1-1-2021

A Novel Approach Toward High Accuracy Indoor Localization}{Masters of Science

Yunshu Wang
University of Mississippi

A NOVEL APPROACH TOWARD HIGH ACCURACY INDOOR LOCALIZATION

A Thesis
presented in partial fulfillment of requirements
for the degree of Masters of Science
in the Department of Computer and Information Science
The University of Mississippi

by

Yunshu Wang

May 2021

ABSTRACT

This thesis presents a novel approach towards high accuracy indoor localization with smartphones. Better than all the existing indoor localization methods, our approach has the advantage of being infrastructural-free, robust, and it does not require any pre-installation in a new environment. To make this goal come true, we built a testbed that only uses the Inertial Measurement Units (IMU) of the smartphones to access the smartphones' raw acceleration and orientation data, then use these data to calculate the user's location by coordinate transformation and interactions. We conducted extensive experiments and evaluations for testbed validation as well as to carefully examine in depth on the well-known drifting problem of the Inertial Navigation System (INS), which is caused by imprecise sensor readings. When walking straight, these sensor errors and noises show no regular patterns on the accelerations. However, after one-time integration on accelerations, the resulting velocities on each dimension show clear drifting patterns where they always drift in linear order, which is caused by the periodic drift in each step. Inspired by the drifting patterns, we found the drifting problem for walking in a straight line can be greatly reduced by performing linear regression on the velocities. As the normal pedestrian's walking behavior can be separated into walking straight section and turn section, we adopt a deep learning approach to treat them differently with different calibration techniques. Our calibrated result shows a great improvement comparing to the result calculated from the raw smartphone data.

ACKNOWLEDGEMENTS

I would like to thank my mentor, Dr.Feng Wang, for giving me directions and made me found my interests. I appreciate his meticulous guidance and supports in my researches and studies. On a personal note, Dr. Wang is my idol that his passion and self-disciplined make me realize the importance of always keep learning. It is my honor to work with Dr. Wang.

I also would like to thank my thesis committee, Dr.Dawn E. Wilkins and Dr.Yixin Chen. I thank them for their help and guidance in my study. I am thankful to Dr. Wilkins that she taught me many useful analysis skills which is very helpful to my researches. I want to thank Dr. Chen for what I have learned from the algorithm classes, it makes me sharper when countering new problems.

I want to thank my parents for their supports and encouragement as well. I am also thankful for the department of computer science at the University of Mississippi providing me a good studying environment.

TABLE OF CONTENTS

ABSTRACT ii

ACKNOWLEDGEMENTS iii

LIST OF FIGURES vi

INTRODUCTION 1

TECHNICAL BACKGROUND AND RELATED WORKS 5

2.1 Android Mobile Application 5

2.2 Inertial Navigation System 6

2.3 Inertial Measurement Units and the Drifting Problem 7

2.4 Firebase Database 8

2.5 Related Works 9

TESTBED DEVELOPMENT 12

3.1 Testbed Architecture 12

3.2 Functionality for Localization 15

TESTBED EVALUATION AND VALIDATION 20

4.1 Synthetic Experiment 20

4.2 Real World Experiments 23

4.3 Interesting Observations 31

4.4 Calibration Technique for Walking Straight 33

4.5 Calibration Technique for Spinning a Circle 36

A NOVEL ENHANCEMENT APPROACH 38

5.1 Main Idea 38

5.2 Walking Behavior Segmentation 39

5.3 Orientation Change 41

5.4 Improve the Accuracy of the Typical Scenario 50

CONCLUSION AND FUTURE WORKS 55

6.1 Conclusion 55

6.2 Future Works 56

BIBLIOGRAPHY 58

VITA 61

LIST OF FIGURES

3.1 Testbed Architecture 13
3.2 Screenshot of the WebGL Application 15
4.1 The Top View of the Experiment Building 21
4.2 Overall Verification 24
4.3 Circle Experiment 25
4.4 Walking 10 Meters, Smartphone Held in the Front 26
4.5 Walking 10 Meters, Smartphone Swing in Hand 27
4.6 Make a Turn, Smartphone in the Front 29
4.7 Stepping in Place, Smartphone Swing in Hand 30
4.8 Static on the Flat Ground, Smartphone in the Front 32
4.9 Calibration Results for Walking with Phone in the Front 34
4.10 Calibration Results for for Smartphone in Hand Swinging 35
4.11 Calibration Results for Circle Experiment 37
5.1 Flowchart of Our Novel Solution 40
5.2 Result of bi-LSTM 42
5.3 Orientation Change Experiment Setup 43
5.4 Orientation Change Experiment Orientation Top View 44
5.5 Original results of Spin Circles 45
5.6 Calibration Result of Spin Circles 46
5.7 Calibration Result of Spin Circles Comparing to the Ground Truth. 47
5.8 Final Locations of Spinning one circle. similar speed. 48
5.9 Final Locations of Spinning one circle, same radius. 49
5.10 Original Make a Turn Problem Results 51
5.11 Calibrated Make a Turn Problem Results 52
5.12 Calibration Results for MaT, Forcing the Velocity After the Turn the Same as Before the Turn 54

Chapter 1

INTRODUCTION

Thanks to the advances of satellite and wireless technologies, nowadays with the Global Position System (GPS) people can accurately know their positions no matter where they are traveling over the world [2]. However, the GPS signals can become too weak to be a reliable source for positioning inside buildings, despite that people stay indoor 80% to 90% of the time [15]. This can be very inconvenient for applications such as indoor navigation, indoor parkade, and localization of indoor objects. Therefore, it is necessary to come up with a strategy that could address the indoor localization problem accurately and efficiently. And many efforts have been devoted towards this goal. For example, Hesch *et al* [5] used a professional Inertial Measurement Unit (IMU) and built an Inertial Navigation System (INS) with the help of Extended Kalman Filter (EKF) and a laser scanner for correction to achieve centimeter-level accuracy. However, such systems usually need additional hardware and equipment, which can take extra cost and be inconvenient in practice.

In recent years, smartphones have become more and more popular in people's daily life. They are capable of performing a variety of functions and embodied with numerous sensors including miniature IMUs, bringing new opportunities to achieve convenient and lightweight indoor localization. Yet one major challenge therein is how to handle the sensor errors that are not negligible especially compared to high-end INSs and can become prob-

lematic when accumulated over time. To overcome this issue, recent researches have mainly proposed two categories of solutions, which are based on fingerprint and pedestrian dead reckoning (PDR), respectively. The fingerprint approach [18] exploits the characteristics of certain sensor data types such as WiFi signal and magnetic field strength, whose readings can be used to uniquely identify a location like fingerprints to identify a person. This approach, however, usually needs lots of effort to build and update a fingerprint database, even if there is only minor changes in the environment, e.g., WiFi access point infrastructure adjustments or changes of magnetic field sources. Compared to the fingerprint approach, the PDR approach [10] [11] does not rely on environmental readings. Instead, it takes the accelerometer data and orientation readings from a smartphone to count the number of footsteps having been taken, and then calculates the current position based on estimated step length and the dead reckoning algorithm [12]. However, footsteps may not be consistently stable and can vary a lot while people are turning, going upstairs/downstairs, taking moving walkways/escalators, using smartphones, etc., which can lead to significant errors for localization.

There are also other proposed methods such as Trilateration [14] and Proximity Estimation [16], which heavily depend on a number of pre-installed external sensors/beacons. Another recently proposed solution is visual localization [10], which, however, needs that the smartphone cameras must be unobstructed with specific holding gestures, and can consume much higher energy compared to other approaches.

As such, it is still elusive to have an all-around solution that can achieve high accuracy, robustness, infrastructure-free[1] and no pre-installation. In this thesis, we take one step further towards tackling this daunting challenge by carefully developing a testbed that can allow us to have a deep investigation of the smartphone-based indoor localization problem and potentially propose promising solutions. To make it infrastructure-free and require no pre-installation, our testbed only accesses the IMU and orientation data from the smartphone

[1]Here infrastructure-free means no infrastructure is used for localization purpose.

for solution design. However, different from the PDR approach that uses filters and counts footsteps, our testbed directly collects all the raw data without any filtering, which provides us an in-depth view of where the sensor errors come from and how they can affect the localization accuracy. Moreover, our testbed not only allows post-diagnosing on collected data but also provides built-in functionalities for localization and supports real-time data processing and visualization that can be extremely valuable for solution development and practical usefulness.

We have conducted extensive experiments to evaluate our testbed, and obtained interesting observations that are very important for our novel solution's design. In particular, we find that although filtering (e.g., low-pass filtering) can help eliminate some noises of the sensor readings, it may also mistakenly remove useful data when people make sudden movements such as turning, starting to go upstairs/downstairs, or changing the way to hold the smartphone. Moreover, we notice that the sensor errors are relatively stable, which keeps almost consistent within the duration of a series of coherent movements. These findings not only validate the effectiveness of our testbed design but also motivate a novel approach to identify and thus compensate for such relatively stable sensor errors to achieve better accuracy.

From many different experiments that we learned the drifting problems for different walking behaviors have different drifting patterns. Therefore, we came up with different strategies to improve the accuracy differently. Because a normal pedestrian's walking behavior can be converted into a combination of walking straight and making a turn, we offered a typical walking trace that is enough to cover all the walking behaviors of a pedestrian on a 2D plane. In the typical walking trace, the user walks straight toward a direction, then makes a 90-degree turn, and walking straight again. Our novel approach segments this typical walking behavior into different walking patterns, and treat them with their distinctive calibration techniques differently. The result after applying our novel indoor localization approach could improve the accuracy of the typical walking trace by a great amount comparing

to the result calculated by the raw sensor data. It proves our observations and the novel indoor localization approach is helpful toward a high accuracy indoor localization.

Chapter 2

TECHNICAL BACKGROUND AND RELATED WORKS

2.1 Android Mobile Application

An Android Mobile Application is a software application that runs on Android mobile devices. It was originated from a modified version of the Linux kernel and other open-source software for touchscreen mobile devices. With more and more demands on the functionality of mobile apps, the mobile app functionalities have rapid expanded from simple software such as email, calculator, and calendar to more complex software including mobile games, navigation systems, and communication applications. Nowadays, there are millions of mobile apps available on different application distribution platforms.

Using Integrated Development Environments (IDEs) such as Android Studio or a browser-based platform named App Inventor, Android apps can be designed with Android Software Development Kit (SDK). [1] Android apps can be written in Kotlin, Java, and C++ languages, and the Android SDK will compile the code and resource files into an Android application package) APK file that is used to be install the app on Android devices.

There are many features in Android, such as interface, home screen, status bar,

Notifications, app lists, navigation buttons, and split-screen view. As for the applications, they can be downloaded and installed with the applications' APK, or by download using an application store program, such as Google Play Store. There are also many third-party application marketplaces existing for Android due to its open nature, including Amazon Appstore and GetJar.

2.2 Inertial Navigation System

An inertial navigation system (INS) usually refers to a navigation device that has a computer, motion sensors, and rotation sensors. It uses the readings from the sensors to continuously calculate the position by dead reckoning.

The history of an INS started with a navigation system developed for rockets. The early German World War II V2 guidance system was such an INS system, which was built by combining two gyroscopes and a lateral accelerometer, with the help of a simple analog computer. With the techniques advanced, nowadays the INS usually uses a barometric altimeter or magnetometers, and sometimes speed measuring devices. The usage of INS also became more widespread. It can be used on robots, vehicles, ships, aircraft, submarines, guided missiles, and spacecraft.

The disadvantage of inertial navigation relies on the known initial velocity and position. However, there are many advantages of INS. Because the INS only depends on Newton's laws of classical mechanics, it is the only form of navigation that does not rely on any external aids or visibility conditions. This makes it a potential solution to work anywhere, even in severe environments, such as inside the tunnels, and underwater. Another advantage of INS is that it does not receive or emit signals, which makes it immune to many forms of jamming and attacks.

2.3 Inertial Measurement Units and the Drifting Problem

An inertial measurement unit (IMU) refers to a device that measures specific force, angular rate, and sometimes the orientation of the device. These data is usually read from a combination of sensors, such as accelerometers, gyroscopes, and sometimes magnetometers. IMUs are always incorporated into INS for navigation purposes. When magnetometers are used, they can provide the initial heading reference.

2.3.1 Sensor Errors

Both gyroscope and accelerometer sensors can generate sensor errors, and they can be categorized as follows:

- Offset Error

  The offset error can be separated into stability performance and repeatability. The stability performance is the drift when the sensor remains in invariant conditions, where the repeatability is the error between two measurements in similar conditions but separated by varied conditions in between.

- Scale Factor Error

  The Scale Factor Error (SFE) is a first-order error cost by the non repeatabilities and nonlinearities.

- Misalignment Error

  The misalignment error is usually cost by the imperfect mechanical installation.

- Noise

  Noise refers to a term for the unwanted modification of the signal, which is usually random and carries no useful information. These noises can be categorized by the "color"

of the noise, which is the statistical properties, and how it modifies the intended signal, such as White noise, Black noise, Gaussian noise, Pink noise or flicker noise, Brownian noise, Contaminated Gaussian noise, Power-low noise, Cauchy noise, Multiplicative noise, Quantization error, Poisson noise, Shot noise, Transient noise, Burst noise, and Phase noise.

- Environment Sensitivity

  The thermal gradients and the accelerations can also affect the accuracy of the sensor accuracy.

2.3.2 Disadvantages

The typical disadvantage of using IMUs for navigation is that it suffers from the accumulated error, which is also known as the drifting problem. This is because the guidance system keeps integrating the acceleration data to calculate velocity and integrate the velocity to get the location. Any measurement error, no matter how big, will be accumulated over time and can cause a huge drifting error when the experiment lasts a longer time. Even with the best accelerometer (at 10 µg), and not considering the error of gyroscope, the losses will be more than 50-meter accuracy after around 17 minutes.

2.4 Firebase Database

Firebase is a platform for creating mobile and web applications, which was originally founded by an independent company called Envolve in 2011, and joined Google in 2014. Firebase Realtime Database was its first product, which is able to synchronizes application data across mobile devices and stores it on the Firebase's cloud. With the help of the Firebase Realtime Database, the developers can develop their applications in real-time.

Firebase launched two products, Firebase Hosting and Firebase Authentication in 2014. In 2016, Firebase launched Firebase Analytics, which expanded its services and became

a unified Backend-as-a-Service (BaaS) platform for mobile developers. Nowadays, Firebase has integrated many other Google services, such as Google Cloud Platform, AdMob, and Google Ads to offer broader products.

2.5 Related Works

Many different methods have been proposed to tackle the indoor localization problem using smartphones, which can be mainly divided into five categories: fingerprinting, trilateration, proximity estimation, visual location, and dead reckoning [10], with each category having its own benefits and drawbacks. Zhang *et al* [18] combined Received Signal Strength Indicator (RSSI) of WiFi and magnetic field as fingerprint and utilized deep learning to achieve a mean error less than 2 meters in most cases. However, the intensive data collecting phase can be extremely time-consuming, and the performance can be easily influenced by the dynamics in the environment. Compared to fingerprint based approach, proximity estimation is very efficient as it only uses the strongest RSSI to determine the location, while its accuracy can be very low [10]. Lok8 positioning system [3] is a representative example of the trilateration approach, which could achieve accuracy in centimeters using ultrasound, yet demands the environment to be unobstructed. These methods all depend on signals input from the external environment, which can be less practical as the required signal or environment may change over time.

