and covariance shift analysis is discussed [48]. These approaches adjust the feature space for small displacement ranges. However, they cannot calibrate for more substantial displacements and if a major displacement occurs, the recognition algorithm will exhibit poor accuracy.

The second approach is to recalibrate the sensor orientation and to transform the sensor readings to the original space in which the system is trained. An orientation independent approach that calculates the transformation rotation matrix with respect to a reference frame and transforms the accelerometer data vector back to this reference frame is proposed [56]. In this investigation, the researchers assume one of the sensor axes is fixed and the rotation occurs along this axis. This is not always true in reality and moreover, this technique estimates the gravity from a period of the same activity or posture (e.g., walking, cycling and standing). This technique does not work for transitional movements, like sit-to-stand or sit-to-lie. In another investigation, the use of signal average to estimate the gravity component and determine the vertical axis of the device is proposed [57]. The vertical axis is determined, and the sensor readings are projected onto this axis [58]. Since the vertical axis alone cannot define a frame, the technique extracts the signal magnitude which is perpendicular to the vertical axis and is along the horizontal axis. The complete frame orientation is calculated from a period of walking [59]. The vertical axis is estimated using the method in [57], while the horizontal axis is calculated as the first component of the PCA, which corresponds to the

direction in which the majority of movement occurs. The disadvantage of these approaches is that they require a period of 10 seconds or more of forward-backward movements to estimate the vertical and horizontal axes. The estimation will be incorrect if non-bipedal movements are present and moreover, the recognition algorithm will fail if the calibration is not performed in a timely manner.

The third approach explores the orientation independent features. The relative energy distribution over five different parts of the body is investigated to classify four different activities [31]. This approach performs well to distinguish dynamic activities that have different energy distribution on different body parts. However, for those activities that have similar energy distribution on different parts, (*e.g.*, walking and running), it may not exhibit acceptable performance. In another study, features in frequency domain are used to distinguish the periodic dynamic activities by analyzing the periodicity between different movements [32]. However, the frequency resolution is a concern when identifying the difference between activities that have similar periodicity. Moreover, these features are not suitable to detect short term transitional movements (*e.g.*, sit-to-stand, stand-to-sit).

In our work, we use the total angle change series as a time domain feature; our approach can recognize either dynamic movements or transitional movements as long as they have different angle change series. This discriminant feature is inherently present in all distinct movements. Our feature set is unique in the sense that we create a time series of total angle change in the duration of an activity at different time scales and we use it as a template for the activity.

As for location displacement of the sensors on body, the literature has considered two scenarios. The first scenario concerns displacement across different body parts; for example, the sensor should be worn in the trouser pocket whereas it is worn in the shirt pocket instead. The on-body location of the wearable sensor can be identified by classifying the accelerometer data when the user is walking [29]. In another work, the sensor location is determined by leveraging concurrent vision-based Kinect sensor skeleton information [52]. Our approach does not attempt to address this type of sensor location displacement and the existing proposed techniques could be leveraged. The second scenario of location displacement is the displacement that occurs on the same body link. The acceleration will be different if the sensor is placed in a different location of the same body link, however, the rotation angles will always be the same for the rigid body links. As a result, our approach will be robust to location displacements of this type. This analogy also holds when handheld devices are used for gesture recognition.

For template refinement and matching, several studies have been proposed to select the best representative signals to cover the variations in user gestures/activities [60, 61]. Generally, these systems will achieve better performance if a larger number of templates from training set is constructed. However, due to the computational constraints of wearable computers, only a smaller subset of templates should be considered. A template that has minimum distance to all other instances in the training set can be selected as the representative template [61]. To address the dynamically changing nature of human gestures, the templates are updated when a more recent gesture is detected or an incorrect gesture is observed by the user [60]. All these studies assume that the entire

template provides a good representation for the gesture/activity. However, in reality, movements may include consistent and inconsistent segments. The system performance can be further enhanced if the inconsistent segments are identified and discarded during the template matching. To the best of our knowledge, this is the first time the notion of template refinement and discarding the inconsistent segments of the templates is considered for gesture/activity recognition.

3.4 Sensor Fusion Enhanced American Sign Language Recognition

SLR systems are well studied in the areas of computer vision and image processing. Two vision-based real-time ASL recognition systems are studied for sentence level continuous American Sign Language using Hidden Markov Model (HMM) [62]. In the first system, the camera is mounted on the desk while in the second system, the camera is mounted on a cap which is worn by the user. They are both tested for 40 signs and achieve 92% and 98% accuracy, respectively. A framework for recognizing the simultaneous aspects of ASL is proposed [63]. This framework targets at addressing the scalability issue associated with HMM. It breaks down the signs into their phonemes and modeling them with parallel HMM. In this way, the state space will decrease significantly as the number of signs increases. Another vision-based SLR system is studied for a medium vocabulary Chinese Sign Language [64]. Robust hand detection, background subtraction and pupil detection are implemented as the first module, followed by a tiered-mixture density HMM. With the aid of a colored glove, this system achieves 92.5% accuracy for 439 Chinese Sign Language words. A combination of three new vision based features are explored for ASL recognition [65].

Three features are mapped into four components of ASL: hand shape, place of articulation, hand orientation and movement. The proposed features achieve 10.90% error rate on an existing dataset.

Glove-based SLR systems implement multiple sensors on the glove and capture the physical features of the gestures. Unlike vision-based systems, they do not require cameras mounted around the user and the system can perform recognition at any place at any time with a wearable glove. A glove-based Australian SLR system is studied using two classifiers (*i.e.* Instance based classifier and decision tree classifier) with some simple features. 80% accuracy is achieved for 95 AUSLAN signs [66]. The performance of artificial neural networks is explored for an ASL recognition system using a sensory glove [67]. It achieves about 90% accuracy for 50 ASL signs.

The low-cost wearable accelerometer and sEMG based SLR systems have the same advantages as glove-based systems compared to vision-based approach while they cost much less than glove based systems since they have fewer sensors deployed. Therefore, this kind of wearable SLR system is gaining more and more popularity in recent years. SLR system has been explored in several studies fusing information from accelerometer and sEMG sensors. The comparison of accelerometer based and sEMG based gesture recognition systems is discussed [27]. It is suggested accelerometer and sEMG sensors are good at capturing different information of gestures and the performance enhancement combining these two modalities has been studied. The experiments show 5% - 10% accuracy improvement is obtained after fusing these two modalities [28]. An accuracy of 93% of recognizing 60 Greek Sign Language signs is

achieved using only one effective sample entropy-based feature set for both accelerometer and sEMG [68]. A Chinese SLR framework is proposed fusing data from an accelerometer and 4-channel sEMG sensors [33]. Auto segmentation is applied to extract sign words from continuous sentences according to sEMG signal intensity. Multiple classifiers are implemented at different stages and the decisions achieved by each individual classifier are fused. At the first stage, the linear discriminate analysis (LDA) classifier is applied for both sEMG and accelerometer data which are able to capture hand shape and hand orientation, respectively. All sEMG and accelerometer features are cascaded and fit into a multi-stream HMM to recognize signs. A Gaussian mixture model is applied to fuse decisions obtained in the first stage. Although this system obtains a 96.5% accuracy for 120 Chinese sign words with sensors deployed on two hands, multiple stages and multiple classifiers make it unfavorable for implementation on real-time wearable computers-based applications which are constrained by limited computational resources. Another system is proposed to detect seven German sign words with 99.82% accuracy achieved using an accelerometer and one channel sEMG [34]. However, this work is not extensively evaluated for a large number of signs and does not include auto-segmentation which makes it difficult to operate in real time. The major differences between our work and the previous works are as follows: 1) An adaptive auto-segmentation is proposed to extract periods during which signs are performed using sEMG. 2) The best feature subset is selected from a broad range of features using information gain criterion and the selected features from different modalities (*e.g.* accelerometer, gyroscope and 4-channel sEMG) are discussed.

3) Gyroscope is incorporated and the significance of adding sEMG is analyzed. 4) Although such a system has been studied for Chinese Sign Language [33], our paper is the first study for American Sign Language recognition fusing these two modalities.

4. OPPORTUNISTIC SENSOR ORIENTATION CALIBRATION*

In this section, the opportunistic sensor orientation calibration algorithm is discussed. For most of the existing activity/gesture recognition algorithm, they have been designed and developed to work with fixed sensor orientation and will not function once a sensor orientation displacement happens. In order to help these systems still work when the sensor orientation displacement happens, we propose this sensor calibration method that calibrates the sensor orientation displacement leveraging environmental camera information and requires zero effort from the user.

4.1 Problem Formulation

4.1.1 Frame Definitions

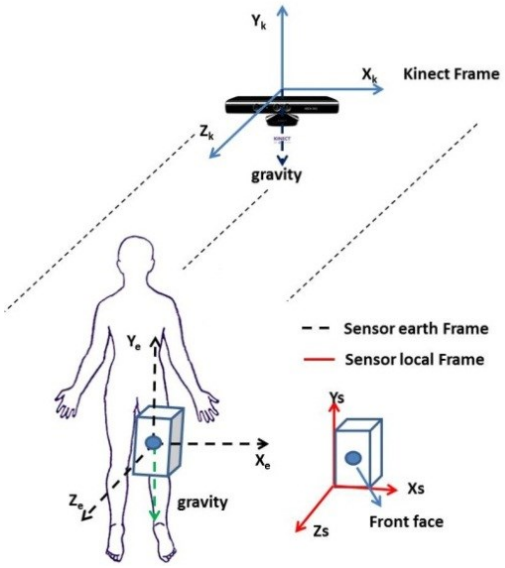


Figure 4.1 - Frame definitions. Reprint with permission from [69].

*Reprinted with permission from "Zero-Effort Camera-Assisted Calibration Techniques for Wearable Motion Sensors." in Proceedings of Wireless Health 2014 on National Institutes of Health, by Jian Wu and Roozbeh Jafari, 2014, ACM, New York. ©2014 by ACM.

There are four coordinate frames in this section when we formulating the problem: the human body frame, the sensor local frame, the sensor earth frame and the Kinect frame. These frames are defined as:

1). *Human body frame*: the human body frame is defined in Figure 4.1. It is the back view of a human body. The axes are represented by X_b, Y_b and Z_b..

2). *Sensor local frame*: The sensor local frame is shown in Figure 3. The face with a circle is the front face of the sensor. The Z_s always points out of the front face and X_s and Y_s are parallel to two sensor edges as shown in the figure.

3). *Sensor earth frame*: Sensor earth frame is defined as the blue dashed lines in Figure 4.1. The positive *Y*-axis Y_e is in the opposite direction of gravity. The positive *Z*-axis Z_e is parallel to the ground and is the projection of the sensor local frame *Z-axis* onto the transverse plane. The *X*-axis X_e is uniquely defined by Y_e and Z_e of a right handed orthogonal coordinate. Notice that the sensor may face in any direction, so the sensor earth frame is not unique and is determined by the direction of projection of the sensor local frame *Z-axis* onto the transverse plane.

4). *Kinect frame*: The Kinect frame coordinates are shown on the Kinect in Figure 4.1. The positive *Z*-axis Z_k points in the direction in which the Kinect is facing. The positive *Y*-axis Y_k is upward and the positive *X*-axis X_k points to the right of the Kinect (when viewed from the front). The tilt of the Kinect is 0 degrees, so Y_k is in the opposite direction of gravity.

4.1.2 Problem Formulation

The problem of the sensor displacement on rigid human body segments is divided into three sub problems in this paper as shown in Figure 4.2. The cylinders represent the human body segments (*e.g.* leg or arm), which have the same reference frame as the human body frame. The first case is the sensor's rotational displacement around Y-axis of human body frame, which is denoted as *yaw rotation* in this paper. The second case is the rotation of the sensor along the Z-axis of the human body frame which is called *roll rotation*. The third case is the sensor's line displacement along the segment which is denoted as sensor translation. The literature has shown that the impact of case 3 is often negligible on signal processing algorithms [23]. In this paper, we only focus on the calibration of the sensor yaw rotation β (case 2) and roll rotation γ (case 1) w.r.t. the human body frame.

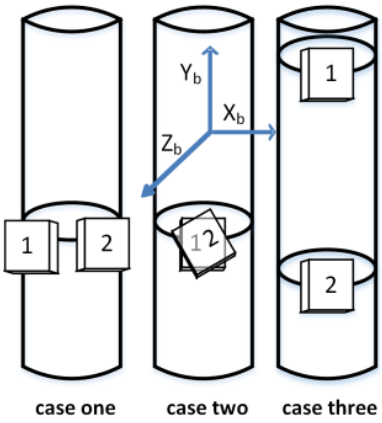


Figure 4.2 - Three cases of sensor displacement. Reprint with permission from [69].

To simplify the problem, we first assume the user faces Kinect camera such that the *Z*-axis of human body frame points in the same direction as the *Z*-axis of the Kinect

frame. Since the human body frame is oriented the same as the Kinect frame in this scenario, the problem becomes determining the sensor yaw rotation -β and roll rotation, γ, w.r.t. the Kinect frame. Once we obtain the results, we relax the assumption so that the human can face anywhere. The yaw rotation α between the Kinect frame and human body frame, which is the rotation about *Y-axis*, can be obtained from Kinect API. Now we can calculate the yaw rotation of sensor frame w.r.t. the human body frame. Since the *Z-axis* of the Kinect frame and the human body frame are in the same plane (human transverse plane), the sensor roll rotation w.r.t. the human body frame is the same as the sensor roll rotation w.r.t. the Kinect frame. The *Z-axis* of the sensor earth frame is defined as the direction of the projection of the sensor front face in the transverse plane of the human body frame, the yaw rotation of sensor local frame w.r.t. the Kinect frame is the same as the yaw rotation of the sensor earth frame w.r.t. the Kinect frame, which is a rotation along Kinect *Y-axis*. The angles used in this paper are defined in Table 4.1.

Table 4.1. Angle definitions. Reprint with permission from [69].

Angle symbol	Angle representation
β	Yaw rotation of the Kinect frame w.r.t. the sensor local frame
γ	roll rotation of the sensor local frame w.r.t. the Kinect frame
α	Yaw rotation of the Kinect frame w.r.t. the human body frame
φ	Yaw rotation of the sensor local frame w.r.t. the human body frame

4.2 Methodology

In this section, we first explain the orientation filter that estimates the orientation of sensor local frame w.r.t. the sensor earth frame. We also introduce the rotation distance between 3-D rotations. Next, the two-step search algorithm is explained. In the first step, a body segment rotation is measured in the sensor earth frame. Meanwhile, the same rotation is measured from Kinect skeleton data after rotating the Kinect frame into the sensor earth frame. By searching for the minimum rotation distance between the rotation measured from inertial sensor and the Kinect, the yaw rotation β between the sensor earth frame and the Kinect frame is determined. The same approach is applied to the second step search to determine the sensor roll rotation w.r.t. the Kinect frame γ. The yaw rotation between the Kinect frame and the human body frame, α, can be obtained from the Kinect API. As we have the yaw rotation between the sensor local frame and the Kinect frame and the yaw rotation between the Kinect frame and the human body frame, the yaw rotation of the sensor local frame w.r.t. the human body frame φ is achieved. Since the *Z-axis* of the Kinect frame and human body frame are in the parallel planes (parallel to human transverse plane), the sensor roll rotation w.r.t. the human body frame is the same as the sensor roll rotation w.r.t. the Kinect frame. Thus, the sensor yaw rotation and roll rotation w.r.t. the human body frames are both calibrated.

4.2.1 Orientation Estimation for Inertial Sensor

The inertial sensor measures 3-axis acceleration and 3-axis angular velocity in its local frame. An orientation filter [70] is used to estimate the orientation of the sensor

local frame w.r.t. the sensor earth frame, which is denoted as q_S^E. q is a quaternion representation of the orientation. S represents sensor local frame and E represents the sensor earth frame. A quaternion q is a four-dimensional complex number ($[q_w\ q_x\ q_y\ q_z]$) that can be used to represent the orientation of a coordinate frame in 3-D space. The relationship between quaternion and a rotation is explained in Section 4.2.2.

4.2.2 Rotation Distance

Before introducing our two-step search algorithm, we briefly discuss rotation distance metrics which are important measurements throughout our algorithm deployment. 3D rotations are widely used in numerous applications such as computer vision, computer graphics and robotics. The evaluation of the distance between two 3D rotations is an important task. There are several rotation distance metrics in the literature based on Euler angles, unit quaternion and rotation matrices [71], which are commonly used to represent 3D rotations. The 3D rotation matrix is a 3×3 orthogonal matrix that is used to perform a rotation in Euclidean space. The matrices form a group called the Special Orthogonal group, SO(3), in which the operations of multiplication and inversion are continuous functions of the matrix entries. In this paper, we use a rotation metric introduced in [72] to derive our algorithm. The metric is denoted as

$$\mathrm{d}(\mathrm{R}_1, \mathrm{R}_2) = \mathrm{tr}(\mathrm{I} - \mathrm{R}_1\mathrm{R}_2^{\mathrm{T}}) \tag{4.1}$$

where R_1 and R_2 are two rotation matrices, I is an identity matrix and tr represents the trace of a matrix.

4.2.3 Two-step Search Algorithm

4.2.3.1 First step yaw rotation search

The first step search finds the *yaw rotation* of the sensor local frame w.r.t. the Kinect frame, which is the same as the yaw rotation of the sensor earth frame w.r.t. the Kinect frame. As shown in Figure 4.3, X_e, Y_e and Z_e are three axes of the sensor earth frame and X_k, Y_k and Z_k are three axes of the Kinect frame. Y and Y_k are parallel to each other and point to the opposite of the gravity vector. The yaw rotation β is the rotation along Y axis from Kinect frame to sensor earth frame.

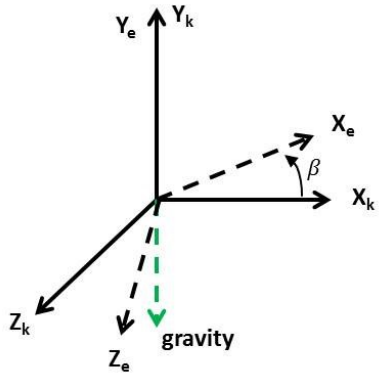


Figure 4.3 - Yaw rotation formulation. Reprint with permission from [69].

Consider the rotation of the body segment AB in Figure 4.4, this rotation can be measured from the inertial sensor in the sensor earth frame or from the Kinect skeleton joint position information in the Kinect frame. If the two frames are rotated into the same reference frame, the measured rotations from these two systems should be the same. In this paper, we rotate the Kinect frame to the sensor earth frame.

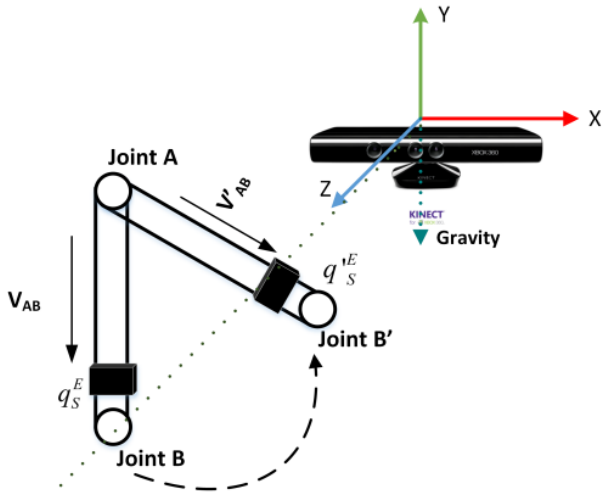


Figure 4.4 - Body segment rotation. Reprint with permission from [69].

First, we calculate the rotation of body segment AB in the sensor earth frame. As discussed in Section 4.2.1, the orientation filter can measure the sensor orientation w.r.t. the sensor earth frame, and q_S^E and ${q'}_S^E$ represent the sensor orientation w.r.t. the sensor earth frame at two different states of the body segment movement. The 'state' represents the state of the body segment at a certain time. For example, for a sit-to-stand movement, one state would be the orientation of body segment (*e.g.* thigh) in sitting posture and the other state would be the orientation of the body segment in standing posture. The segment rotation in the sensor earth frame is calculated by:

$$q_E = (q_S^E)^{-1} \otimes {q'}_S^E \tag{4.2}$$

q_E is the segment rotation in sensor earth frame and $(q_S^E)^{-1}$ is the inverse of q_S^E. The inverse of a quaternion, q^{-1}, represents the inverse rotation of q. $\otimes$ represents quaternion multiplication.

The rotation of segment AB is then constructed from Kinect position data. The position data for joint *A* and joint *B* can be obtained from the Kinect API for every frame, denoted as P_A and P_B respectively. A 5 span moving average filter is applied to the position data to smooth the noise. V_{AB} and V'_{AB} are the body segment vectors at states AB and AB'. They are defined as:

$$V_{AB} = P_B - P_A \tag{4.3}$$

$$V'_{AB} = P'_B - P_A \tag{4.4}$$

If these two vectors are rotated to the sensor earth frame from the Kinect frame, they can be used to construct the segment rotation in sensor earth frame. The rotation from Kinect frame to sensor earth frame is represented by a rotation axis [0, 1, 0] (*Y-axis*) and a rotation angle β. The rotation axis is the axis about which the rotation happens. The representation of the rotation by rotation axis vector **r** and rotation angle θ can be translated to a quaternion representation q by:

$$\left.\begin{aligned} q_x &= r_x * \sin(\theta/2) \\ q_y &= r_y * \sin(\theta/2) \\ q_z &= r_z * \sin(\theta/2) \\ q_w &= \cos(\theta/2) \end{aligned}\right\} \tag{4.5}$$

qx, qy, qz and qw are the four components of the quaternion vector q and *rx*, *ry* and *rz* are the three components of the unit vector ***r***. The Kinect frame w.r.t. the sensor world frame $q_K^E(\beta)$ can then be calculated by (4.5). The 3-D vectors V_{AB} and V'_{AB} are rotated into the sensor earth frame as $V_{AB}^E(\beta)$ and $V_{AB}^{\prime E}(\beta)$ by:

$$[\,0 \quad V_{AB}^E(\beta)] = (q_K^E(\beta))^{-1} \otimes [0 \quad V_{AB}] \otimes q_K^E(\beta) \tag{4.6}$$

$$[\,0 \quad V_{AB}^{\prime E}(\beta)] = \left(q_K^E(\beta)\right)^{-1} \otimes [0 \quad V'_{AB}] \otimes q_K^E(\beta) \tag{4.7}$$

Zero in (4.6) and (4.7) is value of qw of a quaternion. The rotation from $V_{AB}^{E}(\beta)$ to $V_{AB}'^{E}(\beta)$ can be represented by a rotation axis **r** and the rotation angle θ. The axis **r** and rotation angle θ can be achieved by:

$$V_{AB}^{E}(\beta) \cdot V_{AB}'^{E}(\beta) = \|V_{AB}^{E}(\beta)\| \|V_{AB}'^{E}(\beta)\| \cos\theta \tag{4.8}$$

$$V_{AB}^{E}(\beta) \times V_{AB}'^{E}(\beta) = \|V_{AB}^{E}(\beta)\| \|V_{AB}'^{E}(\beta)\| \sin\theta \mathbf{r} \tag{4.9}$$

r and θ are also functions of β, after normalizing **r**, the segment rotation in the sensor earth frame constructed from two segment vectors $q'_{E}(\beta)$ is calculated by (4.5).

