University of California
Los Angeles

Robust and Large-scale Human Motion Estimation with Low-cost Sensors

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical Engineering

by

Hua-I Chang

2016

Abstract of the Dissertation

Robust and Large-scale Human Motion Estimation with Low-cost Sensors

by

Hua-I Chang

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 2016

Professor Gregory J. Pottie, Chair

Enabling large-scale monitoring and classification of a range of motion activities is of primary importance due to the need by healthcare and fitness professionals to monitor exercises for quality and compliance. Video based motion capturing systems (e.g., VICON cameras) provide a partial solution. However, these expensive and fixed systems are not suitable for patients' at-home daily motion monitoring. Wireless motion sensors, including accelerometers and gyroscopes, can provide a low-cost, small-size, and highly-mobile option. However, acquiring robust inference of human motion trajectory via low-cost inertial sensors remains challenging. Sensor noise and drift, sensor placement errors and variation of activity over the population all lead to the necessity of a large amount of data collection. Unfortunately, such a large amount of data collection is prohibitively costly.

In observance of these issues, a series of solutions for robust human motion monitoring and activity classification will be presented. The implementation of a real-time context-guided activity classification system will be discussed. To facilitate ground truth data acquisition, we proposed a virtual inertial measurements platform to convert the currently available MoCap database into a noiseless and error-free inertial measurements database. An opportunistic calibration system which deals with sensor placement errors will be discussed. In addition, a sensor fusion approach for robust upper limb motion tracking will also be presented.

The dissertation of Hua-I Chang is approved.

Lara Dolecek

Mario Gerla

William Kaiser

Gregory J. Pottie, Committee Chair

University of California, Los Angeles

2016

Table of Contents

List of Figures

List of Tables

Acknowledgments

Foremost, I would like to express my sincere gratitude to my advisor Prof. Gregory J. Pottie for the continuous support of my PhD studies and research, for his patience, motivation, enthusiasm, and immense knowledge. During my PhD program, I encountered several setbacks and frustrations for my research. He inspired and guided me to go through these struggles. I could not have imagined having a better advisor and mentor for my PhD studies.

I would also like to thank my dissertation committee: Prof. Bill Kaiser, Prof. Lara Dolecek, and Prof. Mario Gerla, for their advice and help during my PhD studies at UCLA.

I was lucky enough to have some awesome lab mates when in UCLA graduate school. First, I would like to thank my first and only senior lab mate Jay Chien who inspired me and gave lots of valuable knowledge and priceless experiences in doing researches, launching experiments, and mentoring internship students. I have also benefited greatly from both professional and personal discussions with fellow colleagues at the UCLA Wireless Health Institute and the Neurology department: Mahdi Ashktorab, Chris Baek, Victor Chen, Andrew Dorsch, Bruce Dubkin, Christine King, Yeung Lam, Seth Thomas, Yan Wang, Michael Wasko, Xiaoxu Wu, Celia Xu, James Xu, Eric Yuen, Vince Zegarski.

I would also like to thank many of my mentored students in EE180D, EE199 and interns in the UCLA HSSEAS Summer Undergraduate Scholars Program: Faraz Aghazadeh, Marwan Alsaraf, Triana Carmenate, Jimmy Chen, Matthew Dempsey, Vivek Desai, John Goodlad, Natthew Newbill, Oscar Santana, Anchi Su, Andreas Widy, Meng-Huan Wu, Jingtao Xia. These students helped me collecting data, recruited their friends for more data, and implemented algorithms on their own under my guidance. During this time not only did I mentor them with a series of discussions and experiments, but their creative ideas and thoughts also inspired me a lot.

In addition, for the colleagues in UCLA or other universities, I am also grateful for the company during the past few years: Esther Chang, Hung-Bin Chang, Chu-Hsiang Huang,

Jinxi Guo, I-Tan Lin, Yu-Yu Lin, Cheng-An Yang, and many others. Without all of you, it would be a hard time during the past few years.

Last, but not least, I would like to dedicate this thesis to my parents, and my girlfriend Joanna Tong, for their love, patience, and understanding. It is only through their enduring support and dedication that I was able to complete this great task.

Vita

2004–2008	Bachelor of Science, Electrical and Control Engineering Department, NCTU.
2008–2009	Information and Technology Officer, Coast Guard Administration, Taiwan.
2009–2010	Research Assistant, Electrical Engineering Department, NTU.
2010–2012	Master of Science, Electrical Engineering Department, UCLA.
2012–2016	Teaching Fellow, Electrical Engineering Department, UCLA.
2012–2016	Doctor of Philosophy Candidate, Electrical Engineering Department, UCLA.

Publications

Hua-I Chang, V. Desai, O. Santana, M. Dempsey, A. Su, J. Goodlad, F. Aghazadeh, and G. Pottie. "Opportunistic calibration of sensor orientation using the Kinect and inertial measurement unit sensor fusion." In*Proceedings of the conference on Wireless Health, p. 2. ACM, 2015.*

Hua-I Chang, X. Wu, C. H. Huang, Y. Wang, L. Dolecek and G. Pottie, "Virtual inertial measurements for motion inference in wireless health,"*2013 Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, 2013, pp. 966-969.*

Hua-I Chang, C. Chien, J. Y. Xu and G. J. Pottie, "Context-guided universal hybrid decision tree for activity classification," *2013 IEEE International Conference on Body Sensor Networks, Cambridge, MA, USA, 2013, pp. 1-6*

C. Chien, J. Y. Xu, **Hua-I Chang**, X. Wu, G. J. Pottie, "Model construction for Human Motion Classification using Inertial Sensors" in *Information Theory and Applications Workshop (ITA), 2013. IEEE, 2013.*

J. Y. Xu, **Hua-I Chang**, C. Chien, W. J. Kaiser and G. J. Pottie, "Context-driven, Prescription-Based Personal Activity Classification: Methodology, Architecture, and End-to-End Implementation," in *IEEE Journal of Biomedical and Health Informatics, vol. 18, no. 3, pp. 1015-1025, May 2014.*

CHAPTER 1

Introduction

Chapter 1 introduces some preliminary knowledge that is necessary for this thesis. For motion sensing technologies, we will introduce and compare two of the most commonly used motion sensing technologies: inertial sensors and vision-based sensors. For activity classification, we will introduce machine learning techniques that are commonly used in activity classification such as the tree classifier, Naïve Bayes classifier, and support vector machine. For motion reconstruction, we will present the algorithm for trajectory reconstruction and the zero-velocity update technique that deal with sensor drift.

I Motivations and Objective

We are living on an aging planet. According to a current report [1] from the U.S. Census Bureau, for the first time in human history, the number of people over the age of 65 will surpass the number of children under the age of 5 (Fig. 1.1). At the same time, we see an unprecedented rise of chronic afflictions such as stroke and Parkinson's disease. These diseases are the leading causes for the degradation of the quality of life for the elderly. In order to provide targeted treatment and effective rehabilitation, doctors would like to monitor patients' mobility status to assess their condition. The traditional approach relies on patients' self-reports during clinic visits. However, this approach is unreliable and lacks timely feedback. Thanks to the proliferation of wearable inertial sensors, nowadays, these sensing devices can provide valuable information for doctors during diagnostic, treatment and rehabilitation process. This will improve the quality of life for patients with mobility

difficulties.

Our objective is to enable ***robust*** and ***large-scale*** human motion inference with ***low-cost sensors***. We want to acquire robust motion inference, such as knowing when and what kind of activities are performed, and their quality. However, there exist fundamental challenges that have not been solved in the past. When scaling to a large population, model complexity and user compliance will degrade our motion inference. In addition, while low-cost sensors are suitable for large-scale deployment, their uncertainties will also provide inferior performance.

In this thesis, we present a couple of novel algorithms and models to solve these problems. The following goals are pursued:

- Implement a real-time activity classification system for large-scale population, with high accuracy, low model complexity, and targeted monitoring.
- Systematically generate ground truth data without a large amount of human effort involved.
- Provide reliable motion data, without the requirement of pre-defined calibration movements or careful placement of wearable sensors.
- Robustly estimate the motion trajectories of the upper limbs at any given moment of the time.

II Background 1: Motion Sensing Technologies

Inertial sensors and vision-based sensors are the two most popular technologies used for human motion sensing. In this section, we provide a brief introduction to the two systems as well as a comparison of their capabilities and limitations.

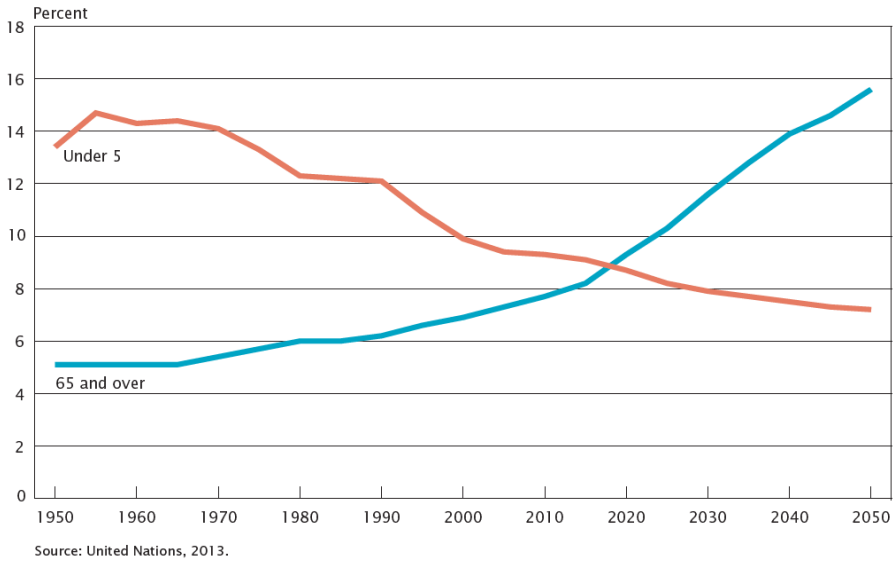


Figure 1.1: Young Children and Older People as a Percentage of Global Population: 1950 to 2050 [1]

II.1 Inertial Sensor Based System

Advances in MEMS technologies have led to the proliferation of wearable inertial sensor based activity monitoring systems. State-of-the-art inertial sensing platforms typically include accelerometers, gyroscopes, and magnetometers. MEMS accelerometers sense both gravitation and externally applied acceleration by measuring the deviation of a mass suspended between several capacitive plates. The measured change in capacitance can be directly correlated with the strength of the applied acceleration. The MEMS gyroscope measures the Coriolis force exerted by a vibrating mass on its supports when the sensor undergoes rotational acceleration. The perpendicular displacement of a suspended proof mass by the Coriolis force results in a change of capacitance in the sensing arms, the value of which is directly correlated with the angular acceleration. An example diagram for both a MEMS accelerometer and gyroscope is shown in Fig. 1.2 and 1.3. The magnetometer is a highly sensitive Hall Effect sensor primarily used to measure the magnetic field of the earth. When a magnetic field is present, the current exposed to this field gets deflected. From this measured deflection the magnetic field force can be calculated.

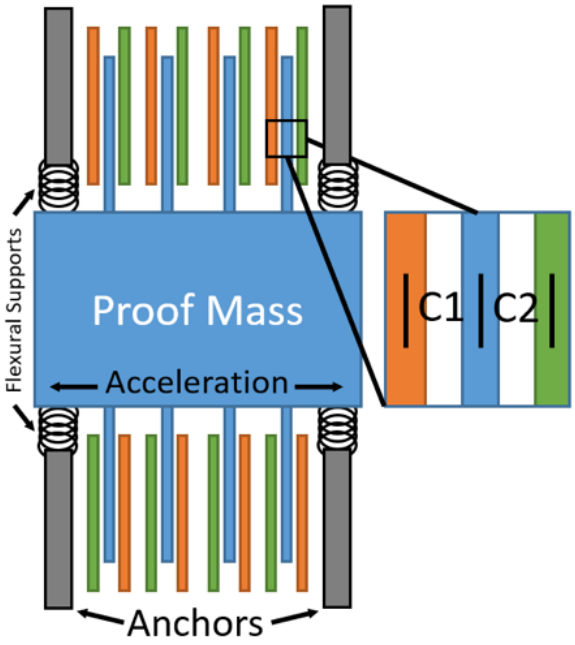


Figure 1.2: Example diagram of a MEMS accelerometer

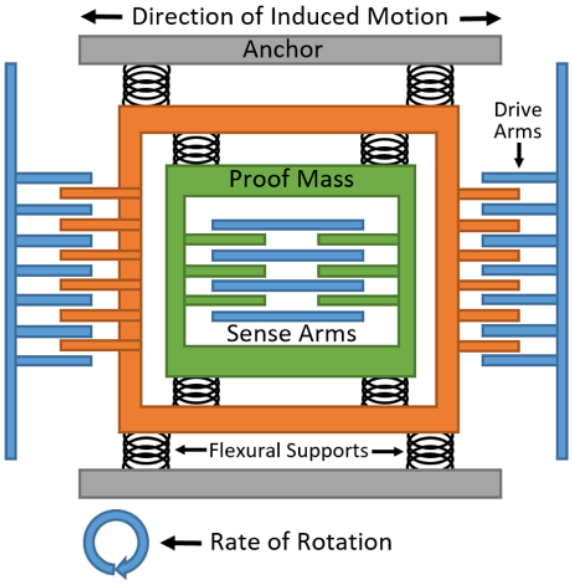
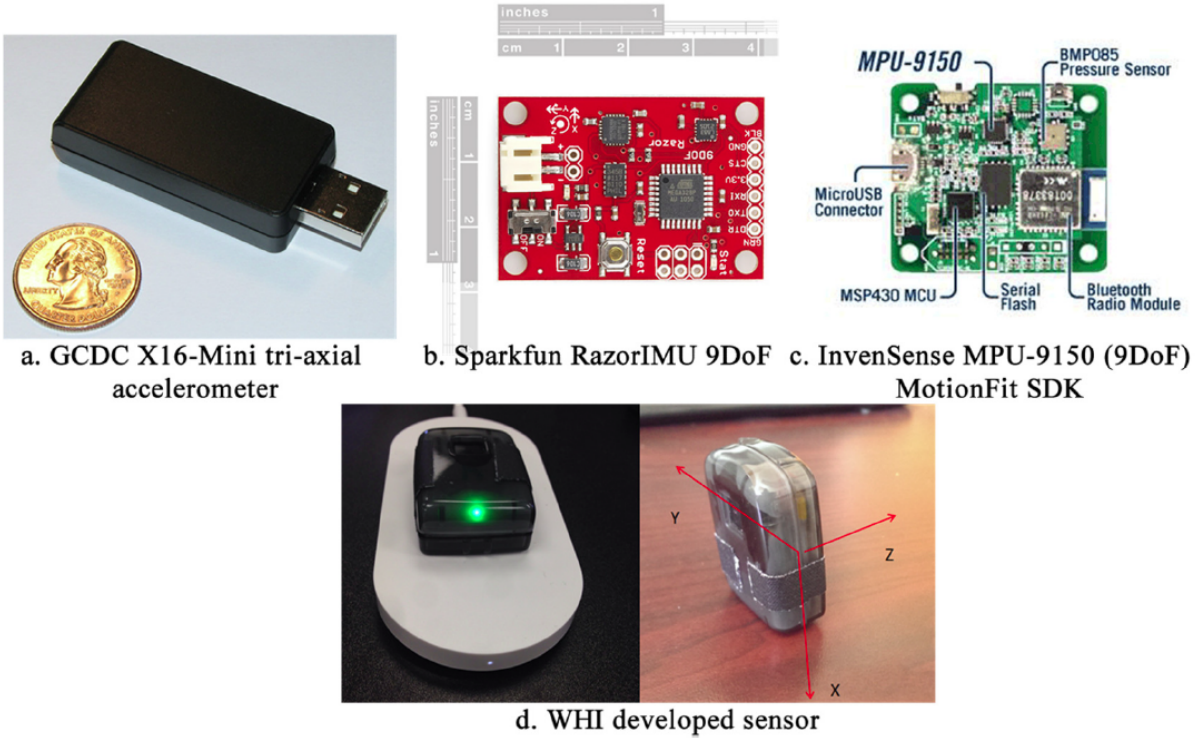


Figure 1.3: Example diagram of a MEMS gyroscope

Activity monitoring using MEMS inertial sensors is rapidly growing. [2] used one tri-axial accelerometer mounted on the waist to classify activities correlated with movements measured in a controlled laboratory. [3] and [4] utilize a Kalman filter to combine accelerometer, gyroscope, and magnetometer sensor data to detect slow moving body rotation and linear translation. In [5], the author developed a biomechanical model to track motions with wearable sensors. Furthermore, inertial sensor based activity monitoring systems have been verified to accurately and reliably characterize the gait of post-stroke patients [6, 7]. In a pioneering clinical trial, a group of physicians and engineers deployed wearable inertial devices on hundreds of post-stroke patients with feedback provided to the physicians and patients on a daily basis. The system proved effective in monitoring activity in the ambulatory community [8, 9]. To detect relative position in 3D space, data from inertial sensors require double integration. Thus, the drift and broadband noise present in MEMS sensor result in rapid accumulation of errors. To meet the stringent accuracy requirements for use in healthcare, algorithms must be developed to reduce the impact of noise on the final results. Fig. 1.4 demonstrates some of the inertial sensors used by the UCLA's Wireless Health Institute (WHI).

II.2 Vision-based System

Vision-based motion sensing systems comprise two major categories: marker-based systems and image-based systems. Marker-based motion capture systems [10,11] track the movement of reflective markers or light-emitting diodes placed on the human body. Thus they indirectly track the movement of body segments as well as the configuration of body joints. For such systems, accurate 3D marker positions in a global frame of reference are computed from the images captured by a group of surrounding cameras using triangulation. Although such systems can provide high-precision joint position in 3D space, they are extremely expensive and time intensive in their deployment. Therefore, they are infeasible for daily activity monitoring.



a. GCDC X16-Mini tri-axial accelerometer

b. Sparkfun RazorIMU 9DoF

c. InvenSense MPU-9150 (9DoF) MotionFit SDK

d. WHI developed sensor

Figure 1.4: Sensors used/developed by Wireless Health Institute (WHI)

Marker-less systems use computer vision techniques to derive motion parameters from the captured video [12]. Recently, low-cost off-the-shelf sensors have exploited depth cameras to capture the movement of human limbs and extract the 3D position of body joints. The Kinect, for example, is a motion-tracking device developed by Microsoft capable of monitoring up to 6 full skeletons within the sensor's field of view. For each skeleton, 24 joints are defined and their positions and rotations tracked. Due to the embedded tracking algorithm's large training data set, the Kinect provides accurate tracking outcomes which can be considered as the ground truth [13]. Another example is the Leap Motion controller, which is designed specifically for motion tracking of hand gestures. In this system, three infrared LEDs and two monochromatic cameras are used to reconstruct the 3D scene and precisely track hand position within a small range. Research suggests that the Leap Motion controller can potentially be extended as a rehabilitation tool in the home environment, removing the requirement for the presence of a therapist [14]. While vision-based systems can provide desirable tracking accuracy, they are not self-contained and require cameras deployed in the environment. Additionally, vision based systems raise privacy concerns and are as yet not feasible for large-scale employment.

In this research, vision-based motion sensing systems we used included the Microsoft Kinect, Leap Motion controller and Vicon-460 camera. Details of these devices will be presented in later chapters.

III Background 2: Activity Classification

The goal of activity classification is to infer which activity is performed, given data and machine learning models. In this section, we introduce several machine learning techniques that are commonly used in activity classification.

Classifier: A classifier is a function which maps the feature vectors into classes. Suppose we have collected a set of training data (TD) consisting of n observations, each observation has p features, and there is one label out of q classes associated with each observation.

$$TD = \big\{(x_i, y_i)|x_i \in \mathbb{R}^p, y_i \in \{1, 2, ..., q\}\big\}, i = 1, 2, ..., n \tag{1.1}$$

where x_i is the feature vector, and the y_i are the labeled classes associated with the features.

The classifier can be thought of as a mapping function that maps the training data to a specific class or partition:

$$f : TD_i \to \hat{y}_i \tag{1.2}$$

where TD_i is the i^{th} instance of the n observations from the training data, and y_i is the classification result of the corresponding i^{th} observation.

III.1 Decision Tree Classifier

A decision tree classifier is a supervised machine learning technique which is simple and widely used in solving classification problems. The main idea is to break down multi-class classification into simpler subsets, stage-by-stage and finally reach the classification results. A decision tree contains multiple internal nodes, and there is a series of carefully crafted test conditions about the features of the test dataset at each internal node. Starting from the root node, new datasets are tested over every test condition and follow the appropriate branch based on the outcome of the test. Each time it makes a decision, a follow-up condition is tested until a conclusion about the class label of the record is reached.

In multi-class classification, the curse of dimensionality is a crucial issue: as the number of classes increases, one usually has to select more features and make decisions in a high-dimensional feature space. Therefore, in order to collect enough training data that is representative of the nature of each class, a huge amount of ground truth data is required as the dimension of the feature space increases. Without sufficient training data, the predictive

power suffers from increasing dimensionality [8]. The decision tree classifier assumes that each class is conditionally independent and performs many classifications targeting smaller classes, instead of a single stage with a huge number of states. Thus each decision is done in a feature space with lower dimensionality. Because of its nature of divide and conquer of the decision-making procedure, the decision tree classifier avoids the curse of dimensionality in multivariate analysis [6][7].

In this research, we use decision tree classifiers with a combination of the Naïve Bayes classifier and support vector machine (SVM) as the decision making kernel in internal nodes for activity classification. Fig. 1.5 shows an example of a classification tree for 6 activities. More details of Naïve Bayes classifiers and SVM will be discussed in the following sub-sections.

III.2 Naïve Bayes Classifier

The Naïve Bayes Classifier is based on the Bayes theorem. It assumes every feature to be independent from the others, and they are Gaussian distributed. Despite its simplicity, Naïve Bayes can often yield better performance than most sophisticated classification algorithms. In addition, the Naïve Bayes classifier also inherits the prosperities of the Bayes classifier, which has the advantage to be more extensible. It requires little effort in classifier retraining and software update upon further expansion or modification of the activity classes.

III.2.1 PDF Estimation

For a given set of feature values, $F = \{f_1, f_2, ..., f_n\}$, the posterior probability for the class C_j among a set of possible activity classes $C = \{c_1, c_2, ..., c_d\}$ is to be found. Using Bayes' rule:

$$p(C_j|f_1, f_2, ..., f_n) \propto p(f_1, f_2, ..., f_n|C_j)p(C_j)$$

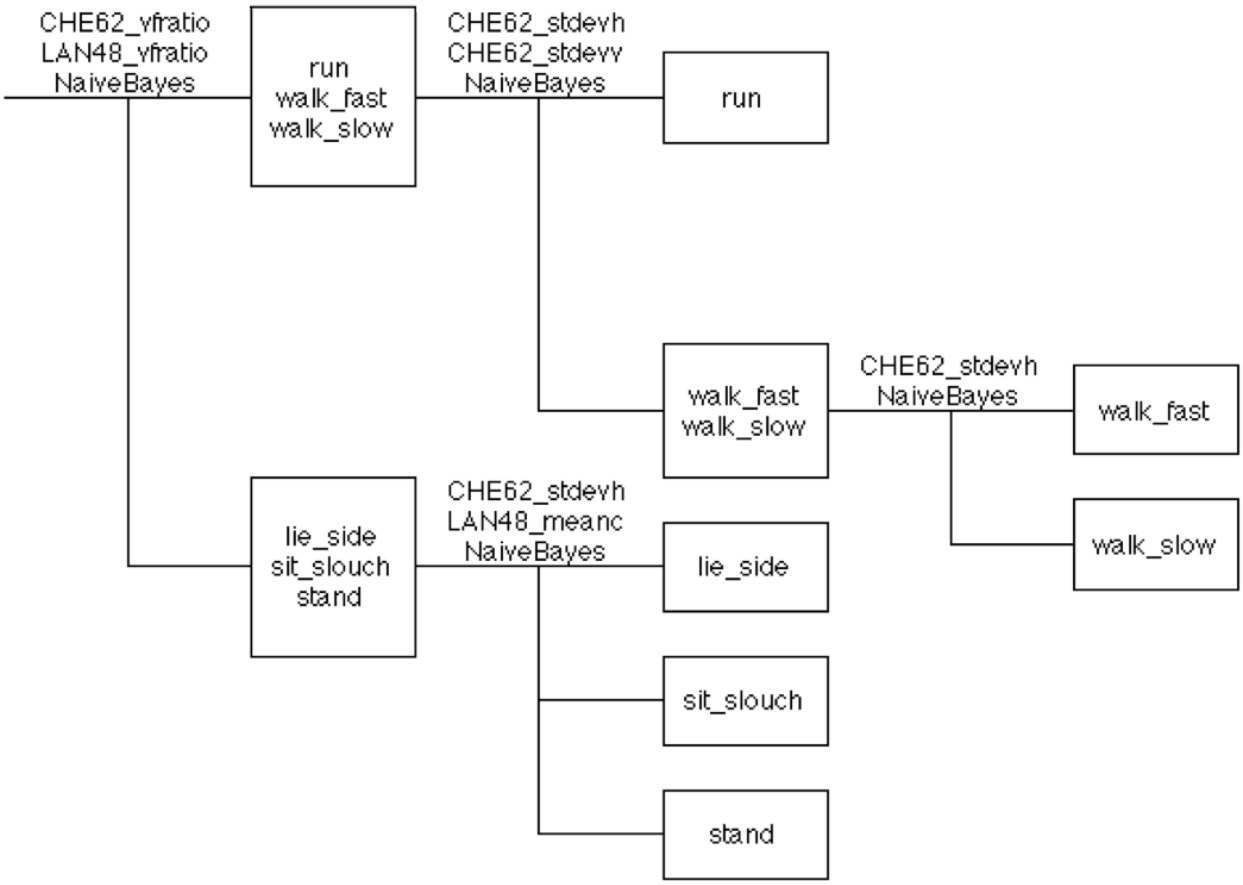


Figure 1.5: Example of a decision tree classifier for 6 activities

where $p(C_j|f_1, f_2, ..., f_n)$ is the posterior probability of class membership, that is, the probability of F belongs to C_j. Since Naïve Bayes assumes that the conditional probabilities of the independent variables are statistically independent, the likelihood can be decomposed to a product of terms:

$$p(F|C_j) \propto \prod_{k=1}^{n} p(f_k|C_j)$$

and the posterior rewritten as:

$$p(C_j|F) \propto p(C_j) \prod_{k=1}^{n} p(f_k|C_j)$$

III.2.2 Classification

Using Bayes' rule above, the classification result of F is labeled with the C_j that achieves the highest posterior probability.

$$C(f_1, f_2, ..., f_n) = \underset{c}{\operatorname{argmax}}\ p(C = c) \prod_{i=1}^{n} p(f_i|C = c)$$

Although it is not always accurate to assume that the features are independent, this assumption does simplify the classification task dramatically, since it allows the class conditional densities $p(f_k|C_j)$ to be calculated separately for each feature. In effect, Naïve Bayes reduces a multi-dimensional density estimation task to a one-dimensional kernel density estimation. Furthermore, the assumption does not seem to greatly affect the posterior probabilities, especially in regions near decision boundaries, therefore, leaving the classification task unaffected.

III.3 Support Vector Machine

A Support Vector Machine (SVM) [15–17] is a non-probabilistic, discriminative classifier formally defined by a separating hyperplane. Given labeled training data (supervised learn-

ing), the algorithm tries to find the optimal hyperplane which has the largest gap that can separate the classes. For example, suppose we have n different features for each sample data. SVM forms a $n-1$ dimensional hyperplane that divides categories given features vectors of n elements lying in a n-dimensional space. Fig. 1.6 shows an example of classification of two classes using the SVM classifier and two features. As shown in the figure, the solid shapes form the support vectors, and SVM tries to find a gap that maximizes the distance between the support vectors.

