2023

LOCALIZATION OF PEOPLE IN GNSS-DENIED ENVIRONMENTS USING NEURAL-INERTIAL PREDICTION AND KALMAN FILTER CORRECTION

Lauren N. Cash
lnc00008@mix.wvu.edu

LOCALIZATION OF PEOPLE IN GNSS-DENIED ENVIRONMENTS USING NEURAL-INERTIAL PREDICTION AND KALMAN FILTER CORRECTION

Lauren Cash

Thesis submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University

in partial fulfillment of the requirements for the degree of

Masters of Science in Mechanical Engineering (M.S.M.E.)

Guilherme Augusto Silva Pereira, Ph.D., Committee Chairperson
Dimas Abreu Archanjo Dutra, Ph.D.
Jason Gross, Ph.D.

Department of Mechanical and Aerospace Engineering

Morgantown, West Virginia
2023

Keywords: Localization, Inertial Navigation, Neural Network, Kalman Filter

Abstract

Localization of People in GNSS-Denied Environments using Neural-Inertial Prediction and Kalman Filter Correction

Lauren Cash

This thesis presents a method based on neural networks and Kalman filters for estimating the position of a person carrying a mobile device (i.e., cell phone or tablet) that can communicate with static UWB sensors or is carried in an environment with known landmark positions. This device is used to collect and share inertial measurement unit (IMU) information — which includes data from sensors such as accelerometers, gyroscopes, and magnetometers — and UWB and landmark information. The collected data, in combination with other necessary initial condition information, is input into a pre-trained deep neural network (DNN) which predicts the movement of the person. The prediction result is then periodically — based on outside measurement availability — updated to produce a more accurate result. The update process utilizes a Kalman Filter approach that relies on empirical and statistical models for DNN prediction and sensor noise. Therefore, the approach combines the principles of artificial intelligence and filtering techniques to produce a complete system which converts raw data to trajectory results of people. The initial tests were completed indoors where known landmark locations were compared with predicted positions. In a second set of experiments, GNSS location signals were combined with position estimation for correction. The final result shows the correction of neural network prediction with data from UWB sensors having known locations. Prediction and correction trajectories are shown and compared with the ground truth for applicable environments. The results show that the proposed system is accurate and reliable for predicting the trajectory of a person and can be used in future applications that require the localization of people in scenarios where GNSS is degraded or unavailable, such as indoors, in forests, or underground.

Acknowledgments

I would first like to recognize my advisor Dr. Guilherme Pereira and thank him for his support and guidance during my time at WVU. I would also like to express gratitude to Dr. Dimas Dutra and Dr. Jason Gross for serving on my graduate committee and helping further my understanding over the course of this project.

I would also like to acknowledge the staff and faculty in the Department of Mechanical and Aerospace and adjacent departments for developing a meaningful learning environment that allowed me to grow into a better student, researcher, and person.

Finally, I want to thank my parents and my fiancé for their encouragement of my dreams and their belief in me, without them I would not have made it quite this far. I want to thank my siblings for their commitment to my success and their motivation when it was needed. I want to thank my friends for their distractions at the most essential times and their reassurance always.

With my eternal appreciation,

Lauren

This work was supported through a subcontract with Kinnami Software Corporation under the Airforce Research Lab SBIR project #FA864922P0911.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

Abbreviation	Description
DNN	Deep Neural Network
DR	Dead-Reckoning
EKF	Extended Kalman Filter
GHPE	Global Human Pose Estimation
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IMU	Inertial Measurement Unit
IN	Inertial Navigation
KF	Kalman Filter
LKF	Linear Kalman Filter
LSTM	Long Short-Term Memory
MEMS	Micro-Electro-Mechanical-System
NAS	Neural Architecture Search
NN	Neural Network
QR	Quick Response
RUKF	Robust Unscented Kalman Filter
SLAM	Simultaneous Localization And Mapping
TCN	Temporal Convolutional Network
UWB	Ultra-Wideband
ZUPT	Zero Velocity Update

Nomenclature

Symbol	Description
X	Position in X Direction *or* Longitude
Y	Position in Y Direction *or* Latitude
θ	Orientation in $X - Y$ Plane *or* Heading
C	Model of the System
$\hat{x}$	State Vector
P	State Covariance Matrix
Q	Process Noise Covariance Matrix
R	Sensor Noise Covariance Matrix
K	Kalman Gain
μ	Error
Cov	Covariance Value
Var	Variance Value
$StDev$	Standard Deviation
Δi	Change in "i"
$acce$	Accelerometer Data — accelerations
$gyro$	Gyroscope Data — angular velocities
rv	Rotation Vector — orientations around an axis, relative to north
grv	Game Rotation Vector — orientations around an axis, relative to other

Chapter 1

Introduction

1.1 Motivation

In a world of ever-increasing interconnectedness, navigation methods have become vital to most of the devices and processes used in everyday life. From social media check-ins and virtual reality simulations to gaming applications and even grocery and food delivery, the services we have started to depend upon are based on the principles of localization. The most common form of this is the Global Navigation Satellite System (GNSS), including the Global Positioning System (GPS) [4]. For GPS to provide an accurate position, at least four of the 31 operational satellites must be able to connect to a receiver, carried by a specific person or embedded in a device on the surface of the Earth [5]. An example GPS satellite is shown in Figure 1.1. However, there are certain circumstances in which satellite navigation is unavailable — such as indoors, underground, or in dense forests. In such cases, it is essential to determine an alternate technique for location-based services. This process — specifically relating to indoor navigation — has been researched intensely in recent years and solutions will continue to be developed for pedestrian, vehicle, and robotic applications.

Having access to similar information and requiring similar constraints, another circumstance in which satellite signals may be inaccessible or hazardous is an unsafe battlespace. Transmissions from a satellite may be interfered with in multiple ways, including jamming

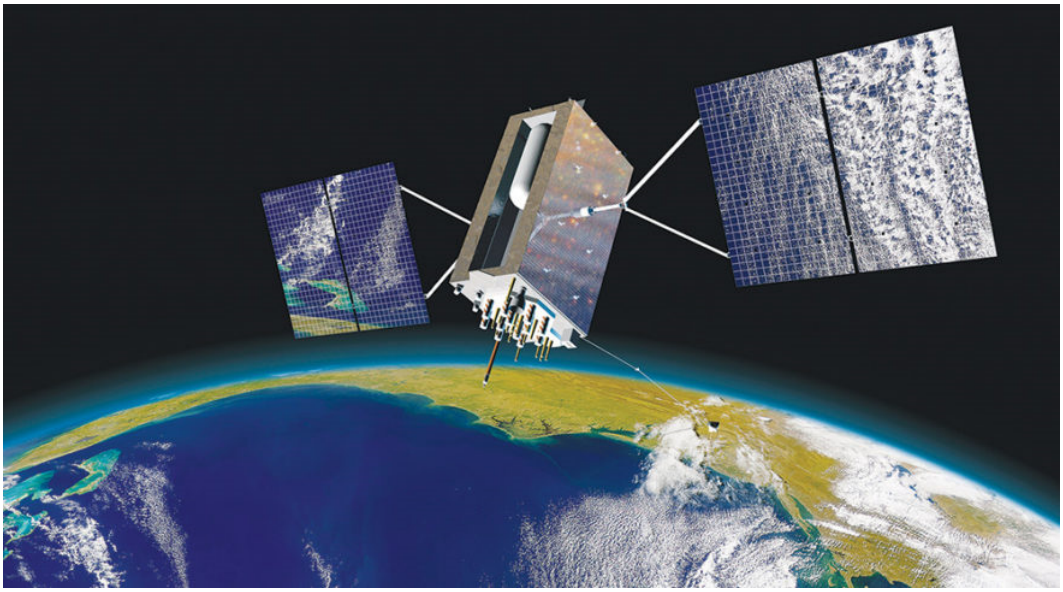

Figure 1.1: A satellite shown above the surface of the Earth. This photo is licensed under CC BY (creativecommons.org/licenses/by/3.0/).

— in which someone attempts to override the true signal with meaningless noise, or spoofing — in which someone creates a new, stronger signal to confuse the receiver [6]. If a military personnel is lost or separated from their group in a place where is it essential to remain hidden, using GNSS services to share a location may reveal unwanted information to enemies. However, in such search and rescue missions, it is still crucial to have a method for determining locations and trajectories of both the search team and lost member — independent of satellite and other such signals.

This thesis describes the process used to create a localization system that predicts the trajectory and orientation of people and corrects the prediction based on other measurement data. The estimation is achieved using data collected from mobile devices with limited or no access to GPS or other satellite systems. This involves the use of a neural network to convert raw Inertial Measurement Unit (IMU) information into location and heading data, a process that has been successfully implemented as a state of the art for inertial navigation. The correction is performed using known landmark locations, limited GPS signals, or Ultra-Wideband (UWB) signals. When possible, the correction process is applied using linear or

Extended Kalman filter techniques in combination with other information to improve the overall trajectory estimation and reduce the error and noise of the entire process.

The use of neural networks to assist with solving a variety of problems has increasingly grown in recent years. Neural networks have broad, diverse applications and are both effective and efficient at creating procedures or finding solutions. Linear and extended Kalman filters are also adaptable to be used in many different ways, and are commonly associated with trajectory estimation. The motivation for the use of a neural network and Kalman filter to accomplish the position and heading predictions is related to the requirements for a fast, active prediction with minimal error that can be performed in a variety of different situations with changing conditions. The objectives of this research are presented in the next section.

1.2 Objectives

The aim of this research is to create a system that is able to estimate the position and heading of a moving person carrying a mobile device (e.g., cellphone or tablet), therefore calculating the person's trajectory from a known or determined initial condition to a final location. We assume that, besides internal IMU data, the mobile device has access to a combination of other information such as landmark locations identified by the person being localized, (limited) GNSS localization information, and/or range between known locations and the person obtained with UWB radios. The next section details the contributions of this thesis.

1.3 Contributions

This thesis presents an improved process for estimating the position and heading of a person carrying a mobile device in GPS-denied and GPS-limited environments. To complete this, a pre-trained neural network has been selected that predicts the position and orientation of a moving person using IMU data. This system will be modified to function in an

online manner where data is provided and the prediction is made in an iterative process. An evaluation will be completed to determine the noise and covariance of the neural network trajectories. The methods that will be used for the update have been selected as known landmark locations, GNSS positions, and UWB range information, therefore a second evaluation will be conducted to determine the noise and covariance of the sensor measurements. To finish the system, linear or extended Kalman filter techniques will be applied to the prediction and covariance information to achieve an estimate for each iteration. These estimates will be combined to create a final trajectory for the motion. This procedure has been calibrated over numerous trials to ensure the process is reliable and complete, and can therefore be applied to real-life scenarios of localization. An example graphic showing the basic process of this system is shown in Figure 1.2.

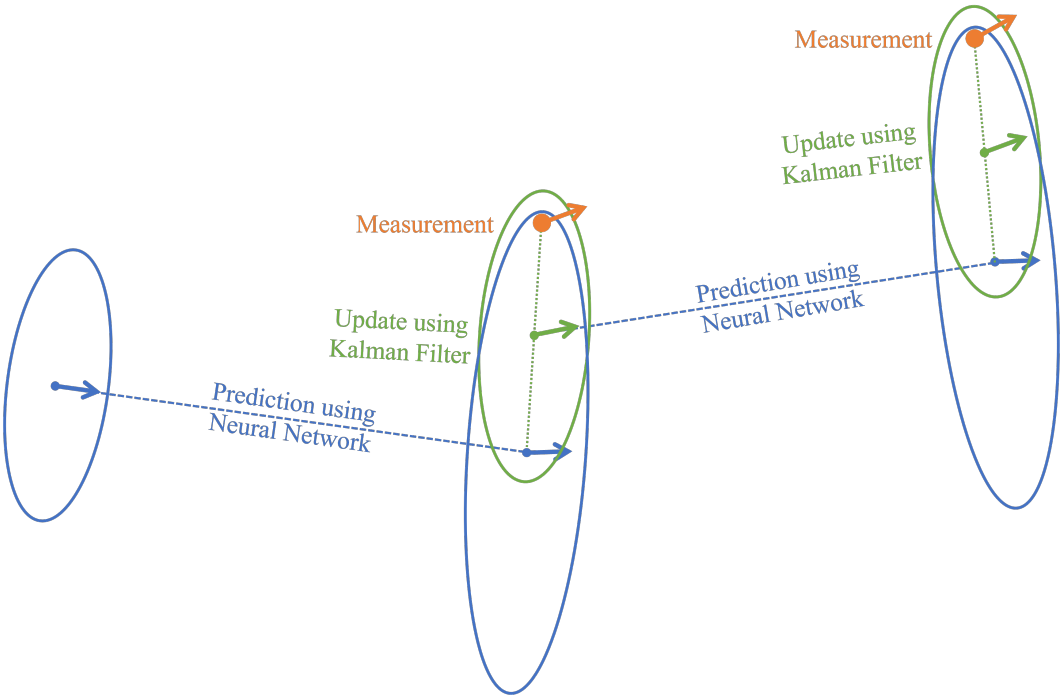

Figure 1.2: A basic overview of the proposed system. With a known initial position, heading, and covariance, a prediction occurs using a neural network to find an intermediate estimate for the next position and heading. A measurement is taken — another approximate for the actual trajectory — which can be used to update the intermediate estimate using a Kalman filter. This produces the new position, heading, and covariance data, which are then used to start the process again.

Therefore, the main contributions are:

1. An online solution for a pre-trained neural network infrastructure that accepts raw IMU data (accelerometer, gyroscope, game rotation vector values) and determines an estimated position.

2. An online solution for a pre-trained neural network infrastructure that accepts raw IMU data (accelerometer, gyroscope, game rotation vector values) and other data (rotation vector) and determines an estimated heading.

3. An empirical analysis performed on the trajectories previously discussed to deterministically calculate error and noise growth and numerically describe the covariance of the neural-network prediction.

4. A statistical evaluation conducted on UWB equipment to quantify the error and covariance of measurement values.

5. An overall solution that updates neural-network predictions with Kalman filtering techniques using predicted location and landmark, GPS, or UWB data. A further explanation of the landmark, satellite, and UWB methods for trajectory correction is shown in Figure 1.3.

1.4 Thesis Organization

This thesis is organized in the following manner. The Introduction in Chapter 1 presents the Motivation, Problem Statement, and Contribution — as previously shown. The Literature Review in Chapter 2 details the state-of-the-art solutions for localization and position prediction. The Background in Chapter 3 discusses important topics relating to the content of this project, including Ultra-Wideband, Inertial Navigation, Kalman Filtering, Neural Networks, and the RoNIN solution. The Methodology is shown in Chapter 4 and the Results are stated in Chapter 5. Finally, all Conclusions and Future Work are explored in Chapter 6.

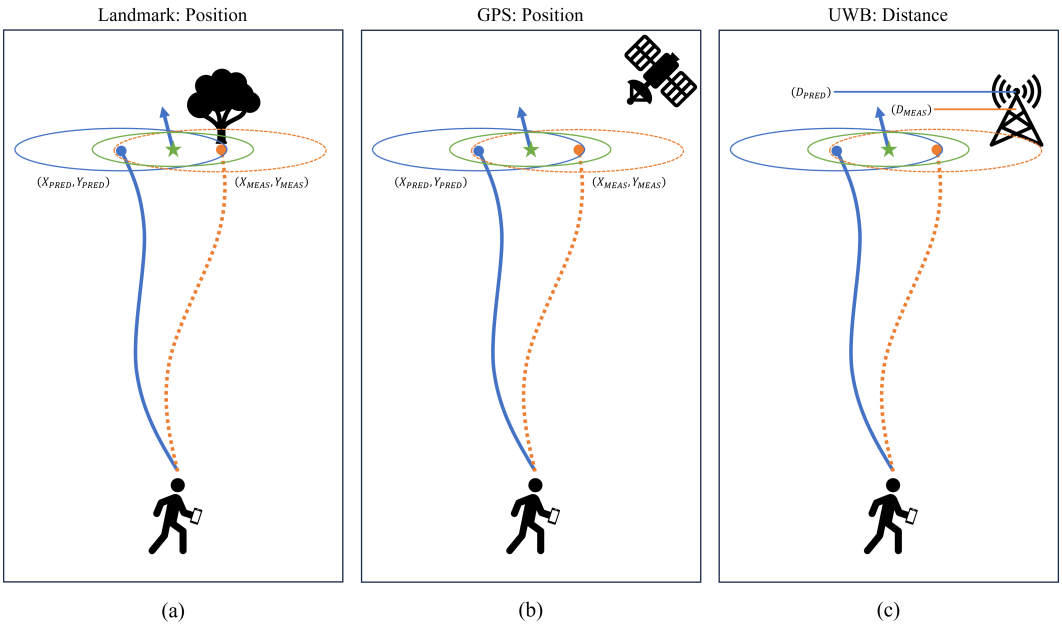

Figure 1.3: Examples of each correction method for the proposed system. For each model, the predicted path is shown in blue and the actual path is shown in orange. (a) The correction of the predicted path using a tree with a known location as a landmark. (b) If available, the update of the trajectory can occur with GNSS signals. For (a) and (b) The predicted position for a specific time is the blue point with the coordinates (X_{PRED}, Y_{PRED}) and the covariance shown. The measurement position for that time is the orange point with the coordinates (X_{MEAS}, Y_{MEAS}) and the covariance shown. The corrected estimate for that time is the green star with the covariance shown. (c) Using UWB sensors to measure the distance, a correction can occur to improve the localization. For (c) The predicted distance for a specific time is the blue line with the length (D_{PRED}) at the blue point with the covariance shown. The measurement distance for that time is the orange line with the length (D_{MEAS}) at the orange point with the covariance shown. The corrected estimate for that time is the green star with the covariance shown.

