INDOOR POSITIONING USING WLAN SIGNAL FINGERPRINT MATCHING AND PATH EVALUATION WITH RETROACTIVE ADJUSTMENT ON MOBILE DEVICES

EROS GULO

A DISSERTATION SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN EARTH AND SPACE SCIENCE
LASSONDE SCHOOL OF ENGINEERING
YORK UNIVERSITY
TORONTO, ONTARIO

MAY 2020

Abstract

Location-aware services have seen great demand recently and the availability of Global Navigation Satellite System (GNSS) has greatly contributed to that. However, GNSS is only reliably available in outdoor environments, leaving a large gap for a solution in indoor environments where people spend most of their time. With the increasing number and usage of mobile devices in people's daily life, indoor positioning, using sensors and other information available on mobile devices, has attracted a lot attention from both academia and industry for the purpose of providing location-aware services. For the last decade, many research groups have reported successful indoor positioning using various sensors, primarily based on Wireless Local Area Network (WLAN) signal fingerprint matching (WSFM). However, the current success is limited to controlled environments and its effective adaptation at large-scale, supporting seamless, precise and long-term navigation in diverse indoor environments, still remains a challenging research problem. In this thesis, I present a novel indoor positioning system, using WSFM and a Path Evaluation and Retroactive Adjustment (PERA) module. The PERA module aims to improve the positioning accuracy by fusing the results obtained by WSFM with a multi-scale movement regularity evaluation. In this framework, my research focuses on how to implement and integrate smoothness regularity of human movement with conventional signal matching-based positioning, which often shows difficulty to regularize movement trajectories. The PERA module intends to improve the positioning accuracy of the system in a similar fashion to Pedestrian Dead

Reckoning (PDR) implemented along with WSFM. PDR algorithms implemented on mobile devices require high-frequency sampling of inertial sensors and coordinate transformations of each sample. The benefit of my approach is that it avoids the requirement of inertial sensor data, and its respective intensive computation and power use, while providing a similar accuracy improvement to PDR. In this thesis, I implemented an end-to-end positioning system, which include fingerprint data collection, as well as a real-time positioning system built based on WSFM and including the PERA module. Experimental results show that the WSFM algorithm yields room level positioning accuracy (less than 3-5 metres of error) 90% of the time across various environments. Furthermore, upon employing the PERA module, positioning error is reduced to less than 2-3 metres 95% of the time across all testing settings. In literature, state-of-the-art Sensor Fusion algorithms combining WSFM and PDR boast mean positioning errors of 1-3 metres, which is arguably sufficient for providing location-based services on mobile devices. With mean positioning errors of 0.8-1.4 metres and the aforementioned upper bounds of positioning error, the experimental results presented herein demonstrate that this more lightweight algorithm requiring less resources is a viable novel approach for positioning using mobile devices in an indoor environment.

Acknowledgements

I would, first and foremost, like to thank my supervisors, Dr. Regina Lee and Dr. Gunho Sohn for giving me this opportunity to work on this research. In particular, Dr. Sohn's guidance has been invaluable to my development as a professional. I would also like to acknowledge his trust in me, which allowed me to develop a sense of confidence and independence in my research, work and life. In the years under his supervision I have known him as a very generous and caring person. I would also like to thank all of my colleagues at the GeoICT lab, Afnan Ahmad, Kivanc Babacan, Solomon Chan, Leihan Chen, Abdel-Hadi Hor, Mojgan Jadidi, Ali Baligh Jahromi, Jaewook Jung, Yoonseok Jwa, Heungsik Kim, Connie Ko, Chao Luo, Kunwoo Park, Phillip Robbins, Mozhdeh Shahbazi, Andreas Wichmann and Junjie Zhang, for their support, advice and friendship during my time on campus. Above all, I would like to thank my family for their immense support during my studies, allowing me to truly focus on my work. Without their support, patience, advice and trust, none of this would have been possible. I owe a great debt to them and will forever strive to make them proud. Finally, I would like to thank York University and all of my sponsors for financially supporting me throughout my studies.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AP	WLAN Access Point
BRG	Bergeron Centre for Engineering Excellence
BSSID	Basic Service Set Identifier (MAC Address)
CB	Chemistry Building
CPU	Central Processing Unit
GNSS	Global Navigation Satellite System
k-NN	k – Nearest Neighbours
LMR	Long Term Movement Regularity
LTE	Long Term Evolution (Mobile Communication Standard)
MAC	Media Access Control (Address)
NFC	Near Field Communication
ML	Bayes Maximum Likelihood
PDR	Pedestrian Dead Reckoning
PERA	Path Evaluation and Retroactive Adjustment
PSE	Petrie Science and Engineering Building
RFID	Radio Frequency Identification
RSS(I)	Received Signal Strength (Indicator)
SMR	Short Term Movement Regularity
SSID	Service Set Identifier
WLAN	Wireless Local Area Network
WSFM	WLAN Signal Fingerprint Matching
WSDFM	WLAN Signal Differences Fingerprint Matching

1. Introduction

In this section, the motivation for and the objectives of this research are discussed, covering the general framework of the work, as well as the specific novel contributions proposed. Finally, an outline of the thesis is provided, as well as a list of academic achievements resulting from this work.

1.1 Motivation

Indoor positioning and navigation is a burgeoning field in the greater umbrella of location based services that has hit its stride in recent years, along with, and primarily because of, the ubiquity of smartphones. The major area of interest in the last few years has been the application of different indoor positioning methods on the smartphone (and tablet) platform. In addition, GNSS positioning and navigation has been very successful and dominant in the context of outdoor positioning on mobile devices, and in turn, has resulted in users demanding the same level of service in the indoor space.

Due to the inherently smaller indoor spaces in relation to outdoor spaces, acceptable accuracy indoors is 1-5 m, as opposed to the 5-20 m acceptable accuracy outdoors (*Senion IPS*, 2016). In addition, there is no set-up cost for GNSS, just a receiver and positioning software built into the mobile device. This results in very high expectations set for indoor positioning solutions; an ideal solution should be sufficiently accurate, have low or no setup and maintenance costs, and should be a universal solution that can be used by most people (i.e., anyone with a smartphone) in most places. This

research looks to build on previous work and approaches to advance towards an ideal solution.

1.2 Research Problem

The current state-of-the-art indoor positioning algorithms are generally approaches that combine a wireless signal-based algorithm, which can provide absolute position estimates, with an inertial sensor-based algorithm that can more accurately describe the user's relative motion. In literature, these algorithms generally boast positioning accuracy in the range of 0.5-3 metres mean positioning error (Guo *et al.*, 2020). The limitations of these approaches are generally associated with the use of inertial sensors as they require high-frequency sampling and heavy computations, and are not standardized or high quality. Furthermore, no reasonable assumption can be made about how the device is attached to the user, which is an assumption that can greatly affect algorithm accuracy (Villien, Frassati and Flament, 2019). Wireless signal-based algorithms, on the other hand, are generally more lightweight, in terms of computation and resources, however, they are susceptible to more frequent outlier user position estimates and the data collection for the training/offline phase can be cumbersome (He and Chan, 2016).

1.3 Research Objectives

1.3.1 General Framework

The major objective of my research is to create a standalone positioning system utilizing the Wireless Local Area Network (WLAN) signals in the indoor environment. The system should be a suitable indoor location provider of location based services on mobile devices (e.g., map or navigation application for a shopping centre or university campus). This system offers the robustness and accuracy of the state-of-the-art hybrid methods while still utilizing the same resources as more lightweight WLAN-based algorithms. To arrive at this objective, I took a deep look into WLAN signal fingerprint matching (WSFM) methods to determine the ideal algorithms and parameters for an implementation on the smartphone platform. Research has shown that WSFM algorithms have a higher performance ceiling than their WLAN signal ranging counterparts, thus, my research aims to optimize the performance of WSFM methods and minimize their deficiencies and weaknesses.

I implemented various matching algorithms, such as weighted and non-weighted k-Nearest Neighbours (k-NN) as well as Bayes Maximum Likelihood (ML), assessing individual as well as cooperative performance. Within the scope of each of these implementations, a wide range of parameters were varied and I investigated their impact on performance. Parameters considered include definitions of the fingerprint and location model, histogram kernelization for ML algorithms, and proximity definition and weighting at various stages of k-NN computation. Finally, a novel Path

Evaluation and Retroactive Adjustment (PERA) module was developed to emulate the smoothness offered by Pedestrian Dead Reckoning (PDR) in hybrid algorithms, and location estimates of previous epochs are retroactively updated, based on several consecutive readings, to ensure more accurate paths and prevent the positioning system from getting *stuck* at local maxima.

A secondary objective of my research is to improve the reference fingerprint database, one of the most cited disadvantages of WSFM methods. In general, the greater the human effort in the creation of the reference database, the greater the overall accuracy of the positioning system. While other approaches have focused on synthesizing the reference database to various degrees, many of these approaches suffer from the same problems as ranging algorithms (although to a lesser degree), as it is difficult to predict the WLAN signal path loss through complex environments such as large commercial buildings. I, instead, focus on making the data collection more efficient, still taking advantage of human effort but getting a better return from it. In general, my research focuses on making the data collection easier – therefore, requiring less *professional* effort – as well as gathering strategic data that better defines the WLAN signal as opposed to blindly increasing the quantity of surveyed data for redundancy.

A third objective is the addition of various efficiencies and computational performance improvements to the physical implementations of the positioning system. Since the various algorithms to be tested will be implemented on Android as well as computer-runnable Java applications, some of which will be able to function in real-time within

the experimental environment. It will be important to not only make sure that the implementations are fully functional but that they also run efficiently. Therefore, a final goal is to take advantage of the computing power of modern mobile devices and implement concurrency as well as efficient memory management in my applications.

1.3.2 Contributions

The main contribution that I have made in this thesis is to present a novel positioning method taking advantage of the smoothness and regularity of human motion. To my knowledge, the PERA module is the first kind of indoor positioning algorithm, which integrates smoothness regularity of human movement with WSFM. PERA addresses the limitation of locality to predict indoor position without considering smooth transitions of current position in relation to previous trajectory. To achieve this goal, I have made three contributions to address the limitations of state-of-the-art indoor positioning methods and implemented an end-to-end system for facilitating fingerprint data collection and real-time positioning.

My first contribution is a study and experimental testing of some of the important parameters and sub-algorithms of WSFM. A variety of parameters were discussed in (Farshad *et al.*, 2013), and sub-algorithms such as k-NN and ML were discussed in (Honkavirta *et al.*, 2009; Dawes and Chin, 2011). Other works such as (Beder and Klepal, 2012; Laoudias, Zeinalipour-Yazti and Panayiotou, 2013) suggested particular improvements to improve WSFM that are experimentally tested to some degree in my

research. My research implements, tests and expands upon some of the findings discussed in the aforementioned works; the experiments also examine some parameters that have not yet been studied on the smartphone platform. In the end, my contribution can be summarized as the development of a novel WSFM algorithm inspired by previous research, employing some novel ideas and adapted to empirical findings; implementation and live testing was a constructive and iterative process until performance was deemed to be satisfactory through qualitative real-time testing. Experimental testing of the final iteration of the WSFM algorithm yielded room level positioning accuracy (3 – 5 metres) 90% of the time across various environments.

My second contribution is the design, implementation and experimental testing of a path evaluation technique that adds another input to the real-time positioning algorithm, as well as retroactively repositions the user's path over the previous several epochs. The purpose and goal of this path evaluation module is to smooth and improve the accuracy of WSFM results, ideally resulting in comparable performance to state-of-the-art hybrid WSFM and PDR algorithms. Most recent smartphone-based indoor localization systems (Hafner *et al.*, 2013; Ebner *et al.*, 2014; Herrera *et al.*, 2014; Subbu *et al.*, 2014; Chen *et al.*, 2015; Tian *et al.*, 2015; Zhang, Chen and Xue, 2018; Dumbgen *et al.*, 2019; Renaudin *et al.*, 2019; Villien, Frassati and Flament, 2019; Guo *et al.*, 2020) use multi-sensor approaches to make up for the shortcomings of basic WLAN signal-based methods. On the other hand, my approach first improves the accuracy of the WSFM algorithm (by implementing optimizations based on the results of the initial experimentation), then addresses the remaining fundamental shortcomings through the

PERA module. This results in comparable performance to the hybrid approaches (positioning error is reduced to less than 2 – 3 metres 95% of the time across all testing settings), while being more energy and computationally efficient.

My third contribution is in the implementation of the entire system on the smartphone platform, where I employ concurrency and memory management to improve the efficiency and responsiveness of mobile implementations of WLAN fingerprint matching systems. As discussed by (Subbu *et al.*, 2014) in his review of smartphone-based indoor localization systems, most of the solutions are constantly communicating with a central server that does the actual computations for the localization. The problems with this approach are firstly, that the constant communication requires active network connectivity and the communication itself has a significant energy cost; and secondly, relying on a server to perform the positioning will inevitably result in a delay between the sensing of the data and the localization result, making for a less responsive system. My work shows how taking advantage of the computing potential of modern mobile devices allows for more efficient implementations for all WLAN fingerprint matching approaches as well as hybrid multi-sensor approaches, resulting in the potential for more responsive client-side implementations.

1.4 Thesis Outline

Chapter 1 introduces the reader to the indoor positioning problem addressed herein, states the objective of the research and provides the motivation for pursuing it. Chapter

2 explores the subject of positioning in indoor environments using mobile devices; some context is provided regarding the devices used, location models employed, various existing location estimation methods, as well as, current state-of-the-art solutions in the field. Chapter 3 describes the physical study area of the research, including the buildings and their various relevant characteristics; it outlines the process, methods and tools of data collection, and it describes how the data are processed and arranged for use in the implemented positioning systems. Chapter 4 provides an overview of the entire final iteration of the positioning system implemented and experimented with as part of this research; it proceeds to explain the details of its various modules and components, describing the methods considered and chosen for each of the modules of the final positioning system presented. Chapter 5 focuses on the implementation of the positioning system explored and discussed in this research, as well as, the implementation of tools used for data collection, and finally, various performance improvements built into the implementation to ameliorate real-time use and analysis. Chapter 6 reviews the results of all of the experimentation performed as part of this research; it describes the methods used to assess the positioning performance of the system in a quantifiable manner and details positioning performance improvements offered by the novel portions of the positioning system. Finally, Chapter 7 provides some closure on the research, reviewing and explaining the findings, and exploring avenues of future work and research on this subject.

1.5 Achievements

The research presented herein was published in the ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences (Gulo, Sohn and Ahmad, 2018). A portion of this research was also awarded third place in the AOLS Annual Geomatics Graduate Student Poster Competition and was published in the Ontario Professional Surveyor magazine (*Ontario Professional Surveyor, Volume 58, No. 2*, 2015).

2. Background

This chapter explores the subject of positioning in indoor environments using mobile devices; some context is provided regarding the devices used, location models employed, various existing location estimation methods, as well as, current state-of-the-art solutions in the field.

2.1 Mobile Devices

2.1.1 Sensors

Due to the multitude of sensors now offered by mobile devices, there is a vast amount of data that becomes available to exploit for the purpose of positioning (Lane *et al.*, 2010). The following is a list of the sensors that may be available in a standard smartphone:

- WLAN Radio
- Bluetooth Radio
- Cellular Radio
- Accelerometer
- Gyroscope
- Magnetometer
- NFC (RFID)
- Barometer
- Camera
- Microphone
- Ambient Light Sensor
- Proximity Sensor

The three radios provide three different sources of electromagnetic signals that can be used both for obtaining raw measurements for positioning purposes and for communication of positioning results. The accelerometer, gyroscope and magnetometer can be used to measure physical movement of the mobile device. NFC

is useful for close proximity wireless data transfer, for example to provide location specific information with a simple small NFC tag at the location. The camera provides a wealth of visual information that can be used both for positioning as well as extracting location specific information. The remaining sensors can be classified as environmental sensors and be used individually or in tandem to extract various information about the mobile device's environment.

2.1.2 Computing Capabilities

Today's smartphones and tablets can perform nearly the same computations as low-end laptops. Furthermore, since 2012, mobile devices have had quad-core processors and more recently even octa-core devices are common. Along with the increase in performance that is possible by an increasing number of cores, energy efficiency has also been a concern of manufacturers. Due to the trend of progressively larger screens and thinner bodies on smartphones along with faster (and more power-hungry) data transmission technologies such as LTE radios, energy on mobile devices always seems to be in short supply. Thus, CPU's have seen great power efficiency improvements, along with the performance improvements.

2.2 Location Models

In order to both estimate and present a location for the user, the positioning system must be set up within a defined location (or spatial) model, or in other words, a system that defines the space within which the user is being located.

2.2.1 Discrete Model

A discrete location model means that the navigable space within the floor or building is partitioned into discrete generally equal spaces. Each individual space is then assigned a unique identifier and its location is generally set as the location of its geometric center. When the positioning algorithm estimates a position for the subject in this location model, it chooses one or more of the discrete spaces as the most likely location(s) of the subject.

2.2.2 Continuous Model

A continuous location model means that no partitioning of the navigable space occurs, the position of a subject can be estimated at any location in the navigable space. This type of location model is generally used with trilateration-based positioning systems, or any other algorithms that are not limited to estimating only the best candidate out of a discrete set.

2.2.3 Hybrid Model

A hybrid location model is any mix of the two aforementioned models, and can have various forms. The most trivial form is when the navigable space is partitioned into discrete spaces and the positioning algorithm estimates the user's position by choosing multiple most likely discrete spaces. The discrete locations associated with the chosen

spaces are averaged and presented in a continuous location model as the user's estimated position.

2.3 Location Estimation Methods

The multitude of sensors now incorporated in mobile devices opens the door to a vast amount of existing indoor positioning methods, and even inspires a few new methods (Subbu *et al.*, 2014; Villien, Frassati and Flament, 2019; Guo *et al.*, 2020). The following is a list of the most popular approaches to indoor positioning on a smartphone:

- WLAN Fingerprint Matching
- WLAN Ranging
- Bluetooth Ranging
- Pedestrian Dead Reckoning
- Magnetic Fingerprint Matching
- Image Recognition / Visual Odometry
- Hybrid / Sensor Fusion

The WLAN and Pedestrian Dead Reckoning approaches were significantly studied prior to the advent of smartphones, whereas the others were only studied in significant detail once smartphones arrived with an increasing variety of sensors. Furthermore, on the mobile platform it has become increasingly common for new research to focus on multi-sensor hybrid approaches, usually cooperatively employing various algorithms using different sensor data.

2.3.1 WLAN Ranging and Trilateration

WLAN signal based approaches can be broadly categorized into two major categories, ranging and fingerprinting. Ranging methods use signal path loss models of varying complexity to estimate the distances to WLAN access points (APs) based on the received signal strengths of the APs. Trilateration is then used to position the user based on the computed distances to the various APs. These methods generally require detailed floor plans of the building and the locations of the APs in the building in order to model the signal propagation of the APs and determine the user's distance to each of them (Torteeka, Chundi and Dongkai, 2014). Recent WLAN APs, however, have introduced support for the IEEE 802.11-2016 standard that includes the fine time measurements (FTM) protocol. The FTM protocol enables a ranging process based on round-trip time (RTT) that has begun to be implemented in trilateration positioning approaches (Yan *et al.*, 2019). Once there is widespread public and commercial deployment of APs and mobile devices supporting the new standard, its viability for indoor positioning can be better quantified.

Bluetooth ranging works in a manner very similar to WLAN ranging. The main difference is that Bluetooth signals are used as opposed to WLAN. This method requires Bluetooth beacons that constantly emit an identifying signal. The user's device then computes the distances to the sensed beacons and trilateration is used to estimate a position for the user. Another key difference with WLAN ranging is that the number of beacons required for equivalent performance between the two methods is

significantly higher than the number of APs required because Bluetooth beacons have a much weaker signal – and therefore, range – due to the importance given to low cost and energy consumption (Martin *et al.*, 2014).

2.3.2 WLAN Fingerprint Matching

On the other hand, fingerprinting methods require AP signal strength maps of the building, or in practical terms, a database of the expected signal strengths of APs at discrete locations in the building (reference fingerprint database); the density of these discrete locations may vary. The user takes a sample of the signal strengths of APs within range (live fingerprint) and various mathematical algorithms can then be used to match the live fingerprint to the most similar location in the database with respect to the signal strengths of the sampled APs. There have also been systems proposed that avoid some or all of the manual data collection by modeling the WLAN signals throughout the positioning area, however, these systems require accurate AP locations and detailed maps of the positioning area. Generally, it is very difficult to accurately model the signal propagation of APs in complex indoor environments, thus, resulting in degraded positioning performance (Honkavirta *et al.*, 2009; Farshad *et al.*, 2013; He and Chan, 2016).

2.3.3 Pedestrian Dead Reckoning

The pedestrian dead reckoning (PDR) algorithm has three components, step detection, step length estimation, and heading estimation. Steps are detected from the periodic

pattern of accelerometer readings while the user is walking; step length is estimated in real-time from the amplitude of the accelerometer readings or based on user characteristics; the user's heading is determined by magnetometer readings and the gyroscope can be used to detect and correct erroneous magnetometer readings from magnetic anomalies due to the physical environment. This method essentially estimates the user's movement based on the inertial sensor measurements provided by the mobile device. The starting location, however, needs to be provided by an external source for this system, or alternatively, the user may be located after a traversing a path sufficiently long and unique to be matched to a likely potential path in the floor plan (Durrant-Whyte and Bailey, 2006b, 2006a; Harle, 2013; Villien, Frassati and Flament, 2019).

2.3.4 Magnetic Anomaly Fingerprint Matching

Magnetic fingerprint matching, as the name implies, has functional similarities to WLAN fingerprint matching. Some of the materials used in the construction of larger commercial buildings, as well as large pieces of furniture within them, can create local anomalies in Earth's magnetic field measured in different areas of a building. The magnetometers available in modern smartphones are able to measure the magnetic field with enough precision to be able to distinguish local field anomalies throughout a building. Thus, a live sample of the magnetic field (fingerprint) is matched to the most similar location in a reference fingerprint database based on the magnetic field measured. Since these magnetometers measure the magnetic field in the three physical

dimensions, this method is functionally equivalent to a WLAN fingerprint matching system where the signal from three APs is sensed throughout the whole building (Li *et al.*, 2012).

2.3.5 Image Recognition and Visual Odometry

Methods taking advantage of the mobile device's camera sensor can be grouped into two broad categories, scene recognition and visual odometry. The former refers to methods that attempt to recognize the location of the user based on photo(s) taken by their mobile device in real-time. This can be accomplished by simple image matching if a considerable database of images with known location exists; alternatively, live images can be used to construct a three-dimensional model of the scene in view and that model then matched to a particular area of the building based on an existing accurate three-dimensional model of the building's interior (Mautz and Tilch, 2011). Visual odometry, on the other hand, uses live video from the camera to estimate the user's movement. By creating and continuously updating a three-dimensional model of the view in the live video, the change in perspective over time can convey the user's movement. Visual odometry is thus very similar to PDR, with the main difference just being how the user's movement is estimated (Nister, Naroditsky and Bergen, 2004).

2.3.6 Sensor Fusion

Finally, hybrid algorithms combine multiple individual positioning methods into a larger more complex positioning system (Hafner *et al.*, 2013; Ebner *et al.*, 2014;

Herrera *et al.*, 2014; Subbu *et al.*, 2014; Chen *et al.*, 2015; Tian *et al.*, 2015; Zhang, Chen and Xue, 2018; Dumbgen *et al.*, 2019; Renaudin *et al.*, 2019; Villien, Frassati and Flament, 2019; Guo *et al.*, 2020). Some implementations employ as many individual algorithms as possible and then attain a final position by averaging all of the estimates, sometimes varying the influence of individual estimates depending on their expected accuracy. Other implementations choose the individual algorithms strategically so that they may reinforce each other's weaknesses; a common example is a WLAN algorithm and PDR algorithm working cooperatively; the WLAN algorithm is preferred to estimate an initial location, while the PDR algorithm is generally superior in estimating the user's movement relative to an initial location, and thus, is given precedence when the user is moving (Harle, 2013; Subbu *et al.*, 2014; Villien, Frassati and Flament, 2019). A general overview of a state-of-the-art sensor fusion algorithm follows.