There are two main benefits of using the SVM in activity classification. First, when using the SVM at internal nodes of tree classifiers, the SVM can effectively draw a decision boundary for activities that are not easily characterized by probabilistic models such as the Naïve Bayes classifier. For example, a single-peak Gaussian model may be insufficient if there is more than one activity within an internal node of a decision tree classifier. On the other hand, the decision boundary of the SVM is only determined by its support vectors and not affected by interior feature points. This also leads to another benefit of using the SVM, which is the robustness when trying to classify activities using features that are susceptible to noise. For example, some features of stationary activities such as energy for siting and standing tend to produce small feature values and thus are susceptible to any external noise or unexpected movements. As a result, the means and standard deviation may not be very representative. In this study, we mainly use the SVMs to classify stationary activities such as standing, lying down, and sitting because the training data are concentrated and a few outliers occur.

III.4 Wireless Health Institute Sensor Fusion Toolkit (WHISFT)

The UCLA Wireless Health Institute (WHI) uses activity classification extensively for both real world and instructional deployments. For real-world deployments we have been collaborating with the UCLA Neurology department and the UCLA Ronald Regan Hospital. There were a number of studies ranging from congestive heart failure patients [18] to intensive care

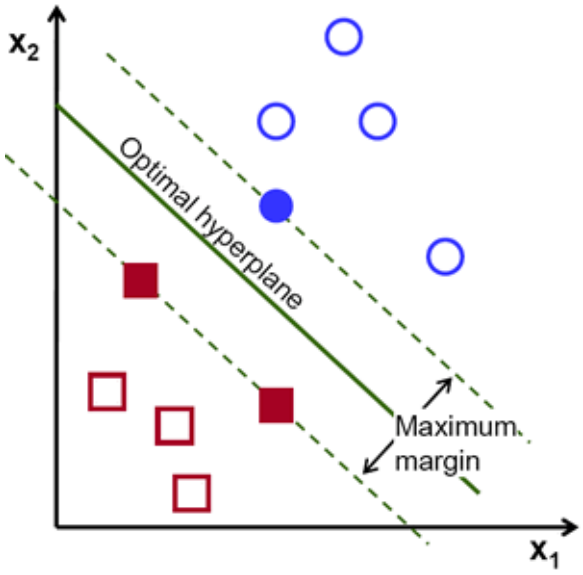


Figure 1.6: Example of SVM on 2 classes

unit cycling restorator [19]. One of the high impact studies included monitoring of exercise intervention effectiveness of acute stroke patients from 150 sites in 12 countries [20]. In these studies, sensors were provided to outpatients to enable follow-up physical activity monitoring in community. Instructional deployments include EE180D and EE202C classes held each year, where students are given projects that allow them to use the sensors and the activity classification techniques to perform studies such as sports activity efficiency, daily energy expenditure and workspace wellness. To facilitate the use-cases, we developed the Wireless Health Institute Sensor Fusion Toolkit (WHISFT). The WHISFT is a suite of accurate classification methods for user activities that has undergone testing in diverse situations and clinical settings [21, 22]. It provides multimodal hierarchical classification based on a set of classifiers such as Naïve Bayes and Support Vector Machine [20].

Starting with raw data from multiple sensors, WHISFT combines streams of data into a single structure. Features such as short time energy, mean, and variance are computed from the combined data structure. There are a number of diverse features, providing freedom in selecting the ones that best suit each application. From the selected features, hierarchical structures (decision tree) can be built to model the classification problem. The decision tree

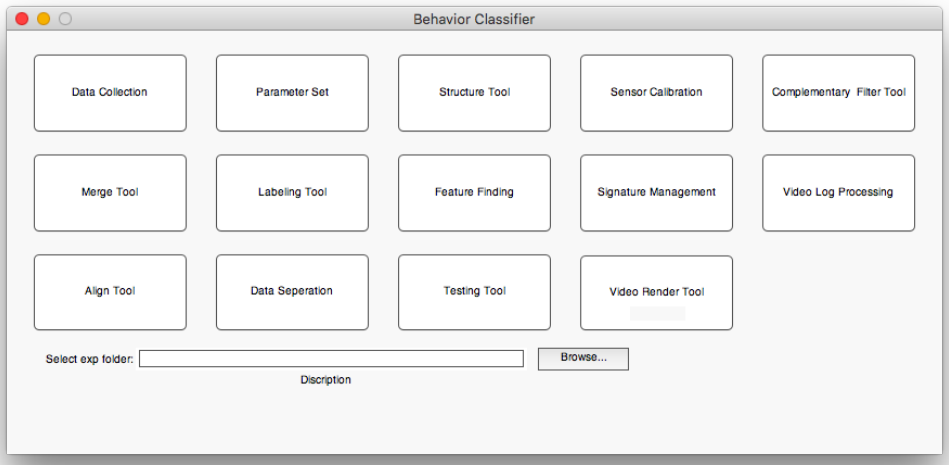


Figure 1.7: WHISFT: function menu

classifier first grossly separates activities into groups sharing similar features, and then further isolates activities within each group using additional features until all can be identified. At each level of the tree, WHISFT uses either a Naïve Bayes or SVM classifier to separate unknown data into one of the branches. The final classification result is produced when a leaf node is reached. Fig. 1.7, 1.8, 1.9, 1.10 and 1.11 show the interface and some sample results of the toolkit.

IV Background 3: Motion Reconstruction with Inertial Sensors

Two of the most widely used inertial sensors for motion reconstruction are gyroscopes and accelerometers. Fig. 1.12 shows the algorithm for trajectory reconstruction.

An accelerometer measures the summation of gravity and motion acceleration. Theoretically, if the motion acceleration can be perfectly isolated, double integration of the motion acceleration yields the motion trajectory. However, it is not trivial to detect the direction of gravity and isolate the motion acceleration when the sensor is experiencing external forces. The gyroscopes can be helpful in determining the direction of gravity. Gyroscopes output

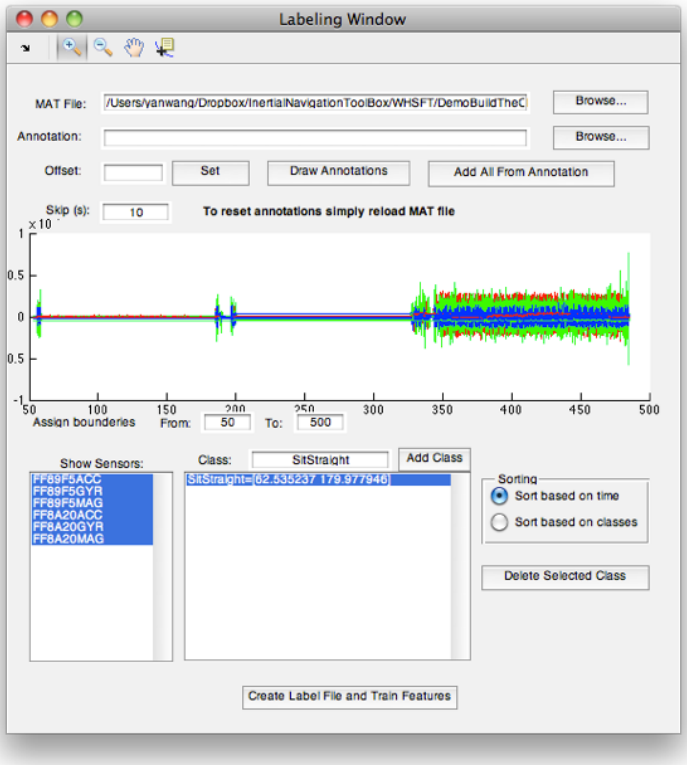


Figure 1.8: WHISFT: data processing (alignment and labeling)

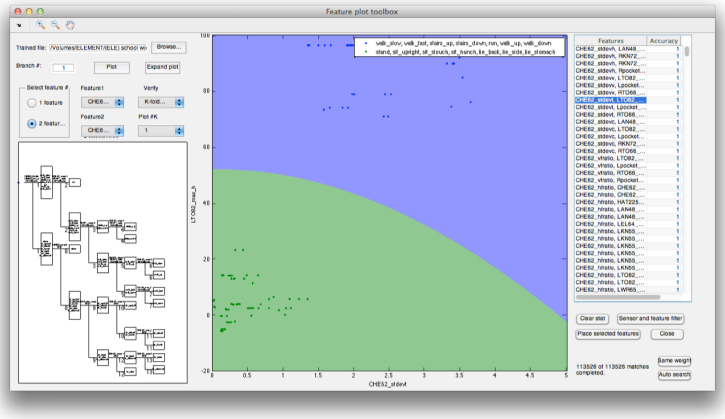


Figure 1.9: WHISFT: decision tree and feature selection

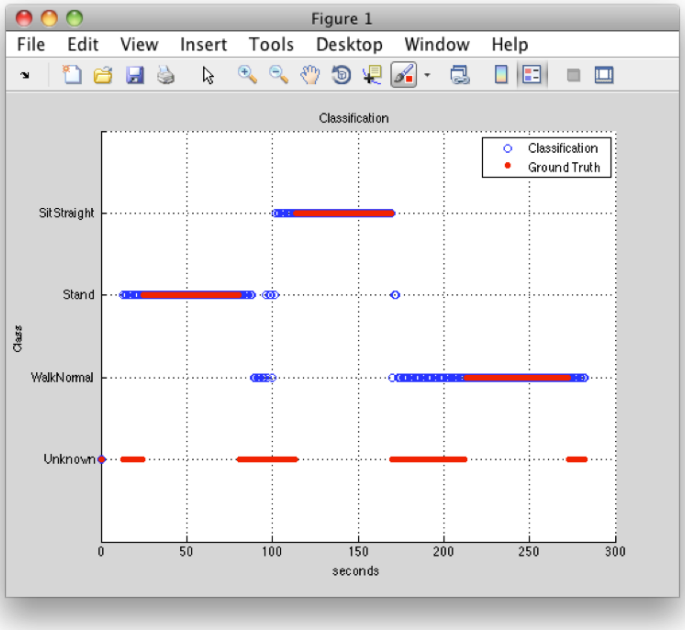


Figure 1.10: WHISFT: classification result

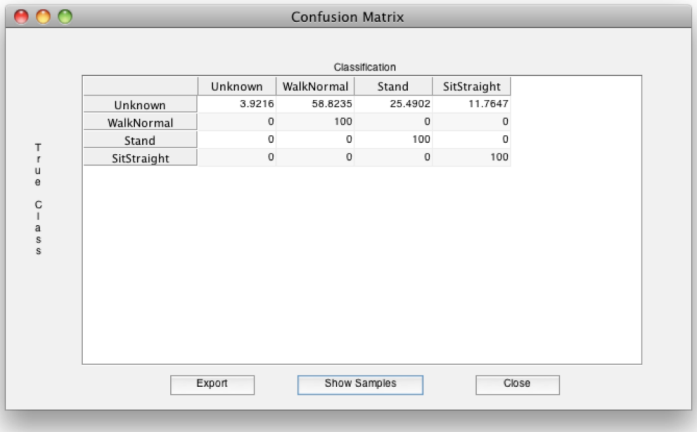


Figure 1.11: WHISFT: classification statistic

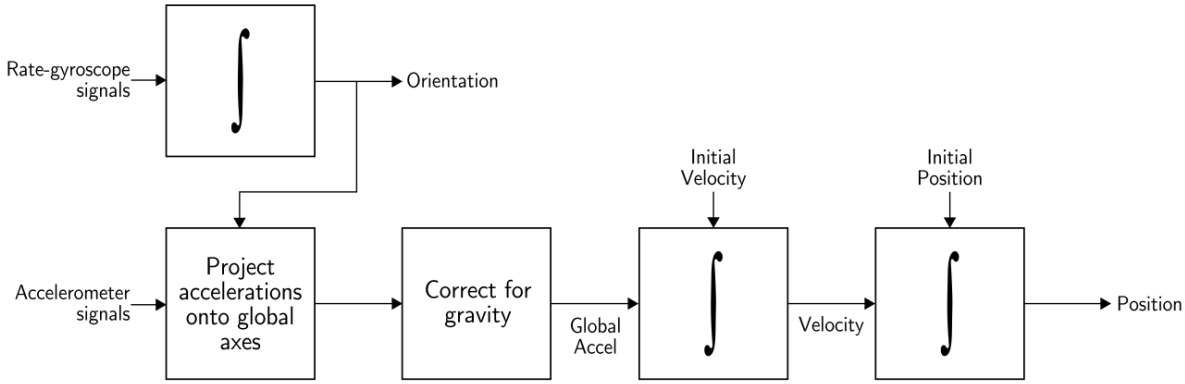
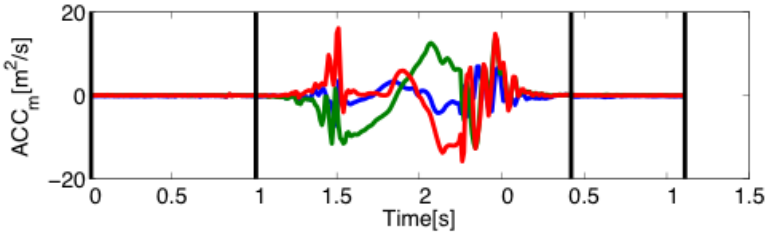


Figure 1.12: Trajectory reconstruction algorithm

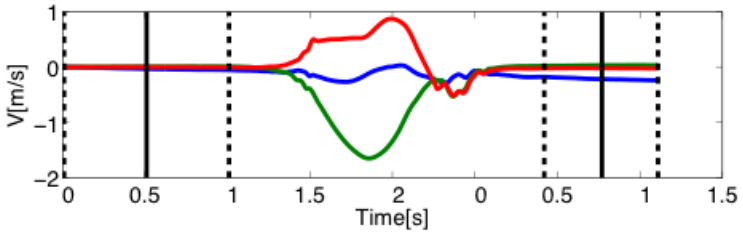
angular velocity, and we integrate the angular velocity for orientation angles. The issue is, gyroscopes have bias, which means its reading won't be exactly zero when the sensor is stationary. In addition, the bias is not fixed nor predictable. This means we cannot get accurate angle from gyros because it will drift over time. As a result, even with the information from gyroscopes, we cannot get accurate motion acceleration from accelerometers without careful calibration.

IV.1 Zero-velocity Update (ZUPT)

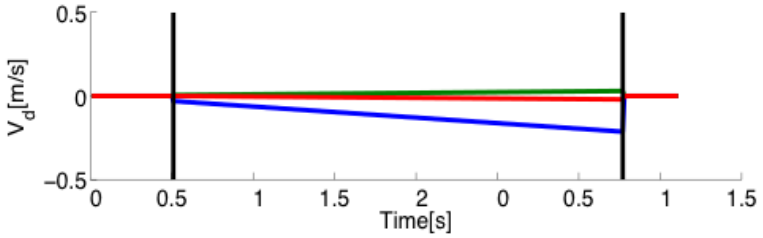
The zero-velocity update (ZUPT) is a commonly used technique to reduce sensor drift [23]. The idea is to set the velocity to zero when the sensor is stationary and smooth the velocity in between before being integrated for position. By doing this, drift can be detected and compensated before it propagates to position estimation. Fig. 1.13 show an example of using ZUPT to compensate velocity drift. However, ZUPT cannot always perfectly remove the velocity drift because there exists some uncertainties when detecting zero-velocity windows. For example, the zero-velocity window might be too short for the algorithm to detect, or the declared zero-velocity window is a false positive. Therefore, many methods have been proposed to provide more accurate trajectory estimation such as using human bio-mechanical models [24], sensor fusion [25], and Non-ZUPT [26].



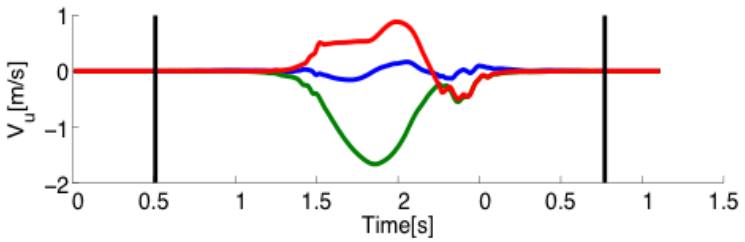
(a) Captured accelerometer data



(b) Double integrated result including drift



(c) Estimated linear drift



(d) Double integrated result after ZUPT is used to remove drift

Figure 1.13: An example of zero-velocity update.

Figure 1.14: Shoe-mounted sensor

IV.2 Inertial Navigation System (INS) Toolkit

The UCLA Wireless Health Institute developed the Inertial Navigation System (INS) toolkit [27] for gait trajectory reconstruction and visualization. The INS toolkit allows researchers to collect motion data from shoe-mounted sensors (Fig. 1.14) and reconstruct motion trajectories. This toolkit has been deployed in many studies ranging from quality assessment of hemiparetic gait [27] to calibration of upper limb motion tracking [28]. For instructional deployments, students in EE180D and EE202C were given projects that allow them to use the sensors and the toolkit to perform studies such as gait cycle analysis and indoor localization. Fig. 1.15, 1.16 and 1.17 show the interface and some sample results of the toolkit. In this research, we used some methods from this toolkit to perform ZUPT and trajectory reconstruction for upper limb movement.

V Organization

The rest of the chapters of this dissertation are organized as follows: In Chapter 2, we present a real-time context-guided activity classification system which can deal with increasing model

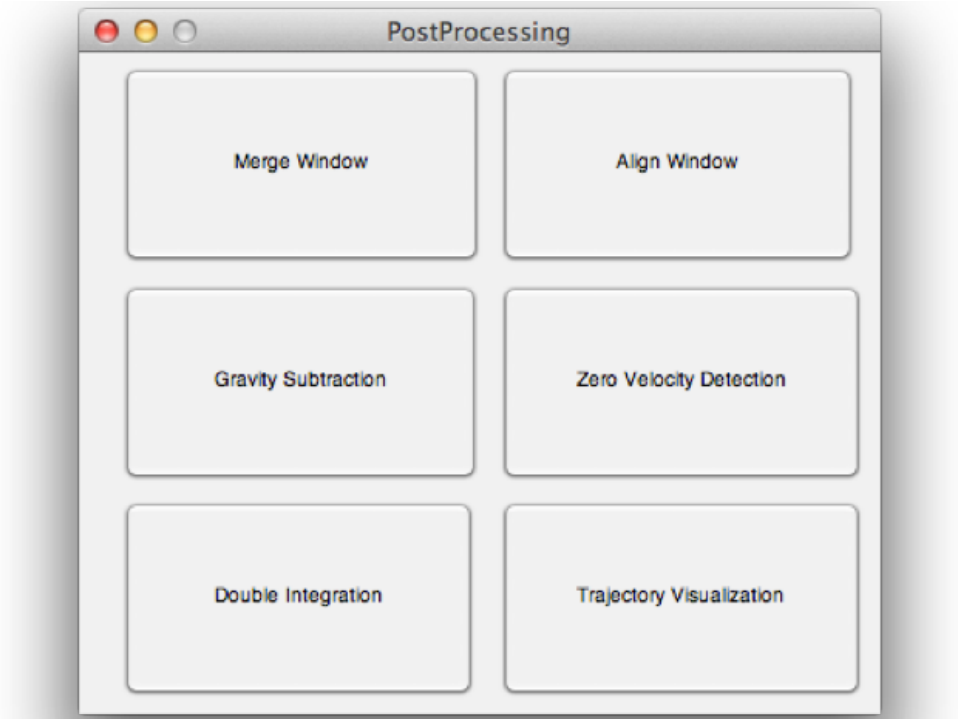


Figure 1.15: INS: function menu

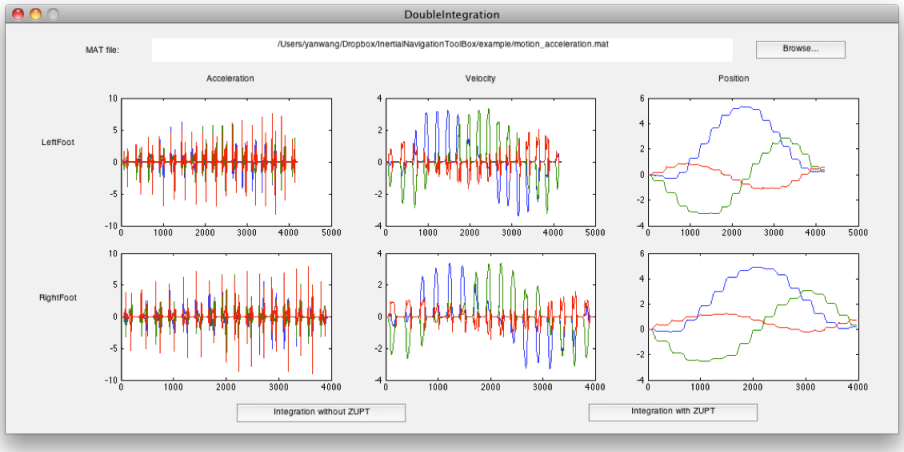


Figure 1.16: INS: zero-velocity update

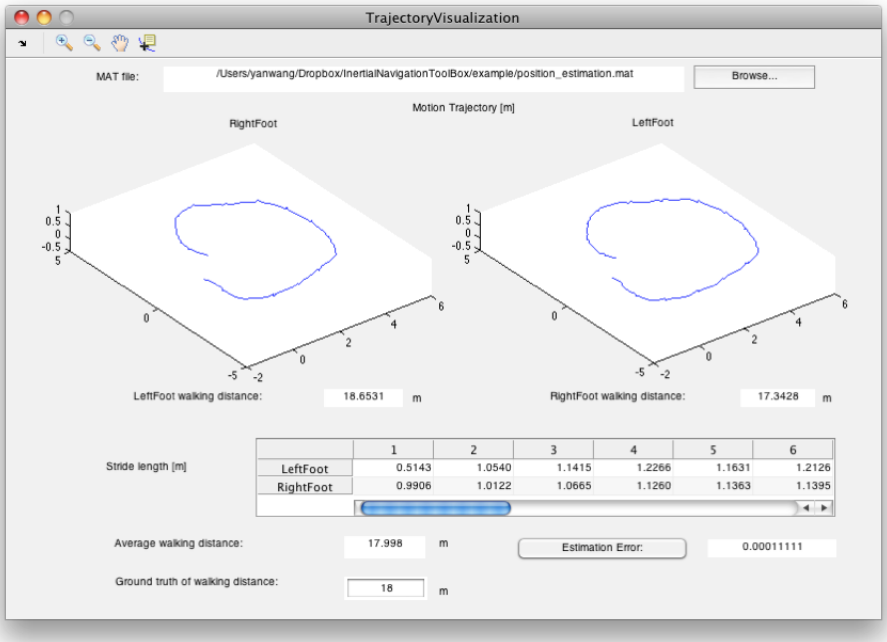


Figure 1.17: INS: reconstructed trajectory

complexity inherent in large-scale populations. Chapter 3 describes a virtual inertial measurements platform which converts the currently available camera motion database into a noiseless and error-free inertial measurements database. In Chapter 4, we propose an opportunistic calibration system which can detect and compensate sensor placement errors. In Chapter 5, we present a sensor fusion approach for robust upper limb motion tracking. Chapter 6 summarizes this thesis, and presents suggestions for future research.

CHAPTER 2

Context-guided Universal Hybrid Decision Tree for Activity Classification

I Introduction

Activity monitoring provides critical benefits for important concerns such as health and wellness promotion, disease treatment and disease condition detection. Through the automatic feedback of activity status to both individuals and health care providers, the quality of health can be improved while reducing the costs. Due to the rapid advance in microelectronics, MEMS inertial sensors, low power processors, and low cost monitoring systems, human activity classification is now possible. The ubiquity of mobile devices also provides a platform for the wireless healthcare community to integrate monitoring and in-field guidance for both advancing and evaluating treatment outcomes. Increased research effort has been devoted to the development of systems that monitor human activities with feasible cost, classify activities with good accuracy, and then analyze these activities with respect to different rules [29, 30].

Some systems [31, 32] based on Naïve Bayes classifiers can provide accuracy up to 90% for classifying a small number of daily activities. However, the use of a single-stage classifier is problematic from many aspects including exploding training data requirements as the number of classes grows. Other approaches [33–35] utilized decision tree classifiers that

can better handle complex decision regions by partitioning them into smaller sets with low dimensional hypothesis spaces at each stage, providing advantages such as reduced training set size, robustness to outliers in training data, extensibility of target classes, and invariance under monotone transformations.

However, decision tree methodologies can suffer from mismatches between assumed and actual distributions for different sets of classes, resulting in poor accuracy, if only a single classifier type at each node is applied. Another issue arises in clinical trials when generalizing the model to a large population. In practice, one can acquire extensive ground truth only for a small set of subjects due to high logistical costs; for the rest, at best only short training is feasible. However, if the tree can be personalized then we can get far better results. One solution is to construct a decision tree structure that fits the population, and then tune only the decision thresholds using short training sequences for individuals. This was attempted in [29], but with inadequate accuracy.

The above methods also face challenges as we scale to large and diverse user communities. The rapidly expanding activity set increases model complexity, which causes degraded classifier performance. In addition, the diverse user community has varied requirements. Thus we need a system that is personalized and provides targeted monitoring of activities under different conditions. The energy efficiency of energy-constrained monitoring sensors should be taken into consideration as well. These objectives require the capability of detecting the location and environmental context [36, 37]. Context information has the potential to directly enhance activity classification accuracy and speed through reduction in search space, and reduce energy demand through context-aware optimization of sensor sampling and operation schedules.

There have been attempts to introduce context awareness into activity classification to facilitate personalization and adaptation [8, 36, 38–40]. These systems achieved limited success due to the ambiguity in the definition of context, and the lack of a system architecture that enables the adaptation of signal processing and sensor fusion algorithms specific to the

task of personalized activity monitoring.

To address the above-mentioned deficiencies, we propose: 1) A universal hybrid decision tree classifier to reduce training efforts; 2) A novel architecture that provides context guided personalized activity. Herein, context is separated from physical activities in order to produce a first level hierarchy, and further achieves personalized activity classification. In addition, our work presents four major contributions: 1) A tree classifier with flexibility of decision rules, adaptation to a population-based model and reduction of training cost; 2) Accurate detection of context with sensor fusion; 3) The integration of context to improve classification accuracy and energy usage; and 4) The ability to target specific physical activities of interest for a given context.

II System Overview and Architecture

Illustrated in Fig. 2.1, the system consists of three parts: sensor modules, an Android device, and a backend server for offline training. Multiple sensor modules, each containing three sensors (gyro, accelerometer and magnetometer), are attached to the body. Each sensor module communicates wirelessly (dashed lines) with an Android device via Bluetooth.

The sensor modules sample data at a predefined rate, aggregate data from each sensor, and then transmit to the Android device. In the training phase, after the user employs the GUI to configure and turn on the sensors, the sensors generate and transmit data to the Android device. Meanwhile, context sensors on the Android device collect environment data such as Wi-Fi fingerprint, audio and time of day. The Android device stores these data locally. The system then prompts the user to provide ground truth labeling for each activity section and current context. When the collection of training data is done, both the sensor data and annotation files are stored in the Android device.