Chapter 2

Literature Review

The localization of people, robots, or objects without access to GNSS services has been researched in many different capacities. These can be split into several different categories, depending on various factors. For this project, research has been directed into two groups: techniques based on the type of sensors utilized and techniques that are reliant on the characteristics of the location of the trajectory.

2.1 Sensor Dependent Prediction

The construction of the measurement and sensing equipment is an essential foundation for localization. Different aspects of the sensor type and mounting can change the final outcome of the solution. Some researchers exploit the consistent nature of the human gait to assist with positioning. Other projects take into account recently developed or improved measurement technologies for additional processing capabilities. To begin the review, wearable devices will be discussed.

2.1.1 Wearable Devices

Often, when estimating the trajectory of a moving person, wearable devices are used to allow consistent positioning of a sensor relative to an agent. This has been implemented

using various methods in the past. One such solution relies on the principles of zero velocity update (ZUPT) in combination with an IMU sensor mounted on a shoe. The main principle of ZUPT can be represented as follows. During a normal stride, a foot can be considered momentarily motionless while touching the ground. Therefore, any accelerometer values (aka acceleration measurements) detected can be considered error or drift and can be effectively canceled to correct the system [7]. Other methods use smartwatch-like devices to collect inertial data, which in turn can then be used to estimate velocities and trajectories to solve short and long-distance navigation problems [8, 9]. Even head-mounted measurement devices have been researched for use in pedestrian inertial navigation [10]. Some examples of these measurement sensors are shown in Figure 2.1.

(a) (b) (c)

Figure 2.1: Examples of wearable devices. (a) An IMU sensor mounted on a shoe. (b) A smartwatch device capable of taking IMU measurements. (c) A band with an attached IMU sensor positioned on the back of the head. These photos are either licensed under CC BY (creativecommons.org/licenses/by/3.0/) or CC BY-NC-ND (creativecommons.org/licenses/by-nc-nd/3.0/).

2.1.2 Advanced Technologies

Technology-driven research has also expanded recently, with the implementation of new techniques coinciding with advancements in sensors and measurement tools. Some solutions have been developed that leverage a neural-inertial framework for ultra-resource-constrained devices. These methods utilize Neural Architecture Search (NAS) and Temporal Convolutional Network (TCN) procedures to train lightweight models for various applications in-

cluding pedestrian, animal, and aerial [11]. Other techniques use millimeter-wave radar to determine position based on velocity measurements and heading information gained from sparse point clouds [12]. Radio sensors have also become increasingly popular, where ultra-wideband (UWB) sensors and mobile agents are used together to improve localization with ranging information [13, 14]. The next section will discuss recent research pertaining to predictions reliant on conditions or locations will be considered.

2.2 Location Dependent Prediction

Research that is applicable to a specific environment often relies on special features of that environment to create or tune the method. Some solutions use knowledge about the area surrounding the prediction to localize the positions. Projects that are concerned exclusively with outdoor localization in areas with known environmental constraints are sometimes accompanied by prior information about the area. Indoor positioning has also been researched extensively, often taking advantage of the predictability of paths in buildings. Other structured environments such as urban cities or underground tunnels are often tested using a combination of aspects from outdoor and indoor positioning. First, this section will explore processes that depend on information about the environmental conditions.

2.2.1 Environment Aware

Many of the state-of-the-art procedures for localization rely on knowledge about the environment or planning information about the path. Landmark-based localization is a straightforward methodology that can be used as a supplement when other methods of location estimation are unreliable or unavailable. This is especially helpful when an accurate mapping of the area can be created or is provided. Based upon the idea of relative positioning: a camera is used to calculate the position of certain known obstacles or tags relative to the agent, which can in turn be used to estimate the global position of the device [15]. Self-localization has also been researched in other satellite-denied areas: underground mines. In these cases, UWB-ranging information from multiple sensors with known locations can be

used for position estimation. This concept is a similar, smaller-scale application of the theory behind GNSS localization. This method has been applied in both low-precision scenarios such as for equipment and high-precision scenarios such as for people [16]. Having similar constraints, positioning for indoor scenarios will be reviewed in the following subsection.

2.2.2 Indoor Localization

Because most people spend the majority of their time inside a building, indoor localization is an important topic to research. Combining many state-of-the-art techniques, a neural inertial navigation technique has been developed that uses IMU history to form velocity vectors [17]. These vectors are then transformed using a neural network to determine the estimated location of a person carrying a mobile device. This procedure utilizes location likelihood calculations and is heavily based on the predictable pattern of human behavior indoors. There has also been research used to create a spatial and temporal hybrid neural network for pose-invariant inertial odometry. This is accomplished using both local and global information from IMU sensors to update the model during indoor localization [18]. A specific indoor scenario that can benefit from localization technologies is large, complex, multi-level indoor areas such as shopping mall complexes. Research has been conducted into a combination of Wi-Fi measurements, Bluetooth capabilities, quick response (QR) codes, and micro-electro-mechanical-system (MEMS) sensors for trajectory optimization and database construction [19]. Among these similar methods [2] was used in this thesis for motion prediction.

2.2.3 Structured Scenarios

Another common method for position estimation involves Simultaneous Localization And Mapping (SLAM) and Cooperative SLAM for groups of ground and aerial robots. These applications involve the use of multi-modal data collection such as lidar, cameras, GNSS, and inertial navigation in large-scale urban environments [20]. There has also been a scene-aware dataset established to support global human pose estimation (GHPE) for human-scene interaction in large urban areas. Human motions are captured using lidar

point clouds, images, videos, and IMU-based motion frames to allow for prediction [21]. Moving towards scenarios that are less defined, outdoor positioning will be investigated next.

2.2.4 Outdoor Localization

Although GNSS or other satellite navigation services are generally accessible outdoors, it is also common to develop methods for use when the signals are either unavailable or too broad for fine motions. In some cases, such as underneath the dense canopy of a forest, position data can be determined by map fusion. Typically, a "reference map" created with canopy height information from an aerial lidar scan is combined with a "local map" determined through a 3D lidar scan from a mobile robot on the floor of the forest [22]. Another application that is increasing in popularity is the use of localization to assist with agricultural purposes including precision farming [23].

2.3 Cooperative Localization

Lastly, a process that can be applied with any sensors or in any environment to improve estimates is cooperative localization. A common technique that involves a system of multiple robots; each robot independently predicts its own positions as well as a confidence value about the reliability of its prediction. Using their predicted positions and relative confidence levels, robots can improve their own estimation using information from other sources. When two robots enter within the communication range, this information is exchanged and used to create a relative measurement between the pair that is used to correct their individual predictions. This method is generalized to apply to a broad range of applications and situations — without a requirement for global communication or large memory storage [24, 25]. In the next chapter, the background principles of the proposed research method will be reviewed.

Chapter 3

Background

This chapter provides background on the topics covered in this thesis such as ultra-wideband sensors, inertial navigation, Kalman filtering, neural networks, and the RoNIN solution, which are used throughout the stages of the proposed method.

3.1 Ultra-Wideband Sensors

Ultra-Wideband (UWB) sensors, which have been used in commercial and research applications for many years, are becoming increasing more utilized as a tool for indoor localization. When implemented in these applications, range information in between two of these sensors is typically used; this is measured using time-of-flight in the radio signals. Generally, this is executed by placing a UWB sensor (designated as an anchor) in a known location and fitting a person or robot with another UWB sensor (designated as a tag) to determine the distance between the pair. UWB techniques are useful because the sensors are inexpensive, energy-efficient, and precise, and are able to provide data in real-time [26, 27]. UWB measurements are commonly combined with inertial navigation methods, the basis of which will be expanded upon in the next section.

3.2 Inertial Navigation

Some of the most common methods for estimating the locations or trajectories of people can be based on technologies already found inside mobile devices. An Inertial Measurement Unit (IMU) is a combination of Micro-Electro-Mechanical-System (MEMS) components such as accelerometers, gyroscopes, and magnetometers. They require little energy, can take measurements constantly, have no mounting restrictions, and are already installed in every modern smartphone. When the standard sensors provided in phones are tested against similar, consumer grade sensors, the accuracy of the measurements has been found as comparable [28]. This indicates that IMU values reported by smartphones are consistent with other methods and can be reliability utilized for calculations.

Accelerometers report the acceleration of the sensor, including the measurement of gravity, along an axis. Gyroscopes report the rate of rotation of the sensor around an axis. Magnetometers report the ambient magnetic field along an axis [29]. Each of these values is determined with respect to three axes, which are shown in Figure 3.1.

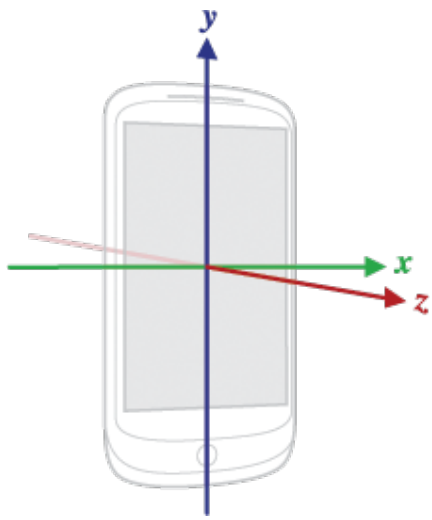

Figure 3.1: The coordinate system used for sensors in Android devices. This photo is licensed under CC BY-SA (creativecommons.org/licenses/by-sa/3.0/).

There are a wide variety of MEMS based techniques for localization. The foundation of nearly all of these rely upon position, velocity, acceleration, and orientation measurements

in combination with other data to predict the motion of a device or the person carrying it [30]. In particular, this research will focus on inertial navigation (IN). IN is a form of Dead-Reckoning (DR) that utilizes IMU sensor information and data processing techniques to transform raw measurements into a useful navigation solution. This method is based on Newton's second law of motion, which states that rates of motion can be estimated if the forces on a body are known. Using an accelerometer to measure accelerations and a gyroscope to measure angular velocities, these formulas can be mathematically solved to determine position information using double integration [31].

Algorithms that rely on the concept of double integration can be considered to have no-priors; they have the simplest procedure. If the IMU sensor acceleration is know, it can be integrated twice to achieve the position. However, this method is susceptible to error, and is difficult to use in practice with additional constraints. One such practical example is [7], which capitalizes on the zero velocity update technique (ZUPT). When the IMU sensor attached to the boot is on the ground, the actual acceleration is momentarily zero. Therefore, any significant value of measured acceleration can be estimated as accumulated errors and corrected.

Algorithms with heuristic priors exploit the repetitive nature of human motion and allow further assumptions to be made about the trajectories. An approach often used, such as in [32], is step counting. The general, baseline assumptions made by the authors include that motion can only occur in the forward direction, the IMU device is held in the hand and always perpendicular to the ground, and that a precise start position is always available. Because these assumptions drive the foundation of the algorithm, movements that are determined to be "non-typical" cannot be captured.

In general, these Inertial Navigation System approaches take raw gyroscope information in the form of angular velocity and integrate to create orientation data. These orientation values are then combined with the raw acceleration values from an accelerometer to project onto a global coordinate frame and compensate for gravity. Double integration is then applied to create position information for the system [1]. This process is described graphically in Figure 3.2.

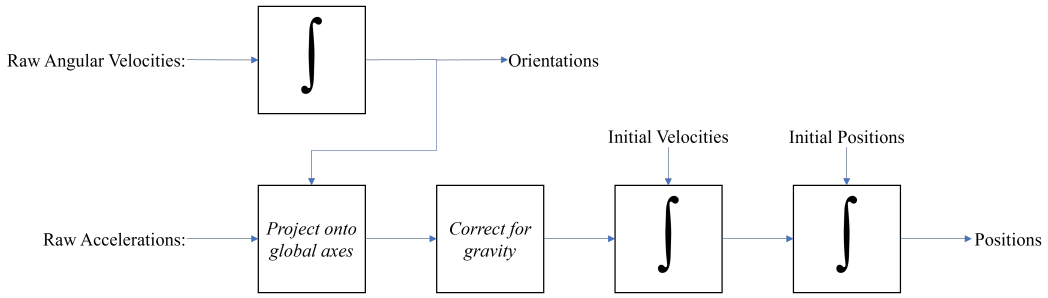

Figure 3.2: A graphical overview of a generic Inertial Navigation System. Gyroscope information is integrated and combined with accelerometer values, which are then integrated twice to achieve orientation and position data. Figure adapted from [1].

Algorithms with data-driven priors use more complicated techniques, such as sensor fusion or neural networks, along with large amounts of data to predict the trajectory of an IMU device. In this case, the INS process previously described can be further combined with magnetometer values in the form of Kalman filter correction to improve the double integration solutions, which will be further detailed in the next section.

3.3 Kalman Filtering

A common issue with double integration or similar processed used in IN is compounding noise. As such, inertial navigation is prone to growing error and internal biases and is therefore usually accompanied by a correction method to minimize these effects [33]. A typical method used to help correct these issues involves the fusion of measurements taken by several sensors. There are many types of sensor fusion that are comprised of varying levels of processing, input/output relationships, or configurations [34]. Perhaps one of the most popular and well-known type of sensor fusion involves using a Kalman filter (KF), where an internal model of a dynamic system is estimated and combined with various measurements to correct a prediction. For example, a common application involves the fusion of GNSS and IMU data. A trajectory is determined using IMU data and intermittent GPS information is used to update and correct the position. All Kalman filtering techniques

utilize an estimation of the states for both the prediction and the observation [35], which are assumed to be corrupted with white, zero mean, Gaussian noise [36].

3.3.1 General Dynamics

In general, Kalman filters can be used to evaluate systems with linear time-varying dynamics, which can be represented with the following notation:

$$x_{k+1} = A_k x_k + B_k u_k + G w_k \,, \tag{3.1}$$

$$y_k = C_k x_k + v_k \,. \tag{3.2}$$

These can be manipulated to be described with iterative cycles of prediction and filtering when provided an initial state vector and covariance matrix, as demonstrated by Ribeiro [36] and briefly detailed in the following sections.

3.3.2 Initial Conditions

Within a Kalman filter is a state vector, denoted as $\hat{x}$, and a covariance matrix, denoted as P. An initial value for the state vector must be given or determined as a function of inputs:

$$\hat{x}(0|0) = \bar{x}_0 \,. \tag{3.3}$$

An initial value for the covariance matrix must also be given or estimated:

$$P(0|0) = \Sigma_0 \,. \tag{3.4}$$

Once the initial conditions are set, a KF becomes an iterative process of prediction and filtering cycles until a final state is reached.

3.3.3 Prediction Cycle

The prediction cycle involves two separate steps: prediction of the state and prediction of the covariance. The intermediate state vector can be determined as a function of the previous state and some change in state:

$$\hat{x}(k+1|k) = A_k \hat{x}(k|k) + B_k u_k \,. \tag{3.5}$$

The intermediate covariance matrix is also calculated as the sum of the previous value and a set value of process noise covariance, denoted as Q, which is uncertainty that can be attributed to the change in state:

$$P(k+1|k) = A_k P(k|k) A_k^T + G_k Q_k G_k^T \,. \tag{3.6}$$

Once these intermediate values are determined, occurring either at regular or irregular time steps, the filtering cycle is applied to improve the prediction.

3.3.4 Filtering Cycle

Prior to the filtering cycle, a value of sensor noise covariance, denoted as R, must be determined. This can be described as uncertainty that can be attributed to measurement capabilities. Combined with the intermediate state covariance value, this creates a gain — denoted as K — for each time step:

$$K(k) = P(k+1|k) C_k^T [C_k P(k+1|k) C_k^T + R_k]^{-1} \,. \tag{3.7}$$

After the gain is calculated, the next state vector and covariance matrix can be updated with the following:

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K(k)\mu \,. \tag{3.8}$$

$$P(k+1|k+1) = [I - K(k) C_k] P(k+1|k) \,. \tag{3.9}$$

where μ is an estimation for the error between the prediction value and the true value.

