Application of Half Spaces in Bounding Wireless Internet Signals for use in Indoor Positioning

BY

Dain Vermaak

Submitted to the graduate degree program in Electrical Engineering & Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Master of Science in Computer Engineering

Chairperson Dr. Joseph Evans

Dr. James Miller

Dr. Gary Minden

Date Defended: 5/4/2015

The Thesis Committee for Dain Vermaak
certifies that this is the approved version of the following thesis:

Application of Negative Half Spaces for Bounding Wireless Internet Signals in Indoor Positioning

Chairperson Dr. Joseph Evans

Date approved: 5/4/2015

Abstract

The problem of outdoor positioning has been largely solved via the use of GPS. This thesis addresses the problem of determining position in areas where GPS is unavailable. No clear solution exists for indoor localization and all approximation methods offer unique drawbacks. To mitigate the drawbacks, robust systems combine multiple complementary approaches. In this thesis, fusion of wireless internet access points and inertial sensors was used to allow indoor positioning without the need for prior information regarding surroundings. Implementation of the algorithm involved development of three separate systems. The first system simply combines inertial sensors on the Android Nexus 7 to form a step counter capable of providing marginally accurate initial measurements. Having achieved reliable initial measurements, the second system receives signal strength from nearby wireless internet access points, augmenting the sensor data in order to generate half-planes. The half-planes partition the available space and bound the possible region in which each access point can exist. Lastly, the third system addresses the tendency of the step counter to lose accuracy over time by using the recently established positions of the access points to correct flawed values. The resulting process forms a simple feedback loop.

A primary contribution of this thesis is an algorithm for determining access point position. Testing shows that in certain applications access points relatively near the user's path of travel can be positioned with high accuracy. Additionally, the nature of the design means that the geometric algorithm has a tendency to achieve maximum performance in environments containing many twists and turns while suffering from a lack of useful data on straight paths. In contrast, winding areas confound the step counter which performs better when used in long straight stretches of constant movement. When combined, these trends complement one another and result in a robust final product.

Contents

Abstract iii

1. Introduction 2

2. Background 3

3. Architecture 8

4. Implementation 10

4.1 Inertial Sensors 10

4.2 Half -Planes 24

4.3 Visualization 44

5 Evaluation 45

5.1 Step Counter 46

5.2 Router Location Determination 47

5.3 Induced Position 53

6 Conclusions 56

7 Future Work 58

Bibliography 59

1. Introduction

Advertising, augmented reality, urban warfare, and limitless other applications all benefit immensely from indoor positioning. Implementations range from simply choosing the best route through a mall to those as complex as inventory management for massive factories or shopping centers. This thesis drew inspiration from its own unique scenario. The initial problem posed required that a rapid response team in search of radioactive material be able to gain a general understanding of their surroundings given little or no previous data. In particular the users involved had to be able to relay directional information sufficient to lead any supporting personnel from the entrance of the building to the site of the radiation. The scenario defined the key assumptions:

1. The solution had to be deployed as a lightweight application and should target the Android environment.
2. The solution should assume no prior knowledge of surroundings save possibly a GPS position at the entrance to the building.
3. Users of the solution would, by necessity, be in almost constant motion as they attempted to locate the threat.

This Masters thesis proposes an approach to indoor positioning that applies geometric methods to existing wireless internet approaches in order to enhance the performance of directional step counters implemented on android devices.

2. Background

Given the vast array of applications for accurate indoor positioning, it is unsurprising that there have been a large number of different approaches. Key approaches usually include Global Positioning Systems(GPS) which, despite a few limitations, still find use in indoor positioning. Another common approach is utilization of wireless internet technology whose rise in popularity, already established infrastructure and lack of dependence on line of sight often make it a more reliable option than GPS. Many notable techniques rely on wireless internet technology and associated access points to derive a position based on strength or propagation time. These techniques often employ learning or statistical algorithms to account for the variation in signal. Wireless internet has been driven in large part by the advent of smart phone technology and as smart phones become more and more a part of society it is only natural that they themselves be considered as a tool for indoor positioning. The ever increasing number of sensors on modern smart phones has given rise to approaches in which inertial sensors are utilized to provide a fairly accurate approximation of position. In contrast to the marginally non-invasive techniques mentioned previously other techniques such as individual tagging of objects with broadcasting units and associated widespread sensor networks have been devised and implemented successfully. Despite the myriad options available, there is no technology currently accepted as clearly superior. Some approaches are exceptionally well suited to certain environments while suffering miserably from their limitations in others.

Having enjoyed success in outdoor positioning, GPS is an obvious first contender for indoor positioning systems. Counting on the presence of orbiting satellites, GPS devices triangulate a user's position on the earth. While inarguably valuable in many situations, both within buildings and without, GPS suffers as a general indoor positioning tool. In order to perform, a GPS unit

must have an unobstructed view of the sky through which to receive signals from no less than four separate satellites. Any obstacle be it concrete, paper, or even passing clouds can render a GPS device useless. As a result, indoor positioning systems generally begin design under the belief that GPS will not be available as a viable primary means of determining position. However GPS still has its uses and in many cases may be employed in a supporting role.