These training data are then used to perform offline model training via the backend server, which consists of two toolboxes. WHISFT is a suite of accurate classification methods for

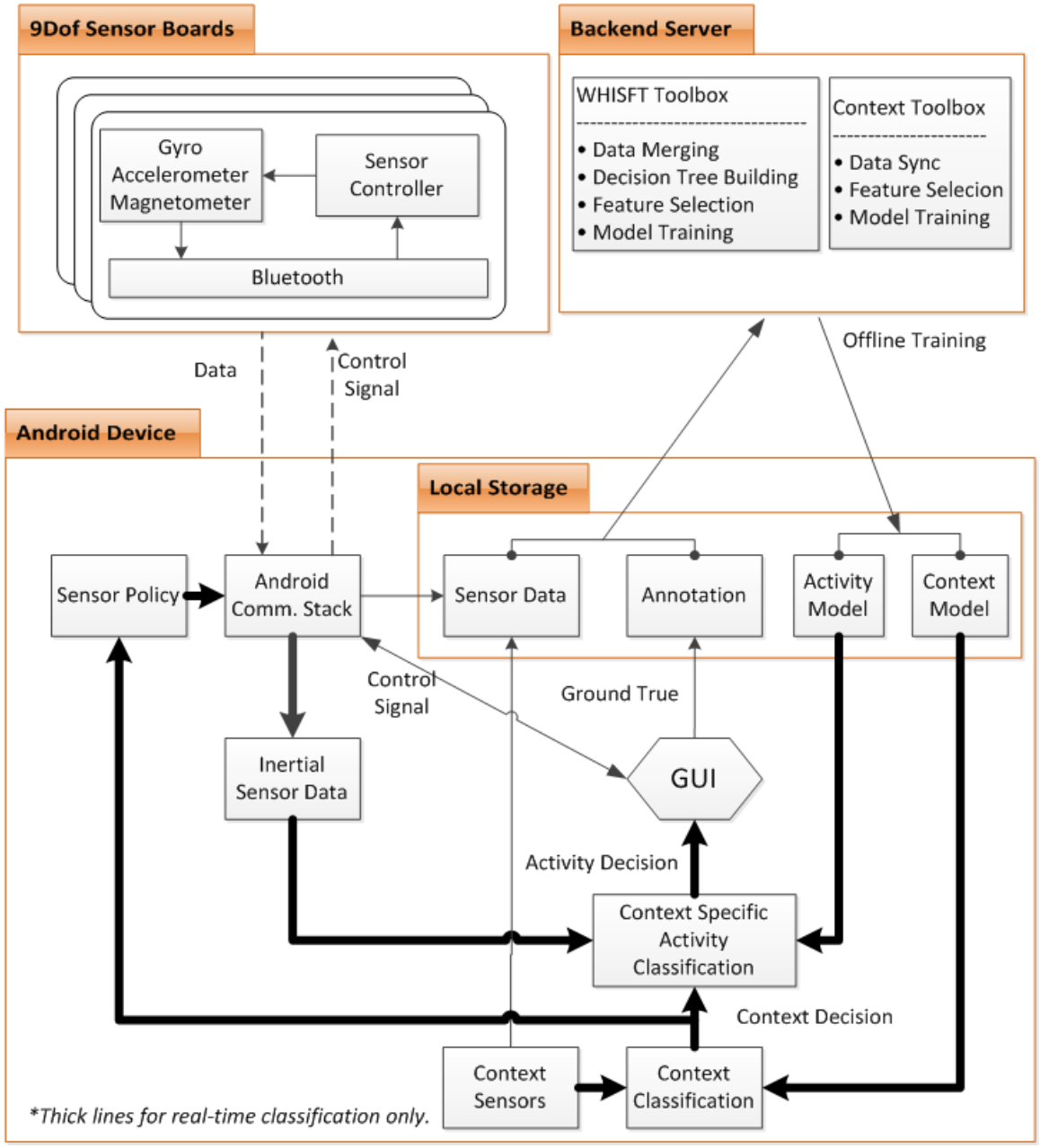


Figure 2.1: System architecture

activities classification that has undergone extensive testing in diverse situations and clinical settings [21]. It provides an end-to-end solution for inertial sensor data processing from raw data to decision tree construction, model training and performance evaluation. The toolbox is capable of performing multimodal hierarchical classification based on a set of classifiers such as Naïve Bayes and Support Vector Machine [20]. The context toolbox is another tool we developed that is able to process context data and build context classification models. We used these two toolboxes to construct and train our classification model based on our proposed algorithm. The models are then transferred to the Android platform for real-time classification.

In the real-time testing phase, the Android App not only stores sensor data locally but also caches data in a queue structure. It also loads the trained model into its classifier. A queue structure is designed to implement a moving window for real-time data processing, where new data is pushed onto the queue, and old data is popped out. The data from the moving window is then fed into the classifier to make a classification decision. The context decision is first determined and then fed into the context specific activity classification block. Based on the context information, a specific activity model is selected to perform activity classification on the inertial data. The classification results are finally made, and then fed back to the user via the GUI.

III Methodology

III.1 Universal Hybrid Decision Tree

In this section, we present a universal hybrid decision tree classifier. This type of classifier fuses various kinds of single-stage classifiers in its nodes, and can also adapt to new incoming data with minimal training. The following shows how we achieve this.

First, a tree classifier T with l internal nodes can be thought of as a classifier consisting of l single-stage classifiers, where each single-stage classifier has its own subset of classes,

features and the decision rules for the node. Therefore we can write T as a combined set,

$$T = \{C(t), F(t), D(t)\} \quad t = 1, ..., l \tag{2.1}$$

where C is the subset of classes of node t, consisting groups of classes associating with that node; F is the feature set used for node t; and D is the decision rule of that node. In this research, the Naïve Bayes classifier and the support vector machine (SVM) were used as possible types of decision rules of internal nodes. Compared to other tree classifiers where only a single type of decision rule is used [41, 42], this hybrid approach takes advantage of more appropriate statistical modeling of different activity classes and therefore achieves higher classification accuracy.

Using this hybrid tree, we then find a classifier with a single structure that can classify multiple subjects' data. The reason behind this universal classifier is that we want to have a model that with minimal additional training can be personalized to subjects. Therefore when we generalize this model the amount of training effort, such as data collection and labeling, can be greatly reduced. We do this by maintaining the structure, features used and decision rules associated with the hybrid tree classifier, and only change the decision thresholds corresponding to different subjects. Thus using only a small amount of additional training we can personalize the classifier to each subject. This procedure can be stated as in Fig. 2.2. This algorithm takes the differences among people into account while maintaining a satisfactory error rate.

III.2 Context Detection

In pervasive computing, the definition of context by Dey [37] has been widely referenced. It is a very broad definition that contains every characteristics of a given situation, in terms of both the environment and the user. While useful for many applications, it is not suitable for leveraging context in monitoring physical activities, since in many cases a context contains

Begin

1. Given we have M subjects with training data named $TD_i, i = 1,\cdots,M$, and we manually form N hybrid decision tree where is tree T_i with l nodes can be written as

$$T_i = \{C(t), F(t), D(t)\}$$
$$i = 1,\cdots,N \quad t = 1,\cdots,l$$

with $l(i)$ internal nodes. The class subset $C(t)$ is determined for every internal node. Let $TD_{j,t}$ be part of the training data TD_j whose classes that are involved in node t of the tree. $P_{e|j,t}(F(t), D(t)TD_{j,t})$ is the probability of error of node t when applying feature set $F(t)$ and decision rule $D(t)$ on training data $TD_{i,t}$.

2. Randomly pick a tree T with l internal nodes
3. **For** $t = 1$ to l

   Find the optimal set $(F^*(t), D^*(t))$ that minimizes the weighted probability of error

$$(F^*(t), D^*(t)) = \arg \min_{(F(t),D(t))} \sum_{j=1}^{M} w_j \cdot P_{e|j,t}(F(t), D(t), TD_{j,t})$$

   where w_j is the weighting function for the subject j , indicating the weighting of that type of people to the general public

4. **If** $\sum_{j=1}^{M} w_j \cdot P_{e|j,t}(F^*(t), D^*(t), TD_{j,t}) > th_{err}$

   Terminate the for loop, go to step 1 and try the next tree T , where th_{err} is the predefined error threshold

   End if

   End for

5. Output the tree classifier

$$T^* = \{C(t), F^*(t), D^*(t)\} \quad t = 1,\cdots,l$$

Figure 2.2: Algorithm of generalizing the hybrid decision tree

physical activities that are underlying in the definition. Some alternative definitions offer different selection of divisions such as external and internal contexts [43, 44] to narrow the extent, but still contain a mix of physical activities with the external environment.

In this study, we address a context as: "a subset of all attributes that characterizes an environment or situation, external to the user." This definition clearly distinguishes between the user's physical activities and external environmental attributes. With this refined definition, the attributes associated with a context or with a physical activity can be easily distinguished. For example, a "cafeteria" environment is a context, and its characteristics may involve certain sound profiles and a set of possible locations. In contrast, "eating in a cafeteria" is not a context, since it contains the user's physical activity of "eating". Thus, we can use context as a first level hierarchy to determine a set of activities of interest based on the user's current situation before carrying out activity classification [45].

Since our definition of context can describe many situations, it allows users to define their context set of interest, identify the required characteristics to distinguish contexts and select necessary sensors based on their objectives. Thus, this generalization requires the system to take account of diverse types of data sources such as GPS coordinates, Wi-Fi fingerprint, background audio noise, and illumination level.

To provide a reliable context decision, multiple classifiers should be employed based on the nature of various data sources and trained separately. After training, the individual classifiers are tested and assigned with voting weights (α) proportional to the perceived accuracies. When an unknown class is encountered, a decision committee (Fig. 2.3) performs sensor fusion as a linear combination of the individual classifiers. The context with the highest vote is chosen. The committee approach also enables adaption to individuals with varying habits. For example, a subject with a regular daily schedule might exhibit higher correlation in time of day relating to context. Thus, we would increase the weight of the classifier based on time-of-day during training, compared to a subject that is less habitual. We choose three classifiers to form our context detection committee for most of the experiment: k-nearest

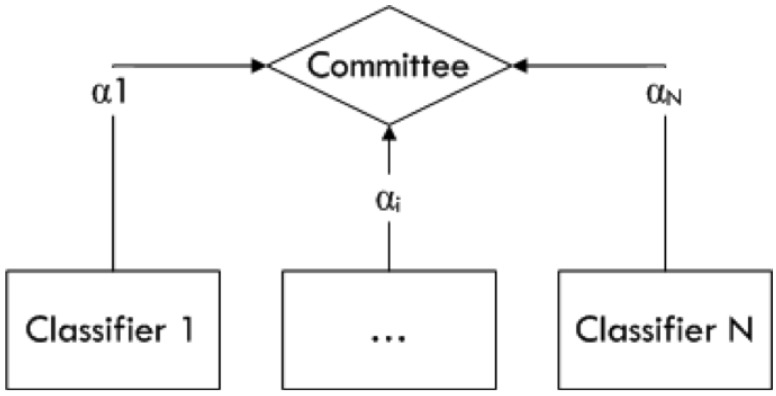


Figure 2.3: Context classification committee

neighbors (kNN) with time-of-day as a feature; kNN with wireless MAC address and signal strength as features; and AdaBoost with audio peak frequency, peak energy, average power and total energy as features.

III.3 Integration of Contexts into Activity Classification

After inferring context from the committee, this information can be used to enhance activity classification. We introduce the concept of context driven activity classification. Fig. 2.4 shows a high-level data flow diagram. The inertial data and context data go through a signal-processing pipeline where a context is first determined. From the context we can extract an activity model from a scenario. The activity model combined with the inertial data gives us an activity classification result.

Based on this framework, there is no single list of comprehensive activities that needs to be built into a monolithic classifier, compared with conventional activity classification. Alternatively, only a small set of activities would be chosen in a specific context, and this set can then be extended or reduced according to our objectives.

This approach brings a number of advantages. First, by pre-selecting the activities of interest (or likely activities), the model complexity of the subsequent activity classification stage can be reduced. This increases the accuracy, improves classification throughput and

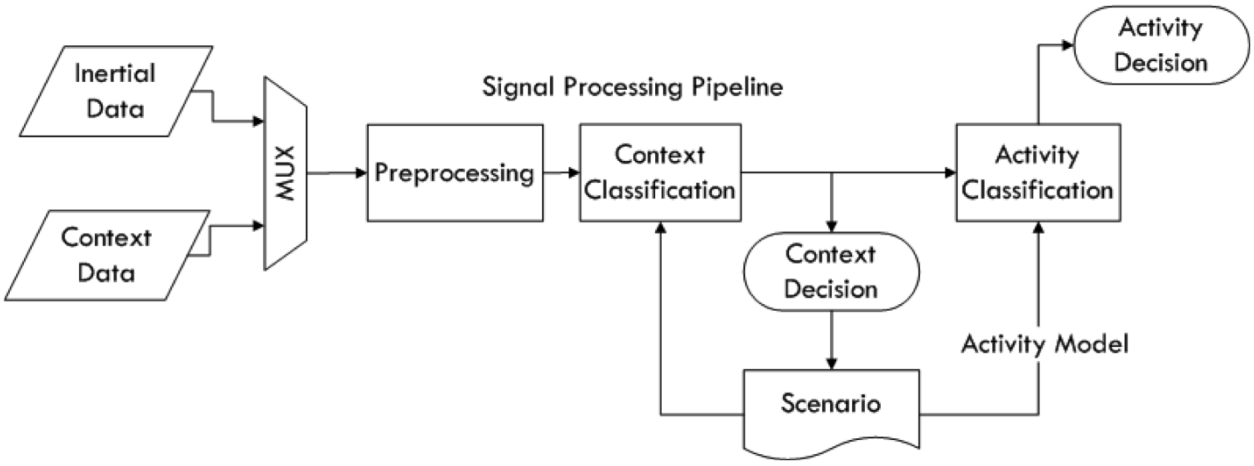


Figure 2.4: High-level decision flow

enables sensor operating time and data sample/transmission optimization. An example of the context specific activity model we generated for the "Cafeteria" context is shown in Fig. 2.5. The activities are on leaf nodes, laid out in a hierarchy. At each branch either a Naïve Bayes or SVM classifier makes the branching decision using features in the model.

In addition, this approach also allows an activity set within a context to be customized to fit a specific situation. To further illustrate this concept, Table 2.1 lists a few possible activity models under different contexts in a clinical application. For example, in the context of patient room, physicians may wish to monitor a patient's mobility status to assess the risk of bedsores and other problems. Another example is the rehabilitation context, where physicians may wish to monitor the patient's performance in exercises and to ensure recommended daily activities are performed as instructed.

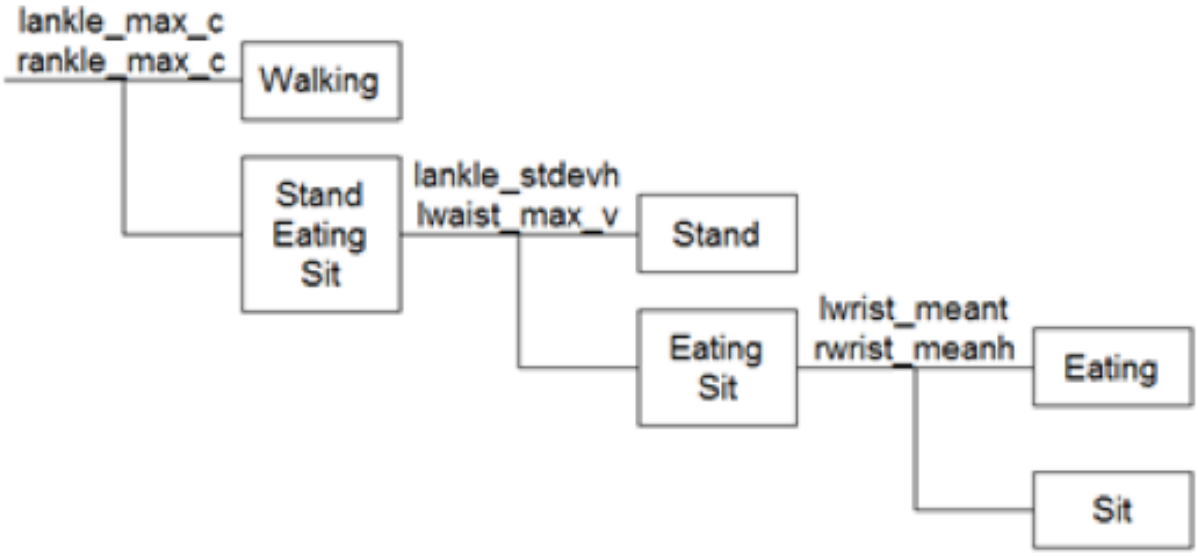


Figure 2.5: Context guided model for cafeteria

Table 2.1: Example Scenarios

Context	Activity Model	Purposes
Patient room	• Sitting • Standing • Lying down	Monitor how long a patient has stayed immobile, assess the risk of bed sores and other problems
Rehabilitation	• Aerobic exercise • Walking Slow • Walking fast • Fall	Monitor patient's performance in exercises
Hall way	• Standing • Walking fast • Walking slow • Fall	Monitor a patient's general physical condition, and detect falls

IV System Evaluation

IV.1 Data Acquisition

In this study, we used SparkFun 9DoF IMU sensors and a Nexus 7 tablet to collect 14 datasets, where each set of data contains 13 activities in 8 different contexts[1]. The procedure of data acquisition is as follows: 14 subjects each attached four 9DoF sensors on right wrist, knee, ankle and mid waist. An assistant carried a Nexus 7 tablet running the Android client to record sensor data and label the ground truth. Each subject spent 30 minutes in each context, and performed every predefined activity under that context for at least 5 minutes. The data were then separated into training (30%) and testing (70%) sets and 10-fold cross-validation was performed to obtain the classification results. Table 2.2 summarizes the collected activities corresponding to different contexts. In the table, the activity "Walking Around" refers to non-sustained walking segments that are typical of walking in confined spaces, while "Walking Normal" refers to sustained long distance walks typical of open space.

IV.2 Result

IV.2.1 Context Classification accuracy

The accuracies of correctly classified instances of individual classifiers in the committee and the overall accuracies are shown in Table 2.3. We noticed that AdaBoost using sound features yield high accuracies for most of the contexts. However, sound features are sensitive to environment variation. There were some cases where misclassification occurred due to vehicles driving nearby or long periods of silence. Time kNN depends heavily on the varied nature of when subjects visit these contexts. Hence, it is also not sufficiently accurate in cases of some spontaneous visit of contexts. Wireless kNN provides good accuracy for indoor

[1]Data collected according to a UCLA IRB approved protocol.

Table 2.2: Scenarios

	Home	Lab	Cafeteria	Outdoors	Class	Bus	Gym	Library
Walking Around	X	X	X		X			X
Walking Normal				X			X	
Walking Upstairs				X				
Walking Downstairs				X				
Sitting Straight	X	X	X	X	X	X		X
Sitting Slouch	X							
Standing			X	X		X		X
Writing		X			X			X
Typing		X						
Eating	X		X					
Sleeping	X							
Running				X			X	
Cycling							X	

Table 2.3: Context Classifier Accuracies (Percentages)

	AdaBoost	Time kNN	Wireless kNN	Committee
Home	100	91	100	100
Lab	78	68	98	95
Cafeteria	100	0	80	100
Outdoors	81	57	56	72
Class	81	43	95	91
Bus	100	23	30	95
Gym	64	9	93	84
Library	59	0	100	94

contexts due to the stable wireless environment. However insufficient accuracies occurred in some cases such as bus and outdoors. In the bus context, the classifier suffers from unstable wireless signals or unseen wireless access points due to the route of the bus. For the outdoor context case, the system tended to detect access points that belong to nearby indoor locations. We observed this issue when walking near a building caused the context to be classified as another context inside the building.

This experimental evaluation reveals the pros and cons of each individual classifier. However, by applying a committee approach that assigned an appropriate combination of each classifier, the system is able to achieve high accuracy for all contexts.

IV.2.2 Activity Classification accuracy

In this section, we first evaluated the classification accuracy of the universal hybrid decision tree classifier, and then verify the enhancement in classification accuracy of the context-guided approach. For the universal hybrid decision tree, we first manually determined the tree structure, and then used 30% of the data from all subjects as training data to select features and classifier types that yield the highest accuracies. After a universal tree is generated, we used the training data of each subject to determine decision thresholds for internal nodes of the tree. The thresholds for each subject have to be determined individually since properties of each set of data are different from other sets. For the case of context-guided

classification, we first performed context classification and follow up by activity classification.

Table 2.4 summarizes classification accuracy in each context. The results indicate that without context information, our proposed tree provides good accuracy in most of the activities, except some activities involved with upper body movement. However, it can be seen that with the integration of context information, there is an overall enhancement in classification accuracy due to the reduction of search space and the size of each classifier. In addition, for those activities involving upper body movement such as typing, writing and eating, a large improvement is observed.

IV.2.3 Potential for Energy Saving

The context driven approach allows us to adjust sensor policy dynamically according to detected context, and thus brings the potential to improve energy efficiency and operation lifetime. Based on scenarios tested in Table 2.2, we formed a sensor requirement profile (Table 2.5), in which blank cells indicate sensors that can be safety turned off without affecting the accuracy of a given context. We evaluated the improvement of system operation time by adopting sensor activation schedules based on contexts. A subject's typical daily schedule on workday and weekend is shown in Fig. 2.6 and 2.7, with the x-axis starting at 8am. Fig. 2.8 shows the comparison of total operation time of context driven sensor activation and continuous sensor activation, which indicates the potential benefits of context driven sensor energy management. This benefit would be more obvious in the situation where many sensors are deployed but only some small subset is required in each context.

V Conclusion

In this study, we demonstrated the advantages of integrating the context and universal hybrid tree classifier for activity classification. The proposed universal hybrid tree structure provides flexibility at the expense of the use of intuition or domain knowledge in its construction.

Table 2.4: Activity Classification Accuracies

Context	Proposed Tree	Proposed Tree with Context	Improve	Context	Proposed Tree	Proposed Tree with Context	Improve
Home				**Outdoors**			
Sleeping	64.47	97.29	32.82	Walking Normal	91.17	93.13	1.96
Slouching	83.18	97.83	14.65	Running	83.26	100	16.74
Eating	91.03	94.24	3.21	Upstairs	82.48	96.84	14.36
Walking Around	92.24	100	7.76	Downstairs	61.09	74.62	13.53
Sitting	83.15	91.42	8.27	Standing	100	100	0
				Sitting	77.23	98.15	20.92
Lab				**Gym**			
Sitting	75.29	84.59	9.3	Cycling	90.7	98.43	7.73
Walking Around	93.82	99.81	5.99	Running	100	100	0
Typing	19.05	92.62	73.57	Walking Normal	100	100	0
Writing	32.89	70.74	37.85	Sitting	81.39	93.68	12.29
Cafeteria				**Library**			
Standing	98.81	100	1.19	Sitting	81.71	98.47	16.76
Walking Around	92.19	98.22	6.03	Walking Around	93.28	96.14	2.86
Sitting	72.89	93.27	20.38	Standing	100	100	0
Eating	86.75	94.73	7.98	Writing	42.79	75.52	32.73
Class				**Bus**			
Walking Around	98.01	100	1.99	Sitting	50.36	95.27	44.91
Writing	33.22	92.77	59.55	Standing	78.44	96.57	18.13
Sitting	97.53	97.53	0				

Table 2.5: Sensor Requirements

	Home	Lab	Cafeteria	Outdoors	Class	Bus	Gym	Library
Right Knee	X			X			X	
Right Ankle		X	X	X	X		X	X
Waist	X		X	X		X		X
Wrist	X	X	X	X	X			X

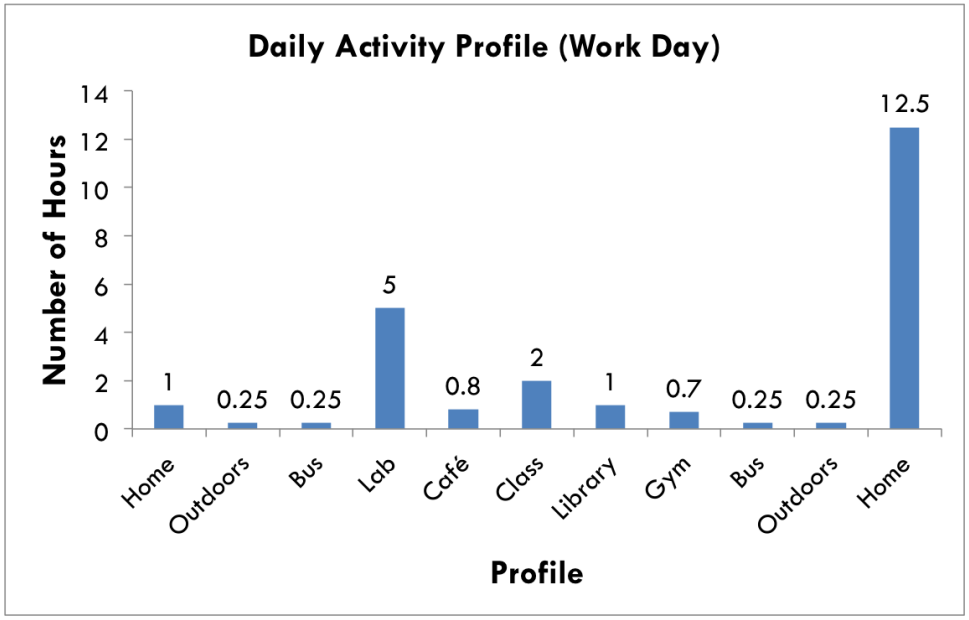


Figure 2.6: User profiles (workday)

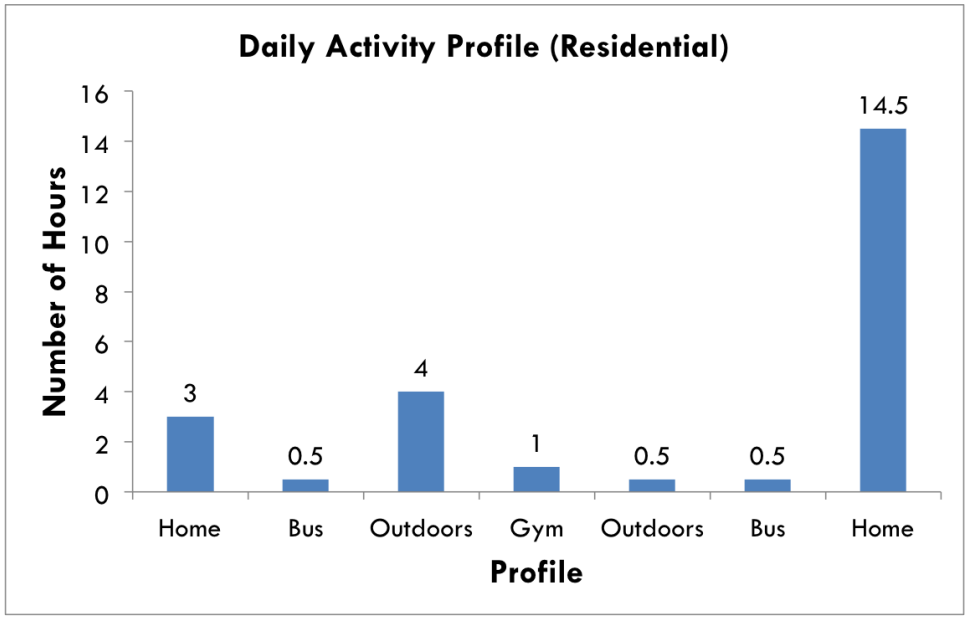


Figure 2.7: User profiles (residential)

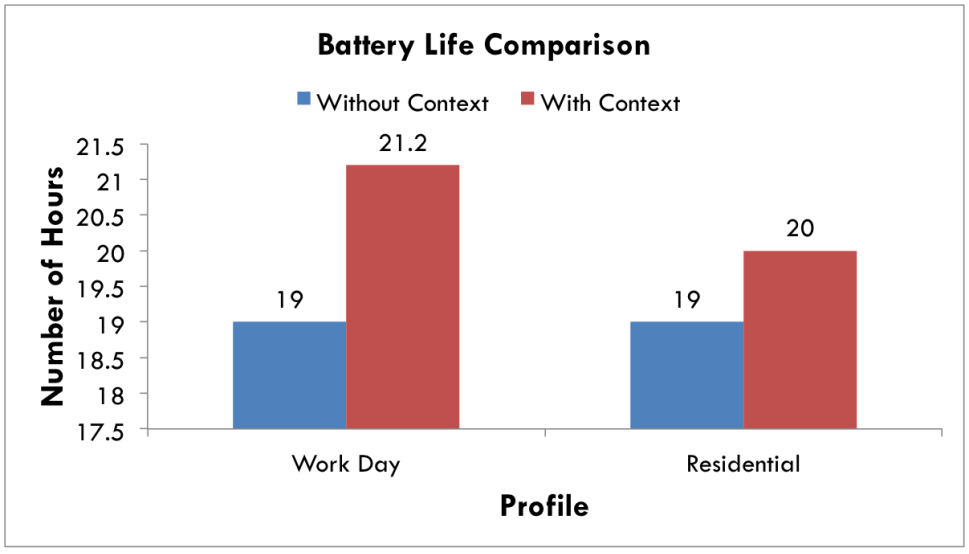


Figure 2.8: Battery life comparison

The effort is rewarded in relative ease of tuning it to new individuals with modest additional training. For scaling to a large population, this could lead to a drastic reduction in effort. In addition, the new context driven approach not only brings improvement in classification accuracy, but also provides the capability of controlling the activation and selection of sensors for energy saving. A number of future research directions are being pursued. Since our context driven approach depends heavily on the quality of context decision, it is of interest how to achieve precise context classification information without needing extensive training.