An overview of the entire Kalman filtering process, from the previous state to the current state, is shown in Figure 3.3.

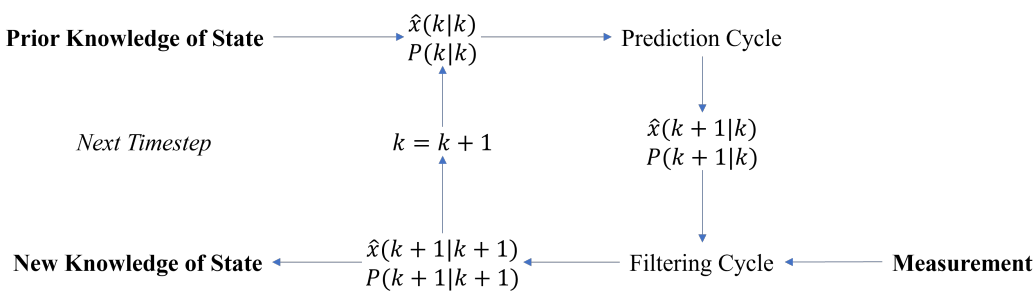

Figure 3.3: The Kalman Filtering Process. A previous estimate of the state is improved through a prediction model, then updated using a measurement through a filtering model to create an estimate of the current state.

3.3.5 Types of Kalman Filters

Both linear and non-linear discrete-time systems can be modeling using this format, using a Linear Kalman Filter (LKF) or Extended Kalman Filter (EKF) respectively [37]. There are also more advanced methods for non-linear systems that can account for stochastic uncertainties in complex computations such as the robust unscented Kalman filter (RUKF) [38]. Due to the linear or non-linear but analytically determinable nature of the INS and landmark/GNSS/UWB measurements taken for this project, and because the previous description is applicable to LKF, only Extended Kalman filters will be expanded on in further detail.

3.3.6 Non-Linear Dynamics

As stated previously, Extended Kalman filters can be used to model non-linear systems, which can be described with the following notation:

$$x_{k+1} = f(x_k) + w_k \,, \tag{3.10}$$

$$y_k = h(x_k) + v_k \,. \tag{3.11}$$

Using the same initial conditions and the same process outline, the previous filter can be expanded to non-linear dynamics by linearizing the system dynamics and observation dynamics around the previous state vector values. For this project, the prediction cycle is replaced with a neural network — which will be discussed in the next section.

3.4 Neural Networks

Neural networks (NNs) are interconnected structures of simple processing units called nodes, which have the ability to adapt or learn from information to which they are exposed. This concept is based on the biological nervous system, where neurons communicate with each other through electrical signals to create the basic computing structure of the brain, as shown in Figure 3.4.

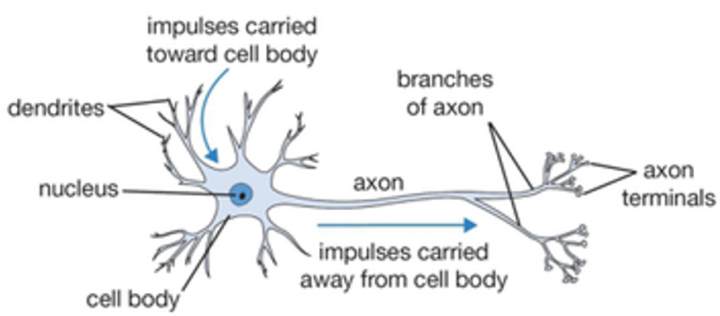

Figure 3.4: A neuron: the biological inspiration behind neural networks. This photo is licensed under CC BY-SA (creativecommons.org/licenses/by-sa/3.0/).

In the artificial equivalent, inputs to the system are summed in a weighted total. This is then given to an activation function to produce an activation value; if this value exceeds a certain threshold, an output is produced, as shown in Figure 3.5.

Artificial neurons can be combined in various structures and tuned to achieve the best possible outcome. Generally, sets of known input-output data are exposed to the neural network in a process called "training". Once the NN can achieve the correct, known output

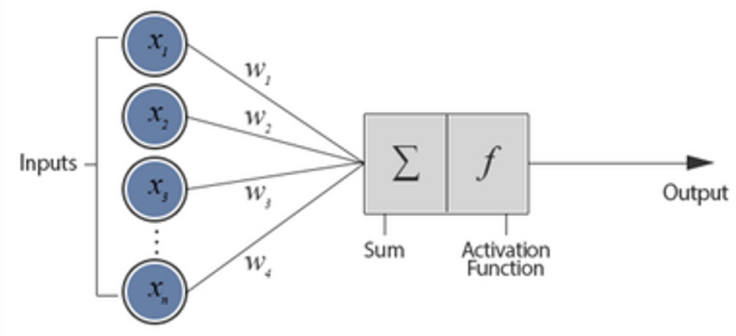

Figure 3.5: A simple representation of the basic data processes inside a neural network. This photo is licensed under CC BY-SA (creativecommons.org/licenses/by-sa/3.0/).

from any known set of inputs, it must be "tested" on sets of input-output data to which it has not previously been exposed. If the NN can still achieve the correct solution, then it is considered complete and can be utilized to correctly predict the behavior of similar real-world scenarios [39].

3.4.1 Deep Neural Networks

A more popular and commonly used type of neural network is a Deep Neural Network (DNN). Neural networks consist of series of smaller processing structures called layers: an input layer receives data and an output layer presents the final result. Between the input and output layers, information is propagated through one or more "hidden layers" — titled such because the user does not see the information processed by these layers. A traditional neural networks consists of only one hidden layer. Deep neural networks have more than one hidden layer, or a total of more than three layers. In current literature, DNNs can contain more than a thousand layers in their structure [40].

Neural networks have been utilized in many different capacities recently; their growing popularity can be attributed to their diverse, accurate, and effective nature. Solution have been created in numerous industries: agriculture, science, medicine, education, and much more. These applications can be classification, modeling, pattern recognition, or other necessary functions of everyday life [41].

3.4.2 Neural Networks for Inertial Navigation

One common use for neural networks includes inertial navigation, such as the case with this project. Although there are other methods to predict location based on IMU data, such as double integration that was mentioned in Section 3.2, these are often subject to noise and errors. One mitigation solution involves the use of Kalman filtering, as discussed in Section 3.3. However, another answer can be the incorporation of a neural network into the prediction process. This is especially helpful when outside sources, such as GNSS or ranging information, is unavailable or unreliable. Some of these methods use NN techniques to calibrate the raw IMU data and reduce the noise to output IMU data that can be much more reliably integrated. Other projects completely replace the integration process with a NN, where the raw IMU data is input to produce position data. This can be applied to typical robotic systems, as well as pedestrians, drones, and vehicles [42]. Two of the premiere, state-of-the-art examples of this structure include RIDI and IONet/OxIOD.

The first inertial navigation algorithm of its type, RIDI [43], classifies human motion into broad categories (standing, walking, turning) then regresses walking velocities while using standing and turning information to compensate for other changes in device position and orientation. After the correction, the industry standard practice of double integration is used to determine position from acceleration.

An important advancement in indoor navigation, IONet [44] developed a deep neural network framework that segments large amounts of inertial data into smaller pieces, which are then subject to less drift error. This network is accompanied by a large dataset — called OxIOD [45] — for training and testing purposes, which contains raw inertial measurements of indoor trajectories of people and trolley tracking. This contains over 158 sequences, totaling over 42 km in distance, and was the largest inertial dataset at the time. The other primary feature is the ground-truth labels applied to each sequence, allowing for error calculations to be confirmed for new applications. The sequences are also varied — the IMU sensors are held in different manners, the motion speed is changed, and numerous combinations of phones and users were tested. This dataset has been used in other learning-based navigation methods to confirm the research methodology.

However, another newer neural network based solution that completely replaces the traditional double integration process with a neural network is the Robust Neural Inertial Network, or RoNIN [2]. The authors of the RoNIN neural network system compared their proposed solution to the previously discussed RIDI [43] and IONet [44] procedures, as well as naive double integration (NDI) and pedestrian dead reckoning (PDR) methods. Both the absolute trajectory error and relative trajectory error [46] are reported in the RoNIN publications [2], and the absolute trajectory error is repeated here in Table 3.1 for discussion.

Table 3.1: The absolute trajectory error evaluation of the paths predicted using five competing methods: Naive Double Integration (NDI), Pedestrian Dead Reckoning (PDR), RIDI, IONet, and RoNIN on the RoNIN dataset. The best ATE is shown in green, the second-best ATE is shown in yellow, the third-best ATE is shown in red. © 2020 IEEE.

	NDI	PDR	RIDI	IONet	RoNIN		
					ResNet	LSTM	TCN
RoNIN Dataset	458.06	27.67	15.66	32.03	5.14	5.32	5.70

The following section details RoNIN, which later becomes the basis of the prediction network created in this thesis.

3.5 RoNIN

The developers of RoNIN [2] expand on the current state of the art to create a next-generation inertial navigation database with increased scale, diversity, and fidelity. It contains over 42 hours of data with 100 human subjects performing natural motions such as walking, sitting, or wandering with the IMU sensor inside a bag or pocket, or held in a hand. A two-device system was used to create ground-truth trajectories of the person carrying the device, allowing the neural network to estimate the person's trajectory — rather than the trajectory of the IMU device.

RoNIN is comprised of three neural network architectures: ResNet, LSTM, and TCN. The data process of these networks is shown in Figure 3.6. For the purposes of this research, the LSTM was selected as it is the most suitable for limited resource applications while remaining within the required accuracy of the project.

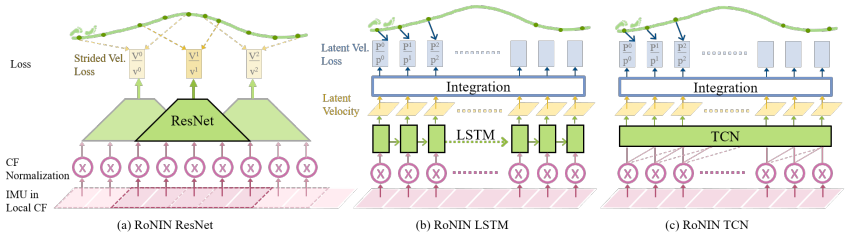

Figure 3.6: The RoNIN ResNet, LSTM, and TCN Neural Networks [2]. © 2020 IEEE.

A long short-term memory (LSTM) neural network is a dynamic system. They are a type of recurrent neural network; these have circular connections between higher- and lower-layer neurons that allow data to be propagated from earlier stages to the current stage. In short: these neural networks have memory. The network used in this project is specifically a bidirectional LSTM network, which propagates data from both past to present and future to present for increased data processing ability [47].

Chapter 4

Methodology

This chapter details the methods used to create the proposed people localization system. An overview of the neural network-based prediction and Kalman filter estimation process, from the previous state to the current state, is shown in Figure 4.1.

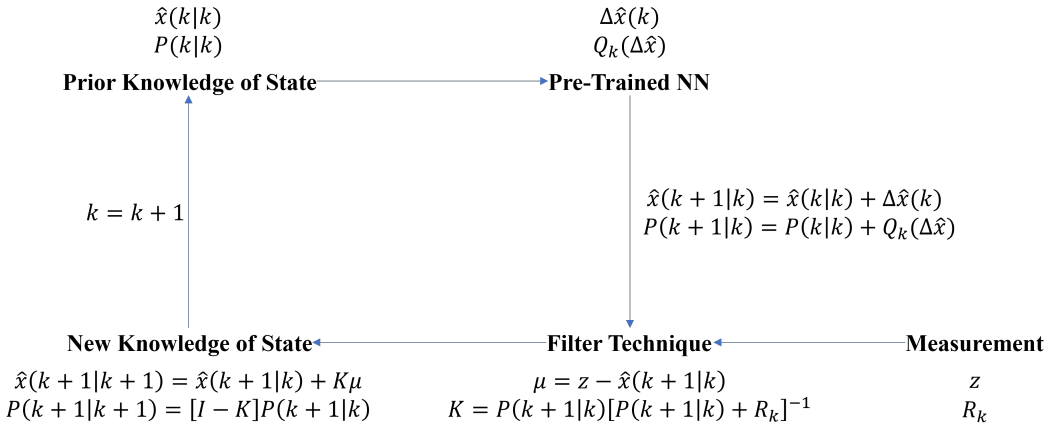

Figure 4.1: The Overall Estimation Process. A previous estimate of the state is improved through a prediction model — comprised of a pretrained neural network, then updated using a measurement through a filtering model — comprised of Kalman filtering techniques, to create an estimate of the current state.

The next sections begin with a description of the design of the position and orientation prediction functions, followed by an outline of the process used to determine the covariance values. Then, a discussion is written for the formation of the optional update techniques.

4.1 Estimation of the Position

For the estimation of the position of a moving person, the IMU data from a mobile device carried by the person is processed using a pre-trained bi-directional LSTM neural network [2]. A set of data for a discrete period of time, in the form of acceleration, gyroscope, and game rotation vector data, is provided to the neural network as the inputs. Then, the corresponding prediction of the position in two dimensions is calculated as the output for that same period of time. A summary of this data flow in this network is shown in Figure 4.2.

Figure 4.2: The input/output data flow for the bi-directional LSTM neural network used for the position estimation.

4.2 Estimation of the Orientation

For the estimation of the orientation, The IMU data is processed using a different pre-trained LSTM neural network [2]. Similar to the position estimation, the acceleration, gyroscope, and game rotation vector data are provided to the network as inputs, and the corresponding heading is calculated as the output. A summary of this data flow in this network is shown in Figure 4.3.

Figure 4.3: The input/output data flow for the LSTM neural network used for the orientation estimation.

For each estimation process, the same data is input into separate neural networks to create the predictions. This process is described in Algorithm 1.

Algorithm 1: Pseudo-code for the Prediction Process

Input :

$$acce = \begin{bmatrix} acce_x & acce_y & acce_z \end{bmatrix}_{M\times 3}$$

$$gyro = \begin{bmatrix} gyro_x & gyro_y & gyro_z \end{bmatrix}_{M\times 3}$$

$$grv = \begin{bmatrix} grv_\omega & grv_x & grv_y & grv_z \end{bmatrix}_{M\times 4}$$

Output:

$$\hat{x}(k+1|k) = \begin{bmatrix} x_{Pred} \\ y_{Pred} \end{bmatrix}_{2\times 1}$$

$$P(k+1|k) = \begin{bmatrix} Var(x_{Pred}) & Cov(x_{Pred}, y_{Pred}) \\ Cov(x_{Pred}, y_{Pred}) & Var(y_{Pred}) \end{bmatrix}_{2\times 2}$$

$$\theta_k = \begin{bmatrix} \theta_{Pred} \end{bmatrix}_{1\times 1}$$

initialize:

$\hat{x}(0|0) = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$

$P(0|0) = \begin{bmatrix} Var(x_0) & Cov(x_0, y_0) \\ Cov(x_0, y_0) & Var(y_0) \end{bmatrix}$

$\Delta x, \Delta y \leftarrow \boldsymbol{Bi\text{-}LSTM}(acce, gyro, grv)$

$Q_k = \sqrt{(\Delta x)^2 + (\Delta y)^2} * \begin{bmatrix} Q_{Pos} & 0 \\ 0 & Q_{Pos} \end{bmatrix}$

$\hat{x}(k+1|k) = \hat{x}(k|k) + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$

$P(k+1|k) = P(k|k) + Q_k$

$\theta_k \leftarrow \boldsymbol{LSTM}(acce, gyro, grv)$

4.3 Estimation of the Position Covariance

To determine the covariance with respect to the changing position, a set path was determined in a controlled environment with landmarks in specific locations. Data collection was initiated along the path, and a marker was placed within the data whenever a landmark was passed. Using this method, distinct straight-line segments were created of varying lengths to mathematically determine the rate at which the error grows. This process was repeated until a large dataset of segments was collected, which was then processed to determine the standard deviation per unit length — to be later transformed into covariance values.

To further analyze the prediction determined in the previous section through the use of Kalman filter techniques, a covariance value for this state must be determined. For the position system, it was observed that the covariance of the position prediction grows as a function of both distance and time:

$$Cov_{overall} = Cov_{distance} + Cov_{time} \,. \tag{4.1}$$

Therefore, several trials were performed to assess the rate at which the covariance of the position grows with respect to distance and time. This process along with the completed trials are discussed further in Chapter 5.

4.4 Estimation of the Orientation Covariance

To determine the covariance with respect to the changing orientation, a set path was determined in a controlled environment with easily separable hallways of differing angles. The halls were split into straight-line segments and assigned a known heading angle within the global system. The error between the predicted heading and actual heading was calculated along the path to determine the rate at which the accuracy changes. Once a large dataset of information was collected, it was processed to determine the standard deviation per unit length — to be later transformed into covariance values.