The PDR approach counts walking steps and then multiples with the estimated step length towards the heading direction to calculate current location based on a given starting point [10] [11]. Many efforts have been made to improve the accuracy of step detection and heading direction. Rather than using a pedometer, recent researches keen on using inertial sensor readings from smartphones to detect steps and predict step length for each step. Lee *et al* [11] applied thresholds on peak and valley of acceleration values from inertial sensors to enhance the step detection. Kiran *et al* [7] designed a robust architecture which detects

the true steps when the user hold the smartphone in front of him. The authors in [8] used an empirical mode decomposition (EMD) filter to remove noise and achieved no overlaps in heading estimation. The accuracy of these methods can achieve up to 99 percent in certain situations. However, the PDR approach only works well when people are walking in certain patterns, which may lead to poor accuracy or even fail if people are crawling, climbing or going upstairs/downstairs, since the fundamental techniques in the approach, i.e., step count and step length estimation may no longer work. Therefore, it calls for new methods not depending on walking steps to break this barrier.

The Inertial Navigation System was first invented for rocket and now are used on mobile robots and vehicles. However, in reality, this approach can accumulate various IMU senor errors over time, which is known as the drifting problem [2] and can severely limit the practical usefulness of this approach for indoor localization. To this end, Lakmal *et al* [9] used filters to increase the accuracy of the approach, which was sliced into steps to avoid the drifting problems. However, the use of filters such as low pass filter and linear accelerometer sometimes may decrease the accuracy, especially when people have sudden movements, and relying on steps may also reduce the practicality. Harle [4] also stated that the INS approach is expected to maintain high accuracy when the sensors are foot-mounted and with the help of the Zero Velocity Updates (ZUPTs) method. However, similar to PDR, using steps confine the practical usefulness of INS.

Other than using filters or setting contrarians on the users' walking behaviors, there are researches focusing on applying post-experiment-processing to the trace to enhance the accuracy of an INS. Schott *et al* [13] proposed a Fuzzy Inference System (FIS) to eliminate the accumulative position errors. The mean idea of the FIS is to eliminate the drifting problem by applying a weighting coefficient ϵ to control the acceleration data, which makes the true velocity stays in a reasonable range. In the FIS, the coefficient ϵ is adjusted when the velocity information from the previous time's step is larger than the predefined thresholds. However, although the authors claimed their solution improves the accuracy by 84% when

comparing to the result without FIS assisted, their solution does not solve the nature of the drifting problem. For example, if the user's speed is constantly changing during the experiment, or the user is taking an elevator that is supposed to move at a high speed, their solution will fail because the coefficient will be miss calculated. This solution would also fail in the situation even when the user stops proceeding, the calculated velocity will keep bouncing between the thresholds.

Due to these aforementioned reasons, we carefully developed a novel indoor localization approach using our testbed to collect all the raw IMU data on smartphones to enable in-depth investigations on where the sensor and drifting errors come from and how they can affect the localization accuracy. Our findings also motivated us proposed a novel promising direction to achieve smartphone-based high accuracy indoor localization in practice.

Chapter 3

TESTBED DEVELOPMENT

3.1 Testbed Architecture

Our testbed targets to reflect all the benefits of our approach, which are real-time, smartphone only, and regardless of the user's walking patterns. A practical testbed for indoor localization should also let people whoever have permission to access the location. Therefore we use IoT to build our testbed.

Our testbed includes four segments: A. A smartphone app that is able to retrieve accelerometer data and smartphone rotation data. B. A database that stores the raw data from the smartphone and updates the location with the information from the server in real-time. C. A server that is always waiting for the database to be updated with new incoming data from the user's smartphone, do calculations and send the user's location to the database. D. An auxiliary 3D user interface that is linked to the database. When the database updates the user's location, this interface will update the user's location on the synthetic building map.

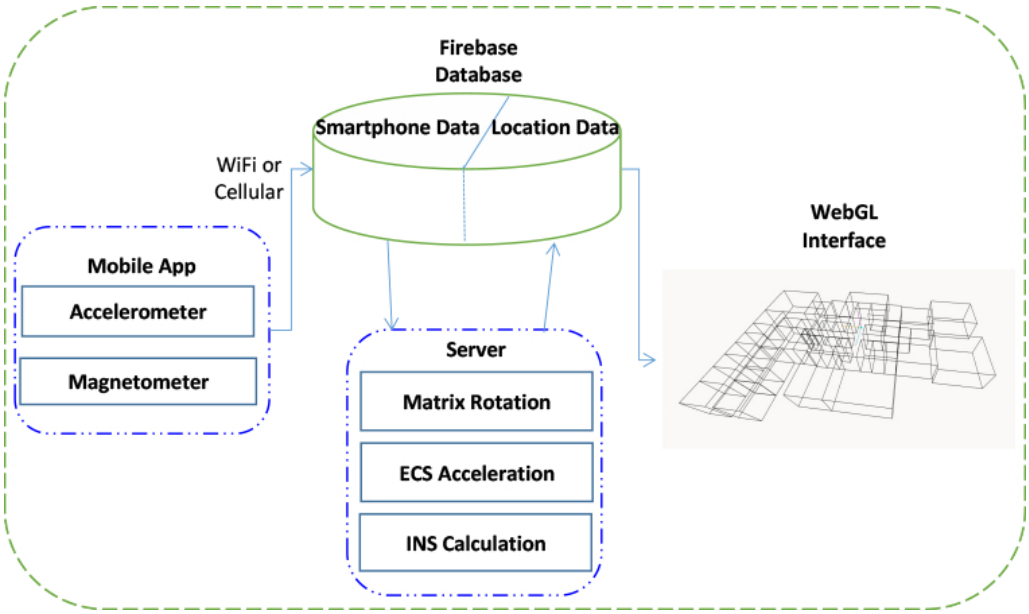


Figure 3.1. Testbed Architecture

3.1.1 Smartphone App

Smartphones nowadays are very popular, they usually own dedicate sensors and have strong processing powers. Therefore, we choose smartphones to access the acceleration and orientation raw data, and send them to the database through the internet. When the app is turned on, it will start reading the acceleration raw data and orientation data. The user can specify the database name he wants to interact with. When the user inputs the database name and clicks on start, the app app will start sending smartphone's IMU data to the database.

For our testbed, the smartphone app reads the raw IMUs' readings, which are based on the smartphone's coordinate system. It sends IMUs' readings to the database at each time the accelerometer or magnetometer senses new data. When the smartphone is not laying flat, the gravity will have contributions to all three axes of the smartphone's coordinates. We will go through how to eliminate gravity efficiently in Section 4.

3.1.2 Real-time Database

For our testbed, a database that could store and access data in real-time is necessary. After the pedestrian enters the table's name of the database and clicks on the start button, the smartphone reads the magnetometer and accelerometer data, which is synchronized at the real-time database. Another responsibility for the database is to store the location information after each time the server did the calculations. The use of the database also gives a benefit that the user's movement is traceable. Anyone who has access to the database should be able to reproduce the moving trace using the data stored in the database.

The real-time database service of the firebase allows the smartphone's application to be able to synchronize the data on the client and the database securely [6]. We use firebase for our testbed to store and access smartphone data and location information in real-time.

3.1.3 Server

The server is the core of the testbed. It undertakes the responsibility of transforming the raw smartphone data into locations. When the server is turned on, it will wait for the database to be updated. When new data arrives in database, the server will read the data from the database simultaneously. Then the server will do the coordinate transformation, transforming the acceleration data from the smartphone coordinate system (SCS) to the Geographical coordinate system (GCS). After that, the server will integrate the acceleration data on each dimension of the GCS coordinate, acquiring the velocity and location on each dimension based on GCS. At the same time, the server is in charge of examining the results. The final step for the server is to update the database with the new location after computation.

Our server is written in python. It depicts 32 figures including the top view, accelerations, velocities, and locations for the researcher to examine the results. The server is also embedded with machine learning techniques for now, and deep learning for our future work to correct the results.

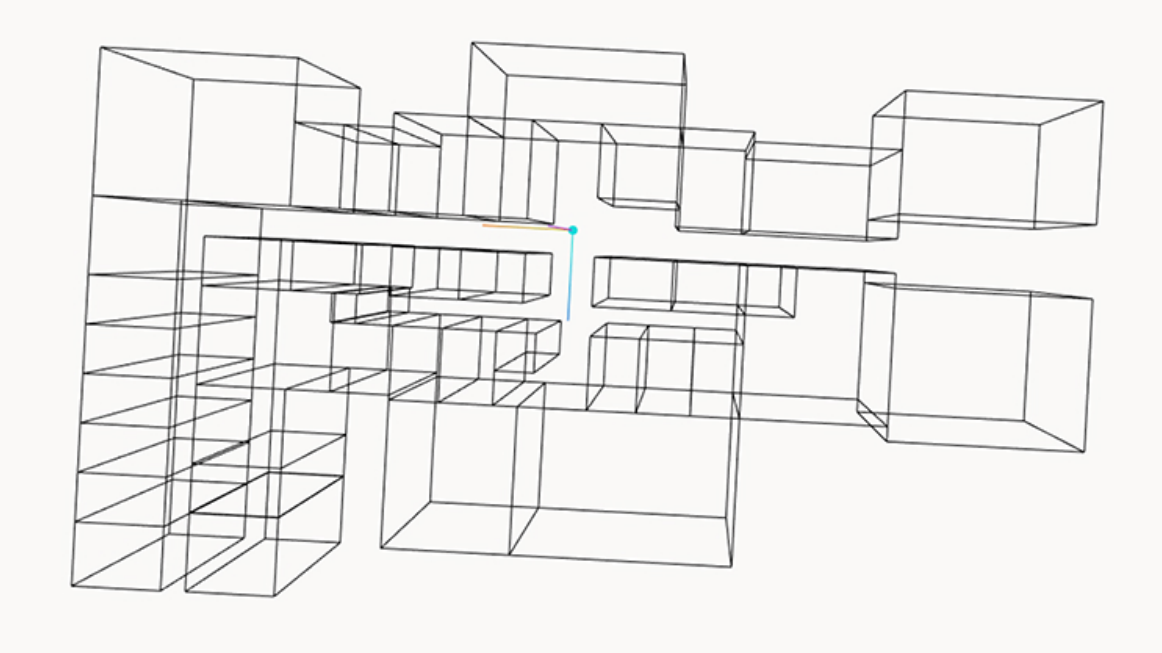


Figure 3.2. Screenshot of the WebGL Application

3.1.4 WebGL Interface

With the architecture of our testbed, not only the user who holds the smartphone knows the user's location, anyone with the access to the firebase is able to access the user's location. WebGL is such a great interface that is able to share the user's location in real-time. A synthetic 3D map of the experiment building is been rendered in the WebGL application. It does no calculation but only plots the user's location based on the firebase database's location storage. This design benefits the cases where firemen, rescue team, and people who need the outside world to share the locations of them. The screenshot of the top view of the WebGL Application is shown in Figure 3.2. The WebGL map is slightly rotated as the building itself is.

3.2 Functionality for Localization

Our testbed tracks the user's location in 3D, i.e., tracking the horizontal and vertical planes to obtain height of location, as well as latitude/longitude. Our testbed cares about the data on

the vertical plane creates additional challenges to consider. Smartphone accelerometer data is based on its own coordinate system, so if the user does not constantly hold their smartphone at a zero slope angle, the acceleration data on the GCS will be incorrect. However, it is not practical for users to constantly keep their smartphones oriented this way. Moreover, the further a user moves while holding their smartphone at an angle, the more errors accumulate to produce a very inaccurate location. Therefore, it is crucial to transform the smartphone coordinate system into the GCS to get accurate acceleration data.

3.2.1 Coordinate Transformation

Previous research has proven that coordinate system transformation is a feasible solution to produce accurate GCS acceleration data [17] [19]. The only data needed to perform the transformation are three rotation angles in the smartphone coordinate system that can be easily retrieved from the smartphone's magnetometer sensor: Pitch(θ), Roll(ϕ), and Azimuth(ψ). To understand these rotation angles, consider the smartphone is lying perfectly flat with its y-axis (the top of the phone) pointing north. This is the initial stage, where[θ, ϕ, ψ] = [0, 0, 0]. When the smartphone is rotating around the x-axis, y-axis and z-axis, rotation angles will be indicated for the roll, pitch and azimuth values, respectively. Using a magnetometer instead of a gyroscope has many benefits. First, The magnetometer is able to determine orientation with respect to the earth's magnetic fields without an initial reference. This means it does not need an initial stage, but can start the localization procedure wherever the smartphone is. Another benefit is that gyroscope sensor is difficult to calibrate after a certain period and can have accumulative error, while magnetometers are easily calibrated by the magnetic fields. However, it is important to note that there are some limitations of using a magnetometer. First, it is susceptible to the influence of magnetic fields. Also a gyroscope is more accurate for a shorter duration. However, we chose a magnetometer for the purpose of long-term accuracy. The experiments in Section 4 proves the robustness of using it as the indicator of the smartphone's orientation.

To calculate the GCS acceleration data, we use the following rotation matrix at the t^{th} sampling:

$$R_x(\theta_t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta_t & sin\theta_t \\ 0 & -sin\theta_t & cos\theta_t \end{bmatrix} \tag{3.1}$$

$$R_y(\phi_t) = \begin{bmatrix} cos\phi_t & 0 & sin\phi_t \\ 0 & 1 & 0 \\ -sin\phi_t & 0 & cos\phi_t \end{bmatrix} \tag{3.2}$$

$$R_z(\psi_t) = \begin{bmatrix} cos\psi_t & sin\psi_t & 0 \\ -sin\psi_t & cos\psi_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

The rotation matrix will be:

$$R_t = R_z(\psi_t)R_x(\theta_t)R_y(\phi_t) \tag{3.4}$$

Note that the sequence is important due to the special feature of the sensor, Azimuth has to be multiplied firstly.

3.2.2 Acceleration Matrix

As discussed in Section 3.1, the accelerometer reads the raw acceleration data with gravity. This means the reading includes gravity on all three axis.

The key point of our proposed method is that we then apply the smartphone's accelerometer matrix a_t^{scs} to the rotation matrix R_t to get the acceleration matrix of the earth, a_t^{GCS} at time t respectively.

$$a_t^{GCS} = R_t a_t^{scs} - g[0, 0, 1]^T \tag{3.5}$$

Where g is the gravity. By applying Equ. 3.5, we get the acceleration matrix a_t^{GCS} on the GCS coordinate at time t without gravity.

3.2.3 Integration Method

In the real world, pedestrian's movement is caused by continuous acceleration. On the dimension d ($d \in x, y, z$), the velocity V^d is integrated by the acceleration, and the location L^d is the integration of the velocity. Assume that the pedestrian's movement starts from static, and his/her velocity and location at time t can be calculated as:

$$V_t^d = \int_{t'=0}^{t'} a_{t'}^d \cdot dt' \tag{3.6}$$

$$L_t^d = \int_{t'=0}^{t'} V_{t'}^d \cdot dt' \tag{3.7}$$

However, smartphone sensors do not provide continuous location data and only detect location at specific time intervals, changing the method of calculating a pedestrian's location.