Now that q_E and $q'_{E}(\beta)$ represent the same rotation in the sensor earth frame, and thus the rotation distance between them should be 0. In reality, the Kinect skeleton vectors do not perform exactly the same as the inertial sensors. The inertial sensor has the rotation along the segment itself (*e.g.* the twist of the arm), but the Kinect will be unable to capture this degree of rotation. As a result, the rotation distance between q_E and $q'_{E}(\beta)$ is not exactly 0. If we search for β from 1 degree to 360 degrees, the targeted β will give the minimum rotation distance between q_E and $q'_{E}(\beta)$. In order to calculate the rotation distance, we first convert q_E and $q'_{E}(\beta)$ to rotation matrices R_E and $R'_{E}(\beta)$. The rotation distance $d(R_E, R'_{E}(\beta))$ can be calculated from (4.1).

Our search algorithm relies on the rotation between two states; if there is no rotation or a very small rotation, the movement will not give a good estimation of the yaw rotation angle β. Before beginning the search, we need to choose two states between which sufficient rotation occurs. It is important that we do not use the rotation between two states measured from inertial sensor to check the rotation since the inertial sensor can have the rotation along the segment itself, which is different from the

measurement of the Kinect system and cannot be used for our algorithm. In this paper, the rotation quality is validated from the segment rotation constructed from Kinect skeleton data. Since the gravity vector remains the same during one rotation of a body segment, the rotation between the body segment vector and the gravity vector q_{BS}^{G} is calculated for each frame during the rotation. By comparing the rotation of a body segment w.r.t. the gravity vector at two states, we can get the rotation distance of this body segment between the two states. For two different states during one body segment rotation, state 1 and state 2, we have q_{BS1}^{G} and q_{BS2}^{G}. Once we convert them to rotation matrices R_{BS1}^{G} and R_{BS2}^{G}, the rotation distance between state 1 and state 2, μ, can be calculated as:

$$\mu = d(R_{BS1}^{G}, R_{BS2}^{G}) \tag{4.10}$$

A rotation distance below 0.1 is considered to be too small to be used in our algorithm. The choice of threshold value of μ is discussed in Section 5.3.1. Based on the above analysis, the first step yaw search algorithm is formulated and described in Algorithm 1.

Algorithm 1. First step yaw direction search

Calculate μ according to (4.10);

if $\mu < 0.1$

The movement is not qualified, choose another one;

else

Continue;

end

for $\beta = 1{:}360$

Calculate $d(R_E, R'_E(\beta))$;

$\hat{\beta} = argmin(d(R_E, R'_E(\beta)))$;

end

return $\hat{\beta}$.

The $\hat{\beta}$ is the estimated yaw rotation of the Kinect frame w.r.t. the sensor local frame, so the yaw rotation of the sensor local frame w.r.t. the Kinect frame is -$\hat{\beta}$.

4.2.3.2 Second step roll rotation search

From the previous section, the estimated sensor yaw rotation -$\hat{\beta}$ w.r.t. the Kinect frame is obtained. In this section, we explain the second step search for the sensor roll rotation w.r.t. the sensor earth frame, which is the same as the sensor roll rotation w.r.t. the Kinect frame.

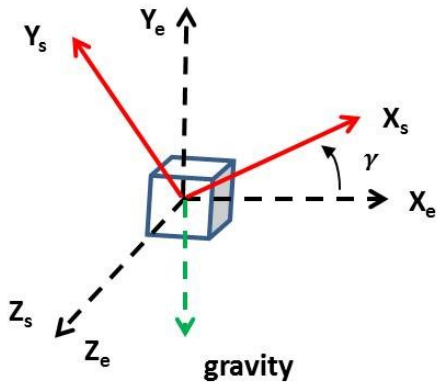


Figure 4.5 - Roll rotation formulation. Reprint with permission from [69].

Figure 4.5 shows the sensor roll rotation about the *Z-axis* of the sensor earth frame. *Xs*, *Ys* and *Zs* are the axes for the sensor local frame and X_e, Y_e and Z_e are axes for the sensor earth frame. Like the first step search, we define two states to calculate the same rotation measured by the Kinect skeleton and the inertial sensor in the sensor earth frame. One state is an ideal state in which the body segment vector is the same as the gravity vector and the inertial sensor only has a rotation of γ about the *Z-axis* of the sensor earth frame. The other state is an arbitrary state with the segment vector V_{AB}. The rotation between these two states is the rotation from the gravity vector to the body segment vector AB.

First, we construct the body segment rotation w.r.t. the gravity vector from the Kinect skeleton in the sensor earth frame. The yaw rotation $\hat{\beta}$ is known, and the vector V_{AB} can be rotated to the sensor earth frame by (4.6) as $V_{AB}^{E}(\hat{\beta})$. The gravity vector remains the same from the Kinect frame to the sensor earth frame and is represented by

[0 -1 0]. Using (4.8), (4.9) and (4.5), the rotation between body segment vector and the gravity vector is calculated and represented by $q'_E(\hat{\beta})$.

Next, the rotation between two states from the inertial sensor is analyzed. The arbitrary state sensor orientation w.r.t. the earth frame is the output of the orientation filter q_S^E. At the ideal state, the sensor orientation w.r.t. the sensor earth frame is composed of the rotation axis *Z-axis* ([0 0 1]) and rotation angle γ. Then from (4.5), the sensor orientation w.r.t. the sensor earth frame at the ideal state is calculated as $q_{iS}^E(\gamma)$. The body segment rotation in sensor earth frame is defined as:

$$q_E(\gamma) = (q_{iS}^E(\gamma))^{-1} \otimes q_S^E \tag{4.11}$$

Covert $q'_E(\hat{\beta})$ and $q_E(\gamma)$ to rotation matrices $R'_E(\hat{\beta})$ and $R(\gamma)$. By searching for the minimum rotation distance $\Omega(\gamma)$ between $R'_E(\hat{\beta})$ and $R(\gamma)$, the γ is determined.

$$\Omega(\gamma) = \mathrm{d}(R'_E(\hat{\beta}), \mathrm{R}(\gamma)) \tag{4.12}$$

The second step of the search algorithm is described in Algorithm 2.

Algorithm 2. Second step roll rotation search

---

for γ = 1:360

Calculate $\Omega(\gamma)$ according to (4.12);

$\hat{\gamma} = argmin(\Omega(\gamma))$;

end

return $\hat{\gamma}$.

4.2.3.3 Sensor orientation with respect to the human body frame

From the two-step search, we get the sensor roll and yaw rotation w.r.t. the Kinect frame. If the Kinect frame is the same as the human body frame (i.e., the subject is facing the Kinect), the roll and yaw are the same as w.r.t. the human body frame. However, if the user is not facing the Kinect such that the *Z-axis* of human body frame is not parallel to the *Z-axis* of the Kinect frame, there will be a yaw rotation between the Kinect frame and the human body frame. Notice that this will only affect the yaw rotation. Fortunately, we can get the yaw rotation between these two frames from Kinect API. The rotation of the hip center along Kinect *Y-axis* represents this yaw rotation. Let this yaw rotation be α, the sensor yaw rotation w.r.t. the human body frame is:

$$\varphi = -\hat{\beta} - \alpha \tag{4.13}$$

4.3 Experimental Setup

In the experiments, the Kinect is placed on a flat table with a tilt angle 0 degrees. The subjects perform daily activities in front of the Kinect. The subjects wear two sensors: one on the thigh and the other on the right upper arm. Four subjects were chosen for the experiments and 6 daily activities were performed to test our approach. The activities are listed in Table 4.2. These predefined activities are just some examples meant to exhibit generalizability of our method.

Table 4.2. Activities for the experiments. Reprint with permission from [69].

No	Activity	No	Activity
1	Walking	4	Sit-to-Stand
2	Kneeling	5	Stand-to-Sit
3	Leg-Lifting	6	Arm Stretching

For each activity, the sensors are placed in four different yaw configurations approximately by the user. Figure 4.6 shows the four yaw configurations for thigh sensor: 0-degree in Figure 4.6.a, 90-degree in Figure 4.6.b, 180-degree in Figure 4.6.c and 270-degree in Figure 4.6.d. For each yaw configuration, the subjects were asked to place the sensors in two random roll rotations. All subjects repeated each activity 8 times.

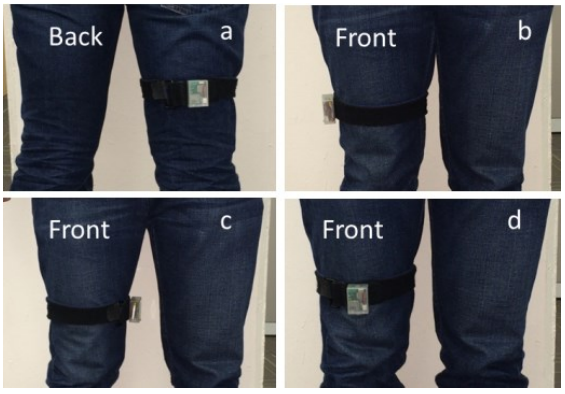


Figure 4.6 - Four different yaw configurations. Reprint with permission from [69].

Moreover, we compare the performance of our calibration method with a non-zero-effort method [59] using an activity recognition application.

4.4 Experimental Results

4.4.1 Yaw Rotation Search

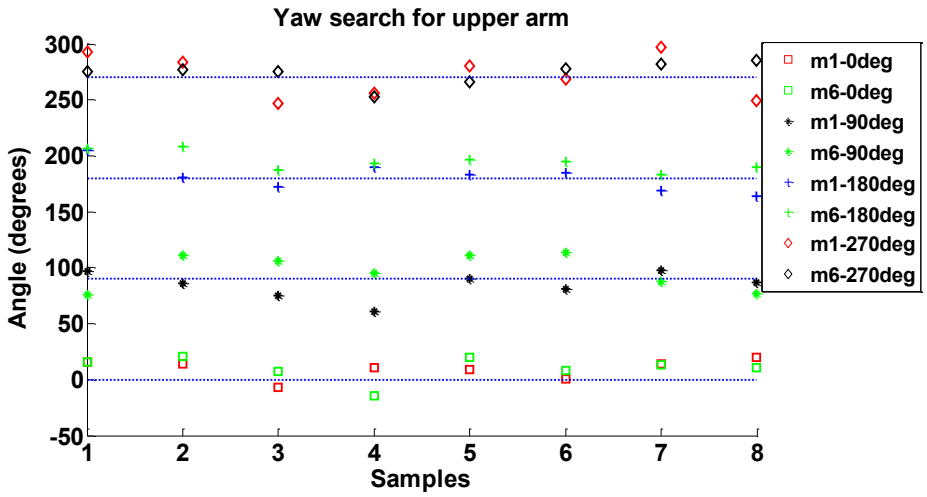


Figure 4.7 - Search results for yaw rotation for upper arm. Reprint with permission from [69].

To validate the yaw search technique, there should be a rotation that occurs during the movement. For the upper arm sensor, the segment has a qualified rotation only for arm stretching and walking, and thus only these two movements are chosen to calibrate the upper arm sensor. For the thigh sensor, all movements are used to calibrate the sensor except for the arm stretching. To test the yaw search technique, we manually place the sensor with the yaw rotation of 0 degrees, 90 degrees, 180 degrees and 270 degrees for all experimental configurations. Since these configurations are only approximately achieved by the user, they are not a gold standard. By checking the distribution of all the search results, we determine the consistence and the correctness of our algorithm. Figure 4.7 shows the search result distribution for the arm sensor. Since the arm only has sufficient rotation during movement 1 and movement 6, the results are

reported only for these two movements. The X-axis represents the number of experiments and trials and the Y-axis represents the search results for yaw angles. The four dashed lines from the bottom to the top are the four expected yaw results $y = 0$, $y = 90$, $y = 180$ and $y = 270$, respectively. The label 'mx-ydeg' in the figure represents the results for movement x with a yaw configuration of y degrees. For example, 'm1-0deg' is the result for movement 1(walking) with the 0-degree yaw configuration. From the figure, we observe that for the two movements, all search results fall around the line which corresponds to their own yaw rotations. The distribution of the results proves our algorithm works well for the two arm sensor movements.

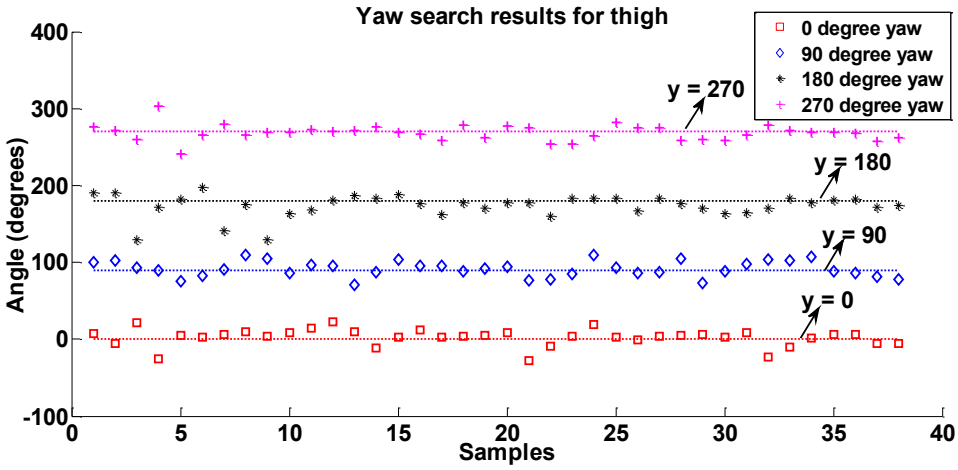


Figure 4.8 - Search results for yaw rotation for thigh. Reprint with permission from [69].

Figure 4.8 shows the distribution of all the search yaw results for the thigh sensor, for all 5 movements. The lines $y = 0$, $y = 90$, $y = 180$ and $y = 270$ are the four lines for the expected results for the configurations of 0 degrees, 90 degrees, 180 degrees

and 270 degrees misplacement. The obtained results for the four configurations all fall near their corresponding lines. The results for the 180-degree configuration have larger deviations than the other three. The reason for this is that when we placed the sensor on the back of the subject; it is harder to find the 180 degree position than in the other three cases. The overall distribution in the figure shows good search results for the thigh sensor for all movements.

4.4.2 Roll Rotation Search

To validate the accuracy of our roll search algorithm, for each movement, we asked the user to stand still at the beginning of the movement and measured the roll rotation from the inertial sensor. The measured roll angle from inertial sensor serves as the ground truth of our algorithm because the orientation filter can achieve a very good accuracy [70] for the roll rotation calculation using the gravity as a reference.

Table 4.3. Roll search errors for different subjects for arm and thigh. Reprint with permission from [69].

Subject Number	Arm Accuracy (RMSE in degrees)	Thigh Accuracy (RMSE in degrees)
Subject 1	10.733	5.98
Subject 2	5.11	5.60
Subject 3	13.5	5.32
Subject 4	9.60	5.60
Total	10.73	5.59

Table 4.3 shows the root mean square errors between the inertial sensors reported roll results and the results obtained by our algorithm for different subjects. The thigh sensor has a 5.59 degree RMSE for all movements and for all the subjects while the arm sensor has a 10.73 degree RMSE. The reason for the difference is that the thigh has less moving freedom than the upper arm and the arm muscle movement will lead to more rotation along the segment than the thigh muscle which will affect the search algorithm. The accuracy of our calibration technique on thigh is very consistent (RMSEs between 5 and 6 degrees) for all subjects, which indicates that the thigh movement does not exhibit significant variations for different subjects. Conversely, the arm sensor accuracy varies from subject to subject because of the greater level of variations for the arm movements. The total RMSEs for both the arm and the thigh indicate that our algorithm achieves a good accuracy for the roll rotation.

4.4.3 Activity Recognition Performance

We also validate our approach leveraging an activity recognition application and compare it with the performance of a non-zero-effort approach [59]. Subjects perform the same set of activities with sensors with random orientations. A template matching algorithm based on dynamic time warping (DTW) is used to implement the recognition using 3-axis accelerations. The template is trained when the sensor is placed without any rotational displacement. During the testing phase, the accelerations are transformed to the original frame (prior to rotation displacement) and the transformed signals are compared to the templates for classification. To determine the global frame as outlined in [59], the user is asked to perform 10 seconds of walking.

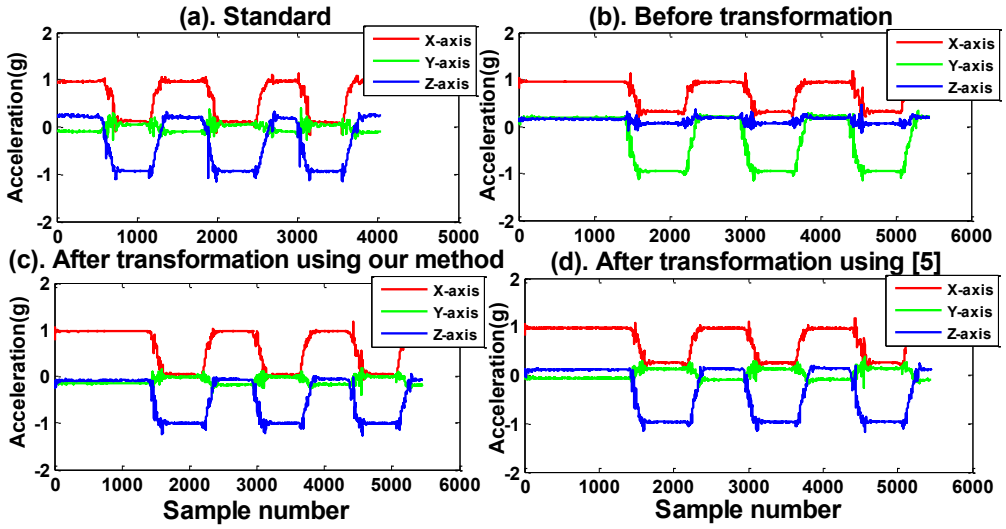


Figure 4.9 - Calibration results for sit-to-stand and stand-to-sit patterns. Reprint with permission from [69].

Figure 4.9 shows the calibration results for sit-to-stand and stand-to-sit patterns. Figure 4.9.a shows the standard pattern where no rotation displacement is present. Figure 4.9.b shows the signals before transformation while a rotation displacement is present and Figure 4.9.c and Figure 4.9.d show the calibrated patterns using our approach and using the approach in [59] respectively. It can be observed that both our method and the method in [59] have good calibration results. The calibrated signals from both approaches are similar to each other and are close to the standard signals.

Table 4.4 Activity recognition accuracy comparison. Reprint with permission from [69].

Method	**Activity Number**					
	1	2	3	4	5	6
Our approach	93%	98%	92%	100%	93%	100%
Approach in [59]	92%	98%	90%	100%	92%	100%

Table 4.4 shows the performance of the activity recognition algorithm leveraging both calibration methods. We can see that both calibration methods achieve similarly high accuracy which illustrates the effectiveness of our zero-effort algorithm in conjunction with activity recognition. The approach in [59] performs slightly better than our approach likely due to the fact that the error from Kinect and skeleton construction likely impacts our calibration whereas vision-based sensors are not used in [59].

4.5 Conclusion

In this work, we proposed a zero effort two-step search algorithm to calibrate orientation of wearable sensors by calculating the orientations of the sensors with respect to the human body frame based on rotation distance optimization. The experimental results from 4 subjects over 6 daily movements show that our algorithm achieves consistent and accurate results. We also evaluate the performance of our method for activity recognition and compare the results with a non-zero-effort approach and the results show our approach achieves similarly good performance.

5. OPPORTUNISTIC SENSOR LOCATION CALIBRATION*

Similar to the previous chapter and in order to address the sensor location displacement, an opportunistic sensor location calibration method is described in this section. It calibrates the sensor location displacement leverage a 3-D camera information and requires zero effort from the user.

5.1 Methodology

5.1.1 Definitions

Before introducing the details of our approach, the term definitions are summarized in Table 5.1 to enhance the readability of the paper and equations. We refer to each sample or data point as a frame. For example, if the camera is capturing video at 30 frames per second, we will have 300 frames over 10 seconds. If the accelerometer is sampling at 200 Hz, we will have 400 frames over 2 second. The 3-D accelerometer measures acceleration which is a combination of acceleration observed due to movements and the gravity along three axes of the sensor and at each frame or sample it generates a 3x1 vector. f_{aj}^{i} denotes the *ith* frame of *jth* accelerometer data. *j* is the index for each wearable accelerometer and we are considering determine the location for multiple accelerometers. Kinect API captures human skeleton from which all joints positions in Kinect frame are obtained. Two joints of a body segment construct a body segment vector. At each frame, Kinect captures all joint positions and segment vectors

are constructed. Each segment vector is a 3x1 vector. f_{Km}^{i} is *ith* frame of *mth* Kinect body segment vector. *m* is the index of each body segment (e.g., arm, thigh). If accelerometer sensor and Kinect sensor are perfectly synchronized, then f_{aj}^{i} and f_{Km}^{i} measure movements occurring at exactly the same time. However, it is not guaranteed that they will be strictly synchronized to each other and a delay of *d** frames may exist between two streams. This is because each sensor modality may have its own clock. In IoT settings, we can always expect lack of perfect synchronization and delays due to wireless communications. Our formulation handles this issue. In this paper, we consider windows of data and *n* is the number of samples in one window. $\boldsymbol{a_{ji}}$ is the normalized vector of f_{aj}^{i} and $\boldsymbol{k_{mi}}$ is the normalized vector of f_{Km}^{i}. $\boldsymbol{A_{jm}}$ is a rotation matrix that transforms *jth* accelerometer vectors to *mth* Kinect segment vectors in a window. e_{jm} is the least square error between *jth* accelerometer and *mth* body segment after solving Wahba's problem.

Table 5.1. Term definition. Reprint with permission from [73].

Term	Definition
$\boldsymbol{f_{Km}^{i}}$	*ith* frame of *mth* Kinect body segment vector, it is a 3 by 1 vector
$\boldsymbol{f_{aj}^{i}}$	*ith* frame of *jth* accelerometer data vector, it is a 3 by 1 vector
d^*	the asynchronization delay between accelerometer data stream and Kinect data stream
n	Total number of samples in a window
$\boldsymbol{a_{ji}}$	*ith* normalized *jth* accelerometer data vector in a window of *n* samples, it is a 3 by 1 vector
$\boldsymbol{k_{mi}}$	*ith* normalized *mth* Kinect body segment vector in a window of *n* samples, it is a 3 by 1 vector
$\boldsymbol{A_{jm}}$	Rotation matrix that transforms *jth* accelerometer vector to *mth* Kinect body segment vector, 3 by 3 matrix
$\boldsymbol{E}(A_{jm})$	The least square error after transforming all *n* samples of *jth* accelerometer vectors to *n* samples of *mth* Kinect body segment vectors in a window.
$\boldsymbol{e_{jm}}$	The least square error between *jth* accelerometer and *mth* body segment after solving wahba's problem

5.1.2 Overview

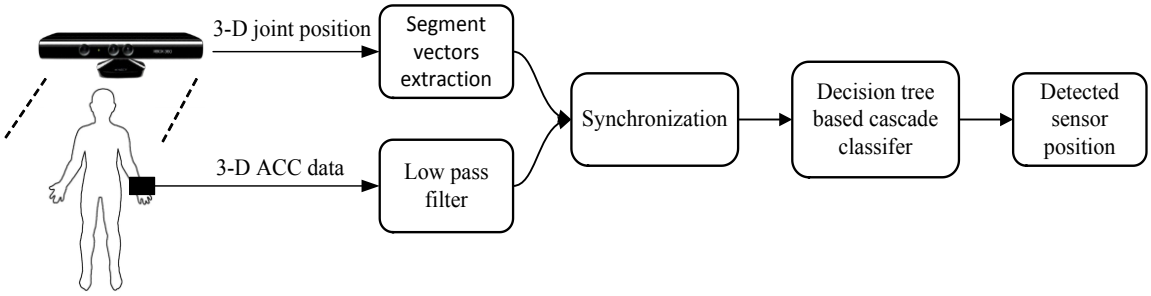


Figure 5.1 - Diagram of proposed approach. Reprint with permission from [73].