In order to evaluate the heading prediction, a separate value for the covariance of the

orientation must also be determined. For the orientation system, it was determined that the covariance of the heading prediction changes only as a function of distance:

$$Cov_{overall} = Cov_{distance} \,. \tag{4.2}$$

The same trial used to evaluate the position prediction was analyzed to assess the rate that the covariance of the heading grows with respect to distance. These evaluations are described in detail in Chapter 5.

4.5 Update Techniques

Once the characteristics of the system are determined, a linear Kalman filter or extended Kalman filter technique for correcting the prediction can be utilized. To achieve this, the state vector and state covariance matrix predicted using the pre-trained neural network are considered intermediate values. These values are then updated using measurement data to produce a better result.

The method of correction varies based on the respective outside measurement type. For this project, these were determined to be landmark locations, GNSS positioning, and UWB range information. Many trials were performed for each sensor type to tune the parameters of the system and verify the procedure.

To complete this evaluation, several other pieces of information are required. This includes a model of the relationship between the input values as well as an error formulation between the prediction value and measurement value. These values must be systematically determined for each sensor circumstance, and applied the procedure at each instance of a sensor measurement. A review of the approach used to complete and implement these stages is given in Chapter 5. This process is also described in Algorithm 2.

Algorithm 2: Pseudo-code for the Correction Process

Input :

$$\hat{x}(k+1|k) = \begin{bmatrix} x_{Pred} \\ y_{Pred} \end{bmatrix}_{2\times 1}$$

$$P(k+1|k) = \begin{bmatrix} Var(x_{Pred}) & Cov(x_{Pred}, y_{Pred}) \\ Cov(x_{Pred}, y_{Pred}) & Var(y_{Pred}) \end{bmatrix}_{2\times 2}$$

$$z$$

Output:

$$\hat{x}(k+1|k+1) = \begin{bmatrix} x_{Corr} \\ y_{Corr} \end{bmatrix}_{2\times 1}$$

$$P(k+1|k+1) = \begin{bmatrix} Var(x_{Corr}) & Cov(x_{Corr}, y_{Corr}) \\ Cov(x_{Corr}, y_{Corr}) & Var(y_{Corr}) \end{bmatrix}_{2\times 2}$$

/* $TYPE(z) \in (Landmark, GNSS, UWB)$ */

if *a measurement value is taken* **then**

  if $TYPE(z) = Landmark$ **then**

    $y = \begin{bmatrix} x_{Pred} \\ y_{Pred} \end{bmatrix}_{2\times 1}$

    $z = \begin{bmatrix} x_{Land} \\ y_{Land} \end{bmatrix}_{2\times 1}$

    $C = I_{2\times 2}$

    $R_k = \begin{bmatrix} R_{x_{Land}} & 0 \\ 0 & R_{y_{Land}} \end{bmatrix}_{2\times 2}$

  else if $TYPE(z) = GNSS$ **then**

    $y = \begin{bmatrix} x_{Pred} \\ y_{Pred} \end{bmatrix}_{2\times 1}$

    $z = \begin{bmatrix} x_{GNSS} \\ y_{GNSS} \end{bmatrix}_{2\times 1}$

    $C = I_{2\times 2}$

    $R_k = \begin{bmatrix} R_{x_{GNSS}} & 0 \\ 0 & R_{y_{GNSS}} \end{bmatrix}_{2\times 2}$

  else if $TYPE(z) = UWB$ **then**

    $y = \left[\sqrt{(x_{Pred} - x_{Sens})^2 + (y_{Pred} - y_{Sens})^2}\right]_{1\times 1}$

    $z = \left[d_{UWB}\right]_{1\times 1}$

    $C = \dfrac{\partial y}{\partial \hat{x}} = \begin{bmatrix} \dfrac{(x_{Pred} - x_{Sens})}{y} & \dfrac{(y_{Pred} - y_{Sens})}{y} \end{bmatrix}_{1\times 2}$

    $R_k = \left[R_{d_{UWB}}\right]_{1\times 1}$

  $S = CPC^T + R$

  $K = PC^T S^{-1}$

  $\mu = z - y$

  $\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K\mu$

  $P(k+1|k+1) = P(k+1|k) - KCP$

4.6 Overall System

With the prediction and filtering cycles decided, these techniques can be used to create and improve the estimate for the system. However, the raw data must first be pre-processed into a more suitable form. Because the standard rate of capture for each sensor is different, the data values must be interpolated to create a common time label. Then, all inputs will have an equivalent time vector and the time column can be eliminated to reduce the matrix size. This process is described in Algorithm 3.

Algorithm 3: Pseudo-code for the Pre-Processing

Input :

$acce_{raw} = \begin{bmatrix} t1 & acce_x & acce_y & acce_z \end{bmatrix}_{M1\times 4}$

$gyro_{raw} = \begin{bmatrix} t2 & gyro_x & gyro_y & gyro_z \end{bmatrix}_{M2\times 4}$

$grv_{raw} = \begin{bmatrix} t3 & grv_\omega & grv_x & grv_y & grv_z \end{bmatrix}_{M3\times 5}$

where $M\#$ is the number of time-steps within the data
and $M1 \neq M2 \neq M3$
and $t\#$ is the time associated with each measurement

Output:

$acce = \begin{bmatrix} acce_x & acce_y & acce_z \end{bmatrix}_{M\times 3}$

$gyro = \begin{bmatrix} gyro_x & gyro_y & gyro_z \end{bmatrix}_{M\times 3}$

$grv = \begin{bmatrix} grv_\omega & grv_x & grv_y & grv_z \end{bmatrix}_{M\times 4}$

where M is the number of time-steps within the data

1 $\delta t = 0.005 sec$
2 $t[0] \leftarrow \boldsymbol{max}(t1[0], t2[0], t3[0])$
3 $t[M] \leftarrow \boldsymbol{min}(t1[M1], t2[M2], t3[M3])$
4 $acce \leftarrow \boldsymbol{interpolate}(acce_{raw}[:, 1:3]$ from $t[0]$ to $t[M]$ for $\delta t)$
5 $gyro \leftarrow \boldsymbol{interpolate}(gyro_{raw}[:, 1:3]$ from $t[0]$ to $t[M]$ for $\delta t)$
6 $grv \leftarrow \boldsymbol{interpolate}(grv_{raw}[:, 1:3]$ from $t[0]$ to $t[M]$ for $\delta t)$

The data is then segmented into 3-second sections (600 data points at a frequency of 200 Hz) to simulate an online procedure. For each iteration, the prediction process coverts the IMU data into an intermediate estimation and the correction process update occurs to produce a final result for the segment. The position and covariance values are saved and the process is repeated for the length of the input data. A complete overview of this process is shown in Algorithm 4.

Algorithm 4: Pseudo-code for the Overall Process

Input :

$acce_{raw} = \begin{bmatrix} t1 & acce_x & acce_y & acce_z \end{bmatrix}_{M1\times 4}$

$gyro_{raw} = \begin{bmatrix} t2 & gyro_x & gyro_y & gyro_z \end{bmatrix}_{M2\times 4}$

$grv_{raw} = \begin{bmatrix} t3 & grv_\omega & grv_x & grv_y & grv_z \end{bmatrix}_{M3\times 5}$

where $M\#$ is the number of time-steps within the data
and $M1 \neq M2 \neq M3$
and $t\#$ is the time associated with each measurement

Output:

$Position = \begin{bmatrix} x \\ y \end{bmatrix}_{2\times M}$

$Covariance = \begin{bmatrix} Var(x_0) & Cov(x_0, y_0) \\ Cov(x_0, y_0) & Var(y_0) \end{bmatrix}_{2\times M}$

initialize:

$Position = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

$Covariance = \begin{bmatrix} Var(x_0) & 0) \\ 0 & Var(y_0) \end{bmatrix}$

$N = 600$

$acce, gyro, grv \leftarrow PreProcessing(acce_{raw}, gyro_{raw}, grv_{raw})$

for $i = [N, range(len(input) + N), N]$ **do**

  $acce_{iter} = acce(i - N : i, :)$

  $gyro_{iter} = gyro(i - N : i, :)$

  $grv_{iter} = grv(i - N : i, :)$

  $\hat{x}(k+1|k), P(k+1|k) \leftarrow Prediction(acce_{iter}, gyro_{iter}, grv_{iter})$

  $\hat{x}(k+1|k+1), P(k+1|k+1) \leftarrow Correction(\hat{x}(k+1|k), P(k+1|k), z)$

  $Position = \textbf{\textit{concatenate}}(Position,\ \hat{x}(k+1|k+1))$

  $Covariance = \textbf{\textit{concatenate}}(Covariance,\ P(k+1|k+1))$

end

An analysis was completed to determine the proper number of time steps in each iteration that would best represent the original prediction of the entire trajectory. To complete this, several data collection paths were chosen arbitrarily and the process was iterated for a varying number of time steps from $N = 200 : 700$. The absolute trajectory error, defined as the root mean squared error between two trajectories [46], was calculated for each prediction with respect to the full prediction without iteration. The absolute trajectory error for several paths are shown in Table 4.1. An example of a path and several predictions with varying time steps are shown in Figure 4.4. Although larger values performed better, it was also important to balance the accuracy with the update time to keep the iterations as close as possible. Using this, the time step was set to be $N = 600$.

Table 4.1: The Absolute Trajectory Error (ATE) for the iteration of the prediction using various time steps. The best ATE for each path is shown in green, the second best ATE is shown in yellow, the third best ATE is shown in red.

Time Steps (N)	Path 1	Path 2	Path 3	Path 4
200	1.60	1.06	1.30	2.20
300	0.966	1.14	0.373	1.71
400	0.254	0.731	0.760	1.82
500	0.571	0.511	0.614	0.752
600	0.167	0.490	0.352	1.47
700	0.421	0.929	0.466	0.718

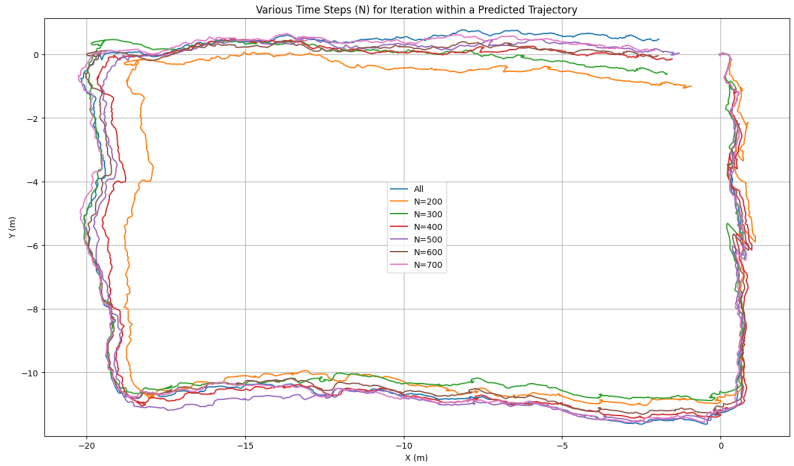

Figure 4.4: The predicted trajectories with varying time steps (N) for iteration. The values range from $N = 200$ to $N = 700$ and the path without iteration is shown in blue.

Chapter 5

Results

As detailed throughout this thesis, the main objective of the research is to develop a system that estimates the trajectory of a person using sensor data collected with a mobile device. In combination with the sensors internal to the device (i.e., the accelerometer, gyroscope, and magnetometer), other information such as landmark locations known by the person being localized, limited or degraded GNSS positions, or range information between a stationary UWB sensor and the mobile device were used to localize the person. The experimental setup and procedure as well as each of the scenarios of prediction and estimation with secondary information will be evaluated in the following sections.

5.1 Experimental Setup

Prior to a discussion on the results of the proposed system, the hardware and software used in this project must be discussed. An Android mobile device, specifically a Samsung Galaxy S20, was selected as the device to perform data collection. A computer running WSL Ubuntu 22.04 with Python software was used to process the data.

As previously discussed in Section 3.5, the prediction solution is based on pre-trained neural networks [2] that convert raw IMU data into position increments and orientation values that are used in the prediction step of a recursive filtering approach. To collect

the IMU data, the mobile device application (app) that was used to create the original dataset was provided by the researchers of the neural networks. However, the dataset and neural networks were originally designed for indoor trajectory prediction. Therefore, the app included information that is specialized for buildings such as a step counter and a magnetic field sensor. In outdoor scenarios, this data does not provide any additional value to the trajectory estimation. A new data recorder application was created to collect only the necessary data in order to be more efficient for the outdoor context of this project. A comparison of the previous and current sensor data types is shown in Table 5.1.

Table 5.1: A comparison of the Android sensor data collected with each application (app) for use in the prediction of the position. *The Rotation Vector data is collected with the current application but is not used as an input to the neural networks.

Sensor	RoNIN	Proposed Method
Gyroscope	Yes	Yes
Gyroscope Uncalibrated	Yes	-
Accelerometer	Yes	Yes
Linear Acceleration	Yes	-
Gravity	Yes	-
Magnetic Field	Yes	-
Magnetic Field Uncalibrated	Yes	-
Rotation Vector	Yes	Yes*
Game Rotation Vector	Yes	Yes
Geomagnetic Rotation Vector	Yes	-
Step Counter	Yes	-
Pressure	Yes	-
Marker	-	Yes
GPS	Yes	Yes
UWB Range	-	Yes

In addition to the required sensors, the sensor data used in the correction process was also collected in this app. The “Marker” data represents a button that is used to collect specific time information at specified landmark locations. The UWB Range includes the distance from one or more UWB anchors to the corresponding tag carried with the mobile device. A summary of the data collection procedure is detailed in the next section.

5.2 Experimental Procedure

The procedure for data collection is relatively straightforward. To initialize the path, a button in the app is clicked to start the collection. During this period, it is essential to ensure the mobile device is held stationary, in a placement parallel to the ground, with the top of the phone pointed along the direction of travel. After approximately 5-10 seconds, the time corresponding to start of the motion is identified using the "Mark Location" button on the application. The path can then be walked; the person carrying the mobile device should travel at an average pace but can perform any natural human motions, such as texting, calling, or even placing the device in a pocket or bag. If the chosen path contains known landmark locations, the "Mark Location" button should be pressed at each known position. Once the path is complete, the "Mark location" button should be pressed once more followed by the stop button to finalize the collection. Once the trial is complete, the data read from each applicable sensor or source is written to a separate text file on the mobile device.

Once the IMU data collection period has ended and all information has been collected, the files must been transferred to a computer for compilation and further processing. The procedures and pseudo-code for this process were discussed at length in Chapter 4. Next, the evaluation process for the properties of the system and the results will be reviewed.

5.3 Evaluation of the Position Covariance

As mentioned in Section 4.3 and Equation (4.1), the covariance of the position prediction is a function of both the distance traveled along the path and the time during data collection. The trials performed to empirically calculate these values will be analyzed in the following subsections.

5.3.1 Covariance with Respect to Distance

In order to complete these trials, a pre-determined path was selected that could be consistently walked throughout many different instances to create a comparable dataset. Specifically, the path used was the ninth floor of the Engineering Sciences Building on the

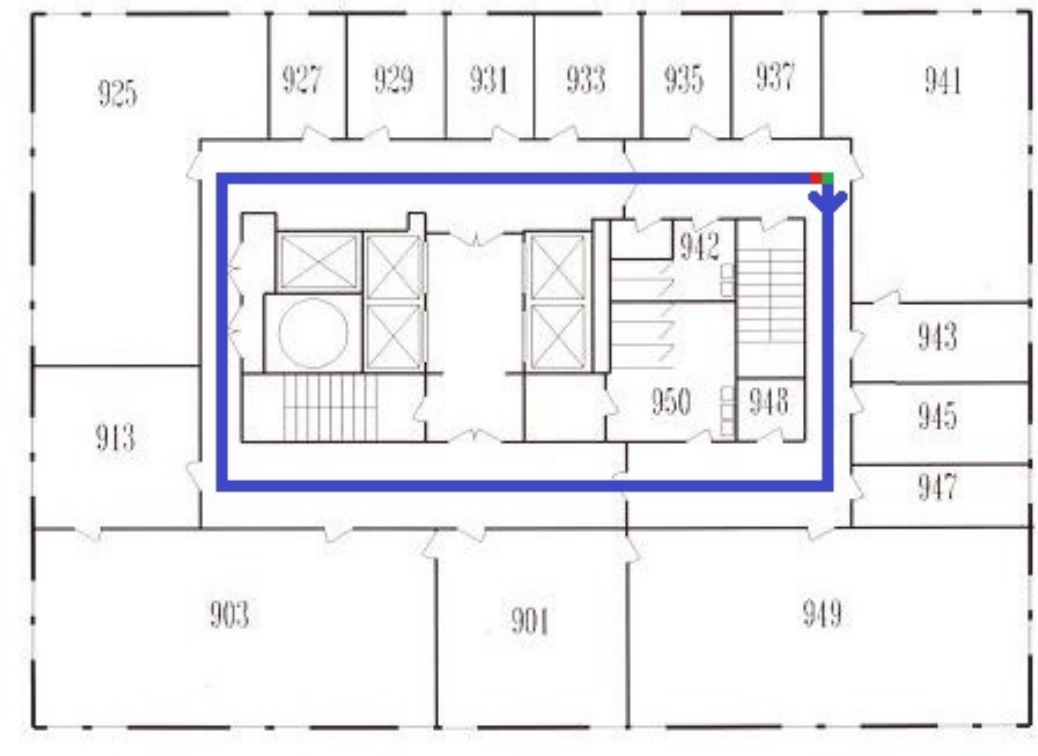

Figure 5.1: The specified path used to complete the trials for the covariance value located on the ninth floor of the Engineering Sciences Building at West Virginia University [3].