Remarks During this project, we collected a huge amount of motion data. We also attached multiple motion sensors at different part of human body, and recorded all the required activities with may repetitions. At the beginning it was quite fun, however, after we repeated the same procedure for tens of subjects, include myself, my colleague, and many volunteers, we realized that there could be a more efficient way to collect ground truth data. This led us to the next project "Virtual Inertial Measurements for Motion Inference in Wireless Health", where we present an efficient way to derive ground truth inertial data.

CHAPTER 3

Virtual Inertial Measurements for Motion Inference in Wireless Health

I Introduction

Many of the most urgent problems in health and wellness promotion, diagnostics and treatment of neurological disease require accurate, reliable and detailed monitoring of human motion. Video based motion capturing systems (e.g., Vicon-460 camera) provide a partial solution. However, these expensive and fixed systems cannot be used for patients' at-home daily motion monitoring. Wireless motion sensors, including accelerometers, gyroscopes and magnetometers, can potentially provide a low-cost, small-size, and highly-mobile option [46] [47] [48].

But, several problems must first be overcome. These issues are illustrated in Fig. 3.1(a). Although high-precision sensing systems can provide low-noise data, they can cost several thousand dollars each and are not affordable in most wireless health applications. On the other hand, low-cost accelerometers are non-ideal as they are too noisy for double integration to produce reliable position estimates, even when orientation is perfectly known. Low-cost gyros have low error with appropriate filtering. Drift can be overcome if they are reset based on identified events (e.g., stance phase of walking). Results in [25] [49] have demonstrated high accuracy in reconstructing walking and upper body motions with such resets and when

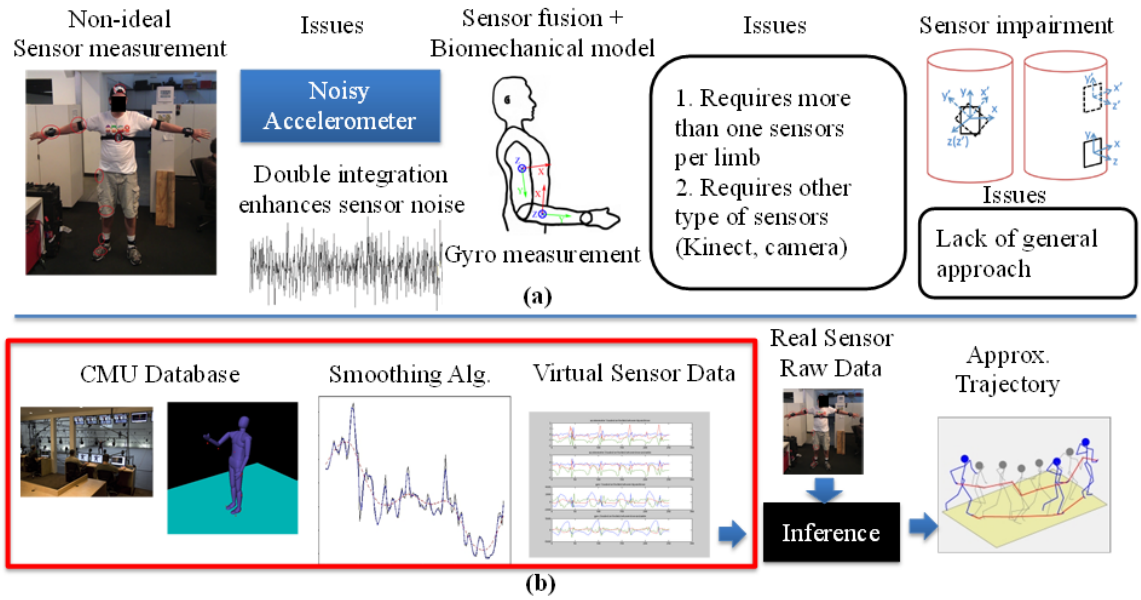


Figure 3.1: Illustration of the existing issues (a) and the proposed framework (b).

each limb segment is instrumented. But, except in laboratory experiments, to instrument one sensor per limb segment is infeasible in most applications. Hence, our challenge becomes the reconstruction of human motion characteristics with only one inertial sensor on each limb.

In addition to measurement noise, sensor misplacement problems, including misorientation and displacement, are common in at-home motion tracking. [50] proposed an orientation calibration method based on opportunistic repetitive motions without the requirement of predefined motions. A general approach to deal with various sensor impairments has not yet been investigated.

To overcome these problems, a model and algorithms must be constructed from datasets that reflect a broad set of impairments and for each of the motions of interest. Unfortunately, such a large amount of data collection is prohibitively costly. Therefore, as shown in Fig. 3.1(b), we propose a virtual inertial measurements platform to convert the currently available camera motion database (e.g., CMU MoCap [51]) into a noiseless and error-free inertial measurements database.

The main contributions of this research are as follows. An algorithm for converting camera based motion databases into inertial measurements (both accelerometers and gyroscopes) was developed. To verify the effectiveness of this method, a KINECT based test-bed was constructed. On this test-bed platform, synchronized camera data and inertial sensor data were collected simultaneously. Comparing the virtual inertial measurements estimated from our algorithm with the real inertial sensor measurements, the effectiveness of our algorithm was demonstrated.

II Inertial Sensor Measurement Modeling

To realize the virtual sensor experiments, we build a motion sensor measurement model. In this model, we derive the motion information from an existing motion database, develop appropriate processing algorithms for the motion data, build the observation model to map the motion data to sensor measurements, and then derive the sensor measurements without measurement noise. Some details follow.

II.1 Camera Motion Database

We employ the CMU motion capture database [51]. It contains 2605 different motion clips of full body MoCap data performed by a total of 144 subjects. The subjects were asked to wear 41 markers (Fig. 3.2) and perform a wide varieties of activities under the VICON camera system. The Vicon motion capture system consists of 12 infrared MX-40 cameras, each of which is capable of recording at 120 Hz with images of 4 megapixel resolution. Motions are captured in a working volume of approximately $3m \times 8m$. Each marker's three dimensional position and rotation information with respect to its parent node were provided in the database.

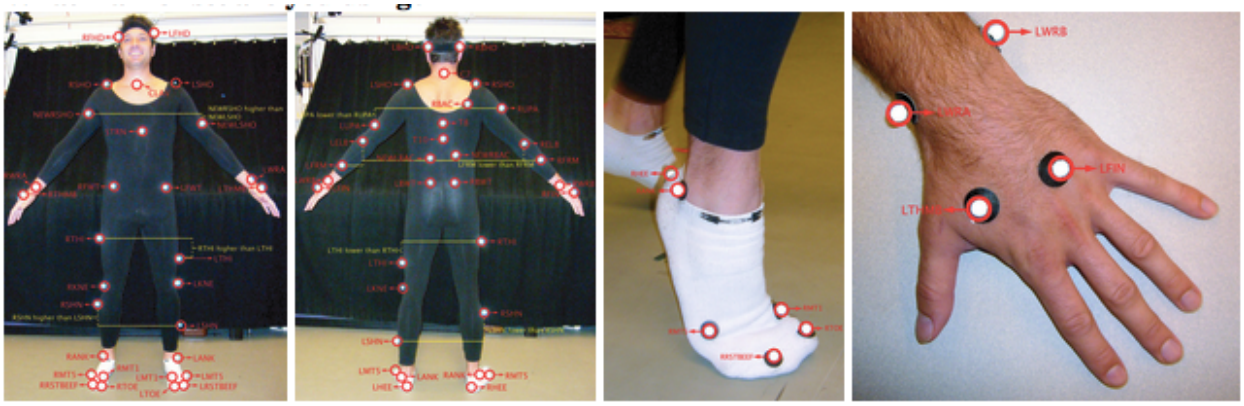


Figure 3.2: CMU motion capture marker placement

II.2 Inertial Sensor Observation Model

II.2.1 Gyroscope measurement model

Gyroscopes measure angular velocity on three axes. [52] shows that angular velocity can be captured by taking the time derivative of the rotation matrix (Equation 3.1). $C(t)$ is the rotation matrix (Equation 3.3) and $\Omega(t)$ contains the gyroscope measurements (Equation 3.2). Projections of the parent limbs' angular velocity onto the local reference frame are added to produce measurements with multiple moving segments.

$$\dot{C}(t) = C(t)\Omega(t) \tag{3.1}$$

$$\Omega(t) = \begin{pmatrix} 0 & -\omega_{bz}(t) & \omega_{by}(t) & 0 \\ \omega_{bz}(t) & 0 & -\omega_{bx}(t) & 0 \\ -\omega_{by} & \omega_{bx}(t) & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.2}$$

$$C(t) = R_z(t)R_y(t)R_x(t) \tag{3.3}$$

$$R_x(\alpha(t)) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha(t)) & -sin(\alpha(t)) & 0 \\ 0 & sin(\alpha(t)) & cos(\alpha(t)) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_y(\beta(t)) = \begin{pmatrix} cos(\beta(t)) & 0 & sin(\beta(t)) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\beta(t)) & 0 & cos(\beta(t)) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$R_z(\gamma(t)) = \begin{pmatrix} cos(\gamma(t)) & -sin(\gamma(t)) & 0 & 0 \\ sin(\gamma(t)) & cos(\gamma(t)) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

II.2.2 Accelerometer Measurement Model

The accelerometer signal consists of linear acceleration and gravity. Linear acceleration can be achieved by the second order differentiation of the position while gravity is a constant in one axis. Adding these two components together, we can then project it back into the local frame using gyroscope orientation information.

II.2.3 Data Smoothing

The basic problem with this dataset along with most other camera data is inaccuracy on the cm scale (e.g., the KINECT has similar resolution issues). With rapid sampling of 120Hz, such random position errors produce large apparent accelerations with double differentiation.

Thus, a smoothing technique is required to compensate the large gain on high frequency caused by the double-differentiation. We have identified the locally weighted scatterplot

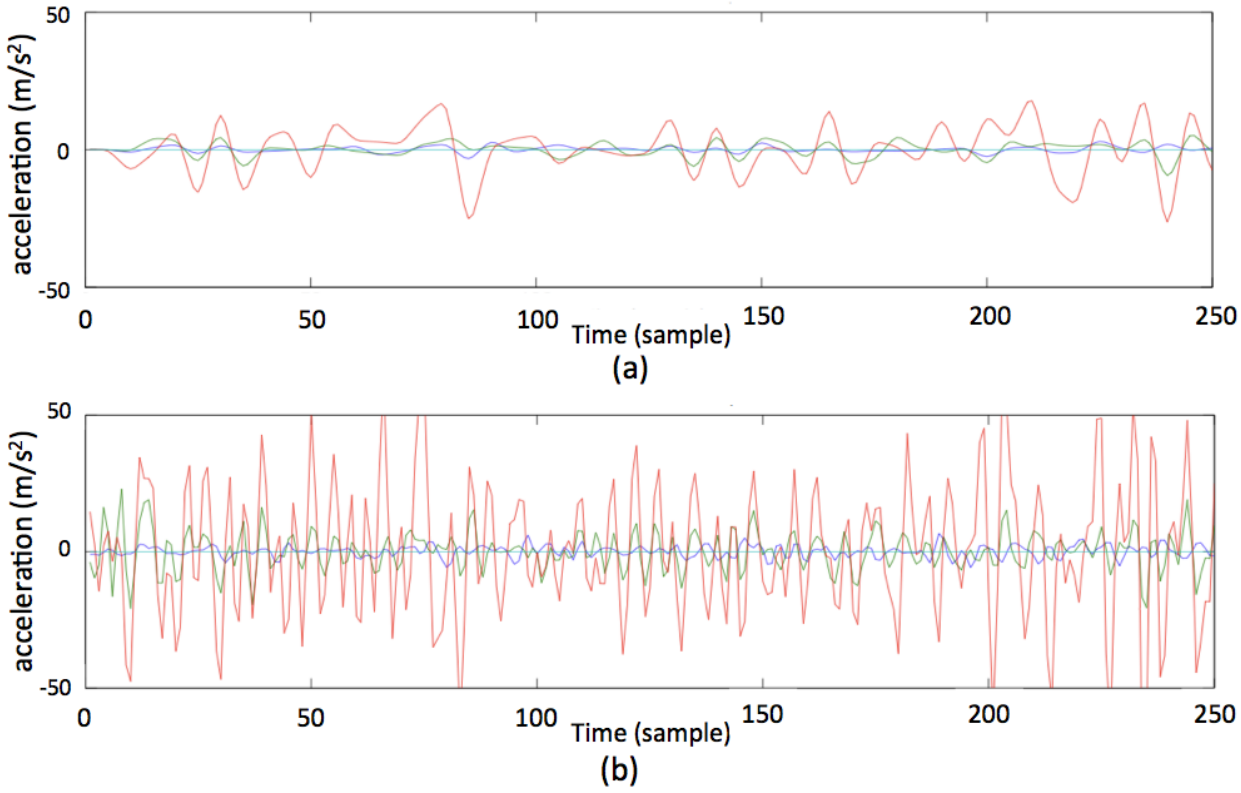


Figure 3.3: Effect of LOWESS function. Double differentiation (virtual accelerator measurement) of (a) smoothed and (b) unsmoothed raw camera data.

smoothing (LOWESS) [53] as a suitable smoothing function for the CMU MoCap dataset through a test-bed platform (more details in the later section) we developed. Instead of specifying a function to fit a model to all of the data in the sample, LOWESS has the advantage that it only requires very few arguments (i.e., a smoothing parameter value and the degree of the local polynomial.) Moreover, LOWESS is well-suited for modeling complex processes for which no theoretical models exist. These advantages combined with the simplicity of the method, make LOWESS a suitable smoothing algorithm for our application. Fig. 3.3 indicates that the LOWESS smoothing procedure attenuates the noise in position data and yields better results for double differentiation.

III Experiments and Results

In order to develop the proposed virtual sensor algorithm, we need both the video and inertial data for validation purposes. Since the CMU motion data base contains only video data, we developed a test-bed platform to collect synchronized motion data. We verified our virtual sensor algorithm over the synchronized motion data and then applied the algorithm to the CMU MoCap database.

III.1 A Test-bed Platform for Algorithm Verification

We integrated the Microsoft KINECT (Fig. 3.4) and inertial sensors to create a test-bed platform for algorithm validation. KINECT is an affordable camera system that can provide the position and orientation of 21 joints on the human body. Compared with the Vicon system, KINECT has similar accuracy in position and acceptable accuracy in orientation. This is enough for the purpose of algorithm development.

We collected synchronized motion data with KINECT and inertial sensors. We then applied various smoothing algorithms to KINECT data and derived the virtual inertial measurements. The resulting virtual inertial measurement was compared with the actual measurements of the inertial sensors. With this test-bed platform, we were able to validate the virtual sensor algorithm and search for a suitable smoothing function.

III.2 Collecting Synchronized Data using KINECT and Inertial Sensors

To collect synchronized motion data, we attached inertial sensors on the subject's upper limb and asked the subject to perform several predefined motions in front of the KINECT. Fig. 3.5 shows the experimental setup. Fig. 3.6 and Fig. 3.7 show the examples of actual and virtual inertial measurements generated from this test-bed platform. In the example, the subject attached a sensor on his right wrist, and drew five circles in the air with pauses. The smoothing function applied in this example is LOWESS. The result shows that both

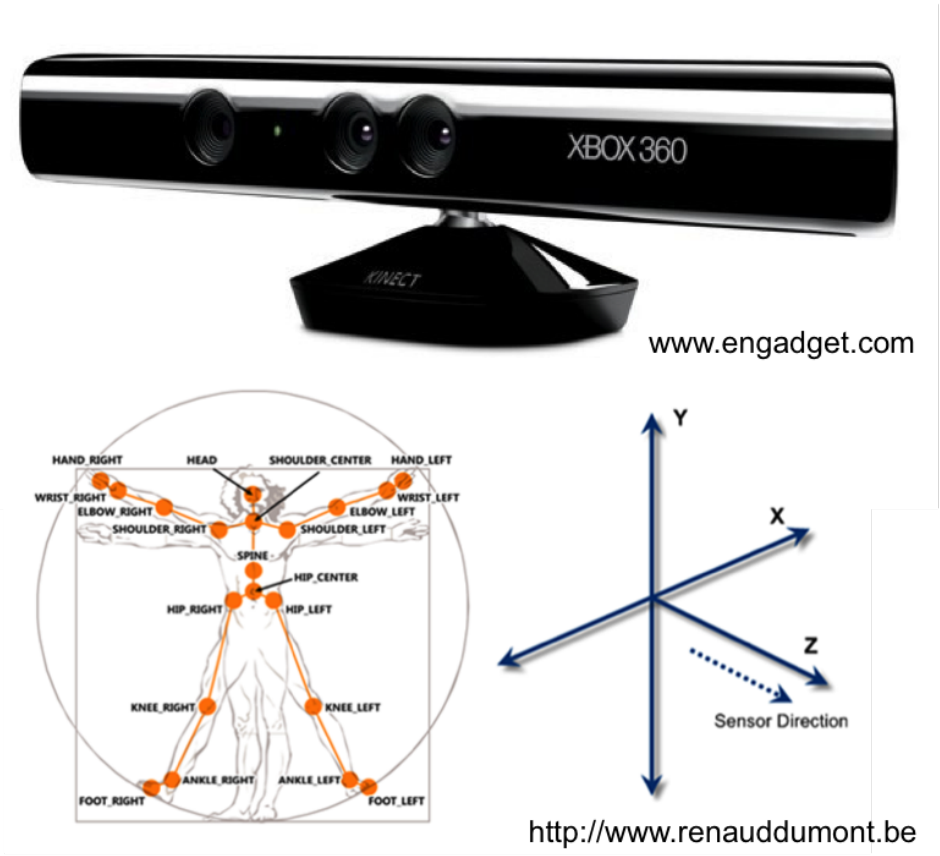


Figure 3.4: Microsoft KINECT controller

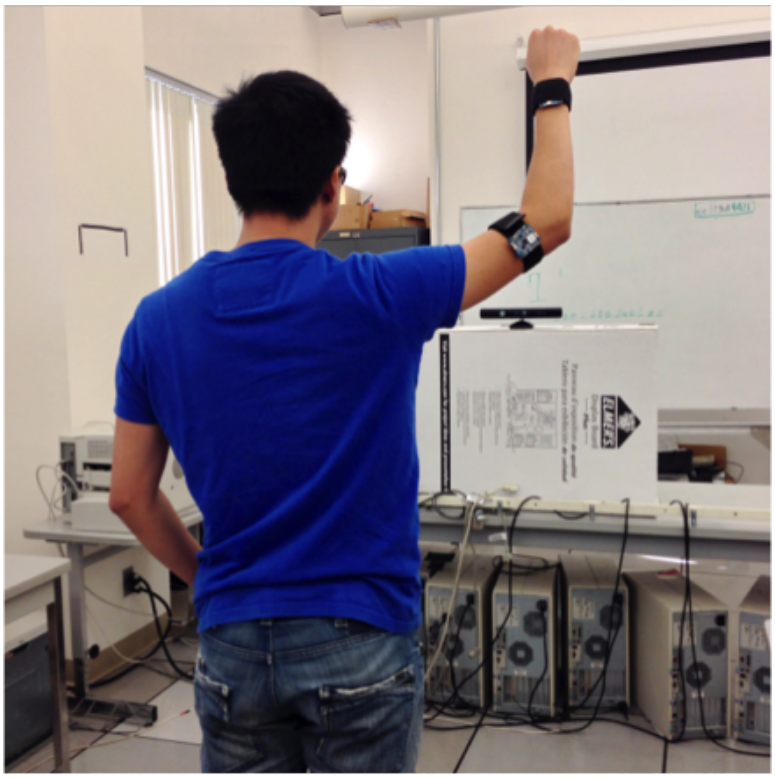


Figure 3.5: Experimental setup of the test-bed platform.

the virtual gyroscope and accelerometer measurements are not satisfactory without applying the smoothing function on the camera data. However, with the smoothing function applied, there is a high similarity between the virtual and actual acceleration. The virtual and actual gyroscope measurements have less agreement, although they share the same periodicity. The lesser similarity is possibly due to the software and hardware limitation of the current version of KINECT in detecting the orientation; we expect a better performance in the next version of KINECT.

III.3 Further Validation with the Vicon System

We've verified our algorithm via the Kinect test-bed platform. However, the Kinect and Vicon camera system have different noise profiles. As as result, we need to further verify the proposed algorithm with the Vicon system. We conducted experiments in the Gait and

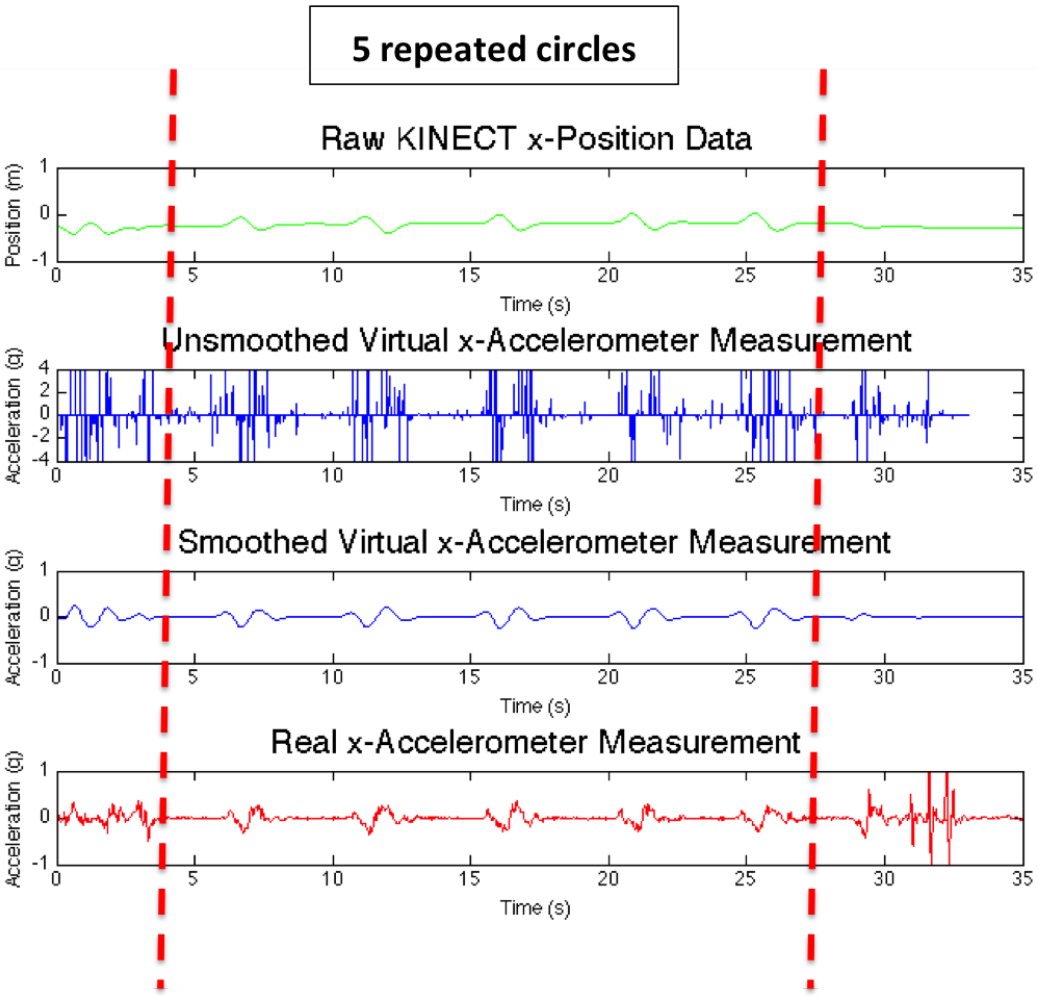


Figure 3.6: Result of the virtual accelerometer measurement from the test-bed platform.

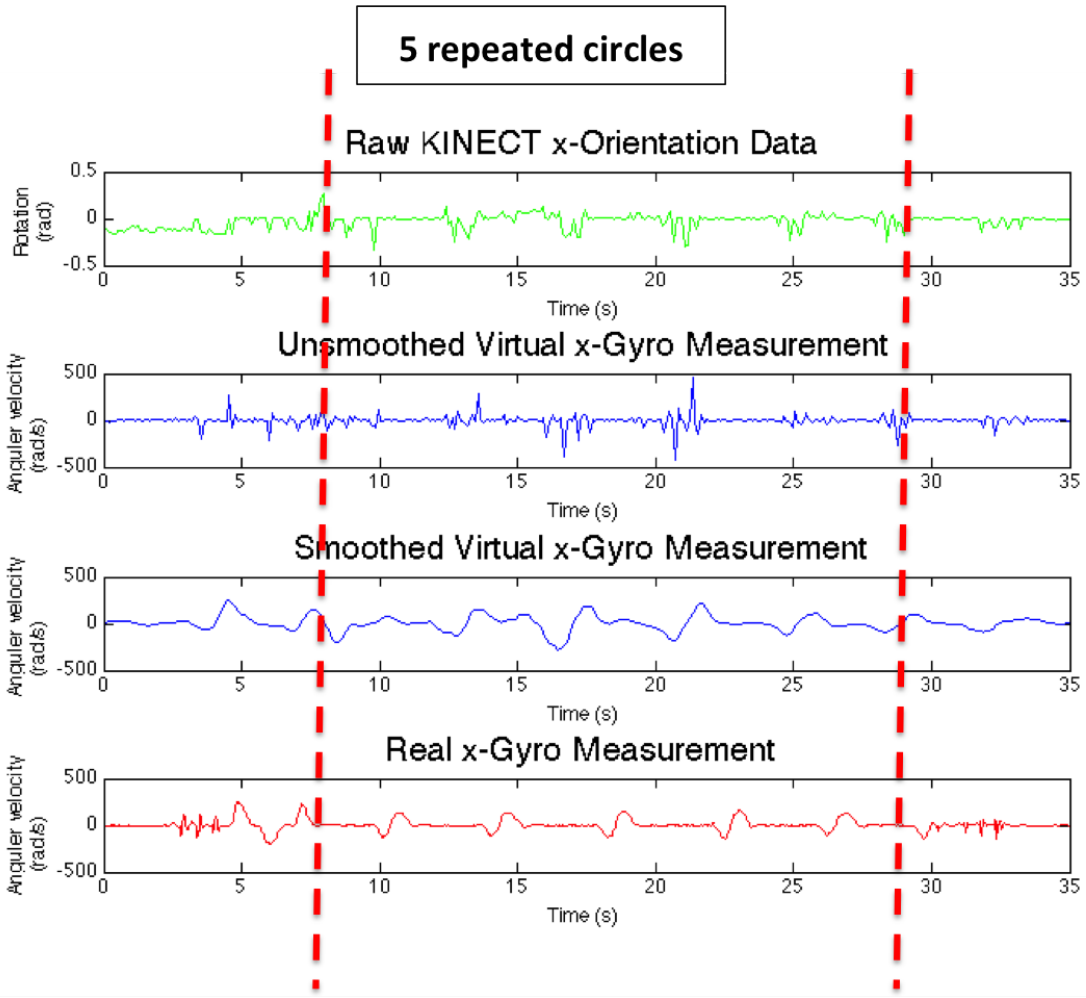


Figure 3.7: Result of the virtual gyroscope measurement from the test-bed platform.