Another common technology used is wireless internet. As the internet has continued to emerge as a dominant communication medium, the number of wireless access points (AP) has increased. This has happened to such a degree that it is often sufficient to rely on the presence of these access points to determine location. Most modern approaches to utilizing an AP for position require two key assumptions. The first assumption is that the position of the AP is known. The position can be either absolute with respect to global values (longitude/latitude) or can be relative (to an entry point for example). Another key assumption is that the signal from an AP is constant over time. The result of this fact is usually a circular area describing the region around a router in which the RSSI (Received Signal Strength Index) can be interpolated. Limitations to using wireless internet as a positioning technique arise from the fact that neither of these two assumptions can be guaranteed. In the first case, while many APs do have the capability to broadcast their position, the vast majority cannot or will not do so. This means that even if APs are present, there is no assurance that they are usable. The second assumption faces a similar problem. Wireless internet signals are not constant. Their signals vary over time in response to wireless traffic, power fluctuations, competing wireless devices, or even interference from other APs. Even the signal strength recorded by the device while stationary will vary. This means that it is difficult to fix a ratio between RSSI and the distance to a device. To further complicate matters, as anyone who has tried to cover a home with a wireless router knows, wireless signals can also be reflected by any number of

physical obstacles. The resulting chaos creates areas in which signal strength is inexplicably stronger or weaker than their immediate surroundings. This thesis attempts to circumvent the weaknesses of wireless internet access points by disregarding both of the initial assumptions. While the shape of the signal will still be considered circular for some purposes, there will be no assumption that the access point is in fact the center of the circle. The second assumption: that the position of the AP is known and constant will also be ignored. Before any AP point is used to determine user position, its own position will first need to be ascertained. The challenge to using wireless internet then becomes an issue of minimizing its limitations.

The success and subsequent inclusion of smart phone technology in society has provided an opportunity for the development of inertial sensors as a positioning device. Simple physics reveals the relationship between acceleration and position. The presence of accelerometers on almost every smart phone device seems to imply a logical path of inquiry. Unfortunately this is not entirely true. Double integration must be applied to the accelerometer reading in order to derive first velocity and position. This means that any errors in a acceleration measurement are compounded exponentially into the position. Further complications are added by the existence of gravity and the noise generated through ambient vibration and unnecessary user movement. Ongoing work in this field continues, using additional sensors such as barometers, gyroscopes, magnetometers, and GPS in tandem with the accelerometer in an effort to defray the error costs. Success has also been had in using accelerometers attached to different locations on a user such as the shoes to provide a much more reliable and predictable pattern of movement. A natural extension of these ideas has led to the use of pedometers. In this conventional use of inertial sensors, reliance is placed on the fusion of the accelerometer with a rotation device (usually a gyroscope or magnetometer) to form a step counter often while simultaneously recording heading. While it has its own set of limitations, a

step counter is limited by the sinusoidal motion inherent in a person's natural stride and therefore tends to suffer less from the exponential error observed in straight dead reckoning. A key limitation of step counters is their tendency to generate false steps. Combined with the fact that every person not only has a distinct stride but also changes their stride for a host of different reasons, step counters make a useful approximation tool rather than a precise measurement. Despite this fact step counters remain one of the more viable options available for indoor positioning. This thesis implements a simple step counter in order to provide an initial estimate of relative position, attempting to contain the error produced by false steps through the use of heading analysis and calibrated thresholds.

Another approach to indoor positioning, and one that finds its niche in inventory management, is the use of sensor tagging. Distaining the less invasive methods mentioned previously, items or locations are "tagged" with sensors that broadcast a signal. The signal is subsequently recorded and processed by receivers which then relay position information to the user. Tagging variants range anywhere from Radio Frequency Identification (RFID) to Infrared (IF) signals and are far more reliable than wireless internet when implemented correctly. Regardless of the type used, however, such an infrastructure must generally be separate and unrelated to a company's information pipeline. As a result they can be very costly to implement due to the additional hardware requirements. This same condition suggests that scaling such a system once it is in place is also prohibitively expensive.

Every approach listed has both strengths and limitations that they bring to the table. GPS provides a means to determine position in a global reference frame yet requires a clear view of the sky to do so. Wireless internet has saturated most places in which indoor positioning would be useful but is fickle to the point of unreliability. Inertial sensors continue to improve though those

currently implemented on widely available smart phone technology contain too much error to be a viable standalone option. Even techniques such as widespread tagging require overhead costs that render it unattainable to all but the most dedicated consumers. Fusion techniques in which two or more of these approaches are combined have shown promise and continue to be an area in which research progresses. In short, indoor positioning is a problem that does not yet have an acceptable general solution.