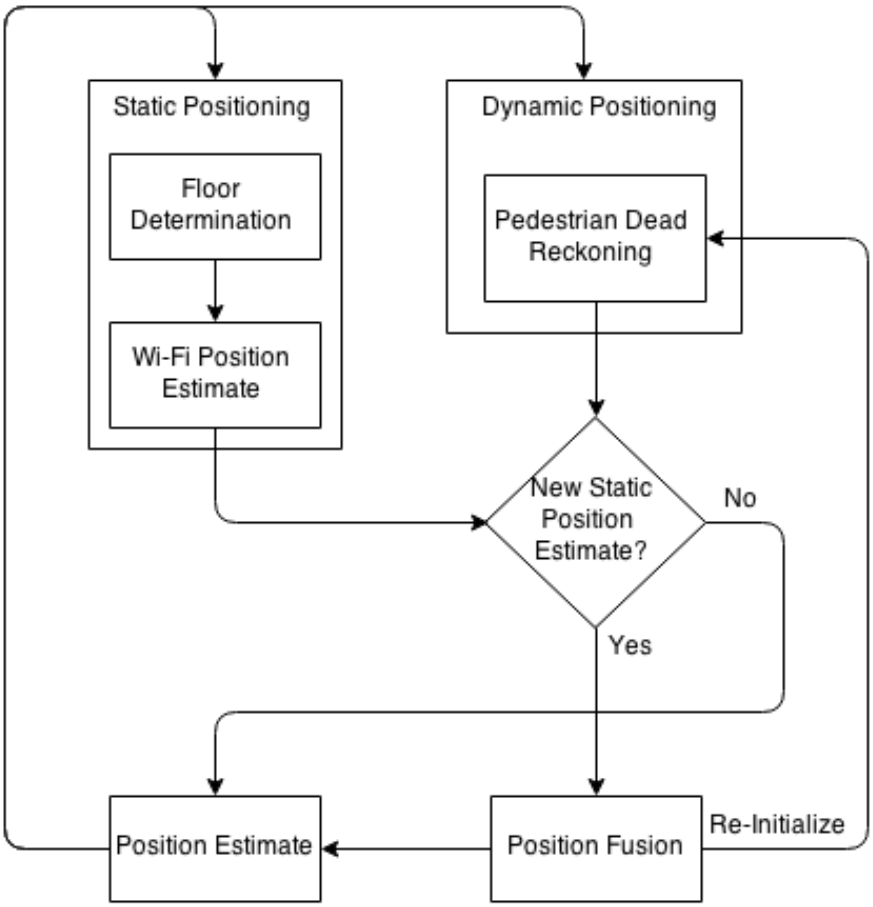

Figure 1. System Overview of a Typical Sensor Fusion Algorithm

3. Location Fingerprinting

This chapter describes the physical study area of the research, including the buildings and their various relevant characteristics; it outlines the process, methods and tools of data collection, and it describes how the data is processed and arranged for use in the implemented positioning systems.

3.1 Study Areas

The positioning system was implemented on two floors of in each of the following three buildings: Petrie Science and Engineering Building (PSE), Chemistry Building (CB), and the Bergeron Centre for Engineering Excellence (BRG). Hallways on each floor were segmented into discrete symbolic locations and each symbolic location was individually fingerprinted.

3.1.1 Building Floorplans and Symbolic Locations

The Bergeron Centre of Engineering Excellence (BRG) building, built in 2015, has four floors. The first and second floor were sampled and reference WLAN RSS fingerprints were created for them. The first floor main narrow hallways accumulate to approximately 90m (sections of 40m, 20m, 20m and 10m) while the first floor main large hallway is 35m long and has two adjacent approximately 80m^2 open areas. The second floor has a main hallway that is L-shaped and approximately 65m long (22.5m + 42.5m) while the second floor secondary hallways accumulate to approximately 35m

(10m +15m + 10m). BRG 1st floor consists of 105 distinct symbolic locations and BRG 2nd floor consists of 55 distinct symbolic locations.

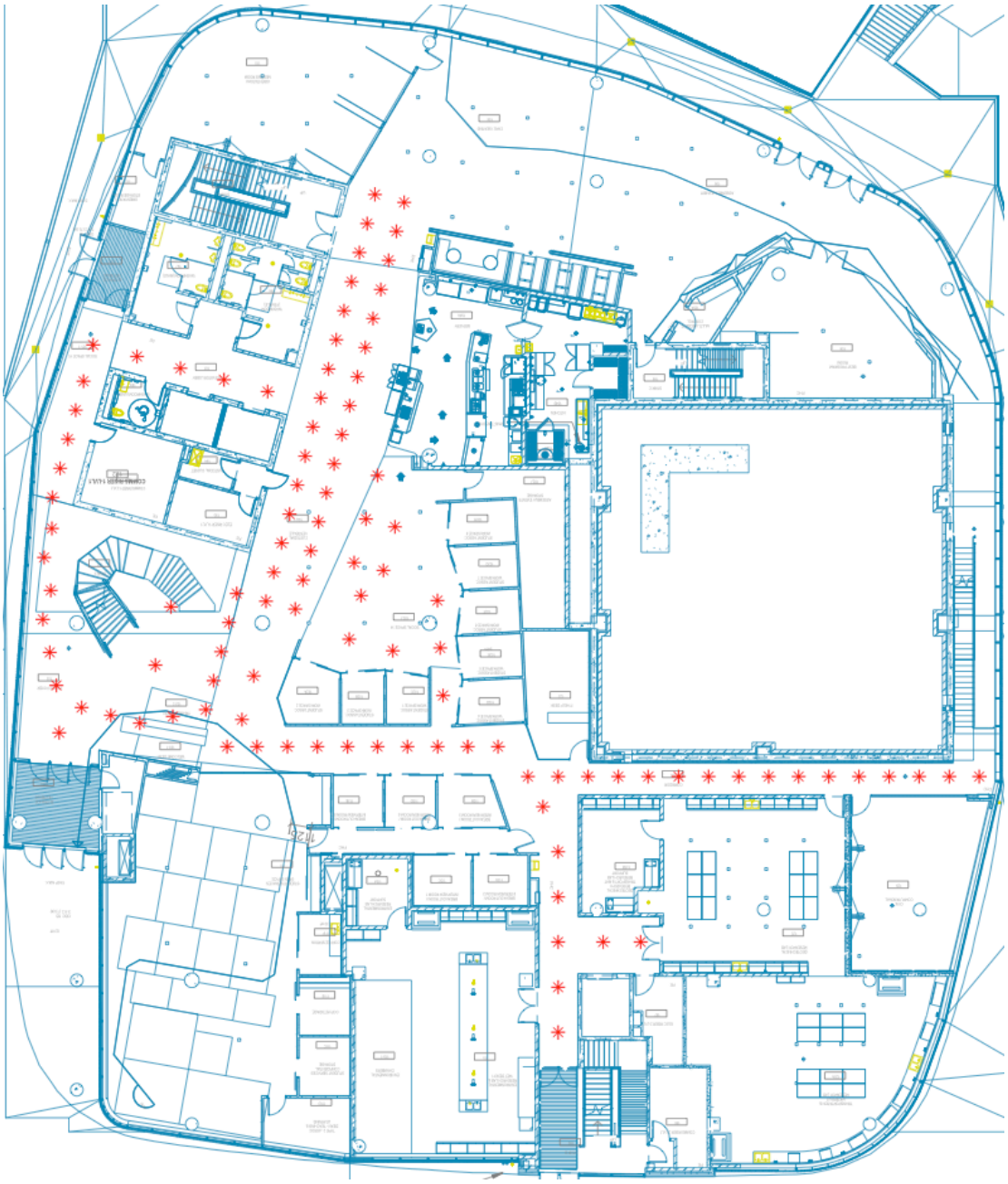

Figure 2. BRG 1st Floor and Symbolic Locations

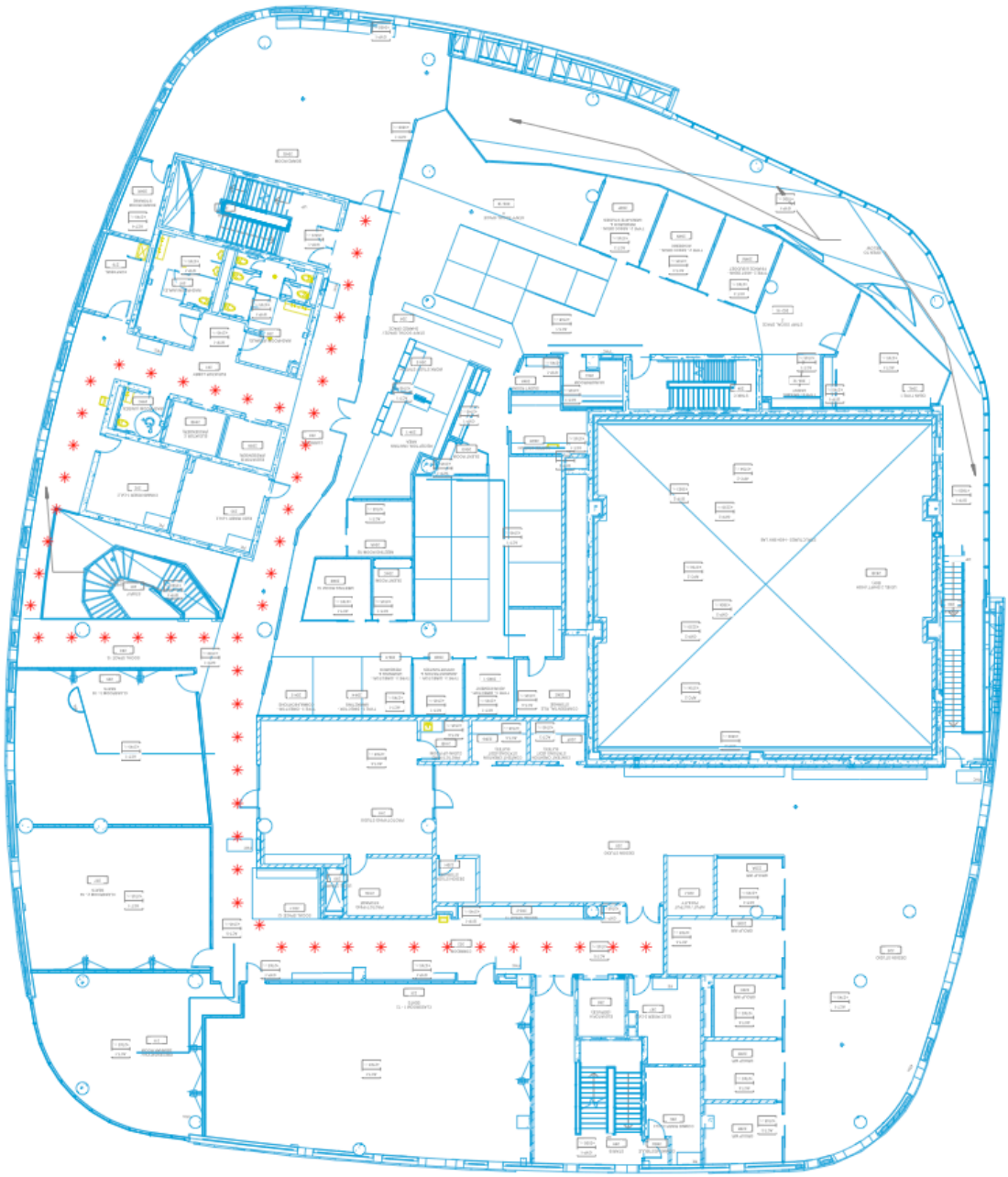

Figure 3. BRG 2nd Floor and Symbolic Locations

Figure 4. CB 1st Floor and Symbolic Locations

The Chemistry Building (CB), built in 1993, has four floors and is connected to the PSE building by through hallways on a few of its floors. The first and fourth floors were sampled and reference WLAN RSS fingerprints were created for them. The fourth floor main L-shaped hallway is approximately 100m long (55m + 45m) while the first floor main L-shaped hallway is approximately 105m long (40m + 20m + 45m), and the

first floor through hallway is approximately 25m long. CB 1st floor consists of 85 distinct symbolic locations and CB 4th floor consists of 70 distinct symbolic locations.

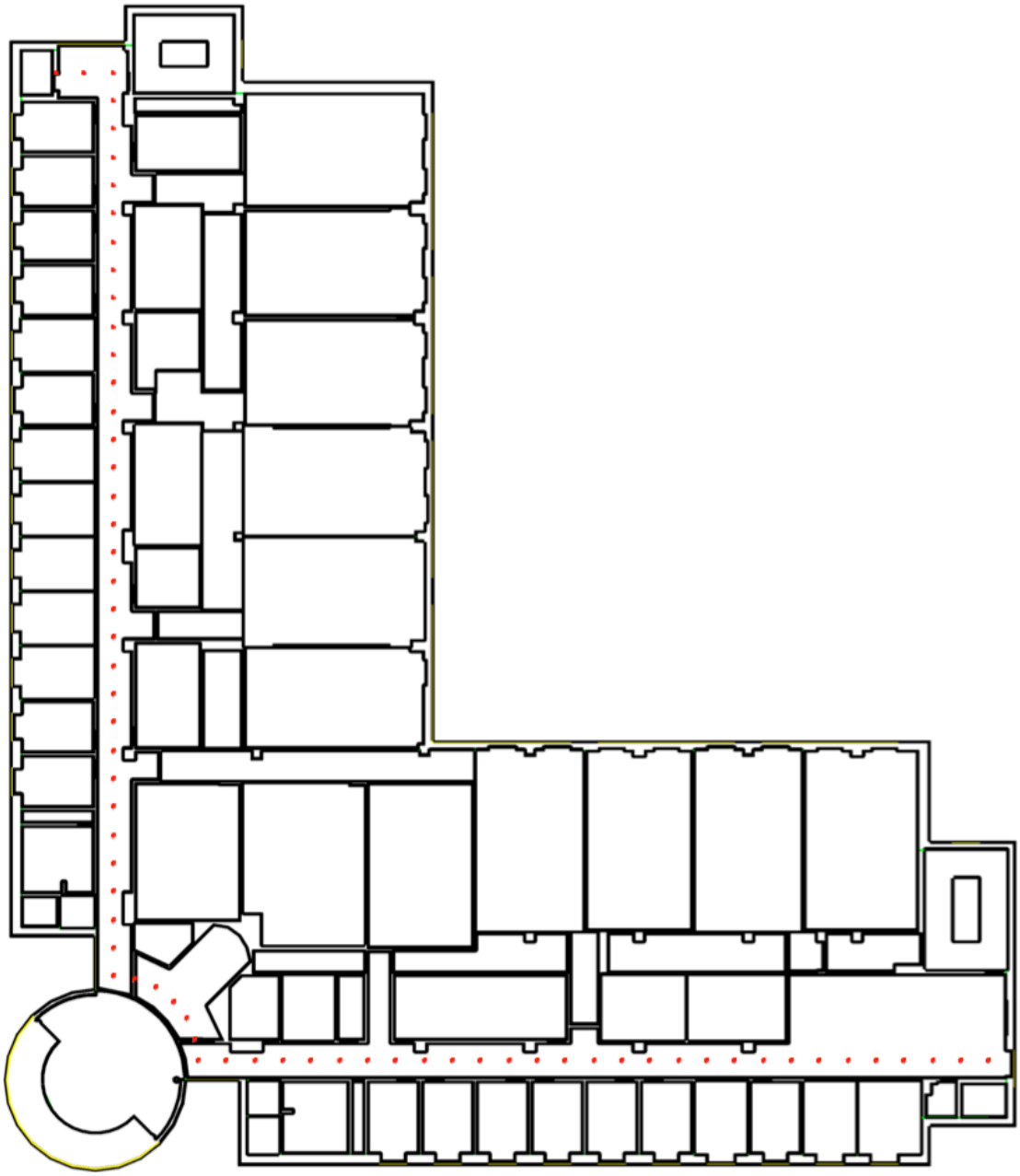

Figure 5. CB 4th Floor and Symbolic Locations

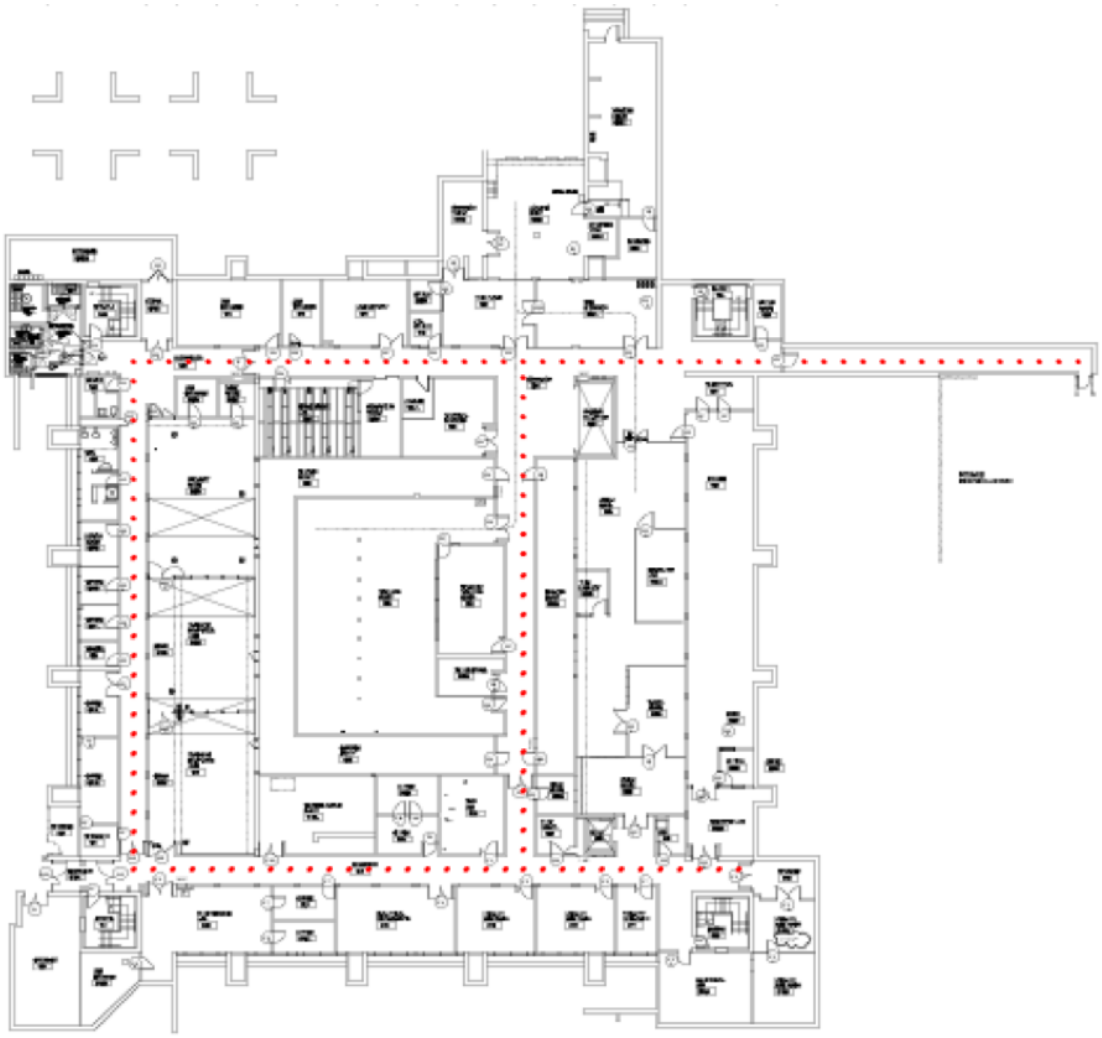

Figure 6. PSE Basement Floor and Symbolic Locations

The Petrie Science and Engineering (PSE) building, built in 1968, has five floors including the basement. The basement and third floor were sampled and reference WLAN RSS fingerprints were created for them. The third floor hallways form a square with approximately 40m sides, while the basement hallways consist of a 30m by 40m rectangle and two additional 18m hallways extending two of the rectangle's sides. PSE

Basement floor consists of 115 distinct symbolic locations and PSE 3rd floor consists of 104 distinct symbolic locations.

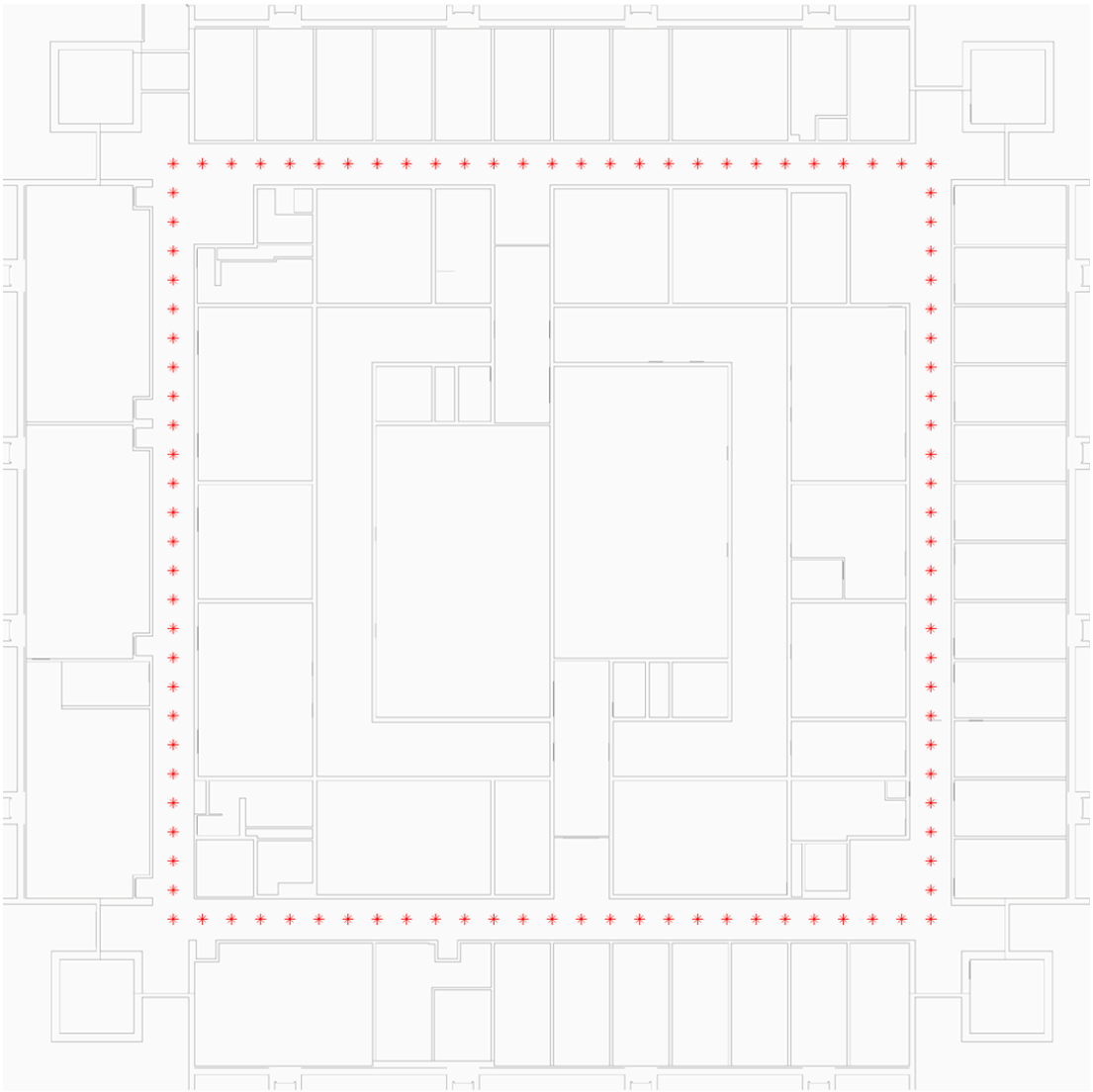

Figure 7. PSE 3rd Floor and Symbolic Locations

3.1.2 AP Locations

PSE has 33 active APs distributed throughout the 3rd floor. They are mainly placed within the offices, with only a couple being in the hallways and only a couple more in the central labs. The distribution is most dense flanking the main hallway, thus it is expected that positioning based on WLAN signals will be relatively accurate here.