In this investigation, we use 8 wearable sensors attached to 8 body segments which are listed in Table 5.3 in Section 5.2. The body segments can be captured by Kinect cameras. The objective of our approach is to find the correct location information for all 8 accelerometers. To achieve this, we use observations made by two modalities and attempt to match them with each other. The accelerometer measures both dynamic acceleration due to movement and gravity vector. The Kinect body segment vector measures the same vector change as gravity change in Kinect frame. Although Kinect body segment does not measure the gravity directly, the rotation of the body segment in Kinect frame is the same as the rotation of gravity vector in sensor frame. Thus, the accelerometer and Kinect segment observes the same rotations from two different frames. Since the accelerometer measures the dynamic motion at the same time, we applied a weighting method to reduce the impact of this part of acceleration which is explained in Section 5.1.5.3. Our approach transforms accelerometer measured gravity vector observations in a window to Kinect measured segment observations. If a sensor is attached to a segment, after transforming the accelerometer vector observations to the

Kinect frame, they will become the same vector observations as this Kinect body segment vector observations. Therefore, we will get smallest transformation error for this pair of accelerometer and Kinect segment compared to errors obtained between this accelerometer and other Kinect segment vectors. This will be explained in details in Section 5.1.4. This is the core idea of our approach. Our proposed approach is shown in Figure 5.1. Motion sensors measure the 3-D acceleration (ACC) data in the sensor local frame, the Kinect measures 3-D joint positions of the human skeleton in the Kinect frame. A low pass filter is applied to the ACC data to remove the high frequency noise. Details of the preprocessing are articulated in Section 5.1.3. The joint position data is used to construct the body segment vectors. We formulate the problem as the Wahba's problem and solve for the frame differences between the ACC vector observations measured from one accelerometer and all the segment vector observations from 8 body segments measured in the Kinect frame. The exact body segment which has the sensor will observe the same movement and thus the accelerometer vector observations and this body segment vector observations are exactly same vector observations in two different frames. For the accelerometer vector observations and the other body segment vectors which do not include this sensor, if the accelerometer vector observations are transformed to the other body segments vector observations, there will be a larger error. The errors will be described in detail in Section 5.1.4. Using a minimum least square error acquired after solving the Wahba's problem between an accelerometer and all body segments, the sensor is assigned to a certain body segment by a decision tree based cascade classifier. The details are presented in Section 5.1.5.1.

5.1.3 Preprocessing

The accelerometer will have high-frequency noise which generates a large amplitude peak in the signal. Since we are calibrating the coordinate differences frame by frame, if the accelerometer vector with the high frequency noise is used, the calibration error will increase. As most human activities of daily living are at a low frequency and in order to remove the high frequency noise existed in accelerometer data, a low pass filter is usually applied with a cut-off frequency of 4 Hz – 8 Hz [74]. In our paper, 5 Hz low pass filter is applied. From the Kinect skeleton data, the joint positions are obtained. Since the body segment vector information is required, we construct the segment vectors from joints positions.

Synchronization of the two systems is important to our approach, because our algorithm assumes each pair of vectors in the two different coordinate frames is observed at the same time. However, our approach adapts the idea of opportunistic sensing and the wearable accelerometer and vision sensor are fused in an opportunistic manner. For example, a user wears his music player on his arm which has an accelerometer. When he enters the gym and performs exercises, the camera at the door captures his skeleton information. The skeleton information and accelerometer data are both sent to cloud and our algorithm calibrates the location sensor on the cloud. Because of the opportunistic nature of our approach, the accelerometer and camera are possibly not well synchronized. The sampling rate of the motion sensor is 200 Hz and the sampling rate of the Kinect is 30 Hz. The Kinect samples are interpolated using cubic splines to 200Hz. Canonical correlation analysis (CCA) has proved to be effective to

synchronize the audio and video features before fusing them [75]. In this paper, we adopt the same method to synchronize the Kinect and accelerometer data. As discussed in Section 5.1.1, we denote the *mth* Kinect body segment and *jth* accelerometer data of the *ith* frame by f_{Km}^{i} and f_{aj}^{i}, respectively. The problem becomes finding the delay d^* to maximize the mutual information between the Kinect and accelerometer. After d^* is obtained, the two modalities are synchronized by shifting d^* samples.

5.1.4 Wahba's Problem Formulation

Wahba's problem, first posed by Grace Wahba in 1965, seeks to find a rotation matrix between two coordinate systems from a set of (weighted) vector observations [76]. In our technique, the accelerometer and Kinect segment observe the same physical movement if the accelerometer is attached to this body segment. It means that 3-D accelerometer observations and 3-D body segment observations from Kinect are the same observations from two different frames. This is explained using Figure 5.2. Two joints A and B construct a body segment. Kinect measures 3-D position data of joint A and B and the vector $\mathbf{V_{AB}}$ can be easily constructed. Assume that this body segment is placed in parallel to gravity, $\mathbf{V_{AB}}$ can be thought of the same vector as gravity since both of them will be normalized to unit vectors in our later formulation. As body segment moves from AB to AB', the body segment vectors measured by Kinect during this period can be considered as gravity vector rotates from AB to AB' position. It is known that the gravitational acceleration measures the rotation of gravity in the earth frame [77]. Thus the normalized vectors measured by Kinect and the normalized gravitational vectors measured by accelerometer can be considered as same vectors measured in

different reference frame. As we discussed in Section 2.1, accelerometer measures not only gravitational acceleration, but also specific force acceleration. Since there is no easy way to separate them, the specific force acceleration will be considered as noise in our algorithm.

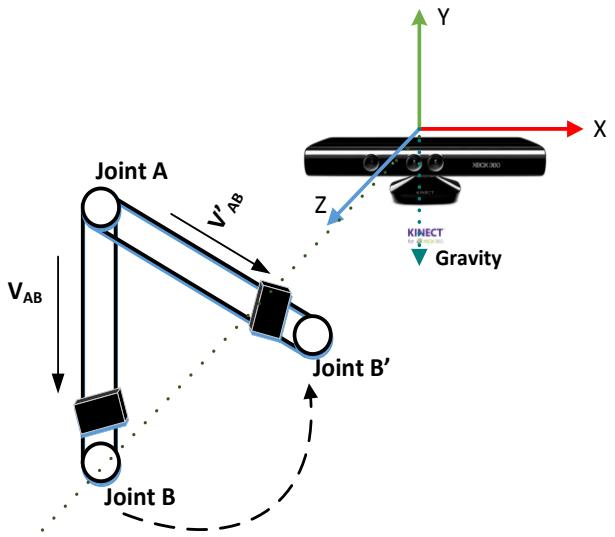


Figure 5.2 - Example of a body segment rotation. Reprint with permission from [73].

Let $\boldsymbol{a_{ji}}$ be the *ith* normalized vector of *jth* accelerometer in a window of *n* observations. $\boldsymbol{k_{mi}}$ is the corresponding normalized *mth* body segment vector measured in Kinect frame. The normalization procedure for accelerometer observation vector and Kinect body segment vector are the same. Both of them are 3D vectors and are normalized to corresponding unit vectors which are required by Wahba problem formulation. $\boldsymbol{A_{jm}}$ is a 3 by 3 rotation matrix which transforms the *jth* accelerometer vectors to *mth* Kinect body segment vectors. The rotation matrix is an orthogonal matrix. To solve for the rotation matrix, we attempt to minimize the least square error cost

function as shown in Eq. (5.1). In Eq. (5.1), $E(\boldsymbol{A_{jm}})$ is the weighted least square error between all *n* observations of *mth* Kinect body segment and *n* transformed *jth* accelerometer observations with rotation matrix $\boldsymbol{A_{jm}}$. Eqs. (5.2) – (5.7) show how the rotation matrix $\boldsymbol{A_{jm}}$ is determined and after $\boldsymbol{A_{jm}}$ is identified, we will obtain the least square error $E(\boldsymbol{A_{jm}})$. This error is the principal deciding parameter used with our classifier at a later stage.

$$E(\boldsymbol{A_{jm}}) \equiv \frac{1}{2}\sum_{i=1}^{n} w_i \left|\boldsymbol{k_{mi}} - \boldsymbol{A_{jm}}\boldsymbol{a_{ji}}\right|^2 \tag{5.1}$$

w_i is non-negative observation weight. The cost function can be rewritten as:

$$E\left(\boldsymbol{A_{jm}}\right) = \sum_{i=1}^{n} w_i - \sum_{i=1}^{n} w_i \, \boldsymbol{k_{mi}^T A a_{ji}} \tag{5.2}$$

Next, we rewrite the cost function as:

$$E\left(\boldsymbol{A_{jm}}\right) = \lambda_0 - tr(\boldsymbol{A_{jm}B_{jm}^T}) \tag{5.3}$$

with

$$\lambda_0 = \sum_{i=1}^{n} w_i \tag{5.4}$$

And $\mathbf{B_{jm}}$ is defined as:

$$\boldsymbol{B_{jm}} \equiv \sum_{i=1}^{n} w_i \boldsymbol{k_{mi}} \boldsymbol{a_{ji}^T} \tag{5.5}$$

tr is the trace of a matrix. The first useful solution to Eq. (5.3) was provided by Paul Davenport in [78]. The same approach is used in this paper to solve this problem and the Eq. (5.3) can be rewritten with the quaternion representation as:

$$E\left(\boldsymbol{A_{jm}(q)}\right) = \lambda_0 - \boldsymbol{q^T} K(\boldsymbol{B_{jm}})\boldsymbol{q} \tag{5.6}$$

where $K(\mathbf{B_{jm}})$ is the symmetric traceless 4 by 4 matrix as shown in Eq. (5.7) and $\boldsymbol{q}$ is a 4-D quaternion which represents the rotation transformation instead of rotation matrix $\mathbf{A_{jm}}$.

$$K(\boldsymbol{B_{jm}}) = \begin{bmatrix} \boldsymbol{B_{jm}} + \boldsymbol{B_{jm}^T} - I_{3\times3}tr(\boldsymbol{B_{jm}}) & \sum_i w_i \boldsymbol{k_{mi}} \times \boldsymbol{a_{ji}} \\ (\sum_i w_i \boldsymbol{k_{mi}} \times \boldsymbol{a_{ji}})^T & tr(\boldsymbol{B_{jm}}) \end{bmatrix} \tag{5.7}$$

It follows that the optimal quaternion is the eigenvector of $K(\mathbf{B_{jm}})$ with the maximum eigenvalue:

$$K(\boldsymbol{B_{jm}})\boldsymbol{q_{opt}} = \lambda_{max}\boldsymbol{q_{opt}} \tag{5.8}$$

$\boldsymbol{q_{opt}}$ is the optimal quaternion solution to Eq. (5.6) and λ_{max} is the maximum eigenvalue of matrix $K(\mathbf{B_{jm}})$.

After the optimal quaternion is obtained, the least square mean error when calibrating the two frames is determined. In this paper, we assign 8 wearable accelerometers to 8 Kinect segments. The 8 Kinect segments we consider are the right upper arm, right forearm, left upper arm, left forearm, right thigh, right lower leg, left thigh and left lower leg. These are the 8 possible body locations that a given sensor will be worn at. The eight errors between one sensor and eight body segments are used to determine the location of this sensor in the following section. Although we use the aforementioned 8 body segments for our experimental validation, our technique can operate with a larger number of body segments.

5.1.5 Auto-localization

Figure 5.3 shows the least square errors between the wearable sensors and the Kinect body segments. The nodes in the top show 8 wearable nodes that need to be

localized. The nodes in the bottom part of the figure represent 8 Kinect segments that are discussed in this paper. e_{jm} is the least square error between *jth* wearable sensor and *mth* Kinect segment solved in a window of observation as discussed in Section 5.1.4. The Kinect segments are shown in Table 5.3 of Section 5.2. For each wearable sensor, we have eight errors. The eight errors are the inputs for one cascade classification to determine the location of this sensor. If there are *n* wearable sensors on the human body, they are treated independently and the same cascade classifier will be used for all of them. In this chapter, 8 independent cascade classifiers are needed to determine the locations of 8 wearable sensors. In the following section, the operation of the classifier for one wearable sensor is discussed.

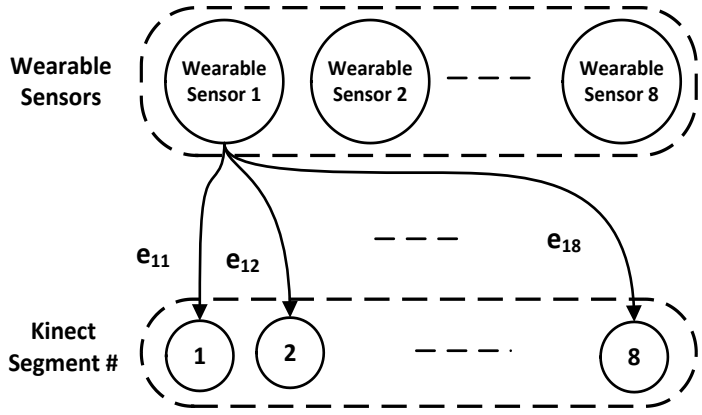


Figure 5.3 - The least square errors between eight wearable sensors and eight Kinect body segments. Reprint with permission from [73].

5.1.5.1 Cascade Classifier

To determine the location of an on-body sensor, a cascade classifier is proposed as shown in Figure. 5.4. In our approach, multiple windows are used to determine the

location of the sensor. Among all windows, some windows offer well pronounced motions and some other windows may not contain motion which will reduce their suitability to aid in localization of on-body sensors. Thus several windows must be considered over time and the notion of cascade classifier is used. Cascade classifier is widely used in the object tracking and face recognition [79, 80]. The cascade classifier has a series of decision nodes since one single decision node is not enough to determine the classification result. Each node will reject some non-target classification labels and the remaining potential candidates will be passed to the next node. In our case, a single decision node is not able to generate a single sensor location as it can reject only some non-target locations. Therefore, a cascade classifier is ideal for our application and each decision node is decision tree based classifier as discussed in Section 5.1.5.2. The input of the l^{th} decision node is error set ES_{jl} which is define to include the least square errors between the j^{th} accelerometer and all the remaining qualified Kinect segments. The qualified Kinect segments are the ones which are not rejected yet. The output will be the remaining qualified Kinect segment indexes after the non-target locations are rejected by this node. These indexes form the qualifying set for each node. The qualifying set for *lth* decision node for *jth* wearable sensor is defined as QS_{jl}. The following example shows how the cascade classifier works. To determine the location of *jth* accelerometer, the input of the first decision node ES_{j1} is $\{e_{j1}, e_{j2}, \ldots, e_{j8}\}$ since all the locations are candidate locations. Based on the decision tree based classifier which is discussed in section 4.3.2, if the locations 1 and 2 are rejected by this decision node. Then the output qualifying set QS_{j1} is $\{3, 4, \ldots, 8\}$. In the second decision node, the input ES_{j2} is $\{e_{j3}, e_{j4},$

…, e_{j8}} in which we only consider the qualifying body segments. Similarly, more decision nodes are cascaded until QS_{jn} only has one element which means 7 non-target locations are rejected by the former nodes and the only one remaining segment will be the target location of *jth* wearable accelerometer. For each decision node, the input error between accelerometer samples and corresponding Kinect segments samples is considered over a 1.5 seconds window. The window size is chosen as 1.5 seconds since it is the time duration for most of the daily activities (*e.g.* sit-to-stand and pick up a cup). The overlapping segment between two consecutive windows is set at 0.5 seconds.

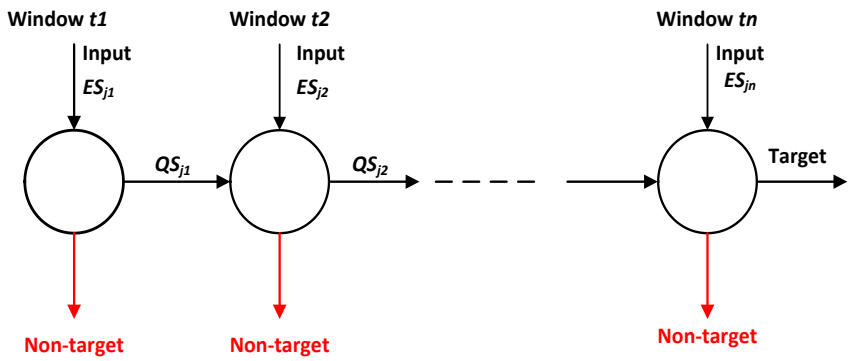


Figure 5.4 - Cascade classifier with multiple decision nodes for jth wearable accelerometer. Each cycle represents a decision tree based classifier at each time window. The decision tree based classifier is explained in section 4.3.2. Each classifier has its own input error set *ES*. ES_{jl} is the input error set for *lth* classifier of *jth* sensor. The output of each classifier is called qualifying set (*QS*) which may include multiple body segments that can be assigned to *jth* wearable accelerometer. The cascade will output the QS_{jn} as the target movement when the size of QS_{jn} becomes 1 and all the non-target locations are rejected. Reprint with permission from [73].

5.1.5.2 Decision Tree Classifier

Each decision node in the cascade classifier itself is a decision tree based classifier that will reject some non-target locations. Figure 5.5 shows the decision tree classifier which is the classifier for each decision node. The input of the *lth* decision node is the error set ES_{jl}. The minimum error from the error set is e_{min} and for each error e_{jl} in this set, we calculate e_{jl}/e_{min}. We consider this ratio instead of using the minimum error because when two or more Kinect body segments experience no movements, they will report similar errors. For example, if a user remains in sitting position and his lower body does not report significant motion, with a sensor attached to the right thigh, all lower body segments including right thigh, right lower leg, left thigh and left lower leg will have similar errors and the smallest is not necessarily choose the right thigh. The error should be sufficiently discriminative to assist the classifier choosing the correct body segment. If the result is smaller than a threshold, the corresponding location will be classified as a possible target location and the index will be provided to the next classifier. If the result is larger than or equal to the threshold, the corresponding segment will be classified as non-target location and will be rejected. The threshold is set at 4.5 which is determined experimentally

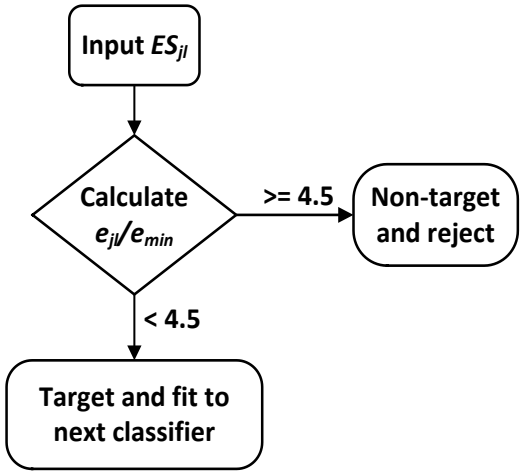


Figure 5.5 - Decision tree classifier. Reprint with permission from [73].

5.1.5.3 Weighting Method

From the problem formulation in Eq. (5.1), there is a weight parameter w_i for each frame of error calculation. If all the observation samples in one classifier window are of equal importance, all the w_i will have equal values in this window and all the values need to be normalized in this window. However, the confidence of the accelerometer observations is related to the speed of the movement. It is well known that the accelerometer measures the combination of the gravitational and dynamic acceleration [57] and the gravitational acceleration is useful information for our algorithm to track the rotations of body segments. If the dynamic acceleration is too large, our estimation error will increase. The raw weight for observation sample *i* is defined as:

$$w_i = \begin{cases} 0, \left| |f_{aj}^i| - g \right| > 0.5g \\ g^2/(g + \left| |f_{aj}^i|_i - g \right|)^2, \quad \left| |f_{aj}^i| - g \right| \leq 0.5g \end{cases} \tag{5.9}$$

g is the gravity constant. In Eq. (5.9), if the absolute value of difference between total acceleration amplitude and gravity constant is larger than 0.5g, the weight is set to 0. It means if the amplitude of dynamic acceleration is larger than 0.5g, the impact of dynamic acceleration is considered too high and we do not consider this sample. When the absolute value of difference is smaller than 0.5g, we use the sample but its weight is penalized based on how much it differs from the gravitational acceleration g.

Normalizing the raw weight in this window, we will obtain the weights as:

$$w_i' = w_i / \sum_{i=1}^{n} w_i \tag{5.10}$$

Figure 5.6 illustrates how weight adjusting method performs better than if equal weights are assigned to all samples for an arm stretch movement with combined fast and slow motion. The user is asked to perform the complete movement at a slow speed at first and then at a fast speed, followed by another slow speed movement. The sensor is attached to the right forearm. Figure 5.6 (1) shows the acceleration amplitude during this activity and Figure. 5.6 (2) shows our proposed weight adjusting method which gives the slow motion higher weight and the fast motion smaller weight since the gravity vectors are affected more by the dynamic motion acceleration during the fast motion. Figure. 5.6 (3) illustrates how weight adjusting method achieves smaller error between sensor measurement and target Kinect segment, and larger average error between sensor and non-target segments. This will in turn yield in better separation between target and non-target locations.

As was stated, we are using the rotations of gravity on the wearable accelerometer and the Kinect body segment to determine the location of the sensors. The

gyroscope can also give us the same information. There are three reasons why we do not use gyroscopes. First, we will need to perform integration over gyroscope readings to obtain rotation information which will introduce integration error impacting our localization algorithm. Second, the accelerometer consumes less power and it is readily available in most wearable sensors. Third, the gyroscope captures the rotation along the vertical direction of the earth while Kinect segment vectors do not offer this information. This difference will lead to poor performance if a match is done between these two modalities.