3. Architecture

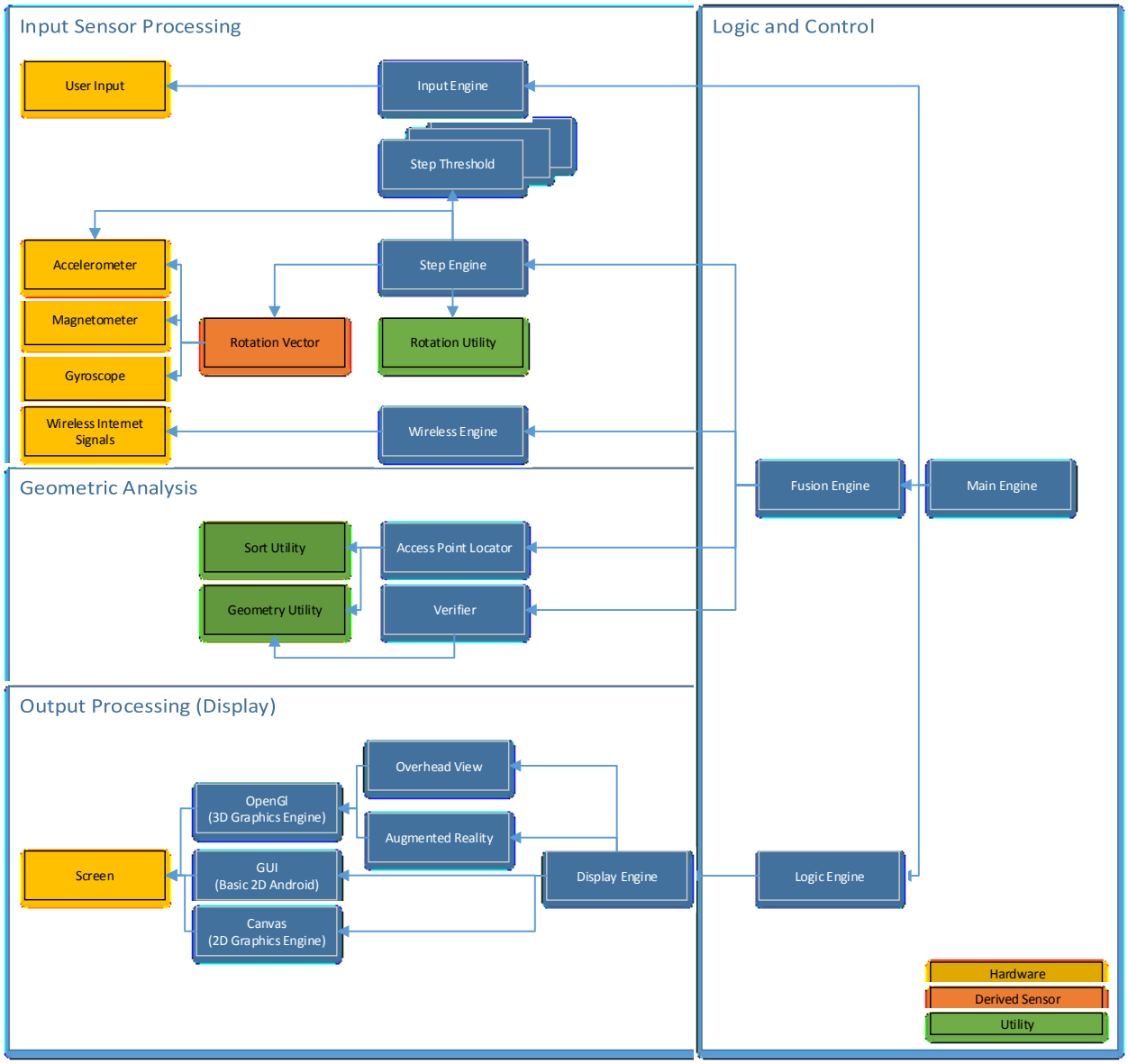

Figure 1 - High Level Architecture

To understand the architecture above, it is important to define the meaning of the arrows. The arrows in the Figure 1 imply a "uses" or "controls" relationship where the arrow end indicates the object that is being controlled. For example: in Figure 2 below, class A uses (or controls) class B.

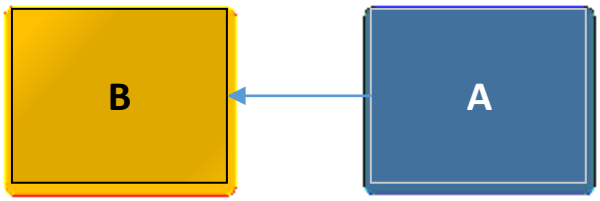

Figure 2 - Arrow Use

As with any large software design effort, care was taken early on to clearly separate and define the many different parts of the final product. The broad scope of the project naturally divided into three distinct parts based on functionality and a fourth acting as a controlling body (Figure 1). As shown above, the first and most hardware intensive section is given over to input processing. Classes within the input processing section are responsible for receiving input from the various on-board sensors and performing any post-processing. Post-processing includes steps such as key interpretation, accelerometer alignment, and so forth. On the other side of Figure 1 is the output processing and the various visualization packages available to the Android device. As the name implies, classes within the output processing section transform the endless streams of incoming data into visual interpretations. Between the input and output sections lies a section termed Geometric Analysis. The Geometric Analysis is the heart and soul of the project and contains concepts and ideas unique to the approach presented in this thesis. Within this section data from processed sensory input is combined and merged to provide a means of verifying the position dictated by the step counter through use of wireless APs. The Fusion engine is then responsible for coordinating what data is provided to each section and acts as a data bus between sections allowing for filtered communication when necessary. The remaining classes in the Logic and Control section simply move data between input and output and provide any additional functionality needed to integrate with device hardware.

4. Implementation

Implementation of a working product on the android device continues to employ the same segregation suggested by Figure 1. The following sections serve to elaborate on the approaches used in each case.