We compared two possible methods to calculate the location from the acceleration. The first way is intuitive: To get the inertial-frame velocity V^d and the location L^d on the dimension d at $(t+1)^{th}$ sample, we assume the acceleration and velocity is constant in the period between the t^{th} sample to the $(t+1)^{th}$ sample. Then we use the following equations to get the velocity V_{t+1}^d and location L_{t+1}^d at the sample time t+1:

$$V_{t+1}^d = V_t^d + a_t^d \cdot \Delta t, \tag{3.8}$$

$$L_{t+1}^d = L_t^d + V_t^d \cdot \Delta t. \tag{3.9}$$

where a_t^d is the accelerometer data reading on the d^{th} dimension at time t, and Δt is the interval between time t and time t+1. In this approach, none of the velocity or the loca-

tion values are calculated by integration. To find the velocity change at each interval, this method treats the pedestrian's movement as a uniformly accelerated rectilinear motion, and to find the location change at each interval, it treats movement as a uniform rectilinear motion. However, questions about error in calculating location arose after observing that the transformed GCS acceleration data slope is not constant, but fluctuates.

Therefore, we proposed a second method that treats the pedestrian's movement as a uniformly variable accelerated motion in each interval, using the following functions to calculate location:

$$V_{t+1}^d = V_t^d + \int_t^{t+1} F_1 \cdot dt \tag{3.10}$$

$$L_{t+1}^d = L_{t+1}^d + \int_t^{t+1} F2 \cdot dt \tag{3.11}$$

where,

$$F1 = a_t^d + (a_{t+1}^d - a_t^d)/\Delta t \cdot t \tag{3.12}$$

$$F2 = V_t^d + 0.5(a_{t+1}^d - a_t^d)/\Delta t \cdot t^2 + a_t^d \cdot t \tag{3.13}$$

We compared these two methods using synthetic data, and found that treating the movement as a variable accelerated motion at each interval produces more accurate results. Results of further experimentation on these two methods will be further discussed in Section 4.

Chapter 4

TESTBED EVALUATION AND VALIDATION

All experiments except the synthetic experiment in 4.1 Synthetic Experiment, were executed in the computer science department building of our university. The building is positioned at an angle of 15 degrees clockwise from North. As shown in Figure 4.1, the long (horizontal) corridor is 38 meters, and the short (vertical) corridor is 20 meters. The smartphone used was a Huawei Mate 20, version 10.1.0.165.

4.1 Synthetic Experiment

In order to determine which method was most accurate to use for our testbed, we tested both methods by simulating pedestrian acceleration data in the form of a sine curve, in one dimension. We chose the sine curve after observing the real-world pedestrian acceleration data graph resembles a sine function, in all three dimensions. We assumed the amplitude of our sine curve was between $3m/s^2$ to $15m/s^2$, depending on whether the smartphone was held in the front of the user or swinging in hand. In each dimension, we assumed the synthetic acceleration equals 3sin(kπt) and 15sin(kπt) to simulate the smartphone held in

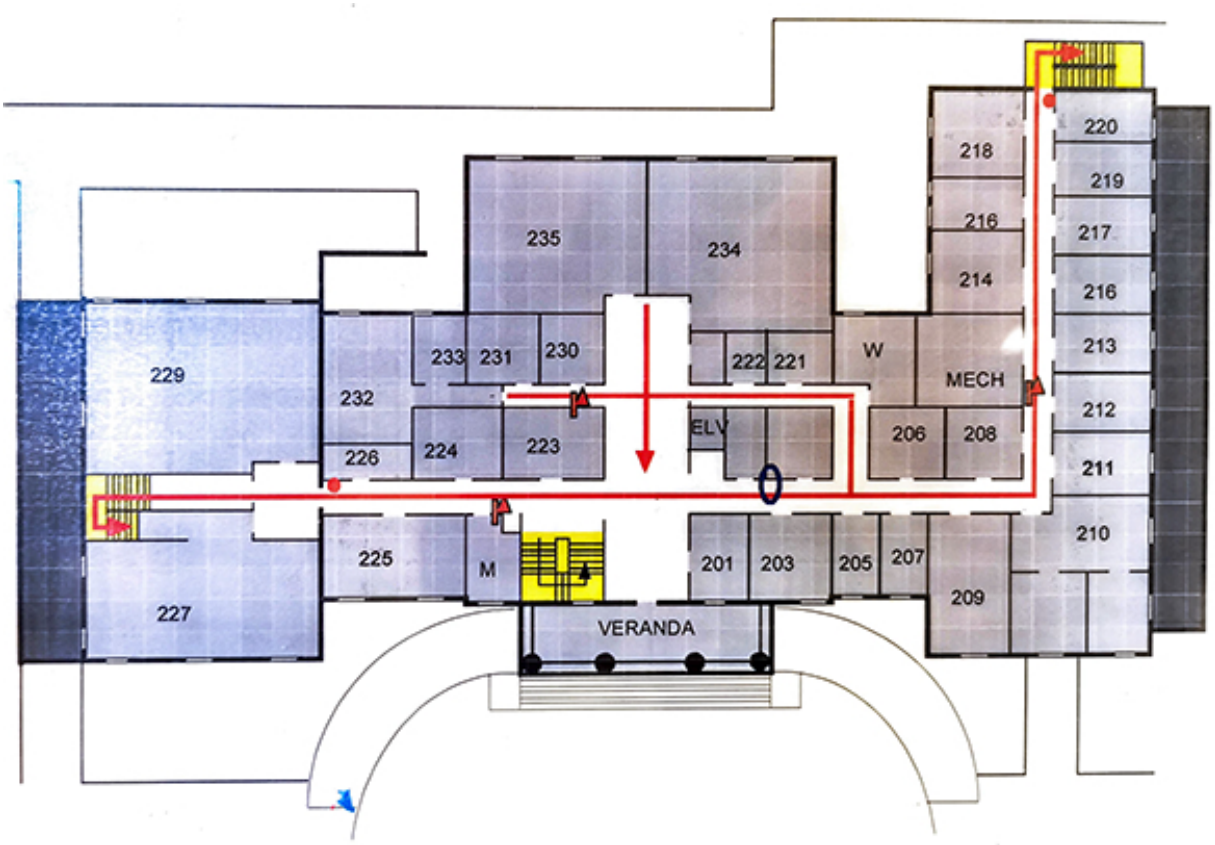


Figure 4.1. The Top View of the Experiment Building

Table 4.1. Synthetic Experiment Results

interval(s)	0.1	0.05	0.02
method 1 error rate(%)	2.3633	0.8291	0.1932
method 2 error rate(%)	0.8521	0.2179	0.03409

Table 4.2. Amplitude: $3\text{m}/s^2$, 1 step per sec. Smartphone in the front, walk slow

interval(s)	0.1	0.05	0.02
method 1 error rate(%)	2.3911	0.7854	0.2256
method 2 error rate(%)	0.8280	0.2131	0.03383

Table 4.3. Amplitude: $15\text{m}/s^2$, 1 step a sec. Smartphone on hand, walk slow

interval(s)	0.1	0.05	0.02
method 1 error rate(%)	9.0464	3.8167	0.7282
method 2 error rate(%)	3.5401	0.8494	0.1347

Table 4.4. Amplitude: $15\text{m}/s^2$, 2 step a sec. Smartphone on hand, walk fast

front of the user and swinging in hand, respectively. kπ is used to determine the speed of the user.

As previously mentioned, the sensor provides data at certain intervals, and the length of these intervals can vary up to 20%. Since previous experimentation found the interval was a constant $C \pm 20\%C$, we assumed this interval for method testing. We ran the test 100 times using the following method to find the error rate for these two methods.

$$ER = (|L_g - L_r|)/L_g \cdot 100\% \tag{4.1}$$

where L_g is the ground truth location and L_r is the resulting location calculated by the approaches. The results are shown in Table 4.1.

Table 4.1 shows the second method is better than the first method in all conditions. As even small errors accumulate and have a greater effect on the final result, we implemented the second method and selected a sample interval of 20 milliseconds. This experiment proved that if the smartphone sensor is accurate, our testbed yields over 99.8% accuracy, no matter how the user holds the smartphone.

However, in reality smartphones experience sensor errors and noise. The following experi-

ments use the real-world sensor data, which is affected by sensor errors and noise.

4.2 Real World Experiments

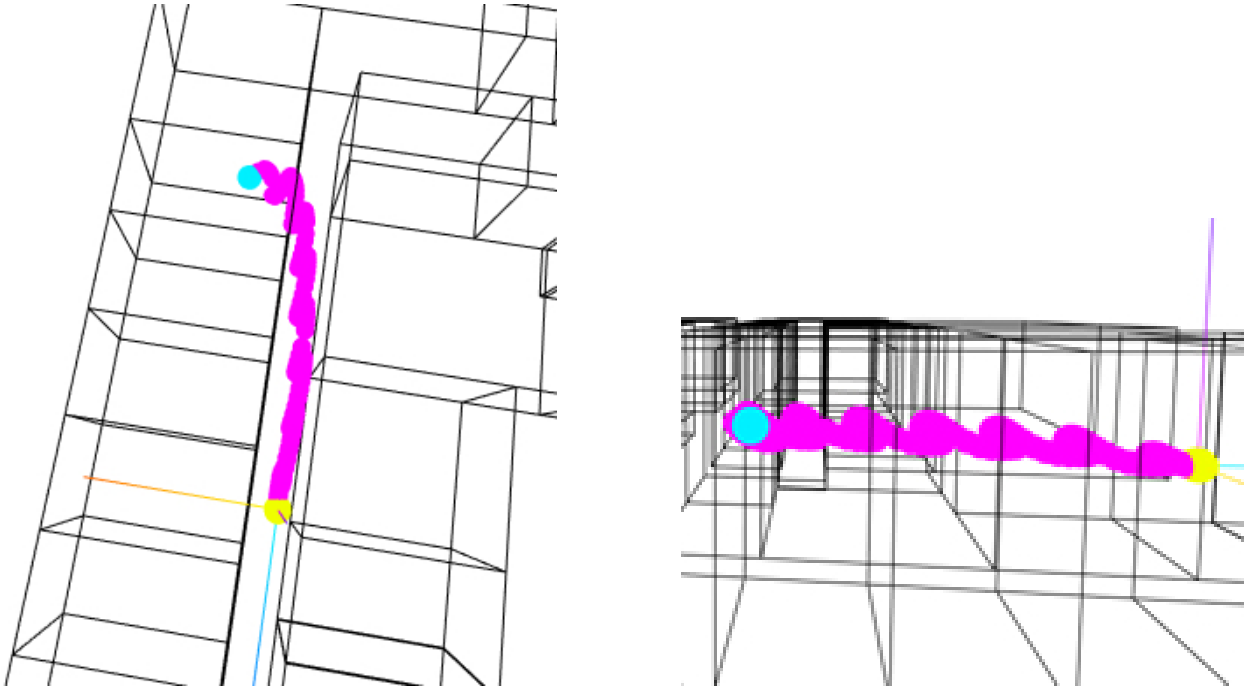
4.2.1 Overall Verification

To verify the overall performance of our testbed, we let the researcher hold the smartphone and walk inside the experiment building. We found our testbed is capable of real-time tracking of the user's movement. One trace of the walking experiments is shown in Figure 4.2. In this experiment, we set the starting point, shown as the yellow sphere, in the short corridor, where is 10 meters to the intersection to the long corridor. The walking trace is depicted as the purple orbs, and the blue sphere shows the researcher's current location(final location). The real-time tracking can show the smartphone in hand swing back and forth with each step in the top view figure, up and down with each step in the side view figure. The drifting problem can be noticed in the figures as well. This data set has been saved, and will be further examined in the walking experiments in Section 4.2.3.

4.2.2 Circle Evaluation

The greatest advantage of our proposed method is that it does not require regular step patterns, allowing for usage in scenarios where the user is not walking. We conducted some experiments on the testbed where the user used irregular footsteps to spin in a circle, bring the smartphone back as close to the starting point as possible.

Figure 4.3 shows the result of one of the circle experiments. In Figure 4.3a, the red dotted line shows the movement of the user. A circle pattern is shown, but the ending point does not overlap with the starting point. Figure 4.3b shows the acceleration in GCS. Figures 4.3c and 4.3d show the GCS velocities and locations, respectively. Velocities on the horizontal plane resemble Sine and Cosine curves. However, neither final velocity on the x



(a) Top View. (b) Side View

Figure 4.2. Overall Verification

or y-axis go back to zero at the end of the experiment.

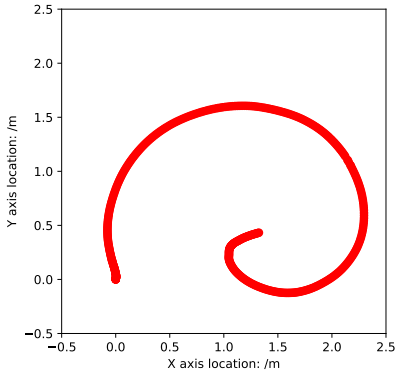
4.2.3 Walking Experiments

Since walking is the most common way for humans to move inside buildings, we did multiple experiments for different walking patterns.

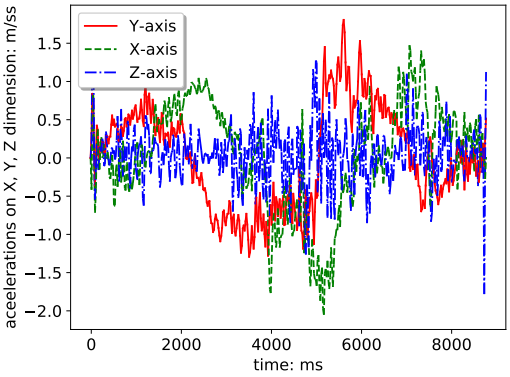
4.2.3.1 Smartphone in the Front

In this trial, the user walked 10 meters down the short corridor at an angle 15 degrees East from North.