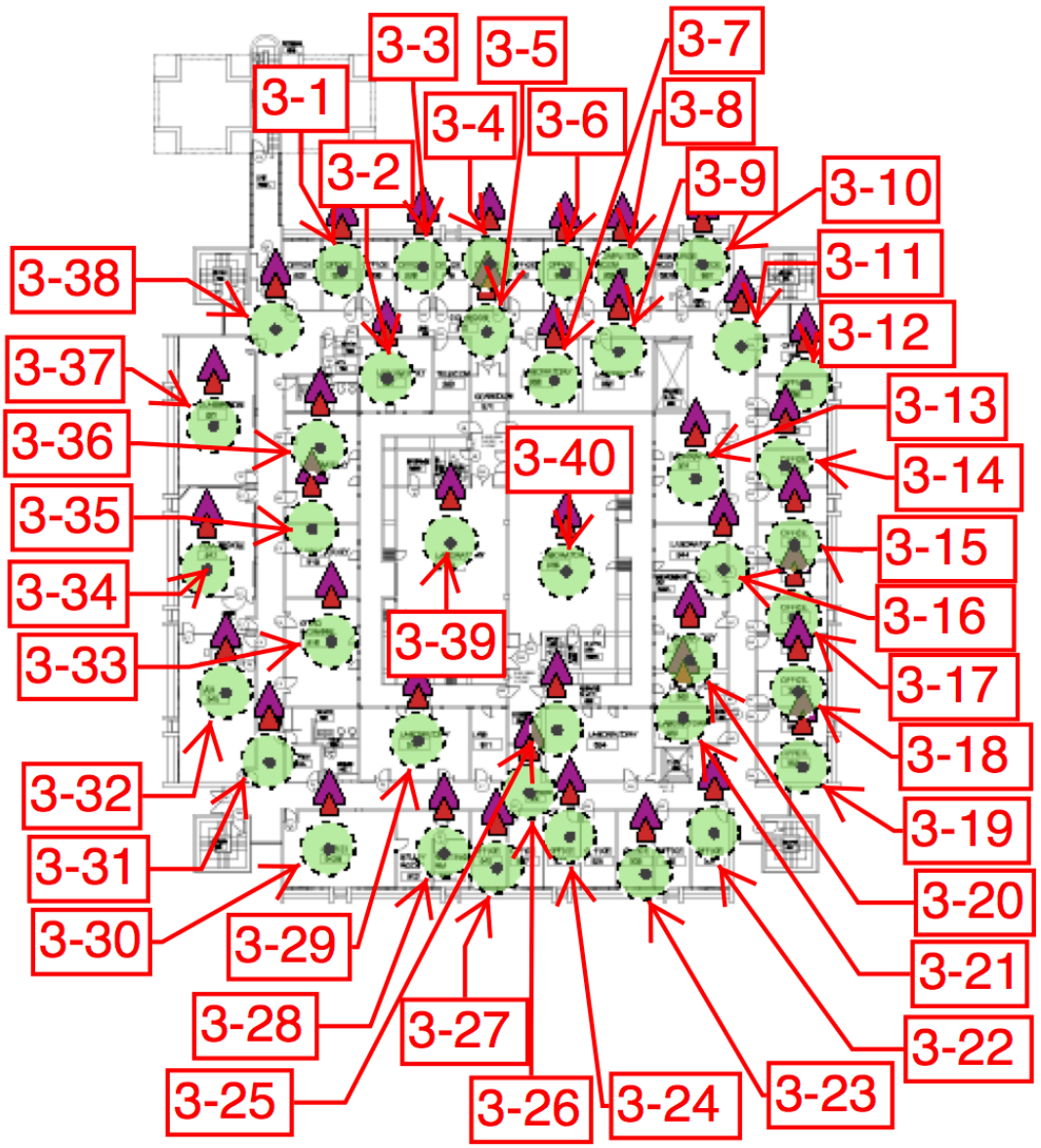

Figure 8. AP Distribution on PSE Third Floor

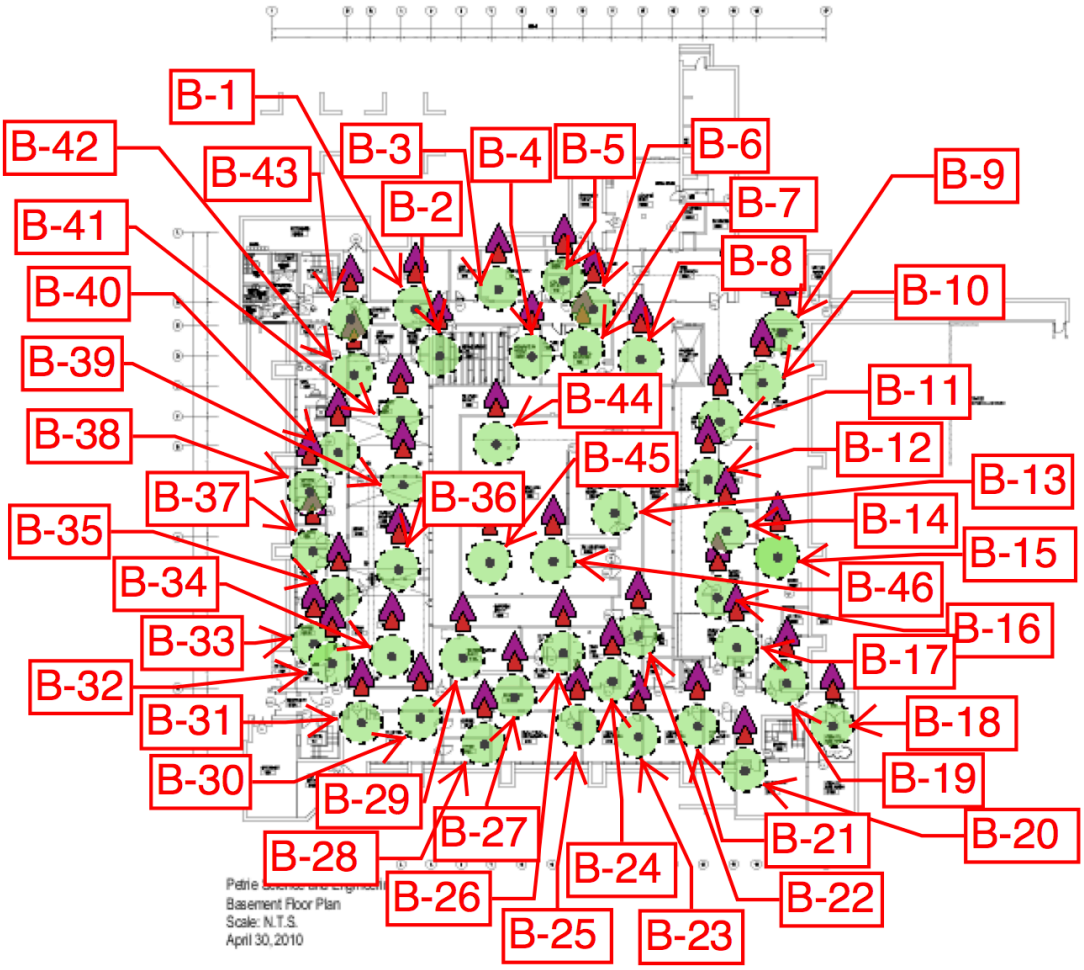

Figure 9. AP Distribution on PSE Basement Floor

PSE has 29 active APs distributed throughout the Basement floor. Similarly to the third floor, the distribution is most dense flanking the main hallway, thus it is expected that positioning based on WLAN signals will be relatively accurate here, except perhaps in the top-right section of the map and the long narrow hallway extending from there.

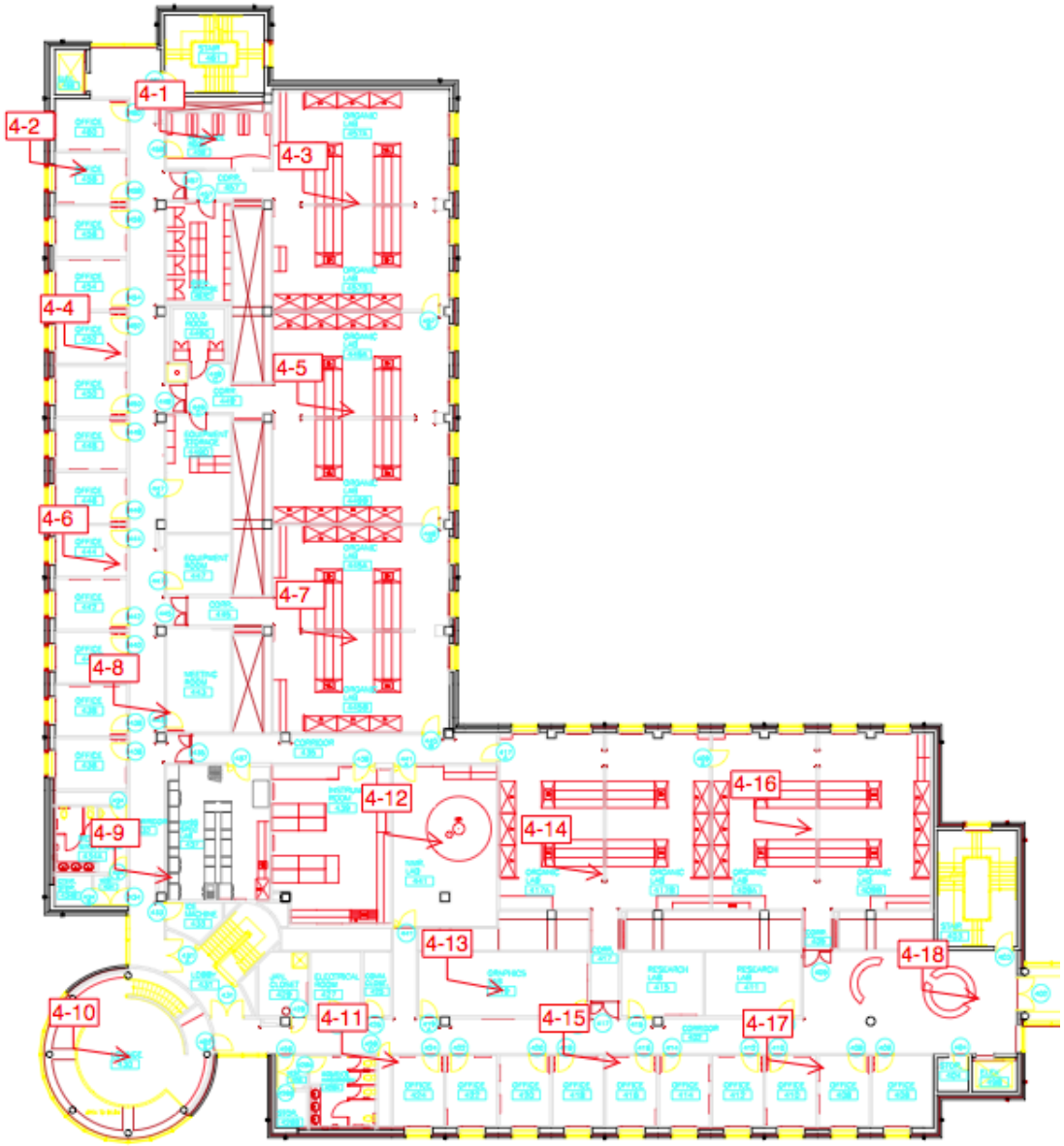

Figure 10. AP Distribution on CB Fourth Floor

CB has 18 APs distributed throughout the 4th floor. The distribution is more sparse but still relatively uniform throughout the floor. Positioning based on WLAN signals is expected to be generally good here as well.

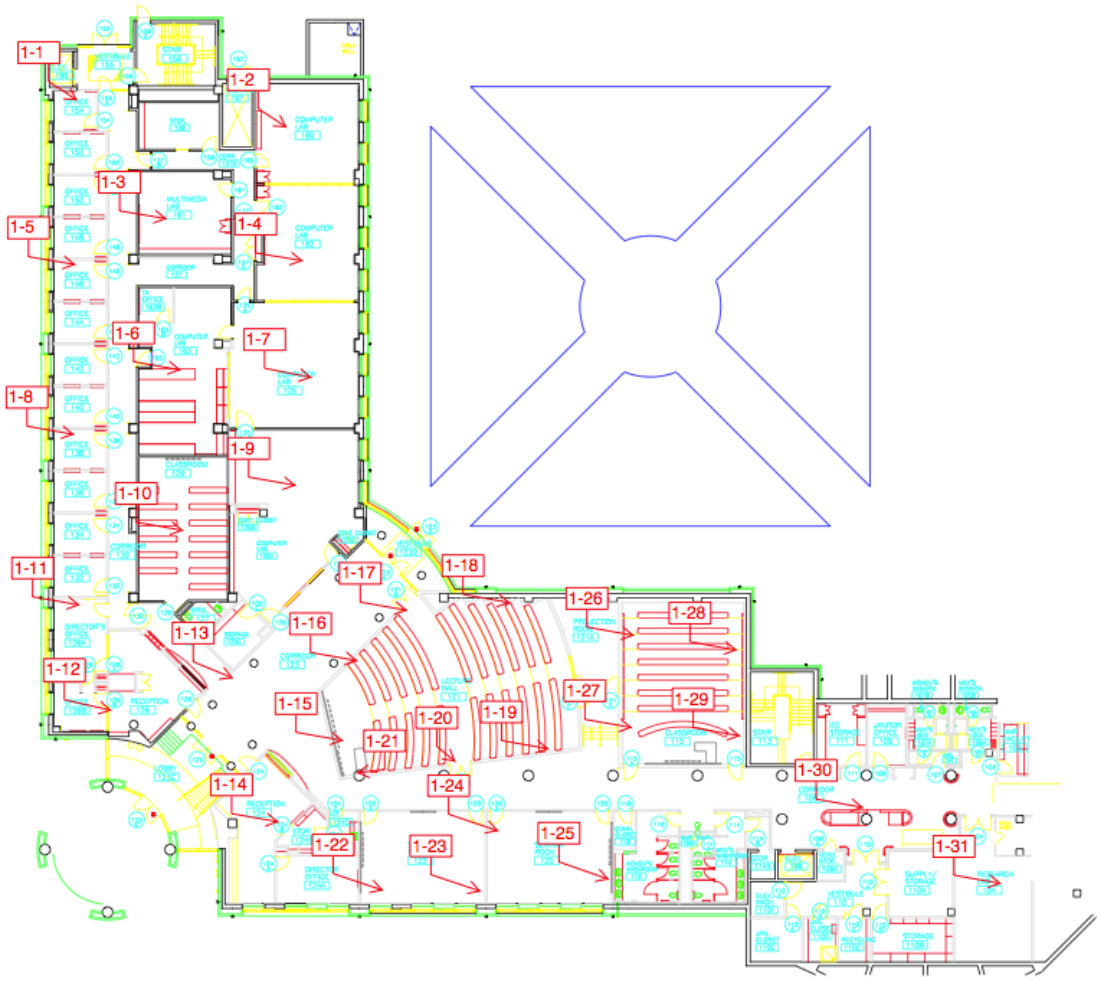

Figure 11. AP Distribution on CB First Floor

CB has 31 APs distributed throughout the 1st floor. This floor has a similar area to the 4th floor of CB, however, it has a much higher number of APs, resulting in a higher density. The distribution is similar to that of the 3rd floor of the PSE building, thus, WLAN based positioning accuracy is expected to be very good here.

Figure 12. AP Distribution on BRG Second Floor

BRG has 45 APs distributed throughout the 2nd floor. The distribution of APs is quite uniform and dense throughout the navigable areas of this floor, thus, positioning based on WLAN signals is expected to be relatively accurate here. The large square space in

the middle left sixth of the floor is a large open industrial experiment lab in which no APs could be installed, and the far left hallway is open to below.

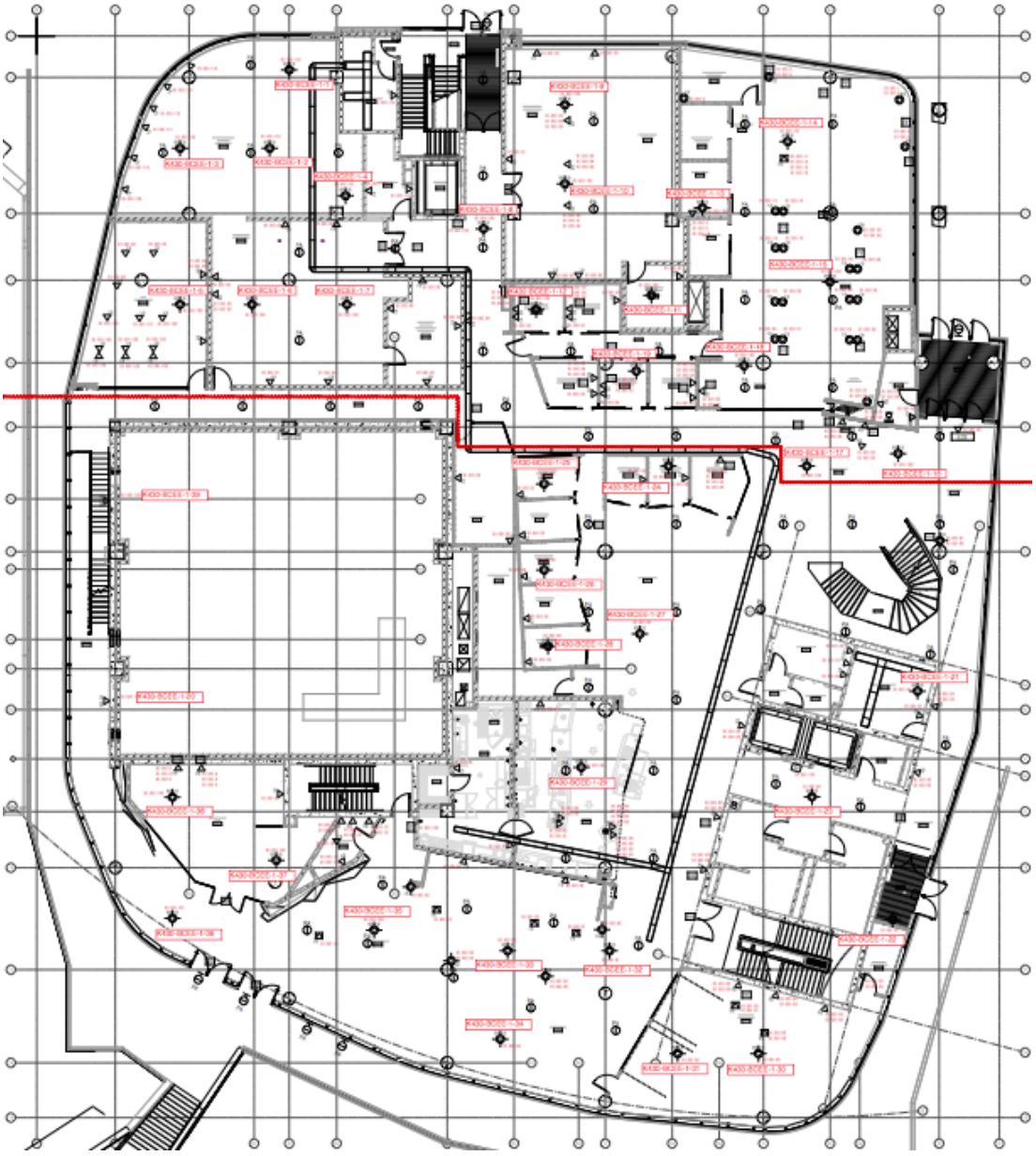

Figure 13. AP Distribution on BRG First Floor

BRG has 39 APs distributed throughout the 1st floor. The distribution is relatively uniform and dense in the top and bottoms thirds of the floor, thus, WLAN based positioning is expected to be relatively accurate in these areas. However, the distribution is rather sparse near the far left hallway and the middle right sixth of the floor, which will result in relatively poorer performance of the positioning in these areas. The large square space in the middle left sixth of the floor is a large open industrial experiment lab in which no APs could be installed.

3.1.3 WLAN Signal Characteristics

The APs used by York University in their public WLAN infrastructure are set to transmit on both 2.4 and 5 GHz frequencies. Furthermore, they offer access to three different networks. This results in each physical AP presenting six virtual APs to users. The impact of virtual APs on WLAN fingerprint matching was discussed in (Farshad *et al.*, 2013), and the results therein indicated that considering virtual APs as physical APs will have a similar positive effect on positioning accuracy to increasing the density of physical APs in the indoor space.

Another finding from (Farshad *et al.*, 2013) was that a 5 GHz AP signal is more reliable than its 2.4 GHz counterpart for WLAN fingerprint matching. In addition, in practice, WLAN scans on a mobile device become more frequent when the device is limited to using only one of the frequencies. The trade-off of using only fingerprints on one frequency is explored experimentally.

3.2 Data Collection

3.2.1 Collection Tools

An Android application was specifically developed for collecting WLAN AP fingerprints for this research.

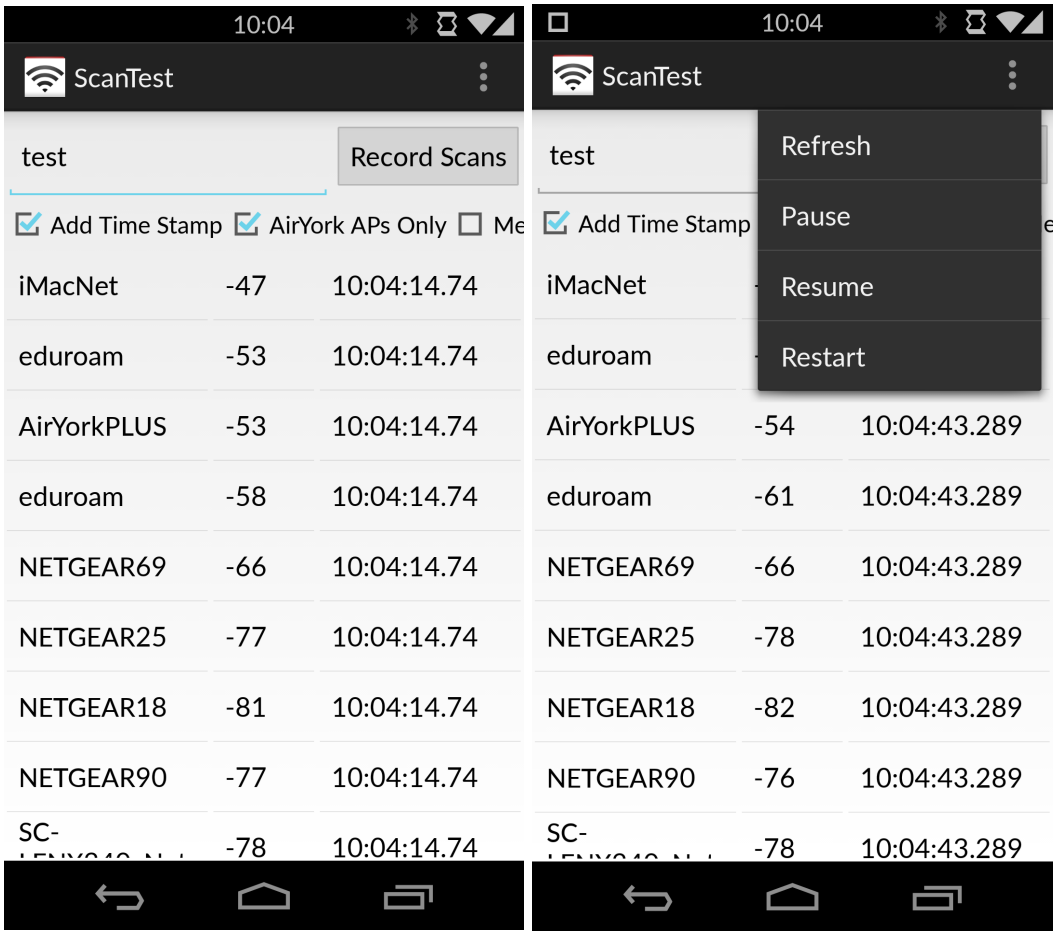

Figure 14. Screenshots of the application used to collect fingerprints.