4.1 Inertial Sensors

An important component of this thesis is the implementation of a simple pedometer capable of recording direction as well as steps taken in that direction. Newer Android devices include a step counter implementation as a separate sensor. While a tempting proposition, it quickly became evident that building one from existing architecture would provide a few benefits over using pre-built step counters. In the first case, the integration of direction as well as distance travelled would be far simpler if it could be attached directly to the accelerometer callbacks. Even such a small change would allow for a cleaner overall architecture. Since many current android devices lack even the standard sensors it was deemed unrealistic to target a sensor that was both non-standard and relatively new. The various inertial sensors on the Android Nexus 7 are therefore used to define a step counter capable of providing an acceptable baseline measurement.

Inertial sensors are generally defined as accelerometers and gyroscopes, though for the purposes of navigation magnetometers are often adopted to improve performance. The navigation algorithm presented in this thesis relies on the accelerometer to determine the time at which a step begins and ends in order to update the pedometer. In conjunction to the accelerometer a means to determine device orientation is required. The rotation vector was selected from the options available for a pair of significant reasons. The first reason to obtain orientation data was that the accelerometer was fixed to the device axis of rotation and so was unable to determine the direction

of gravity without a delay for calibration. Orientation information provides a way to circumvent this problem via axis alignment. The second use of orientation data is to provide a simple heading measurement for the steps taken. Despite the efforts of aligning the various axis, the signal received from the accelerometer is still too noisy for immediate use and requires a band-pass filter to remove the different obfuscating frequencies. Once filtering is complete, final accommodations need to be made for situations in which the pedometer records false steps. This is especially true in the case where the device itself was held stationary. Fusing these many parts into a single cohesive whole provides a way to implement a fairly accurate step counter in a manner less resource intensive and more platform independent than possible alternatives.

4.1.1 Accelerometer

Accelerometers, as the name implies, measure acceleration in m/s^2. Any change in velocity experienced by the accelerometer is recorded and displayed as occurring across one of the three axis coordinates. Fortunately, according to Android, the accelerometer is also present on the majority of android devices. Unfortunately the implementation of the algorithm uses the accelerometer present on the Android Nexus 7 tablet and so is subject to its coordinate reference frame. For android accelerometers the x-axis is considered to be the axis pointing to the right of the device while the y-axis points to the top of the device. The z-axis then points directly out of the front screen at the user. See Figure 3 below.

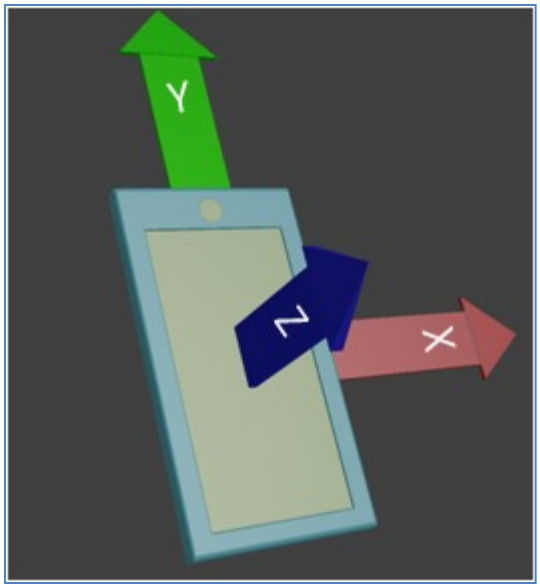

Figure 3 - Accelerometer Axis

With the grim understanding that later filtering would be needed, if only for the gravitational component, a sampling rate that android refers to as a "rate suitable for games" was chosen. Investigation showed that the selected sampling rate produced a delay of 20000 microseconds or an equivalent frequency of 50 Hz. Given that the frequency was well above the optimistic 6 Hz selected as the maximum frequency of a human's walking speed, it was considered a reasonable choice.

One thing to note is that, true to physical systems, accelerometers always have an influence due to gravity. An accelerometer left stationary will still record an acceleration of $9.81\ m/s^2$in the direction of the earth's center. Gravity becomes a serious issue as it will not necessarily align with any device axis and cannot therefore be easily removed. Since the device can be held at any orientation, the accelerometer's measurement of gravity can likewise point in any direction relative to the device. Beyond gravitational force, other sources of error were found to effect the accelerometer. Large amounts of noise arise from human contact and a person's general inability to maintain a position of absolute stability. A common approach when dealing with gravity is to

implement a calibration stage in which the device is placed on a fixed surface and measurements are taken over time to establish the initial direction of gravity. Once complete, device orientation must then be tracked in order to maintain the derived direction. Given the additional overhead both in time to calibrate correctly and resources needed to keep track of gravity, this approach was rejected. Instead of relying on calibration, the orientation sensors on the Android device were used to align gravity to a fixed axis and, along with a some careful signal processing, eventually remove it from the equation entirely.