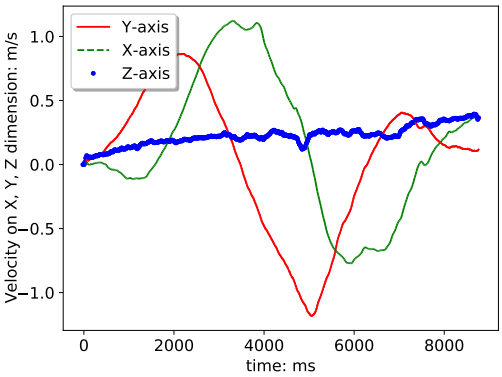
In Figure 4.4, the top view graph initially shows the user's movement in the correct direction and then immediately drifts in the wrong direction. The final location is 6.79 meters, with -6.56 meters on the x axis and -1.75 meters on the y axis, where the ground truth distance is 10 meters. Therefore the error rate is actually more than the 32.06% rate read by the server, because due to the aforementioned drifting problem, the direction is also



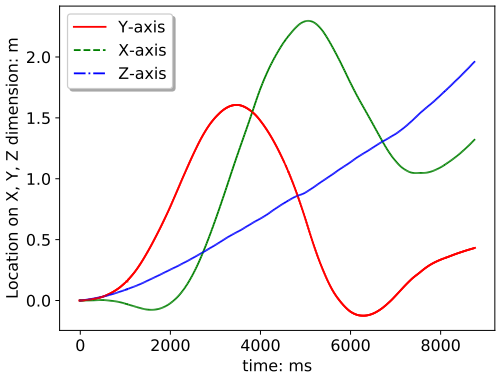
(a) Top View



(b) Accelerations on GCS

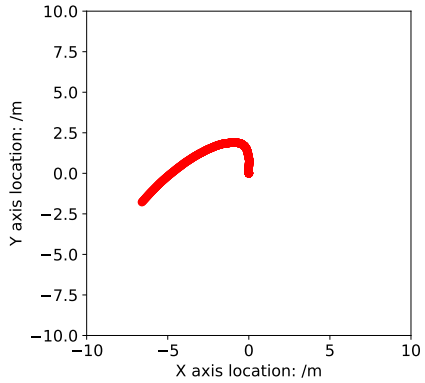


(c) Velocities on GCS

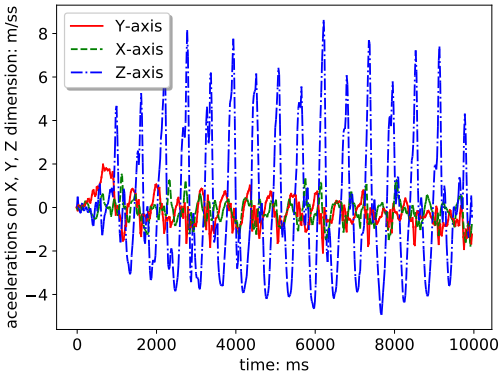


(d) Locations on GCS

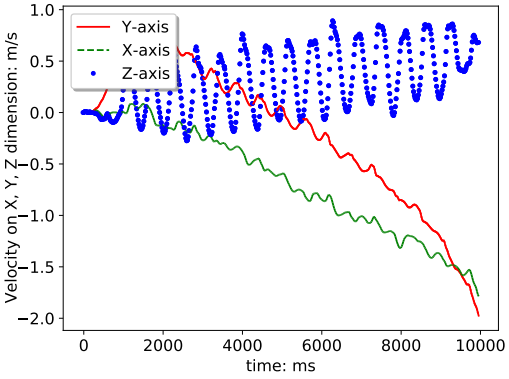
Figure 4.3. Circle Experiment



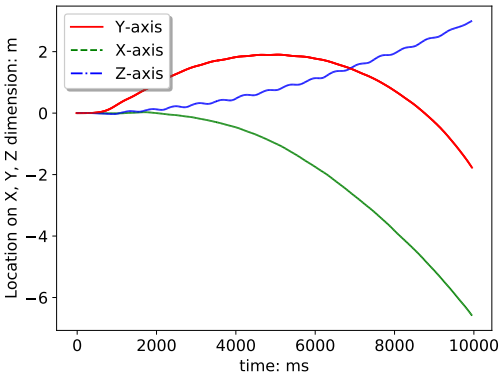
(a) Top View



(b) Accelerations on GCS

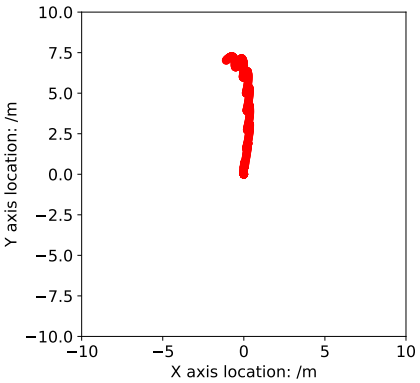


(c) Velocities on GCS

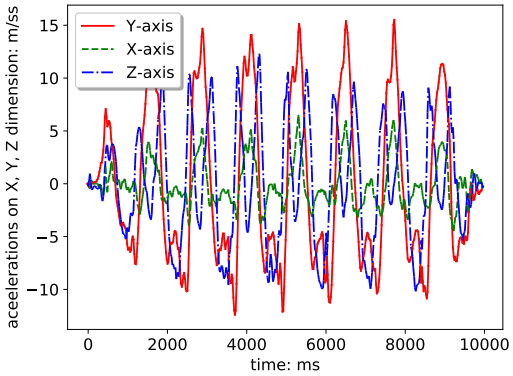


(d) Locations on GCS

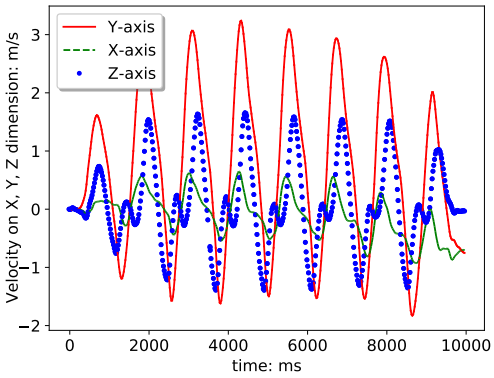
Figure 4.4. Walking 10 Meters, Smartphone Held in the Front



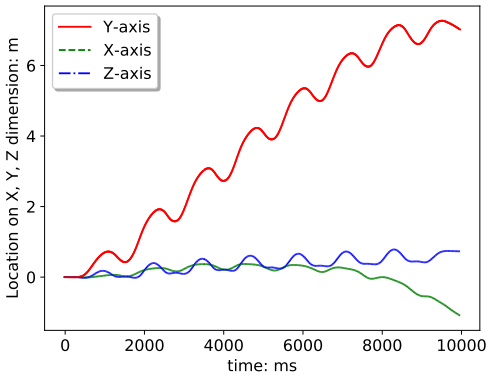
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



(d) Locations on GCS

Figure 4.5. Walking 10 Meters, Smartphone Swing in Hand

incorrect.

Figure 4.4b shows the acceleration on the GCS. Figure 4.4c indicates all the velocities have clear patterns of swinging on linear lines with different slops, where the actual velocities should be straight lines swinging up and down on a constant value. This lead to the locations on GCS in Figure 4.4d drift with time. This is a good example of how INS method in real-world always introduces a huge drifting problem.

4.2.3.2 Smartphone Swinging in Hand

In this experiment, the user held the smartphone swinging in hand, walking toward the same direction for the same distance as the previous experiment.

The top view graph shows less of a direction drifting problem compared to the user holding the smartphone in front. Figure 4.5b and Figure 4.5c both show wave graphs from this movement. However, the final calculated location is 7.14 meters, with -1.11 meters on x-axis location and 7.05 meters on the y-axis. Since the ground truth is 10 meters, the error rate is 28.58% without any calibration. This dataset has less direction drift compared to that of having the smartphone in the front, but it is still far from accurate. The peak values are progressively lower and lower. Figure 4.5d shows the location goes forward and comes backwards with each step and swing of the hand.

4.2.3.3 Make a Turn after Walking Straight

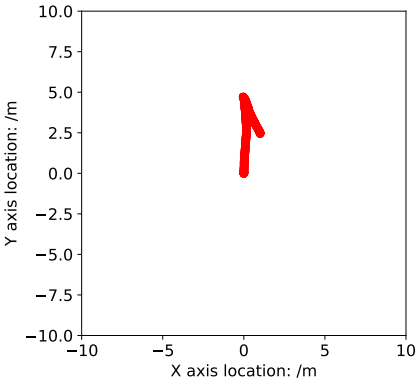
In this experiment the user held the smartphone while walking down the short corridor and then making a right turn into the long corridor. The starting point is 5.5 meters from the intersection of the two corridors and the ending point is 3 meters from the intersection.

In Figure 4.6, the top view walking movement shows a "right turn," but the turning angle is smaller than the actual turning angle, which is shown at second 6 by GCS velocity. Figure 4.6b shows the accelerations after transformation into GCS. In the velocities graph at 4.6c , the velocities on the X and Y-axis drift differently before and after the turn, while the velocity on the Z-axis remains at the same drift throughout the experiment. Figure 4.6d illustrates the why top view goes down after the turn.

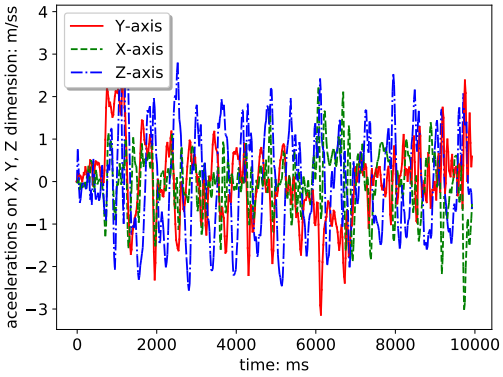
4.2.3.4 Walking in Place

Unlike PDR, our testbed does not require step counting or step length information, which allows it to detect walking in place. In the experiment, the user picked the smartphone up from a table and walked in place for 9 seconds, swinging the smartphone in hand, and then put the smartphone back down on the table. The result is shown in Figure 4.7.

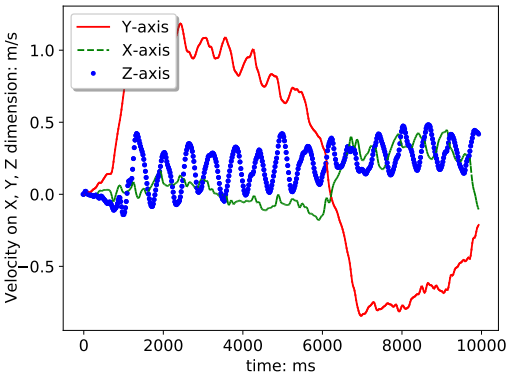
Figure 4.7a shows the top view of walking in place drifts dramatically after the first step. Figure 4.7b shows the accelerations on GCS. The velocity graph in Figure 4.7c shows



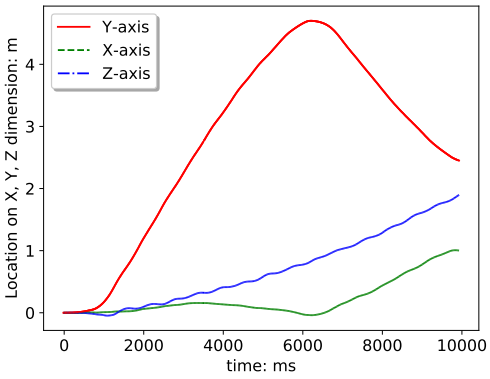
(a) Top View



(b) Accelerations on GCS

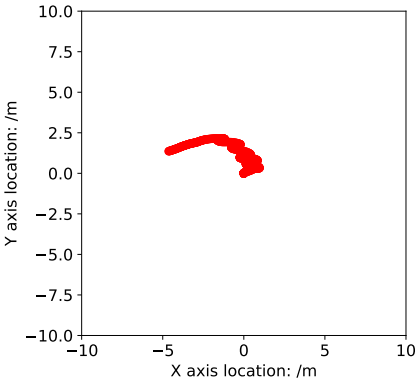


(c) Velocities on GCS

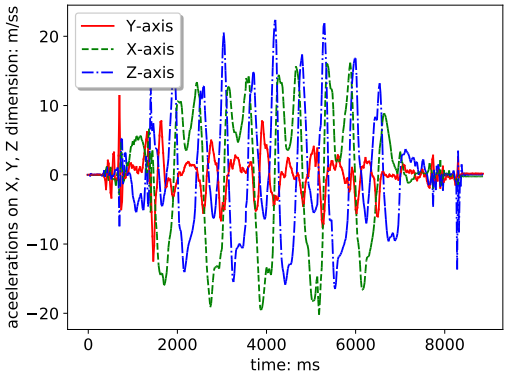


(d) Locations on GCS

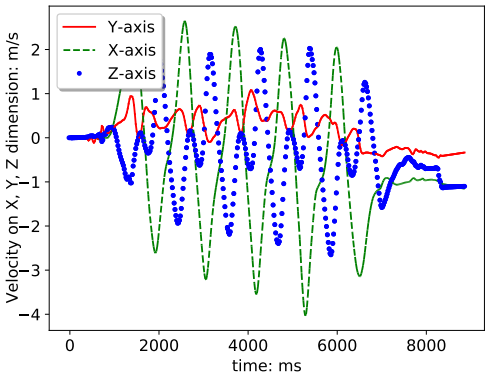
Figure 4.6. Make a Turn, Smartphone in the Front



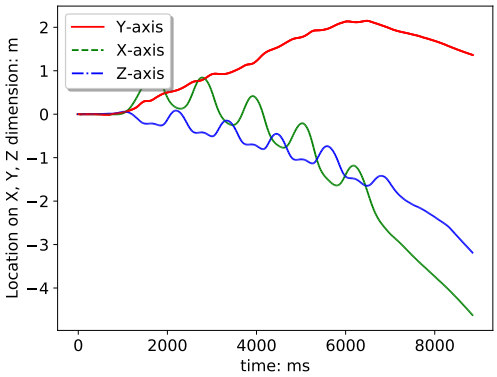
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



(d) Locations on GCS

Figure 4.7. Stepping in Place, Smartphone Swing in Hand

the velocities are decreasing at a constant rate. The true location graph in Figure 4.7d shows the location of the smartphone does not go back to the starting point after step 1, and the location drifts further with each step.

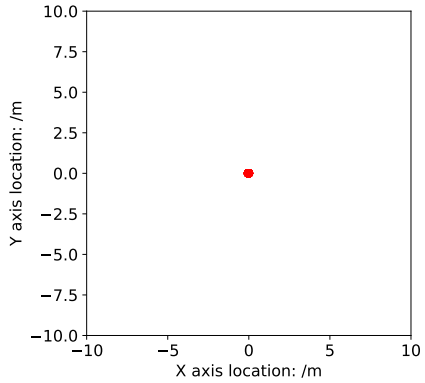
4.2.4 Static Evaluation

From these walking experiments, we found the velocity graphs all display linear pattern and the location graphs display a quadratic pattern. To verify our findings, we conducted multiple experiments to test the results when the smartphone was been placed statically. We placed the smartphone on the flat ground, with different tilt angles. One of the results is shown in Figure 4.8.

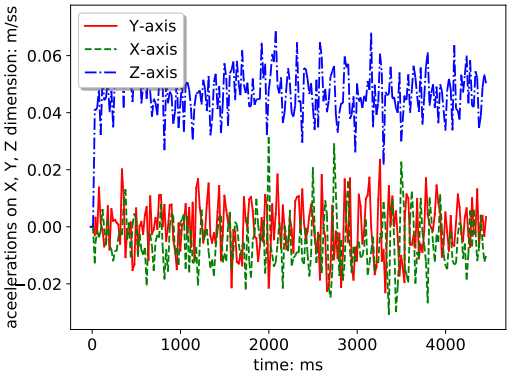
Once again, the results proved our assumption that the velocities have a linear pattern and the locations have a quadratic pattern in each experiment, even when there is no movement. Figure 4.8b shows the accelerations on the X and Y-axis swing around 0. However, the acceleration on the Z-axis swings around 0.04m/s^2. This is because GCS Z-axis acceleration readings have fluctuating levels of sensor errors. We subtracted a value of -9.79632m/s^2 to help eliminate gravity, but because the value of the sensor errors fluctuate, we could not consistently eliminate all errors. Figures 4.8c and 4.8d show more clear patterns of the drifting problem of the velocities and locations.