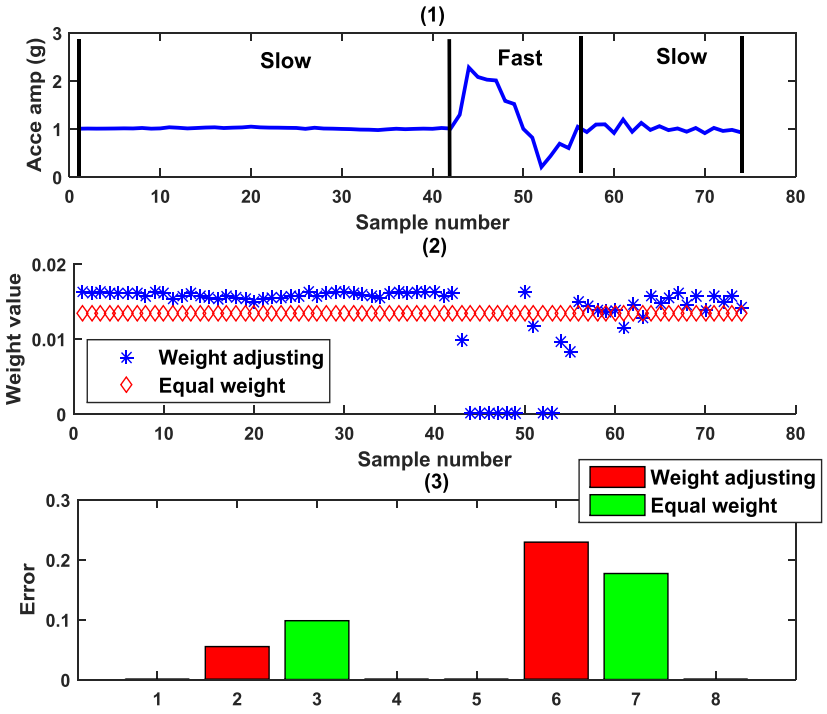


Figure 5.6 - (1). Acceleration amplitude of an arm stretch activity, including fast and slow motion. (2). Normalized equal weights and adjusted weights. (3). Errors for target location and average of non-target locations of using weight adjusting approach and equal weight approach. Reprint with permission from [73].

5.1.5.4 Kinect Occlusion Consideration

Kinect suffers from the occlusion (*i.e.* line of sight) problem when some parts of the body are not visible. If an occlusion for a joint occurs, the position data will not be correct, which will decrease the performance of our algorithm. To solve this problem, we look at the skeleton joints tracking state from the Kinect API. The API offers three states: 'tracked', 'inferred' and 'non-tracked'. When the tracking state is 'inferred', it means the data is calculated from the other tracked joints and the confidence in the data is low. The state 'non-tracked' means the joints are not available. In this paper, if there are any 'inferred' and 'non-tracked' state for any joint during the decision window, this decision window will not be considered and the algorithm will process the next window. The reason for such a strict constraint is that if the tracking status of one joint is not reliable, it may result in a wrong body segment vector which consists of this joint. The wrong body segment vector may result in a false positive that adversely affects the decision of our algorithm. In future work, we will improve our algorithm to also operate with 'inferred' states.

Figure. 5.7 (2) shows that during the kneeling movement with the left lower leg, the left foot joint is inferred for some samples and the least square mean errors between the left lower leg sensor and 8 Kinect segments are shown in Figure. 5.7 (1). Because of the effect of the occlusion of the left foot, the error between the left lower leg sensor and the left lower leg (#8) is much larger than the error with respect to the non-target segment right thigh (#5). This will possibly result in misclassification and hence it is very important to eliminate these illegal cases before the classification takes place.

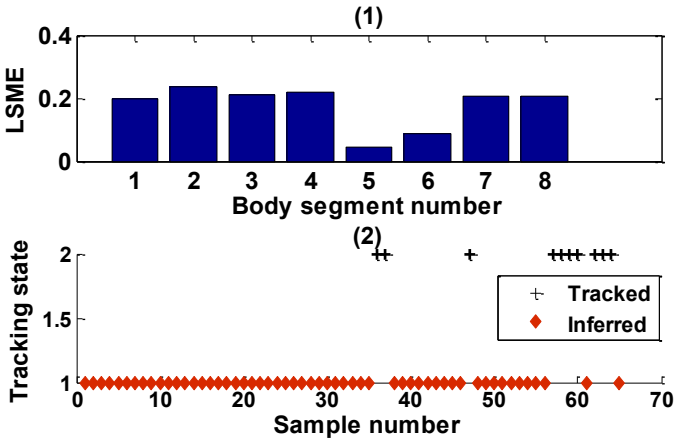


Figure 5.7 - (1) Least mean square errors between left lower leg sensor and 8 Kinect body segments. (2) Tracking state of left foot joint during the left lower leg kneeling activity. Reprint with permission from [73].

5.2 Experimental Setup

To test the effectiveness of our method, two experiments are designed. In the first experiment, we test our algorithm with a number of individual activities which take about 1.5 seconds to complete. The activities are listed in Table 5.2. For most activities, we list two labels: left and right. These labels refer to which side of the body is used to perform the activity. For example, 'Bend and grasp (right)' means this activity is performed by right arm. In the second experiment, two complicated daily tasks – cooking and playing basketball – are considered. For the first experiment, 5 subjects perform 10 repetitions for each activity. For the second experiment, the same subjects perform 3 minutes of each task 3 times in the experiment area. Table 5.3 shows the 8 sensor placements for our experiment and their corresponding index numbers.

Table 5.2. Daily activity list. Reprint with permission from [73].

Number	Activity	Number	Activity
1	Bend and grasp (right)	7	Sit to stand
2	Bend and grasp (left)	8	Stand to sit
3	Kick (right)	9	Arm lateral raise(left)
4	Kick (left)	10	Arm lateral raise(right)
5	Leg lifting (right)	11	Pick up phone (right)
6	Leg lifting (left)	12	Pick up phone (left)

Table 5.3. Sensor placement list. Reprint with permission from [73].

Number	Sensor location	Number	Sensor location
1	Right upper arm	5	Right thigh
2	Right forearm	6	Right lower leg
3	Left upper arm	7	Left thigh
4	Left forearm	8	Left lower leg

5.3 Experimental Results

5.3.1 Results of Simple Activities

Note that the durations of the activities we investigate are about 1.5 seconds and only one window is used to determine the location. Thus the outputs will be the outputs of our first stage classifier. To look at the discriminative ability of each activity for each

location, two classification performance metrics (precision and recall) are discussed. The definitions of precision and recall are as follows [81]:

$$precision = \frac{tp}{tp+fp} \tag{5.11}$$

$$recall = \frac{tp}{tp+fn} \tag{5.12}$$

where *tp* is the number of true positives, *fp* is the number of false positives and *fn* is the number of false negatives for location recognition. We have 8 locations with one sensor on each location and the objective is to find a matching between sensors and locations. The performance measures are considered for each location. If a sensor is attached at one location, and it is classified to this location, it would be considered as *tp.* If the sensor is attached to another location and it is assigned to this location, it would be considered *fp* for this location. If this sensor is classified as another location instead of the correct location, it would be considered as *fn.* For each location, the precision is the ratio between correctly recognized instances for this location and the summation of correctly recognized instances and the instances which are recognized as this location incorrectly by the algorithm. This measure provides how well the algorithm can separate the false positives that are confused with the correct location. The recall is the ratio between number of correctly recognized instances for one location and the summation of the number of correctly recognized instances and the number of instances which are belong to this location and recognized as other locations incorrectly. It measures how well the algorithm recognizes the correct sensor locations.

Figure. 5.8 shows the recall and precision for all the locations from the 12 different activities. The values are averaged for 5 subjects. From the three plots, we can see that

the recalls for all locations from all the activities are 100% which means our algorithm will not reject the true positives for the 12 daily activities. Note that for all the simple activities, we can only investigate one-stage classification as the movements are short (1.5 seconds) and a high recall rate is the most important metric since we do not want the true positive instances to be rejected at every stage. In Figure. 5.8 (a), for the Bend and grasp (right), the precisions of right upper arm (#1) and right forearm (#2) are both at 50%, which means only one additional false positive is detected. For this movement, right upper arm and right forearm cause false positives for each other since they have similar motions. The same result is observed for bend and grasp (left). For the first two movements, other locations have a smaller precision because when the sensor is attached to other locations, those body segments do not observe movements leading to potential confusion among these body segments. For the kick (right) and kick (left) activities in Figure. 5.8(a), only thighs (#5 or #7) and lower legs (#6 or #8) of the corresponding body side (right and left) have larger precision and all other locations have a similar small precision since they all do not move during the activities. For leg lifting (right) and left lifting (left) in Figure. 5.8 (b), right thigh (#5) and left thigh (#7) have large precision values, respectively. It means leg lifting will reject most of the false positive cases when the sensor is attached to the thighs. In these two activities, even though lower legs (#6 and #8) have major movement, they have small precision values. This is because in the leg lifting movement, the lower legs are almost parallel to the gravity. They don't have major change along the gravity, thus the gravitational change measured by accelerometer is very small. Since the gravitational acceleration change is

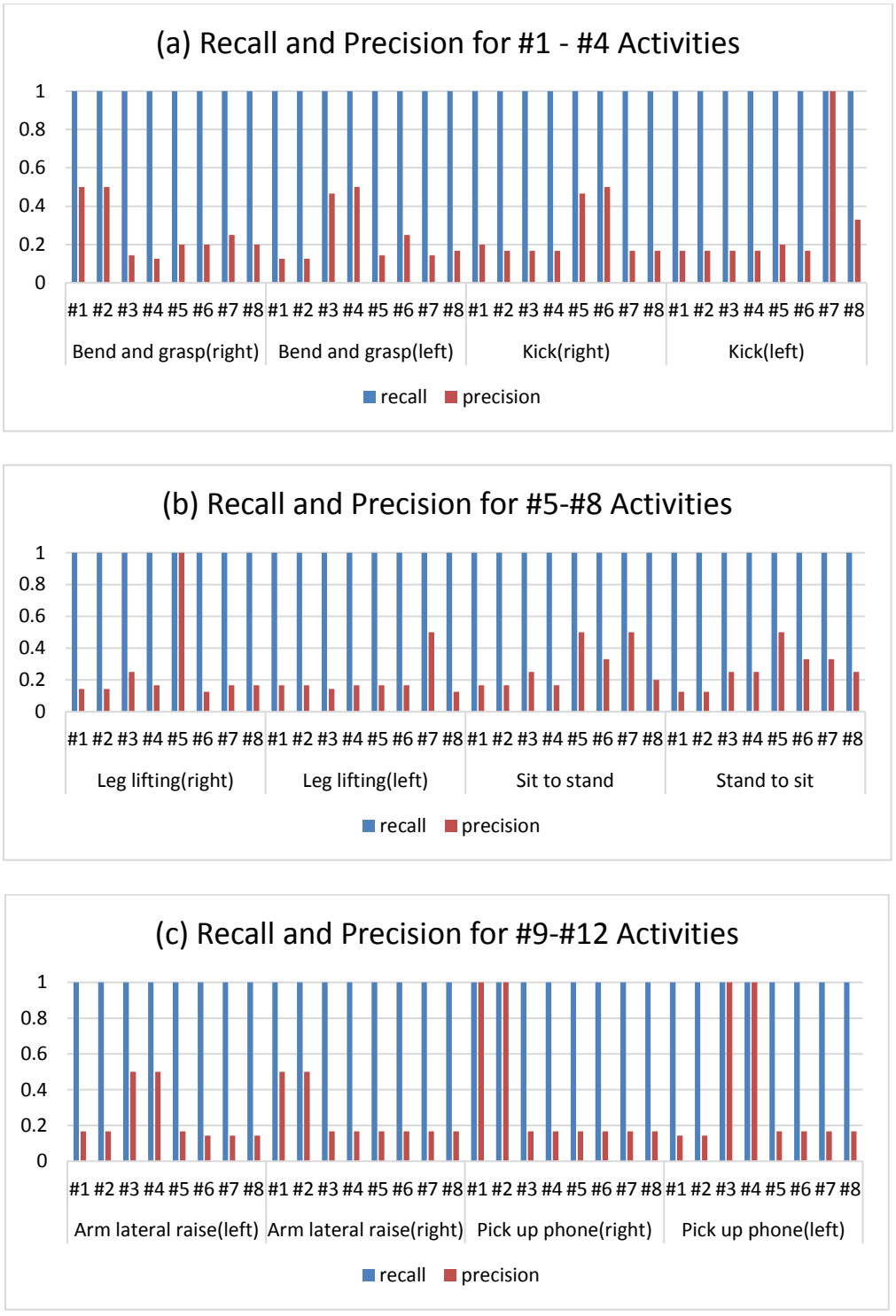


Figure 5.8 - Recall and precision for 8 locations from (a). #1-#4 activities (b). #5-#8 activities (c). #9 - #12 activities. Reprint with permission from [73].

the useful information to our algorithm and the force specific acceleration is considered noise, low precision values are achieved. For the sit to stand and stand to sit, the precisions of both thighs (#5 and #7) are bigger than the others due to the major motion occurring at the thighs and our algorithm will give much smaller least square errors between motion sensor and thighs than between motion sensor and the other body segments. This will result in rejection of the other body locations. Similarly, for sit to stand and stand to sit, the other body segments do not have major rotation along the gravity and it results in the poor precision.

In Figure. 5.8(c), for arm lateral raise (left) activity, left upper arm (#3) and left forearm (#4) have the same precision of 50%. During this activity, the forearm and upper arm have the same rotations along the gravity vector and they are the only false positive for each other. For the other 6 body locations, there are no motions during this activity and they are the false positives for others. The activity arm lateral raise (right) has the same explanation. For the pick-up phone activity, the precision for the forearm (#2 or #4) and upper arm (#1 or #3) are both 100%. It means this activity will determine the sensor location of upper arm and forearm without any false positives.

As discussed in Section 5.1.5.1, our algorithm uses a cascade classifier to continuously reject the non-target locations until only one remains and it will be determined as the target location. All the simple activities in this section can only construct one classifier and cannot offer the final classification results. To explore the ability of the cascade classifier, a varying number of sequences of simple activities are

cascaded manually and the average precisions are analyzed. Since all the simple activities offer 100% recall, any combination of them will also offer 100% recall.

Figure. 5.9 shows the average precision for different number of combined simple activities. The *X-axis* is the number of activities combined and Y-axis is the average precision. Different lines represent different body locations. From the figure, we can see that the precision change for different locations are very consistent. Also, a combination of 5 activities will achieve about 50% precision for all locations which means there is only one other location confused with the target location. A very sharp slope is observed from number of 10 to number of 11. It means when the activities increase from 10 to 11, there is very big increase in precision. Also, if all 12 simple activities are combined, all locations will get 100% precision which proves our algorithm will work well as long as enough good movements are provided.

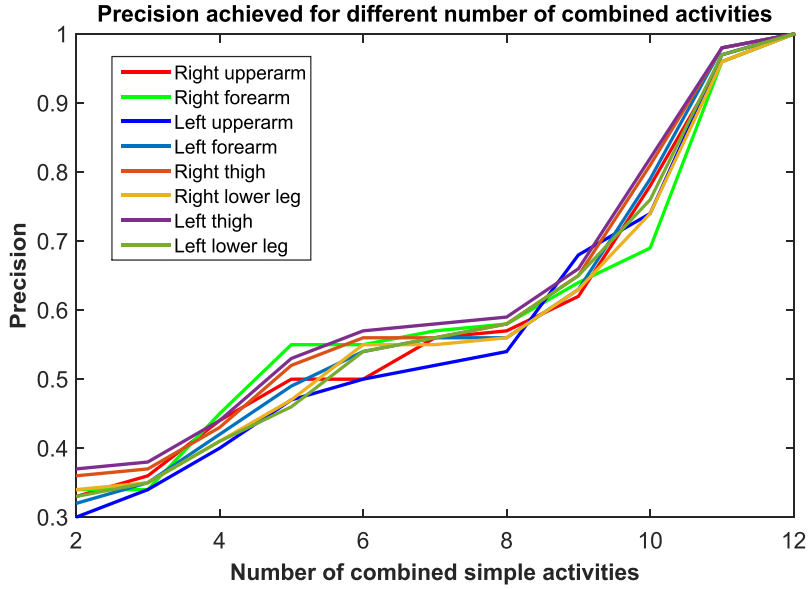


Figure 5.9 - Average precision for difference number of combined simple activities. Reprint with permission from [73].

5.3.2 Results of Complicated Daily Motion Tasks

To further evaluate how our algorithm performs for complicated motion tasks, each subject is asked to perform two complicated tasks (cooking and playing basketball) for 3 minutes for each. They repeat each task three times. For this experiment, recall is used to evaluate the performance of our algorithm. The reason why we look at recall is that for a final decision, either it is correct and not correct. For correct instances, we consider them as true positives and for incorrect instances, we consider them as false negatives. Therefore, the recall measures the ratio between the correct recognized instances and all the instances belong to this location.

Figure. 5.10 shows the recall of different body locations for cooking and playing basketball averaged on 5 subjects. From the figure, we can see our algorithm achieves good performance for both cooking and playing basketball. The average recall for both tasks is 82.56%. For the cooking task, the recalls for the upper body locations (right upper arm, right forearm, left upper arm and left forearm) are slightly bigger than the recalls for the lower body locations (right thigh, right lower leg, left thigh and left lower leg). This is because for the cooking task, there is more motion involvement from arms than from thighs and lower legs and this will result in better discriminative performance. For playing basketball, our algorithm offers similar performance for both upper body locations and lower body locations since they are both involved in a large amount of motion. Also, it is observed from the figure that our algorithm performs better for cooking than playing basketball. The main reason is that the motion involved in basketball is much faster than it is in cooking. Our weighting method partly addresses

this issue, but if there are too many bad samples, it will lead to slightly poorer performance.

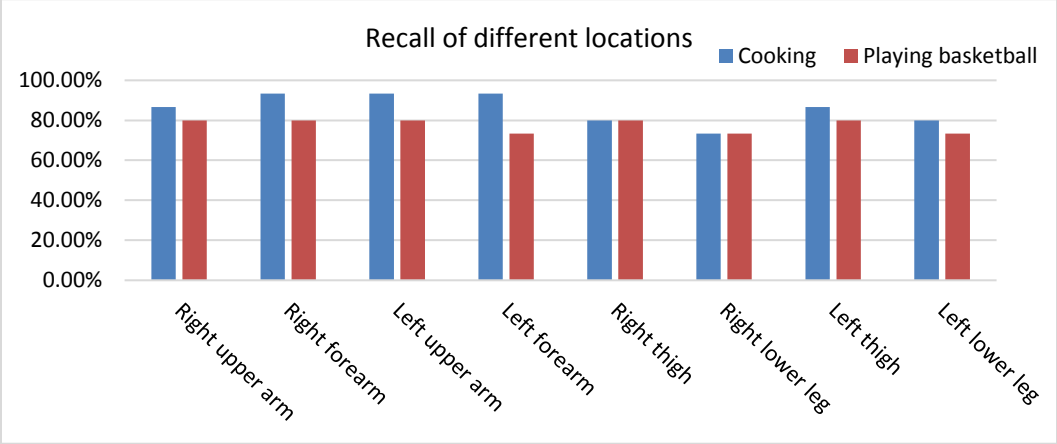


Figure 5.10 - Recall of different locations for cooking and playing basketball. Reprint with permission from [73].

Besides the final recall, we also look at how long it takes for the cascade classifier to achieve the final decision. Figure. 5.11 shows the time taken for the cascade classifier to achieve the final decision for different locations of these two different complicated motion tasks. The time is averaged from 5 subjects. From the figure, we can see that for playing basketball, the classifier achieves final decision for all locations in less than 60 seconds. This is because playing basketball consists of a lot of motions that can be used to reject non-target locations at this decision node. Although the performance is worse than cooking as discussed in Figure. 5.11, the time taken to achieve the final decision is less than cooking. For cooking, it takes longer for the four segments of lower body to achieve a final decision. The reason is that for the cooking experiments, there is much more upper body motion than lower body motion which makes it easier and faster to determine the sensor locations on the upper body.

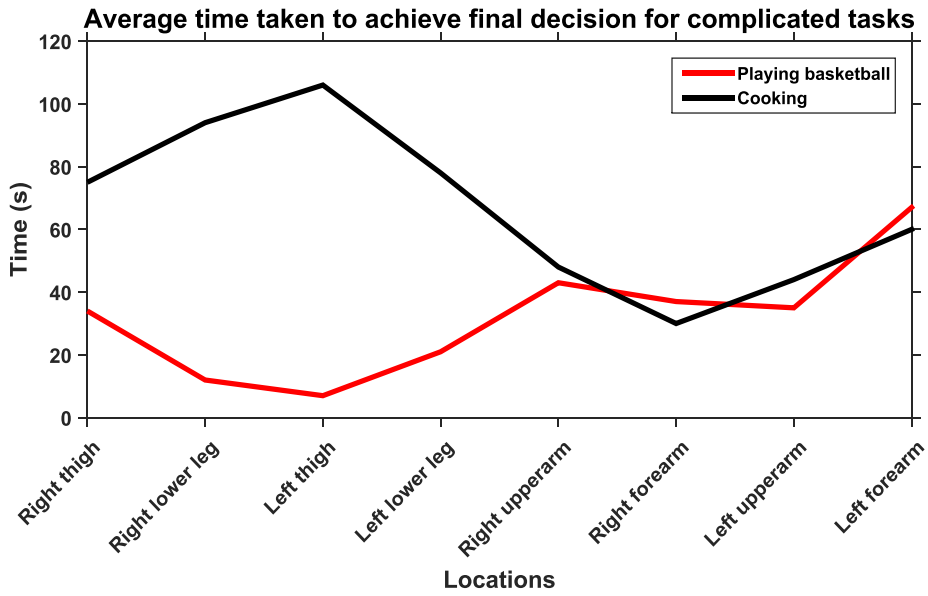


Figure 5.11 - Average total time taken for cascade classifier to achieve final decision. Reprint with permission from [73].

5.4 Conclusion

In this part, we proposed an effortless vision-assisted localization and calibration technique for wearable inertial sensors. A cascade decision tree based classifier determines the on-body sensor locations based on the least square errors obtained by solving the Wahba's problem between accelerometer and Kinect skeleton segment vectors. Our proposed weighting adjusting scheme and the vision occlusion consideration ensure our approach operates robustly. We evaluate our approach with two experiments: simple daily activities and complicated motion tasks. Our approach achieves 100% recall for simple actions and 82.56% recall for complicated motion tasks.

Wearable devices are important parts of IoT devices and the on body placement of wearable devices are important to develop robust algorithms for different

applications. The calibration of the placement of wearables is important and should be accomplished seamlessly. Fortunately, IoT settings provide opportunities to fuse information from environmental and wearable sensors. These opportunistic fusion techniques enable the calibration of one modality using other modalities. In our paper, camera and vision-based IoT devices help calibrate the placement of wearables. However, the same idea can be explored between other modalities of devices. How diverse sensors can benefit from similar techniques would be an interesting topic to investigate.

In our research, we assume the wearable data and vision device data can be extracted for the same person and processed in the cloud. However, in reality, if multiple individuals are appearing in front of the camera, challenges associated with determining who should be assigned to which wearable set must be addressed. Although this is out of scope of our paper, a possible solution for extension of Wahba's to include a larger number of sensors and body segments from multiple users.

6. ORIENTATION INDEPENDENT ACTIVITY/GESTURE RECOGNITION*

In order to address the challenges of sensor orientation displacement, speed variation and activity/gesture inconsistence, we propose an orientation independent and speed independent activity/gesture recognition algorithm in this chapter that works with arbitrary sensor orientation and different movement speed. A novel feature set is proposed that functions irrespective of how the sensors are oriented. A template refinement technique is proposed to determine the best representative segment of each gesture thus improving the recognition accuracy.