4.1.2 Rotation Vector

Further implementation of the algorithm requires knowledge of device orientation for two separate aspects. The first is to mitigate the signal noise introduced by gravity and the second is to track the direction of travel. The Android Nexus 7 offers three sensors for determining orientation: gyroscope, magnetometer, and rotation vector. The rotation vector is a compound device resulting from the fusion of the accelerometer, gyroscope, and magnetometer and as such is generally the best choice for orientation detection. The result of a rotation vector sampling event is a three dimensional vector. Android defines this vector as containing the three axis coordinates of the four-variable unit quaternion Q described below and defines the magnitude as $\sin\left(\frac{\theta}{2}\right)$.

$$Q = \left(\cos\left(\frac{\theta}{2}\right), x * \sin\left(\frac{\theta}{2}\right), y * \sin\left(\frac{\theta}{2}\right), z * \sin\left(\frac{\theta}{2}\right)\right)$$

To understand why quaternions are useful it is important to first understand gimbal lock. Gimbal lock is a condition that occurs when two axis of rotation align. When this occurs there is no way to separate them, as any change to the one will invariably effect the other in the same manner. The results of this condition are universally bad. Unexpected results appear

inconsistently, making gimbal lock a difficult problem to debug. A unit quaternion as used in this thesis is also known as a versor and provides a method of performing three dimensional rotation. The rotation occurs in a manner similar to rotation matrices which rely on Euler angles but without the threat of gimbal lock. In the quaternion above the first term describes the magnitude of a rotation and the last three represent its axis. At this point it is worth noting that the coordinate system of the rotation vector is not the same as that of the accelerometer. Rather than being fixed to the device frame, it is fixed to the earth's reference frame (see Figure 4).

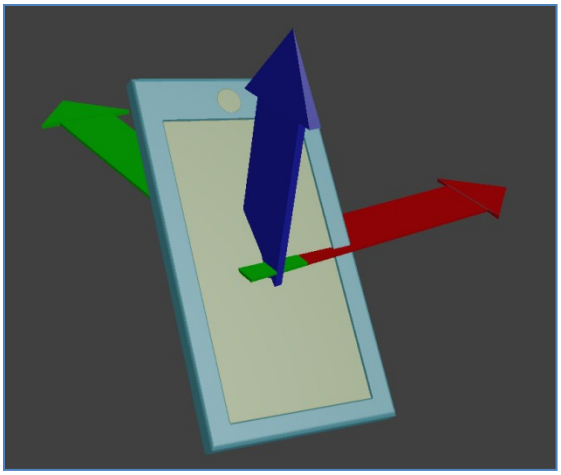

Figure 4 - Rotation Vector Axis

As a result, the z-axis points directly up into space in a direction perpendicular to the tangent to the earth's surface. The y axis points in the direction tangent to the earth's surface and towards magnetic north. The x-axis, therefore, points in a direction perpendicular to y and z which becomes the earth's "magnetic" east. These axis are fixed irrespective of the device's orientation and the rotation vector returns the quaternion necessary to bring the device back to this frame. A side effect of the rotation vector's orientation is that the z-axis of the orientation vector always points in a direction parallel to gravity which is exactly what is needed to align the accelerometer.

4.1.3 Axis Alignment

Differences in the axis frame of reference between the accelerometer and the rotation vector represent a large hurdle in the construction of any step counter. At its core the step counter needs to measure acceleration only in the direction of gravity. Therefore, the ideal situation is that no matter what the orientation of the device, the negative z-axis should always point to the center of the earth. Recall that the accelerometer defines the x-axis as device right, the y axis as device up, and the z axis as the direction pointing directly out of the screen. In contrast the rotation vector defines the x-axis as pointing to the earth's east, the y-axis as pointing to the earth's north, and the z-axis pointing into space. Given that the rotation vector knows exactly what the direction of gravity is, the problem becomes one of aligning the accelerometer's frame of reference to the rotation vector's frame of reference.

Given a rotational vector sample:

$$S_{rotationvector} = (A, B, C)$$

The three components can be defined mathematically as per the definition of the rotation vector:

$$A = x * \sin\left(\frac{\theta}{2}\right)$$

$$B = y * \sin\left(\frac{\theta}{2}\right)$$

$$C = z * \sin\left(\frac{\theta}{2}\right)$$

Under the assumption that the magnitude of the rotation vector is defined to be $\sin\left(\frac{\theta}{2}\right)$ it is possible to derive θ from the magnitude and, subsequently, the final quaternion:

$$|R| = sin\left(\frac{\theta}{2}\right) = \sqrt{A^2 + B^2 + C^2}$$

$$\theta = 2 * arcsin(|R|)$$

$$Q_R = \left(A, B, C, cos\left(arcsin(|R|)\right)\right)$$

$$Q_R = \left(A, B, C, \sqrt{1 - |R|^2}\right)$$

Q_R is therefore described as a unit quaternion rotating around the axis defined by A,B, and C by a given amount as expressed as $\sqrt{1 - |R|^2}$. Any three dimensional point $\vec{P} = (P_x, P_y, P_z)$ can be rotated around the axis defined by $Q_R$by first transforming the point it into a quaternion and then applying quaternion multiplication:

$$Q_P = (P_x, P_y, P_z, 0)$$

$$Q_F = Q_R * Q_P * Q_P^{-1}$$

Where $Q_P$is the point $\vec{P}$ transformed into its quaternion form and Q_F is the result of multiplication of this quaternion with Q_R defined above. In the implementation an additional step is added during the conversion of the final quaternion back into three dimensional space. As the implementation was to use OpenGl as a rendering platform the z-axis was flipped so as to match the OpenGl coordinate reference-frame. Using this method any sample taken from the accelerometer is first converted to a quaternion, multiplied through the rotation quaternion, and transferred back into OpenGl space. The result is an accelerometer sample that is aligned to the earth's reference frame with its y axis pointing up into space, its z-axis pointing south, and its x-axis pointing east regardless of which way was the device was held (Figure 5).