Evansdale Campus of West Virginia University. The hall in this floor forms a rectangle, which was advantageous for repeating multiple trials. A map of the Engineering Sciences Building ninth floor with the specified path is shown in Figure 5.1.

Along the hall, four corners and six landmarks placed on the ground were used to create ten straight segments of three different lengths. The individual segments along a singular hall could also be combined to produce longer-length segments for the purpose of this comparison, to create a total of six different groups for evaluation. The chosen landmarks, distances of the paths, and the division of the groups are shown in Figure 5.2.

The segments were then partitioned into corresponding groups for each length. The error of each segment was calculated as the difference between the predicted location and the actual location of the marker along the direction of travel. The average and standard deviation of the errors for each length were calculated, and the standard deviation of each

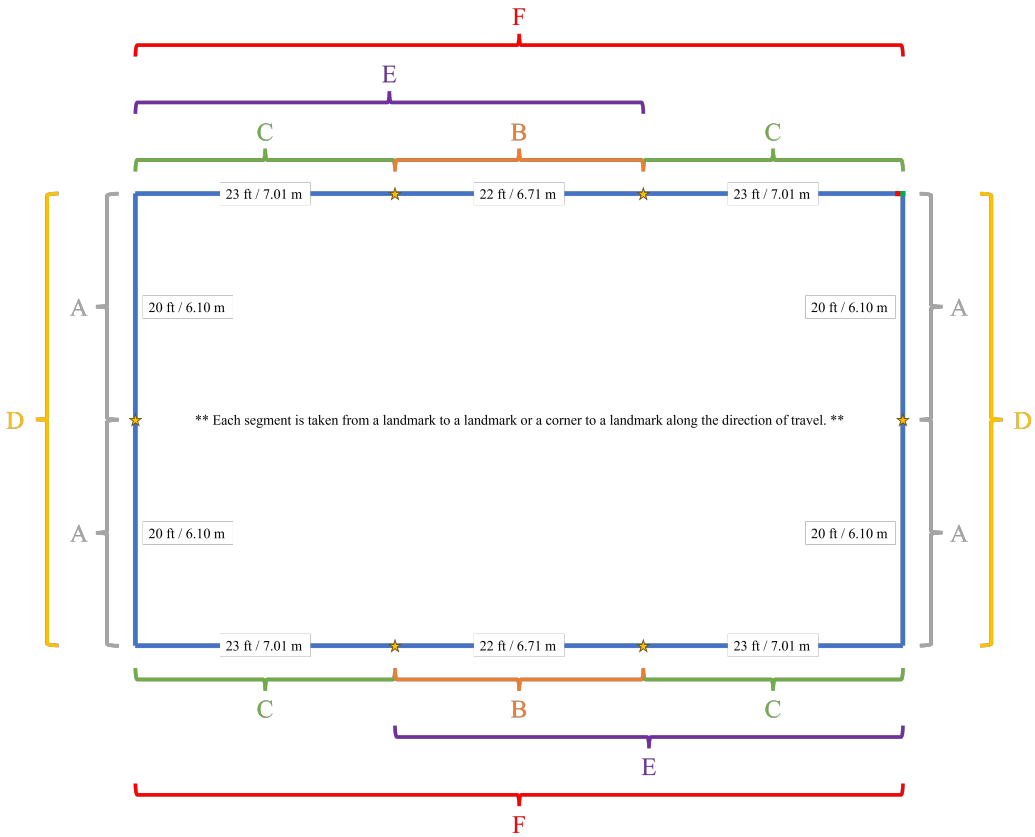

Figure 5.2: The chosen landmark locations and measurements of the path lengths of the ninth floor of the Engineering Sciences Building.

group was plotted against the length of the group, to determine a trendline for the standard deviation as a function of the distance. This trendline can be described by:

$$StDev_{distance} = 0.0336 \times (distance) + 0.149\,, \tag{5.1}$$

which is also shown in Figure 5.3.

During these trials, it was discovered that the error in the direction of travel was always relatively large compared to the error in the direction normal to the path, regardless of whether the segment was along the x direction or y direction. Therefore, for the calculation of the covariance matrix in terms of position with respect to two directions, the same value

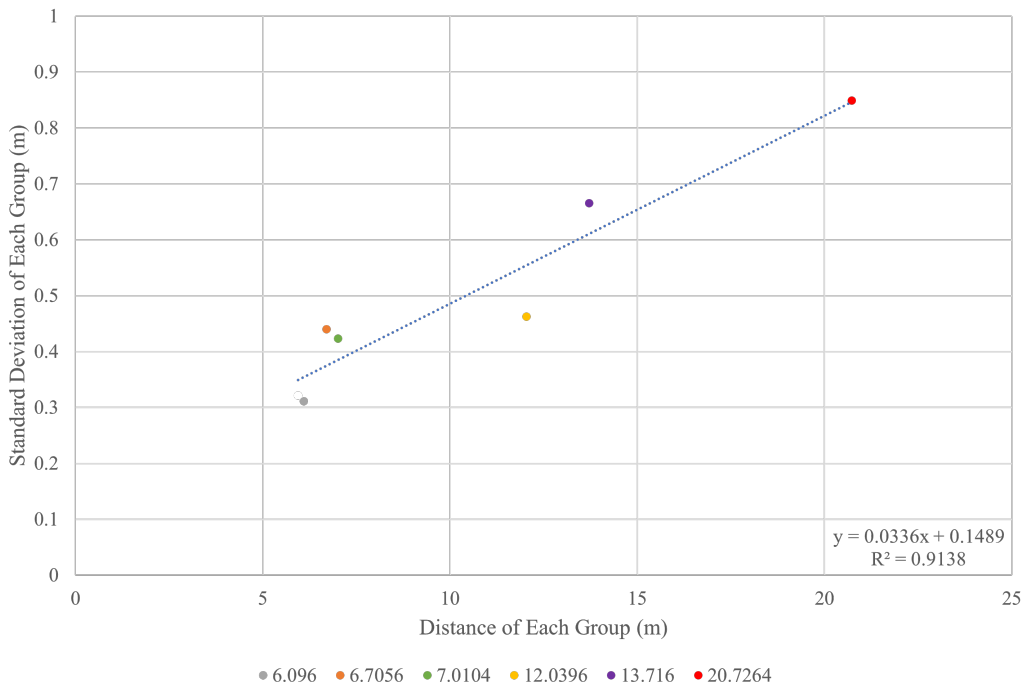

Figure 5.3: A plot of the standard deviation of the error values for each group of distances compared to the respective lengths. The trendline is shown on the graph, represented by the dashed line.

was given to x and y, which were also set to be independent of each other.

Also during these trials, the path predicted from the bi-LSTM neural network was consistently smaller in all directions (in x and y along the direction of travel) than the actual path in terms of predicted marker position compared to actual marker position. An evaluation was performed to calculate the actual distance traveled as a function of the predicted distance traveled on the same data used for the standard deviation trial. The trendline can be described with the following equation:

$$distance_{actual} = 1.346 \times (distance_{predicted}) + 0.198\,, \tag{5.2}$$

which is also shown in Figure 5.4.

Therefore, at each point a prediction occurs, the length for the prediction of the new position is scaled in each direction, to allow for a more accurate representation of the true

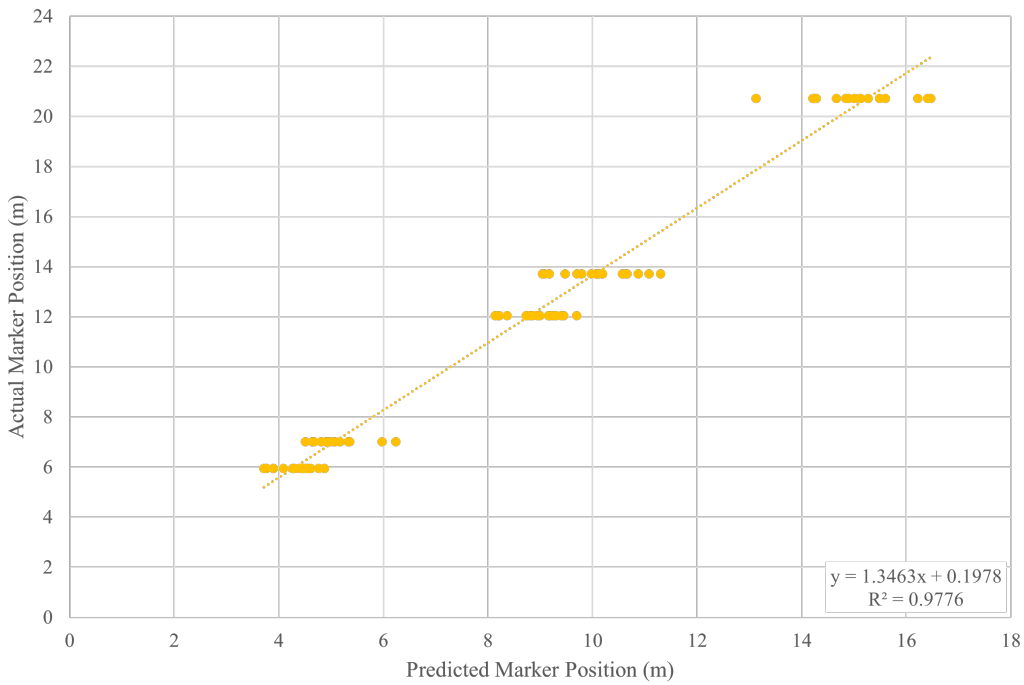

Figure 5.4: A plot of the actual marker position lengths compared to the predicted marker position lengths. The trendline is shown on the graph, represented by the dashed line.

trajectory.

An example of the predicted trajectory for one of the trials, including the predicted landmark positions and the actual landmark positions, is shown in Figure 5.5.

5.3.2 Covariance with Respect to Time

A separate trial was conducted to determine the effects of elapsed time on the overall prediction and noise of the system. For several trials of varying time, data was collected by creating motion of the mobile device while the person holding the device remained stationary. Because the mobile device had motion, IMU measurements were collected; the data was then run through the bi-LSTM neural network to create a prediction for the position.

Because the person remained motionless, the true position is considered to be zero for all values. Therefore, any different prediction of the position was considered to be an error. These error values were relatively small and do not need to be considered or corrected in

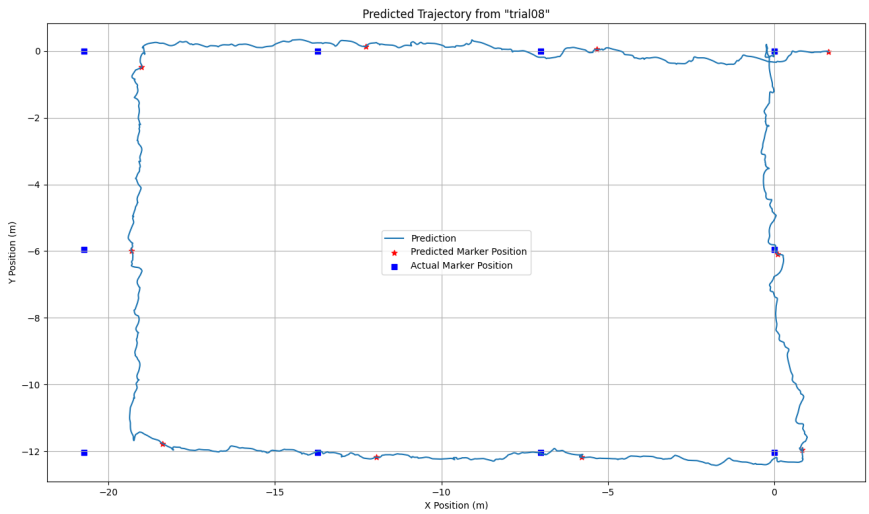

Figure 5.5: The predicted trajectory, predicted landmark positions, and actual landmark positions for a selected dataset.

any form for the previous evaluation. However, the standard deviation of this dataset was moderate and should be compensated in the covariance model. Therefore, a set value of standard deviation was determined (to be later added to the calculated total) as shown in the following equation:

$$StDev_{time} = 1.658\,, \tag{5.3}$$

which is considered an effective correction for the contents of this project.

5.3.3 Total Position Covariance

As shown in Equation (4.1), the total covariance is the sum of the covariance due to distance and the covariance due to time. To calculate this value, the overall standard deviation must first be determined, as shown:

$$StDev_{overall} = 0.0366 \times (distance) + 0.149 + 1.658\,, \tag{5.4}$$

where the distance is the total Euclidean distance traveled from the previous position to the current state, which is the total length of the iteration at the current state. Therefore, for this system, the overall covariance can be described by:

$$Cov_{overall} = (0.0366 \times (distance))^2 + (0.149 + 1.658)^2 \,. \tag{5.5}$$

5.4 Evaluation of the Orientation Covariance

As mentioned in Section 4.4 and Equation (4.2), the covariance of the orientation prediction is a function of the distance travel from the start of the path to the current state. The trials that were previously performed and analyzed, as described in Section 5.3, were reassessed to empirically evaluate these values.

5.4.1 Covariance with Respect to Distance

To determine the covariance of the orientation with respect to distance, the same information used previously was processed through the LSTM neural network to create a series of predictions for the orientation of the person carrying the mobile device. The same path shown in Figure 5.1 on the ninth floor of the Engineering Sciences Building on the West Virginia University campus was used for the data collection. However, in this case, the evaluation was based on the error between the known heading and the estimated heading. To determine this value, the path was split into four segments along the halls and each segment was assigned a known heading along the desired path. This heading was decided with respect to the positive y direction. The division of the halls and their respective known headings as well as the chosen segments for evaluation are shown in Figure 5.6.

5.4.2 Total Orientation Covariance

The error was calculated as the difference between the known heading and the estimated heading for each location along the path. Then, the average and standard deviation of the error were determined, and the standard deviation of each group was plotted against the

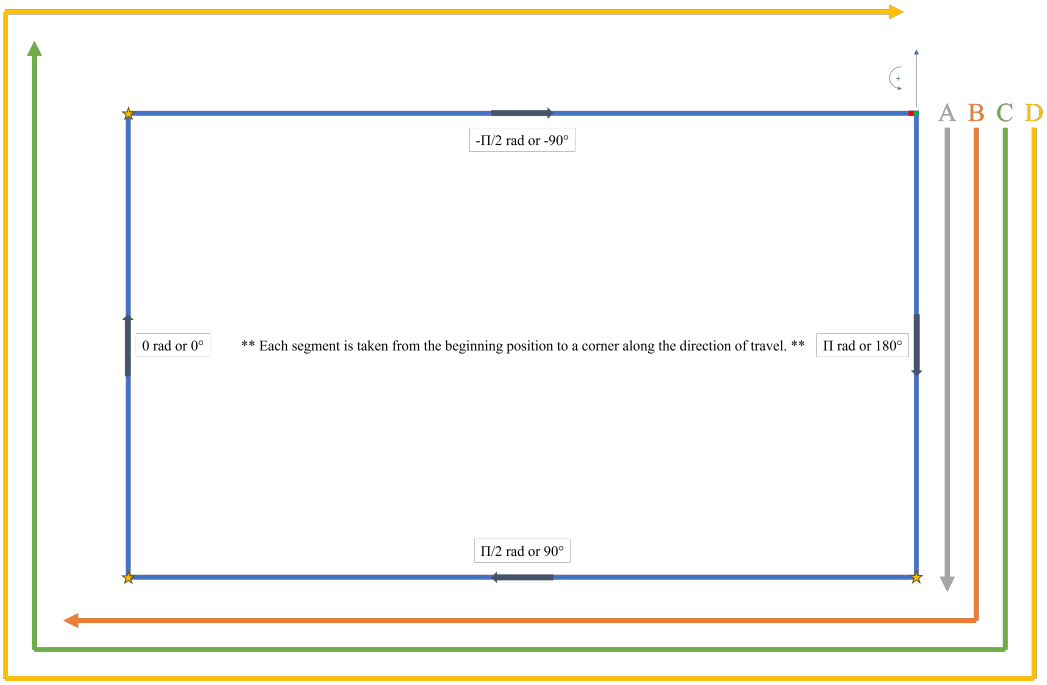

Figure 5.6: The chosen segments and known orientation of the paths of the ninth floor of the Engineering Sciences Building.

length of each group, to again determine a trendline for the standard deviation as a function of the distance. This trendline can be described with the following equation:

$$StDev_{distance} = -0.0606 \times (distance) + 20.783\,, \tag{5.6}$$

which is also shown in Figure 5.7. The covariance of the system can therefore be described as

$$Cov_{distance} = (-0.0606 \times (distance))^2 + (20.783)^2\,. \tag{5.7}$$

An example of the predicted headings for one of the trials, including the predicted landmark positions and the actual landmark positions, is shown in Figure 5.8.