6.1 Methodology

6.1.1 Definitions

In this paper, a customized 6-axis wearable IMU is used for the study. The IMU has a 3-axis accelerometer and 3-axis gyroscope. The 3-axis accelerometer measures gravity and dynamic acceleration which is caused by the motion. The 3-axis gyroscope measures 3-axis angular velocity. The symbols used in this paper are defined in Table 6.1. At each time corresponding to sample number *i*, both accelerometer and gyroscope generate a 3 by 1 vector $\boldsymbol{a_i}$ and $\boldsymbol{w_i}$. They are the raw sensor data input for our system. $\boldsymbol{A}$ and $\boldsymbol{W}$ represent time series of accelerometer and gyroscope readings, respectively. $\widetilde{I_J}$ denotes a set of feature vector instances constructed during the training phase. I_{ij} refers to *i-th* feature vector instance of *j-th* class. The construction of I_{ij} is introduced in

*Reprinted with permission from "Orientation Independent Activity/Gesture Recognition Using Wearable Motion Sensors" by Jian Wu and Roozbeh Jafari, 2018. IEEE Internet of Things Journal, ©2014 by IEEE.

Section 6.1.3.A. A_i denotes the amplitude of acceleration $\boldsymbol{a_i}$ at time *i*. $\boldsymbol{M}$ is a time series of acceleration amplitude with *n* samples.

Table 6.1. Symbol definitions. Reprint with permission from [82].

Symbol	Definition	Explanation
$\boldsymbol{a_i}$	$[a_{ix}, a_{iy}, a_{iz}]$	3 by 1 vector, accelerometer readings along x-axis, y-axis and z-axis at time *i*
$\boldsymbol{w_i}$	$[w_{ix}, w_{iy}, w_{iz}]$	3 by 1 vector, angular velocity along x-axis, y-axis and z-axis at time *i*
$\boldsymbol{A}$	$\boldsymbol{a_1}, \boldsymbol{a_2}, \cdots, \boldsymbol{a_i}, \cdots, \boldsymbol{a_n}$	Time series of accelerometer readings with *n* samples
$\boldsymbol{W}$	$\boldsymbol{w_1}, \boldsymbol{w_2}, \cdots, \boldsymbol{w_i}, \cdots, \boldsymbol{w_n}$	Time series of angular velocity with *n* samples
$\widetilde{I_j}$	$\{I_{ij}, i = 1,2, \cdots n\}$	A set of instances of *j-th* class constructed during the training phase; I_{ij} represents *i-th* feature vector instance of *j-th class*
A_i	$\|\boldsymbol{a_i}\|$	The magnitude of *i-th* accelerometer reading
$\boldsymbol{M}$	$A_1, A_2, \cdots, A_n$	Accelerometer magnitude series

6.1.2 Overview

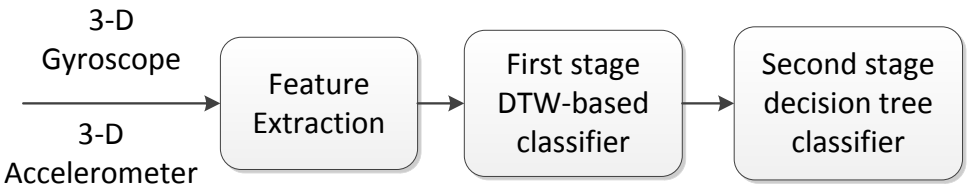


Figure 6.1 - System diagram. Reprint with permission from [82].

Figure. 6.1 shows our proposed recognition technique which addresses all aforementioned practical challenges. A novel orientation independent time series feature set is extracted from the gyroscope data by integrating the 3-axis angular velocity during a short period covering one action, gesture or human movement. Then a first stage

DTW-based classifier is used to recognize the gesture/activity. Intuitively, we consider the series of total angle change during this period. Most activities have the same series of angle change irrespective of the speed at which the activities are performed. Thus, the first stage classification is orientation independent and speed independent because the changes are independent of the sensor orientation and movement speed. The inconsistent segments are also analyzed and discarded by applying zero-padding DTW in this stage. However, our first stage classifier cannot distinguish the reversible activities (*e.g.*, sit-to-stand and stand-to-sit) because they have the same angle changes. We further propose a second stage decision tree classifier to distinguish the reversible activities by determining the magnitude of the acceleration which is also orientation independent. The second stage decision tree classifier is only deployed for the reversible activities and is not required for many activities such as hand gestures if they are not reversible.

6.1.3 Feature Extraction

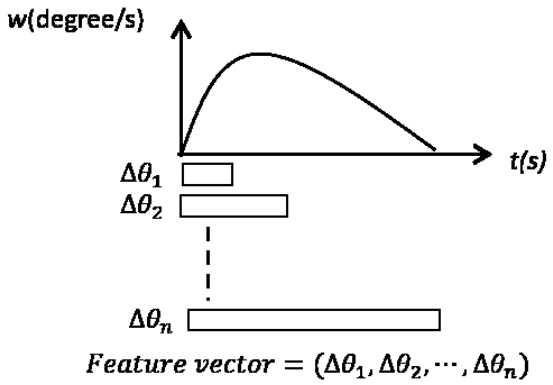


Figure 6.2 - Feature vector creation. Reprint with permission from [82].

Human movements or gestures can be classified by a feature set derived from the total angle change observed on each body segment, independent of orientation and

speed. The total angle change will be an integration of the total angular velocity over a short period of time (*i.e.*, covering one activity). Given an angular velocity time series $\boldsymbol{W}$ with *n* samples, the total angle change $\Delta\theta_n$ during the whole period can be obtained. However, only one feature value is not enough to uniquely describe the entire movement. Therefore, we construct a feature vector for a movement. This feature vector is constructed with total angle changes during different time periods. Figure. 6.2 provides an illustrative example for this feature set . $\Delta\theta_i$ elements in the figure are calculated as in (6.1).

$\Delta\theta_i =$

$$\sqrt{(\sum_{j=1}^{i} w_{jx} * \Delta t)^2 + (\sum_{j=1}^{i} w_{jy} * \Delta t)^2 + (\sum_{j=1}^{i} w_{jz} * \Delta t)^2} \quad (1)$$

Δt is the time duration between two samples, which is the reciprocal of the sampling frequency. $\Delta\theta_1$ is the total angle change during the first Δt seconds. $\Delta\theta_2$ is the total angle change during the first $2\Delta t$ seconds and $\Delta\theta_n$ represents the total angle change during the first $n\Delta$t seconds. A feature vector of a movement is called an instance of this movement. This instance captures important rotation characteristics and details during the movement. In the feature vector, all *n* angle changes are speed and orientation independent. However, the size of the feature vector varies because the length of a movement is not likely fixed and can be shorter or longer. This challenge of varying length of the movement is addressed by the DTW.

6.1.4 First-stage DTW-based Classifier

6.1.4.1 DTW with auto-segmentation

DTW is a template matching algorithm for measuring similarity between two time series with different durations [53, 83]. By investigating the warping distance, the algorithm can perform the segmentation automatically which will be an inherently challenging task for classic classification algorithms. Given two time series $X = x_1, x_2, \cdots, x_i, \cdots, x_n$, and $Y = y_1, y_2, \cdots, y_i, \cdots, y_n$, the cumulative distance is calculated as in (6.2).

$$D(x_i, y_j) = d(x_i, y_j) + min\begin{cases} D(x_{i-1}, y_j) \\ D(x_i, y_{j-1}) \\ D(x_{i-1}, y_{j-1}) \end{cases} \tag{6.2}$$

$d(x_i, y_j)$ is the distance between x_i and y_j. Any suitable distance function can be used to estimate this, and we use the following distance function in our investigation.

$$d(x_i, y_j) = \|x_i - y_j\| \tag{6.3}$$

By recording the minimum cumulative distance path, *D*, the warping path between two signals is obtained. In traditional classification paradigms, we first perform the segmentation so that the feature extraction and classification are performed for a certain segment. One popular segmentation technique is sliding window, in which a segment of signal in a fixed-size window is classified. This window moves through the whole incoming signal and time series in each window is classified as a certain target movement or non-target movement. However, due to the speed variation and the fact that different activities/gestures have different lengths, a fixed-size window cannot cover the exact duration of an activity/gesture. Thus, the accuracy of classification will be

negatively impacted. In this paper, DTW is applied to complete the segmentation. A feature vector is constructed in a window with size that is slightly larger than the longest activities/gestures and the segments of potential activities/gestures within this window will then be determined. Subsequently, the potential segments are supplied to the DTW. In this paper, we construct the feature vector in a window whose size is set to 2.5 seconds, about 1.5 times of the longest movement duration (sit-to-lie) for activity recognition. For gesture recognition, the window size is chosen as 14 seconds, which is 1.5 times of the longest gesture duration (drinking coffee). This guarantees that the time series feature vector covers all target movements even if they are performed at a slower speed. From the cumulative distance table between the template and the incoming time series feature vector, the matching part will be selected, thereby realizing segmentation automatically.

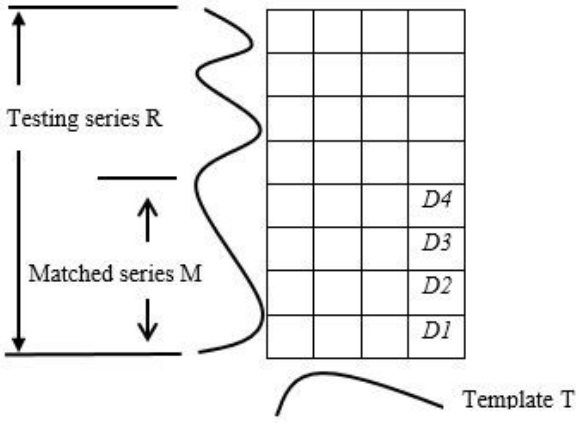


Figure 6.3 - Auto-segmentation by DTW. Reprint with permission from [82].

Figure. 6.3 shows an example of the auto-segmentation function of DTW. Assume that the template T has m samples and that the incoming feature series R (incoming unknown movement) has n samples, where $n > m$. The table in Figure. 6.3 provides the cumulative distance table D, as described in (2), of the two series [84].

The minimum cumulative distance, without loss of generality as the m^{th} column in the distance table, is considered as the warping distance and the corresponding row index will be the last sample of the matched series M within R. The warping distance and index are defined in (6.4).

$$\left[D_{warp}, index\right] = \min(D_i), \forall i \in [1, n] \tag{6.4}$$

D_i is the i^{th} element in the m^{th} column in the distance table. In the example in Figure. 6.3, D_4 will be selected as the D_{warp}, which is the warping distance between template T and series M.

6.1.4.2 Template selection

Selecting multiple templates with larger variations covers more cases and may provide a better accuracy for the DTW. However, this increases the computational complexity of the signal processing. Considering the resource constrained nature of the processing units in wearable computers, we only choose one template for each activity. This assumption however does not necessarily need to be enforced and one may choose to consider multiple templates representing each movement. The template for j-th movement/gesture is chosen from a collection of n instances of the movement/gesture according to the criterion in (6.5).

$$T = \underset{I_{ij} \subseteq \tilde{I}_j}{\operatorname{argmin}} \left(\sum_{k=1}^{n} D\left(I_{ij}, I_{kj}\right) \right) \tag{6.5}$$

Where I_{ij} represents *i-th* feature vector instance of class *j*. All these instances are constructed during the training phase with annotated activity/gesture data. *D* represents the DTW distance. The selected template is essentially closest to all the other instances of the movement and serves as the best representative template.

6.1.4.3 Threshold choice based on maximum-margin hyperplane

To distinguish the target movements from the non-target movements, a threshold is chosen for the DTW distance. If the threshold is too tight, certain target movements may not be detected resulting in a higher false negative rate. If the threshold is too loose, non-target movements may be classified as target leading to a large false positive rate. In this paper, we use a 5% maximum-margin hyperplane (MMH) as the threshold [85]. The MMH is the best separation margin between two classes. The 5% MMH is explained in Figure. 6.4. The *X-axis* represents movement instances (and their indices) and the *Y-axis* shows the DTW distance between target movement template and movement instances. The black dots and white dots represent DTW distances of non-target and target instances, respectively. The *Y-axis* value of the top red line is the mean of the smallest 5% distance samples between target movement template and non-target instances. The bottom black line is the mean of the largest 5% distance samples between the target movement template and target movement instances. The MMH is the average of these two lines represented as the dash-dotted line. We choose the mean of 5% instead of the largest or smallest sample to eliminate the effect of outliers.

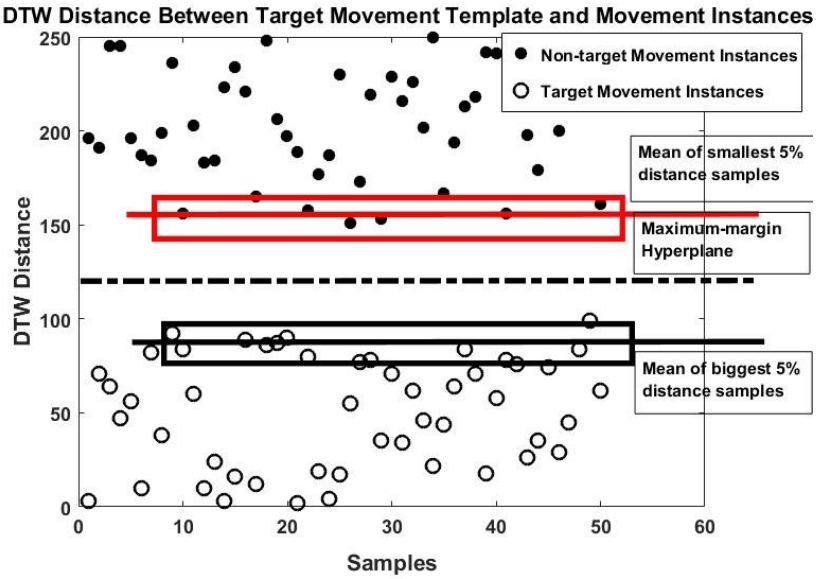


Figure 6.4 - Maximum-margin hyperplane definition. Reprint with permission from [82].

6.1.5 Inconsistent Segment Analysis and Star-padding DTW

For the activities of daily living considered in this paper, there are typically no inconsistent segments during movements since the durations of activities are typically short. For the gestures, however, like "Drinking coffee", "Picking up cell phone" or "Eating" certain inconsistent segments are present. The algorithm proposed in this section detects the inconsistent segments. Once we determine the inconsistent segment in the template, the star-padding DTW is used to complete the recognition [86]. The samples of inconsistent segment is replaced by special value ('*'), which has zero distance with all other samples in distance matrix. This will eliminate the effect of inconsistent segments when calculating the warping distance.

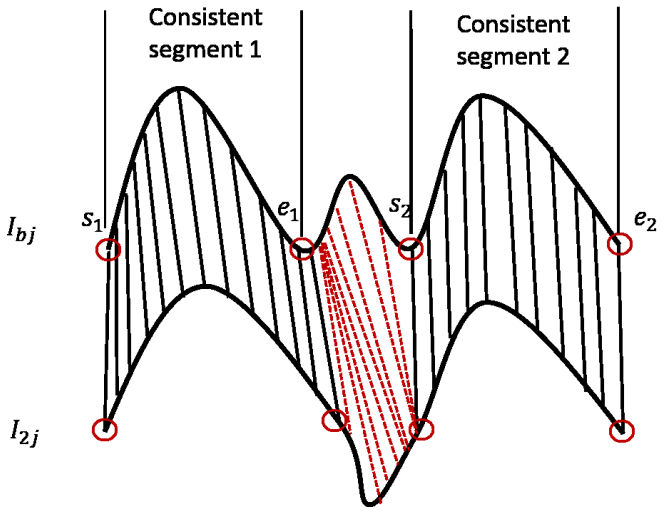


Figure 6.5 - Inconsistent analysis example. Reprint with permission from [82].

The first step is to identify consistent and inconsistent segments. Figure. 6.5 shows an example of the inconsistent movement analysis. In this figure, there are two instances I_{bj} and I_{2j} from *j-th* class. Two consistent segments and one inconsistent segment exist among these two instances. The consistent segment 1 is determined by the starting point s_1 and the ending point e_1 and the consistent segment 2 is determined by the starting point s_2 and the ending point e_2. The solid black lines that connect samples of the two instances show the warped samples of the consistent segments after DTW is applied. The dashed red lines show the warped samples of the inconsistent segment after DTW is applied. In the consistent segments, the warped samples from the two instances are close to each other and thus, the distances between all warped samples should be small. On the other hand, the warped samples of the inconsistent segment could be much different and thus, the distances between all warped samples could be larger. Our goal is to determine all the consistent segments guided by the DTW sample-wise distance.

To formally define the objective, for a given set of movement instances $\widetilde{I_j}$, we want to determine a set of consistent segment sample indices $\boldsymbol{F}=$ $\{(s_1, e_1), (s_2, e_2), \cdots, (s_i, e_i), \cdots ((s_n, e_n))\}$ for a beacon instance I_{bj}. s_i and e_i represents starting sample index and ending sample index of *i-th* consistent segment of the instance. In our approach, a random instance is picked as the beacon instance I_{bj}. I_{bj} is the beacon instance for *j-th* class. After applying DTW between beacon instance and any other instance, the sample distance along the warping path of the consistent segment should be small and consistent while the sample distance along the path of the inconsistent segment could be large and inconsistent. Let d_{ik} denote sample distance between *i-th* sample of beacon instance and the warped sample from *k-th* instance. It is possible that *i-th* sample from beacon instance is warped to several samples of *k-th* instance. In this case, the first warped sample distance is chosen as d_{ik}. Now that we obtain the sample distances between the beacon instance and all the other instances, a unique distance for *i-th* sample of the beacon instance is calculated as in (6.6).

$$d_i = \frac{\sum_{k=1}^{n} d_{ik}}{n} \tag{6.6}$$

In order to determine $\boldsymbol{F}$, an unsupervised-learning technique hierarchical clustering is applied [87]. Since all d_i's from consistent segments should be small and consistent, they will be clustered to the same cluster while all d_i's from inconsistent segments will not be grouped immediately with consistent d_i's. In the meantime, for grouping and clustering, a spatial constraint must be enforced to avoid clustering samples that have

ALGORITHM 6.1 CONSISTENT SEGMENTS DETECTION

Input: a set of movement instance $\widetilde{I_j}$;

Output: a set of consistent segment ***F***;

1: Pick an random instance from $\widetilde{I_{ij}}$ as I_{bj};

2: **for** *i* from 1 to n

3: do DTW(I_{bj}, I_{ij});

4: **end**

5: **for** all samples in I_{bj}

6: Calculate sample distance d_i according to (6.6);

7: **end**

8: perform hierarchal clustering for all d_i according to distance function (6.7);

9: the largest clusters with small intra-class distances correspond to consistent segments and are used to construct ***F*** vector;

small distance but come from different segments of the movement. For example, the *4-th* sample should be clustered with *5-th* sample before it will be clustered with *216-th* sample if all of them are from the same consistent segment. Therefore, we refine the distance function for clustering to accommodate this objective:

$$f(d_i, d_j) = (d_i - d_j)^2 + \alpha * |i - j|^2 \quad (6.7)$$

In (6.7), the first term indicates that all distances from consistent segment tend to be small and the second term adds a spatial constraint. α is a weight parameter that controls the spatial constraint for various applications. After the clustering is completed,

the largest size clusters that exhibit small intra-cluster distances are used to determine the s_i and e_i for each consistent segment to construct the $\boldsymbol{F}$ vector. The samples in each selected cluster should come from the same segment and the average linkage distance for this cluster should not be large. This algorithm is summarized in Algorithm 6.1.

6.1.6 Second Stage Decision Tree Classifier

With the first stage DTW distance threshold based classifier, we can classify different activities except for the reversible instances. For example, sit-to-stand and stand-to-sit have similar angular change patterns, and they will exhibit small DTW distance to each other considering our proposed feature set. In order to distinguish between the reversible movements, the amplitude of the accelerometer reading is considered.

The accelerometer data is often noisy and therefore, before calculating the amplitude, a 5Hz low pass filter is applied to the raw acceleration samples.

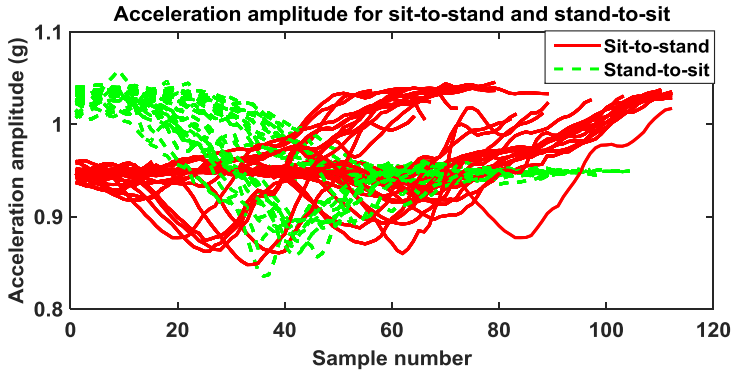


Figure 6.6 - Acceleration amplitude for sit-to-stand and stand-to-sit. Reprint with permission from [82].

Figure. 6.6 shows the amplitude of the acceleration for sit-to-stand and stand-to-sit movements of a thigh sensor from the auto-segmentation. For the sit-to-stand, the leg has an upwards acceleration first which is against the gravity and thus leading to an acceleration whose amplitude is smaller than 1*g*. For the end of sit-to-stand, there will be an acceleration downwards which is in the same direction as gravity. This results in an acceleration whose amplitude is larger than 1*g*. To distinguish the sit-to-stand from stand-to-sit, we look for the order of the maximum peak and minimum valley. For sit-to-stand, the maximum peak happens after the minimum valley while for stand-to-sit, the maximum peak occurs first followed by the minimum valley.

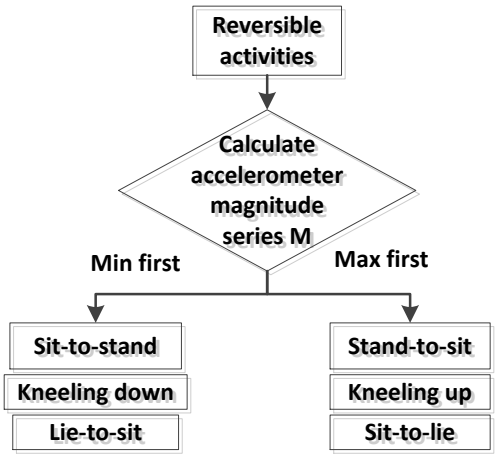


Figure 6.7 - Second stage decision tree classifier. Reprint with permission from [82].

Similarly, we can distinguish between the kneeling down and kneeling up and between the sit-to-lie and lie-to-sit by looking at the thigh sensor and ankle sensor, respectively. The second stage classifier based on decision tree is shown in Figure. 6.7. The reason why decision tree is applied is that it is easy to interpret. For example, it is

easy to understand when an accelerometer magnitude peak happens first, it is stand-to-sit; while when an accelerometer magnitude valley happens first, it is sit-to-stand. The three sets of reversible activities are classified by observing the order of occurrence for the maximum peak and the minimum valley samples. Note that for the gesture recognition task, there are no reversible movements and thus the second stage classification is not necessary.