The application interface shows a constantly updating list of visible APs, showing their SSID, RSSI, and time of the reading. The interface also contains a text box for the name of this collection instance and a button to write the collected readings to a .txt file. Furthermore, options exist to timestamp the collection, filter out non-university

infrastructure APs, and to compute and record only the mean of the RSSI readings obtained for each AP. The scanning for APs can be paused/resumed, restarted, and manually refreshed if automatic scans stop occurring. The application was installed on multiple devices that were used for collecting fingerprints over the course of the research. Devices included the LG Nexus 4 and 5, Samsung Nexus 10, Sony Xperia Z3 Compact, and OnePlus One.

Table 1. Phones Used for Data Collection

Phone	CPU	WLAN
LG Nexus 4	Qualcomm Snapdragon™ S4 Pro (APQ8064), Quad-Core 1.5GHz	Dual-Band Wi-Fi (2.4G/5G) 802.11 a/b/g/n
LG Nexus 5	Qualcomm Snapdragon™ 800 (MSM8974), Quad-Core 2.26GHz	Dual-band Wi-Fi (2.4G/5G) 802.11 a/b/g/n/ac
Samsung Nexus 10	Exynos 5250, Dual-Core 1.7 GHz Cortex-A15	Dual-Band Wi-Fi (2.4G/5G) 802.11 a/b/g/n
Sony Xperia Z3 Compact	Qualcomm Snapdragon™ 801 (MSM8974AC), Quad-Core 2.5 GHz	Dual-band Wi-Fi (2.4G/5G) 802.11 a/b/g/n/ac
OnePlus One	Qualcomm Snapdragon™ 801 (MSM8974AC), Quad-Core 2.5 GHz	Dual-band Wi-Fi (2.4G/5G) 802.11 a/b/g/n/ac

3.2.2 Collection Methods

At each discrete symbolic location, fingerprint collection lasts approximately two minutes. The subject performing the data collection holds the mobile device in their

hand at about chest height, inputs the location information/name into the fingerprint collection app, resets the collection process (by clicking “restart”) and rotates their body 90˚ after every 30 seconds of data collection. After two minutes have passed and the subject has collected fingerprints in all four directions, the “Record Scans” button is pressed to record the collected data to a .txt file and complete the process for that particular symbolic location.

3.2.3 Collected Data

- Building
    - Floor
        - Symbolic Location 001
            - AP_1 (MAC, SSID, number of RSSI readings)
                - $RSSI_1$:timestamp
                - $RSSI_2$:timestamp
                - …
            - AP_2 (MAC, SSID, number of RSSI readings)
                - $RSSI_1$:timestamp
                - $RSSI_2$:timestamp
                - …
            - …
        - Symbolic Location 002
            - AP_1 (MAC, SSID, number of RSSI readings)
                - $RSSI_1$:timestamp
                - $RSSI_2$:timestamp
                - …
            - AP_2 (MAC, SSID, number of RSSI readings)
                - $RSSI_1$:timestamp
                - $RSSI_2$:timestamp
                - …
            - …
        - …

Figure 15. Format of Collected Data

3.3 Fingerprint Database

3.3.1 Storage

Primarily, the fingerprint database is stored on .txt files; each symbolic location has one text file from each device used to collect fingerprints at that location. The format of each file is as follows:

Symbolic Location 001 (# of APs)

- AP_1 (MAC, SSID, number of RSSI readings)
  - $RSSI_1$:timestamp
  - $RSSI_2$:timestamp
  - …
- AP_2 (MAC, SSID, number of RSSI readings)
  - $RSSI_1$:timestamp
  - $RSSI_2$:timestamp
  - …
- …

Figure 16. Format of a Reference Fingerprint for a Symbolic Location

Once the positioning algorithm is initialized (.txt files are read and a *live* database structure is created on memory), the positioning application can create a backup txt file of the database containing only the simplified information (see section 3.3.3) in order to speed up initialization for the next time positioning is attempted on the same floor with the same initialization parameters.

3.3.2 Filtering

The main filtering that occurs is with respect to the APs that are considered. The data collection application collects readings from all APs in range, however, the positioning algorithm only uses readings from the APs that are part of the university's WLAN infrastructure. When creating the *live* database, APs are filtered based on their SSID (whitelist of university infrastructure SSID's used). Other filtering may also occur based on the initialization parameters, for example, APs with very erratic RSSI readings (beyond a certain threshold), or APs with very low mean RSSI (again, beyond a certain threshold), may be removed from the database.

Finally, filtering generally also occurs for real-time readings during positioning as well as test readings used for performance evaluation. The filtering in these cases occurs on a fingerprint-by-fingerprint basis; a fingerprint may be discarded or combined with the following/preceding fingerprint depending on the number of APs represented in that fingerprint and/or the timestamps of RSSI readings contained in it. Occasionally, fingerprints read by the positioning algorithm are incomplete, thus, they will contain fewer RSSI readings than expected for that particular location; the remaining expected RSSI readings then come in the following fingerprint with timestamps a millisecond later than the previous fingerprint. Thus, in these cases, the two fingerprints are combined into one before being input in the positioning algorithm. In the much rarer cases where the incomplete fingerprint is not followed by its complementary incomplete fingerprint, the fingerprint is simply discarded as it is deemed unreliable.

The general criterion for designating a fingerprint as incomplete is a threshold for the minimum expected AP RSSI readings for any fingerprint attained on a particular floor; this threshold is based on the number of APs visible at any given position on that floor.

3.3.3 Simplification

In this context, simplification refers to the creation of the *live* fingerprint database used by the positioning algorithm in real-time. During initialization, the application reads all of the fingerprint txt files for all symbolic locations on the building and floor where the user is to be positioned. The application then creates a data structure that contains reference fingerprints for every symbolic location on that floor. Generally, a mean value is calculated from the multiple RSSI readings for each AP in the txt files; this mean value is then added to the data structure and the individual RSSI readings are discarded to conserve memory. Depending on the initialization parameters, the data structure could also contain the number of readings as well as a measure of the variability of the signal strength for each AP at each symbolic location. When the Bayes Maximum Likelihood algorithm is used for positioning, then normalized RSSI histograms are created for every AP at every symbolic location. To speed up initialization on future positioning sessions, all the info retained in the *live* fingerprint database is written to a new file, thus, allowing future initializations with the same parameters to recreate the *live* fingerprint database without having to do any of the computations in the first initialization.

3.4 Deployment Time

Total deployment time including data collection, processing and inclusion in the system, to enable positioning on a floor was about one working day. Planning out symbolic locations in a floor would require about 2 hours. Collecting data on those points would require about 2 to 4 hours, or approximately 1 hour of data collection for every 200 metres squared of navigable area. Finally, the symbolic locations and their respective reference WLAN signal fingerprints are added to the positioning application, requiring 1 to 2 hours, before positioning is enabled on the floor. Thus, a turnaround time of one day per floor or one week for an average building on a university campus was possible using the system developed as part of this research.

4. Location Estimation

In this thesis, I present a novel indoor positioning system that combines a conventional WSFM algorithm as a baseline positioning method, with which a novel concept of smooth movement regularity is integrated. In this chapter, the algorithmic details of the positioning system are discussed; it is further broken up into three subsections. The first two sections discuss each of the two general types of WSFM algorithms, deterministic and probabilistic, respectively; both types were implemented as part of this research, using *k*-NN and ML algorithms, respectively. The third subsection describes the details of the entirely novel PERA module and how it fits into the positioning system as a whole.

4.1 Introduction

The positioning system presented in (Bahl and Padmanabhan, 2000) is generally considered as the first using a WSFM algorithm. Since then, there have been many academic and commercial implementations, such as (Youssef and Agrawala, 2008; Martin *et al.*, 2010; Bolliger, 2011; Mirowski *et al.*, 2011; Beder and Klepal, 2012; Machaj and Brida, 2012; Laoudias, Zeinalipour-Yazti and Panayiotou, 2013; Bai *et al.*, 2014; *Ekahau RTLS*, 2015; He and Chan, 2016; Liu *et al.*, 2019), aiming to improve upon it. Given the multitude and diversity of these implementations, there have also been many academic research works exploring the field as a whole or examining the effects of modifying various parameters of WSFM algorithms (Kjærgaard, 2007;

Curran *et al.*, 2011; Dawes and Chin, 2011; Lemic *et al.*, 2014; Zhou and Wieser, 2018).

The comparative survey conducted by (Honkavirta *et al.*, 2009) was found to be particularly insightful; it provided detailed descriptions of the various algorithms it examined and was a useful reference for the design process of the WSFM algorithm discussed herein. On the other hand, findings in (Farshad *et al.*, 2013) addressed some of the questions regarding particular aspects of the implementation of the algorithm on the smartphone platform and in a university building setting. In addition to the above, previous research from this lab in indoor positioning utilizing WLAN signals (Chan, 2013), and its findings, also served to inspire the WSFM algorithm discussed in this work.

4.2 Positioning System Overview

The positioning algorithm posited in this research is principally made up of two separate algorithms, WLAN Signal Fingerprint Matching (WSFM) algorithm and a Path Evaluation (PE) algorithm. An overview of the entire positioning system described above is illustrated in Figure 17.

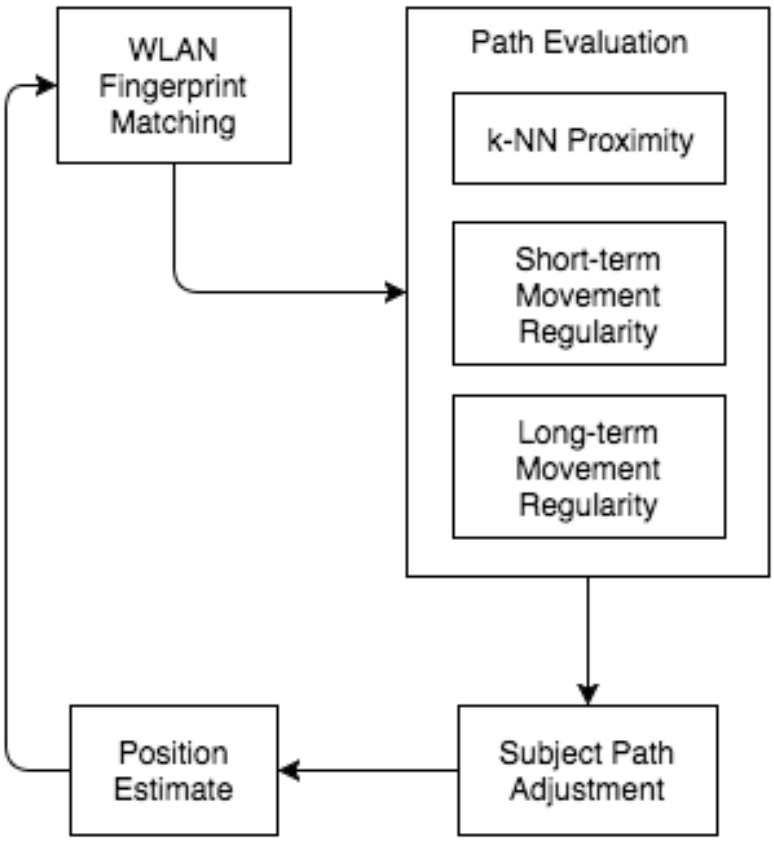

Figure 17. System Overview of k-NN + PERA Algorithm

Initially, the WSFM algorithm estimates the *k* most likely locations where the user may be located; these *k* locations are input for the second PE algorithm which determines the most likely recent path of the user based on the movement pattern in each of the permutations of the last few epochs of WSFM estimates. The user's recent path is then adjusted to correspond to the most likely estimate of the PE module and the path's end point is presented to the user as their current location. Each of the modules and sub-modules of the positioning system will be described in detail in this chapter.

4.3 *k*-Nearest Neighbours

4.3.1 *k*-Nearest Neighbours Algorithm

The *k*-Nearest Neighbours (*k*-NN) algorithm is a simple machine-learning algorithm, generally used in pattern recognition. The algorithm consists of searching through a set of reference points for the *k*-nearest points of that set to a query point in the same dimensional space. If there are multiple query points, the algorithm is repeated for each one. Nearness or proximity can be arbitrarily defined, although often Euclidean or Manhattan distance is used as a metric of proximity. Upon the identification of the *k* nearest reference points, those points can be used to either assign a class (*k*-NN classification) or assign a value (*k*-NN regression) to the query point.

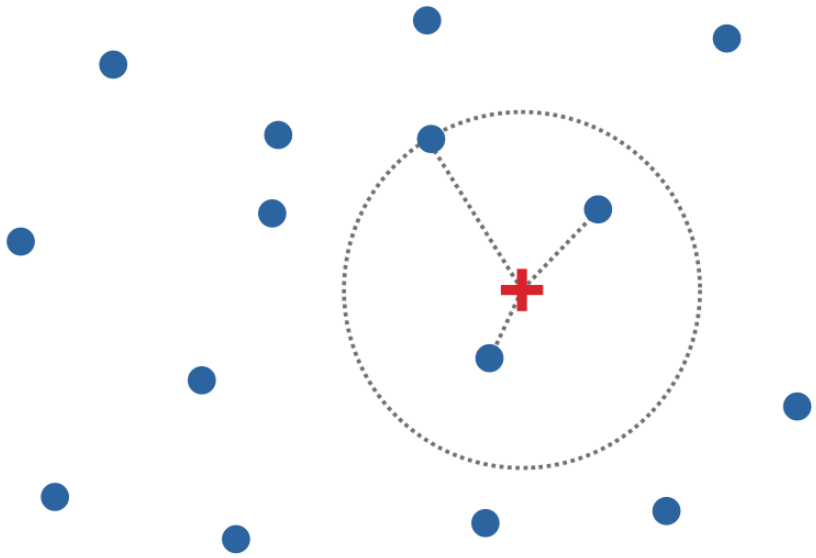

Figure 18. Diagram of k-Nearest Neighbours

4.3.2 Application to WLAN Fingerprint Matching

The k-NN algorithm can be utilized in WLAN Signal Fingerprint Matching (WSFM), a method of positioning in (mostly indoor) areas with WLAN infrastructure. WSFM requires a database or map of the signal strength of all WLAN access points (APs) throughout the area where positioning is possible. The user then obtains a sample of the signal strengths of all APs (fingerprint) at their particular location, and the WSFM algorithm matches the fingerprint to the most similar location (based on the AP signal strengths) in the database. A fingerprint typically contains the information in the table below, with the set of RSSI readings, denoted by $RSSI^S = \{RSSI_1, RSSI_1, RSSI_1, \dots, RSSI_n\}$, used as the input/measurement for *k*-NN.

Table 2. Information Typically Collected in a Fingerprint

Time Received	AP Network Name	AP Identifier	Signal Strength
t_1	$SSID_1$	$BSSID_1$	$RSSI_1$
		$BSSID_2$	$RSSI_2$
		$BSSID_3$	$RSSI_3$
	$SSID_2$	$BSSID_4$	$RSSI_4$
	$SSID_3$	$BSSID_5$	$RSSI_5$
		$BSSID_6$	$RSSI_6$
	$SSID_4$	$BSSID_7$	$RSSI_7$
		$BSSID_8$	$RSSI_8$
	$SSID_5$	$BSSID_9$	$RSSI_9$

The *k*-Nearest Neighbours algorithm (*k*-NN) is used to match the subject's fingerprint to the symbolic location with the reference fingerprint most similar to it. Every location estimation, that is, whenever the subject collects a new fingerprint, *k* -NN compares the subject's fingerprint to every single symbolic location in the reference fingerprint database and returns the *k* symbolic locations that are nearest to it in terms of AP signal strengths.

When using k-NN, the database is set up as a set of reference points (or reference fingerprints) in signal space that correspond to particular locations in the positioning area. Thus, let the reference database be $L = \{l_1, l_2, \dots, l_p\}$, where p represents the number of specific locations represented in the database and l_i typically contains the information in the table below. Also contained in l_i is $RSSI^R = \{RSSI_1, RSSI_1, RSSI_1, \dots, RSSI_m\}$, the reference point in signal-space used in *k*-NN. Finally, once all the distances are calculated between $RSSI^S$ and every $RSSI^R$, they are sorted and the *k* points in *L* with the smallest distances to the fingerprint are selected to estimate a location for that fingerprint. In my research, I am also experimenting with a modified version of this algorithm (WSDFM) that uses a different definition of the fingerprint, namely the differences between the signal strengths, as opposed to the absolute signal strengths, of all visible APs. Therefore, distance calculations in this version of the algorithm are in $d(d-1)/2$ dimensions when *d* APs are visible in the fingerprint.

Table 3. Table of Reference Fingerprints

Symbolic Location	Geometric Location	AP Network Name	AP Identifier	Mean Signal Strength
001	$(x,y)_{001}$	$SSID_1$	$BSSID_1$	$RSSI_1$
			$BSSID_2$	$RSSI_2$
			$BSSID_3$	$RSSI_3$
		$SSID_2$	$BSSID_4$	$RSSI_4$
		$SSID_3$	$BSSID_5$	$RSSI_5$
			$BSSID_6$	$RSSI_6$
		$SSID_4$	$BSSID_7$	$RSSI_7$
			$BSSID_8$	$RSSI_8$
		$SSID_5$	$BSSID_9$	$RSSI_9$

4.3.3 Definition of Proximity

Based on empirical findings during my experimentation and similar findings in prior research from our lab (Chan, 2013), the metric of proximity chosen for my implementation is a normalized Manhattan distance; for each fingerprint's $RSSI^S$, the distances are calculated between it and the $RSSI^R$ of every reference fingerprint in L. The proximity metric, $d_{SS}(l)$, is described by the equation below.

$$d_{SS}(l) = \frac{\sum_{i=1}^{m} \left| RSSI_i^{R(l)} - RSSI_i^{S} \right| + \left(\beta_0 (m-n)\right)}{m} \tag{1}$$

$$L^{k\min} = \arg \underset{k=5}{k\min}_l \, d_{SS}(l), \qquad L^{k\min} \subset L \tag{2}$$

The symbolic locations corresponding to the k-smallest distances in signal space ($L^{k\min}$) are identified. Furthermore, the dimension mismatch penalty (β_0) is discussed in subsection 4.3.4.3.

4.3.4 Mismatch In Distance Dimensions

It should be noted that $RSSI^S$ and $RSSI^R$ are often not defined in the same dimensions, that is to say *n* and *m*, the APs represented in $RSSI^S$ and $RSSI^R$, respectively, are often not the same. Not all APs are necessarily visible in every location of the positioning area, thus, the potential mismatch of dimensions must be dealt with.

4.3.4.1 Normalization

One way to deal with a mismatch of dimensions is to normalize the proximity metric, in this case, Manhattan distance. This allows the proximity metric to represent the average distance per dimension as opposed to the combined distance in all dimensions. A solution that relies solely on normalization will compute a distance using only the dimensions that match between $RSSI^S$ and $RSSI^R$, i.e., those that correspond to the same APs. This means that non-matching dimensions are simply ignored and this can result in outliers and generally poor accuracy when the number of matching dimensions is very low.

4.3.4.2 Artificial RSS

Another way to deal with dimension mismatch is to insert artificial values into $RSSI^S$ for the dimensions of $RSSI^R$ that are not represented in $RSSI^S$. In this case, the value inserted (RSS*) is usually the lowest possible RSS value that can be sensed, often set to -100 dBm. This solution makes the assumption that RSS* is *practically* equivalent to the real RSS in that spot (which is too low to be sensed by the subject's device). Supporting rationale for this method is that by inserting a set minimum value (RSS*), the penal impact on the metric for that reference location will be proportional to the strength of the missing AP signal (missing dimension).

4.3.4.3 Hybrid Solution for Mismatch

The two methods of dealing with dimension mismatch discussed above can be used together in a hybrid solution where the artificial RSS* readings are inserted into the $RSSI^S$ and then the Manhattan distance is also normalized. The rationale for the latter is that even $RSSI^R$ of different symbolic locations will vary in the dimensions represented (APs *visible*), thus, normalization can still help make the proximity comparison fairer.

This hybrid method certainly improves the accuracy of positioning of the *k*-NN algorithm, however, during performance testing, a different hybrid method (penalization method) was also tested and showed to further improve accuracy. The penalization method opts to add a set penalty to the Manhattan distance for every

dimension mismatch as opposed to employing artificial RSS readings. This penalty is essentially an artificial difference between $RSSI^R$ and $RSSI^S$ in one dimension. The actual penalty was found heuristically by trying various values, and tests showed that using one constant value for the penalty resulted in better accuracy than the varying penalization resulting from inserting the artificial RSS values. This penalization method is demonstrated in the mathematical definition of our proximity metric in the $(\beta_0(m-n))$ term. Here, β_0 is the penalty value and $(m-n)$ determines how many times to add the penalty to the proximity metric (i.e. how many dimensions are represented in $RSSI^R$ but not in $RSSI^S$.

4.3.5 *k*-NN Weighting

Whether performing k-NN classification or regression, when k > 1, there needs to be some kind of scheme to combine the input/effect of all k neighbours. In the simplest case, an equal weighted mean of the neighbours is computed. Another very common method is to take a weighted mean of the neighbours, where the weight of each neighbour is inversely proportional to its rank or k-value. Other, more complex, methods can also take into account the actual metric (e.g. Euclidean distance) of the nearest neighbours calculation for each of the neighbours to determine that neighbour's weight, or simply retain all of the neighbours and use an additional algorithm to obtain a single estimate from the *k* neighbours. The various methods of assigning weight to the neighbours can also be broadly categorized into two categories, a priori and a

posteriori weighting; however, some complex methods can even perform both types of weighting.

4.3.5.1 A Priori Weighting

A priori weighting generally refers to scalar constant weights applied to the neighbours. This generally includes the trivial case, where all neighbours are equally weighted, and cases where the weights are based on the rank. Rank based weights will generally be something along the lines of 1/Rank, or in other cases, each rank can have a specific arbitrarily predefined weight (i.e. $1^{st} = 1.0$, $2^{nd} = 0.8$, $3^{rd} = 0.6$, and etc.) generally based on some prior knowledge about the likelihood of each neighbour being *correct*.

4.3.5.2 A Posteriori Weighting

A posteriori weighting generally refers to the determination of weights for the neighbours after the *k*-nearest neighbours have been determined. This can be accomplished in a number of ways. The simplest way is to use the distance metric of each neighbour that determined its proximity to the query point (e.g. the weight could be $1/d_i$). A more complex way is to use an alternate algorithm to determine the weights of the neighbours. The secondary algorithm can be entirely independent of the k-NN metric, and thus, can result in weights that are not proportional to the k-NN ranks of the neighbours. In my experimentation, I attempted the use of a Bayes Maximum Likelihood algorithm to calculate a posteriori weights of the k-nearest neighbours,

however, the algorithm was not able to improve accuracy significantly over the use of a priori or k-NN based a posteriori weights.