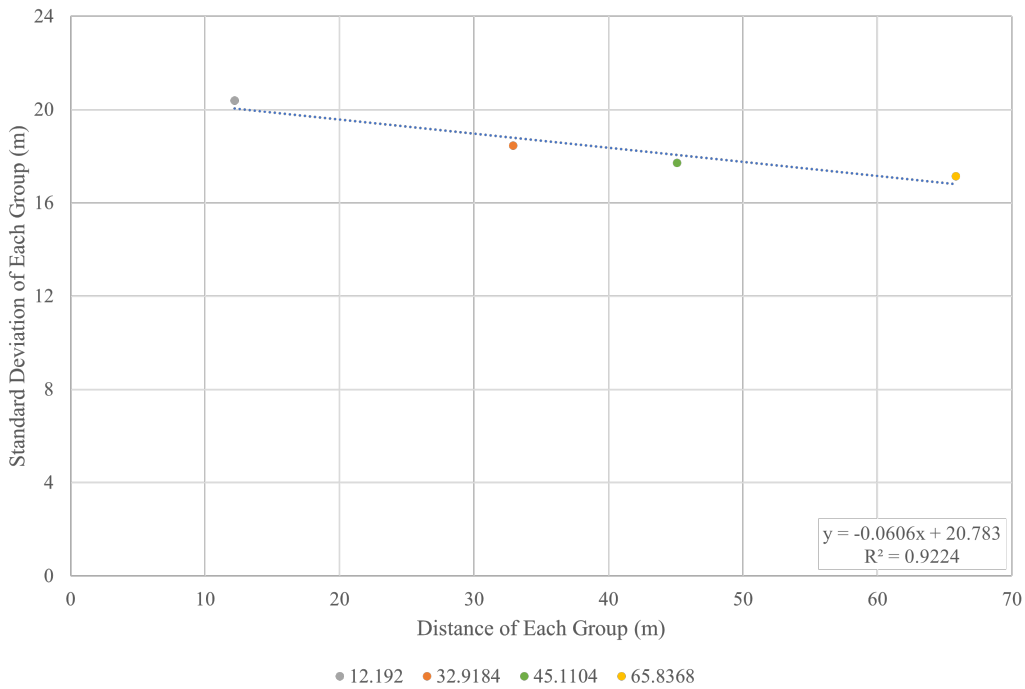

Figure 5.7: A plot of the standard deviation of the error values for each group of orientations compared to the respective lengths. The trendline is shown on the graph, represented by the dashed line.

5.5 Application of the Update Technique

Equation (3.3) shows that, in a recursive filtering estimator, the initial state vector must be given or determined. For this project, the initial state is determined using GPS information (when available) and the Rotation Vector values that were collected using the previously discussed app running on an Android phone. For this project, the Rotation Vector values refer to the data collected from a sensor on an Android device that reports the orientation relative to the East-North-Up coordinates frame. The format of the initial state vector is as follows:

$$\hat{x}(0|0) = \begin{bmatrix} x_{GPS}(0) \\ y_{GPS}(0) \\ \theta_{RV}(0) \end{bmatrix}, \tag{5.8}$$

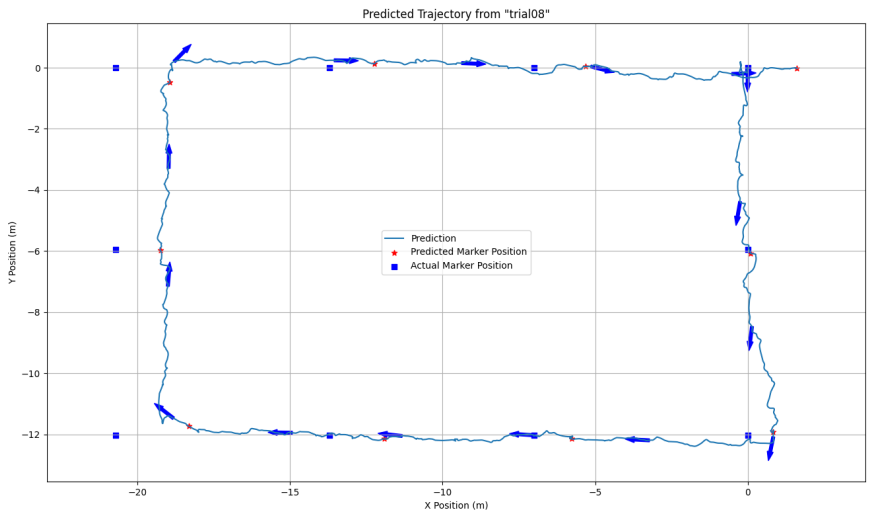

Figure 5.8: The predicted trajectory, predicted headings, predicted landmark positions, and actual landmark positions for a selected dataset.

where x_{GPS} and y_{GPS} are longitude and latitude values (or set to another known value when GPS is not available) and θ_{RV} is calculated as:

$$\theta_{RV} = \arctan\left(\frac{2.0 \times (w \times z + x \times y)}{w \times w + x \times x - y \times y - z \times z}\right), \tag{5.9}$$

using a quaternion to rotation matrix conversion [48] and a rotation matrix to Euler angles conversion [49].

Because the x position and y position values were determined to be independent from each other, and because the heading calculation uses a different neural network, the model for the system can be considered 3×3 identity.

Equation (3.4) shows that the initial state covariance matrix must be given or estimated. For this project, the covariance values were calculated in sections 5.3 and 5.4. The format

of this matrix is as follows:

$$P(0|0) = \begin{bmatrix} Var_x & 0 & 0 \\ 0 & Var_y & 0 \\ 0 & 0 & Var_\theta \end{bmatrix}. \tag{5.10}$$

Because the data collection system and IMU information is the same for the position and heading estimations, and the evaluation was independent of the axis, the values for Var_x and Var_y are estimated to be equivalent. The covariance formula for the position given in Equation (5.5) has two parts: a variable value (slope) and a constant value (intercept) which are summed to produce the overall value. The intercept value is determined to be the estimate for the initial state covariance for the position, and the amount is set as:

$$Var_{P_{Pos}} = (0.149 + 1.658)^2 \,, \tag{5.11}$$

where the initial state covariance matrix for the position values are:

$$P(0|0)_{Pos} = Var_{P_{Pos}} \,. \tag{5.12}$$

The covariance formula for the orientation given in Equation (5.7) also has two parts: the variable value and the constant value. The intercept value is determined to be the estimate for the initial state covariance for the orientation. Because this information is not easily determinable, it was assumed to have a value that is an over-approximation of the true value to ensure adequate representation of the relationships between variables, and the amount is set as:

$$Var_{P_{Ori}} = (20.783)^2 \,, \tag{5.13}$$

where the initial state covariance matrix for the orientation values are:

$$P(0|0)_{Ori} = Var_{P_{Ori}} \,. \tag{5.14}$$

The complete initial state covariance matrix can then be written as:

$$P(0|0) = \begin{bmatrix} Var_{P_{Pos}} & 0 & 0 \\ 0 & Var_{P_{Pos}} & 0 \\ 0 & 0 & Var_{P_{Ori}} \end{bmatrix}, \tag{5.15}$$

$$P(0|0) = \begin{bmatrix} (0.149 + 1.658)^2 & 0 & 0 \\ 0 & (0.149 + 1.658)^2 & 0 \\ 0 & 0 & (20.783)^2 \end{bmatrix}. \tag{5.16}$$

During the prediction cycle, a value for the process noise covariance matrix must be calculated. The same size as the state covariance matrix, this matrix represents the noise of the system that can be attributed to the change in state. For this project, the process noise covariance for the position is represented by the slope value determined in Equation (5.5). This value is variable based on the change in position and is calculated as:

$$Var_{Q_{Pos}} = (0.0366 \times (distance))^2 . \tag{5.17}$$

The process noise covariance for the orientation is represented by the slope value determined in Equation (5.7). However, because the calculated value is negative, the estimated value can be chosen to be some arbitrarily small value, such as half of the process noise covariance value for position. This estimate is also variable based on the prediction for the change in position and is estimated as:

$$Var_{Q_{Ori}} = (0.0366/2 \times (distance)^2) . \tag{5.18}$$

The complete process noise covariance matrix can be written as:

$$Q_k = \begin{bmatrix} Var_{Q_{Pos}} & 0 & 0 \\ 0 & Var_{Q_{Pos}} & 0 \\ 0 & 0 & Var_{Q_{Ori}} \end{bmatrix}, \tag{5.19}$$

$$Q_k = (distance) \times \begin{bmatrix} (0.0366)^2 & 0 & 0 \\ 0 & (0.0366)^2 & 0 \\ 0 & 0 & (0.0366/2)^2 \end{bmatrix}. \tag{5.20}$$

Prior to the filtering cycle, a value for the sensor noise covariance matrix must also be calculated. This information is related to the sensor measurement type. Thus the sensor measurement matrix can be different for the marker positions, GNSS locations, and UWB measurements. This matrix is set as a constant value for each evaluation, which are described in the following paragraphs.

The initial trial with the known landmark positions can have an estimated sensor noise covariance value because the precise positioning of the mapping of the landmark locations does not designate a specific empirical value. This value can be approximated to be equal to the initial state covariance value as:

$$R_k = P(0|0) = \begin{bmatrix} Var_{P_{Pos}} & 0 & 0 \\ 0 & Var_{P_{Pos}} & 0 \\ 0 & 0 & Var_{P_{Ori}} \end{bmatrix}. \tag{5.21}$$

For the second trial for the comparison between the GNSS locations, a separate estimation must be made. According to an empirical study [50], the value varies between $0.2\,m$ and $1\,m$ depending on the elevation angle of the coordinating satellite. As a generic approximation within this range, this value will be approximated as one half of the initial state covariance value as:

$$R_k = 0.5 \times P(0|0) = 0.5 \times \begin{bmatrix} Var_{P_{Pos}} & 0 & 0 \\ 0 & Var_{P_{Pos}} & 0 \\ 0 & 0 & Var_{P_{Ori}} \end{bmatrix}. \tag{5.22}$$

An new statistical survey was performed for the final trial with the UWB sensors in the given experimental conditions to determine the last sensor noise covariance value. A UWB anchor was placed at a known location and a tag was moved in one foot increments, with

a set of measurements being taken at each distance. The error between the actual distance and the measured distance was calculated, and the standard deviation of each distance was plotted against the length, to determine a trendline for the standard deviation as a function of the distance. This trendline can be described with the following equation:

$$StDev_{UWB} = 0.0004 \times (distance) + 0.0233\,, \tag{5.23}$$

which is also shown in Figure 5.9.

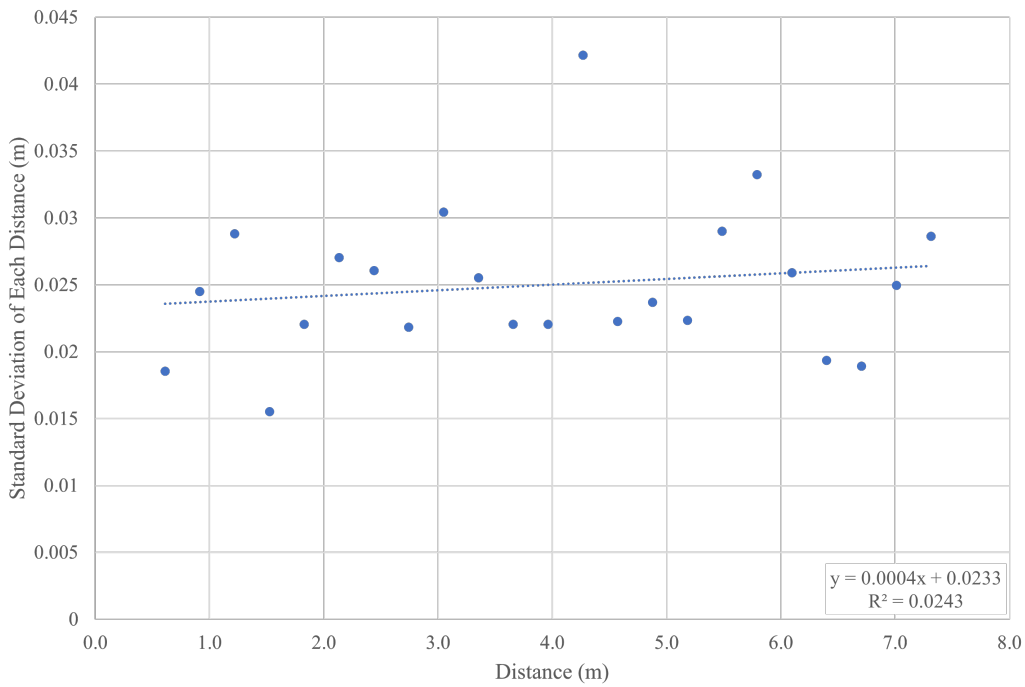

Figure 5.9: A plot of the standard deviation of the error value for each distance compared to the respective length. The trendline is shown on the graph, represented by the dashed line.

Because the slope value of the line is small — $4\,mm$ of growth in the standard deviation per $1\,m$ of length — the value of the covariance is set as the intercept value determined in the trial, as shown in the following equation:

$$Var_{UWB} = (0.0233)^2\,, \tag{5.24}$$

which is considered an appropriate estimation for this project.

The sensor noise covariance matrix for a UWB range sensor can be described using a diagonal matrix related to the number of measurement values. For one sensor, the matrix is 1×1 and the sensor noise covariance value for the UWB correction method can be described as follows:

$$R_k = \begin{bmatrix} Var_{UWB} \end{bmatrix} . \tag{5.25}$$

Another change caused by the introduction of UWB range sensors is the measurement model of the system, previously described as a 3×3 identity matrix due to the direct relationship between the prediction and the measurements. When utilizing UWB as a measurement sensor, the model of the system becomes non-linear. In the update of filter, it should now be described with the Jacobian of the modeling equation. This is referred to as an Extended Kalman filter, which applies the same prediction and update techniques to non-linear systems, as detailed in Section 3.3.6. For the range sensor, the modeling equation is the Euclidean distance between two points: the stationary sensor location (assumed to be known) and the variable prediction location. This can be written as:

$$\hat{z_k} = \sqrt{(x_{Pred} - x_{Sensor})^2 + (y_{Pred} - y_{Sensor})^2} . \tag{5.26}$$

The Jacobian is the first-order partial derivatives of the equation with respect to the inputs; in this case the predicted location. This can be written as:

$$C = \frac{\partial f(x)}{\partial x} = \begin{bmatrix} \frac{\partial \hat{z_k}}{\partial x_{Pred}} & \frac{\partial \hat{z_k}}{\partial y_{Pred}} \end{bmatrix} , \tag{5.27}$$

where

$$\frac{\partial \hat{z_k}}{\partial x_{Pred}} = \frac{x_{Pred} - x_{Sens}}{\sqrt{(x_{Pred} - x_{Sensor})^2 + (y_{Pred} - y_{Sensor})^2}} , \tag{5.28}$$

and

$$\frac{\partial \hat{z_k}}{\partial y_{Pred}} = \frac{y_{Pred} - y_{Sens}}{\sqrt{(x_{Pred} - x_{Sensor})^2 + (y_{Pred} - y_{Sensor})^2}} . \tag{5.29}$$

Therefore,

$$C = \begin{bmatrix} \frac{x_{Pred} - x_{Sens}}{\hat{z_k}} & \frac{y_{Pred} - y_{Sens}}{\hat{z_k}} \end{bmatrix} . \tag{5.30}$$

Also during the trials performed to calculate the covariance, the distance measurement reported by the UWB anchor-tag sensors was consistently different than the actual, known length between the pair. An evaluation was performed on the previously described data to calculate a prediction for the actual length as a function of the measured length. The trendline can be described with the following equation:

$$length_{actual} = 0.9837 \times (distance) + 0.0538 \,, \tag{5.31}$$

which is also shown in Figure 5.10.