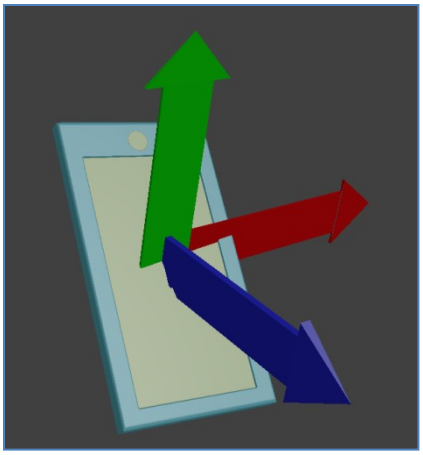

Figure 5 - Final Reference Axis

The end result of the whole process is that the y-value of the acceleration vector always points directly towards the center of the earth.

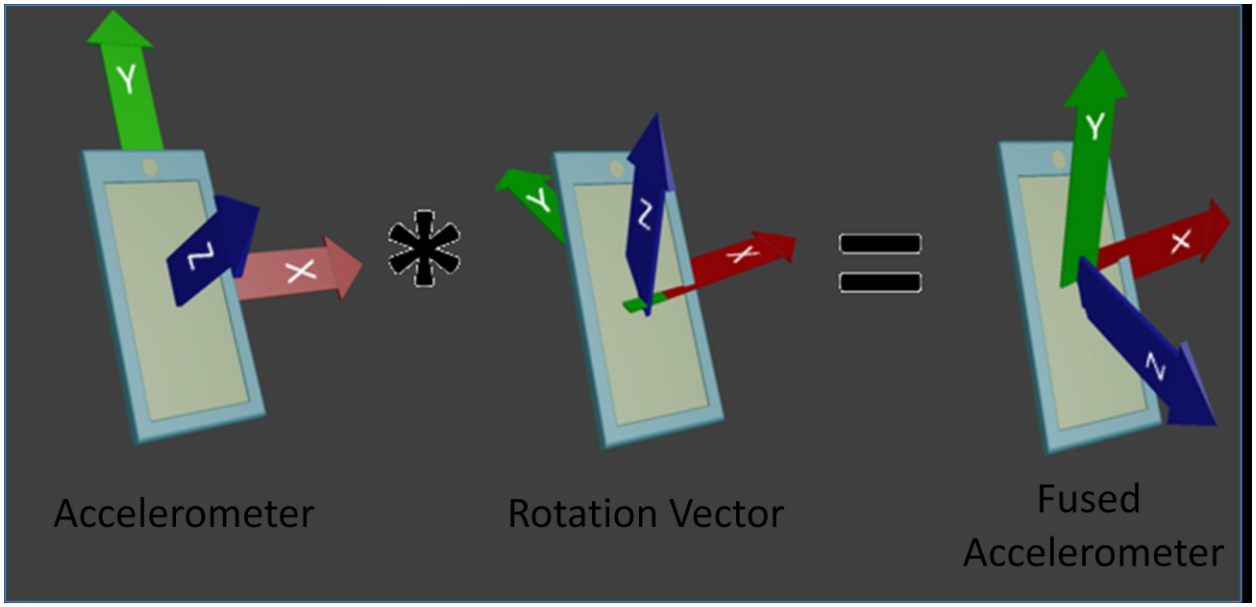

Figure 6 - Axis alignment

4.1.4 Compass

The compass is a necessary part of the design as it provides heading information essential when using the step counter for position. Rather than use the magnetometer as is usually done, the

existing architecture was leveraged to make use of the rotation vector. First a point was projected out the back of the device at a vector (0,1,-1). This was accomplished simply by applying the rotation quaternion derived previously to convert the point into the earth's reference frame. Once the point was projected it was normalized and the result divided it into its y and x-z components. The y-component was then used as a measurement of device tilt as shown in Figure 7. As the device is turned the y-component of the resultant vector goes from -1 to 1 in a smooth arc. Simple linear interpolation between zero and 180^{o} was then assumed in order to simply equations.

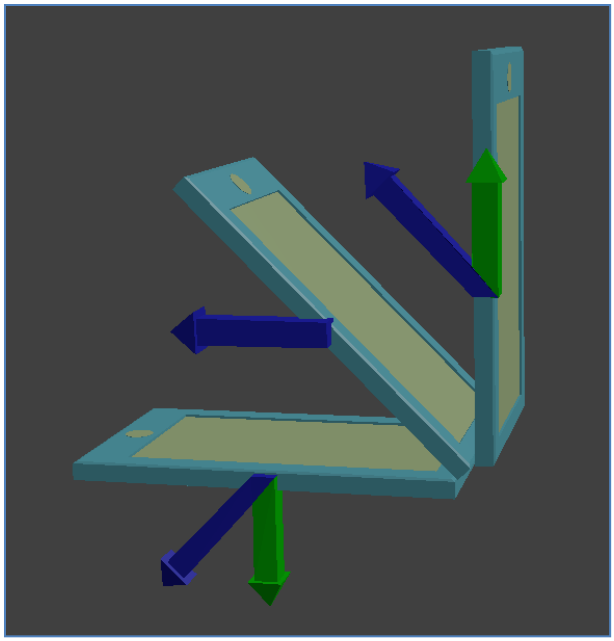

Figure 7 - Compass Tilt

Normalizing the x-z component of the projected point returns a unit vector that points away from the device in the direction (on the correct x-z-plane) towards the top of the device. Figure 8 below shows how, as the device is rotated, the x-z vector (red) clearly points in the direction of the top of the device.