4.3.6 Uncertainty

Given that this positioning system is based on calculating a position from measured data, it stands to reason that there is some uncertainty in the measurement and that uncertainty could be used to estimate the level of confidence or uncertainty in the estimated position. Unfortunately, in the implementation of this algorithm on mobile devices, determining and propagating uncertainty is not possible. In early versions of WSFM algorithms, WLAN signal measurements were taken on computers, where software allowed for high frequency sampling of signals in the order of 10-100 Hz. On the other hand, mobile devices only allow access to the wireless radio tools built into the operating system, which only allow for the retrieval of WLAN signal measurements at frequencies of 0.1-1 Hz, varying between different devices. With frequencies this low, it is not possible to calculate any sort of uncertainty value from the spread of readings, as each individual reading must be used for a separate positioning estimate. Furthermore, if the operating system is collecting multiple readings and only outputting an average value, it does not offer any method of retrieving the individual samples or their variance. For this reason, uncertainty of measurements is not considered in the proposed positioning system.

4.4 Bayes Maximum Likelihood

4.4.1 Bayes Maximum Likelihood Classifier

A probabilistic framework was also explored in the search for the ideal algorithm for this application; the one implemented was a Bayes classifier with naïve Bayes conditional probability model and maximum likelihood decision rule. In short, the algorithm chooses the most likely class of a measurement vector based on the product of the probabilities of each of the vector's elements belonging to a particular class. This algorithm assumes all variables (elements of the measurement vector) are independent, each measurement is independent, and all class labels have a uniform prior distribution.

Let the measurement vector be $\mathbf{x}$, where $\mathbf{x} = (x_1, x_2, \dots, x_n)$, thus, the probability of each class label, C_k, given $\mathbf{x}$ is $p(C_k \mid x_1, x_2, \dots, x_n)$. We can decompose the conditional probability using Bayes' Theorem to get

$$p(C_k \mid \mathbf{x}) = \frac{p(C_k)p(\mathbf{x} \mid C_k)}{p(\mathbf{x})} \tag{3}$$

We assume that $p(C_k)$ and $p(\mathbf{x})$ are effectively constant, thus the numerator is equivalent to the joint probability model. Furthermore, using the chain rule for repeated applications of the definition of conditional probability and provided the assumption that each element x_i is conditionally independent of every other element x_j for $j \neq i$ given the category C_k, we can express the probability as

$$p(C_k \mid \mathbf{x}) \propto p(C_k, x_1, x_2, \dots, x_n) = p(C_k)p(x_1|C_k)p(x_2|C_k) \dots p(x_n|C_k) \tag{4}$$

Using Maximum Likelihood as the decision rule the simplified equation for the classifier is

$$\hat{y} = \underset{k \in \{1,2,\ldots,K\}}{\operatorname{argmax}} \prod_{i=1}^{n} p(x_i \mid C_k) \tag{5}$$

where $\hat{y}$ is the class label of the most likely class C_k.

4.4.2 Application to WLAN Fingerprint Matching

To apply the Bayes Maximum Likelihood Classifier to the WLAN Fingerprint Matching problem the set of symbolic locations in the positioning space become the class labels, the measurement vector is WLAN fingerprint where each AP's RSSI value is considered an independent variable, and $p(x_i \mid C_k)$ is calculated based on the histogram of the AP's RSSI values in the reference fingerprint for the symbolic location. The aforementioned probability is equal to the count of the measured RSSI value in the histogram divided by the total number of samples described by the histogram. By default, each possible RSSI value in the histogram has a count of 1 in order to avoid probabilities of zero, which would, in turn, result in the product being equal to zero. Finally, the symbolic location with the maximum product of probabilities of measured RSSI values is chosen as the best estimate of the user's position at the time of the WLAN scan. Furthermore, we can also generalize the equation above for *k* most likely symbolic locations, and use the relevant symbols discussed above to attain

$$L^{k\min} = \underset{p \in \{1,2,\ldots,P\}}{\arg k\max} \prod_{i=1}^{n} p\left(RSSI_i \mid l_p\right), \qquad L^{k\min} \subset L \tag{6}$$

where

$$p\left(RSSI_i^S \mid l_p\right) = H_i^{l_p}\left(RSSI_i^S\right) \tag{7}$$

4.4.3 Histogram Kernelization

Given that the histogram of the AP's RSSI values in the reference fingerprint for the symbolic location (used to calculate the probability of a measured RSSI value) are based on a finite number of samples, there are certain RSSI values that have a non-zero probability but do not exist in the reference sample. Unchanged, these histograms would result in live measurement values being erroneously given a default probability, as well as disproportionately differing probabilities for similar live measurement values (i.e. -75 dBm receiving a 0.2 probability while -76 dBm receives a 0.01 probability). In order for the reference histograms to have accurate counts for every single possible discrete RSSI value, the amount of samples required would be infeasible. Thus, one way to ameliorate this issue is to smooth out the histograms using a kernelization technique; the count for each possible discrete value becomes the mean of the counts of all the values in the local neighbourhood (i.e. the count of -80 dBm is the mean of the counts for -78 dBm to -82 dBm). This significantly improved positioning accuracy and the ±1 dBm (3 discrete values) kernel was heuristically determined to be the best kernel size.

4.5 RSS Differences vs. Absolute RSS

The aforementioned k-NN and ML algorithms are generally run with absolute RSSI measurements as they are provided by the device taking the readings and as they have been described in the sections above. It is possible, however, to redefine the fingerprint

to represent the signal strength differences between the various visible APs and run the aforementioned algorithms. This was posited by Laoudias in (Laoudias, Zeinalipour-Yazti and Panayiotou, 2013) to circumvent the problem of the heterogeneity of the devices (and their WLAN antennas). Since different devices contain different antenna designs and different WLAN chipsets, the signal strength received by different devices in the same location will inevitably vary, however, the signals from all detected AP's are affected equally (i.e. the signals from all detected AP's will be stronger or weaker depending on the capabilities of each specific device). The algorithm computes the differences between the RSSI values for any two AP's in range. By using the differences as opposed to the raw RSSI values, the device's specific bias on the measured RSSI will be eliminated. In addition, this also increases the number of readings fed into the algorithm, from *n* RSSI readings from *n* AP's in range to $n(n-1)/2$ RSSI differences. A detailed description follows.

To begin, as we have described in the prior sections, the fingerprint database contains a set of symbolic locations (i.e. rooms of a building) defined as

$$L = \{l_1, l_2, l_3, \dots, l_k\} \tag{8}$$

and for each l there is a set of RSSI readings, one for every AP within range in that particular location, defined as

$$RSSI^l = \{RSSI_1^l, RSSI_2^l, RSSI_3^l, \dots, RSSI_m^l\} \tag{9}$$

where: l denotes a symbolic location

$RSSI_i^l$ denotes a pre-recorded signal strength in the database from the *i*-th AP in dBm, at location l

m denotes the total number of AP's within range at location l

When the subject's device is being localized, it takes a set of RSSI readings of all AP's in range at that location, defined as

$$RSSI^s = \{RSSI_1^s, RSSI_2^s, RSSI_3^s, \dots, RSSI_n^s\} \quad (10)$$

where: s denotes the subject's location

$RSSI_i^s$ denotes the signal strength observed by the subject from the *i*-th AP in dBm, at location s

n denotes the total number of AP's within range at location s

Then, differences in signal strength between every possible pairing of AP's are calculated, or in other words, two new sets are created with the differences of every possible unique pairing of RSSI readings in each of the two RSSI sets above. These two new sets are defined as

$$\Delta RSSI^l = \{\Delta RSSI_1^l, \Delta RSSI_2^l, \Delta RSSI_3^l, \dots, \Delta RSSI_{m(m-1)/2}^l\} \quad (11)$$

$$\Delta RSSI^s = \{\Delta RSSI_1^s, \Delta RSSI_2^s, \Delta RSSI_3^s, \dots, \Delta RSSI_{n(n-1)/2}^s\} \quad (12)$$

where: l, s, m, n as previously defined

$\Delta RSSI_j^l$ denotes the difference between the pre-recorded signal strengths of the *j*-th possible pairing of AP's in the database, in dBm, at location l

$\Delta RSSI_j^s$ denotes the difference between the signal strengths of the *j*-th possible pairing of AP's observed by the subject, in dBm, at location s

At this point, the input data for the localization algorithm is ready, and we proceed to compute the classification metric, in this case, a slightly modified Manhattan distance.

$$d_{Man*} = \frac{\sum_{j=1}^{w} \left| \Delta RSSI_j^l - \Delta RSSI_j^s \right|}{w} \tag{13}$$

$$l_{1_{Man*}} = \arg \underset{k=1;\, l_1 \in L}{\text{kmin}} (d_{Man*}) \tag{14}$$

Where: w denotes the quantity of AP pairings seen in both $\Delta RSSI^l$ and $\Delta RSSI^s$

$l_{1_{Man*}}$ denotes the symbolic location most closely representing the subject's real location, based on our modified Manhattan distance metric

In cases where a particular pairing of AP's is only present in one of the sets of RSSI differences, that signal strength difference is ignored in the computation of the Manhattan distance metric. The final equation, then, determines the symbolic location ($l_{1_{Man*}}$) that most closely represents the subject's real location by choosing that which corresponds to the smallest normalized Manhattan distance (d_{Man*}).

The above algorithm was implemented and a number of data sets were run through it to determine how effective it was for minimizing the effect of device heterogeneity on positioning accuracy. Our testing initially found that the signal strength difference definition of the fingerprint was fairly resilient to device heterogeneity, and positioning performance was fairly consistent regardless of which device was used to create the reference dataset and which was used for positioning. After initially making the assumption that device heterogeneity had a significant effect when using traditional

absolute signal strength fingerprints, based on Laoudias' findings (Laoudias, Zeinalipour-Yazti and Panayiotou, 2013), we decided to perform our own testing and found that the effect was actually not significant. The algorithm was also tested in real-time in our local testbed; we found no discernible difference between the performances of the two different fingerprint definitions. It is possible that device heterogeneity was once a significant problem, but with the devices tested in our experiments (smartphones and tablets released as early as 2010) its effect was smaller than the effect of noise in the signal measurements. Device heterogeneity could be a problem when readings are taken by different types of devices (i.e. a combination of mobile devices, laptops and/or IoT devices), however, that is beyond the scope of our research.

4.6 Path Evaluation and Retroactive Adjustment

The possible paths of the subject over the last five epochs are assessed based on three criteria; these can be briefly described as *k*-NN score/proximity, and short-term and long-term movement regularity.

4.6.1 *k*-NN Proximity

Although the top k locations from k-NN are attained, this does not mean that they are equally valued. Thus, their proximity measures are also retained so that they may be considered in the k-NN proximity criterion of the path assessment stage. This criterion can be scored in three ways; the first is relative score by comparing each of the k

locations (l_i) to the *nearest* location (l_1), the second is an absolute score of the proximity, and the third is another more complex relative score.

$$kNNP(l_i^t) = \frac{d_{SS}(l_1)}{d_{SS}(l_i)} \tag{15}$$

$$kNNP(l_i^t) = \frac{1}{d_{SS}(l_i)} \tag{16}$$

$$kNNP(l_i^t) = \frac{d_{SS}(l_1)}{1 + d_{SS}(l_i) - d_{SS}(l_1)} \tag{17}$$

Heuristically, it was determined that the first way of scoring this criterion was the most effective.

4.6.2 Short-Term Movement Regularity

The second criterion scored in the path evaluation is the short-term movement regularity. This criterion essentially represents how likely it is that the subject moved between two locations in the time between two epochs. This is scored by comparing the physical distance between the two locations with the distance that can be walked at an average walking speed during the time between the two location estimates (i.e. time between consecutive k-NN estimates). The score is calculated with the equation below, where Δt is elapsed time in seconds, wS is an average walking speed in m/s, and $d_{PS}(l_i,l_j)$ is the Euclidean distance between the symbolic locations with t being the epoch.

$$SMR\left(l_i^t, l_j^{t-1}\right) = \frac{wS \cdot \Delta t}{d_{PS}\left(l_i^t, l_j^{t-1}\right)} \tag{18}$$

Thus, the score is at a maximum when d_{PS} is less than (i.e. subject is stationary [dummy 1 cm value inserted for d_{PS} in this situation to avoid division by 0] or moving slowly)

or equal to the average distance walked in the elapsed time between the consecutive epochs. On the other hand, the score decreases as the physical distance between the locations increases (once it is above the expected walked distance).

4.6.3 Long-Term Movement Regularity

The third and final criterion scored in the path assessment is long-term movement regularity. Since the path assessment is performed for a recent portion of the subject's estimated path, namely the 5 last epochs, this criterion represents how well the subject's movement matches its net displacement over the 5 epochs. This criterion is scored by summing up the total distance travelled by the subject as they move (or stay stationary) from epoch to epoch, then comparing it to the net displacement of the subject over the 5 most recent epochs. The score is calculated with the equation below, where $d_{PS}(l_i,l_j)$ is the Euclidean distance between the symbolic locations and t is the epoch.

$$LMR = \frac{d_{PS}(l_i^t, l^{t-5})}{\sum_{e=t}^{t-4} d_{PS}\left(l_i^e, l_j^{e-1}\right)} \tag{19}$$

This score is at a maximum for this criterion when the total distance travelled is equal to the net displacement; conversely, it is at a minimum when there is a lot of movement but no net displacement. This essentially discourages a path with stuttering (i.e. one step forward, one step back, two steps forward, while subject only walked two steps forward), and tends to smooth the subject's estimated path. To make the comparison less cumbersome, the net displacement considered in the equation is direction-less; furthermore, the code implemented also automatically inserts a value of 1 for the score if the total movement (and therefore also the net displacement) is 0.

4.6.4 Final Score

The aforementioned three scores all have different ranges and they do not follow the same curve. Thus, sigmoid functions are employed to reconcile the distributions of each of the movement regularity scores. Furthermore, the sigmoid functions allow for each of the scores to have a more precise and intended effect on the final overall score for the path segment being considered. The following are the sigmoid functions used for each criterion score and their respective graphs.

$$SMR_S\left(l_i^t, l_j^{t-1}\right) = \frac{3(SMR - 0.5)}{2\sqrt{9(SMR - 0.5)^2 + 1}} + 0.5 \tag{20}$$

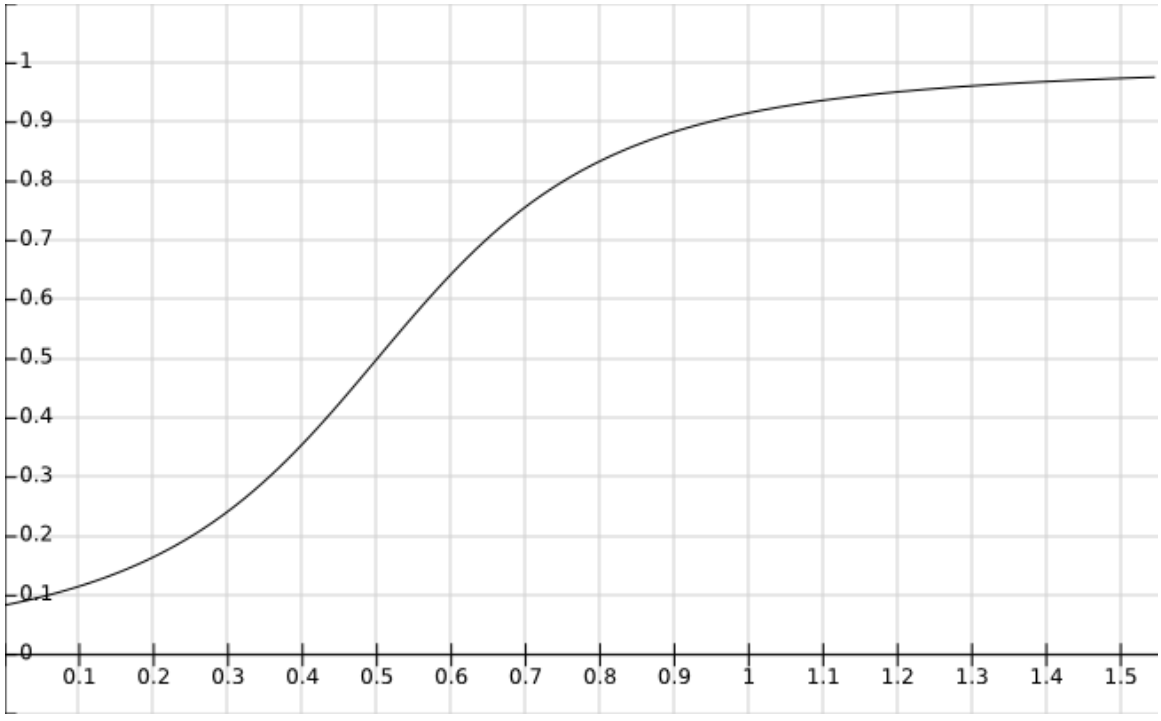

Figure 19. SMR Sigmoid Function

As can be seen in the graph, the SMR sigmoid function greatly favours situations where the subjects is moving at a normal pace (SMR is approximately 1) or staying stationary (SMR is significantly greater than 1), with the latter being slightly more favoured. As the movement increases, the score drops to discourage very fast (and likely incorrect) movement. The function outputs about half a maximum score when the subject has moved twice as far as expected and a quarter of the maximum score when the movement is three times what's expected.

On the other hand, the LMR sigmoid function does not need to deal with input LMR values greater than 1, and additionally, the score drops much faster than the SMR sigmoid function. The scoring function is intentionally more punitive for the LMR because the distances being compared in the LMR function are larger than those in the SMR function, thus, the LMR sigmoid scoring function needs to penalize even small relative differences.

$$LMR_S = \frac{15(LMR - 0.75)}{8\sqrt{9(LMR - 0.75)^2 + 1}} + 0.6 \tag{21}$$

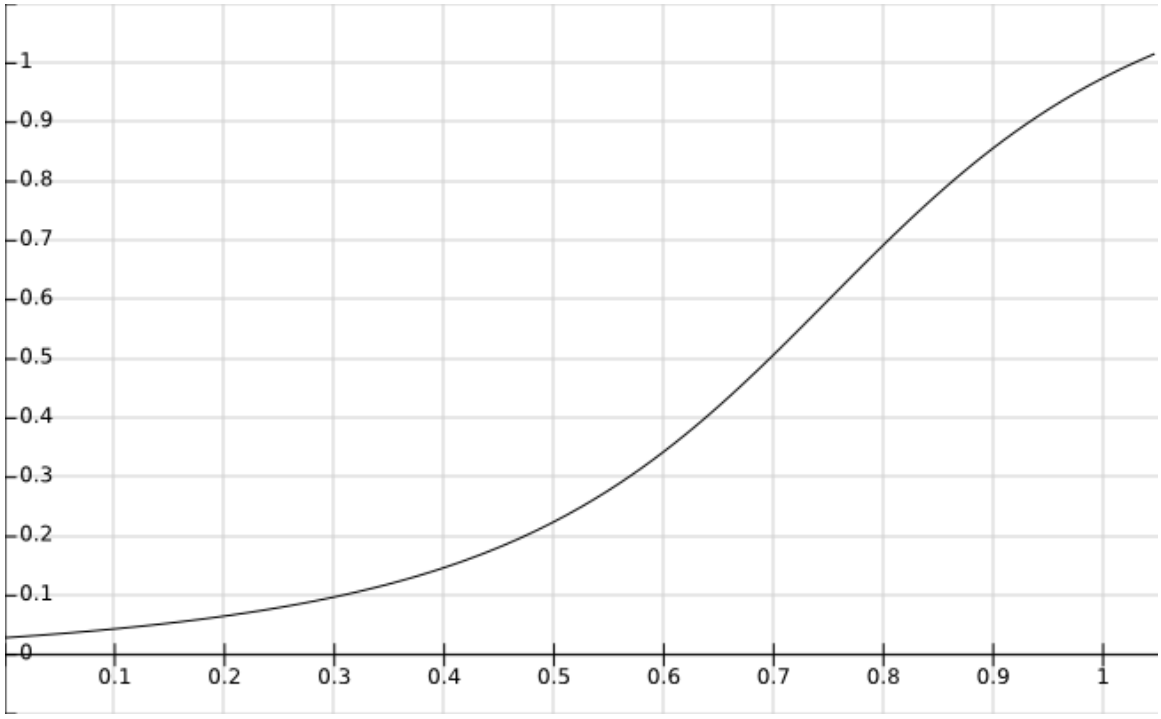

Figure 20. LMR Sigmoid Function

Once all the criteria are scored for all of the position estimates in a particular potential path, the final path score for that segment (5 epochs) is given by the following function.

$$\begin{aligned} PathScore = {} & kNNP(l_i^{t-4}) \cdot SMR_S\left(l_i^{t-4}, l_j^{t-5}\right) \cdot kNNP(l_i^{t-3}) \cdot SMR_S\left(l_i^{t-3}, l_j^{t-4}\right) \\ & \cdot kNNP(l_i^{t-2}) \cdot SMR_S\left(l_i^{t-2}, l_j^{t-3}\right) \cdot kNNP(l_i^{t-1}) \cdot SMR_S\left(l_i^{t-1}, l_j^{t-2}\right) \\ & \cdot kNNP(l_i^{t}) \cdot SMR_S\left(l_i^{t}, l_j^{t-1}\right) \\ & \cdot (LMR_S)^3 \end{aligned} \quad (22)$$

As mentioned earlier, the *k* chosen locations from each of the previous five epochs are retained. Thus, when we say all potential paths, we mean all possible ways to traverse

the following matrix from left to right starting from the final chosen location at epoch *t-5* (l_1^{t-5} in the following example).

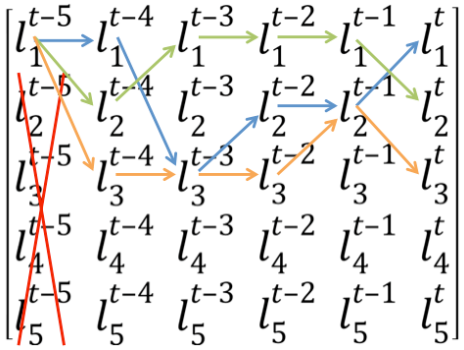

Figure 21. Hypothetical Matrix Representation of the Path Evaluation at t = 5

Once every possible path is scored, the path with the highest score is chosen as the best path and its final location (at *t*) is presented to the subject as the subject's current position. Say, for example, that the following was the highest scoring path:

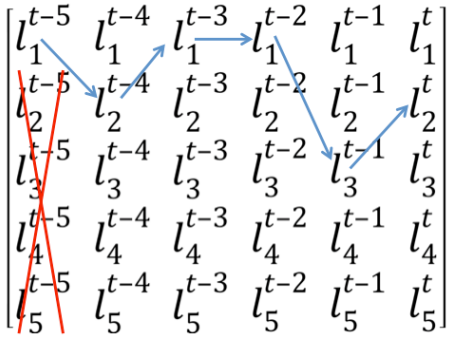

Figure 22. Hypothetical Matrix Representation of the Highest Scoring Path at t = 5

At the next epoch (new set of *k* locations provided by *k*-NN), shown in the matrix below, the starting location on the left side of the matrix is l_2^{t-5}. This is the same

location as l_2^{t-4} from the previous matrix, and was the only one retained from that epoch after being in the best path the last time the recent paths were assessed.

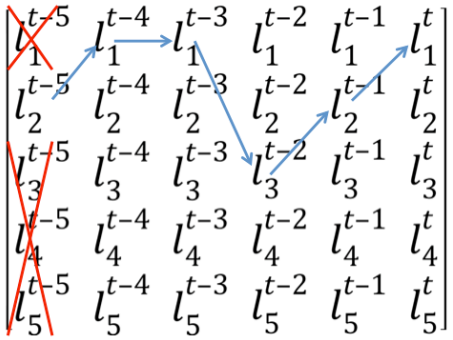

Figure 23. Hypothetical Matrix Representation of a Path at t = 6

Another way to describe the path assessment process is to think of a tree created by all the locations suggested by k-NN over the number of epochs considered. The following simplified example considers *k*-NN (where *k*=3) over 3 epochs. All the possible paths to go from root to leaf, in the tree below, represent all possible recent paths the user could have traversed. These paths are scored and then ranked and the discrete locations of the highest ranked path, from root to leaf, are chosen as the subject's most likely path over the course of the epochs considered.