4.3 Interesting Observations

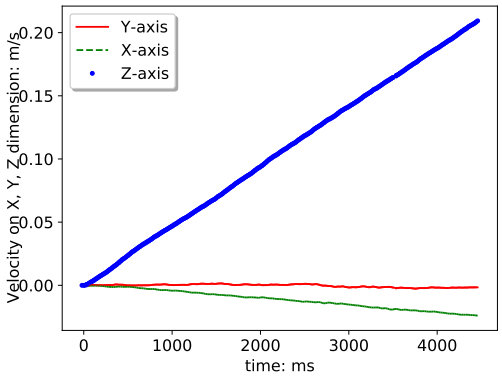
Through our testing and experimentation, we have found that velocities drift in linear patterns and the locations graphs drift like quadratic curves. However, when we use synthetic data to generate random noise, the graphs show irregularities. This means the drift in each dimension is caused by a constant-level of error in the acceleration after transformation to GCS. The square root after and before transformation to GCS remain the same. We found that the square root of the three accelerations were not always the same even when the



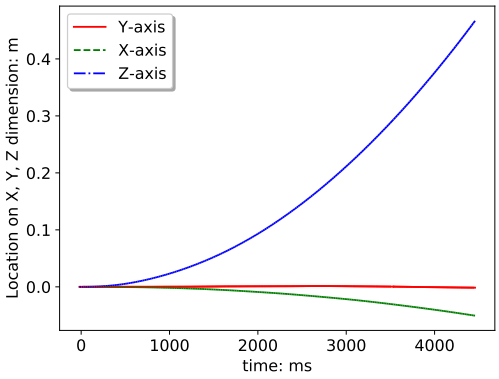
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



(d) Locations on GCS

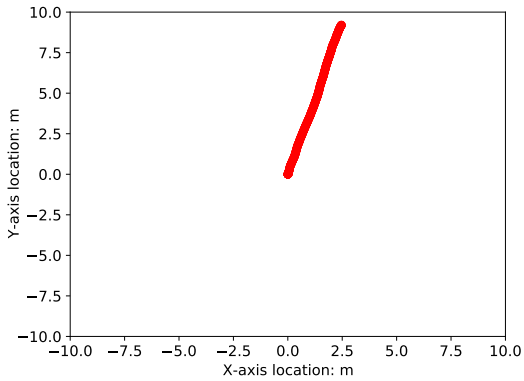
Figure 4.8. Static on the Flat Ground, Smartphone in the Front

smartphone was placed statically. The square root of the accelerations is between 9.61m/s^2 to 9.87m/s^2 with different orientations, while the gravity of the experiment location should be 9.796m/s^2. Because we compensated for the effect of gravity on the Z-axis acceleration after transformation to GCS, the sensor errors will likely maintain a constant value on the Z-axis. The accelerations on the X and Y-axis do not require compensation, so they are more accurate. However, like the acceleration on the Z-axis, they will also be influenced by the sensor errors.

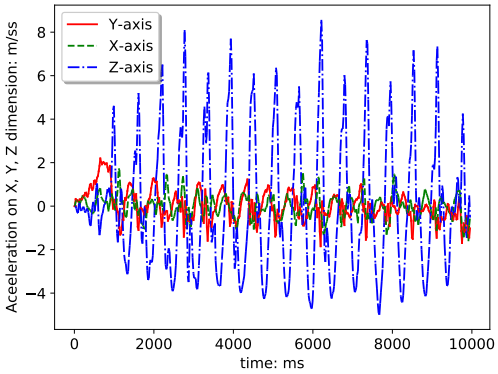
In the final results, it is evident that raw smartphone data can somewhat show user movement, but fails to locate the user precisely. Without any calibration, the drifting problem is too severe to achieve high accuracy. Although the drifting problem may vary case by case, our testbed identifies that there are distinct patterns to the drifting problem in each experiment. To prove our findings, we applied the "Final Velocity to Zero"(FVZ) method to the circle experiment, and our testbed drew an almost perfect circle. We applied linear regression to the walking experiments, and our testbed achieved over 98% accuracy in both data sets. The drifting problems for walking in place and static smartphone placement experiments can also be greatly eliminated by applying a linear regression. Both FVZ and linear regression can compensate for over 90% of location drifts when the smartphone is placed statically. These results show our testbed's drifting problem can be almost completely compensated for or even eliminated with the help of our finding. It proves our testbed is very effective at indicating sensor errors and is very promising for future indoor localization solutions.

4.4 Calibration Technique for Walking Straight

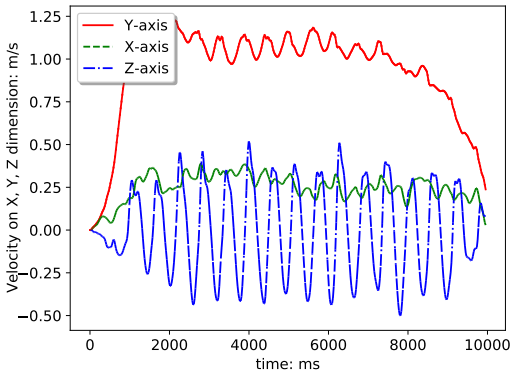
Motivated by our observations, we implemented linear regression on the velocities on each dimension. Linear regression technique finds the drifting rate of the velocities on each dimension, which indicates the acceleration error on the GCS. We then subtract the drifting



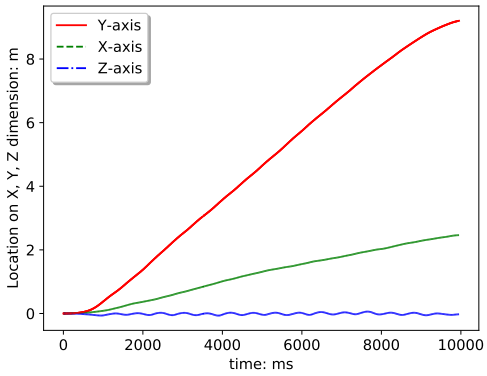
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



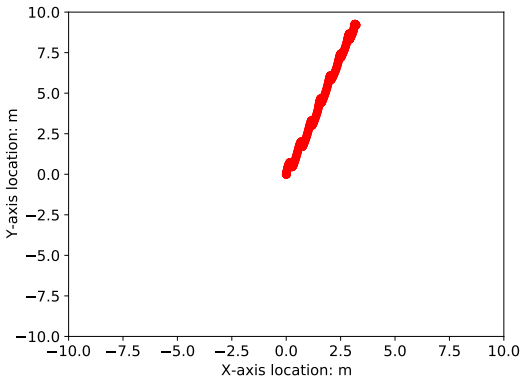
(d) Locations on GCS

Figure 4.9. Calibration Results for Walking with Phone in the Front

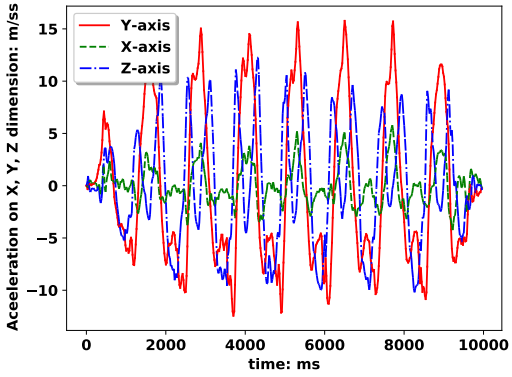
rate from the acceleration on each dimension to get the calibrated accelerations.

Calibration results of using linear regression on the experiment of walking with a smartphone in the front are shown in Figure 4.9. The top view in Figure 4.9a shows the walking direction maintains a straight line toward the correct direction. The distance calculated after calibration also shows great improvement as high as 9.6 meters, which is only 4% to reach the ground truth. The velocities in Figure 4.9c show the speed on the x, y axis swings on a constant value, which is also in line with reality. The greatest finding is that by looking at the location graph in Figure 4.9d, all the locations on the z-axis stays at zero while bouncing up and down with the user's walking steps.

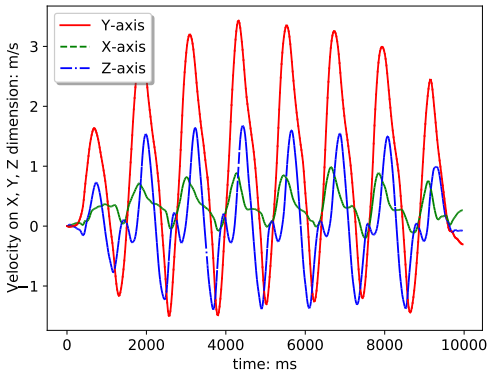
The experiment of walking straight with a smartphone in hand and swinging is shown



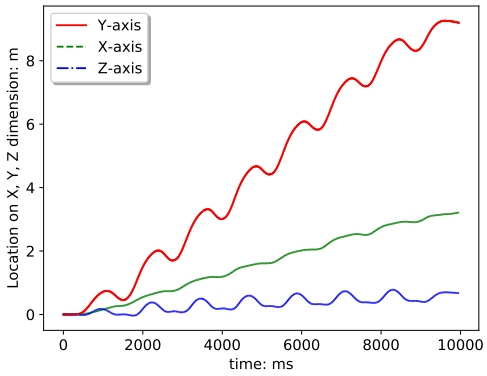
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



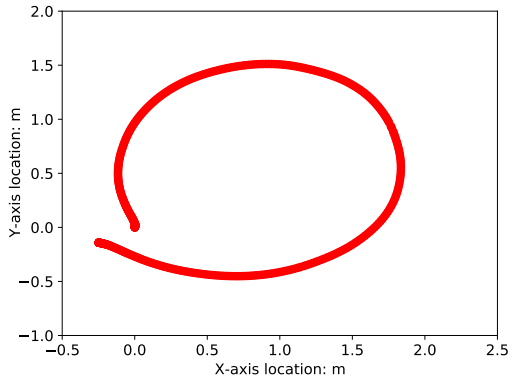
(d) Locations on GCS

Figure 4.10. Calibration Results for for Smartphone in Hand Swinging

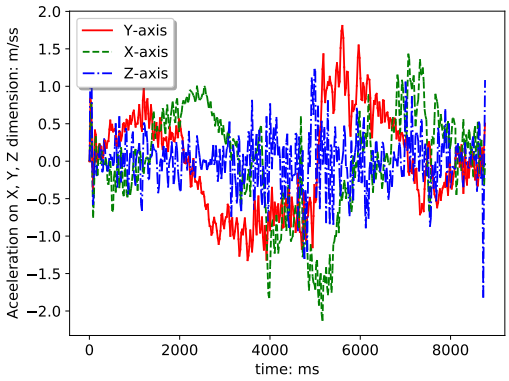
in Figure 4.10. The top view in Figure 4.10a also shows the user walks in the correct direction. The distance is measured as 9.8 meters, which is even higher than walking with the smartphone in the front. This is because when the smartphone is been held in hand, the walking pattern of every step is more periodic comparing to the smartphone in the front, where the accelerating section and slowing down section is different than the stable speed section. The velocities in Figure 4.10c are beautifully sticking to the reality which has a higher amplitude than the smartphone in the front. The locations are shown in Figure 4.10d. The z axis's location shows the user swinging hand with a high amplitude, while the locations on the x and y axis show the smartphone goes backward after each time the smartphone goes forward, which also sticks to the reality.

4.5 Calibration Technique for Spinning a Circle

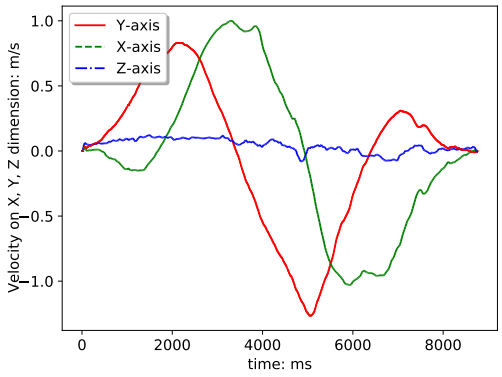
In the spinning a circle experiment, as a known fact, the smartphone's initial and final velocities are static. However, we found the velocity graphs generated by the raw smartphone data are not static when the experiment end. Inspired by the known conditions, we got the drifting rate from the final velocities calculated by the raw smartphone data. Figure 4.11 shows the calibrated result for the circle experiment. As shown in the velocities figure in 4.11c, this calibration technique is the FVZ method which forces the final velocity to go back to zero to find the drifting rate, then the drifting rate is also deducted from the recorded acceleration data for calibration. With FVZ method, the top view shown in Figure 4.11a shows an almost perfect circle.



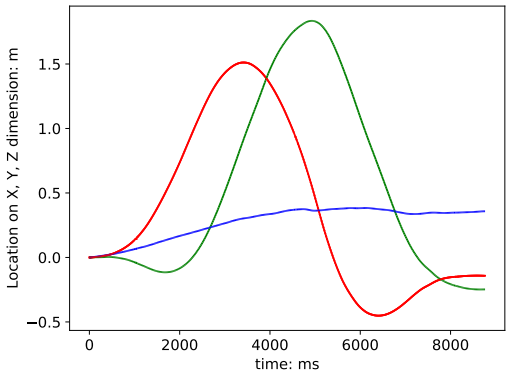
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



(d) Locations on GCS

Figure 4.11. Calibration Results for Circle Experiment

Chapter 5

A NOVEL ENHANCEMENT APPROACH

5.1 Main Idea

As discussed in the previous chapter, we validated our testbed and found the drifting problem shows clear patterns through extensive experiments. The experiment results show the drifts always have linear orders for velocities and quadratic orders for locations when walking straight. Therefore, the accuracy of the walking straight problem can be greatly improved by subtracting the drifting rate from the accelerations on each dimension, where the drifting rate can be calculated by applying linear regression on the velocities of each dimension. When orientations are changing, we can apply the known conditions of the velocity before and after the orientation change to reduce the sensor errors. However, in the real world, a typical walking trace of a pedestrian is not simply just walking straight or spinning a circle. It needs a new strategy to track a continuous walking trace that combines multiple different walking behaviors. Motivated by our observations, we proposed our novel indoor localization approach which only uses raw smartphone data.

To come up with a new strategy for indoor localization, we converted the problem

into solving it in a simple and typical scenario, where the user walks on a 2D plane. The user starts by walking straight in a corridor, then makes a turn(MaT), and ends up walking straight again for a certain distance in another corridor. Since the pedestrian's walking trace can be broken down into several repeated traces as in our typical scenario, as long as we can get an accurate result for this typical scenario, we prove our novel indoor localization approach is successful.

Since walking straight and orientation change behaviors suit different calibration techniques to be highly accurate, our novel approach needs to recognize the user's walking behavior and treats different walking patterns with different calibration techniques. To make our approach be able to recognize then segregate the user's walking behavior automatically, we built an LSTM RNN and trained it with thousands of data from several traces of the typical scenario. For the walking straight sections before and after the turn, we implemented linear regression on the velocity to find the drifting rate, then subtract the drift rate from the accelerations to eliminate the drifts. For the turn section, similar to the speeds that are static before and after the experiment in the spinning a circle problem, the true velocity calculated by the square root of the velocities on the x and y axis should be the same before and after making the turn. If we can prove the drifting problem shows the same drifting rate on the x and y axis when the orientation is changed, we can then separate the drifting rate equally on the x and y axis to reduce the drift on the make a turn section. After all these processes, we can then compare the calibrated result to the original result. The complete workflow for our approach is shown in Figure 5.1.