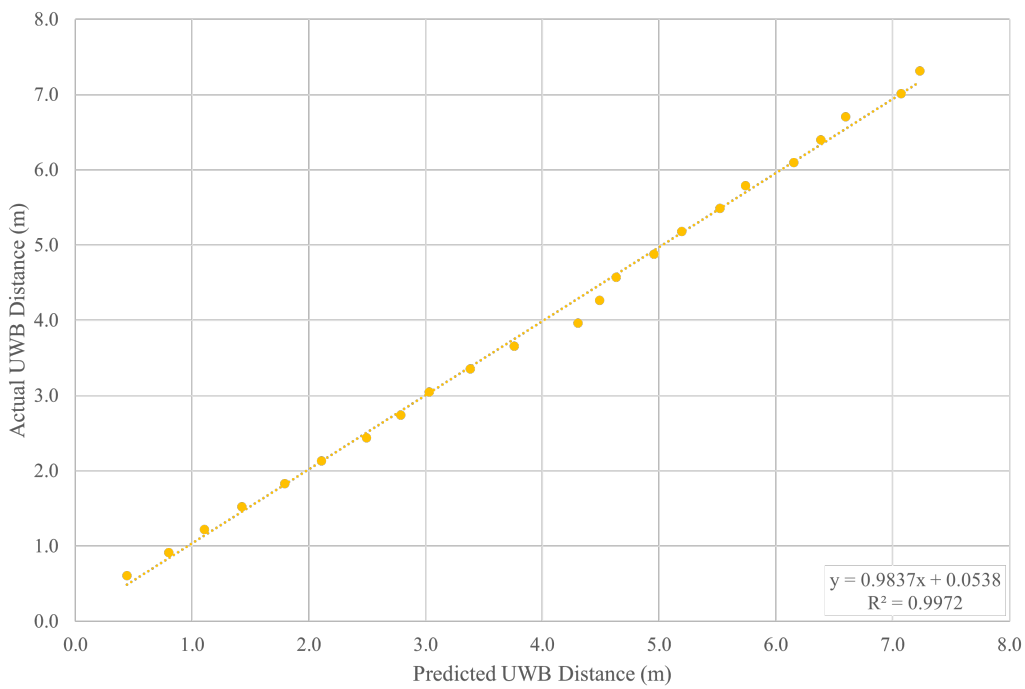

Figure 5.10: A plot of the actual length between UWB sensors compared to the measurement value at the respective length. The trendline is shown on the graph, represented by the dashed line.

As with the statistical evaluation preformed for the prediction of the pre-trained neural

network [2], each measurement value from the UWB sensors is calibrated to allow for a more accurate correction of the length.

5.6 Correction Process Solutions

The following sections detail the update results achieved using the processes described in Chapter 4 and the values previously determined in this chapter. It begins with an evaluation of the landmark-based correction, moves to the GNSS localization method, and then shows the UWB update procedure. Lastly, a comparative example between the indoor methods is shown as well as an analysis using multiple corrections in a single trajectory.

5.7 Landmark Based Correction

The first method for trajectory correction uses position based updates with known landmark locations. In this case, the landmark locations are considered directly on the path. The "landmark" is truly a mark on the ground designating a specific position; when the person carrying the device steps on the mark, a button is pressed so that the specific input time can be identified. The output that corresponds with the landmark input is then corrected with the known marker location, using the previously described Kalman Filter techniques. An example actual path, predicted trajectory without correction, and a predicted trajectory with correction are shown in Figure 5.11.

5.8 GNSS Based Correction

Due to the degradation of GNSS signals indoors, this methodology can only be applied to trajectories that are outside. For the current project, the paths were located in an arbitrary outdoor location where GNSS signals were always available. However, the same system can be applied to GNSS-limited situations where the correction will only occur if the satellite signals can be accessed and are deemed safe. In this method, the initial GNSS latitude and longitude positions are given as the initial state vector. The path is then calculated

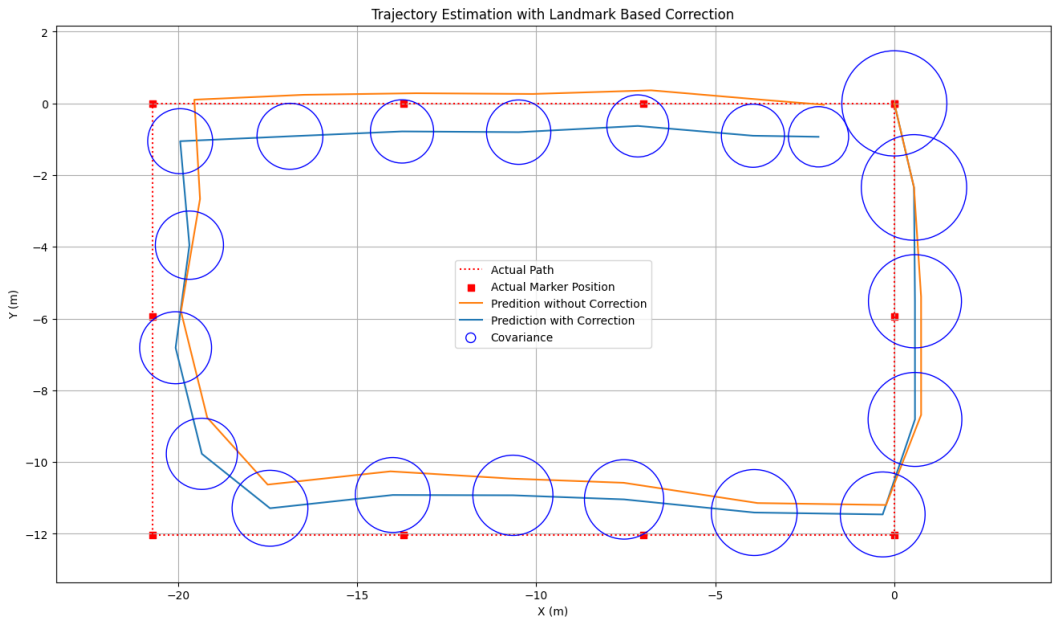

Figure 5.11: An example trajectory for a given set of raw data. The actual path of the hallway is represented by the dotted red line, with the actual marker positions shown as red squares. The predicted path without the Kalman filter correction technique is drawn in orange. The predicted path with the Kalman filter correction technique is drawn in blue with the covariance at each correction shown.

as degrees (°) instead of meters (m) and the same processes is completed. The marker positions are arbitrary locations where the update process has been selected to occur. An example GNSS path, predicted trajectory without correction, and a predicted trajectory with correction are shown in Figure 5.12.

5.9 UWB Based Correction

The final measurement discussed in this thesis for Kalman filter based correction involves the use of UWB radios to determine range. The distance between a known sensor location and the predicted position of the person is compared with the length value determined by the UWB tag/anchor pair. These experiments were again performed indoors in the same location as the landmark trials. However, as opposed to the initial trial, these updates were performed at all intervals where the UWB readings were accessible. For the first set of

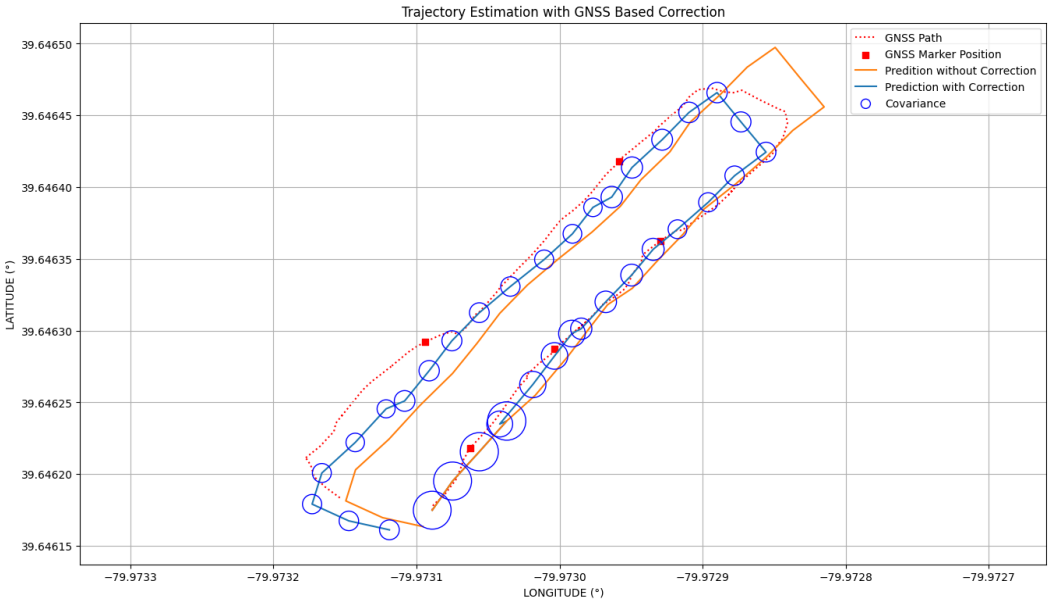

Figure 5.12: An example trajectory for a given set of raw data. The GNSS path is represented by the dotted red line, with the marker positions shown as red squares. The predicted path without the Kalman filter correction technique is drawn in orange. The predicted path with the Kalman filter correction technique is drawn in blue with the covariance at each correction shown.

estimations, this generally included the first and second hallways. An example actual path, predicted trajectory without correction, and predicted trajectory with correction for this sensor are shown in Figure 5.13. For the second set of estimations, this generally included the third and fourth hallways. An example actual path, predicted trajectory without correction, and predicted trajectory with correction for this sensor are shown in Figure 5.14.

During this trial, both the sensor located at the bottom right corner and the sensor located at the top left corner were active. Therefore, ranging data was collected over the whole trial (when accessible) for both sensors. Because these sensor measurements were taken independently from each other, but along the same path and during the same trial, the sensor measurement correction process for each can be used in series to update the path. An example path, predicted trajectory without correction, and predicted trajectory with correction using two sensors are shown in Figure 5.15. This trajectory can be considered a combination of the previous two trajectories, which were shown in Figures 5.13 and 5.14,

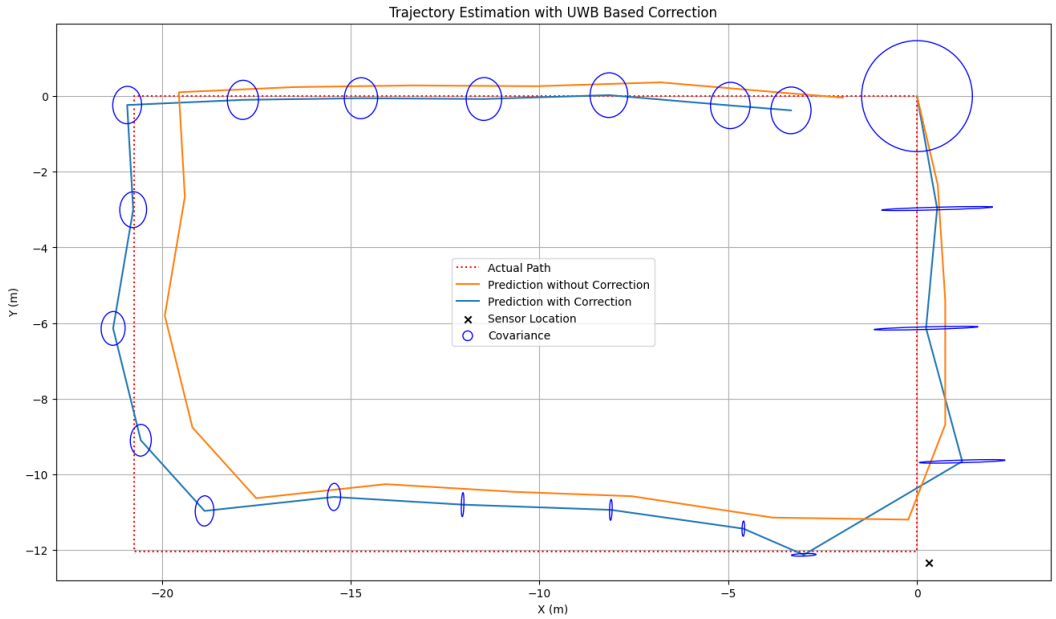

Figure 5.13: An example trajectory for a given set of raw data. The UWB measurements occur in the right and bottom hallways.

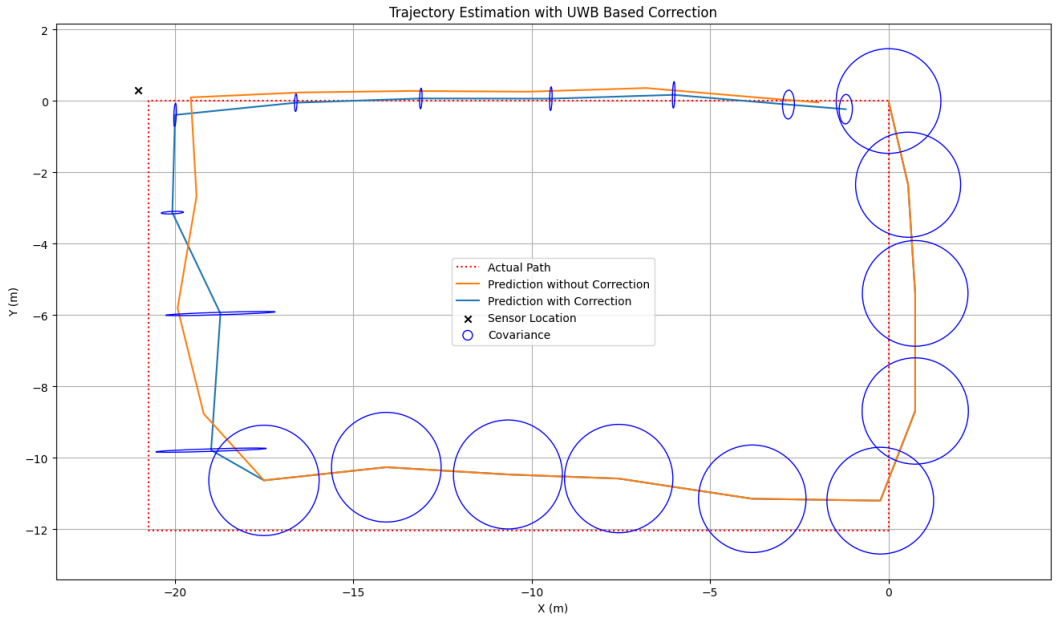

Figure 5.14: An example trajectory for a given set of raw data. The UWB measurements occur in the left and top hallways.

and is a considerably better representation of the actual path.

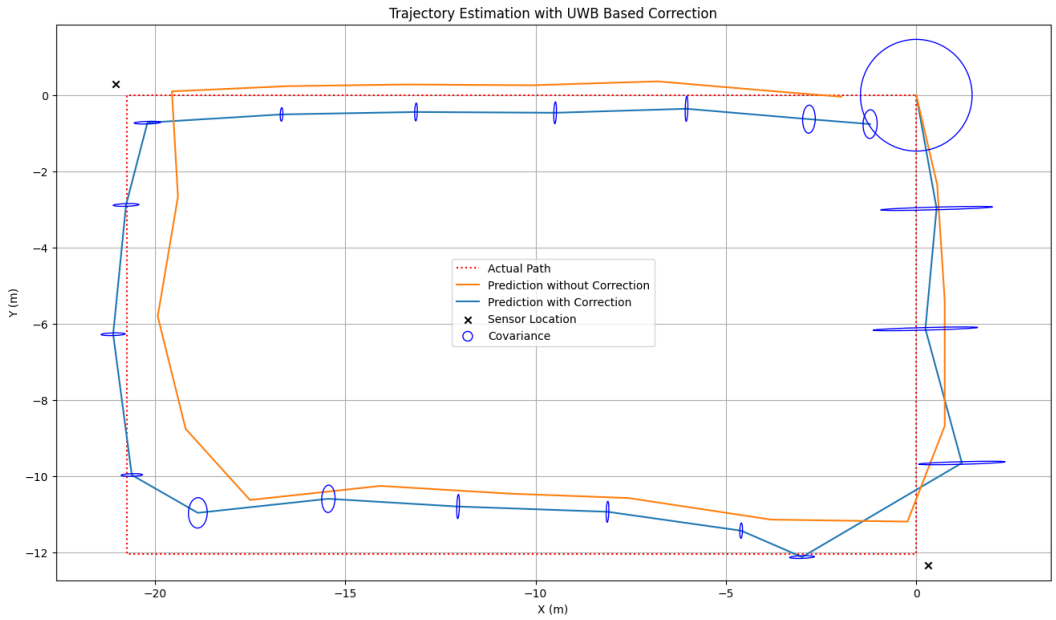

Figure 5.15: An example trajectory for a given set of raw data. The actual path of the hallway is represented by the dotted red line. The predicted path without the Kalman filter correction technique is drawn in orange. The predicted path with the Kalman filter correction technique is drawn in blue with the covariance at each correction shown. In this instance, both the sensor located at the bottom right and the sensor located at the top left were used to update the path.