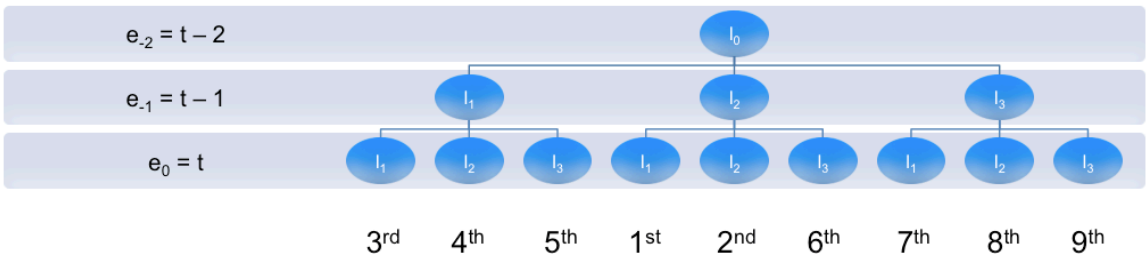

Figure 24. Hypothetical Tree Representation of the Path Evaluation at t = 3

5. Implementation

This chapter focuses on the implementation of the positioning system explored and discussed in this research, as well as, the implementation of tools used for data collection, and finally, various performance improvements built into the implementation to ameliorate real-time use and analysis.

5.1 WLAN RSS Collection Android Application

In order to collect RSS samples of APs (fingerprints) in our positioning environment, an Android application was developed as a part of this research. Given that the actual objects being localized are mobile devices, it follows logically that the environment should also be sampled with mobile devices. This application has four main components: the first component polls the Android system for a WLAN scan and receives the scan results; the second component filters the scan results to attain only samples from the official university APs; the third component creates and maintains a small database of all the attained information in the current collection session; and finally, the fourth component, upon completion of the collection session, formats the attained information and writes it to a text file. In addition to these internal components, the user interface of the application is another important aspect that will be described herein.

The first component attains and uses an instance of the WiFiManager Android class to poll the WLAN radio for a scan. A BroadcastReceiver class is implemented in the WiFiReceiver nested class to receive the scan results provided by the WiFiManager. The scan results contain various identifying information about the APs along with the RSSI; for each AP, these include the BSSID (a unique identifier for the AP, also known as a MAC address), the SSID (the name of the network being broadcasted by the AP) and a timestamp of when the AP's signal was sensed by the radio. Scan results also contain various other pieces of information about the frequency, capabilities, and operator of the AP, however, that information is not used. The RSSI is a measure of the strength of the AP signal received by the WLAN radio. On Android, RSSI is provided in integer dBm and values can range from 0 (strongest) to -100 (weakest), although values generally observed only range from -30 to -90. The BSSID is a series of 6 8-bit hexadecimals separated by colons, and as aforementioned, is a unique hardware address for the AP. The SSID is the name of the network broadcasted by the AP. This field is empty if the AP does not broadcast its network name, and in this case, the WiFiReceiver class assigns a dummy network name to the AP. Finally, the timestamp in the scan results is provided as an integer number of microseconds (long type to avoid overflow) since the device was turned on. The provided timestamp value is converted to the number milliseconds since January 1st 1970, in order to have an absolute time and be able to compare between various recording sessions; the WiFiReceiver class also does this conversion.

The second component is relatively straightforward. The university WLAN infrastructure APs are easily distinguishable from other personal APs due to the fact that they each broadcast one of a few specific different network names (SSIDs). As mentioned above, the SSID broadcasted by each AP is available from the WLAN scan results, thus, this filtering of irrelevant APs is simply a matter of checking each AP's broadcasted SSID against a whitelist of university infrastructure SSIDs. All data collected from AP's that do not broadcast a whitelisted SSID is simply discarded at this step. The application, however, also offers an option to toggle this component on or off, allowing the user to retain all collected data and skip this filtering step.

Once the set of scan results has passed through the second filtering component, the third component organizes the retained information into a data structure consisting of an ArrayList of AccessPoint objects. The AccessPoint class is a custom nested class created for the purpose of containing all the information attained about a particular AP into one neat object. Each instance of the class contains fields for the BSSID and SSID, as well as two equal length ArrayLists, one containing the various RSSI readings captured for that AP and the second containing the respective timestamps of each readings. Each AccessPoint object can be created with or without a set of RSSI readings, and has its own methods for the addition or removal of RSSI readings. The data structure is continuously populated as scan results come through the first two components.

Once the collection session is completed, the fourth component reads all of the attained information in the data structure and proceeds to format and then write all the information into a text file. The data structure is read one AccessPoint object at a time, and the information for that AP is sorted in the following format then written to a file. The next AP's information is then written below the preceding AP's information.

AP_1 (BSSID, SSID, number of RSSI readings)
    $RSSI_1$:timestamp
    $RSSI_2$:timestamp
    …

AP_2 (BSSID, SSID, number of RSSI readings)
    $RSSI_1$:timestamp
    $RSSI_2$:timestamp
    …

Figure 25. Format of Collected RSS Data

The final important aspect of the application is the user interface. The largest portion of the user interface shows a constantly updating list of visible APs, showing their SSID, RSSI, and time of the reading. In order to name the information gathered in a particular collection session there is a text box at the top of the interface; the button to the right of it instructs the aforementioned fourth component to write the readings collected thus far to a .txt file with the provided name. The row of checkboxes below presents the user with options to timestamp the collection, filter out non-university infrastructure APs, and to compute and record only the mean of the RSSI readings obtained for each AP (this function, if toggled on, is performed in the fourth component). Finally, the overflow menu allows the user to pause/resume the scanning

for APs, restart the collection session, and manually refresh if automatic scans stop occurring. Screenshots of the application are provided below.

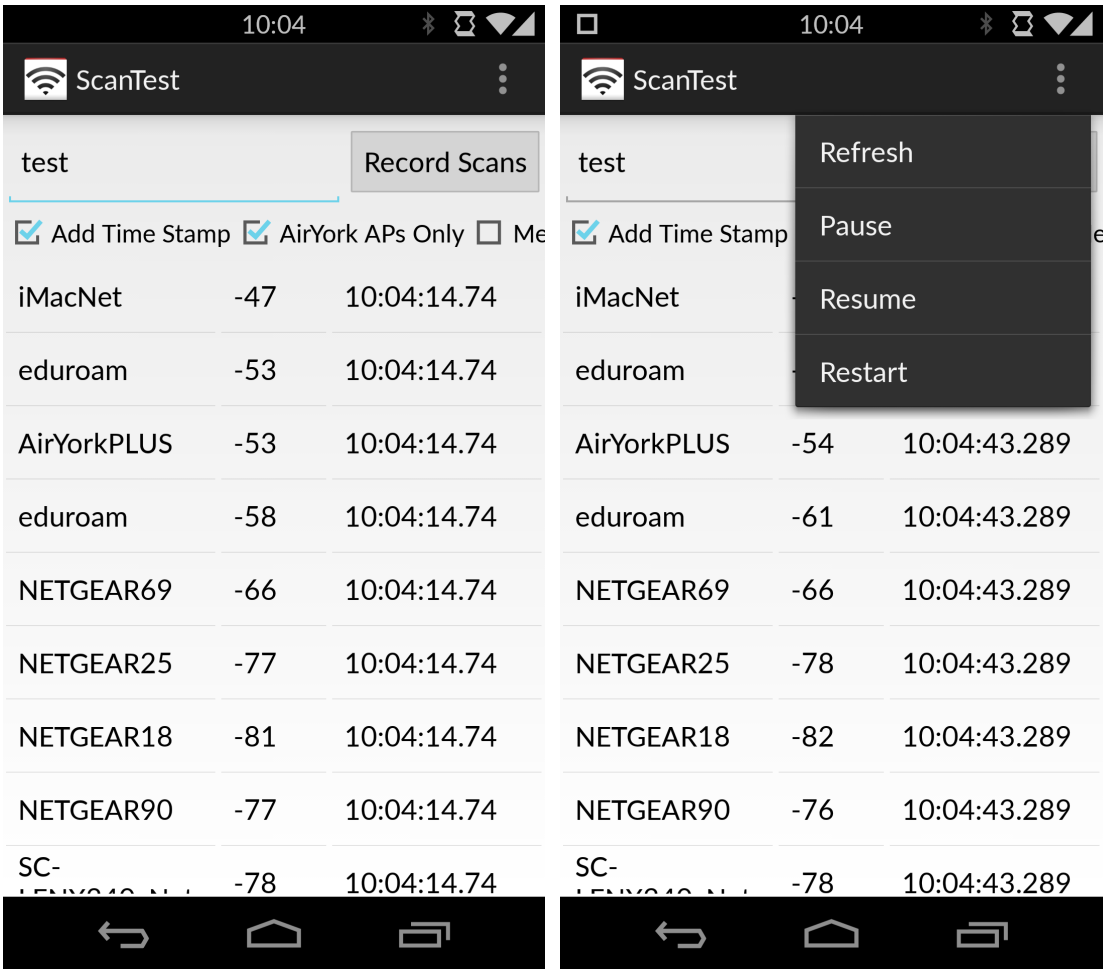

Figure 26. Screenshots of the Data Collection Application

5.2 Live Self-Positioning Android Application

The main application for this research was the live self-positioning application running on Android, named MobilePosition. This was the application used to test the real-time performance of the algorithm. Given that it also requires the RSSI readings for all the various APs within range, this application borrows some of the main components from

the ScanTest (WLAN RSS Collection) application. The MobilePosition application essentially has two main functions and two activities to perform each of them, respectively. The initial function is to build a reference fingerprint database to be later used for positioning. The second and primary function is to actually position/localize the user within the building floor based on the real-time WLAN RSSI readings being captured by the mobile device. As with the ScanTest application, each of the two activities have their own respective user interface. The initial setup activity offers the user a number of options to choose regarding the positioning functionality of the primary activity. On the other hand, the positioning activity offers the user a map of the building's floor plan indicating the user's estimated location.

The two main functions of the setup activity are the reading of all of the reference fingerprint text files, and the creation and population of a reference fingerprint database data structure. A single nested class (RSSRecordReader) within the activity accomplishes both of these functions; this was necessary in order to achieve the concurrent/parallelized implementation of the functions, and will be further explained in the next subsection. To begin, the text files containing the reference fingerprints are organized in such a way that each text file contains all of the reference fingerprints for a particular symbolic location. Furthermore, the text files are named in such a way that they are ordered by the symbolic location number, and each floor of each building has its own folder of reference fingerprint text files. Thus, once the setup activity receives the instructions from the user, it sets the folder corresponding to the user's choice of building and floor, and proceeds to create the empty shell of the reference fingerprint

database data structure. This time, the data structure is an ArrayList of LocationCell objects, LocationCell being a custom class created to hold all of the reference fingerprint information for a particular symbolic location. Instances of RSSRecordReader are then run to read the aforementioned text files and insert the fingerprint information into the reference fingerprint database.

Once the reading of the reference fingerprint text files is completed, the fully populated reference fingerprint database is stored as a static variable in a custom Database class. The aforementioned class also keeps static variables for several other algorithm and performance parameters; these parameters are mostly chosen by the user in the set up screen and are also set at this time. Finally, once all of the variables in the Database class are populated/set, the PositionActivity is started to begin positioning the user.

Upon start up, the PositionActivity grabs all of the parameters set in the Database class and initializes all of its variables and components. It employs a WiFiReceiver nested class (very similar to the one in the ScanTest application) which, upon receiving new WLAN scan results, creates a Fingerprint object with the data and begins the process of computing an estimate for the user's location. The first step in the positioning process is running the subject's newly acquired fingerprint through the nearest neighbours algorithm to determine the *k*-nearest symbolic locations to the subject in signal space. The indices corresponding to these are then added to a matrix of nearest neighbours that holds the determined *k*-nearest neighbours from the last five epochs. During the first five epochs of the current positioning session, the user is simply

presented with the location corresponding to *the* nearest neighbour; from the sixth epoch onwards, the path assessment module is used to determine the most likely path based on the moving regularity criteria as described in section 4.4 and the user is presented with the location corresponding to the end point of the most likely path.

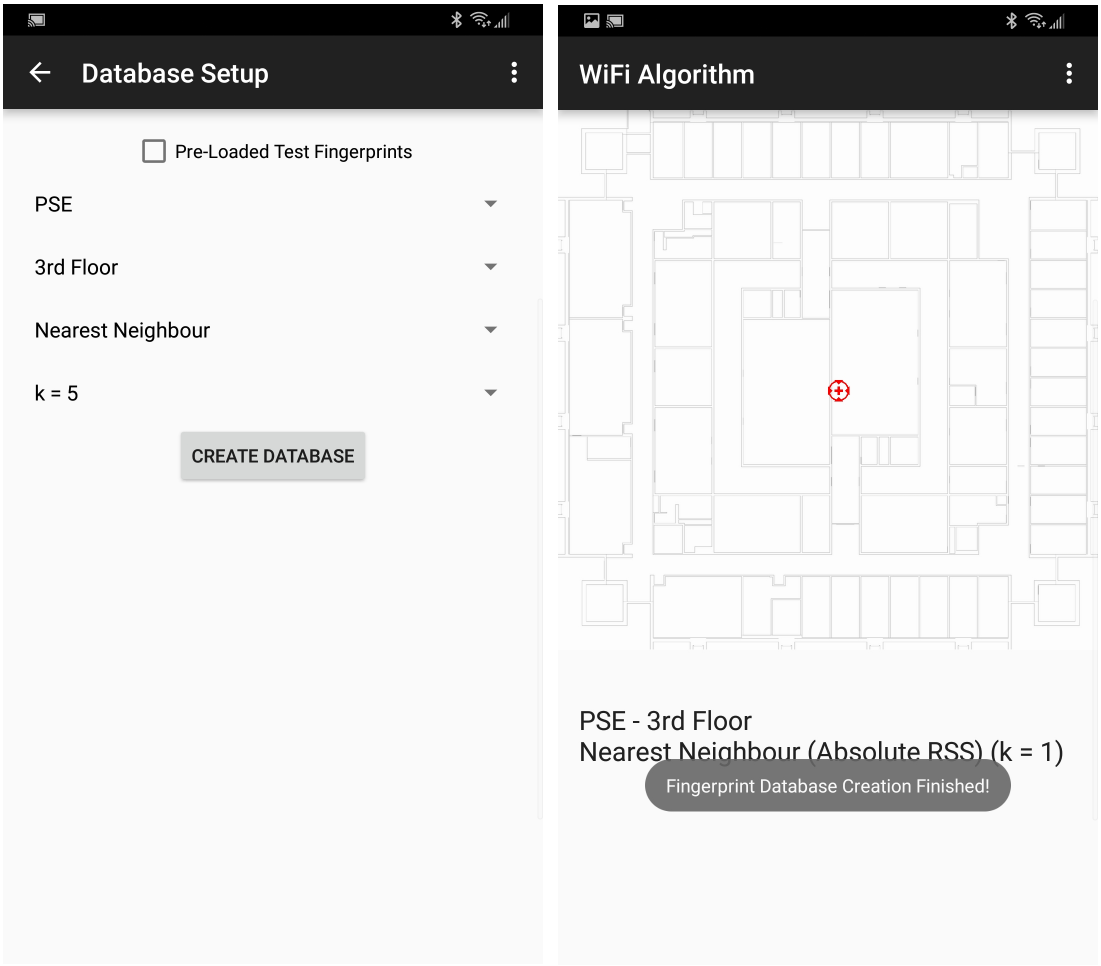

Figure 27. Screenshots of the MobilePosition Application

Every time the user's position is estimated anew, it is shown to the user by a red crosshair placed on the estimated location on the building's floor plan that's displayed

on the screen. In addition to this, the position's index and timestamp of estimation is presented below the floor map. Finally, the application also keeps a log of the estimated positions and various other data that was used to assess performance and debugging purposes. All of the logged data is written to a file upon termination of the positioning session.

5.3 Concurrent Implementation

The live positioning application was later improved to take advantage of the multi-core computing available on most modern day smartphones. Two parts of the application were modified to process the computations in a parallelized manner. Firstly, the reading of the reference fingerprint text files is split amongst multiple threads so that they may be read concurrently. Various instances of RSSRecordReader are then run on parallel threads, each reading a portion of the aforementioned text files one by one and inserting the fingerprint information into the reference fingerprint database. Secondly, the computation of the distance matric for the *k*-NN algorithm is also parallelized. Given that the distance needs to be computed for each symbolic location, this can be split into multiple threads with each thread computing a portion of the distances each time there are new scan results from the WiFiManager.

5.4 Java Version of Positioning Application

The live positioning application on Android was also written as a Java application that could be run on a computer. The two main reasons for this undertaking were to have

an implementation where experimental changes to the algorithm could be made and tested more easily, and where pre-recorded test readings can be easily run through the algorithm to evaluate performance. As a result, the Java version of the positioning algorithm is not capable of running in real time, but instead can only process pre-recorded RSS readings. However, it was very helpful for clearly seeing the effect of minor and major algorithm changes on positioning accuracy and performance; large data sets of pre-recorded readings could be fed into the algorithm, run very quickly, and thoroughly logged to gather detailed insight into the effect of the algorithmic changes being tested. Furthermore, the fact that the Android MobilePosition application's algorithm was also written in Java allowed for very quick and easy porting of algorithmic improvements.

6. Results and Discussion

This chapter reviews the results of all of the experimentation performed as part of this research. It describes the methods used to assess the positioning performance of the system in a quantifiable manner and details positioning performance improvements offered by the novel portions of the positioning system.

6.1 Introduction

The WLAN Fingerprint Matching and Path Evaluation and Retroactive Adjustment positioning system was deployed on multiple floors of multiple buildings on York University's Keele Campus, and the developed Mobile Position application could be installed on any Android device to qualitatively assess the positioning performance. The quantitative testing of the positioning algorithms was performed using two distinct data sets in order to quantify its positioning accuracy. Both data sets input into the positioning system for testing consist of real RSS measurements gathered on-site using the same WLAN RSS data collection application. The differences between the two collection methods and creation of the testing data sets are described in the following sections. In the charts detailing the results:

1. ***k*-NN** refers to the first implementation was our improved WLAN Fingerprint Matching algorithm.
2. ***k*-NN + PERA** refers to the second implementation including the Path Evaluation and Retroactive Adjustment module in addition to the WLAN Fingerprint Matching in the first implementation.

6.2 Deployment

The methods discussed in Chapter 3 were used to deploy the positioning system on six floors across three buildings. Once these methods were finalized, total deployment time including data collection, processing and inclusion in the system, to enable positioning on a floor was about one working day. Planning out symbolic locations in a floor would require about 2 hours; collecting data on those points would require about 2-4 hours, or approximately 1 hour of data collection for every 200 metres squared of navigable area. Finally, the symbolic locations and their respective reference WLAN signal fingerprints are added to the positioning application, requiring 1-2 hours, before positioning is enabled on the floor. A turnaround time of one day per floor or one week for an average university campus building is comparable to commercial indoor positioning solutions (*Senion IPS*, 2016; *HERE Indoor Positioning*, 2016).

In addition to the deployment time, it should be noted that the data storage and application implementation was also quite space efficient and responsive. The entire application and data for the three buildings where positioning was enabled was under 5 MB in size. Position estimates were also provided to the user instantly after every WLAN RSS measurement provided by the device, resulting in a very responsive experience; this was achieved through a client-side implementation and the use of parallel computing utilizing the device's multi-core processor.

6.3 Real-Time Positioning

Much of the testing of the positioning system was performed in person, by using the MobilePosition application within the various areas where positioning was enabled across campus. Using the application for real-time positioning required choosing your building and floor on the Database Setup page of the application; within seconds the positioning algorithm was activated and the user's current estimated location was displayed on a map of the chosen floor. As showcased in Figures 28 and 29, positioning accuracy was very good the majority of the time. Position estimates during real-time use of the application were very rarely noticeably incorrect; that is to say that the estimated location, when it was incorrect, was close enough that the user usually would not notice that there was any error. On occasion, when a noticeable egregious error was made by the positioning system, it would often recover to an accurate position on the following live fingerprint reported to the algorithm by the device. The PERA module further reduced the occurrence of egregious positioning errors, making the aforementioned situations increasingly rare. The only area where some instability of position estimates was noticed was in large open areas in the BRG building; this is expected, to some extent, due to the homogeneity of WLAN RSS in open areas, as there are no physical obstacles to the radiation distinctively attenuating the signals of the various APs visible to the user's mobile device. Overall, the positioning performance of the application in real-time use was very good and the experience was comparable to GNSS-based positioning outdoors.