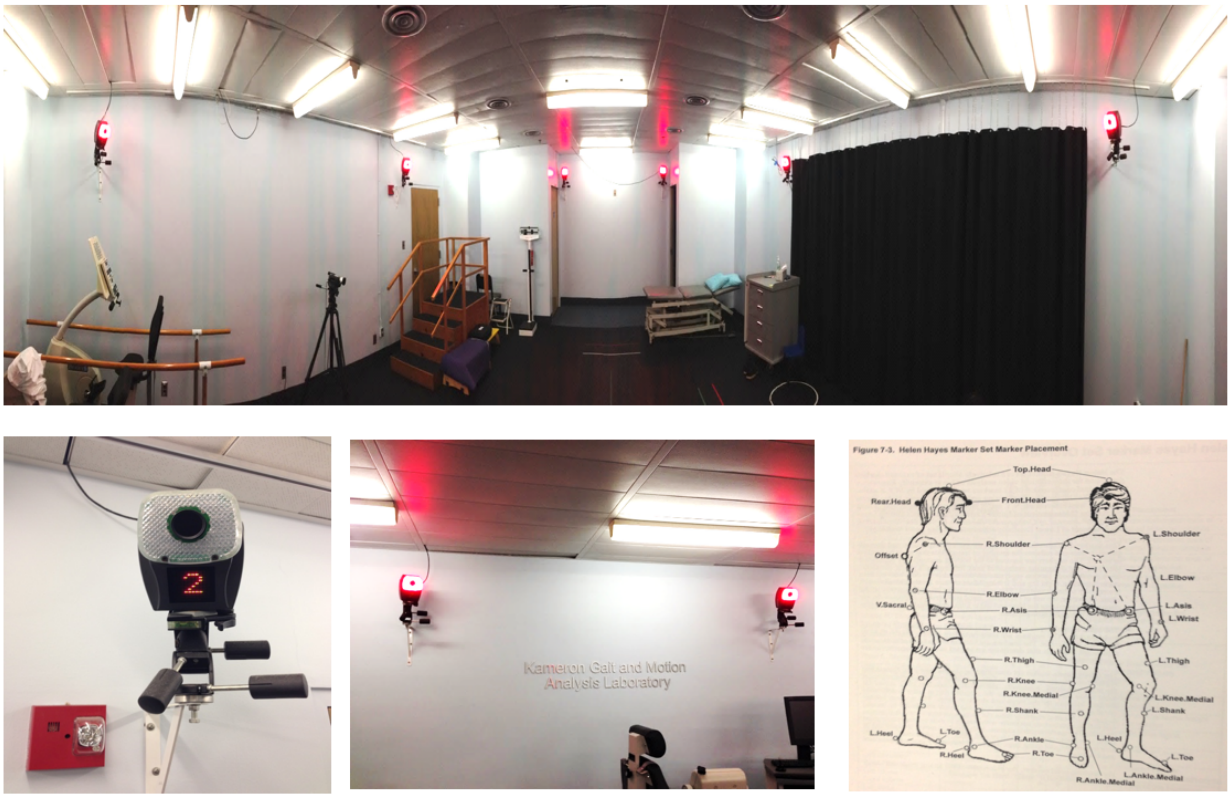


Figure 3.8: Gait and motion analysis lab

Motion Analysis Lab at the UCLA Rehab center (Fig.3.8). Our plan was to collect synchronized motion data from both the body-mounted inertial sensors and the Vicon system, and then compare those two measurements. We attached IMU sensor on both feet of the subject and asked the subject to perform several pre-defined movements while the Vicon system was recording. We applied the proposed algorithm to the Vicon system's data and converted them to a virtual inertial measurement. Results showed that the virtual inertial measurement derived form the Vicon system matched the measurement from the inertial sensors very well. Fig. 3.9 shows one set of results of the experiment. The agreement of the IMU and Vicon data further confirms the validity of our proposed algorithm.

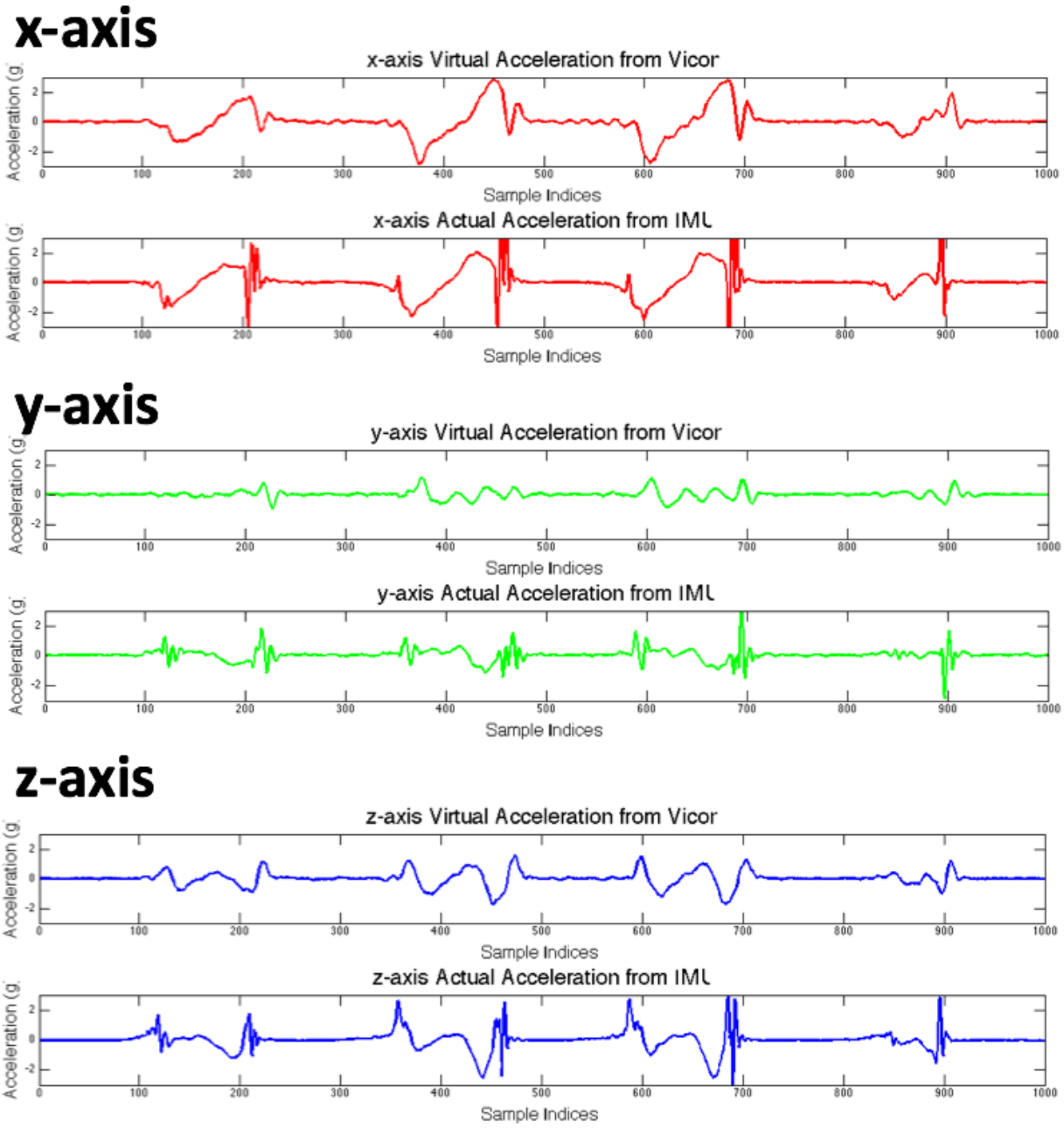


Figure 3.9: Results: Gait and Motion Analysis Lab

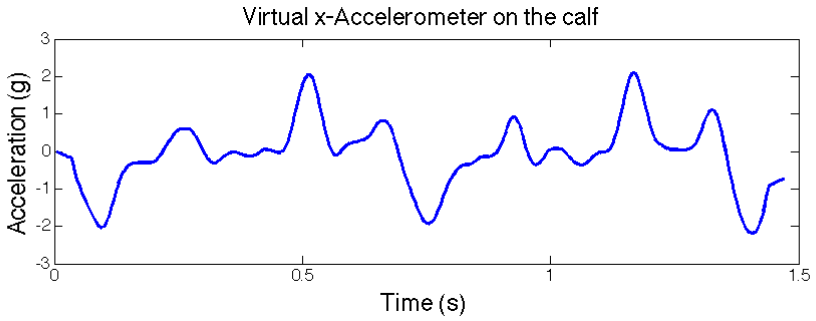


Figure 3.10: Virtual acceleration derived from CMU MoCap with LOWESS smoothing

III.4 Results of the Proposed Algorithm on the CMU Data

The experiment results from the test-bed platform support the validity of our virtual sensor algorithm and indicate that LOWESS is a good candidate for the smoothing function. We utilized Matlab's LOWESS smoothing function with smoothing parameter set to 0.02 and a first-order polynomial as the regression function. We then applied the same algorithm to the CMU MoCap data and derived the virtual acceleration. Fig. 3.10 and Fig. 3.11 show that both the virtual acceleration on the calf and thigh shows clear periodicity. These results further indicate that the proposed virtual sensor approach is feasible and that the test-bed platform is an efficient tool for searching suitable smoothing functions.

Based on what we learned form previous experiments, we cooperated with a group of EE180D students and developed a Matlab toolbox (Fig. 3.12). The toolbox can systematically generate VIM from the MoCap database. It can also simulate sensors at 30 locations with arbitrary orientation errors.

IV Conclusion

In this study, the virtual inertial sensor measurement was successfully derived from an existing MoCap database. A test-bed platform for algorithm verification was also developed. With the data processing and sensor measurement models developed in this research, we are

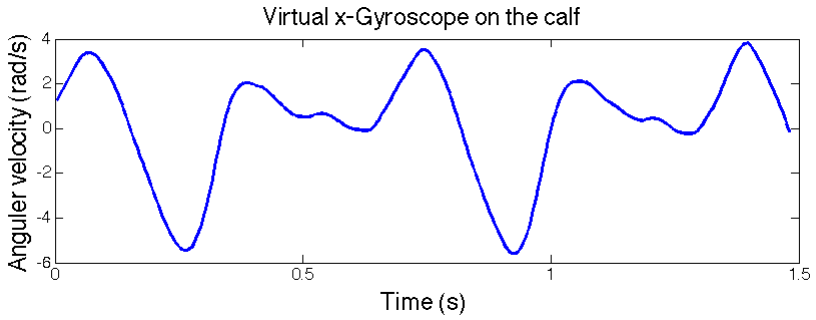


Figure 3.11: Virtual gyroscope measurements derived from CMU MoCap with LOWESS smoothing

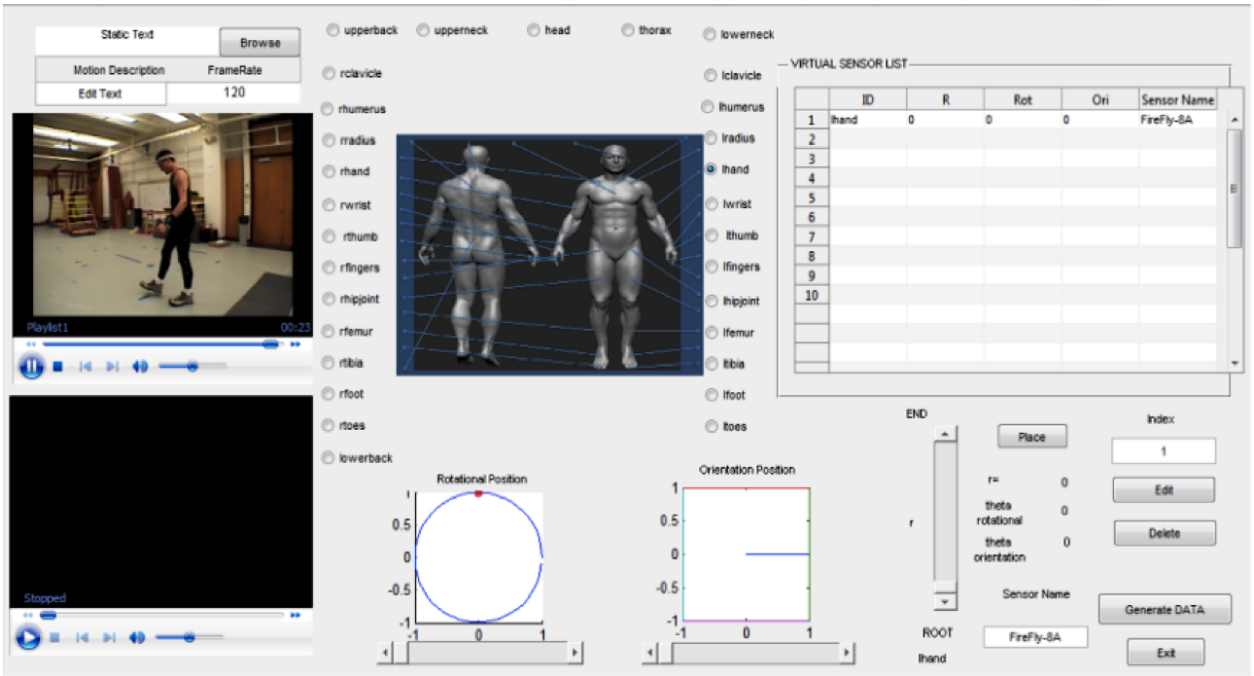


Figure 3.12: Matlab Toolbox: Virtual Inertial Measurement

able to generate various virtual sensor measurements from the motion database.

Future work includes further validation of the feasibility of the proposed approach with synchronized MoCap system and inertial sensor data, followed by systematic derivation of virtual sensor measurements from the CMU MoCap database. With this sensor simulation in hand, we can emulate sensor impairments as follows: orientation error can be represented by multiplying a rotation matrix; position error can be modeled by changing the coordinate values in the local frame. Combining with real data from human trials, we can further derive a measurement noise model and adjust the robust inference algorithms accordingly to enhance its performance when applied on real data.

Remarks Now, we know that the Kinect can provide low-cost ground truth. Can we leverage this powerful tool? In the next project, we use the Kinect's measurement as the ground truth to detect and compensate IMU misplacement errors.

CHAPTER 4

Opportunistic Calibration of Sensor Orientation using the Kinect and Inertial Measurement Unit Sensor Fusion

I Introduction

While wireless motion sensors, including accelerometers, gyroscopes and magnetometers, have been proven to be a low-cost, small-size, and highly-mobile option [54] [55] for human motion tracking and activity classification, sensor misplacement is an important issue that needs to be taken into account when deploying such systems in the real world. This placement error will degrade performance in motion classification and motion reconstruction. However, most research assumes that sensor placements are well-defined and fixed over time [56]. These assumptions are impractical in many medical settings, since patients and clinicians may not always follow instructions to place sensors correctly, and the sensor position may also change over time due to attachment issues.

Attempts to solve sensor misplacement issues can be categorized into two approaches: (1) finding orientation-invariant features [57] [58] [59] such as power spectral density or the Fourier transform, and (2) calibration through a series of pre-defined gestures or movements [60]. While the first approach could be successful for classification problems, they cannot provide detailed motion inference such as motion tracking or motion reconstruction.

The second approach might have limited application in the medical context due to the fact that it requires certain problem-specific calibration motions to be performed. In reality, we cannot rely on patients to perform such calibration motions, especially when the calibration is complicated. Wu et al. [61] proposed an orientation calibration method based on opportunistic repetitive motions without the requirement of pre-defined movements. However, this approach requires a training process to collect repetitive motion signatures, and hence it is limited to the lower body. To address the above-mentioned deficiencies, we propose a novel calibration process based on sensor fusion using the Microsoft Kinect and inertial measurement unit (IMU) sensors. The goal of this study is to provide reliable motion data in real-time, without the requirement of calibration activities, training processes and careful placement of the wearable sensors.

The remainder of this chapter is organized as follows: in Section II, we will provide background information on acquiring motion inference using IMU sensors and the Kinect. Different types of sensor misplacement will also be discussed. In Section III, the system architecture and each functional block will be presented. Finally, system verification and an example that demonstrates orientation calibration for trajectory reconstruction will be explained in Section IV.

II Technical Background

II.1 Motion Inference with IMU Sensors

Acquiring motion inference from IMUs has been of interest in the medical community for many years. Many approaches [29] [30] exploit features from IMU data to achieve motion classification. Others focus on motion tracking [62] [63] and utilize various techniques to extract position and rotation information. While high-end IMUs can provide reliable measurement, they are not suitable for large deployments due to costs. On the other hand, low cost accelerometers are non-ideal as they are too noisy for double integration to produce

reliable position estimates, even when orientation is perfectly known. Low cost gyroscopes have low error with appropriate filtering. Drift can be overcome if they are reset based on identified events (e.g., stance phase of walking). In addition to measurement noise, sensor misplacement problems are common in at-home motion tracking. In this research, we used the MUP-9250 motion sensor from InvenSense to explore sensor misplacement issues. The sensor board is Bluetooth enabled and can provide 9-axis inertial data, which includes 3-axis accelerometer, gyroscope and magnetometer.

II.2 Motion Tracking with the Kinect

The Kinect 2.0 is a motion-tracking device developed by Microsoft. It can track up to 6 full skeletons in the view of its camera. For each skeleton, 24 joints are defined and the positions and orientations can be tracked. The Kinect is also able to track the orientation of each joint and output rotation quaternions with respect to the earth frame. In addition to tracking joints, the Kinect is able to distinguish hand gestures in the following categories: inferred (not tracked), closed/open hand, lasso, and unknown. The Kinect uses the orientation of the thumb to help it track the wrist orientation. When the hand is open, a green circle is drawn around the hand as illustrated in Figure 4.1. When a joint is inferred, the color of the joint becomes yellow and the corresponding bone is drawn by a gray line instead of a bold green line. As shown in Figure 4.1, the hands are open and the right foot is obscured.

Since the Kinect's tracking algorithm has been trained with a large amount of data, it provides highly accurate motion inference that can be considered as the ground truth [64]. However, the subject must be in front of the Kinect at all times due to the nature of its camera-based motion tracking. In addition, the joints positions will contain spontaneous jittering due to noise and inferred tracking states. Hence, inferred joints are less reliable than visible joints. When using the Kinect's data as the ground truth, appropriate smoothing and filtering should be applied. In this research, we used the Kinect's position and orientation data to derive virtual acceleration as the ground truth. We also relied on joint's tracking

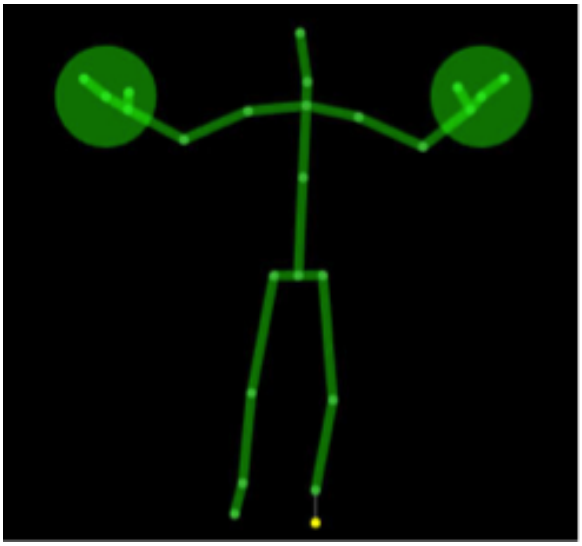


Figure 4.1: An example of skeleton tracking.

states and hand gestures to select calibration data.

II.3 Sensor Misplacement

Although misplacement has different forms, [61] shows that all misplacement within a limb can be decomposed into the 3 cases shown in Figure 4.2, if we model a limb as a thin cylinder and assume a sensor is always placed such that the x-y plane is firmly attached to the limb.

Misorientation is defined as when a sensor is placed at the correct position but with some rotation around the z-axis (Figure 4.2(a)). In this case, signals at the z-axis are invariant. Therefore, a rotation matrix in the x-y plane is sufficient to model this distortion. In a rotational displacement case (Figure 4.2(b)), since the limb is modeled as a thin cylinder, the displacement between the correctly placed sensor and the incorrectly placed sensor is negligible. As a result, signals for the x-axis and z-axis can be modeled by a rotation transformation, and signals for the y-axis are invariant. In the case of linear displacement (Figure 4.2(c)), the orientation of the sensor is unchanged and only y-axis translation exists. In this research, we aim to address misplacement with a combination of the first two cases.

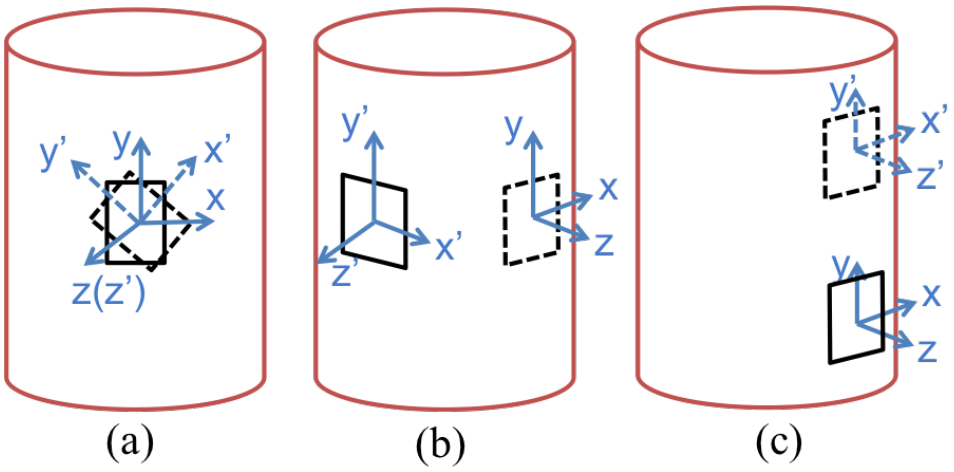


Figure 4.2: Definition of sensor misplacement. (a) Misorientation (b) Rotational Displacement (c) Linear Displacement

II.4 Kinect-IMU Calibrator Application

We developed the Kinect-IMU Calibrator application to detect and compensate for sensor misplacement. This application is capable of recording and visualizing motion data from both the Kinect and IMUs, identifying the calibration window, calculating correction quaternions, and applying correction to misplaced IMUs. Figure 4.3 shows a snapshot of the user interface.

III Method

III.1 System Architecture

Figure 4.4 shows the overall architecture of our system. The system receives real-time inertial data from body-mounted IMUs with some degree of orientation error. This orientation error is due to the compliance issue that the subject did not place IMUs on the nominal position that was instructed. Thus, the measurement is distorted. Once the subject appears in front of the Kinect and the Kinect identifies the subject's skeleton with high confidence, the calibration process is triggered. The system then collects a short segment of joint position data and generates corresponding virtual inertial measurements for calibration. By

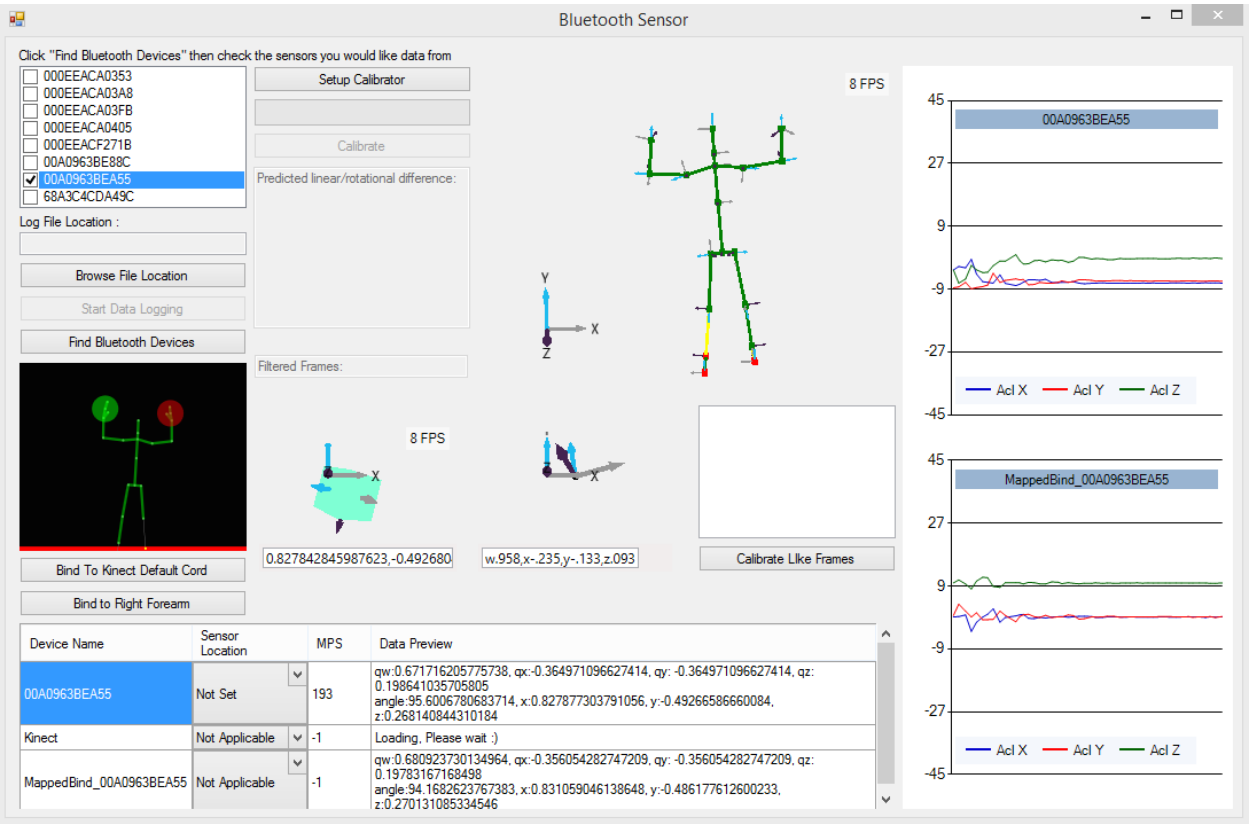


Figure 4.3: The user interface of the Kinect-IMU Calibrator.

comparing such virtual inertial measurement with the actual inertial measurements from the IMUs, misplacement can be detected and compensated behind the scenes. Details of each functional block will be discussed in the following sections.

III.2 Virtual IMU Algorithm

The Kinect reports position and orientation data for any joint, which serves as the ground truth for motion inference. In order to compare this ground truth with the actual inertial measurement from the IMUs, the position data must be converted to a comparable form such as virtual acceleration.