6.2 Experimental Setup

We evaluate our approach using two different example applications: activity recognition of daily living and hand gesture recognition. For the activity recognition of daily living experiment, two sensors were worn by 10 participants. One was attached to the thigh and the other one was attached to the ankle. While for the hand gesture recognition application, 6 participants were enrolled with one sensor attached to the user's wrist. For both experiments, the users were asked to place the sensors in 6 different orientations (spanning 360 degrees), and perform 20 repetitions of each activity/gesture at each orientation. Each activity/gesture was performed at a slow speed for the first two orientations, at a normal speed for the middle two orientations and at a faster speed for the last two orientations. While the proposed approach is speed independent for most activities and gestures, the activities of walking and running are special cases. Due to biomechanical reasons, different speeds of walking and running have different angle rotations, so our approach would consider them as different movements. The participants performed these two movements at the normal speed for 6 orientations. The sampling frequency was set to 200Hz. The activities of daily living are

listed in Table 6.2 and the hand gestures are listed in Table 6.3. During the data collection, a camera recorded the movements along with the sensors. Both the sensor data and video data were synchronized. Then a visualization tool is used to segment and annotate the collected data [35]. The tool displays the sensor data and video data at the same time, hence segmentation can be performed and the data is annotated based on the video information. This segmentation serves as the gold standard.

In order to estimate the gravity vector and the horizontal direction vector to transform the accelerometer data to a global frame, a certain duration, such as a 10 second time window is required in certain existing approaches [56, 58, 59]. Their methods are good for the continuous movements or postures, such as sitting, lying, walking, and running, and do not perform well for the transitional movements since gravity has a major effect on some transitional movements. Thus, we compare our approach for the task of activity recognition with a pocket cell phone based orientation independent approach proposed by Sun *et al* [88]. The authors of that investigation extract 22 features from 3-axis accelerometer data and the magnitude of the accelerometer data, which include mean, variance, correlation, FFT energy and frequency domain energy. Since our approach uses two sensors, we extract 22 features from each sensor to construct the feature vector. Sun's approach is unable to do the segmentation and it uses a windowing technique that does not necessarily capture the beginning and the end of movements at the beginning and the end of the windows. By comparison, we use the camera labeled annotations to serve as the beginning and the end

of each window for their approach. This improves the performance of Sun approach. The LibSVM in Weka is used as the classification algorithm [89].

Table 6.2. Activities list. Reprint with permission from [82].

Activity Number	Activity	Activity Number	Activity
1	Sit-to-stand	6	Kneeling up
2	Stand-to-sit	7	90degree-turn
3	Sit-to-lie	8	Walking
4	Lie-to-sit	9	Running
5	Kneeling down		

We design four experiments to evaluate our approach. In the first experiment, we analyze the performance of auto-segmentation of our approach and determine if the beginning and the end of movements are detected correctly. The second experiment evaluates the subject dependent classification performance, in which 10-fold cross validation is used to evaluate the dataset for each person separately [90]. In the third experiment, the subject-independent classification performance is evaluated with the leave-one-subject-out testing method. There are 4 classification performance metrics in our paper, which are accuracy, precision, recall and F-score [91]. In the fourth experiment, the performance of our inconsistent movement analysis algorithm and template refinement is studied.

6.3 Experimental Results

6.3.4 Subject Dependent Classification Results

In this experiment, 10-fold cross validation is used for each subject separately and the classification accuracy is analyzed.

Table 6.3. Gestures list. Reprint with permission from [82].

Number	Gesture	Number	Gesture
1	Wave goodbye	5	Eating with spoon
2	Hand shake	6	Eating with hand
3	Picking up cell phone	7	Eating with chopstick
4	Drinking coffee		

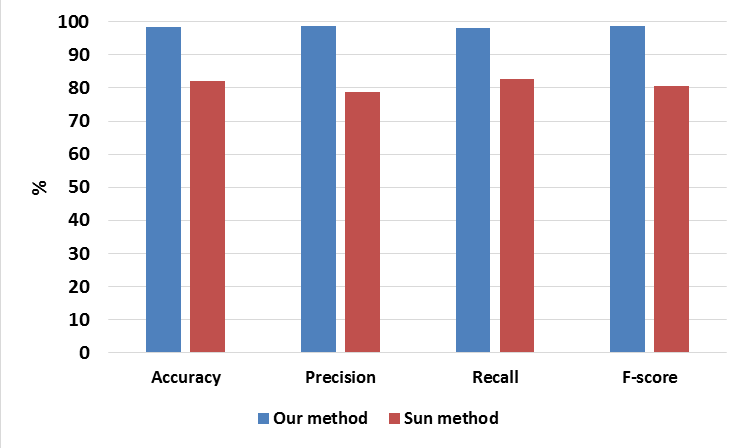


Figure 6.8 - Subject dependent classification results for activity recognition task. Reprint with permission from [82].

Figure. 6.8 shows the classification performance of our approach compared to Sun approach. All the results are calculated as the average for all 9 activities. From the

figure, we can see our approach achieves a very good performance in accuracy, precision, recall and F-score for the subject dependent testing compared to Sun's method. Our approach offers 98.52%, 98.62%, 98.21% and 98.65% for accuracy, precision, recall and F-score, and the improvements of these metrics compared to Sun method are 16.41%, 18.67%, 15.48% and 18.01% respectively.

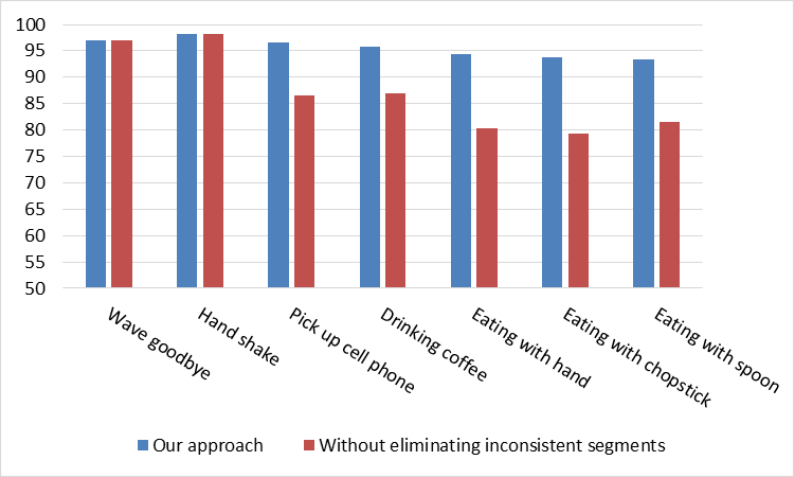


Figure 6.9 - Subject dependent classification results for hand gesture recognition task. Reprint with permission from [82].

Figure. 6.9 shows the subject dependent classification accuracy for hand gesture recognition task when we do and we do not eliminate the inconsistent segments. From the figure, we can see that for both 'Wave goodbye' and 'Hand shake', our approach achieves good performance and it is similar to the case when we don't consider the inconsistent segments and do not eliminate the effect of those segments. For 'Picking up cell phone', 'Drinking coffee' and three different styles of 'eating', our approach achieves better performance than the same approach without eliminating the inconsistent

segments. Specifically, our approach shows ~10% accuracy improvement. This is because 'Picking up cell phone', 'Drinking coffee' are similar to each other and they both have inconsistent segments in middle. Similarly, the three types of 'eating' are similar to each other and they all have inconsistent segments in middle. If the inconsistent segments are considered to be part of the whole gesture, the effect will cause confusion between these gestures. Since Sun approach is proposed for activity recognition using cell phone, we do not compare to their approach for hand gesture recognition.

6.3.5 Subject Independent Classification Results

Different human subjects may perform the same activity in a slightly different way. The subject independent classification analysis tests how robust the recognition system is with respect to different subjects. In the subject independent test, the leave-one-subject-out testing method is applied.

Figure. 6.10 shows the precision and recall for different activities in the subject independent test. From the figure, we observe that our approach outperforms the Sun approach for most of the activities. Only for 90degreeTurn, walking and running, Sun approach achieves similar performance to ours. One possible reason is that the frequency domain features used by Sun approach are good for discriminating the periodic movements (e.g. walking and running). We noted that our classification precisions and recalls for sit-to-lie and lie-to-sit are much lower than for the other activities. This is because one of the human subjects performed the sit-to-lie and lie-to-sit activities in a very different manner compared to the other subjects.

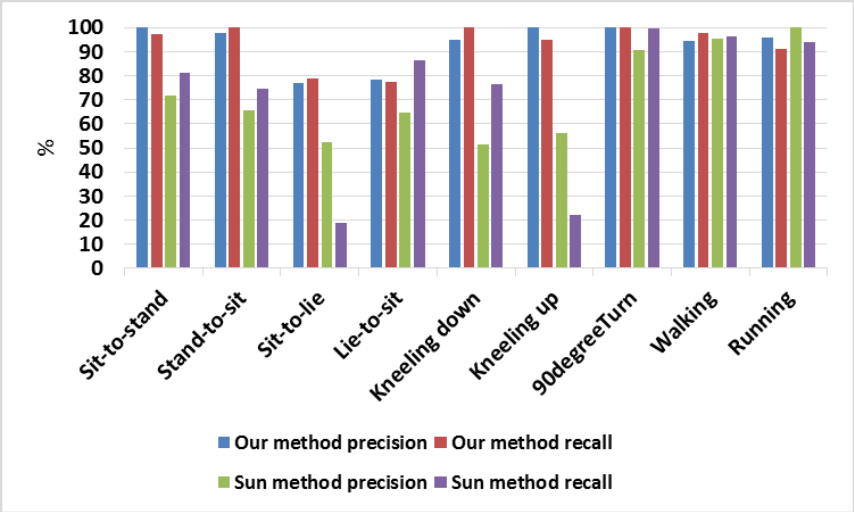


Figure 6.10 - Precision and recall for different activities for the subject independent test. Reprint with permission from [82].

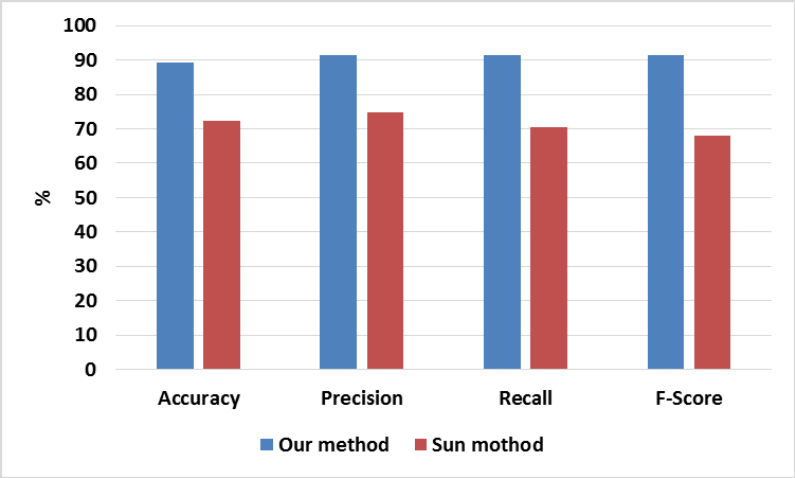


Figure 6.11 - Subject independent classification results for activity recognition task. Reprint with permission from [82].

Figure. 6.11 shows the subject independent classification results for the activity recognition task. The figure shows our method achieves much better performance than Sun's method. It indicates that our method is more robust to subject variation. The reason may be that the activities chosen are well distinguishable from each other. The

small variation caused by the subjects does not affect our algorithm too much. The improvements of our method with respect to accuracy, precision, recall and F-score are 17.14%, 16.27%, 19.86% and 23.42% respectively.

Figure. 6.12 shows the subject independent accuracy for the gesture recognition task. As shown in the figure, our approach achieves good performance for all seven hand gestures and our approach performs much better when the inconsistent parts are eliminated for both 'Picking up cell phone', 'Drinking coffee' and three different types of 'eating'.

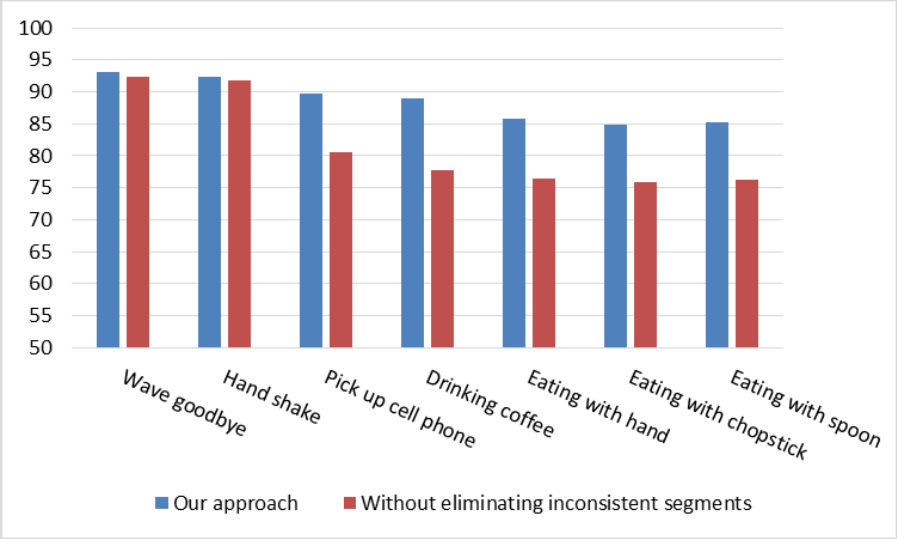


Figure 6.12 - Subject independent accuracy for gesture recognition task. Reprint with permission from [82].

6.3.6 Inconsistent Movement Analysis

Five gestures in our investigation involve inconsistent segments: 'Picking up cell phone', 'Drinking coffee' and three different types of 'eating'. All of them typically include three segments. For 'Drinking coffee', the three segments can be described as follows: 1) picking up the cup, 2) a set of arbitrary movements in middle that may or may not be present at every instance of drinking, and 3) bringing the cup to the mouth and tilting it to drink the coffee. For 'Picking up cell phone', three segments are typically observed: 1) taking cell phone out of pocket and lifting the cell phone to a certain level to read the caller information. 2) a set of different movements in middle that may or may not present at every instance (*e.g.* shift or rotate the cell phone to unlock) and 3) bringing the cell phone close to the ear to answer the phone call. For different styles of 'Eating', three segments are observed: 1) picking up food, 2) a set of arbitrary movements in middle that may or may not be present at every instance, and 3) bring the food to the mouth and feed it to the mouth. For all gestures, we observe that the first and third segments are consistent between all the instances while the second segment may vary a lot from instance to instance. To evaluate our inconsistent segment analysis technique, we use the same error metrics defined by (6.8) – (6.10). Here t_1, t_2 are the starting and ending points of the inconsistent segment of one movement and t_1', t_2' are the starting and ending point of the corresponding gold standard segment obtained from video. The average error for these five gestures are shown in Table 6.4. In Section 6.3.4 and Section 6.3.5 the inconsistent movement analysis improves the classification performance significantly and these errors are good enough for our application.

6.4 Conclusion

To the best of our knowledge, the feature set and the signal processing algorithms described in this work have been proposed for the first time. Our proposed techniques address several important challenges: sensor orientation variations, movement speed variations, and the inconsistent segments present in some movements. In our approach, once an activity/gesture is detected, the time duration of the activity is calculated. The movement's speed is an interesting context for pervasive computing applications as we can infer if the person is in a hurry or is tired. Besides, our approach works for both dynamic periodic movements (*e.g.*, walking and running) and transitional movements (*e.g.*, sit-to-stand and sit-to-lie) while most orientation-independent frequency-based activity recognition algorithms previously published operate solely on dynamic periodic movements.

One limitation of our approach is that we used two sensors to recognize the activities of daily living listed in this research. If we only use one thigh sensor, our approach cannot separate sit-to-lie and sit-to-stand well. This is naturally due to the fact that the two movements have the same footprint on the thigh. As a potential alternative to using multiple sensors, finer-grained orientation independent features from accelerometers could be considered to help distinguish these two movements. In our future work, we will enhance the recognition accuracy of our algorithm to cover a larger number of movements using only one sensor. The other limitation of our approach is that different subjects have slightly different activity templates which decreases the cross subject classification performance. Selecting a larger number of templates will help

enhance the cross subject classification accuracy while increasing the computational cost.

In this chapter, we proposed an activity/gesture recognition approach using wearable motion sensors that can address several practical challenges and detect useful context information. An orientation independent, speed independent feature set is proposed, and a two-stage signal processing algorithm is suggested to perform the activity/gesture recognition. A template refinement technique is proposed to eliminate the negative impact of the inconsistent segments of a movement. Two example applications (*i.e.,* activity recognition and gesture recognition) are utilized for the evaluation. The experimental results show good classification accuracy while retaining robustness to several practical challenges associated with wearable motion sensors in real-world scenarios.

7. SENSOR FUSION ENHANCED ASL RECOGNITION SYSTEM*

As mentioned in section 1.3, one of the challenges of IMU based activity/gesture is that IMU is not able to capture tiny movement which is crucial to some application. In this section, we propose a sensor fusion framework that fuses IMU and sEMG sensors for an example application – American Sign Language recognition. We show that the fusion will enhance the system performance significantly compared to using only the IMU.

7.1 Proposed Sensor Fusion System

7.1.1 System Overview

The block diagram of our proposed multi-modal ASL recognition system is shown in Figure. 7.1. Two phases are included: training phase and testing phase. In the training phase, the signals from 3-D accelerometer (ACC), 3-D gyroscope (GYRO) and four channel sEMG are preprocessed for noise rejection and synchronization purposes. The sEMG based auto-segmentation technique obtains the beginning and ending of a sign for both IMU and sEMG. As the segmentation is done, a broad set of well-established features are extracted for both IMU and sEMG signals. All extracted features are then put into one feature vector. The best feature subset is obtained using an information gain (IG) based feature selection scheme. Four different classifiers are evaluated (i.e. decision tree, support vector machine, NaïveBayes and nearest neighbor) on the selected feature subset and the best one is selected. In the testing phase, the same

*Reprinted with permission from "A Wearable System for Recognizing American Sign Language in Real-Time Using IMU and Surface EMG Sensors" by Jian Wu, Lu Sun and Roozbeh Jafari, 2016. IEEE Journal of Biomedical and Health Informatics, ©2016 by IEEE.

techniques are repeated for preprocessing and segmentation. The selected features are extracted and recognition of the sign is achieved by the chosen classifier.

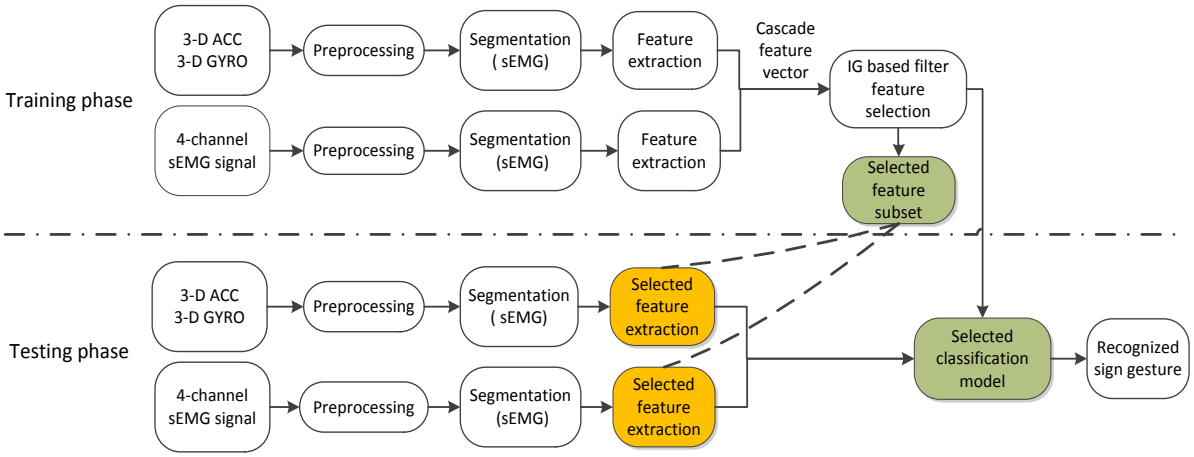


Figure 7.1 - Diagram of proposed system. Reprint with permission from [92].

7.1.2 Preprocessing

The synchronization between IMU and sEMG data is important for fusion. In our system, IMU data samples and sEMG data samples are sent to a PC via Bluetooth and time-stamped with the PC clock. The synchronization is done by aligning samples with the same PC clock. Bluetooth causes a transmission delay (5-20ms) for both IMU and sEMG data and this small synchronization error is negligible for the purposes of our system. To remove low frequency noise in sEMG, a 5Hz IIR high pass filter is used since the frequency components of sEMG beyond the range of 5Hz – 450Hz are negligible [93]. The raw data is used for accelerometer and gyroscope.

7.1.3 Segmentation

Automatic segmentation is crucial for real-time applications. It extracts the period during which each sign word is performed such that the features can be extracted on the correct segment before classification is done. For certain parts of some signs, only finger movements are observed and no obvious motion signal can be detected from the wrist. Thus, sEMG signals are used for our automatic segmentation technique since sEMG signals can capture larger number of movements.

To explain our segmentation technique, we first define the average energy E of four sEMG channels in an n sample window in Equation (7.1). $S_c(i)$ denotes i^{th} sample of c^{th} channel of sEMG. m is total number of channels which equals four in our case. A non-overlapping sliding window is used to calculate E in every window. The length of the window is set to 128 milliseconds, which covers 128 samples with the 1000 Hz sampling frequency. If E in five continuous windows are all larger than a threshold T, the first sample of the first window will be taken as the beginning of a gesture. If E in four continuous windows are all smaller than the threshold, the last sample in the last window is considered to be the ending of this gesture.

$$E = \frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{m} s_c^2(i) \tag{7.1}$$

Different people have different muscular strengths which will result in different E. A simple threshold may not be suitable for all subjects. An adaptive estimation technique is proposed to adjust the threshold according to different subjects and different noise

levels on-line. The proposed approach is explained in two steps. In the first step, the average energy E is calculated for five continuous windows. If all five E is smaller than $a*T$, it is assumed no muscle activity is detected and the threshold is updated with $b*T$ in the second step. a is called the converge parameter and this reduces the threshold T when quiet periods are detected. b is the diverge parameter which enlarges the threshold T as the noise level increases. The values of a, b and T are set to be 0.5, 4 and 0.01 for the system empirically. 0.01 is much bigger than E for all subjects and the user is requested to have a 2-3 seconds quiet period at the beginning of system operation to have the system converge to a suitable threshold.