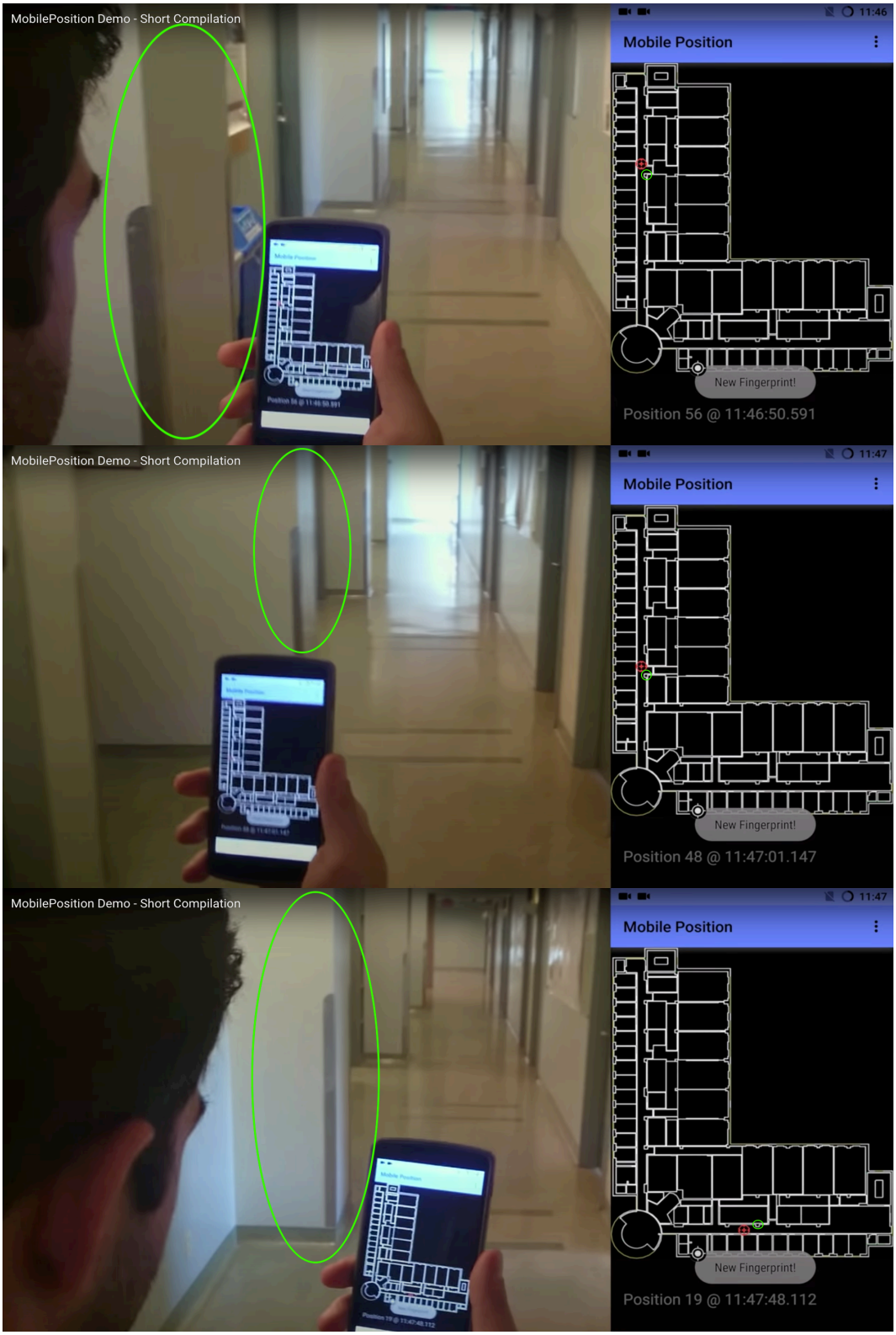

Figure 28. Real-Time Positioning in the Chemistry Building

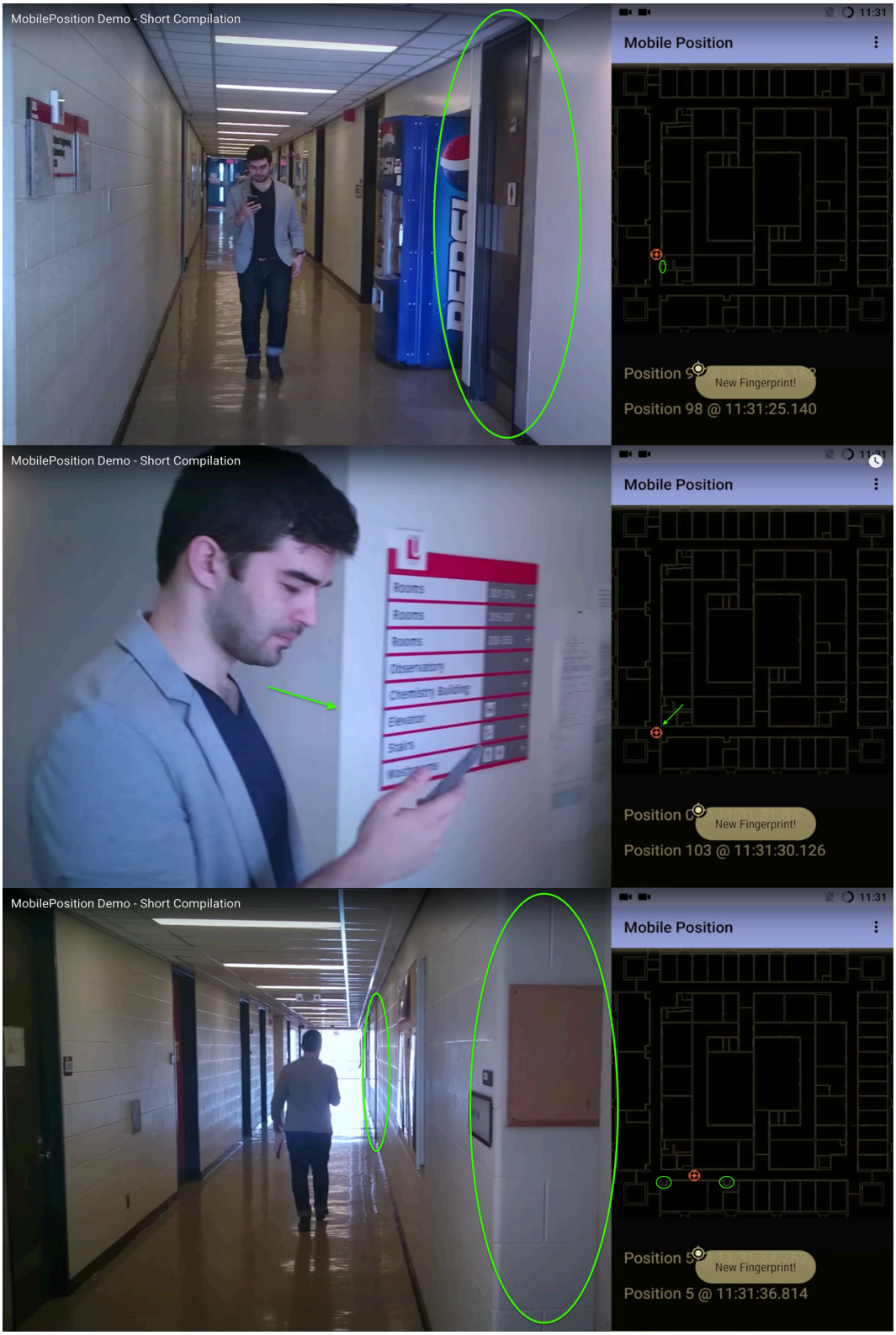

Figure 29. Real-Time Positioning in the Petrie Science and Engineering Building

6.4 Synthetic Path RSS Measurements

The first set of RSS measurements gathered for testing the positioning system, were acquired in the same fashion as the reference RSS measurements. The collection was done by standing stationary for a period, collecting multiple sets of RSS measurements, saving the results, then moving to the next spot to gather another set of measurements. In order to simulate positioning performance for a user walking along a path, the collected test data was rearranged such that a series of RSS measurements collected at consecutive adjacent locations could be fed into the positioning system. These rearranged testing data are referred to as synthetic paths; they were created by taking one set of RSS measurements from each location (where such measurements were collected) and arranging them so that they are fed into the order they would be seen by the user's device if they were traversing a path across the positioning area. Since many sets of RSS measurements were collected, this allowed for a multitude of synthetic paths with entirely unique measurements.

To be clear, although dubbed synthetic paths in this work, these testing datasets are entirely comprised of real collected WLAN RSS measurements; the synthetic adjective simply means the measurements have been temporally rearranged to simulate a user traversing a path along the hallways of one of the buildings where the positioning system is deployed. This method of testing was necessary due to the difficulty of acquiring testing data with frequent and reliable location ground truth while the user is

moving. It was extremely difficult, if not impossible, to associate individual WLAN RSS measurements with their respective location ground truth (i.e. the physical location of the user when the measurement was reported by the device) if the user is walking a path along the hallways of a building. Since individual WLAN RSS measurements are independent of each other, by arranging measurements with accurate ground truth locations as a series of consecutive measurements taken along a path, it becomes possible to accurately test the performance of the positioning system for a moving user.

Table 4. Summary of Positioning Performances Evaluated with *k*-NN and *k*-NN + PERA in three buildings at York University

	Positioning Error (Metres)					
Building - Floor	PSE-L3		CB-L4		BRG-L2	
Algorithm / % of Est. Pos.	*k*-NN	*k*-NN + *PERA*	*k*-NN	*k*-NN + *PERA*	*k*-NN	*k*-NN + *PERA*
Mean	1.3	0.8	1.3	0.8	2.2	1.4
80%	< 1.5	< 1.5	< 1.5	< 1.5	< 3.7	< 1.8
90%	< 3.0	< 1.5	< 3.0	< 1.5	< 5.5	< 3.7
95%	< 4.6	< 1.5	< 3.4	< 3.0	< 7.1	< 3.7
99%	< 6.1	< 3.0	< 6.0	< 3.0	< 9.2	< 8.6
100%	< 10.6	< 6.1	< 13.5	< 9.0	< 14.6	< 11.0

By running these synthetic paths through the positioning system, we collected thousands of positioning data points from the two aforementioned versions of our

implemented positioning system. Table 4 shows a summary of positioning performance evaluated with two different algorithms of *k*-NN and *k*-NN + PERA in three buildings including Petrie Science and Engineering Building (PSE), Chemistry Building (CB) and the Bergeron Centre for Engineering Excellence (BRG) at York University's Keele Campus. Furthermore, Figures 30, 31 and 32 show comparisons of cumulative positioning error between the two algorithms. We observed that *k*-NN + PERA results in significantly improved consistency of positioning accuracy compared to *k*-NN. This level of positioning accuracy can certainly be useful for location-aware applications in the indoor space.

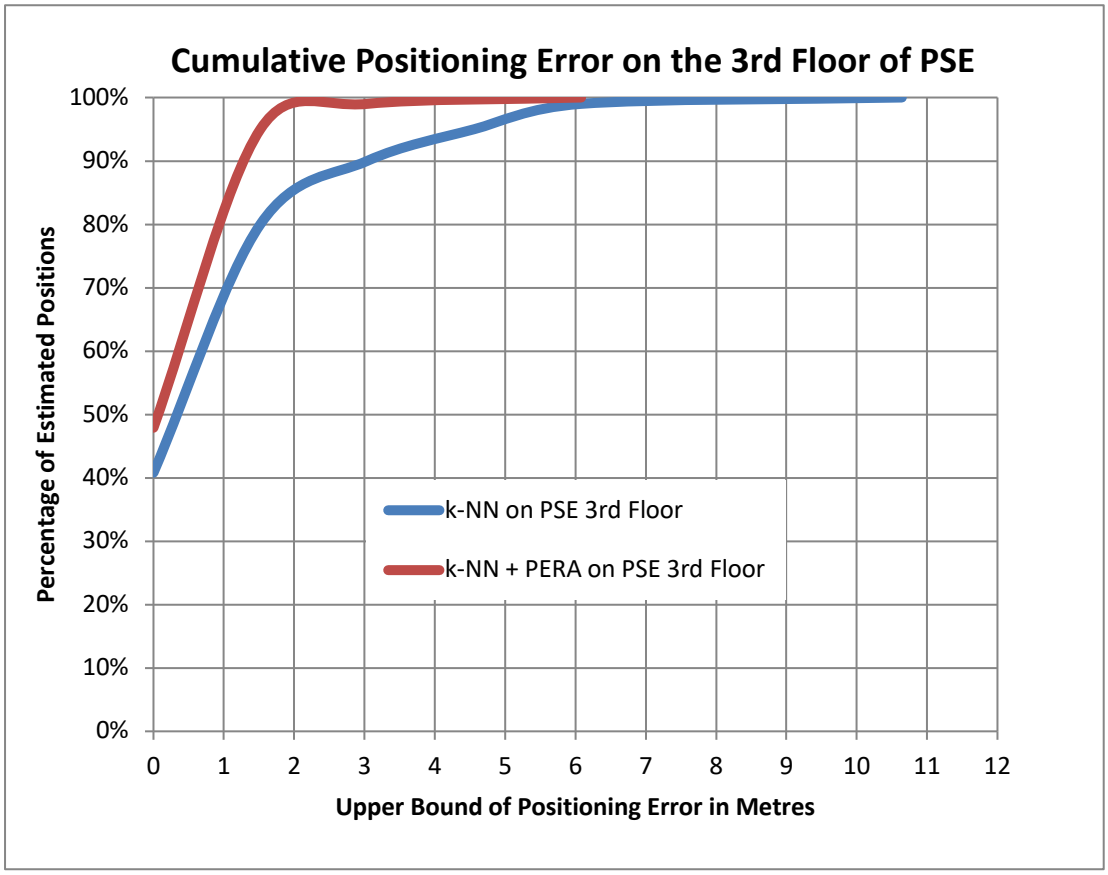

Figure 30. Cumulative Positioning Error on the 3rd Floor of PSE

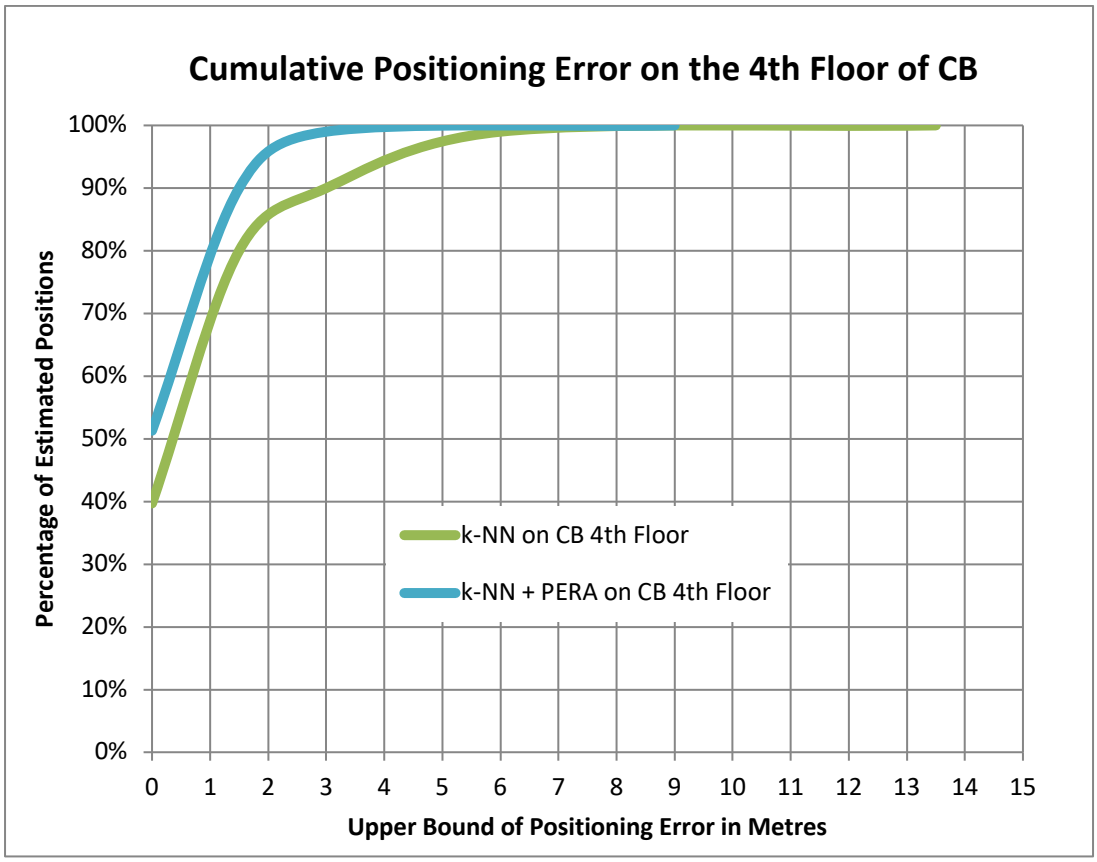

Figure 31. Cumulative Positioning Error on the 4th Floor of CB

We see the greatest accuracy within the PSE building, mean positioning error for *k*-NN was 1.3 m, while mean error for *k*-NN + PERA was 0.8 m; positioning error was less than 1.5 metres 90% of the time for both *k*-NN and *k*-NN + PERA. Meanwhile, at 99% of the estimates, upper bound of positioning error was 6.1 metres for *k*-NN, while for *k*-NN + PERA positioning error was less than 3.0 metres 99% of the time. We believe this is due to the fact that it is the oldest of the three buildings and has thick concrete walls throughout; this results in greater heterogeneity of RSS from the various access points throughout the positioning area, therefore resulting in better performance

primarily by the *k*-NN algorithm. The main influence of the PERA module is preventing some of the rare egregious errors of the *k*-NN algorithm.

We see similar accuracy within the Chemistry building, mean positioning error for *k*-NN was 1.3 metres, while mean error for *k*-NN + PERA was 0.8 metres; positioning error was less than 1.5 metres 80% of the time for both *k*-NN and *k*-NN + PERA. Meanwhile, at 99% of the estimates, upper bound of positioning error was 6.0 metres for *k*-NN, while for *k*-NN + PERA positioning error was less than 3.0 metres 99% of the time. Chemistry building is much newer than the PSE building, but has similar sized hallways and likely relatively similar structural materials and access point distribution. Similarly, we see the influence of the PERA module in preventing some of the more egregious positioning errors of the *k*-NN algorithm.

In BRG mean positioning error for *k*-NN was 2.2 metres, while mean error for *k*-NN + PERA was 1.4 metres; we observed lower accuracy compared to the other two buildings, with positioning error less than 3.7 metres 80% of the time for *k*-NN and positioning error less than 1.8 metres 80% of the time for *k*-NN + PERA. Meanwhile, at 95% of the estimates, upper bound of positioning error was 7.1 metres for *k*-NN, while for *k*-NN + PERA positioning error was less than 3.7 metres 95% of the time. The most likely reason for the poorer performance in this building is the fact that it is a brand new building with large open hallways and a general open concept style. This results in more homogeneity in the RSS of the access points in large swaths of the positioning area; in open areas where the signal can travel unimpeded, it becomes

difficult for WLAN Fingerprint Matching algorithms, in general, to differentiate between adjacent locations.

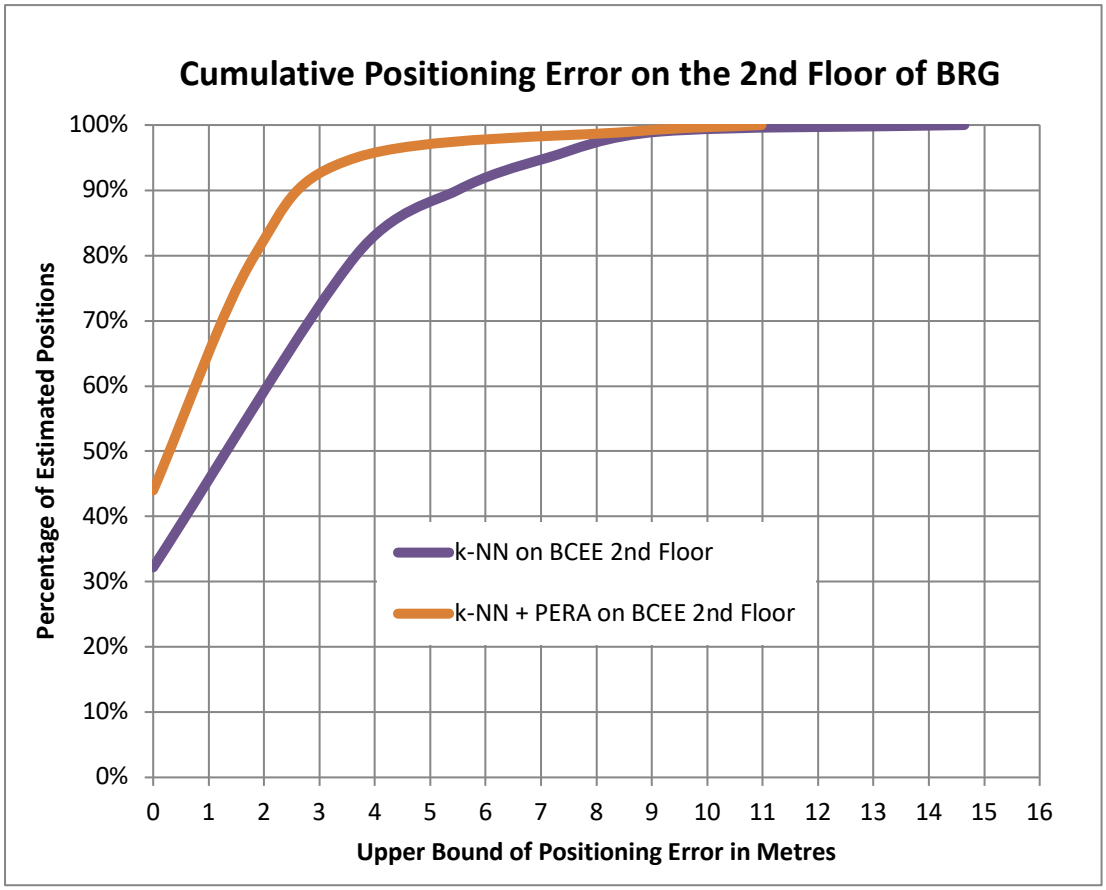

Figure 32. Cumulative Positioning Error on the 2nd Floor of BRG

6.5 Real Path RSS Measurements

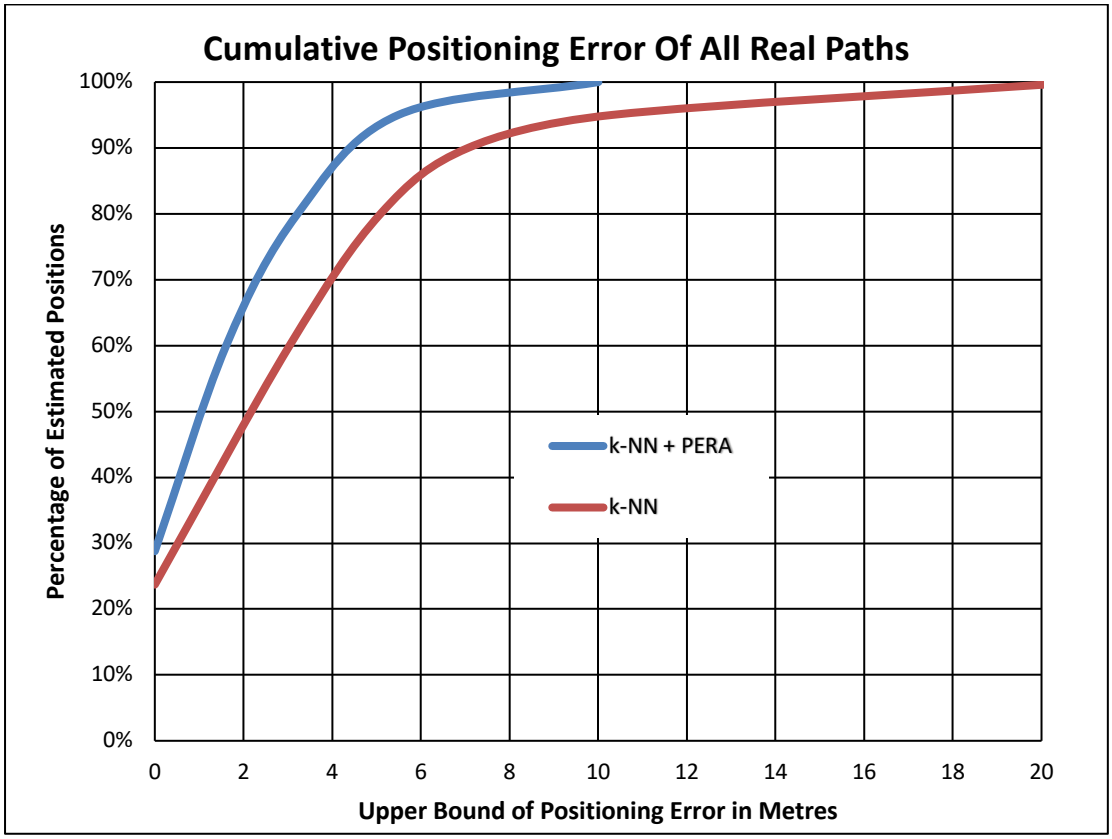

Figure 33. Cumulative Positioning Error of All Real Paths

An additional data set of about one thousand sets of RSS measurements was collected by actually walking along particular paths while recording data. The data collection, in this case, consisted of planning a few routes in a few different locations of PSEB and BRG, then walking each one while collecting data from multiple phones. The routes were all planned such that they passed a few easily identifiable landmarks or turns so that the time could be recorded at these key locations to help define the ground truth for each route. That is to say that though the spatial ground truth was pre-determined (and walked accordingly), discrete locations along the route also need to have

timestamps so that they may be compared with the correct positioning estimate (nearest in time) to evaluate the accuracy of the positioning system. The ground truth for each route was broken down into the discrete symbolic locations best representing the route based on the arrangement of symbolic locations of that floor; the symbolic locations representing the anchor points had accurate recorded timestamps, whereas the symbolic locations in-between had interpolated timestamps based on the two surrounding anchors. Due to the above, there's a bit more uncertainty in the accuracy of the ground truth; seemingly resulting in greater positioning error for these measurements.

In total, the real path testing data set consisted of 966 individual WLAN RSS measurements reported by 3 devices over the course of 33 walked paths. Figure 33 displays the positioning system's cumulative error for these measurements. As expected, due to the inaccuracy of the ground truth locations of these measurements both *k*-NN and *k*-NN + PERA have a greater degree of positioning error for the real path measurements compared to the synthetic path measurements. The median positioning error in these testing datasets was 2.5 metres for *k*-NN and 1.5 metres for *k*-NN + PERA; we observed positioning error under 5.1 metres 80% of the time for *k*-NN, while positioning error was under 3.2 metres 80% of the time for *k*-NN + PERA. More significantly, we observed positioning error under 10.3 metres 95% of the time for *k*-NN, while positioning error was under 5.5 metres 95% of the time for *k*-NN + PERA. As we can see, *k*-NN + PERA still provides significantly better positioning accuracy than *k*-NN; just as with the synthetic path measurements, the upper bound of positioning error is most significantly decreased in *k*-NN + PERA compared to *k*-NN.