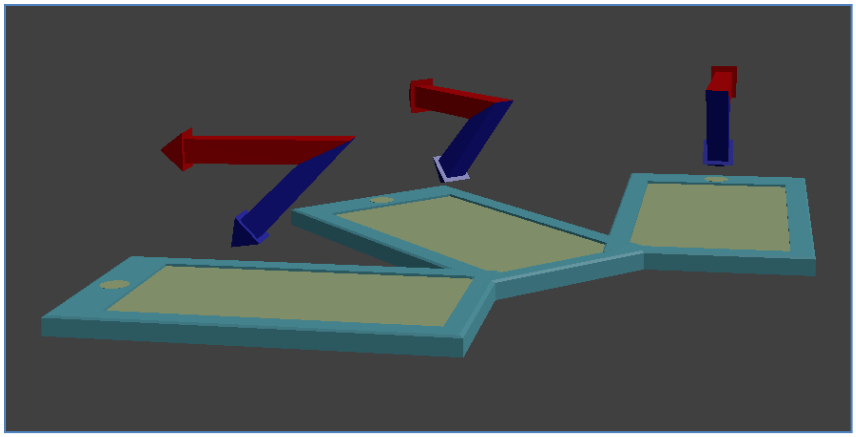

Figure 8 - Compass Azimuth

4.1.5 Signal Processing

With the accelerometer firmly pinned to the earth's reference frame, the y-axis is isolated and measurements taken. Brief analysis show that the results still contain too much noise to be of use. Two factors are responsible for noise on the accelerometer. The first, as previously mentioned, is gravity, which is realized as a low frequency signal. The second is high frequency noise due to vibration. Given these facts, a band pass filter is applied via combined low and high pass digital filters. The graphs shown below were synthesized using Matlab to demonstrate the effect of the filters on the initially noisy signal shown in Figure 10.

A simple digital low pass filter was used to filter out the high frequency noise (Figure 11). .

$$Given:$$

$$s \equiv sample, the\ accelerometer\ y\ axis\ value$$

$$f \equiv cutoff\ frequency$$

$$i \equiv interval, time\ between\ samples$$

$$v \equiv previous\ filtered\ value$$

$$\tau = \frac{1}{2\pi f}$$

$$\alpha = \frac{i}{\tau + \alpha}$$

$$v = (a * s) + \big((1 - \alpha) * v\big)$$

$$return\ v$$

For the purposes of the implementation a conservative cutoff frequency of 6 Hz is selected by reasoning that the average human being is unable to walk at a speed of more than 6 steps a second. To correct the effects of gravity it is necessary to use a high pass filter (Figure 12).

$$Given:$$

$$s \equiv sample, the\ accelerometer\ y - axis\ value$$

$$f \equiv cutoff\ frequency$$

$$i \equiv interval, time\ between\ samples$$

$$v \equiv previous\ filtered\ value$$

$$w \equiv previous\ unfiltered\ value$$

$$\tau = \frac{1}{2\pi f}$$

$$\alpha = \frac{i}{\tau + \alpha}$$

$$v = (\alpha * v) + \alpha * (s - w)$$

$$w = s$$

$$return\ v$$

With the assumption that the force of gravity is constant, a cutoff frequency of 1 Hz is chosen for the high pass filter in order to account for any extra noise.