5.2 Walking Behavior Segmentation

In order to treat different walking patterns with their best calibration techniques, the first thing we need to do is to recognize and segregate the walking trace into different walking patterns. To segment the walking behavior of the user, we designed a Bidirectional Long-

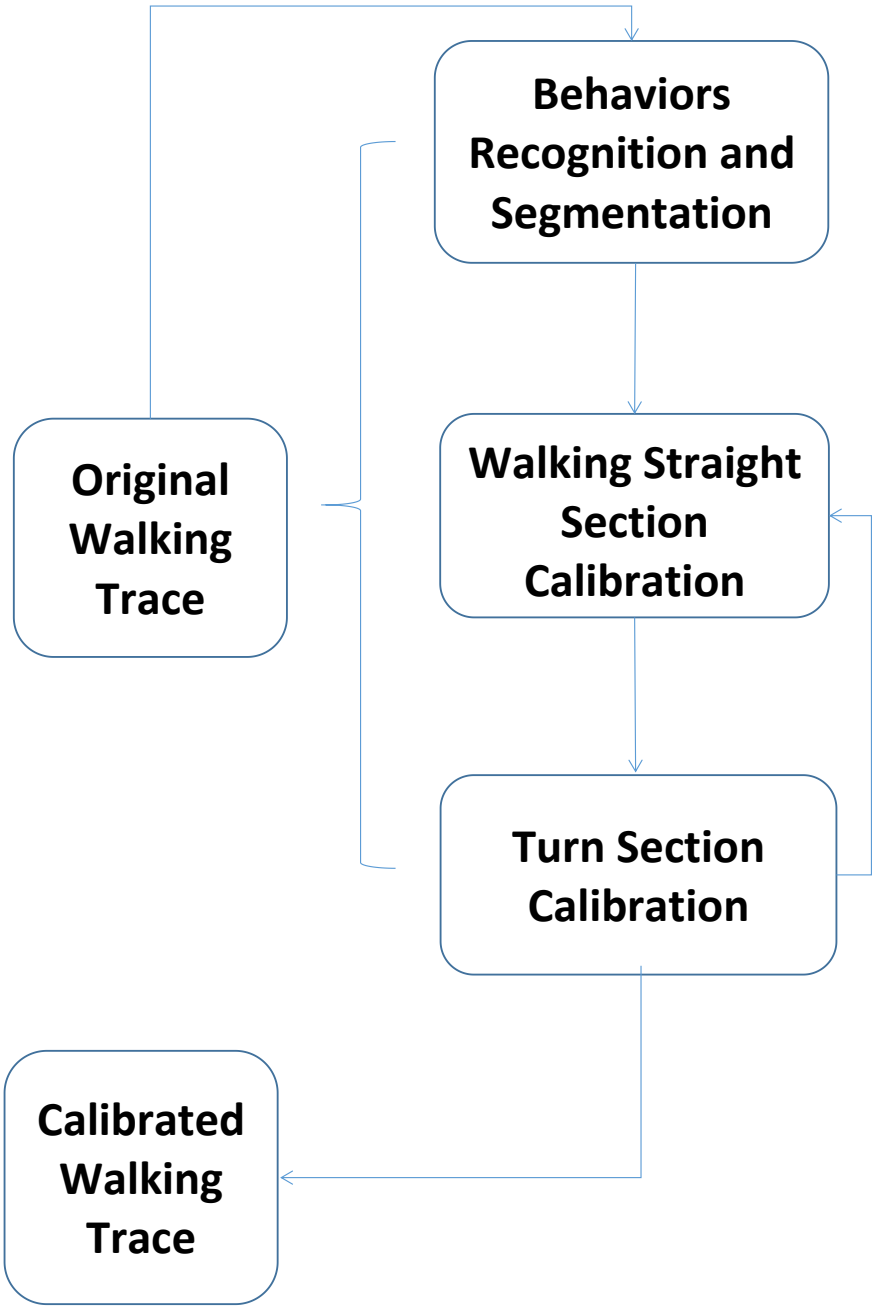


Figure 5.1. Flowchart of Our Novel Solution

Short-Term-Memory (LSTM) Recurrent Neural Network (RNN). Since the data sequentially update our testbed platform, we think RNN is the best deep learning model to segment walking behaviors because RNNs are best at modeling sequential data.

By observation, we found the "turn" sections in all experiments were around 2 seconds while walking straight sections various case by case. We chose the time window as 2 seconds to let the "turn" section falls in an entire time window as much as possible. For each experiment, we marked the "turn" section by observation. We then got time windows every 20 milliseconds from the start of the experiment until 2 seconds before the experiment ends. For each time window, if the time window does not fall in the "turn" section, we label it as zero. If the time window falls in the "turn" section, we label it as the portion that falls in the "turn" section. For example, if the time window has 0.5 seconds falls in the "turn" section where the turn section is 2 seconds, the label of it will be 0.25. If the time window has 1 second falls in the "turn" section, its label will be 0.5. We conducted the MaT experiment 10 times and used 8 of them as training samples, and 2 of them for testing samples.

We chose 5 features for the input layer: the calculated velocities on the x, y axis, and three orientation data read from the smartphone. We used these 5 features as the input layer to train and test the bi-LSTM model and get over 95% accuracy for recognizing the walking behavior. The training and testing results are shown in Figure 5.2. More features such as locations on the x, y axis and all three axes' acceleration data worth to be tested for future works.

5.3 Orientation Change

After segmentation, we intend to treat each walking pattern with its best calibration technique to get the highest accuracy. In the previous section, we have proved the walking straight problem can be greatly improved using linear regression, the only section needed to be calibrated is the turn section. However, since the user's moving speed does not go

```
mnist_train 3330
mnist_test 398
Get data ready...
Training started...
TRAIN | Epoch: 1/10 | Loss: 0.23 | Accuracy: 0.96
TRAIN | Epoch: 2/10 | Loss: 0.10 | Accuracy: 0.97
TRAIN | Epoch: 3/10 | Loss: 0.10 | Accuracy: 0.97
TRAIN | Epoch: 4/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 5/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 6/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 7/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 8/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 9/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 10/10 | Loss: 0.09 | Accuracy: 0.97
Testing Started...
TEST | Average Accuracy per 38 Loaders: 0.95128
```

(a) LSTM Training Result

```
mnist_train 3330
mnist_test 3330
Get data ready...
Training started...
TRAIN | Epoch: 1/10 | Loss: 0.23 | Accuracy: 0.93
TRAIN | Epoch: 2/10 | Loss: 0.10 | Accuracy: 0.97
TRAIN | Epoch: 3/10 | Loss: 0.10 | Accuracy: 0.97
TRAIN | Epoch: 4/10 | Loss: 0.10 | Accuracy: 0.97
TRAIN | Epoch: 5/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 6/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 7/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 8/10 | Loss: 0.10 | Accuracy: 0.97
TRAIN | Epoch: 9/10 | Loss: 0.09 | Accuracy: 0.97
TRAIN | Epoch: 10/10 | Loss: 0.09 | Accuracy: 0.97
Testing Started...
TEST | Average Accuracy per 332 Loaders: 0.96577
```

(b) LSTM Testing Result

Figure 5.2. Result of bi-LSTM

Figure 5.3. Orientation Change Experiment Setup

back to zero in the turn section, the FVZ method is not eligible to be applied to enhance the accuracy of the turn section. Inspired by the FVZ method, we can find the drifting rate of the MaT section by letting the true velocity be the same before and after making the turn. In order to separate the drifting rate on the x and y axis fairly and analyze the drifting problem caused by the orientation change, we designed another experiment where the experimental setup is shown in Figure 5.3.

In the experiments of orientation change, the smartphone is been placed on the keyboard on an executive chair, this makes the radius of the circle 0.5 meters. The chair was been spun at speed as uniform as possible by one researcher, while another researcher recording the top view of the experiment which makes the ground truth location be able to be retrieved. The first picture of the top view video for the orientation experiment is shown in Figure 5.4.

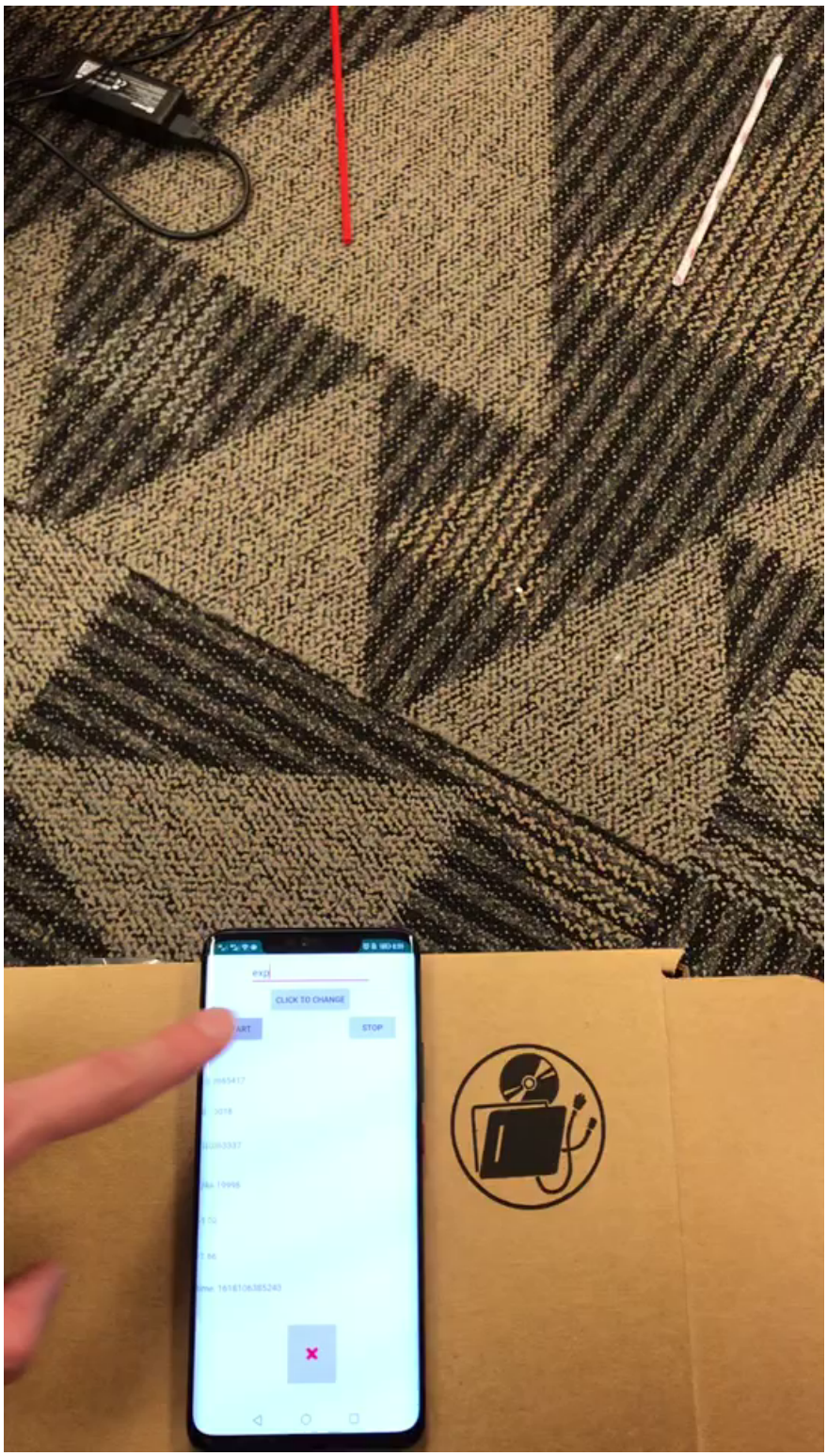
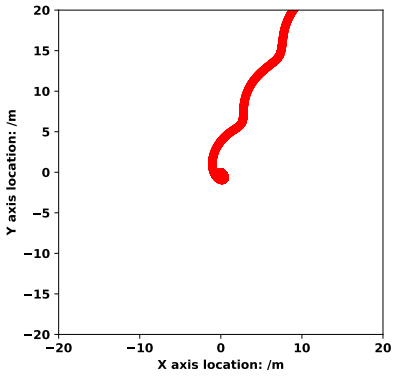
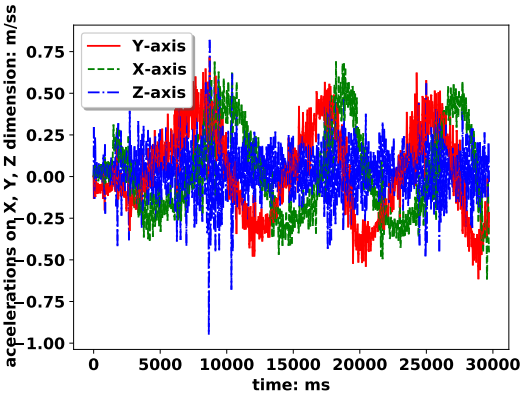


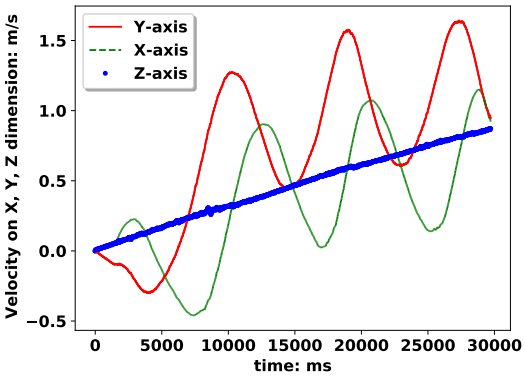
Figure 5.4. Orientation Change Experiment Orientation Top View



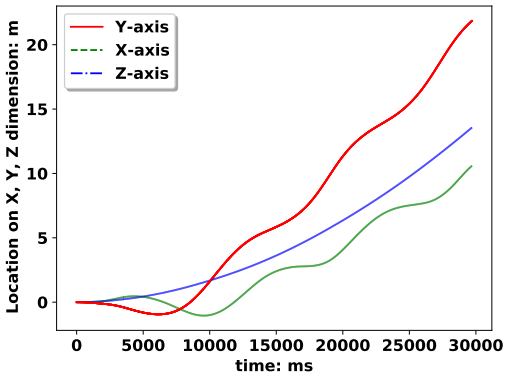
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS

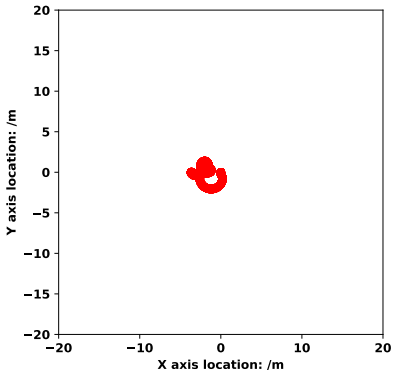


(d) Locations on GCS

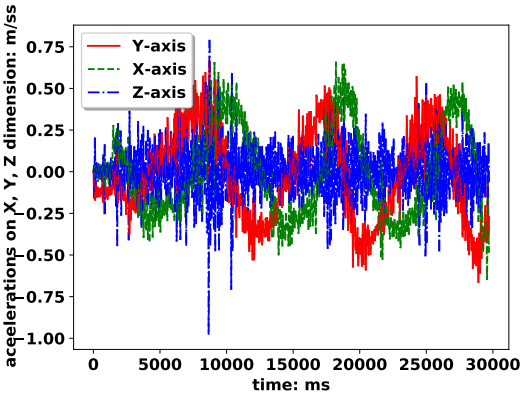
Figure 5.5. Original results of Spin Circles

The chair was spun 3 around 3 times, the experiment results with raw data are shown in Figure 5.5. In Figure 5.5a, the top view shows the trace is a north-east movement, where the actual movement is spinning 3 circles. The variant between the actual movement to the resulting trace can be explained by Figure 5.5c. The velocities on x and y axis show clear patterns again, where the calculated velocities on the horizon plane drift in a linear order periodicity.