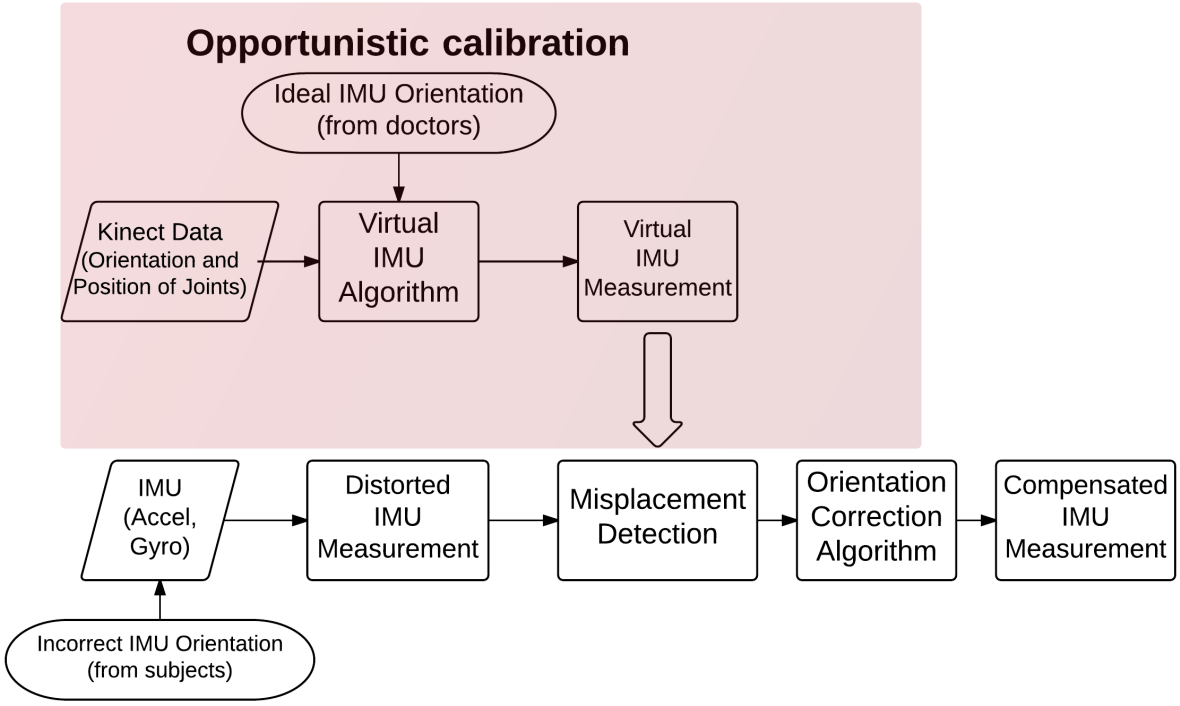


Figure 4.4: System architecture

III.2.1 Double Exponential Smoothing

Since the position data from the Kinect contains jittery noise, without proper data smoothing, such noise will be hugely amplified during the double differentiation in the next step. Double exponential smoothing is chosen because of its simplicity and low latency compared with other existing algorithms [65] [66]. Double exponential smoothing is accomplished by use of Eq. 4.1-3, where y_t represents the input raw data and S_t represents the smoothed data.

$$S_t = \alpha y_t + (1-\alpha)(S_{t-1} + b_{t-1}), 0 \leq \alpha \leq 1 \tag{4.1}$$

$$b_t = \beta(S_t - S_{t-1}) + (1+\beta)b_{t-1}, 0 \leq \beta \leq 1 \tag{4.2}$$

$$S_0 = y_0; b_0 = y_1 - y_0 \tag{4.3}$$

To determine the smoothing parameters $[\alpha, \beta]$ for our system, we collected a short segment of motion data using the IMU and Kinect. A data set that contains five repetitions of hand waving was recorded by the Kinect and correctly placed IMU at the wrist. The position data from the Kinect was first smoothed with different smoothing parameters and then converted to virtual acceleration data. The optimal $[\alpha, \beta]$ were determined by an exhaustive search in increments of 0.01 for each parameter for the combination that minimized the error between the correctly placed IMU's acceleration reading and the virtual acceleration in a least-square sense. As a result, we chose $[\alpha, \beta] = [0.35, 0.53]$ for our system.

III.2.2 Generating the Virtual Acceleration

The smoothed position data was differentiated twice with respect to time and yielded the virtual linear acceleration. In order to generate the virtual acceleration that is comparable to the IMU's measurement, a virtual gravity vector was added to the virtual linear acceleration.

Since the position data and virtual acceleration are both in the Kinect's reference frame, it is required to transform the measurement from the Kinect's frame to the frame of the correctly placed IMU. For each joint, the Kinect reports a rotational quaternion with the Kinect's frame as the reference. The correct sensor position is given and can be represented by a rotational quaternion $q_{Kinect\_joint}^{Correct\_IMU}$ with the a specific joint's frame as the reference. Thus, the virtual acceleration can be transformed to the frame of the correctly placed IMU:

$$q_{Kinect\_ref}^{Correct\_IMU} = q_{Kinect\_joint}^{Correct\_IMU} * q_{Kinect\_ref}^{Kinect\_joint}$$

$$acc_{Correct\_IMU} = q_{Kinect\_ref}^{Correct\_IMU} * acc_{Kinect\_ref} * {q_{Kinect\_ref}^{Kinect\_IMU}}^{-1}$$

III.3 Orientation Calibration

The orientation calibration process opportunistically compared the virtual acceleration with the actual IMU data, detects mis-orientation and provides compensation.

III.3.1 Identity Calibration Windows

When the Kinect detects a subject in its view, the calibration process is triggered. A buffer will cache 2 seconds of high quality synchronized data for both the actual and virtual acceleration. Since a patient would have the sensor in the same position for a relatively long period of time and the calibration only needs a small amount of data, we can be very selective about the data that we record to be used in the calibration algorithm. We implemented a cherry-picking filter for data quality assurance, which passes data only when the Kinect provides reliable measurements. To be considered as reliable data, the Kinect should have an unobstructed view of the joint of interest and identify the joint as being tracked. In some cases, the Kinect does not have a clear view of a limb and may report inferred position of the limb, which is not an accurate measurement of the actual position. If the Kinect is inferring or not tracking the position of the limb, then that data is not recorded in the buffer.

While the Kinect reports tracking state quite accurately, we noticed that in some cases, the tracking state might be misleading and a tracked joint does not necessarily yield high quality data. For example, the orientation of the wrist joint can only be accurately determined when the thumb is also visible. This provides an additional constraint to improve the quality of the recorded data. If the limb being tracked is the right wrist, we only record the data when the thumb is tracked and the hand is an open gesture. This ensures that the Kinect can more reliably determine the orientation of the wrist for calibration. Figures 4.5 and 4.6 show a comparison of when the Kinect can determine the current rotation of the wrist. In our experiments, we also observed that high acceleration tends to produce unreliable calibration results. Hence, if either the virtual or actual acceleration is above 15 m/s^2, we do not record that data into the buffer.

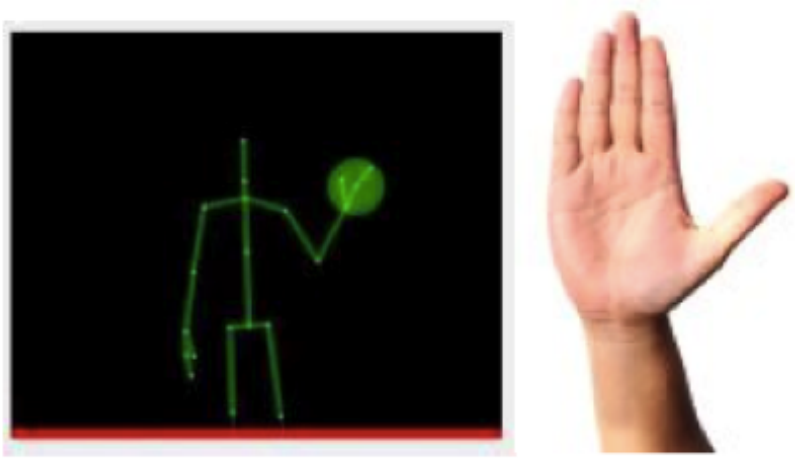


Figure 4.5: Open hand: the wrist orientation can be reliably determined.

III.3.2 Calibration and Compensation

Once the calibration buffer is full, the system is ready to determine the orientation offset. By examining the angle of acceleration vectors between the virtual and actual acceleration, the orientation error of IMU sensors can be identified and thus compensated. In order to find the rotation from the actual acceleration vector (IMU) to the virtual acceleration vector (Kinect), we first acquire the axis of rotation by taking the cross product of the two vectors. Then, the rotation angle is found by taking the inverse cosine of the dot product of the two normalized vectors. The axis of rotation and rotation angle gives the axis-angle representation of the rotation from the actual to virtual acceleration vectors. To convert from an axis-angle representation to rotation quaternions, the following equations are used [67], where (x, y, z) is the rotation axis, $angle$ is the rotation angle, and (q_w, q_x, q_y, q_z) is the equivalent rotation quaternion vector.

$$q_w = \cos(angle/2)$$

$$q_x = x * \sin(angle/2)$$

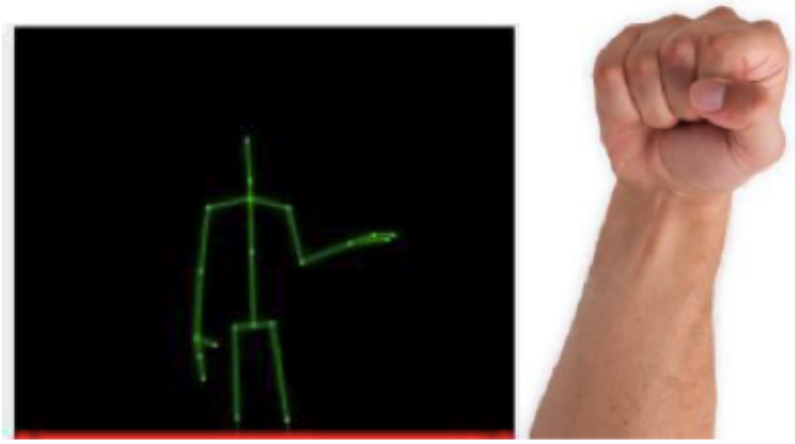


Figure 4.6: Closed hand: the wrist orientation cannot be reliably determined.

$$q_y = y * \sin(angle/2)$$

$$q_z = z * \sin(angle/2)$$

A typical 2-second calibration window contains about 60 such rotation quaternion vectors. After taking the average of these vectors, we get a single quaternion vector which represents the orientation error. Finally, this rotation quaternion vector can be used to calibrate sensor orientation either in real-time or offline. Given a rotation quaternion q that describes the rotation of the acceleration vector from a wrongly placed sensor (at orientation O1) to the correctly placed sensor (at orientation O1), we can calibrate the sensor orientation as follows: $[0, a_{O1}] = q * [0, a_{O2}] * q^{-1}$, where a_{O2} and a_{O2} are the acceleration measurements from the wrongly and correctly placed sensor, respectively.

IV Results

To verify our system, we designed the experiment setup as follows. We defined the correct sensor orientation O1 at the right wrist as shown in Figure 4.7. An IMU sensor was attached on the subject's right wrist with orientation O2, which might have been different than O1

due to placement error. We asked the subject to perform some activity without being tracked by the Kinect and then appear in front of the Kinect for a short amount of time until the Kinect finished the calibration process.

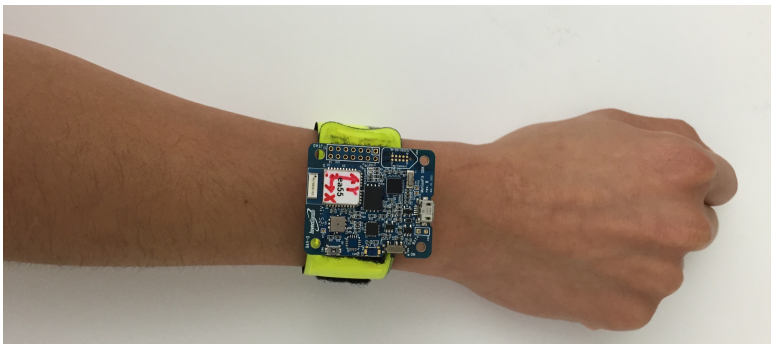


Figure 4.7: IMU placement with correct sensor orientation O1 at the wrist joint.

IV.1 Virtual Acceleration as the Ground Truth

It is important to verify that the virtual acceleration provided by the Kinect is reasonably accurate enough to simulate the acceleration measurement from a correctly placed IMU sensor on the wrist. We placed 1 sensor at the correct orientation O1, and compared the acceleration reading with the virtual acceleration from the Kinect. Figure 4.8 shows both the actual and virtual x-axis acceleration readings for hand waving motions. As expected, these two acceleration measurements matched very well. This indicates that the virtual acceleration from the Kinect can serve as a reasonable ground truth to simulate correctly placed IMU sensors.

Since the IMU sensor is in the correct orientation, we expect the calibration process to report the rotation quaternion that describes the rotation from $acc_{correct}$ to $acc_{virtual}$ to be close to an identity rotation quaternion, which is $[q_w, q_x, q_y, q_z] = [1, 0, 0, 0]$. We performed 2 trials of calibration, and the results are listed in Table 4.1. As expected, for both trials, the rotation quaternions were close to the identity rotation quaternion. Therefore, this indicates that the calibration works for the case where the sensor is approximately placed in the correct

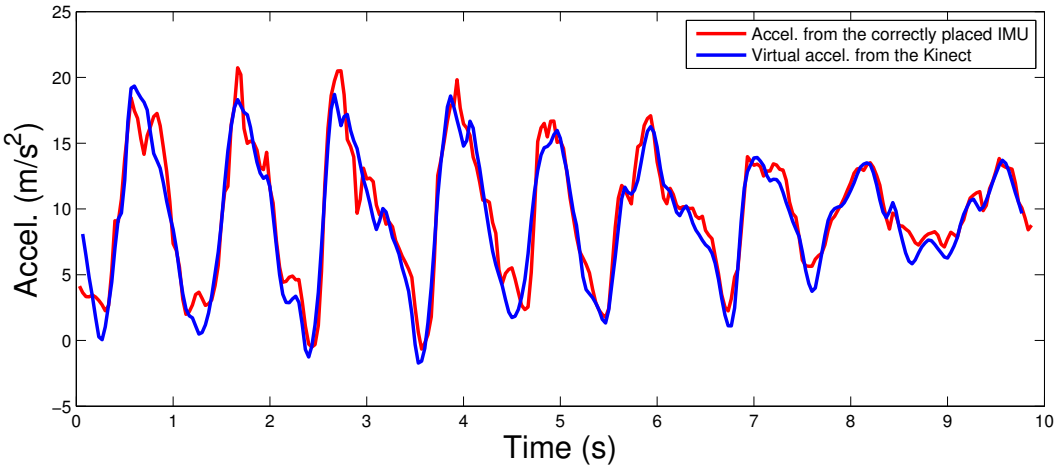


Figure 4.8: Comparison between the virtual and actual acceleration.

orientation.

Table 4.1: Calibration results for the correctly placed sensor

	q_w	q_x	q_y	q_z
Trial 1	0.992	0.004	0.050	0.110
Trial 2	0.994	0.032	0.049	0.086

IV.2 Calibration Results from Misplaced IMU and Virtual Acceleration: (Sensor Recovery Result)

IV.2.1 Compensated Misplaced IMU vs. Virtual Acceleration

The calibration data that describes the rotation between the raw IMU sensor data and the virtual acceleration data were in the form of rotation quaternions. If we apply such rotation quaternions to the raw IMU sensor data, we could expect that the rotated (rectified) IMU sensor data will match the virtual acceleration data.

To verify this, we placed a sensor at the subject's right wrist with some incorrect orientation. The subject first performed some motion activity and then appeared in front of the Kinect to trigger the calibration process. The application recorded both the raw IMU accel-

eration and virtual acceleration, and reported the calibration result in the form of a rotation quaternion. We applied the rotation to the raw IMU sensor acceleration and compared it to the Kinect's virtual acceleration data. We verified the compensation in two different cases: stationary activity and motion activity.

For the first case, the subject stood still so the data contains only gravity. As shown in Figure 4.9, the rotated acceleration data matched the Kinect's virtual acceleration data. For the second case, the subject perform repeating movements. As shown in Figure 4.10, the rotated acceleration data also matched the Kinect's virtual acceleration data. We had 3 subjects performe this calibration process for 5 trials, and we got similar results that the algorithm is able to compensate the misplaced IMU. Note that sensor orientations are arbitrary among subjects, and the orientations stayed the same during trials 1 to 5 for a single subject. Table 4.2 shows the rotation quaternions calculated from the calibration process for each subject and each trial, and the results are consistent among trials for a single subject. Therefore, we conclude that the calibration process produces reliable results.

IV.2.2 Rectified Misplaced IMU vs. Correctly Placed IMU

To further verify the accuracy of the calibration mechanism, we placed two sensors at the subject's right wrist with different orientations. One sensor was placed with a correct orientation O1, and the other was placed with an incorrect orientation O2. The sensor with orientation O1 provided a baseline for evaluation purposes. Figure 4.11 shows an example configuration of the two sensors. Ideally, when the calibration result is applied to the data from the incorrectly placed sensor, the compensated data should match the data from the correctly placed sensor.

Table 4.2: Calibration results (Rotation quaternions) for different subjects and trials.

	Subject 1	Subject 2	Subject 3
Trial 1	(0.8887, 0.2273, 0.2757, 0.2871)	(0.7829, 0.4863, 0.2182, 0.3206)	(0.6385, 0.7667, 0.0601, 0.0283)
Trial 2	(0.8939, 0.2328, 0.2888, 0.2513)	(0.7899, 0.4895, 0.2013, 0.3150)	(0.6391, 0.7646, 0.0793, 0.0249)
Trial 3	(0.8879, 0.2063, 0.2652, 0.3138)	(0.7898, 0.4861, 0.1992, 0.3164)	(0.6429, 0.7617, 0.0753, 0.0274)
Trial 4	(0.8990, 0.2078, 0.1427, 0.3580)	(0.7968, 0.4904, 0.1828, 0.3017)	(0.6479, 0.7556, 0.0840, 0.0457)
Trial 5	(0.8858, 0.2070, 0.2711, 0.3147)	(0.7978, 0.4955, 0.2387, 0.2466)	(0.6423, 0.7616, 0.0741, 0.0280)

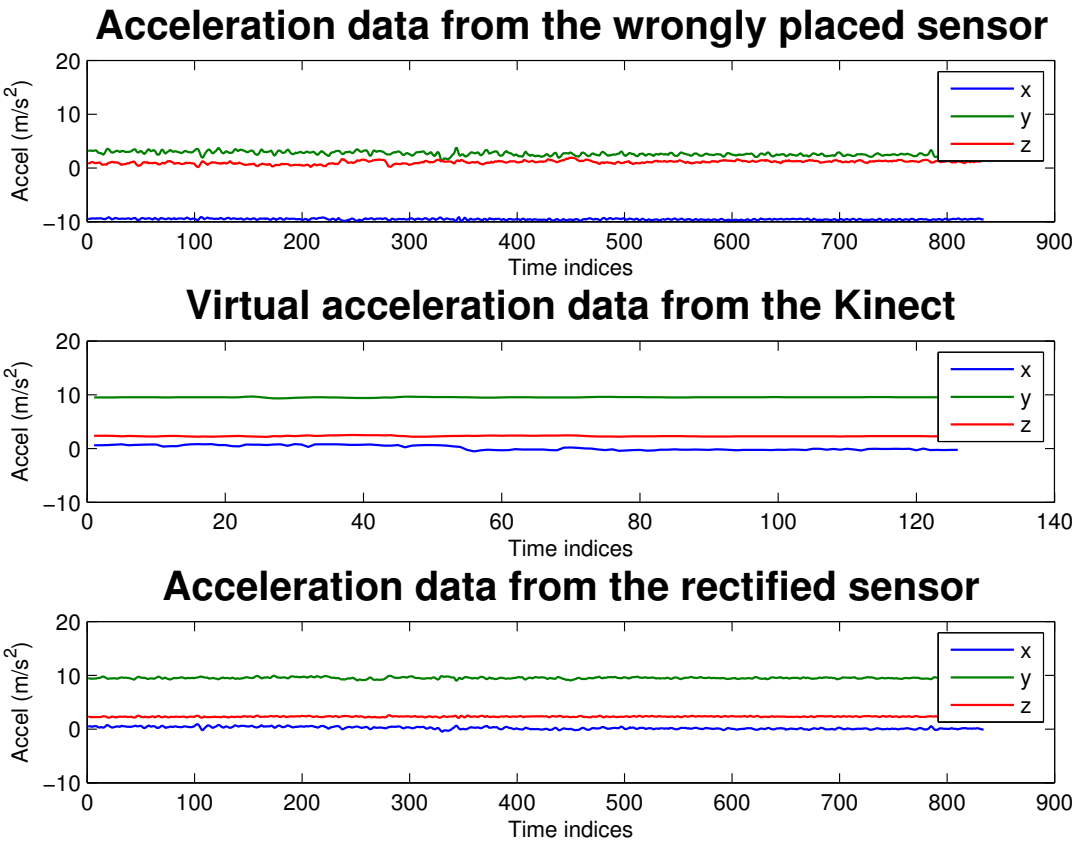


Figure 4.9: Comparison of the rectified and virtual acceleration data. (stationary)

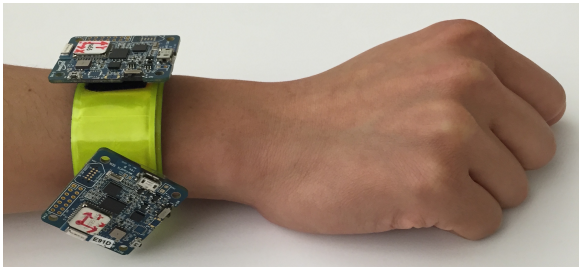


Figure 4.11: Image of the two-sensor experiment setup.

We attached two sensors on the subject's right wrist as described above, instructed the subject to perform some random movements, and collected data from both sensors at the same time. Then, without moving the two sensors, we used the application to obtain

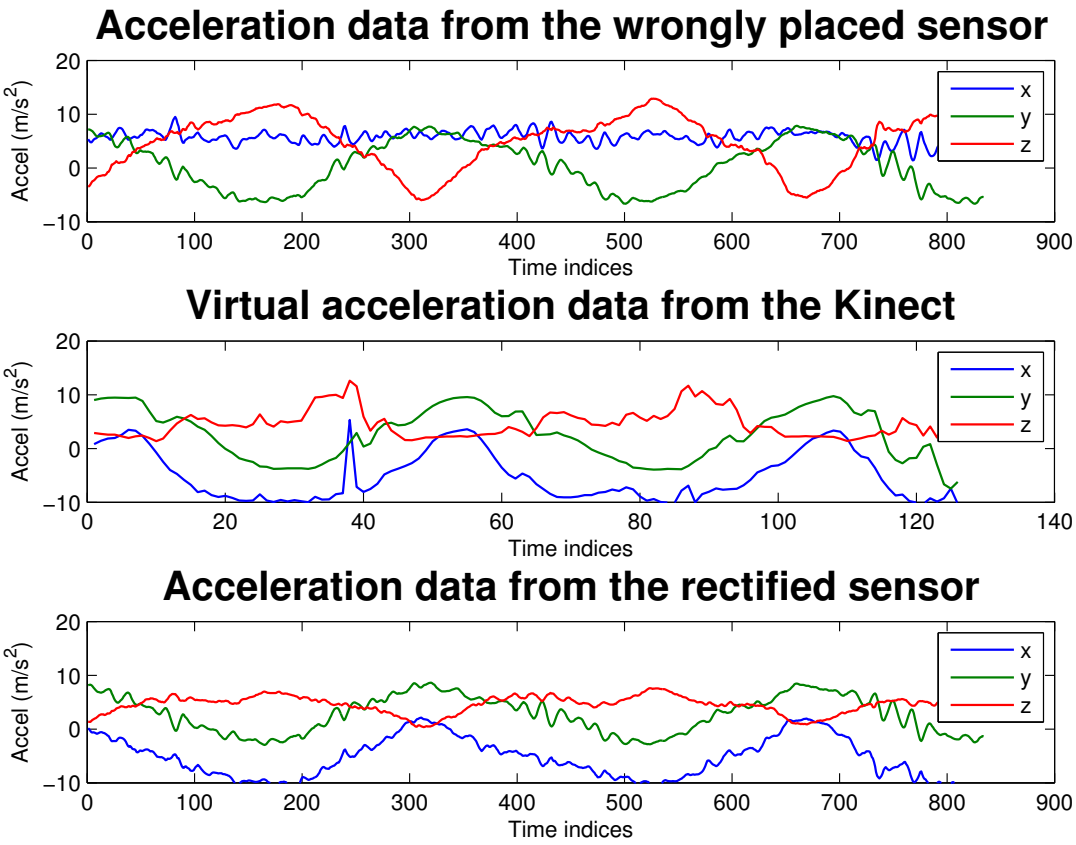


Figure 4.10: Comparison of the rectified and virtual acceleration data. (motion)

the calibration results in the form of a rotation quaternion. We then applied the rotation quaternion to the data from the incorrectly placed sensor and compared it with the data from the correctly placed sensor.

In one trial, we applied the rotation quaternion that we obtained from the calibration results to the incorrectly placed sensor accelerations and compared them with the other accelerations. Figures 4.12 shows comparisons among the incorrectly placed sensor accelerations, rectified sensor accelerations using the rotation quaternion, and the correctly placed sensor accelerations. As shown in these figures, the rotated data matched well with the correct data for all 3 axes. The results that were obtained from other trials that are not

shown were consistent with the results for the trial that are described here. Therefore, we conclude that the calibration results from the calibration process is accurate.

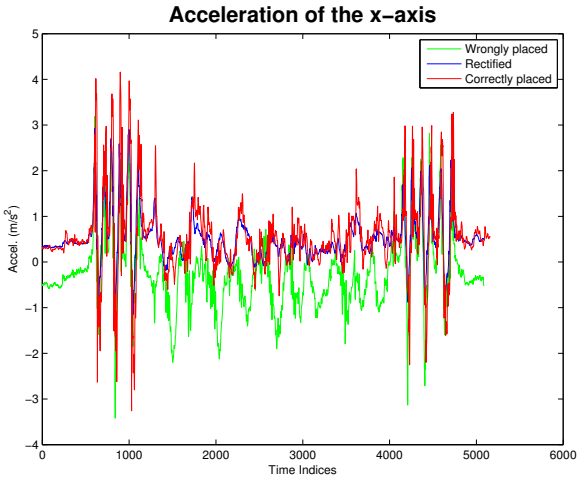
IV.3 Example Application: Orientation Compensation for Upper Body Trajectory Reconstruction

We designed an example to demonstrate an application of the proposed calibration method. The goal was to reconstruct trajectories of the wrist joint using a misplaced IMU. Trajectory reconstruction was achieved using the approaches described in [68] with zero-velocity update [69] for sensor drift compensation.

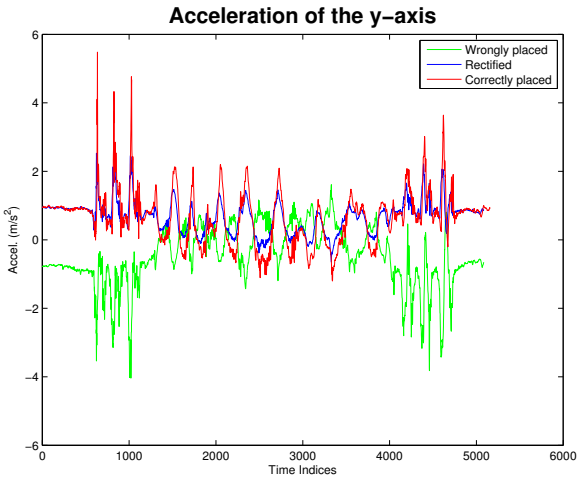
Two sensors were mounted at the subject's right wrist. One was attached at the correct position shown in Figure 4.11 to serve as a baseline for comparison, and the other one at an arbitrary orientation and rotational displacement. The subject was instructed to use his right wrist to trace a mark of a 30x30 cm square on a table. After the drawing was done, the subject walked into the Kinect's view and triggered the calibration process while the attachment of sensors remained unchanged. We then rectified the data of the misplaced sensor and reconstruct trajectories. Here we defined the positive directions of x, y, and z to be the right, front, and up directions of the subject, respectively. Figure 4.13 and 4.14 show the reconstructed trajectories for two different cases of misplacement. In both cases, the reconstructed trajectories of the rectified sensor matched fairly well with the correctly placed sensor.