7.1.4 Feature Extraction

A large number of features have been proposed and studied for both sEMG and IMU sensors for detecting activities or gestures. We adopt some of these well-established features in our paper [94-98]. Table 7.1 and Table 7.2 show features from sEMG and IMU sensors, respectively. The dimension of each feature is also listed in the table. The sEMG features are extracted for all four channel signals and the total dimension is 76. The IMU sensor features are extracted for 3-axis accelerometer, 3-axis gyroscope and the magnitude of accelerometer and gyroscope. It leads to a 192 dimension feature space. The features from sEMG and IMU sensors are combined into the final feature vector of dimension 268.

Table 7.1. sEMG features. Reprint with permission from [92].

Feature name (dimension)	Feature name (dimension)
Mean Absolute Value (1)	Variance (1)
Four order Reflection Coefficients (4)	Willison Amplitude in 5 amplitude ranges (5)
Histogram (1)	Modified Median Frequency (1)
Root Mean Square (1)	Modified Mean Frequency (1)
Four order AR coefficients (4)	

Table 7.2. IMU sensor features. Reprint with permission from [92].

Feature name (dimension)	Feature name (dimension)
Mean (1)	Variance (1)
Standard Deviation (1)	Integration (1)
Root Mean Square (1)	Zero Cross Rate (1)
Mean Cross Rate (1)	Skewness (1)
Kurtosis (1)	First three orders of 256-point FFT Coefficients (3)
Entropy (1)	Signal Magnitude Area (1)
AR coefficients (10)	

7.1.5 Feature Selection

Feature selection provides a way to select the most suitable feature subset for certain tasks from the well-established features. It reduces over fitting problems and information redundancy existing in the feature set. It can also suggest the best feature subset if a smaller feature set is required by applications with limited computational resources.

There are three different feature selection methods which are filter methods, wrapper methods, and embedded methods [99]. Wrapper methods generate scores for each feature subset based on a specific predictive model. Then, cross validation is done for each feature subset. Based on the prediction performance, each subset is assigned a score and the best subset is chosen. Filter methods use general measurement metrics of a dataset to score a feature subset instead of using the error rate of a predictive model. Some common measures are mutual information and inter/intra class distance. The embedded methods perform the feature subset selection in conjunction with the model construction. In our work, an information gain filter method is used in conjunction with a ranking algorithm to rank all the features. The best n features form the best feature subset which is evaluated with different classifiers. The choice of n is discussed in Section 7.3.2. Compared to wrapper methods, the features selected by filter methods will operate for any classifier instead of working only with a specific classifier.

7.1.6 Classification

Four popular classification algorithms are studied in this paper: decision tree (DT) [100], support vector machine (LibSVM) [101], nearest neighbor (NN) and

NaiveBayes. Weka, a widely used open source machine learning tool, is applied for the implementations of these four algorithms [89]. The radial basis function (RBF) kernel is selected for the LibSVM and the best kernel parameters are tuned using a grid search algorithm. The default parameters are selected for the other three classifiers. In machine learning, it is usually hard to determine which classifier is more suitable for a specific application and thus it is worth testing several algorithms before we choose one.

7.2 Experimental Setup

7.2.1 Sensor Placement

The signs can involve one hand or two hands. In our paper, we only look at the right hand movements for both one-hand or two-hand signs. If they system is deployed on two hands, it will increase the recognition accuracy. Figure. 7.2 shows the sensor placement on right forearm of the user. Four major muscle groups are chosen to place four channel sEMG electrodes: (1) extensor digitorum, (2) flexor carpi radialis longus, (3) extensor carpi radialis longus and (4) extensor carpi ulnaris. The IMU sensor is worn on the wrist where a smart watch is usually placed. To improve signal-to-noise ratio of sEMG readings, a bi-polar configuration is applied for each channel and the space between two electrodes for each channel is set to 15 mm [102]. The electrode placements are also annotated in the figure.

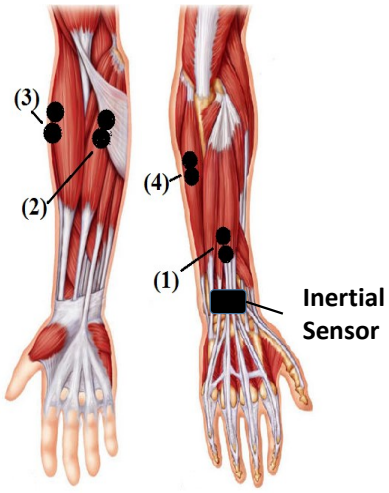


Figure 7.2 - Placement of sEMG electrodes. Reprint with permission from [92].

7.2.2 Data Collection

80 commonly used ASL signs in daily conversations are selected in our paper. Three male and one female volunteer are recruited for data collection. They are all first time learners and did not know ASL before. For each subject, the data is collected from three sessions on three different days and during each session, the subject repeats each sign 25 times. The dataset has 24000 instances in total.

7.2.3 Experiments

Four different experiments are conducted to test our system: intra-subject testing, all cross validation, inter-subject testing and intra-subject cross session testing. For intra-subject testing, the data collected from three sessions of same subject is put together and a 10-fold cross validation is done for the data collected from each subject separately. 10-fold cross validation means the data is split into 10 subsets randomly and the model is trained with 9 subsets and tested on the 10^{th} subset. This process is repeated for 10 times

and the average was taken over. For the all cross validation analyses, data from all four subjects are put together and a 10-fold cross validation is performed. For the inter-subject testing, the classifier is trained with data from three subjects and tested on the fourth subject. The performance is averaged for four tests. The feature selection for the first three experiments is carried out during all cross validation since it has data from all four subjects which makes it a good generalization for classification algorithms. For the intra-subject cross session testing, the feature selection is performed and the classifier is trained with two sessions from each subject and tested on the third session of the same subject. The process is repeated three times for each subject and the performance is averaged for each subject. This experiment would give an indication of how well the system will generalize to new data collected in future for the same subject.

7.3 Experimental Results

7.3.1 Auto-segementation

In our experiment, we do not have a gold standard (*e.g.* video record) and thus it is hard to measure the error of our automatic segmentation technique. However, we know the total number of signs each subject performed and the number of signs our algorithm recognized. An error rate (ER) is defined as:

$$ER = \frac{|detected\ nums - performed\ nums|}{perfomed\ nums} \tag{7.2}$$

detected nums and *performed nums* are numbers of signs our algorithm detected and numbers of signs the user actually performed, respectively. The ER of our approach is 1.3% which indicates our segmentation technique achieves a good performance. The

intra-subject classification results in section V.C also indicate suitable performance of the segmentation.

7.3.2 Feature Selection

All 268 features are ranked with a score obtained from information gain criterion. The highest ranked ones are selected to form the best subset. To decide the size of best feature set, all cross validation is performed on four different classifiers as feature subset size increases from 10 to 268.

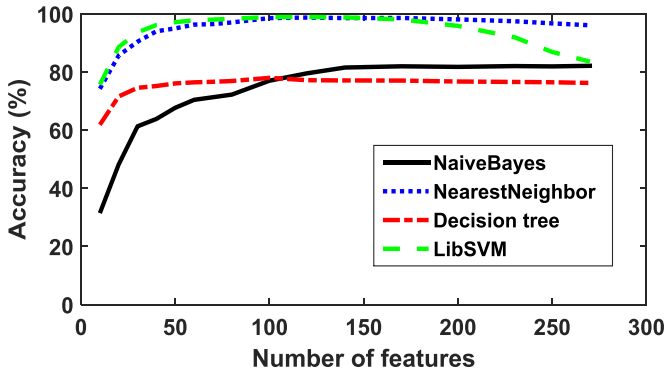


Figure 7.3 - Results of feature selection. Reprint with permission from [92].

Figure. 7.3 shows classification accuracies of four classifiers as the size of the best feature subset increases. It is seen from the figure that as the size of feature subset increases, the accuracies of all classifiers increase. However, when the feature number is bigger than 120 for LibSVM and nearest neighbor, their accuracies start to decrease as a result of over-fitting. This illustrates one of the reasons why feature selection is necessary.

Table 7.3 lists four data points when classifiers achieve best performance.

Table 7.3. Optimal data point of feature selection. Reprint with permission from [92].

Classifier	Optimal point (feature number, accuracy)
NaiveBayes	(270, 82.13%)
Neareast Neighbor	(120, 98.73%)
Decision Tree	(100, 78.00%)
LibSVM	(120, 98.96%)

Table 7.4. Number of features selected from different sensors. Reprint with permission from [92].

Sensor	Number of feature	Sensor	Number of feature
Accelerometer	21	sEMG2	2
Gyroscope	10	sEMG3	0
sEMG1	4	sEMG4	3

Figure. 7.3 shows that when number of selected features becomes 40, LibSVM already achieves 96.16% accuracy. Due to the computational constraints associated with wearable systems, the feature size is thus selected to be 40. Among the 40 features, the numbers of features selected from different sensors are shown in Table 7.4. More than half of the features are selected from accelerometer which means accelerometer plays most important role in recognizing signs. Accelerometer measures both gravity and acceleration caused by movement. Gravity is usually the major part which is capable of capturing hand orientation information. It indicates hand orientation information is more

significant than hand shape when distinguish different signs. Ten features from gyroscope are selected which means that the hand and arm rotation is also valuable information. Nine selected sEMG features make this modality necessary for our system.

To have a better understanding of the importance of different sensor features, forty selected features are listed in Table 7.5 along with their rankings. In the table, Acc_x, Acc_y and Acc_z represent accelerometer readings along *x-axis*, *y-axis* and *z-axis*, respectively. Similarly, Gyro_x, Gyro_y and Gyro_z are gyroscope readings along *x-axis*, *y-axis* and *z-axis*, respectively. From the table, we can see that most of the accelerometer features have very high rank which indicates accelerometer is the most important modality in our system. The gyroscope features have higher ranks than sEMG features on average. Although the gyroscope is not as important as the accelerometer, it contributes more than sEMG. sEMG features are the least important among the three modalities which indicates it may not be significant in our system. Among accelerometer and gyroscope features, the most important ones include mean, integration, standard deviation, RMS and variance. Mean absolute value, variance and RMS are valuable features for sEMG signal. One interesting observation of sEMG features is that four selected features from channel one have higher ranks than the others from channel two and channel four. Channel one is placed near the wrist where a smart watch is usually worn. In reality, if only one electrode is available, channel one would be selected and it can be integrated into a smart watch to capture the most important sEMG features.

Table 7.5. Fourty selected features. Reprint with permission from [92].

Rank	Feature name	Rank	Feature name	Rank	Feature name	Rank	Feature name
1	Mean of Acc_y	11	RMS of Gyro_x	21	RMS of sEMG1	31	Signal magnitude area of Acc_x
2	Mean of Acc_z	12	RMS of amplitude of accelerometer	22	Zero cross rate of Acc_y	32	Variance of sEMG4
3	RMS of Acc_x	13	Mean of amplitude of accelerometer	23	Variance of Gyro_z	33	Entropy of Gyro_x
4	RMS of Acc_z	14	Mean of Acc_x	24	Standard deviation Of Gyro_z	34	RMS of sEMG4
5	RMS of Acc_y	15	Signal magnitude area of Acc_x	25	Variance of Acc_y	35	Signal magnitude area of Gyro_x
6	Integration of Acc_y	16	Standard deviation of Acc_z	26	Standard deviation of Acc_y	36	Zero cross rate of Acc_z
7	Integration of Acc_x	17	Variance of Acc_z	27	Modified mean frequency of sEMG1	37	Mean absolute value of sEMG4
8	Integration of Acc_z	18	Standard deviation of Gyro_z	28	Mean absolute value of sEMG1	38	Signal magnitude area of Gyro_z
9	Entropy of Acc_x	19	Variance of Gyro_x	29	Auto-regression coefficient of Acc_x	39	RMS of sEMG2
10	RMS of Gyro_z	20	Variance of sEMG1	30	Mean absolute value of sEMG2	40	Mean of amplitude of gyroscope

7.3.3 Classification Results

Table 7.6 shows the classification results of intra-subject testing on four subjects. In this experiment, each classifier is trained and tested with data from the same subject. We can see that nearest neighbor and LibSVM achieve high accuracies while decision tree classifier obtains the lowest accuracy. Nearest neighbor classifier is a lazy learning classifier and it does not require a trained model. In the testing phase, it compares the testing instance with all instances in the training set and assigns it a same class label as the most similar instance in the training set. It does not scale well as the size of the training set increases since the testing instance needs to be compared to all instances in the training set. LibSVM trains a model based on training data. As the size of training set increases, it only increase the training time without affecting the time needs in testing phase. This is crucial for real time applications. Therefore, LibSVM is the one we select for our system implementation. The results achieved for 80 signs are consistent with the results obtained for 40 signs in our prior investigation [103]. It indicates our technique scales well for intra-subject testing.

Table 7.6. Results of intra-subject validation. Reprint with permission from [92].

	NaiveBayes	**DT**	**NN**	**LibSVM**
Subject 1	88.81%	83.89%	96.6%	98.22%
Subject 2	97.01%	91.54%	99.16%	99.48%
Subject 3	92.74%	81.97%	92.89%	96.61%
Subject 4	91.15%	77.98%	95.77%	97.23%
Average	93.68%	83.85%	96.11%	97.89%

Table 7.7 shows classification results of all cross validation. For all classifiers, the classification results with sEMG and without sEMG are given. The classification with sEMG means we use all 40 features while without sEMG means we only use 31 features from accelerometer and gyroscope. The performance improvement with adding sEMG is also listed in the table.

Table 7.7. Results of all-cross validation. Reprint with permission from [92].

	NaiveBayes	**DT**	**NN**	**LibSVM**
Accuracy with sEMG	63.87%	76.18%	94.02%	96.16%
Accuracy without sEMG	48.75%	68.93%	87.62%	92.29%
Improvement	15.12%	7.25%	6.4%	3.84%
Precision with sEMG	66.9%	76.3%	94.0%	96.7%
Precision without sEMG	51.8%	69.0%	87.7%	92.3%
Improvement	15.1%	7.3%	6.3%	4.4%
Recall with sEMG	63.9%	76.2%	94.0%	96.7%
Recall without sEMG	48.8%	68.9%	87.7%	92.3%
Improvement	15.1%	7.3%	6.3%	4.4%
F-score with sEMG	63.6%	76.2%	94.0%	96.7%
F-score without sEMG	47.6%	68.9%	87.6%	92.3%
Improvement	16.0%	7.3%	6.4%	4.4%

Among four classifiers, LibSVM achieves the best performance in accuracy, precision, recall and F-score while NaïveBayes gives the worst performance. The accuracy, precision, recall and F-score are very close to each other for all classifiers

which indicates all classifiers achieve balanced performance on our dataset. With 40 features, LibSVM achieves 96.16% accuracy. It is consistent with the results (95.16%) we obtained for 40 sign words with 30 features in our prior study [103]. This proves the scalability of approach for all cross validation test.

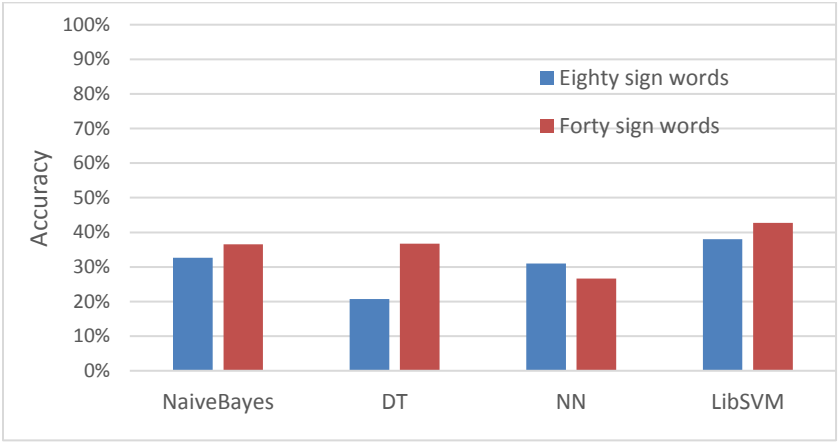


Figure 7.4 - Results of inter-subject testing. Reprint with permission from [92].

The improvement after adding the sEMG modality is most significant for NaiveBayes classifier. It achieves about 15% improvement for all four classification performance metrics. However, for our chosen classifier LibSVM, the accuracy improvement is about 4% while the error rate is reduced by 50%. It indicates the sEMG is necessary and significant. The significance of sEMG is further analyzed in next section.

Figure. 7.4 shows the average accuracy of inter-subject testing for both eighty sign words and forty sign words. It is seen from the figure, none of the classifiers offer good accuracy for recognizing 40 or 80 signs. LibSVM still offers the best performance among four classifiers. There are three reasons for such low accuracies. First, different

people perform the same signs in different ways. Second, all subjects in our experiment are first time ASL learners and never had experience with ASL before. Even though they follow the instructions, the gestures for the same signs are different from each other. Third, different subjects have very different muscular strength and thus leading to different sEMG features for same signs. From the comparison between accuracy of 40 signs and 80 signs, our technique offers low accuracy for all classifiers consistently. For NaiveBayes, NN and LibSVM, the accuracy obtained from 40 signs is higher than obtained from 80 signs. However, NN offers higher accuracy for 80 signs surprisingly. The results suggest our system is not suitable for inter-subject test. It is suggested that the system should be trained on each subject before using it to obtain a high accuracy.

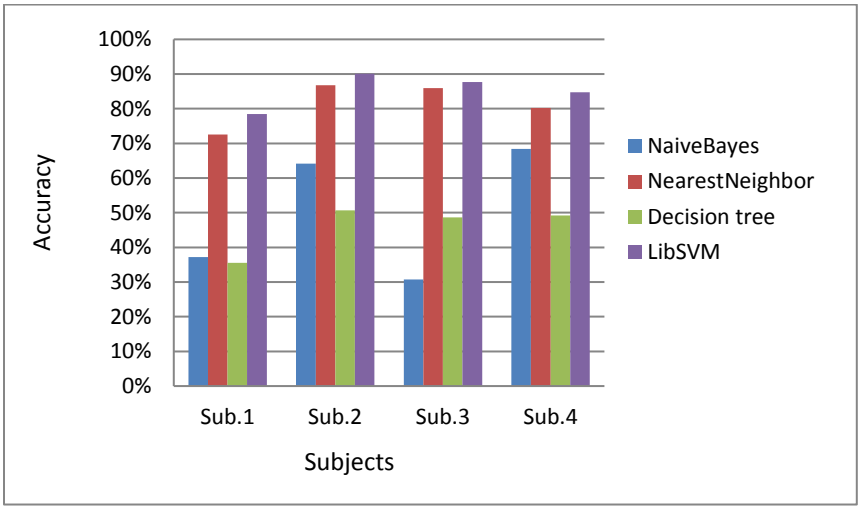


Figure 7.5 - Results of intra-subject cross session testing. Reprint with permission from [92].

The first three experiments show our system achieves suitable performance if the system is trained and tested for the same subject and the system obtains less ideal

performance for inter-subject testing. We further investigate how well the system will generalize for new data collected in future for the same subject. Figure. 8 shows the results of the intra-subject cross session testing in which the feature selection is performed and the classifier is trained with two days data from the same each subject and is tested on data of the third day for the same subject. This process is repeated three times for the same subject and the accuracy measures are averaged. We can see that both NaiveBayes and decision tree yield poor accuracies while LibSVM offers best accuracy. Table VIII shows the average accuracy of different classification algorithms between four subjects. LibSVM achieves 85.24% which is less suitable than the 96.16% of intra-subject testing. Two reasons may explain this performance decrease. The first reason is that the user may have placed the sensors at slightly different locations for the sEMG and IMU sensors, and with a slightly different orientation for the IMU sensor. The second reason is that all four subjects are first time learner who have not developed consistent patterns for signs. They may have performed the same signs somewhat differently on different days.

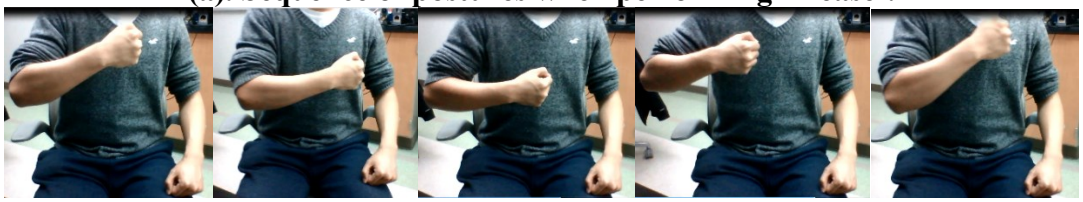
Table 7.8. Results of intra-subject cross session testing. Reprint with permission from [92].

Classifier	Accuracy	Classifier	Accuracy
NaiveBayes	50.11%	NN	81.37%
DT	46.01%	LibSVM	85.24%

7.3.4 Significance of sEMG

From the analysis of inter-subject testing in previous section, LibSVM achieves about 4% improvement for accuracy, precision, recall and F-score while the error rates for these metrics are reduced by about 50%. In this section, we further analyze the importance of sEMG. In American Sign Language, there are some signs which are very similar in arm movement and are different in hand shape and finger configurations (*e.g.* fist and palm). The sEMG is able to capture the difference of finger configuration and to distinguish these signs. If only inertial sensor is considered, the exactly same motion profile will make these signs confusing relative to each other. Figure. 7.6 shows an example of sequences of postures when the user is performing two signs 'Please' and 'Sorry'. We can see from the figures, the arm has the same movement which is drawing a circle in front of chest. The inertial sensor will offer same readings for these two different signs. However, the hand is closed (i.e. fist) when performing 'Sorry' while it is open (i.e. palm) when performing 'Please'. This difference can be captured by sEMG and thus they will be distinguishable if sEMG is included.

(a). Sequence of postures when performing 'Please'.



(b). Sequence of postures when performing 'Sorry'.

Figure 7.6 - Sequence of postures when performing 'Please' and 'Sorry'. Reprint with permission from [92].

Table 7.9. 10 signs with most TP rate improvement. Reprint with permission from [92].

Sign ID	Sign	Improvement
29	Thank	21%
19	My	18.2%
9	Have	16.7%
24	Please	16.7%
37	Work	16.5%
57	Tall	14.3%
67	Girl	13.9%
26	Sorry	13.8%
76	Doctor	12.5%
66	Boy	12.5%

Instead of average improvement, the improvement of true positive (TP) rate is analyzed to show how the sEMG impacts each individual sign. TP rate is rate of true positive and true positives are number of instances which are correctly classified as a given class. The improvement of TP rate of each sign with sEMG can tell how much sEMG will help for each individual signs. Figure. 7.7 shows the TP rate improvement for 80 signs and the improvement is sorted in descend order. From the figure, we can see that for most of signs (last 29-80), the rate of improvement is within the range of [-5%, 5%]. However, for the signs from 1 to 11, the improvement is bigger than 10% which is very helpful for recognizing these signs. In Table 7.9, 10 signs are listed with the highest TP rate improvement. We can see that 'Sorry' and 'Please' are both improved significantly since they are confused with each other. In reality, it is important to eliminate the confusion between signs which have similar motion profile but different sEMG characteristics. Therefore, the sEMG is significant for our system.