5.10 Comparison between Indoor Methods

Because the same data can be used for both landmark position and UWB range estimation, a direct comparison can be made between the two methods. Intuitively, it could be anticipated that the the UWB based correction method would perform better than the landmark based correction, due to the increased number of updates along the path. This holds true, especially at locations between the landmark positions. Due to the continuous updates, the UWB path also has less abrupt changes and appears smoother. However, for the beginning of the trajectories along the first hall, all methods achieved a similar path. This is illustrated with an example trajectory in Figure 5.16.

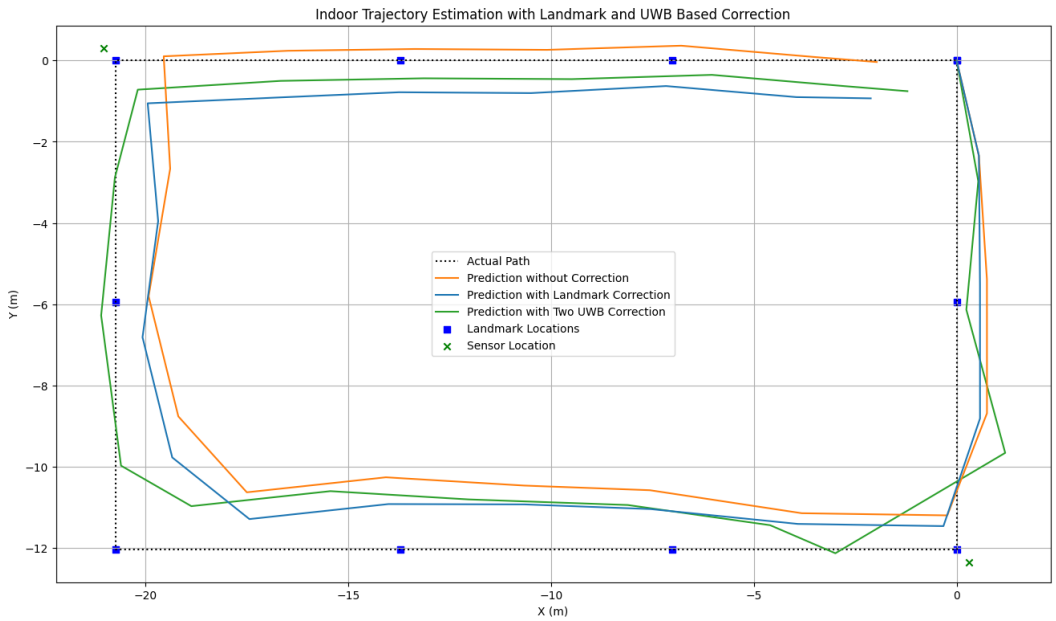

Figure 5.16: An example trajectory for a given set of raw data. The actual path of the hallway is represented by the dotted black line. The predicted path without the Kalman filter correction technique is drawn in orange. The predicted path with the landmark correction technique is drawn in blue. The predicted path with the UWB correction method is drawn in green.

5.11 Correction with Multiple Methods

In the same manner as the previous section, because the same path can be corrected with various methods, a solution was created which combined the correction techniques for both landmark position and UWB range update terms. This is a cumulative procedure that iteratively checks whether the sensor values are available for each method instead of only checking for one. This is shown in Figure 5.17 with the same data previously discussed.

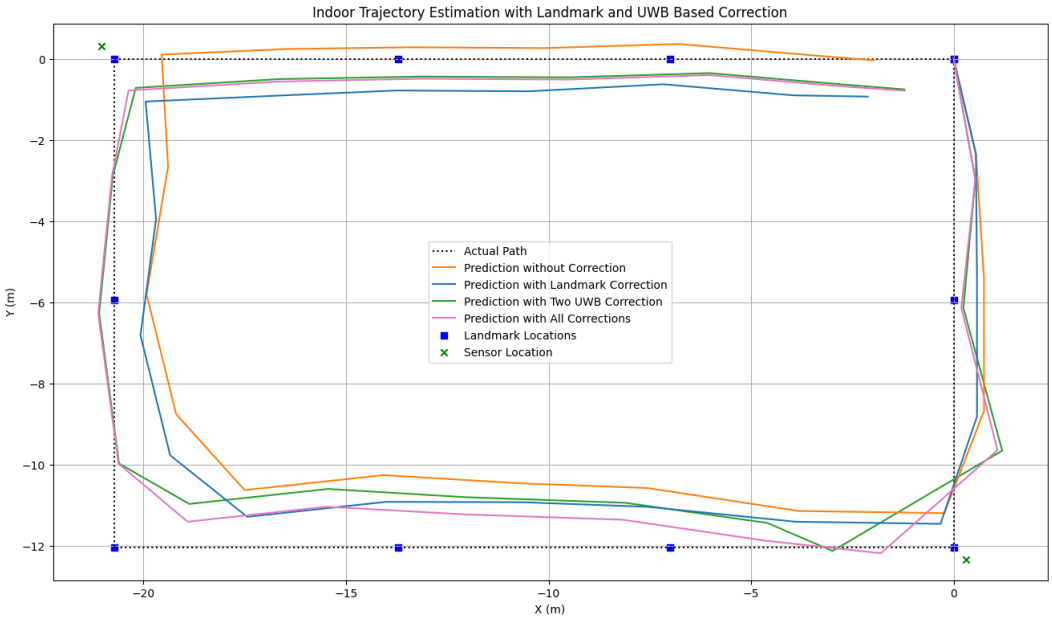

Figure 5.17: An example trajectory for a given set of raw data. The actual path of the hallway is represented by the dotted black line. The predicted path without the Kalman filter correction technique is drawn in orange. The predicted path with the landmark correction technique is drawn in blue. The predicted path with the UWB correction method is drawn in green. The predicted path using both landmark and UWB information is drawn in pink.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The most common, well-known method of localization is considered to be GNSS positioning. However, there are many situations where satellite signals are inaccessible or limited. This can include situations where positioning is required indoors, under dense tree cover, or underground. This research aimed to create a system that estimates the trajectory of a moving person carrying a mobile device from a known initial location to an unknown final position under these circumstances. The prediction was to be determined in terms of the position and the orientation of the person. Based on an empirical and quantitative analysis of a pre-trained neural network using IMU data in combination with outside sources — such as landmark locations, GNSS signals, and/or UWB range information — it can be concluded that the proposed solution is able to provide an estimate that is consistent with the true trajectory. The results of the correction analysis also indicate that the Kalman filter-based update techniques effectively improve the intermediate prediction made by the neural network, to create a better representation of the motion.

To achieve this complete solution, certain system properties were evaluated. Initially, the covariance caused by the changes in the position — the process noise covariance — was calculated as a function of the distance offset with an intercept value. Then, each of

the correction methods was analyzed to determine their respective sensor noise covariance values.

Each proposed solution was analyzed and compared to the ground truth for the collected set of IMU data. Each set of prediction results was shown to be within the expected uncertainty of the actual trajectory. This confirms that the proposed method was able to create a robust, accurate system that creates consistent position and orientation predictions.

6.2 Future Work

For future work within the previously completed analysis, it would be interesting to test the applicability of Allan variance to the current evaluation, to estimate the stability of the process noise — something that was relied upon heavily throughout this process. To expand on these results, future work should be performed to create a system for UWB ranging correction with multiple sensors. This can be achieved by creating different correction terms for each individual sensor with a different known location. The next step would be a system where the known location of a single UWB anchor is variable with little uncertainty — such as moving with a car localized with GNSS positioning. The last suggestion for this project would be a variable location UWB sensor where the "known" position is an estimate based on a similar method as proposed in this thesis. If two people have position estimates using inertial navigation, and each person has the capability to both send and receive ranging information, cooperative localization can be performed in the same manner to improve each individual estimate.

Bibliography

[1] Oliver J Woodman. An introduction to inertial navigation. Technical report, University of Cambridge, Computer Laboratory, 2007.

[2] Sachini Herath, Hang Yan, and Yasutaka Furukawa. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3146–3152. IEEE, 2020.

[3] Directions and maps: Statler college information technology systems. `https://its.statler.wvu.edu/info-faqs/directions-and-maps`, 2023. accessed 2023-11-09.

[4] GPS: The global positioning system. `https://www.gps.gov/`, 2023. accessed 2023-10-09.

[5] The global positioning system: Global positioning tutorial. `https://oceanservice.noaa.gov/education/tutorial_geodesy/geo09_gps.html`, 2023. accessed 2023-10-09.

[6] Paul Tullis. GPS down. *Scientific American*, 321(6):39–45, 2019.

[7] Lauro Ojeda and Johann Borenstein. Non-GPS navigation for security personnel and first responders. *Journal of Navigation*, 60(3):391–407, 2007.

[8] Bor-Shing Lin, I-Jung Lee, Shun-Pu Wang, Jean-Lon Chen, and Bor-Shyh Lin. Residual neural network and long short-term memory–based algorithm for estimating the motion trajectory of inertial measurement units. *IEEE Sensors Journal*, 22(7):6910–6919, 2022.

[9] C-Y Lin, Y-E Lu, C-H Huang, and K-W Chiang. A cnn-speed-based gnss/pdr integrated system for smartwatch. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48:235–241, 2023.

[10] Xinyu Hou and Jeroen HM Bergmann. HINNet: inertial navigation with head-mounted sensors using a neural network. *Engineering Applications of Artificial Intelligence*, 123:106066, 2023.

[11] Swapnil Sayan Saha, Sandeep Singh Sandha, Luis Antonio Garcia, and Mani Srivastava. Tinyodom: Hardware-aware efficient neural inertial navigation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(2):1–32, 2022.

[12] Wei Li, Ruizhi Chen, Yuan Wu, and Haitao Zhou. Indoor positioning system using a single-chip millimeter wave radar. *IEEE Sensors Journal*, 23(5):5232–5242, 2023.

[13] Franck Malivert, Ouiddad Labbani-Igbida, and Hervé Boeglen. Comparison and improvement of 3D-multilateration for solving simultaneous localization of drones and UWB anchors. *Applied Sciences*, 13(2), 2023.

[14] Xianjia Yu, Paola Torrico Morrn, Sahar Salimpour, Jorge Pena Queralta, and Tomi Westerlund. Loosely coupled odometry, uwb ranging, and cooperative spatial detection for relative monte-carlo multi-robot localization. *arXiv preprint arXiv:2304.06264*, 2023.

[15] Lalindra Jayatilleke and Nian Zhang. Landmark-based localization for unmanned aerial vehicles. In *2013 IEEE International Systems Conference (SysCon)*, pages 448–451. IEEE, 2013.

[16] Abdellah Chehri, Paul Fortier, and Pierre Martin Tardif. Uwb-based sensor networks for localization in mining environments. *Ad Hoc Networks*, 7(5):987–1000, 2009.

[17] Sachini Herath, David Caruso, Chen Liu, Yufan Chen, and Yasutaka Furukawa. Neural inertial localization. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6594–6603. IEEE, 2022.

[18] Yingying Wang, Hu Cheng, and Max Q.-H. Meng. Spatiotemporal co-attention hybrid neural network for pedestrian localization based on 6D IMU. *IEEE Transactions on Automation Science and Engineering*, 20(1):636–648, 2023.

[19] Yue Yu, Yi Zhang, Liang Chen, and Ruizhi Chen. Intelligent fusion structure for Wi-Fi/BLE/QR/MEMS sensor-based indoor localization. *Remote Sensing*, 15(5), 2023.

[20] Yilin Zhu, Yang Kong, Yingrui Jie, Shiyou Xu, and Hui Cheng. GRACO: a multimodal dataset for ground and aerial cooperative localization and mapping. *IEEE Robotics and Automation Letters*, 8(2):966–973, 2023.

[21] Yudi Dai, Yitai Lin, Xiping Lin, Chenglu Wen, Lan Xu, Hongwei Yi, Siqi Shen, Yuexin Ma, and Cheng Wang. SLOPER4D: A scene-aware dataset for global 4D human pose estimation in urban environments. *arXiv preprint arXiv:2303.09095*, 2023.

[22] Lucas Carvalho de Lima, Milad Ramezani, Paulo Borges, and Michael Brünig. Air-ground collaborative localisation in forests using lidar canopy maps. *IEEE Robotics and Automation Letters*, 8(3):1818–1825, 2023.

[23] Yayun Du, Swapnil Sayan Saha, Sandeep Singh Sandha, Arthur Lovekin, Jason Wu, S Siddharth, Mahesh Chowdhary, Mohammad Khalid Jawed, and Mani Srivastava. Neural-kalman gnss/ins navigation for precision agriculture. In *International Conference on Robotics and Automation (ICRA)*, 2023.

[24] Lukas Luft, Tobias Schubert, Stergios I Roumeliotis, and Wolfram Burgard. Recursive decentralized collaborative localization for sparsely communicating robots. In *Robotics: science and systems*, 2016.

[25] Lukas Luft, Tobias Schubert, Stergios I Roumeliotis, and Wolfram Burgard. Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication. *The International Journal of Robotics Research*, 37(10):1152–1167, 2018.

[26] Glenn MacGougan, Kyle O'Keefe, and Richard Klukas. Ultra-wideband ranging precision and accuracy. *Measurement science and technology*, 20(9):095105, 2009.

[27] Mohammed Ayman Shalaby, Charles Champagne Cossette, James Richard Forbes, and Jerome Le Ny. Calibration and uncertainty characterization for ultra-wideband two-way-ranging measurements. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4128–4134, 2023.

[28] M Bochkati and T Pany. Does the android operating system provide what the mems-imu manufacturers promise? In *2021 DGON Inertial Sensors and Systems (ISS)*, pages 1–17, 2021.

[29] Android open source project. `https://source.android.com/docs/core/interaction/sensors/sensor-types`, 2022. accessed 2023-08-28.

[30] Günther Retscher. Indoor navigation—user requirements, state-of-the-art and developments for smartphone localization. *Geomatics*, 3(1):1–46, 2023.

[31] Naser El-Sheimy and Ahmed Youssef. Inertial sensors technologies for navigation applications: State of the art and future trends. *Satellite Navigation*, 1(1):1–21, 2020.

[32] Wonho Kang, Seongho Nam, Youngnam Han, and Sookjin Lee. Improved heading estimation for smartphone-based indoor positioning systems. In *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications-(PIMRC)*, pages 2449–2453. IEEE, 2012.

[33] Guenther Retscher and Allison Kealy. Navigation based on sensors in smartphones. In *Positioning and Navigation in Complex Environments*, pages 368–396. IGI Global, 2018.

[34] Wilfried Elmenreich. An introduction to sensor fusion. *Vienna University of Technology, Austria*, 502:1–28, 2002.

[35] Jurek Z Sasiadek. Sensor fusion. *Annual Reviews in Control*, 26(2):203–228, 2002.

[36] Maria Isabel Ribeiro. Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics*, 43(46):3736–3741, 2004.

[37] Milad Nazarahari and Hossein Rouhani. 40 years of sensor fusion for orientation tracking via magnetic and inertial measurement units: Methods, lessons learned, and future challenges. *Information Fusion*, 68:67–84, 2021.

[38] K Xiong, CL Wei, and LD Liu. Robust unscented kalman filtering for nonlinear uncertain systems. *Asian Journal of Control*, 12(3):426–433, 2010.

[39] Kevin Gurney. *An introduction to neural networks*. CRC press, 1997.

[40] Vivienne Sze, Yu-Hsin Chen, Tien-Ju Yang, and Joel S Emer. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329, 2017.

[41] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, 2018.

[42] Changhao Chen. Deep learning for inertial positioning: A survey. *arXiv preprint arXiv:2303.03757*, 2023.

[43] Hang Yan, Qi Shan, and Yasutaka Furukawa. Ridi: Robust imu double integration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 621–636, 2018.

[44] Changhao Chen, Xiaoxuan Lu, Andrew Markham, and Niki Trigoni. Ionet: Learning to cure the curse of drift in inertial odometry. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[45] Changhao Chen, Peijun Zhao, Chris Xiaoxuan Lu, Wei Wang, Andrew Markham, and Niki Trigoni. Oxiod: The dataset for deep inertial odometry. *arXiv preprint arXiv:1809.07491*, 2018.

[46] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012.

[47] Ralf C Staudemeyer and Eric Rothstein Morris. Understanding LSTM–a tutorial into long short-term memory recurrent neural networks. *arXiv preprint arXiv:1909.09586*, 2019.

[48] Evandro Bernardes and Stéphane Viollet. Quaternion to euler angles conversion: A direct, general and computationally efficient method. *Plos one*, 17(11):e0276302, 2022.

[49] Gregory G Slabaugh. Computing euler angles from a rotation matrix. *Retrieved on August*, 6(2000):39–63, 1999.

[50] Rodrigo Leandro, Marcelo Santos, and Karen Cove. An empirical approach for the estimation of gps covariance matrix of observations. In *Proceedings of the 61st Annual Meeting of The Institute of Navigation (2005)*, pages 1098–1104, 2005.