V Conclusions

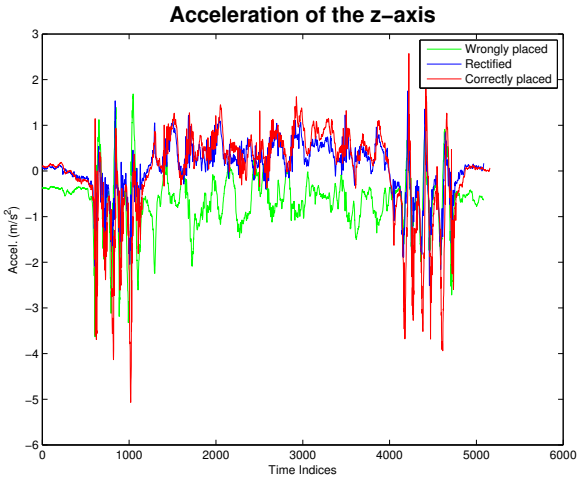
In this research, we demonstrate a system that can fuse the Kinect and IMU data to achieve opportunistic calibration of sensor orientation. We considered sensor misplacement cases with the combinations of misorientation and rotational displacement. We verified the validity of the proposed system by comparing acceleration measurements between rectified sensors



(a) x-axis



(b) y-axis



(c) z-axis

Figure 4.12: Comparison of the original, rectified and correct accelerations for the x-axis, y-axis, and z-axis.

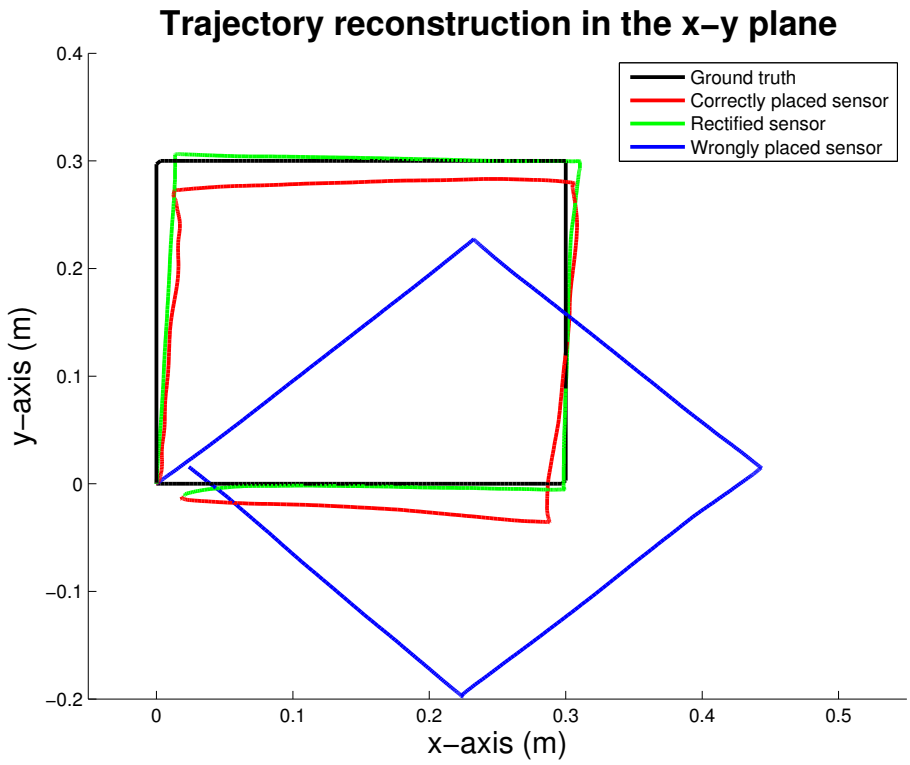


Figure 4.13: Trajectory reconstruction with sensor misplacement: 45 degrees misorientation around z.

and correctly placed sensors. We also showed an example of trajectory reconstruction with misplaced sensors. The results indicate that our system can detect and rectify misplaced sensors well.

Since there are no specific calibration postures or activities required during the calibration, this system is practical for deploying outside the lab environment. The outcomes of this research will facilitate ground-truth collection in the clinic, and also provide reliable motion inference for health monitoring in the community. Our study was however limited to healthy subjects. In the future, we plan to investigate performance with patients in both clinical and home settings.

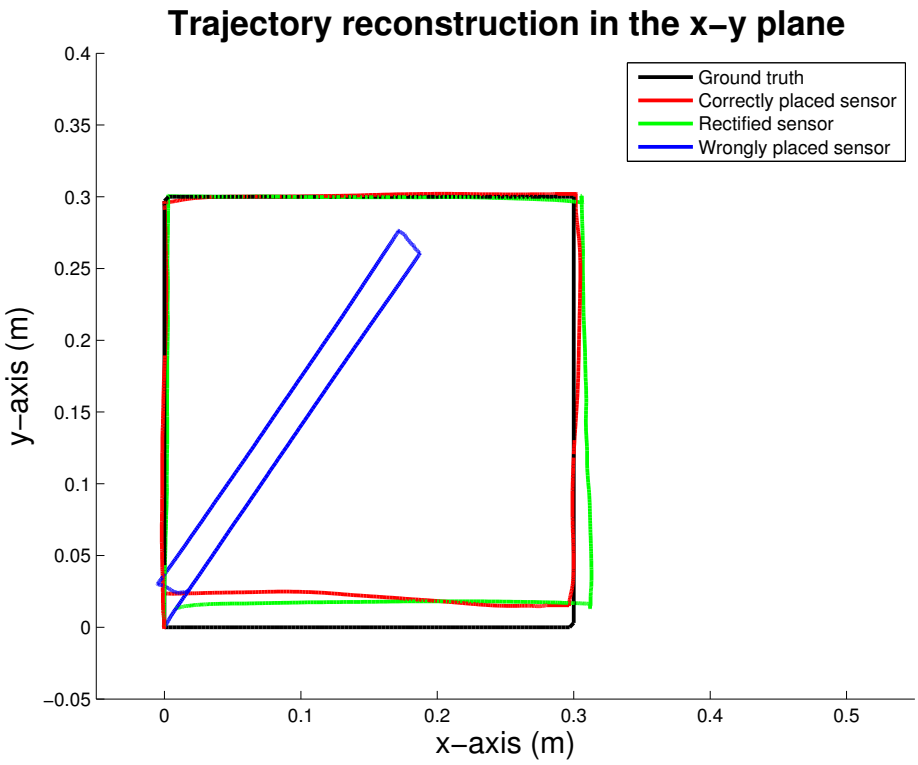


Figure 4.14: Trajectory reconstruction with sensor misplacement: 90 degrees rotational displacement around y and 45 degrees misorientation around z.

Remarks Earlier in this chapter, we mentioned that there are two types of sensor uncertainties: sensor misplacement and sensor drift. Here we've provided a solution for sensor misplacement, what can we do about sensor drift? We answered this question in Chapter 5, where we proposed a sensor fusion approach that deals with sensor drift for upper limb motion tracking.

CHAPTER 5

Robust Upper Limbs Motion Tracking using Sensor Fusion of the Leap Motion Controller and IMUs

I Introduction

Upper body motion tracking is especially a top priority in health care, since it provides crucial inference to assess the mobility of hands and digits with musculoskeletal and neural disorders [70].

Much research has been conducted to enable robust trajectory reconstruction for upper limbs, and two of the most common sensing technologies are: (1) Vision-based system and (2) MEMS inertial sensors. Vision-based systems include the Vicon system, Microsoft Kinect, and Leap Motion controller. In [71], the Vicon system was used to study upper limb kinematics for hemiparetic stroke. In [72], the Kinect camera was used to evaluate upper extremity reachable workspace. These devices provide accurate tracking results, but they either suffer from limited tracking range or are too expensive to be deployed outside of lab settings.

On the other hand, MEMS inertial sensors are lightweight, portable and can be deployed outside of the lab environment for remote motion tracking. However, without precautionary measures, sensor drift will degrade tracking results. In [55], orientations of joints were estimated using kinematic models and unscented Kalman filters under slow and fast motions. However, the results were limited to simple arm movements. In [54], a continuous-wavelet-transform based method was performed to analytically integrate accelerometer data to avoid

integration drifts when integrating numerically. However, subjects in that study only performed motions slowly, and some reconstructed patterns are only recognizable rather then accurate.

In this research, we use complementary filters to fuse trajectory estimates from the Leap Motion controller and IMUs to reconstruct upper limb trajectory. Our goal is to estimate the trajectory of the upper limbs at any given moment with high accuracy.

The remainder of the chapter is organized as follows: in Section II, we will provide background information on acquiring motion inference using IMU sensors and the Leap Motion controller. Sensor fusion using complementary filters will also be discussed. In Section III, the system architecture and each functional block will be presented. Finally, experiment design, verification and trajectory reconstruction will be explained in Section IV.

II Technical Background

II.1 Motion Inference with IMU Sensors

Acquiring motion inferences from IMUs has been of interest in the medical community for many years. Many approaches [29] [30] exploit features from IMU data to achieve motion classification. Others focus on motion tracking [62] [63] and utilize various techniques to extract position and rotation information. While high-end IMUs can provide reliable measurement, they are not suitable for large deployments due to costs. On the other hand, low-cost IMUs have scalability but they provide non-ideal measurements. For example, a low-cost gyroscope has floating bias and thus the orientation estimate will drift over time. A common approach to reconstruct a motion trajectory from an IMU is to remove gravity from the acceleration and doubly integrate the dynamic acceleration. Since the orientation estimate from a low-cost gyro is not accurate, the gravity component cannot be perfectly removed from the acceleration measurement. As a result, the residual of the gravity compo-

nent will be greatly amplified during the double integration process, and yields tremendous drift.

The zero-velocity update (ZUPT) is a commonly used technique to eliminate this drift [69]. However, this process cannot perfectly remove the velocity drift because there are uncertainties in detecting zero-velocity windows. As a result, the trajectory estimates from low-cost IMUs will still drift over time.

In this research, we use the InvenSense MUP-9250 motion sensor to calculate position estimation. The sensor board is Bluetooth enabled and can provide 9-axis inertial data, which includes 3-axis accelerometer, gyroscope and magnetometer.

II.2 Motion Tracking with the Leap Motion

The Leap Motion controller is a motion-tracking device developed by Leap Motion. It was designed as a gesture controller for computers. The device uses two monochromatic IR cameras and three IR LEDs to observe a roughly hemispherical area, to a distance of about 2.5 to 60 cm. It can track hands, fingers and finger-like tools, and report discrete positions, gestures and motion.

Since the Leap Motion controller's tracking algorithm has been trained with a large amount of data, it provides highly accurate motion inference that can be considered as the ground truth. In [73], the overall average accuracy of the Leap controller was shown to be 0.7 millimeters. Compared with the Microsoft Kinect, the Leap Motion controller is more suitable for our application since it provides more precise motion tracking for upper limbs.

However, the Leap Motion controller also has some limitations. Due to the nature of its camera-based tracking approach, the joint being tracked must be in the view of the device at all times. In addition, the joints positions will contain spontaneous jittering noise due to inferred tracking states. Hence, inferred joints are less reliable than visible joints. The device reports a score in the range of 0 to 1 to represent the confidence level of tracking

states for each frame.

In this research, we use the Leap Motion controller's position and orientation data to estimate trajectories of joints of upper limbs. We also rely on the confidence score of tracking states for sensor fusion.

II.3 Sensor Fusion and the Complementary Filter

In real-time motion tracking applications such as flight navigation or robotics orientation estimation, the complementary filter is often used for its simplicity and efficiency [74] [75]. It fuses multiple estimation measurements that have noise of complementary spectral characteristics [76].

For example, we can estimate the orientation of the sensor either using accelerometers and magnetometers, or integrating the gyros. However, the result would suffer from long-term drift when integrating the gyros, and instantaneous noise when using accelerometers and magnetometers to estimate orientations. A complementary filter can be used to filter the orientation estimation using accelerometers and magnetometers with a high-pass filter, and the orientation estimation using gyros with a low pass filter. We then sum up the two filtered signals to remove the corresponding noises and achieve a better estimate of orientation.

In this research, we apply the complementary filter to fuse trajectory estimates from the Leap controller and the IMU.

III Method

III.1 System Architecture

Figure 5.1 shows the overall architecture of our system. The system receives real-time inertial data from body-mounted IMUs and joint positions from the Leap Motion controller. We first derive trajectory estimates from the IMU data, and then perform a rigid transform to match

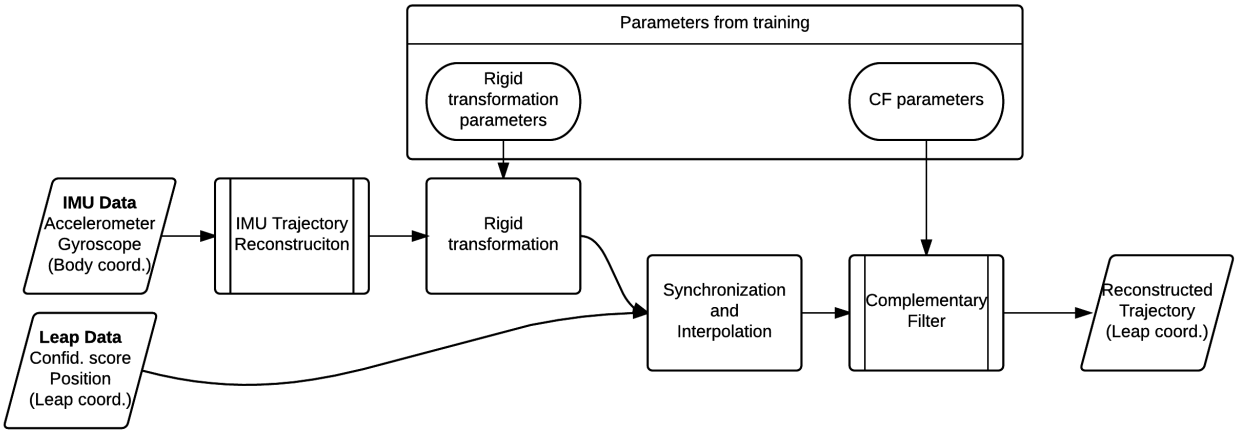


Figure 5.1: System architecture

the Leap's coordinate system. After signal synchronization, a complementary filter fuses the trajectory estimates from the IMU and Leap, and reconstructs robust trajectory estimates. Details of each functional block will be discussed in the following sections.

III.2 Hardware Setup

We designed a 40 x 30 x 12 (cm) box with a piece of transparent acrylic on the top (Figure 5.2) to hold the Leap Motion controller. The controller sits at the center of the bottom of the box and has clear view above the acrylic surface. There is a 20x20 (cm) ground truth square marked on the acrylic surface, and its center has been calibrated to be aligned with the Leap Motion controller's origin.

III.3 Signal Processing

III.3.1 IMU Trajectory Reconstruction

The IMU reports acceleration and angular velocity in the IMU's body frame. It also reports rotation quaternions that describe the rotation between the initial orientation and current orientation. Using this information, the acceleration measurement is first rotated back to the initial coordinate frame, and then the gravity component can be removed. We apply

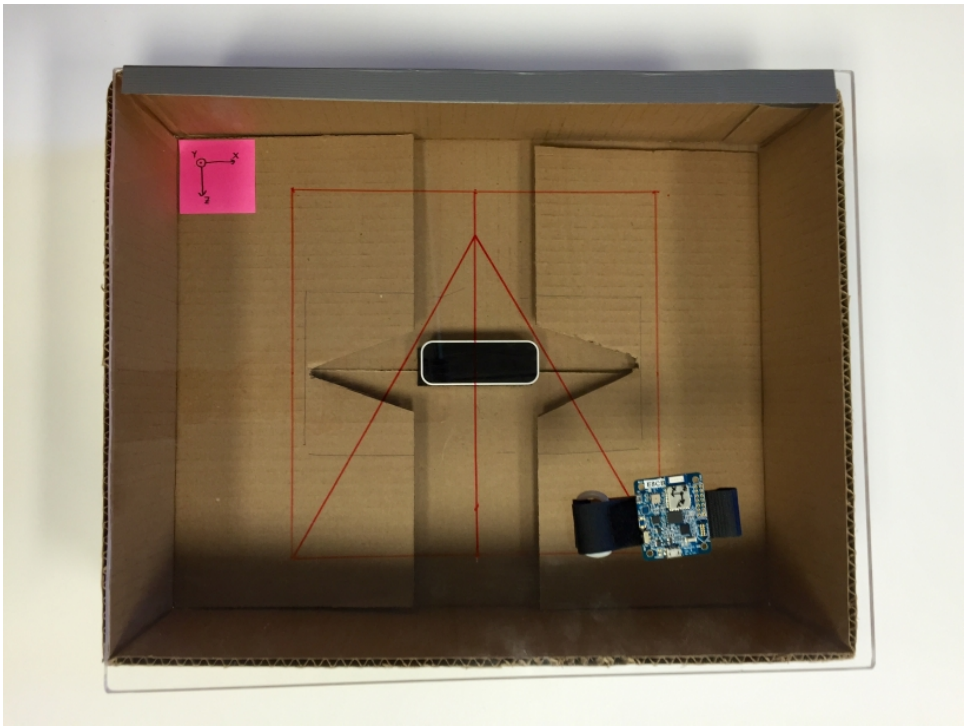


Figure 5.2: Leap sensing platform

ZUPT and double integration to get the reconstructed trajectory.

Figure 5.3 (IMU) shows an example of reconstructed trajectory of square drawings at normal speed. In most cases, ZUPT provides reliable results if stationary periods are correctly detected and thus allows drift cancelation. However, if the motion is fast and stationary periods cannot be clearly identified, drift will build up. Figure 5.4 (IMU) shows the reconstructed trajectory of square drawings at fast speed.

III.3.2 Rigid Transformation

Since the initial relative orientation between the IMU and the Leap can be arbitrary, it is necessary to transform them to the same coordinate system for further analysis. We assume that the Leap controller is physically fixed and thus is our reference coordinate. During the training process, the subject moves his wrist and follows the square marked on the box for three repetitions. Given the training data, we calculate a rigid transform [77] which translates

the IMU's trajectory estimations from the IMU's initial frame to the Leap's reference frame.

III.3.3 Synchronization and Interpolation

We rely on an external signature for synchronization, since there is no handshake communication between the Leap Motion controller and the IMU. At the beginning and the end of each measurement sessions, we ask the subject to lift his/her hands and tap the box for three times to serve as significant signatures in the collected signals, which helps us to align the measured data. The IMU used in this research is configured with a 200 Hz sampling rate, whereas the Leap Motion controller has frame rate that varies from 20 to 200 fps depends on the user's settings and available computing power. Thus, linear interpolation is used to deal with missing data of the Leap Motion controller.

III.3.4 Complementary Filter Sensor Fusion and Parameters Tuning

For our system, there are two types of trajectory estimates that have noise with complementary spectral characteristics. The trajectory estimate from the IMU sensors is accurate in the short term but will suffer from long term drift. On the other hand, the Leap Motion controller is immune from long term drift but has spontaneous jittering noise. In order to remove the corresponding noises, we use the complementary filter to filter the IMU's trajectory estimates with a high-pass filter, and the Leap Motion controller's position estimates with a low-pass filter. We also take the trustworthiness of the Leap's trajectory estimate into account when applying sensor fusion. From our experience, a data frame reported by the Leap with confidence score higher than 0.1 is considered as useable data. If the Leap's trajectory estimate is useable, Eq. 5.1 is applied for sensor fusion. Otherwise, Eq. 5.2 is applied.

$$CF_{est.}(t) = \alpha IMU_{est.}(t) + (1-\alpha)Leap_{est.}(t), 0 \leq \alpha \leq 1 \tag{5.1}$$

$$CF_{est.}(t) = IMU_{est.}(t) \tag{5.2}$$

To determine the optimal α for the complementary filter, we performed an exhaustive search on the training data. We declare the optimal parameter when the mean square error between the reconstructed trajectory and the ground truth squares is minimized.

IV Results

To verify our method, we designed the experiment setup as follows. We attached an IMU sensor at the subject's wrist with arbitrary orientation and asked the subject to place his/her hand on the Leap box top. The subject first performed synchronization signature and training patterns (three hand taps, and three square traces) before performing any of the tasks. After that, the subject performed various pre-defined tasks and followed this by three hand taps for synchronization signature.

IV.1 Trajectory reconstruction

In this experiment, we asked the subjects to perform 4 different tasks including 10 drawing of squares and triangles, at both normal and fast speed. Figure 5.3, 5.4, 5.5 and 5.6 show the reconstructed trajectory of each task from IMU's estimation, Leap's estimation and the sensor fusion result of the complementary filter. The results show that the IMU's estimation produces a smooth trajectory but suffers from drift in fast motion. On the contrary, the Leap's estimation is immune from drift but contains much jitteriness, especially for those regions with lower confidence scores. The sensor fusion result of the complementary filter provides a nice blend of the two above-mentioned estimations. Table 5.1 shows the mean square error of reconstructed trajectory for each method, where the error is defined as the difference between reconstructed patterns and the ground truth.

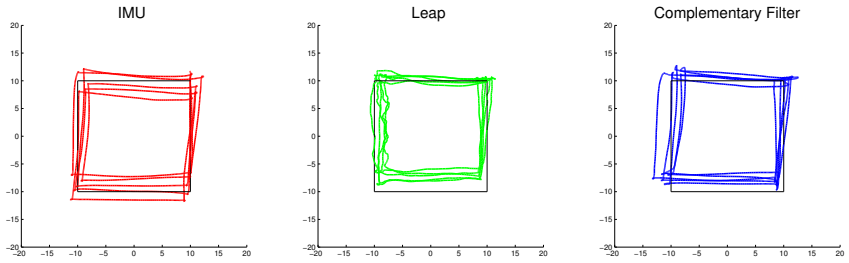


Figure 5.3: Reconstructed trajectory: square

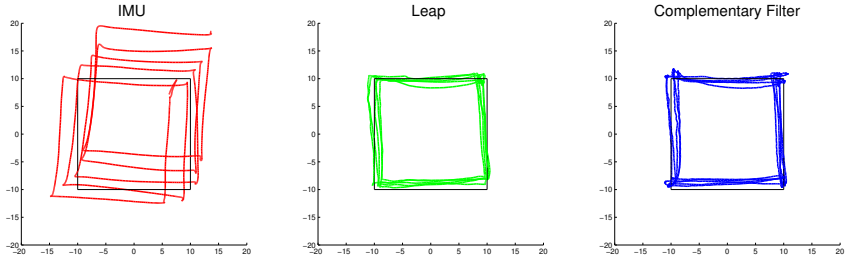


Figure 5.4: Reconstructed trajectory: square (fast motion)

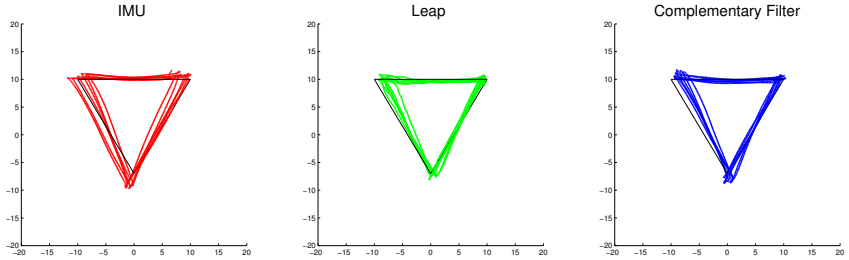


Figure 5.5: Reconstructed trajectory: triangle

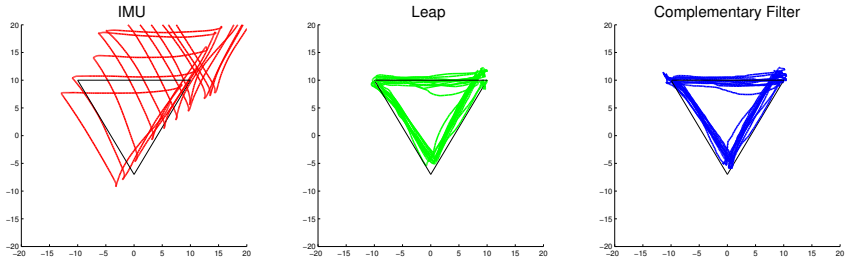


Figure 5.6: Reconstructed trajectory: triangle (fast motion)

Table 5.1: Mean Square Error (in cm)

	IMU	Leap	Complementary
Square	2.24	2.56	2.54
Square (fast)	83.71	2.91	2.85
Triangle	1.21	1.45	1.39
Square (fast)	1159.29	3.76	4.32

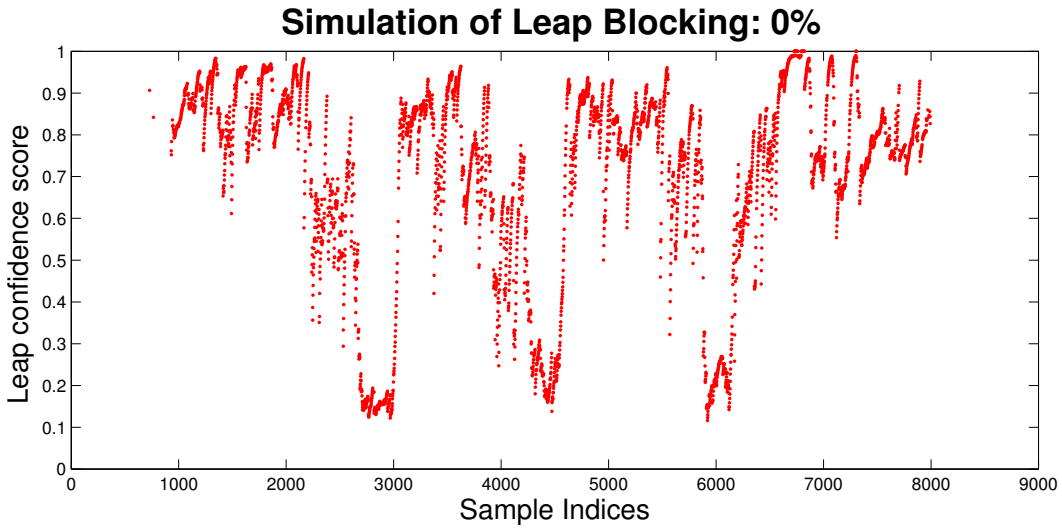


Figure 5.7: Simulation of Leap blocking: 0%

IV.2 Trajectory Reconstruction with Leap Blocking

In the real world scenario, trajectory estimates from the Leap Motion controller might be unavailable due to obstructions or exceeding the sensing range. To verify whether our system can handle this situation, we designed an experiment to simulate Leap blocking. Instead of physically blocking the view of the Leap Motion controller, we did a simulation by setting a portion of the confidence score to zero. We corrupted the Leap Motion controller's data in steps of 10% starting from 0% to 90% and examined the error between our reconstructed trajectory and the ground truth. Figure 5.7 shows the confidence score of a motion segment where the wrist was always in the sensing range, while Figure 5.8 is a simulation of blocking happening 40 % of the time.