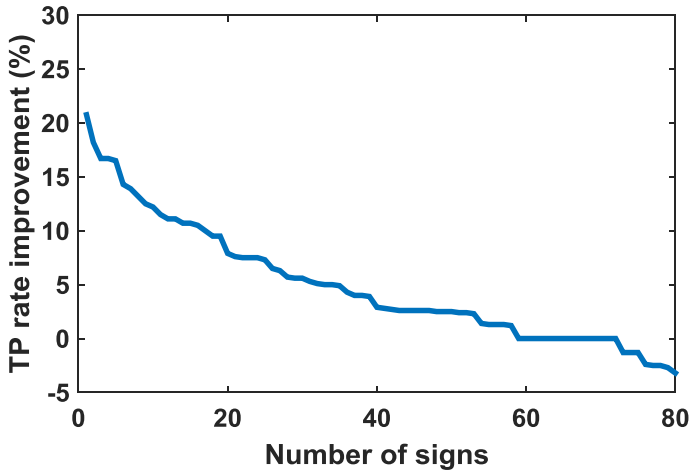


Figure 7.7 - TP rate improvement of all signs. Reprint with permission from [92].

7.4 Conclusion

A wearable real-time American Sign Language recognition system is proposed in this chapter. This is a first study of American Sign Language recognition system fusing IMU sensor and sEMG signals which are complementary to each other. Feature selection is performed to select the best subset of features from a large number of well-established features and four popular classification algorithms are investigated for our system design. The system is evaluated with 80 commonly used ASL signs in daily conversation and an average accuracy of 96.16% is achieved with 40 selected features. The significance of sEMG to American Sign Language recognition task is explored.

8 CONCLUSION

This dissertation document described the research that addressed four practical challenges associated with wearable IMU based activity/gesture recognition: sensor orientation displacement, sensor location displacement, movement speed variation and the lack of tiny movement information. This research not only proposed robust signal processing techniques to address these challenges, but also proposed seamless plug-in solution for the existing systems whose design does not consider the sensor displacement challenges.

To address the sensor orientation displacement challenge, the research first considered to develop an opportunistic calibration method leveraging environmental camera information for the existing systems who suffer from this issue. We introduced a 3-D camera Kinect to help calibrate the sensor orientation displacement w.r.t human body. We proposed a zero-effort two-step search algorithm to calibrate orientation of wearable sensors by calculating the orientations of the sensors with respect to the human body frame based on rotation distance optimization. The experimental results from 4 subjects over 6 daily movements show that our algorithm achieves consistent and accurate results. We also evaluate the performance of our method for activity recognition and compare the results with a non-zero-effort approach and the results show our approach achieves similarly good performance.

Similarly, to address the sensor location displacement challenge, the research proposed an opportunistic calibration method leveraging environmental camera (*i.e.*) information. A cascade decision tree based classifier determines the on-body sensor

locations based on the least square errors obtained by solving the Wahba's problem between accelerometer and Kinect skeleton segment vectors. Our proposed weighting adjusting scheme and the vision occlusion consideration ensure our approach operates robustly. We evaluate our approach with two experiments: simple daily activities and complicated motion tasks. Our approach achieves 100% recall for simple actions and 82.56% recall for complicated motion tasks.

The most important part of this research is that it proposed a robust orientation independent, speed independent activity/gesture recognition system for future system design. To the best of our knowledge, the feature set and the signal processing algorithms described in this work have been proposed for the first time. Our proposed techniques address several important challenges: sensor orientation variations, movement speed variations, and the inconsistent segments present in some movements. Besides, our approach works for both dynamic periodic movements (*e.g.*, walking and running) and transitional movements (*e.g.*, sit-to-stand and sit-to-lie) while most orientation-independent frequency-based activity recognition algorithms previously published operate solely on dynamic periodic movements. In this research, an orientation independent, speed independent feature set is proposed, and a two-stage signal processing algorithm is suggested to perform the activity/gesture recognition. A template refinement technique is proposed to eliminate the negative impact of the inconsistent segments of a movement. Two example applications (*i.e.,* activity recognition and gesture recognition) are utilized for the evaluation. The experimental results show good

classification accuracy while retaining robustness to several practical challenges associated with wearable motion sensors in real-world scenarios.

Finally, in order to address the challenge of lack of tiny motion information of IMU sensors, this research proposed a sensor fusion framework for an example application American Sign Language recognition. It fuses information from IMU sensors and its complementary sensor modality sEMG. Feature selection is performed to select the best subset of features from a large number of well-established features and four popular classification algorithms are investigated for our system design. The system is evaluated with 80 commonly used ASL signs in daily conversation and an average accuracy of 96.16% is achieved with 40 selected features. The significance of sEMG to American Sign Language recognition task is explored.

The findings of this work can help alleviate the main roadblocks to applying IMUs based activity/gesture recognition in practice.

REFERENCES

[1] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 34, no. 3, pp. 334–352, 2004.

[2] J. Cherian, V. Rajanna, D. Goldberg, and T. Hammond, "Did you remember to brush?: a noninvasive wearable approach to recognizing brushing teeth for elderly care," in *Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare*, pp. 48–57, ACM, 2017.

[3] E. S. Sazonov, G. Fulk, N. Sazonova, and S. Schuckers, "Automatic recognition of postures and activities in stroke patients," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pp. 2200–2203, IEEE, 2009.

[4] Y.-J. Chang, S.-F. Chen, and J.-D. Huang, "A kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities," *Research in developmental disabilities*, vol. 32, no. 6, pp. 2566–2570, 2011.

[5] A. Hayes, P. Dukes, and L. F. Hodges, "A virtual environment for post-stroke motor rehabilitation," *Clemson University, Clemson*, 2012.

[6] D. González-Ortega, F. Dáz-Pernas, M. Martnez-Zarzuela, and M. Antón-Rodrguez, "A kinect-based system for cognitive rehabilitation exercises monitoring," *Computer methods and programs in biomedicine*, vol. 113, no. 2, pp. 620–631, 2014.

[7] E. S. Sazonov, G. Fulk, J. Hill, Y. Schutz, and R. Browning, "Monitoring of posture allocations and activities by a shoe-based wearable sensor," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 4, pp. 983–990, 2011.

[8] E. Kantoch and P. Augustyniak, "Human activity surveillance based on wearable body sensor network," in *Computing in Cardiology (CinC), 2012*, pp. 325–328, IEEE, 2012.

[9] E. E. Stone and M. Skubic, "Passive, in-home gait measurement using an inexpensive depth camera: Initial results," in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on*, pp. 183–186, IEEE, 2012.

[10] G. Paragliola and A. Coronato, "Intelligent monitoring of stereotyped motion disorders in case of children with autism," in *Intelligent Environments (IE), 2013 9th International Conference on*, pp. 258–261, IEEE, 2013.

[11] N. F. Sheet, "Centers for medicare & medicaid services website," 2017.

[12] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE transactions on cybernetics*, vol. 43, no. 5, pp. 1318–1334, 2013.

[13] E. Lawrence, C. Sax, K. F. Navarro, and M. Qiao, "Interactive games to improve quality of life for the elderly: Towards integration into a wsn monitoring system," in *eHealth, Telemedicine, and Social Medicine, 2010. ETELEMED'10. Second International Conference on*, pp. 106–112, IEEE, 2010.

[14] K. Gerling, I. Livingston, L. Nacke, and R. Mandryk, "Full-body motion-based game interaction for older adults," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1873–1882, ACM, 2012.

[15] M. Karam, *A framework for research and design of gesture-based human-computer interactions*. PhD thesis, University of Southampton, 2006.

[16] M. S. Ryoo, "Human activity prediction: Early recognition of ongoing activities from streaming videos," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 1036–1043, IEEE, 2011.

[17] W. Niu, J. Long, D. Han, and Y.-F. Wang, "Human activity detection and recognition for video surveillance.," in *ICME*, vol. 1, pp. 719–722, 2004.

[18] O. C. Ann and L. B. Theng, "Human activity recognition: a review," in *Control System, Computing and Engineering (ICCSCE), 2014 IEEE International Conference on*, pp. 389–393, IEEE, 2014.

[19] J. Freer, B. Beggs, H. Fernandez-Canque, F. Chevriert, and A. Goryashko, "Automatic recognition of suspicious activity for camera based security systems," 1995.

[20] R. Polana and R. Nelson, "Low level recognition of human motion (or how to get your man without finding his body parts)," in *Motion of Non-Rigid and Articulated Objects, 1994., Proceedings of the 1994 IEEE Workshop on*, pp. 77–82, IEEE, 1994.

[21] R. Polana and R. Nelson, "Detecting activities," in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*, pp. 2–7, IEEE, 1993.

[22] marketresearchengine, “Wearable devices market by product analysis (wrist-wear, foot-wear, eye-wear, body-wear, neck-wear); by application analysis (fitness and sports, infotainment, healthcare, defense, enterprise and industrial) and by regional analysis â€“ global forecast by 2016 - 2022,” Tech. Rep. SEWDM717, July 2017.

[23] N. Kale, J. Lee, R. Lotfian, and R. Jafari, “Impact of sensor misplacement on dynamic time warping based human activity recognition using wearable computers,” in *Proceedings of the conference on Wireless Health*, p. 7, ACM, 2012.

[24] Y. Ke, R. Sukthankar, and M. Hebert, “Spatio-temporal shape and flow correlation for action recognition,” in *2007 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2007.

[25] E. Shechtman and M. Irani, “Space-time behavior based correlation,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 405–412, IEEE, 2005.

[26] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, “Behavior recognition via sparse spatio-temporal features,” in *Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on*, pp. 65–72, IEEE, 2005.

[27] D. Sherrill, P. Bonato, and C. De Luca, “A neural network approach to monitor motor activities,” in *Engineering in Medicine and Biology, 2002. 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society EMBS/BMES Conference, 2002. Proceedings of the Second Joint*, vol. 1, pp. 52–53, IEEE, 2002.

[28] X. Chen, X. Zhang, Z.-Y. Zhao, J.-H. Yang, V. Lantz, and K.-Q. Wang, "Hand gesture recognition research based on surface emg sensors and 2d-accelerometers," in *Wearable Computers, 2007 11th IEEE International Symposium on*, pp. 11–14, IEEE, 2007.

[29] K. Kunze, P. Lukowicz, H. Junker, and G. Tröster, "Where am i: Recognizing on-body positions of wearable sensors," in *Location-and Context-Awareness*, pp. 264–275, Springer, 2005.

[30] K. Kunze, P. Lukowicz, K. Partridge, and B. Begole, "Which way am i facing: Inferring horizontal device orientation from an accelerometer signal," in *Wearable Computers, 2009. ISWC'09. International Symposium on*, pp. 149–150, IEEE, 2009.

[31] N. Pham and T. Abdelzaher, "Robust dynamic human activity recognition based on relative energy allocation," in *Distributed Computing in Sensor Systems*, pp. 525–530, Springer, 2008.

[32] M.-M. Bidmeshki and R. Jafari, "Low power programmable architecture for periodic activity monitoring," in *Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems*, pp. 81–88, ACM, 2013.

[33] Y. Li, X. Chen, X. Zhang, K. Wang, and Z. J. Wang, "A sign-component-based framework for chinese sign language recognition using accelerometer and semg data," *Biomedical Engineering, IEEE Transactions on*, vol. 59, no. 10, pp. 2695–2704, 2012.

[34] J. Kim, J. Wagner, M. Rehm, and E. André, "Bi-channel sensor fusion for automatic sign language recognition," in *Automatic Face & Gesture Recognition, 2008. FG'08. 8th IEEE International Conference on*, pp. 1–6, IEEE, 2008.

[35] T. R. Bennett, C. Savaglio, D. Lu, H. Massey, X. Wang, J. Wu, and R. Jafari, "Motionsynthesis toolset (most): a toolset for human motion data synthesis and validation," in *Proceedings of the 4th ACM MobiHoc workshop on Pervasive wireless healthcare*, pp. 25–30, ACM, 2014.

[36] J. L. Crassidis, K.-L. Lai, and R. R. Harman, "Real-time attitude-independent three-axis magnetometer calibration," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 1, pp. 115–120, 2005.

[37] P. D. Groves, *Principles of GNSS, inertial, and multisensor integrated navigation systems*. Artech house, 2013.

[38] P. G. Savage, "Strapdown inertial navigation integration algorithm design part 1: Attitude algorithms," *Journal of guidance, control, and dynamics*, vol. 21, no. 1, pp. 19–28, 1998.

[39] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3d tracking of hand articulations using kinect.," in *BMVC*, vol. 1, p. 3, 2011.

[40] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Tracking the articulated motion of two strongly interacting hands," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 1862–1869, IEEE, 2012.

[41] B. Lange, C.-Y. Chang, E. Suma, B. Newman, A. S. Rizzo, and M. Bolas, "Development and evaluation of low cost game-based balance rehabilitation tool using the microsoft kinect sensor," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pp. 1831–1834, IEEE, 2011.

[42] V. Nathan, J. Wu, C. Zong, Y. Zou, O. Dehzangi, M. Reagor, and R. Jafari, "A 16-channel bluetooth enabled wearable eeg platform with dry-contact electrodes for brain computer interface," in *Proceedings of the 4th Conference on Wireless Health*, p. 17, ACM, 2013.

[43] C. J. De Luca, L. Donald Gilmore, M. Kuznetsov, and S. H. Roy, "Filtering the surface emg signal: Movement artifact and baseline noise contamination," *Journal of biomechanics*, vol. 43, no. 8, pp. 1573–1579, 2010.

[44] I. Mesa, A. Rubio, I. Tubia, J. De No, and J. Diaz, "Channel and feature selection for a surface electromyographic pattern recognition task," *Expert Systems with Applications*, vol. 41, no. 11, pp. 5190–5200, 2014.

[45] K. Kunze and P. Lukowicz, "Dealing with sensor displacement in motion-based onbody activity recognition systems," in *Proceedings of the 10th international conference on Ubiquitous computing*, pp. 20–29, ACM, 2008.

[46] U. Steinhoff and B. Schiele, "Dead reckoning from the pocket-an experimental study," in *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pp. 162–170, IEEE, 2010.

[47] K. Forster, D. Roggen, and G. Troster, "Unsupervised classifier self-calibration through repeated context occurences: is there robustness against sensor displacement to gain?," in *Wearable Computers, 2009. ISWC'09. International Symposium on*, pp. 77–84, IEEE, 2009.

[48] R. Chavarriaga, H. Bayati, and J. D. Millán, "Unsupervised adaptation for acceleration-based activity recognition: robustness to sensor displacement and rotation," *Personal and Ubiquitous Computing*, vol. 17, no. 3, pp. 479–490, 2013.

[49] A. Vahdatpour, N. Amini, and M. Sarrafzadeh, "On-body device localization for health and medical monitoring applications," in *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*, pp. 37–44, IEEE, 2011.

[50] D. Weenk, B.-J. F. Van Beijnum, C. T. Baten, H. J. Hermens, and P. H. Veltink, "Automatic identification of inertial sensor placement on human body segments during walking," *Journal of neuroengineering and rehabilitation*, vol. 10, no. 1, p. 31, 2013.

[51] K. Kunze and P. Lukowicz, "Using acceleration signatures from everyday activities for on-body device location," in *Wearable Computers, 2007 11th IEEE International Symposium on*, pp. 115–116, IEEE, 2007.

[52] G. Bahle, P. Lukowicz, K. Kunze, and K. Kise, "I see you: How to improve wearable activity recognition by leveraging information from environmental cameras," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on*, pp. 409–412, IEEE, 2013.

[53] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series.," in *KDD workshop*, vol. 10, pp. 359–370, Seattle, WA, 1994.

[54] T. Helten, M. Muller, H.-P. Seidel, and C. Theobalt, "Real-time body tracking with one depth camera and inertial sensors," in *Computer Vision (ICCV), 2013 IEEE International Conference on*, pp. 1105–1112, IEEE, 2013.

[55] G. Pons-Moll, A. Baak, T. Helten, M. Muller, H.-P. Seidel, and B. Rosenhahn, "Multisensor-fusion for 3d full-body human motion capture," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 663–670, IEEE, 2010.

[56] S. Thiemjarus, "A device-orientation independent method for activity recognition," in *Body Sensor Networks (BSN), 2010 International Conference on*, pp. 19–23, IEEE, 2010.

[57] D. Mizell, "Using gravity to estimate accelerometer orientation," in *2012 16th International Symposium on Wearable Computers*, pp. 252–252, IEEE Computer Society, 2003.

[58] J. Yang, "Toward physical activity diary: motion recognition using simple acceleration features with mobile phones," in *Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics*, pp. 1–10, ACM, 2009.

[59] A. Henpraserttae, S. Thiemjarus, and S. Marukatat, "Accurate activity recognition using a mobile phone regardless of device orientation and location," in *Body Sensor Networks (BSN), 2011 International Conference on*, pp. 41–46, IEEE, 2011.

[60] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan, "uwave: Accelerometer-based personalized gesture recognition and its applications," *Pervasive and Mobile Computing*, vol. 5, no. 6, pp. 657–675, 2009.

[61] J.-K. Min, B. Choe, and S.-B. Cho, "A selective template matching algorithm for short and intuitive gesture ui of accelerometer-builtin mobile phones," in *Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on*, pp. 660–665, IEEE, 2010.

[62] T. Starner, J. Weaver, and A. Pentland, "Real-time american sign language recognition using desk and wearable computer based video," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 12, pp. 1371–1375, 1998.

[63] C. Vogler and D. Metaxas, "A framework for recognizing the simultaneous aspects of american sign language," *Computer Vision and Image Understanding*, vol. 81, no. 3, pp. 358–384, 2001.

[64] L.-G. Zhang, Y. Chen, G. Fang, X. Chen, and W. Gao, "A vision-based sign language recognition system using tied-mixture density hmm," in *Proceedings of the 6th international conference on Multimodal interfaces*, pp. 198–204, ACM, 2004.

[65] M. M. Zaki and S. I. Shaheen, "Sign language recognition using a combination of new vision based features," *Pattern Recognition Letters*, vol. 32, no. 4, pp. 572–577, 2011.

[66] M. W. Kadous *et al.*, "Machine recognition of auslan signs using powergloves: Towards large-lexicon recognition of sign language," in *Proceedings of the Workshop on the Integration of Gesture in Language and Speech*, pp. 165–174, Citeseer, 1996.

[67] C. Oz and M. C. Leu, "American sign language word recognition with a sensory glove using artificial neural networks," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 7, pp. 1204–1213, 2011.

[68] V. E. Kosmidou and L. J. Hadjileontiadis, "Sign language recognition using intrinsic-mode sample entropy on semg and accelerometer data," *Biomedical Engineering, IEEE Transactions on*, vol. 56, no. 12, pp. 2879–2890, 2009.

[69] J. Wu and R. Jafari, "Zero-effort camera-assisted calibration techniques for wearable motion sensors," in *Proceedings of the Wireless Health 2014 on National Institutes of Health*, pp. 1–8, ACM, 2014.

[70] S. O. Madgwick, A. J. Harrison, and R. Vaidyanathan, "Estimation of imu and marg orientation using a gradient descent algorithm," in *Rehabilitation Robotics (ICORR), 2011 IEEE International Conference on*, pp. 1–7, IEEE, 2011.

[71] J. J. Kuffner, "Effective sampling and distance metrics for 3d rigid body path planning," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 4, pp. 3993–3998, IEEE, 2004.

[72] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[73] J. Wu and R. Jafari, "Seamless vision-assisted placement calibration for wearable inertial sensors," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 3, p. 71, 2017.

[74] R. Bartlett, *Introduction to sports biomechanics*. E & FN Spon London, 1997.

[75] M. E. Sargin, Y. Yemez, E. Erzin, *et al.*, "Audiovisual synchronization and fusion using canonical correlation analysis," *Multimedia, IEEE Transactions on*, vol. 9, no. 7, pp. 1396–1403, 2007.

[76] G. Wahba, "A least squares estimate of satellite attitude," *SIAM review*, vol. 7, no. 3, pp. 409–409, 1965.

[77] S. Madgwick, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," *Report x-io and University of Bristol (UK)*, vol. 25, pp. 113–118, 2010.

[78] G. M. Lerner, “Three-axis attitude determination,” *Spacecraft Attitude Determination and Control*, vol. 73, pp. 420–428, 1978.

[79] R. Lienhart, L. Liang, and A. Kuranov, “A detector tree of boosted classifiers for real-time object detection and tracking,” in *Multimedia and Expo, 2003. ICME’03. Proceedings. 2003 International Conference on*, vol. 2, pp. II–277, IEEE, 2003.

[80] P. Viola and M. J. Jones, “Robust real-time face detection,” *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[81] D. M. Powers, “Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation,” 2011.

[82] J. Wu and R. Jafari, “Orientation independent activity/gesture recognition using wearable motion sensors,” *IEEE Internet of Things Journal*, 2018.

[83] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” *Readings in speech recognition*, vol. 159, 1990.

[84] P. Senin, “Dynamic time warping algorithm review,” *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, vol. 855, pp. 1–23, 2008.

[85] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152, ACM, 1992.

[86] Y. Sakurai, C. Faloutsos, and M. Yamamuro, “Stream monitoring under the time warping distance,” in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pp. 1046–1055, IEEE, 2007.

[87] L. Rokach and O. Maimon, "Clustering methods," in *Data mining and knowledge discovery handbook*, pp. 321–352, Springer, 2005.

[88] L. Sun, D. Zhang, B. Li, B. Guo, and S. Li, "Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations," in *Ubiquitous intelligence and computing*, pp. 548–562, Springer, 2010.

[89] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[90] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14, pp. 1137–1145, 1995.

[91] D. Powers, "Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation (tech. rep.)," *Adelaide, Australia*, 2007.

[92] J. Wu, L. Sun, and R. Jafari, "A wearable system for recognizing american sign language in real-time using imu and surface emg sensors.," *IEEE J. Biomedical and Health Informatics*, vol. 20, no. 5, pp. 1281–1290, 2016.

[93] R. Merletti and P. Di Torino, "Standards for reporting emg data," *J Electromyogr Kinesiol*, vol. 9, no. 1, pp. 3–4, 1999.

[94] A. Phinyomark, C. Limsakul, and P. Phukpattaranont, "A novel feature extraction for robust emg pattern recognition," *arXiv preprint arXiv:0912.3973*, 2009.

[95] M. Zhang and A. A. Sawchuk, "Human daily activity recognition with sparse representation using wearable sensors," *IEEE journal of Biomedical and Health Informatics*, vol. 17, no. 3, pp. 553–560, 2013.

[96] S. H. Khan and M. Sohail, “Activity monitoring of workers using single wearable inertial sensor,”

[97] O. Paiss and G. F. Inbar, “Autoregressive modeling of surface emg and its spectrum with application to fatigue,” *IEEE transactions on biomedical engineering*, no. 10, pp. 761–770, 1987.

[98] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, “A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer,” *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 5, pp. 1166–1172, 2010.

[99] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *The Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[100] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[101] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[102] M. Z. Jamal, “Signal acquisition using surface emg and circuit design considerations for robotic prosthesis,” 2012.

[103] J. Wu, Z. Tian, L. Sun, L. Estevez, and R. Jafari, “Real-time american sign language recognition using wrist-worn motion and surface emg sensors,” in *Wearable and Implantable Body Sensor Networks (BSN), 2015 IEEE 12th International Conference on*, pp. 1–6, IEEE, 2015.