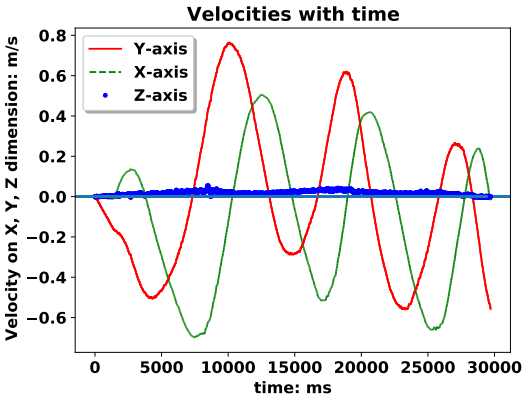
To analyze the drifting problem, the actual velocities on the x, y axis is supposed to be n·$sin(k\pi t)$, where n and k are determined by the spinning movement. However, the calculated velocities on the x, y axis are n·$sin(k\pi t)$ + ct, where c is a constant value caused by sensor errors. Intuitively, linear regression can be used to eliminate the linear order error.



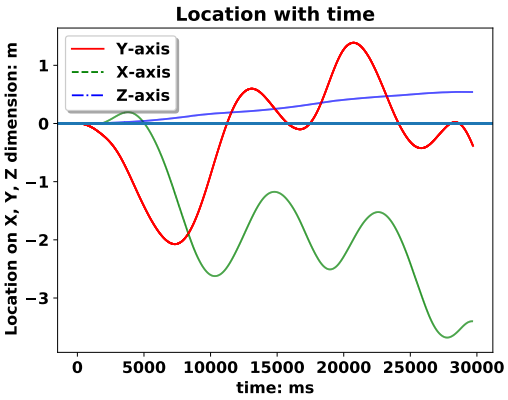
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS

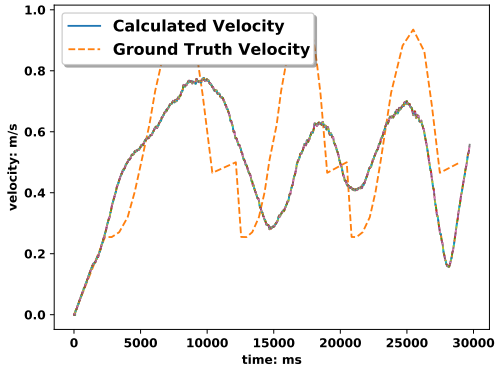


(d) Locations on GCS

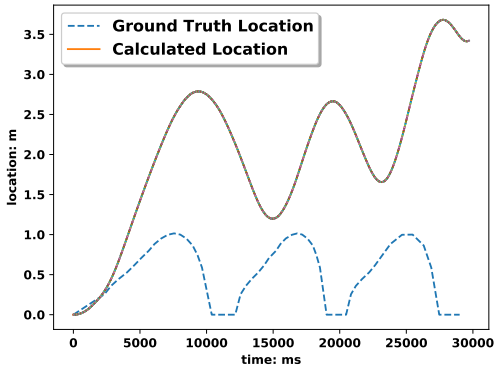
Figure 5.6. Calibration Result of Spin Circles

Calibration results of applying linear regression to cancel the orientation error are shown in Figure 5.6.

As shown in Figure 5.6a, the movement trace no longer looks like a walking trace but drawing actual circles. The Velocities shown in Figure 5.6cindicate the velocities on the x and y axis maintain sine and cosine patterns without drifting patterns. Although the movement trace is not perfect, it is greatly improved comparing to the original trace. The comparison of the calibrated results of true velocity and location with the ground truth is shown in Figure 5.7. The calculated true location is calculated by the square root of locations on the x and y axis. The calculated true velocity is calculated by the square root of the velocities on the x and y axis.



(a) Velocities on GCS



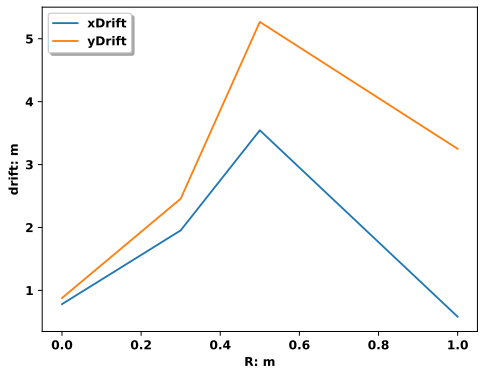
(b) Locations on GCS

Figure 5.7. Calibration Result of Spin Circles Comparing to the Ground Truth.

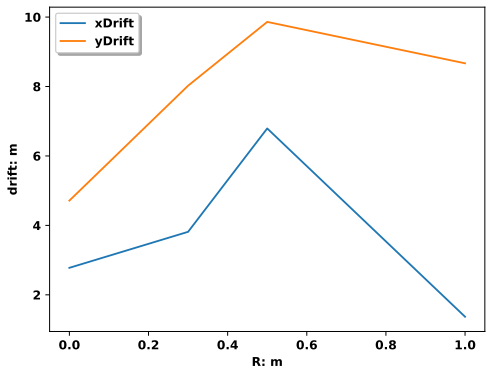
Another experiment we did for orientation change found that the drifting patterns on the x, y axis show similar drifts when the orientation changed. In this experiment, we compared the drifting problem on the x, y axis with different spinning radius and different spinning speeds. The drifts on the x and y axis are compared by the calculated final locations on the x and y axis when spinning one circle.

To show the drifting pattern on the x, y axis is always similar and does not impact by the spinning speed and spinning radius, we conducted two experiments. In the first experiment, the smartphone is spun at a similar speed for one circle with a different radius, the final location on the x and y axis are compared in the same graph. In the second experiment, the smartphone is spun with different speeds but the same radius and the final locations are used to plot the graph. The experiments results are shown in Figure 5.8 and Figure 5.9.

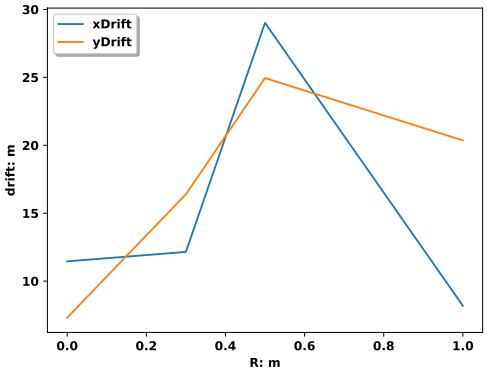
As the experiment results, we found the drifting problem for orientation change shows similar patterns on the x and y axis, and it is not influenced by the turning radius and turning speed. The drifting pattern also shows the static and periodic drifting rate for each full turning round. Motivated by this founding, we came up with the idea to improve the accuracy of the orientation change section in the typical scenario.



(a) Drift vs. Different Radius, speed = fast

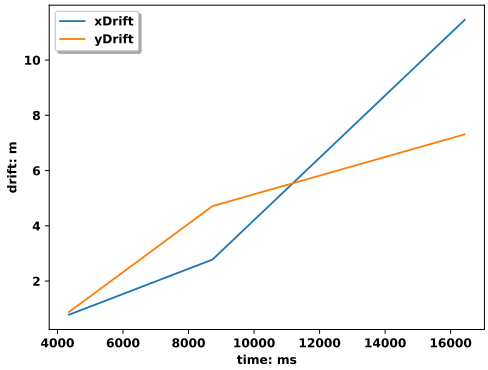


(b) Drift vs Different Radius, speed = slow

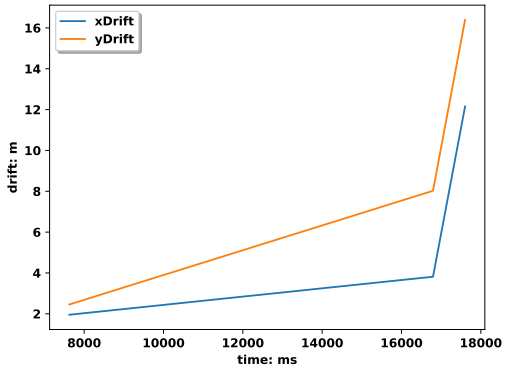


(c) [Drift vs Different Radius, speed = very slow

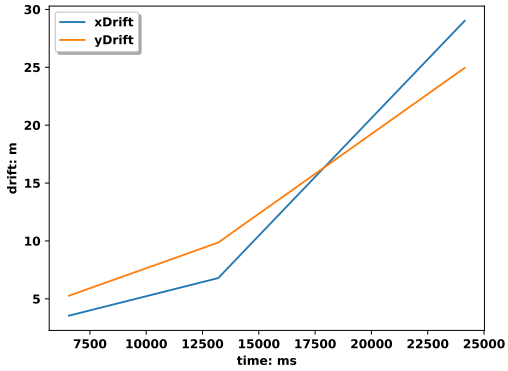
Figure 5.8. Final Locations of Spinning one circle. similar speed.



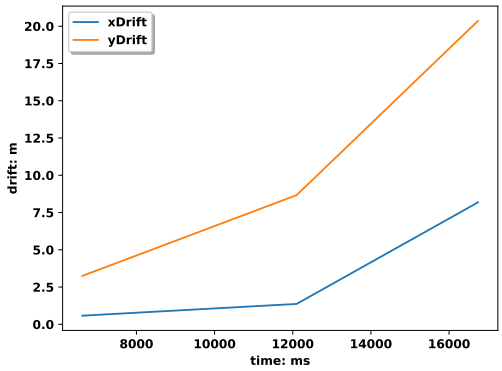
(a) Drift vs. Different speed, radius = 0 m



(b) Drift vs Different speed, radius =0.3 m



(c) [Drift vs Different speed, radius = 0.5 m



(d) [Drift vs Different speed, radius = 1 m

Figure 5.9. Final Locations of Spinning one circle, same radius.

5.4 Improve the Accuracy of the Typical Scenario

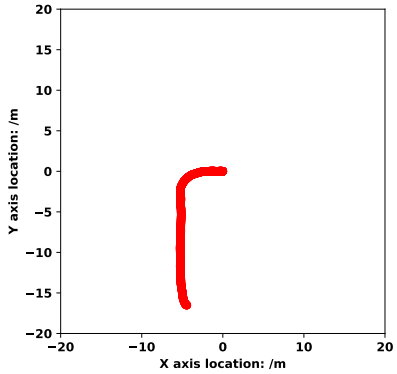
In the real world, a pedestrian's walking trace can be viewed as a Make a Turn (MaT) problem where the user's trace contains only two different patterns: walking straight feature turn feature. Since we have improved the accuracy of walking straight and spinning a circle, we want to take one step further to improve the accuracy of a walking trace in the real-world scenario.

We conducted an experiment where the user holds the smartphone in front of him, walking straight in the Weir Hall's corridor toward the west for 6 meters, then make a left turn of 90 degrees, keep walking 8.5 meters toward the south after making the turn. The experiment results without calibration are shown in Figure 5.10.

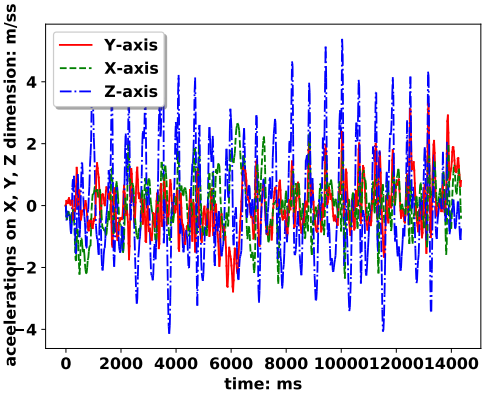
In Figure 5.10a, although the top view of the walking trace shows the turn is 90 degrees, it does not show the Weir Hall corridor's rotation angle. At the same time, the walking distance along the corridor after the turn is too far away from the truth, where the calculated result on the y axis is 17 meters, which is twice the distance to the ground truth. The distance variance can be explained by the velocity figure shown in Figure 5.10c, the velocities show linear order drifting patterns in each walking straight section. The drift has a higher impact on the velocity on the y axis because this velocity drifts to around 1 m/s before the turn, which leads to the initial velocity after the turn a lot faster than it is supposed to be. Therefore, the location on the y axis drifts in quadratic order, which leads the calculated distance twice as the ground truth location.

We applied linear regression on both walking straight sections before and after the turn and calibrated each walking straight section with a different drifting rate. The calibrated result is shown in Figure 5.11.