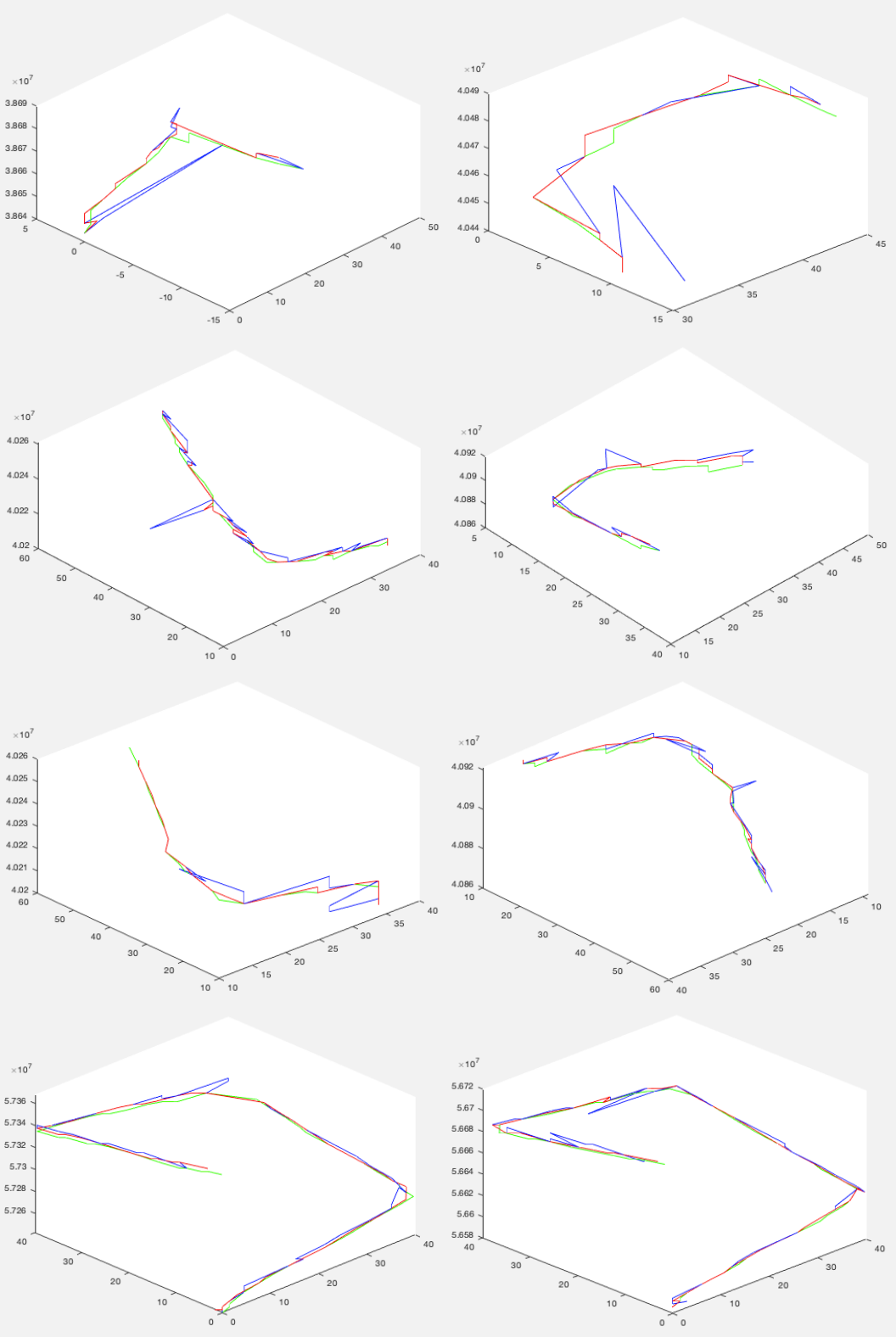

Figure 34. Visualizations of Some of the Real Path Testing Positioning Results

This highlights, once again, that the primary benefit of the PERA module is to avoid egregious positioning errors, and to generally make the positioning of a subject in motion more stable. Some examples of paths where this effect was clear can be seen in Figure 34. In this figure, the green lines represent the ground truth of the paths, the blue lines represent the series of position estimates provided by the *k*-NN algorithm, while the red lines represent the series of position estimates provided by the *k*-NN + PERA algorithm. The z-axis in each plot is time, in order to show the progression of the user along the path walked. In each path there is at least one occasion where the *k*-NN algorithm makes an egregious error that is not present in the corresponding estimated path from the *k*-NN + PERA algorithm. As those errors would constitute very irregular or unusual movement on the part of the user, they score low on the PERA module's criteria and thus do not get chosen as part of the user's most likely recent path.

Generally, WSFM algorithms provide accurate positioning, however they will occasionally provide unusually erroneous results; this can be caused by a few factors, the most common of which being incomplete or inaccurate RSS measurements. To elaborate, *incomplete* RSS measurements refer to situations where the mobile device taking the readings provides an incomplete set of the RSS that's expected in a particular area, that is to say, some AP's that should be visible in a particular area are not seen in the set of RSS values provided by the mobile device's WLAN Manager. *Inaccurate* RSS measurements refer to situations when RSS readings from one or more of the AP's visible in a particular area are much different from the values expected. This can be due to unusual obstructions causing the signal to be seriously reflected or absorbed,

malfunction or displacement of the AP, or more rarely, inadequate sampling of the signal by the mobile device's WLAN radio. When these situations occur, the PERA module will choose the most realistic of the erroneous position estimates provided by the WSFM algorithm, thereby minimizing the error in the final positioning result.

7. Conclusions

This final chapter provides some closure on the research. In the first part, the findings of this work are reviewed and explained. The context of the research process is discussed, and its success is evaluated. The second part of this chapter explores avenues of future work and research on this subject, breaking down potential future research into three separate categories.

7.1 Conclusions

In this work, I presented a positioning system, primarily based on wireless signal fingerprint matching, and including a module that evaluates the movement regularity of the subject being positioned. Initially, a WSFM algorithm estimates the *k* most likely locations where the user may be located; these *k* locations are input for the second PERA algorithm which determines the most likely recent path of the user based on the movement pattern in each of the permutations of the last few epochs of WSFM estimates. The user's recent path is then adjusted to correspond to the most likely estimate of the PERA module and the path's end point is presented to the user as their current location. A system overview of the entire positioning system was illustrated in Figure 17.

The WSFM algorithm was gradually developed based on general concepts and inspired by experimental implementations found in literature. It was also developed empirically,

with adjustments made and features added or omitted based on findings during implementation and testing. In some cases, solutions or features proposed in previous research were found to be either ineffective or counter-effective, while in other cases, they improved performance more than expected. Similarly, my own intuitions resulted in improvement and degradation of performance in equal parts. Once the performance of the WSFM algorithm was satisfactory, the objective became to create a novel algorithm or module to complement it and substantially improve the overall positioning accuracy of the system.

The inspiration for the new module came from sensor fusion positioning systems that utilize a WSFM algorithm along with a PDR module. As described in Chapter 2, these positioning systems effectively adjust the output of the WSFM module based on the movement of the user described by the PDR module. They have two important downsides, however; PDR algorithms can be inaccurate due to unreliable or imprecise sensor data and, given the required frequency of inertial sensor readings required, PDR modules can be quite computationally heavy and power hungry for mobile devices. Thus, the objective for the PERA module was to find a way to offer a correction or adjustment to the WSFM module's output without accidentally degrading accuracy nor substantially increasing computational load or energy use.

The entire end-to-end positioning system was implemented on the Android platform and deployed across several floors of multiple buildings on York University's Keele campus. The ScanTest Android application was developed as part of this research for

the purpose of data collection. The MobilePosition Android application was developed to be able to use the proposed positioning system in real-time, as well as, determine computational improvements that could be made to further enhance the user's experience. Positioning was enabled on two floors on each of the PSE, CB and BRG buildings. Finally, a Java application, that could be run on computers, was also developed to rapidly test and evaluate large datasets of testing measurements in order to provide performance evaluations of the positioning system.

As can be seen in the experimental results, the WSFM algorithm, referred in the previous section as *k*-NN, has respectable performance in its own right, achieving room-level accuracy (less than 3 - 5 metre error) 90% of the time, despite all positioning experiments taking place in open hallways. Furthermore, we show that with the PERA module, positioning accuracy is improved to less than 2 - 3 metre error about 95% of the time. In most applications, this level of accuracy is adequate for indoor positioning on the smartphone platform and comes without the added energy and computation cost of using the inertial sensors in a PDR algorithm. This sort of performance can also be considered *practically* equivalent, in my opinion, to the performance of GNSS positioning on smartphones in outdoor use cases. I believe such a system can be employed for providing location based services in indoor environments and given the relatively small data and computational requirements, it can become the ubiquitous positioning method if there is buy-in by the major players in the smartphone industry (e.g., Google, Samsung, Apple and etc.).

7.2 Future Research

Looking towards the future, this work has opened up several doors for further research in this field. Firstly, a deep sensitivity analysis is possible, if not warranted, as there were many parameters and aspects of the algorithm determined empirically during the implementation phase. Future research can also fall into a few different categories; it could explore methods of making the solution more accurate, more scalable, or more general and universally applicable.

7.2.1 Sensitivity Analysis

In the experiments performed as part of this research, the occupancy of the building was never at sufficient degree to significantly affect positioning performance. However, positioning performance can suffer at very high occupancy levels where hundreds of people may be simultaneously connected to individual APs; no such level of occupancy was observed in this research. Measurements were collected and testing was done at various times of the year, week, and day with different devices and, as such, some of the variance in positioning performance could be due to the occupancy level of the building. A sensitivity analysis could isolate and quantify the effect of high occupancy, if any, on the positioning accuracy of the proposed system.

If there is a rearrangement of the APs on a floor or building, positioning performance will degrade. The offline phase would need to be repeated as new data collection would be required. The degree to which this is necessary or when depends on the degree of

change in the configuration of the APs and could fall under a deeper sensitivity analysis of the proposed positioning system. It would be possible to quantify how much change the proposed system is resilient to and how much new data would be required to return to the initial positioning performance. This question is also addressed in existing research works, however, the answer can be specific to the individual implementation of the algorithm studied.

Another aspect of the arrangement of the APs is that it may have an effect on positioning performance. In the experimental settings in this study there were relatively dense and uniform distributions of APs; that is another aspect that could be studied in a sensitivity analysis. Furthermore, in settings where a new Wi-Fi network is about to be set up, the aforementioned sensitivity analysis can inform the arrangement of the APs in order to better support WSFM positioning systems.

Spatial resolution of the symbolic locations and data collection was chosen empirically based on the desired positioning precision and an expectation that more data would result in better positioning accuracy. A sensitivity analysis could study how much could data collection be reduced without deteriorating positioning performance. Data collection for some percentage of symbolic locations could be skipped, interpolating data instead, and the effect on positioning performance could be quantified with respect to the reduction in deployment time and cost.

7.2.2 Accuracy

One of the most important aspects of a positioning system is how accurate it is. Although I was content with the performance of the positioning system described in this work, there was certainly room for improvement. Particularly, I felt that accuracy suffered in large open areas as the WSFM algorithm was not able to distinguish position as precisely due to the homogeneity of WLAN RSS. As for the PERA module, it is not as effective in cases of unusual user movement and may be temporarily detrimental to accuracy. To this end, future work could explore the inclusion of additional data sources or further sophistication of existing algorithms. For example, inertial sensors could be incorporated as part of a dead reckoning module informing the PERA module; alternately, BLE signals could be incorporated into the WSFM algorithm to increase the variety of its inputs. The PERA module could also potentially be improved through the use of deep learning, making a future version of the PERA module more elaborate, context-aware, robust and decisive in its effect on the final estimate of the user's position and recent path.

7.2.3 Generality

Another possible extension to this research is making it more general and universally applicable as a positioning solution. Current deficiencies, in this regard, are a lack of any mechanisms for automatically determining the building that the user is currently in (in a potential multi-building deployment), or whether they are, in fact, inside; nor is there a mechanism to determine the floor the user is currently on (in a potential

deployment in a multi-floor building). These are both uniquely difficult problems that require robust solutions in order to develop a comprehensive indoor positioning system. This makes these problems excellent extensions of this research; future research towards this end could explore the adaptability of the existing algorithm for these problems, or alternately, explore other algorithms and sources of data. For example, an indoor-outdoor detection module could examine WLAN, BLE and GNSS signals in order to determine whether the user is inside a particular building. On the other hand, barometric pressure data could be used to detect when the user transfers from one floor to another. Finally, the algorithm could also be adapted to use high-band 5G cells in a similar fashion to WLAN APs, allowing the system to be deployed in even more places and take advantage of more reliable sources of data.

7.2.4 Scalability

The final logical extension of this research is an exploration of its scalability; can the positioning system be enabled across several buildings as part of a single deployment, and how can deployment become faster and easier? The primary obstacle to deployment, in the work presented here, is the segmentation of the positioning area and collection of reference fingerprints. Although this work features some innovation in these respects, they still remain a substantial part of the work required for deployment, and some of it requires a professional to execute. Future research could explore further simplifying the deployment process, such that it no longer requires professional work or perhaps becomes fully or mostly automated. Crowdsourcing could also be useful in

this regard with a potential solution employing RFID tags or QR codes to quickly identify location and begin data collection. The other consideration that pertains to the scalability of this, or any other, indoor positioning system is how will it perform in very large buildings or a multi-building deployment. Performance, in this context, refers not only to positioning accuracy but also the computational performance on the user's device. As the size of the deployment is scaled up, the amount of data processed by the positioning system increases, both in total and for each instantaneous position estimate. Although, computational efficiency was addressed in this research, I suspect there is still much more room for improvement.

References

Bahl, P. and Padmanabhan, V. N. (2000) 'RADAR: An in-building RF-based user location and tracking system', in *Proceedings - IEEE INFOCOM*. doi: 10.1109/infcom.2000.832252.

Bai, Y. B. *et al.* (2014) 'A new method for improving Wi-Fi-based indoor positioning accuracy', *Journal of Location Based Services*, 8(3), pp. 135–147. doi: 10.1080/17489725.2014.977362.

Beder, C. and Klepal, M. (2012) 'Fingerprinting based localisation revisited: A rigorous approach for comparing RSSI measurements coping with missed access points and differing antenna attenuations', in *2012 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2012 - Conference Proceedings*. doi: 10.1109/IPIN.2012.6418940.

Bolliger, P. (2011) 'Robust Indoor Positioning through Adaptive Collaborative Labeling of Location Fingerprints', (20109). Available at: http://www.vs.inf.ethz.ch/publ/papers/bolligph-thesis-2010.pdf.

Chan, S. (2013) *A Hybrid Probabilistic Approach for WIFI Based Indoor Location Estimation*. York University.

Chen, Z. *et al.* (2015) 'Fusion of WiFi, Smartphone Sensors and Landmarks Using the Kalman Filter for Indoor Localization', *Sensors*, 15(1), pp. 715–732. doi: 10.3390/s150100715.

Curran, K. *et al.* (2011) ‘An evaluation of indoor location determination technologies’, *Journal of Location Based Services*, 5(2), pp. 61–78. doi: 10.1080/17489725.2011.562927.

Dawes, B. and Chin, K.-W. (2011) ‘A comparison of deterministic and probabilistic methods for indoor localization’, *Journal of Systems and Software*. Elsevier Inc., 84(3), pp. 442–451. doi: 10.1016/j.jss.2010.11.888.

Dumbgen, F. *et al.* (2019) ‘Multi-modal probabilistic indoor localization on a smartphone’, *2019 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2019*. doi: 10.1109/IPIN.2019.8911765.

Durrant-Whyte, H. and Bailey, T. (2006a) ‘Simultaneous Localisation and Mapping (SLAM): Part II The State of the Art’, *Robotics and Automation Magazine*, 13(3), pp. 1–10.

Durrant-Whyte, H. and Bailey, T. (2006b) ‘Simultaneous localization and mapping: Part I The Essential Algorithms’, *IEEE Robotics and Automation Magazine*, 13(2), pp. 99–108. doi: 10.1109/MRA.2006.1638022.

Ebner, F. *et al.* (2014) ‘Robust self-localization using Wi-Fi, step/turn-detection and recursive density estimation’, *IPIN 2014 - 2014 International Conference on Indoor Positioning and Indoor Navigation*, (October), pp. 627–635. doi: 10.1109/IPIN.2014.7275537.

Ekahau RTLS (2015). Available at: www.ekahau.com.

Farshad, A. *et al.* (2013) ‘A microscopic look at WiFi fingerprinting for indoor mobile phone localization in diverse environments’, *2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013*, (October), pp. 28–31. doi: 10.1109/IPIN.2013.6817920.

Gulo, E., Sohn, G. and Ahmad, A. (2018) ‘Indoor positioning using wlan fingerprint matching and path assessment with retroactive adjustment on mobile devices’, *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*. Copernicus Publications, 42(4), pp. 321–327. doi: 10.5194/isprs-archives-XLII-4-253-2018.

Guo, X. *et al.* (2020) ‘A survey on fusion-based indoor positioning’, *IEEE Communications Surveys and Tutorials*. IEEE, 22(1), pp. 566–594. doi: 10.1109/COMST.2019.2951036.

Hafner, P. *et al.* (2013) ‘Evaluation of smartphone-based indoor positioning using different Bayes filters’, *2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013*, (October). doi: 10.1109/IPIN.2013.6817876.

Harle, R. (2013) ‘A survey of indoor inertial positioning systems for pedestrians’, *IEEE Communications Surveys and Tutorials*, pp. 1281–1293. doi: 10.1109/SURV.2012.121912.00075.

He, S. and Chan, S. H. G. (2016) ‘Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons’, *IEEE Communications Surveys and Tutorials*. IEEE, 18(1), pp. 466–490. doi: 10.1109/COMST.2015.2464084.

HERE Indoor Positioning (2016).

Herrera, J. C. A. *et al.* (2014) 'Pedestrian indoor positioning using smartphone multi-sensing, radio beacons, user positions probability map and IndoorOSM floor plan representation', *IPIN 2014 - 2014 International Conference on Indoor Positioning and Indoor Navigation*, (October), pp. 636–645. doi: 10.1109/IPIN.2014.7275538.

Honkavirta, V. *et al.* (2009) 'A comparative survey of WLAN location fingerprinting methods', *Proceedings - 6th Workshop on Positioning, Navigation and Communication, WPNC 2009*, 2009, pp. 243–251. doi: 10.1109/WPNC.2009.4907834.

Kjærgaard, M. B. (2007) 'A taxonomy for radio location fingerprinting', *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4718 LNCS, pp. 139–156. doi: 10.1007/978-3-540-75160-1_9.

Lane, N. D. *et al.* (2010) 'A survey of mobile phone sensing', *IEEE Communications Magazine*, 48(9), pp. 140–150. doi: 10.1109/MCOM.2010.5560598.

Laoudias, C., Zeinalipour-Yazti, D. and Panayiotou, C. G. (2013) 'Crowdsourced indoor localization for diverse devices through radiomap fusion', *2013 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2013*, (October), pp. 28–31. doi: 10.1109/IPIN.2013.6817906.

Lemic, F. *et al.* (2014) 'Experimental decomposition of the performance of fingerprinting-based localization algorithms', in *IPIN 2014 - 2014 International Conference on Indoor Positioning and Indoor Navigation*, pp. 355–364. doi: 10.1109/IPIN.2014.7275503.

Li, B. *et al.* (2012) ‘How feasible is the use of magnetic field alone for indoor positioning?’, *2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, (November), pp. 1–9. doi: 10.1109/IPIN.2012.6418880.

Liu, W. *et al.* (2019) ‘Survey on CSI-based indoor positioning systems and recent advances’, *2019 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2019*. IEEE, (61871054), pp. 1–8. doi: 10.1109/IPIN.2019.8911774.

Machaj, J. and Brida, P. (2012) ‘Optimization of rank based fingerprinting localization algorithm’, *2012 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2012 - Conference Proceedings*, (November). doi: 10.1109/IPIN.2012.6418921.

Martin, E. *et al.* (2010) ‘Precise Indoor Localization Using Smart Phones’, *MM ’10 Proceedings of the International conference on Multimedia*, pp. 787–790. doi: 10.1145/1873951.1874078.

Martin, P. *et al.* (2014) ‘An ibeacon primer for indoor localization’, *BuildSys 2014 - Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pp. 190–191. doi: 10.1145/2674061.2675028.

Mautz, R. and Tilch, S. (2011) ‘Survey of optical indoor positioning systems’, in *2011 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2011*. doi: 10.1109/IPIN.2011.6071925.

Mirowski, P. *et al.* (2011) ‘KL-divergence kernel regression for non-Gaussian fingerprint based localization’, *2011 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2011*. doi: 10.1109/IPIN.2011.6071928.

Nister, D., Naroditsky, O. and Bergen, J. (2004) 'Visual odometry', in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. IEEE, pp. I-652-I-659 Vol.1. doi: 10.1109/CVPR.2004.1315094.

Ontario Professional Surveyor, Volume 58, No. 2 (2015) 'Ninth Annual AOLS Graduate Student Geomatics Poster Session Award Winners', pp. 18–19.

Renaudin, V. *et al.* (2019) 'Evaluating Indoor Positioning Systems in a Shopping Mall: The Lessons Learned from the IPIN 2018 Competition', *IEEE Access*, 7, pp. 148594–148628. doi: 10.1109/ACCESS.2019.2944389.

Senion IPS (2016). Available at: https://senion.com/senion-ips/.

Subbu, K. *et al.* (2014) 'Analysis and status quo of smartphone-based indoor localization systems', *IEEE Wireless Communications*, 21(4), pp. 106–112. doi: 10.1109/MWC.2014.6882302.

Tian, Z. *et al.* (2015) 'Smartphone-Based Indoor IntegratedWiFi/MEMS Positioning Algorithm in a Multi-Floor Environment', *Micromachines*, 6(3), pp. 347–363. doi: 10.3390/mi6030347.

Torteeka, P., Chundi, X. and Dongkai, Y. (2014) 'Hybrid technique for indoor positioning system based on Wi-Fi received signal strength indication', *IPIN 2014 - 2014 International Conference on Indoor Positioning and Indoor Navigation*, pp. 48–57. doi: 10.1109/IPIN.2014.7275467.

Villien, C., Frassati, A. and Flament, B. (2019) ‘Evaluation of an indoor localization engine’, *2019 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2019*. IEEE. doi: 10.1109/IPIN.2019.8911799.

Yan, S. *et al.* (2019) ‘Wi-Fi RTT based indoor positioning with dynamic weighted multidimensional scaling’, *2019 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2019*. IEEE, pp. 12–15. doi: 10.1109/IPIN.2019.8911783.

Youssef, M. and Agrawala, A. (2008) ‘The Horus location determination system’, *Wireless Networks*, 14(3), pp. 357–374. doi: 10.1007/s11276-006-0725-7.

Zhang, Y., Chen, J. and Xue, W. (2018) ‘Unsupervised indoor localization based on Smartphone Sensors, iBeacon and Wi-Fi’, *Proceedings of 5th IEEE Conference on Ubiquitous Positioning, Indoor Navigation and Location-Based Services, UPINLBS 2018*. IEEE. doi: 10.1109/UPINLBS.2018.8559713.

Zhou, C. and Wieser, A. (2018) ‘CDM: Compound Dissimilarity Measure and an Application to Fingerprinting-Based Positioning’, *IPIN 2018 - 9th International Conference on Indoor Positioning and Indoor Navigation*. IEEE. doi: 10.1109/IPIN.2018.8533681.