Figure 5.9 and 5.10 shows the reconstructed trajectories for 0% and 40% blocking. The

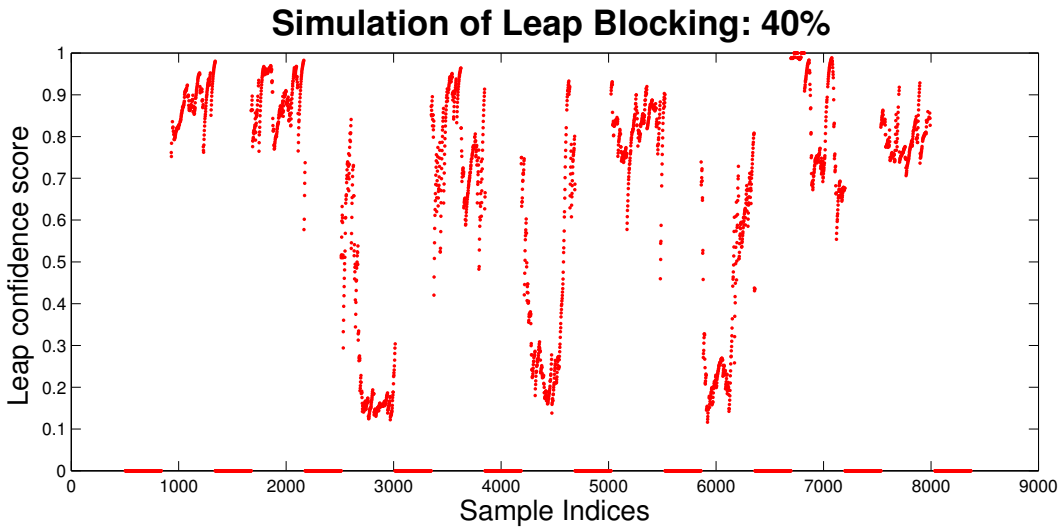


Figure 5.8: Simulation of Leap blocking: 40%

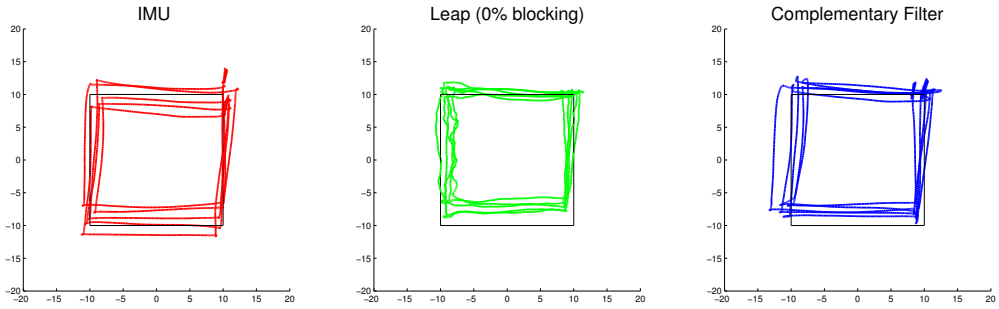


Figure 5.9: Reconstructed trajectories (0% Leap blocking)

results show that even with 40% blocking, the complementary filter still performs fairly well. For each trajectory estimate with different levels of blocking, we calculated the mean square error to the ground truth square. Results in Figure 5.11 suggest that our method can provide better trajectory reconstruction than using the IMU or the Leap Motion controller alone.

For further validation, we reconstruct the motion trajectory with Leap blocking due to being out of the sensing range. The subject followed a ground truth guide and moved his wrist from the box top to the table, back and forth for 10 times. Since the Leap has limited sensing range, no trajectory estimate from the Leap was available when the subject's wrist moved out of the Leap's view. Figure 5.12 shows the Leap's confidence score becomes zero

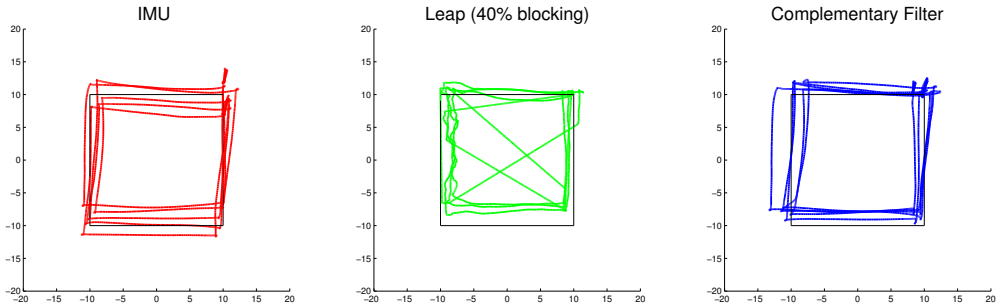


Figure 5.10: Reconstructed trajectories (40% Leap blocking)

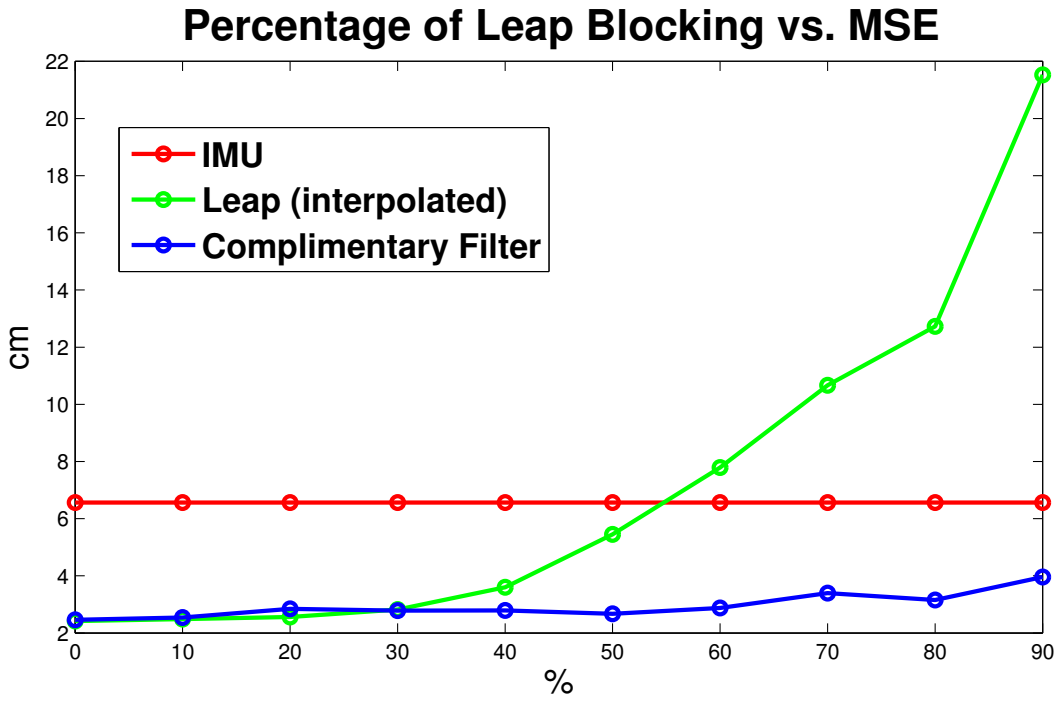


Figure 5.11: Percentage of Leap Blocking vs. MSE

when the wrist is out of sensing range. Figure 5.13 shows the reconstructed trajectory for three types of trajectory estimates. The IMU's estimate provides continuous motion tracking but contains drift, whereas the Leap's estimate doesn't drift but has missing data. With the complementary filter, we effectively eliminated the drift and achieved continuous motion tracking.

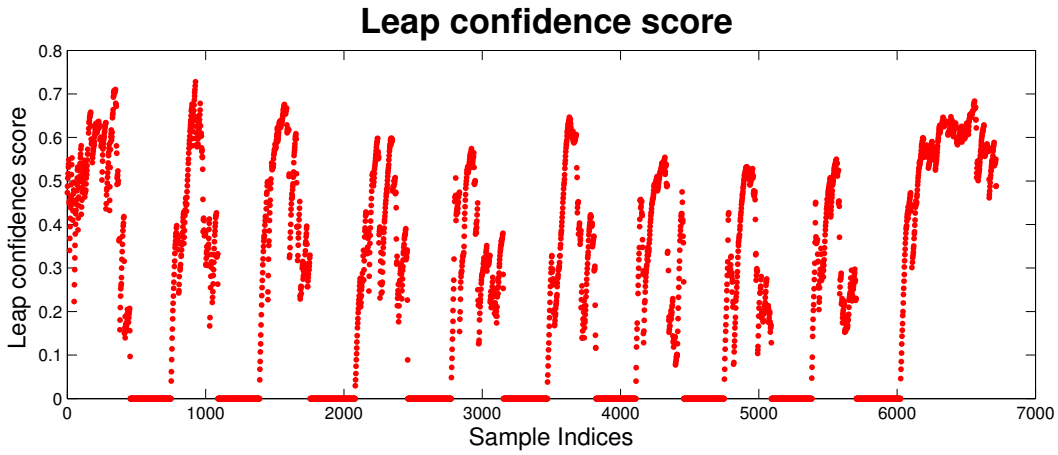


Figure 5.12: Leap confidence score

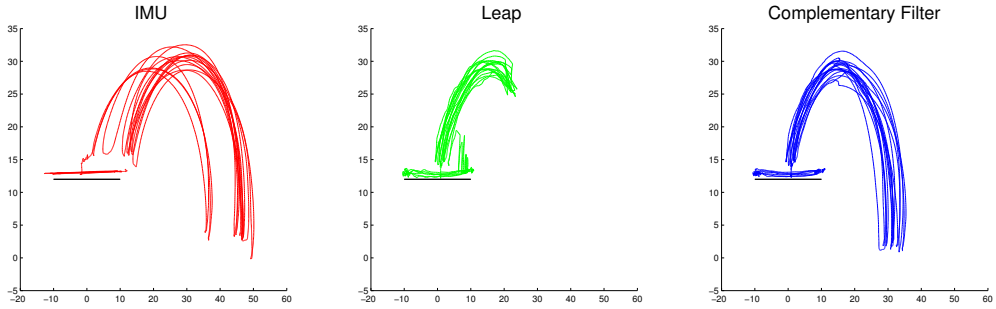


Figure 5.13: Reconstructed trajectory

V Conclusions

In this research, we present a sensor fusion approach that tracks and reconstructs upper limb motion using the Leap Motion controller and IMUs. We also designed and conducted experiments to evaluate the performance. The results indicate that our method provides a solution to address the integration drift of the IMU and the blocking issues of the Leap Motion controller.

The outcomes of this research will enable low-cost means that benefit medical professionals and therapists who want to analyze human motion trajectories in detail. In addition, it can also be included in the remote monitoring system for many medical purposes, e.g., a medical surveillance system which keeps track of patients requiring long-term care, or a

system to see if the patients in rehabilitation have followed doctors' directions to exercise for a prescribed amount of time daily. Our study was however limited to healthy subjects. In the future, we plan to investigate performance with patients in both clinical and home settings.

CHAPTER 6

Conclusions and Future Research

I Research Contribution

In this dissertation, we solved a series of problems that centered at enabling ***robust*** and ***large-scale*** human motion inference with ***low-cost sensors***.

We started from the implementation of a real-time activity classification system that integrates universal hybrid decision tree and context information to deal with scalability issues. We achieved advanced classification accuracy and energy efficiency. We realized that collecting ground truth inertial data is a burden and sought for an efficient way to acquire data. This led us to Project 2: "Virtual Inertial Measurements for Motion Inference in Wireless Health".

In Project 2, we developed the virtual inertial measurement algorithm and validated our method through both low-cost Kinect and high-end Vicon system. We also developed a Matlab toolbox to systematically generate virtual sensor measurements from the CMU MoCap database. The potential of using the low-cost Kinect as the ground truth inspired us to dive into Project 3: "Opportunistic Calibration of Sensor Orientation using the Kinect and Inertial Measurement Unit Sensor Fusion".

In Project 3, we proposed an elegant solution to deal with sensor orientation issues and verified its validity. And then, we looked beyond the sensor orientation issues and moved one step further to deal with sensor drift in project 4: "Robust Upper Limbs Motion Tracking using Sensor Fusion of the Leap Motion Controller and IMUs".

In Project 4, we presented and validated a sensor fusion approach to achieve robust upper limbs motion tracking. We also overcame inertial sensor drift issues and enabled continuously motion tracking when the Leap controller has obscured view.

In conclusion, all of the 4 projects are tightly connected with my research objective: enabling ***robust*** and ***large-scale*** human motion inference with ***low-cost sensors***. They all used low-cost sensors, tried to solve issues in scalability, and aimed to provide robust human motion inference.

II Future Research

Future work exists in integration of system in above-mentioned projects to enable continuous whole-body motion tracking at a residential setting and interactive guidance for rehab exercise.

To enable continuous whole-body motion tracking at a residential setting, the following three challenges must be addressed: model complexity, user compliance (sensor attachment) and sensor uncertainties (drift, sensing range). Based on the current research presented in this thesis, we have provided solutions to these challenges.

For interactive guidance for rehab exercise, the Leap motion tracking box presented in the Project 4 would be an effective tool for rehab purposes. For example, physicians can prescribe rehab tasks such as reaching and grasping to patients. The motion data acquired from the Leap motion tracking box will be used to determine the quantity and quality of rehab tasks, and then provide timely feedback to both patients and physicians.

References

[1] W. He, D. Goodkind, and P. Kowal, "An aging world: 2015," *Washington, DC: US Census Bureau*, 2016.

[2] M. Mathie, B. G. Celler, N. H. Lovell, and A. Coster, "Classification of basic daily movements using a triaxial accelerometer," *Medical and Biological Engineering and Computing*, vol. 42, no. 5, pp. 679–687, 2004.

[3] D. Roetenberg, P. J. Slycke, and P. H. Veltink, "Ambulatory position and orientation tracking fusing magnetic and inertial sensing," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 5, pp. 883–890, 2007.

[4] H. Schepers and P. Veltink, "Stochastic magnetic measurement model for relative position and orientation estimation," *Measurement Science and Technology*, vol. 21, no. 6, p. 065801, 2010.

[5] D. Roetenberg, H. Luinge, and P. Slycke, "Xsens mvn: full 6dof human motion tracking using miniature inertial sensors," *Xsens Motion Technologies BV, Tech. Rep*, 2009.

[6] K. Saremi, J. Marehbian, X. Yan, J.-P. Regnaux, R. Elashoff, B. Bussel, and B. H. Dobkin, "Reliability and validity of bilateral thigh and foot accelerometry measures of walking in healthy and hemiparetic subjects," *Neurorehabilitation and Neural Repair*, vol. 20, no. 2, pp. 297–305, 2006.

[7] B. H. Dobkin, X. Xu, M. Batalin, S. Thomas, and W. Kaiser, "Reliability and validity of bilateral ankle accelerometer algorithms for activity recognition and walking speed after stroke," *Stroke*, vol. 42, no. 8, pp. 2246–2250, 2011.

[8] X. Xu, M. A. Batalin, W. J. Kaiser, and B. Dobkin, "Robust hierarchical system for classification of complex human mobility characteristics in the presence of neurological disorders," in *2011 International Conference on Body Sensor Networks*. IEEE, 2011, pp. 65–70.

[9] Y. Wang, X. Xu, M. Batalin, and W. Kaiser, "Detection of upper limb activities using multimode sensor fusion," in *2011 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2011, pp. 436–439.

[10] M. Windolf, N. Götzen, and M. Morlock, "Systematic accuracy and precision analysis of video motion capturing systems - exemplified on the vicon-460 system," *Journal of biomechanics*, vol. 41, no. 12, pp. 2776–2780, 2008.

[11] B. Carse, B. Meadows, R. Bowers, and P. Rowe, "Affordable clinical gait analysis: An assessment of the marker tracking accuracy of a new low-cost optical 3d motion analysis system," *Physiotherapy*, vol. 99, no. 4, pp. 347–351, 2013.

[12] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 16, 2011.

[13] Š. Obdržálek, G. Kurillo, F. Ofli, R. Bajcsy, E. Seto, H. Jimison, and M. Pavel, "Accuracy and robustness of kinect pose estimation in the context of coaching of elderly population," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2012, pp. 1188–1193.

[14] J. Y. Tung, T. Lulic, D. A. Gonzalez, J. Tran, C. R. Dickerson, and E. A. Roy, "Evaluation of a portable markerless finger position capture device: accuracy of the leap motion controller in healthy adults," *Physiological measurement*, vol. 36, no. 5, p. 1025, 2015.

[15] L. Bottou and C.-J. Lin, "Support vector machine solvers," *Large scale kernel machines*, pp. 301–320, 2007.

[16] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[17] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics Springer, Berlin, 2001, vol. 1.

[18] M.-k. Suh, L. S. Evangelista, V. Chen, W.-S. Hong, J. Macbeth, A. Nahapetian, F.-J. Figueras, and M. Sarrafzadeh, "Wanda b.: Weight and activity with blood pressure monitoring system for heart failure patients," in *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*. IEEE, 2010, pp. 1–6.

[19] L. K. Au, M. Batalin, B. Jordan, C. Xu, A. A. Bui, B. Dobkin, and W. J. Kaiser, "Demonstration of whi-fit: a wireless-enabled cycle restorator," in *Wireless Health 2010*. ACM, 2010, pp. 190–191.

[20] C. Chien and G. J. Pottie, "A universal hybrid decision tree classifier design for human activity classification," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2012, pp. 1065–1068.

[21] B. Dobkin, X. Xu, M. Batalin, S. Thomas, and W. Kaiser, "Accelerometry algorithms for activity recognition," *Stroke*, 2011.

[22] B. Logan and J. Healey, "Sensors to detect the activities of daily living," in *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*. IEEE, 2006, pp. 5362–5365.

[23] A. R. Jimenez, F. Seco, C. Prieto, and J. Guevara, "A comparison of pedestrian dead-reckoning algorithms using a low-cost mems imu," in *Intelligent Signal Processing, 2009. WISP 2009. IEEE International Symposium on*. IEEE, 2009, pp. 37–42.

[24] C. Chien, J. Y. Xu, H.-I. Chang, X. Wu, and G. J. Pottie, "Model construction for human motion classification using inertial sensors," in *IEEE Workshop on Information Theory and Applications, San Diego*, 2013.

[25] C. Chieh, J. Xia, O. Santana, Y. Wang, and G. Pottie, "Non-linear complementary filter based upper limb motion tracking using wearable sensors," in *International Conference on Acoustics, Speech, and Signal Processing*, May 2013.

[26] X. Wu, Y. Wang, and G. Pottie, "A non-zupt gait reconstruction method for ankle sensors," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2014, pp. 5884–5887.

[27] Y. Wang, J. Xu, X. Xu, X. Wu, G. Pottie, and W. Kasier, "Inertial sensor based motion trajectory visualization and quantitative quality assessment of hemiparetic gait," in *Proceedings of the 8th International Conference on Body Area Networks*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013, pp. 169–172.

[28] Y. Wang, J. Xu, X. Wu, G. Pottie, and W. Kaiser, "A simple calibration for upper limb motion tracking and reconstruction," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2014, pp. 5868–5871.

[29] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen, "Activity classification using realistic data from wearable sensors," *IEEE Transactions on information technology in biomedicine*, vol. 10, no. 1, pp. 119–128, 2006.

[30] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *International Conference on Pervasive Computing*. Springer, 2004, pp. 1–17.

[31] L. Atallah, B. P. Lo, R. C. King, and G.-Z. Yang, "Sensor placement for activity detection using wearable accelerometers." in *BSN*, 2010, pp. 24–29.

[32] X. Long, B. Yin, and R. M. Aarts, "Single-accelerometer-based daily physical activity classification," in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2009, pp. 6107–6110.

[33] G. Landeweerd, T. Timmers, E. S. Gelsema, M. Bins, and M. Halie, "Binary tree versus single level tree classification of white blood cells," *Pattern Recognition*, vol. 16, no. 6, pp. 571–577, 1983.

[34] X. Li and R. C. Dubes, "Tree classifier design with a permutation statistic," *Pattern Recognition*, vol. 19, no. 3, pp. 229–235, 1986.

[35] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," 1990.

[36] A. Schmidt, M. Beigl, and H.-W. Gellersen, "There is more to context than location," *Computers & Graphics*, vol. 23, no. 6, pp. 893–901, 1999.

[37] A. K. Dey, "Understanding and using context," *Personal and ubiquitous computing*, vol. 5, no. 1, pp. 4–7, 2001.

[38] D. P. Siewiorek, A. Smailagic, J. Furukawa, A. Krause, N. Moraveji, K. Reiger, J. Shaffer, and F. L. Wong, "Sensay: A context-aware mobile phone." in *ISWC*, vol. 3, 2003, p. 248.

[39] K. Van Laerhoven, A. Schmidt, and H.-W. Gellersen, "Multi-sensor context aware clothing," in *Wearable Computers, 2002.(ISWC 2002). Proceedings. Sixth International Symposium on*. IEEE, 2002, pp. 49–56.

[40] L. Han, S. Jyri, J. Ma, and K. Yu, "Research on context-aware mobile computing," in *Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on*. IEEE, 2008, pp. 24–30.

[41] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[42] G. De'ath and K. E. Fabricius, "Classification and regression trees: a powerful yet simple technique for ecological data analysis," *Ecology*, vol. 81, no. 11, pp. 3178–3192, 2000.

[43] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger, J. Altmann, and W. Retschitzegger, "Context-awareness on mobile devices-the hydrogen approach," in *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. IEEE, 2003, pp. 10–pp.

[44] P. Prekop and M. Burnett, "Activities, context and ubiquitous computing," *Computer communications*, vol. 26, no. 11, pp. 1168–1176, 2003.

[45] J. Y. Xu, Y. Sun, Z. Wang, W. J. Kaiser, and G. J. Pottie, "Context guided and personalized activity classification system," in *Proceedings of the 2nd Conference on Wireless Health*. ACM, 2011, p. 12.

[46] S. Suvorova, T. Vaithianathan, and T. Caelli, "Action trajectory reconstruction from inertial sensor measurements," in *International Conference on Information Science, Signal Processing and their Applications*, 2012.

[47] S. Bertrand, T. Hamel, H. Piet-Lahanier, and R. Mahony, "Attitude tracking of rigid bodies on the special orthogonal group with bounded partial state feedback," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, 2009, pp. 2972–2977.

[48] M. El-Gohary, L. Holmstrom, J. Huisinga, E. King, J. McNames, and F. Horak, "Upper limb joint angle tracking with inertial sensors," in *Engineering in Medicine and Biology Society,EMBC, 2011 Annual International Conference of the IEEE*, 2011, pp. 5629–5632.

[49] Y. Wang, C. Chien, J. Xu, G. Pottie, and W. Kaiser, "Gait analysis using 3d motion reconstruction with an activity-specific tracking protocol," in *The 38th International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, May 2013.

[50] X. Wu, Y. Wang, C. Chieh, and G. Pottie, "Self-calibration of sensor misplacement based on motion signatures," in *Annual Body Sensor Networks Conference*, May 2013.

[51] "CMU graphics lab motion capture database," *Available online: http://mocap.cs.cmu.edu/.*

[52] O. Woodman, "An introduction to inertial navigation," *University of Cambridge Tech. Report*, 2007.

[53] D. S. Cleveland WS, "Locally-weighted regression: an approach to regression analysis by local fitting," 1988.

[54] S. Suvorova, T. Vaithianathan, and T. Caelli, "Action trajectory reconstruction from inertial sensor measurements," in *Information Science, Signal Processing and their Applications (ISSPA), 2012 11th International Conference on.* IEEE, 2012, pp. 989–994.

[55] M. El-Gohary, L. Holmstrom, J. Huisinga, E. King, J. McNames, and F. Horak, "Upper limb joint angle tracking with inertial sensors," in *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE.* IEEE, 2011, pp. 5629–5632.

[56] C.-Y. Lee and J.-J. Lee, "Estimation of walking behavior using accelerometers in gait rehabilitation," *International Journal of Human-friendly Welfare Robotic Systems*, vol. 3, no. 2, pp. 32–36, 2002.

[57] M. Villamizar, A. Sanfeliu, and J. Andrade-Cetto, "Orientation invariant features for multiclass object recognition," in *Progress in Pattern Recognition, Image Analysis and Applications.* Springer, 2006, pp. 655–664.

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

[59] K. Forster, P. Brem, D. Roggen, and G. Troster, "Evolving discriminative features robust to sensor displacement for activity recognition in body area sensor networks," in *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2009 5th International Conference on.* IEEE, 2009, pp. 43–48.

[60] A. Friedman, N. Hajj Chehade, C. Chien, and G. Pottie, "Estimation of accelerometer orientation for activity recognition," in *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*. IEEE, 2012, pp. 2076–2079.

[61] X. Wu, Y. Wang, C. Chien, and G. Pottie, "Self-calibration of sensor misplacement based on motion signatures," in *Body Sensor Networks (BSN), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–5.

[62] C. Chien, J. Xia, O. Santana, Y. Wang, and G. J. Pottie, "Non-linear complementary filter based upper limb motion tracking using wearable sensors," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 963–967.

[63] Y. Wang, C. Chien, J. Xu, G. Pottie, and W. Kaiser, "Gait analysis using 3d motion reconstruction with an activity-specific tracking protocol," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1041–1045.

[64] S. Obdrzalek, G. Kurillo, F. Ofli, R. Bajcsy, E. Seto, H. Jimison, and M. Pavel, "Accuracy and robustness of kinect pose estimation in the context of coaching of elderly population," in *Engineering in medicine and biology society (EMBC), 2012 annual international conference of the IEEE*. IEEE, 2012, pp. 1188–1193.

[65] M. Edwards and R. Green, "Low-latency filtering of kinect skeleton data for video game control," in *Proceedings of the 29th International Conference on Image and Vision Computing New Zealand*. ACM, 2014, p. 190.

[66] B. Alkire, "Character animation and gesture-based interaction in an immersive virtual environment using skeletal tracking with the microsoft kinect," Ph.D. dissertation, Citeseer, 2012.

[67] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," *Matrix*, vol. 58, pp. 15–16, 2006.

[68] O. Woodman, "An introduction to inertial navigation." University of Cambridge, Tech. Rep., 2007.

[69] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *Computer Graphics and Applications, IEEE*, vol. 25, no. 6, pp. 38–46, 2005.

[70] S. Williams, R. Schmidt, C. Disselhorst-Klug, and G. Rau, "An upper body model for the kinematical analysis of the joint chain of the human arm," *Journal of biomechanics*, vol. 39, no. 13, pp. 2419–2429, 2006.

[71] B. Hingtgen, J. R. McGuire, M. Wang, and G. F. Harris, "An upper extremity kinematic model for evaluation of hemiparetic stroke," *Journal of biomechanics*, vol. 39, no. 4, pp. 681–688, 2006.

[72] G. Kurillo, A. Chen, R. Bajcsy, and J. J. Han, "Evaluation of upper extremity reachable workspace using kinect camera," *Technology and Health Care*, vol. 21, no. 6, pp. 641–656, 2013.

[73] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler, "Analysis of the accuracy and robustness of the leap motion controller," *Sensors*, vol. 13, no. 5, pp. 6380–6393, 2013.

[74] W. T. Higgins, "A comparison of complementary and kalman filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 11, no. 3, pp. 321–325, 1975.

[75] S. Osder, W. Rouse, and L. Young, "Navigation, guidance, and control systems for v/stol aircraft." 1973.

[76] J. L. Marins, X. Yun, E. R. Bachmann, R. B. McGhee, and M. J. Zyda, "An extended kalman filter for quaternion-based orientation estimation using marg sensors," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 4. IEEE, 2001, pp. 2003–2011.

[77] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *JOSA A*, vol. 4, no. 4, pp. 629–642, 1987.