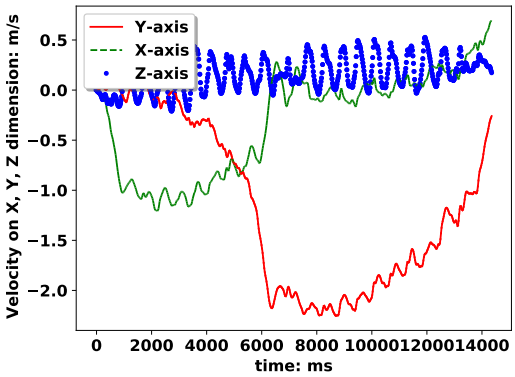
The top view in Figure 5.11a shows a better result than the original calculated result on both directional and distance. As shown in the velocities graph in Figure 5.11c, the velocities on the x, y axis no longer show any drifting patterns, and the distance before the



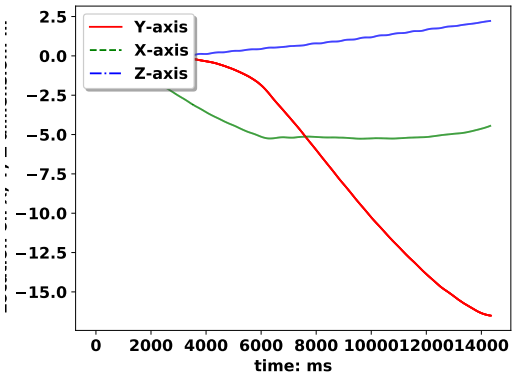
(a) Top View



(b) Accelerations on GCS

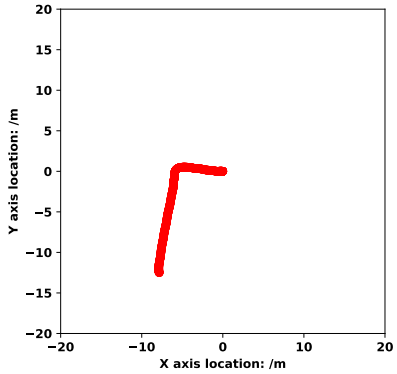


(c) Velocities on GCS

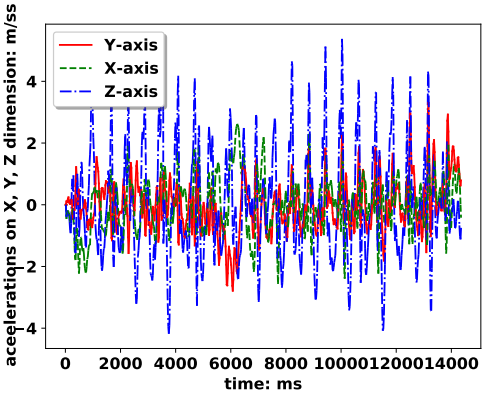


(d) Locations on GCS

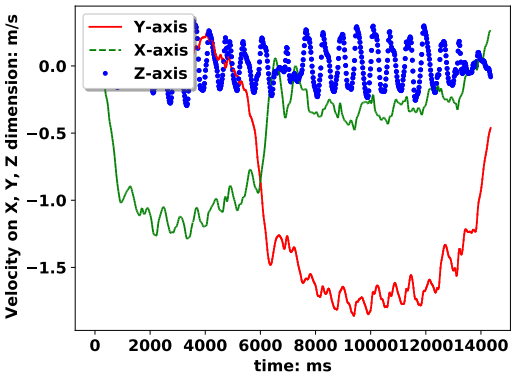
Figure 5.10. Original Make a Turn Problem Results



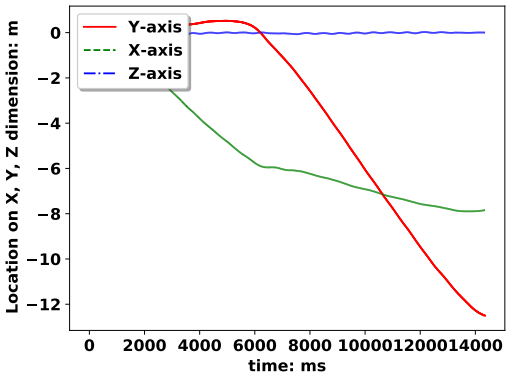
(a) Top View



(b) Accelerations on GCS



(c) Velocities on GCS



(d) Locations on GCS

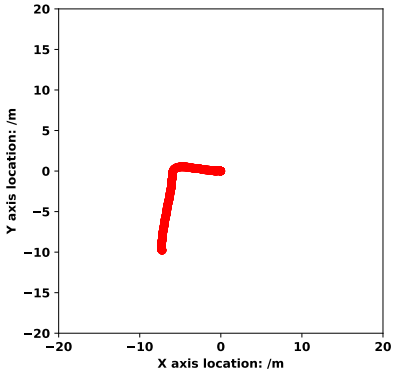
Figure 5.11. Calibrated Make a Turn Problem Results

turn shows the user walked 6 meters, which is the same as the ground truth. However, the final location is still variant from the ground truth. As shown in Figure 5.11d, the velocity on the y axis is still over 12 meters, which is still 4 meters away from the ground truth. Since the walking trace shows an accurate walking trace before making the turn and the velocity graph shows there are no drifts in any direction, it is likely that the distance error is caused by the "turn" section, which causes the velocities after the turn section is higher than the actual velocities on both x and y axis. This means only use linear regression on the walking straight section is not enough to get the accurate distance. The next step is to improve the accuracy of the turn section by the findings we got from the orientation change experiments.

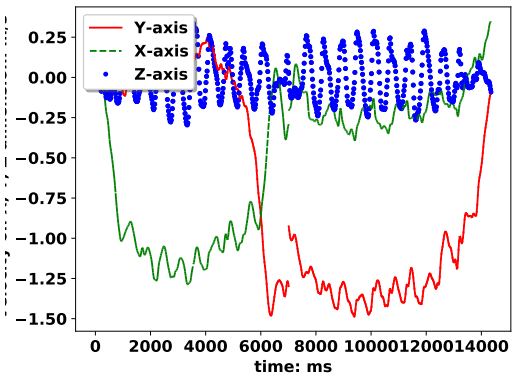
Inspired by the findings of the previous section, we calculated the drifting rate on the true velocity by calculating the difference between the calculated true velocities before and after the turn. Since we have conducted experiments showing the drifting problem on the x and y axis showing the same patterns, we segmented the drifts on the x, y axis equally to force the true velocity to be the same value before and after the turn.

By forcing the true velocity after the turn to the same value as it before the turn, the result does improve. However, we need to do more experiments to learn how to make the turn section generate the correct value, so the velocities and locations do not change suddenly. The calibrated results are in Figure 5.12.

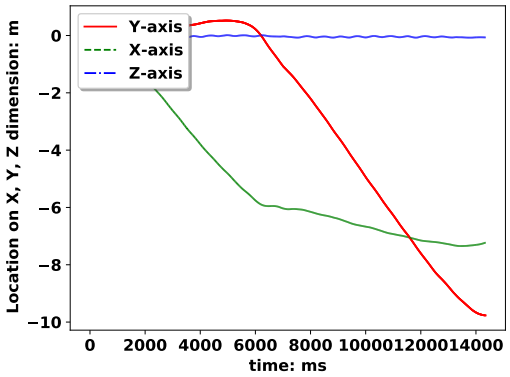
As shown in the velocity ground in Figure 5.12b, the velocities on the x, y axis show a clear abrupt partition at 7 seconds, which was caused by forcing the velocity after the turn section the same value as before the turn section. The abrupt partition is also shown in the true velocity for the same reason, which is shown in Figure 5.12d. However, the abrupt partition does not show in the top view trace or the location graph. As the result, the calibrated result on the y axis is less than 10 meters, which is a great improvement comparing to the original result calculated by the raw smartphone data.



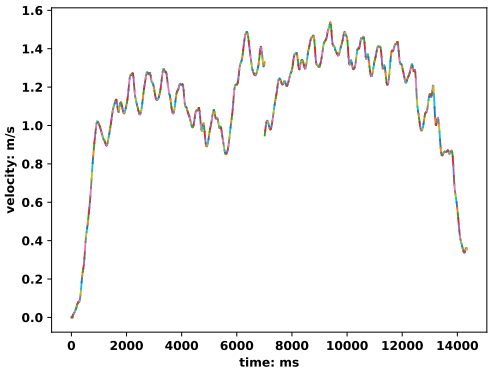
(a) Top View



(b) Velocities on GCS



(c) Locations on GCS



(d) True Velocities on GCS

Figure 5.12. Calibration Results for MaT, Forcing the Velocity After the Turn the Same as Before the Turn

Chapter 6

CONCLUSION AND FUTURE WORKS

6.1 Conclusion

Since all the existing indoor localization approaches have reached a dead-end, it is time to come up with a new indoor localization solution. In order to find a better way to solve indoor localization problem, we built a testbed and conducted extensive experiments and found the drifting problem shows clear patterns and can be removed by different techniques. Motivated by our observations, we proposed our novel indoor localization solution which only uses raw smartphone sensor data. Our novel approach uses a LSTM deep learning model to recognize the user's walking behavior automatically and separate a typical walking behavior into walking straight and turn sections. Then we used different calibration techniques to best remove the sensor error and improved the accuracy by a significant amount comparing to the original result. For the future work, we will update our project to make it be able to segment walking behavior and perform different calibration techniques based on its own decisions.

6.2 Future Works

My thesis presents our finding and a novel feasible solution to tackle the indoor localization problem. However, there are many improvements to be made, and experiments that worth to be further explored. During my experiments and observations, I have came up with many ideas that may be able to further improve the accuracy of our solutions.

The first thing to be done in the future should be to assemble the 3 sections and make it be able to automatically perform post-experiment processing then get the calibrated result. After this step, it is worth trying if the project can be converted to a real-time platform, but this may highly depend on the overall computation overhead.

More experiments needed for segmenting the MaT problem. We will conduct experiments with different users with different high, different weights, and different walking behaviors. We will also update the Bi-LSTM model with more useful features to let it better segment the user's walking behaviors.

The abrupt partition on the x and y axis can be used to learn the drifting rate in the turn section. We did not use it to calibrate the turn section because it should be the result calculated by the turn section. However, the drifting rate can be the label reference for the deep learning model to train the deep learning model.

Since the drifting on orientation change is a constant value that can be calculated using linear regression, we will also design a deep learning model to learn the drifting value when orientation change.

Similar to the orientation change can be learned by deep learning models, walking straight problem should be able to be learned by deep learning models to recognize different walking pace and different walking speed. This deep learning model should be able to tell from the raw sensor data if the user is walking in a certain speed and pace, so it can improve the accuracy of any walking problem one step further.

Even walking straight problems can be future explored, such it can be viewed as a

speed-up section, a constant-speed section, and a slow-down section. Linear regression is very good at the constant speed section, but it can influence, and at the same time, be influenced by the speed-up and slow-down section. The speed-up and slow-down section when walking straight may be further improved by another deep learning model.

Gyroscope can be implemented to keep the orientation change more accurate.

The last experiment I believe that should be done, and it is also my strongest feeling is that the sensor errors can be divided into two kinds of errors: one caused by the white noise that has no clear patterns and produce less damage to the result (because the error made in the latter will offset the noise made in the front); the second kind error is caused by the always existed sensor errors, which has a higher effect to the result. Both errors are exited on both accelerometer and magnetometer. The second kind of error on the accelerometer, for example, will always read -0.005 m/s^2 on x-axis when the smartphone is statically located on a flat plane and it's oriented at a certain orientation, while might be 0.005 m/s^2 if flip the smartphone, and at the same time, if magnetometer is accurate. This number will be enlarged if the acceleration is bigger, and it's various smartphones to smartphones. To prove this assumption, there is no need if using deep learning because simple experiments should be enough. The process can be done by rotating the smartphone with different orientations and leave it statically, recording the accelerations on each dimension. Since gravity is around 9.8 m/s^2, it should be enough to prove the assumption. If the assumption can be proved, this should be a huge breakthrough to totally solve the indoor localization problem because we proved our solution should yield over99.8% accuracy if the second kind of sensor error does not exist. All we need to remove the sensor error is to offset it with the negative value that can be calculated.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] L. Allison and M. M. Fuad. Inter-app communication between android apps developed in app-inventor and android studio. In *2016 IEEE/ACM International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, pages 17–18, 2016.

[2] K. Chiang, H. Chang, Y. Li, G. Tsai, C. Tseng, Y. Tien, and P. Hsu. Assessment for ins/gnss/odometer/barometer integration in loosely-coupled and tightly-coupled scheme in a gnss-degraded environment. *IEEE Sensors Journal*, 20(6):3057–3069, 2020.

[3] V. Filonenko, C. Cullen, and J. Carswell. Indoor positioning for smartphones using asynchronous ultrasound trilateration. *ISPRS International Journal of Geo-Information.*, 2:598–620, 06 2013.

[4] R. Harle. A survey of indoor inertial positioning systems for pedestrians. *IEEE Communications Surveys Tutorials*, 15(3):1281–1293, 2013.

[5] J. A. Hesch, F. M. Mirzaei, G. L. Mariottini, and S. I. Roumeliotis. A laser-aided inertial navigation system (l-ins) for human localization in unknown indoor environments. In *2010 IEEE International Conference on Robotics and Automation*, pages 5376–5382, 2010.

[6] C. Khawas and P. Shah. Application of firebase in android app development-a study. *International Journal of Computer Applications.*, 179:49–53, 06 2018.

[7] M. P. R. S. Kiran, P. Rajalakshmi, M. K. Giluka, and B. R. Tamma. A novel system architecture for real-time, robust and accurate step detection for pdr based indoor localization. In *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*, pages 754–759., 2018.

[8] B. Labinghisa and D. M. Lee. Drift-free indoor pedestrian dead reckoning using empirical mode decomposition. In *2019 25th Asia-Pacific Conference on Communications (APCC)*, pages 262–266., 2019.

[9] H. D. R. Lakmal and J. Samarabandu. Spatial direction corrections to improve indoor localization using inertial navigation with sensors on a smartphone. In *2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–4, 2016.

[10] C. Langlois, S. Tiku, and S. Pasricha. Indoor localization with smartphones: Harnessing the sensor suite in your pocket. *IEEE Consumer Electronics Magazine*, 6(4):70–80., 2017.

[11] H. Lee, S. Choi, and M. Lee. Step detection robust against the dynamics of smartphones. *Sensors*, 15(10):27230–27250, 2015.

[12] A. Poulose and D. S. Han. Indoor localization using pdr with wi-fi weighted path loss algorithm. In *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 689–693, 2019.

[13] D.J. Schott, F. Höflinger, R. Zhang, L.M. Reindl, and Hai Yang. Fuzzy inference system assisted inertial localization system. In *2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pages 89–93, 2017.

[14] M. Shchekotov. Indoor localization method based on wi-fi trilateration technique. In *Proceeding of the 16th Conference of Fruct Association*, pages 177–179, 2014.

[15] J. Son and Y. S. Son. A correlation analysis of indoor environmental quality and indoor air quality using iot. In *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 977–979., 2019.

[16] F. Zafari, I. Papapanagiotou, M. Devetsikiotis, and T. Hacker. An ibeacon based proximity and indoor localization system. *arXiv preprint arXiv:1703.07876*, 2017.

[17] M. Zhang, Y. Wen, J. Chen, X. Yang, R. Gao, and H. Zhao. Pedestrian dead-reckoning indoor localization based on os-elm. *IEEE Access*, 6:6116–6129., 2018.

[18] W. Zhang, R. Sengupta, J. Fodero, and X. Li. Deeppositioning: Intelligent fusion of pervasive magnetic field and wifi fingerprinting for smartphone indoor localization via deep learning. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 7–13., 2017.

[19] K. Zhao, T. Liu, and J. Zhao. An orientation-independent geomagnetic indoor localization method for mobile phone. In *2019 Chinese Automation Congress (CAC)*, pages 1725–1730., 2019.

VITA

Yunshu Wang

702-215-9618 | yunshuwang95@gmail.com

EDUCATION

Bachelor of Science in Computer Science at the University of Mississippi, Aug. 2016 – May 2019, University, Mississippi.

Master of Science in Computer Science at the University of Mississippi, Aug. 2019 – May 2021, University, Mississippi. Thesis title: "A Novel Approach Toward High Accuracy Indoor Localization".

RESEARCH EXPERIMENT

GRADUATE RESEARCH ASSISTANT, University of Mississippi Dept. of Computer and Information Science, University, MS, Aug. 2019-May, 2021.

UNDERGRADUATE RESEARCH ASSISTANT, University of Mississippi Dept. of Computer and Information Science, University, MS, Nov. 2017-Dev 2018.

UNDERGRADUATE RESEARCH INTERSHIP, University of Mississippi Dept. of Bio-molecular Science, University, MS, May 2018-Aug 2018

TEACHING EXPERIMENT

GRADUATE TEACHER ASSISTANT, University of Mississippi Dept. of Computer and Information Science, University, MS, July 2019-May, 2021.

UNDERGRADUATE TEACHER ASSISTANT, University of Mississippi Dept. of Computer and Information Science, University, MS, July 2017-May 2019

PUBLICATION Y. Wang, L. Easson, F, Wang. 2021. *Testbed Development for a Novel Approach Towards High Accuracy Indoor Localization with Smartphones.* ACMSE.