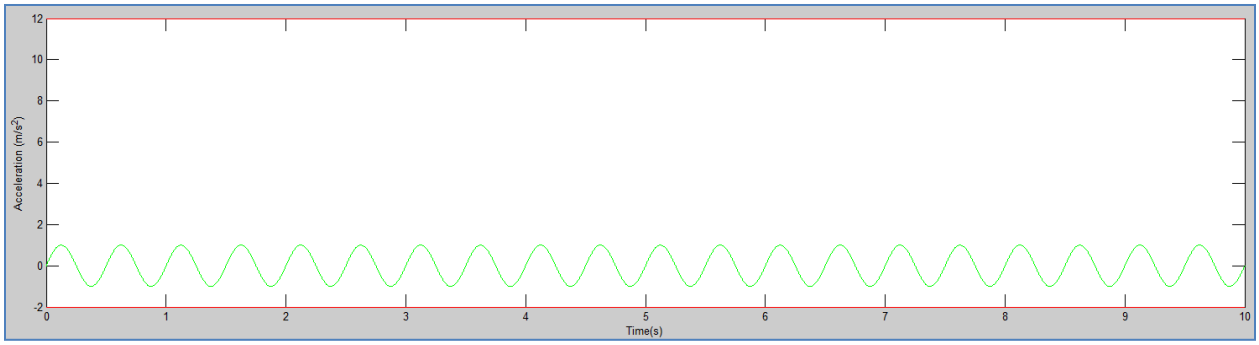

Figure 9 - Clean Original Signal

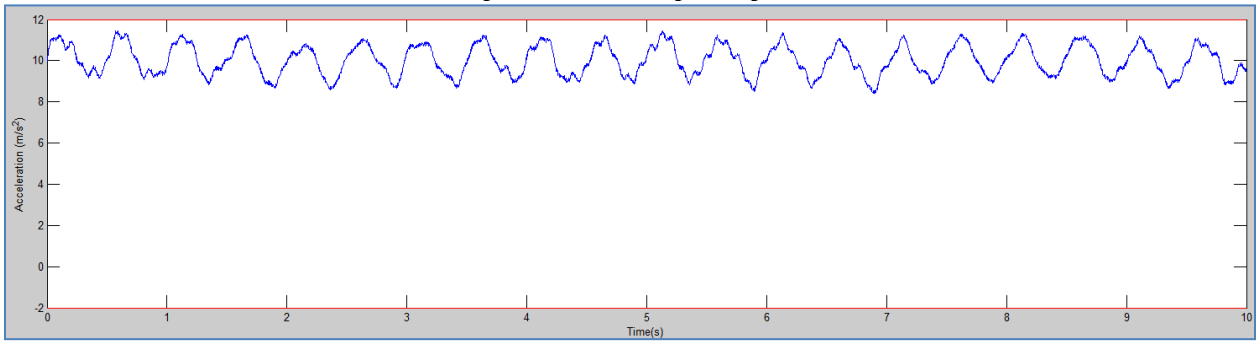

Figure 10 - Device Acceleration

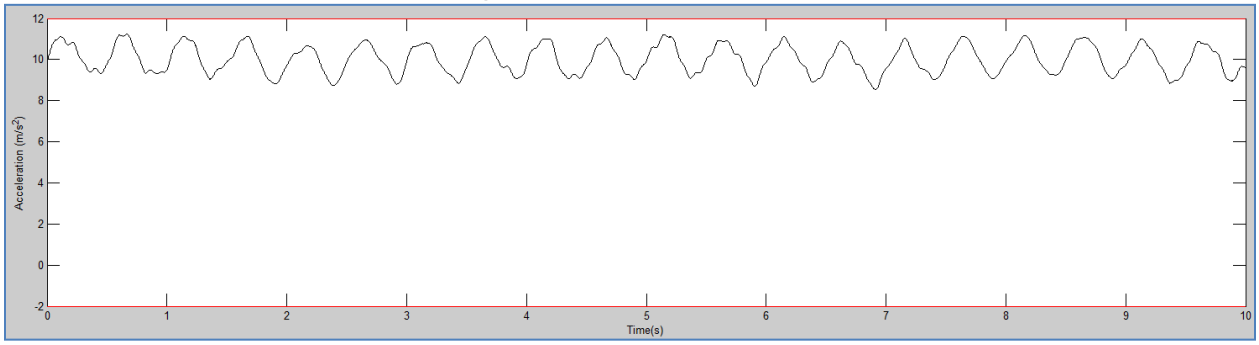

Figure 11 - First Order Low Passed Signal

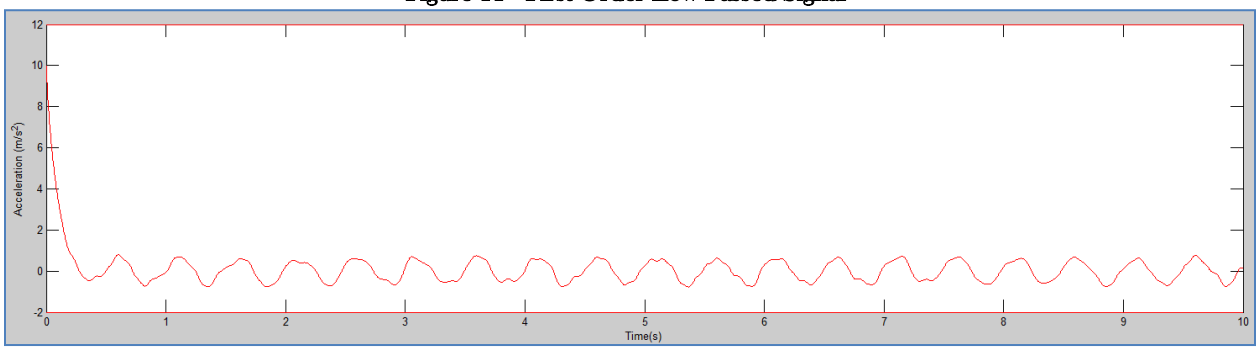

Figure 12 - High Passed Signal

4.1.6 Step Counter

Designing the perfect step counter requires more than electronic experience. Study is needed into human anatomy, physiology and psychology, as well as dynamics, physical mechanics, and signal analysis. The goal of this thesis is not to design the perfect step counter but to build a step counter from existing architecture capable of providing an acceptable model for use with wireless AP algorithms presented. As shown in (Zhao, 2010) the ideal shape of a human being's stride is a sine wave leading to an approach featuring peak-detection. The step counting algorithm is designed such that a state change would occur only when a certain acceleration threshold was reached as shown in Figure 13.

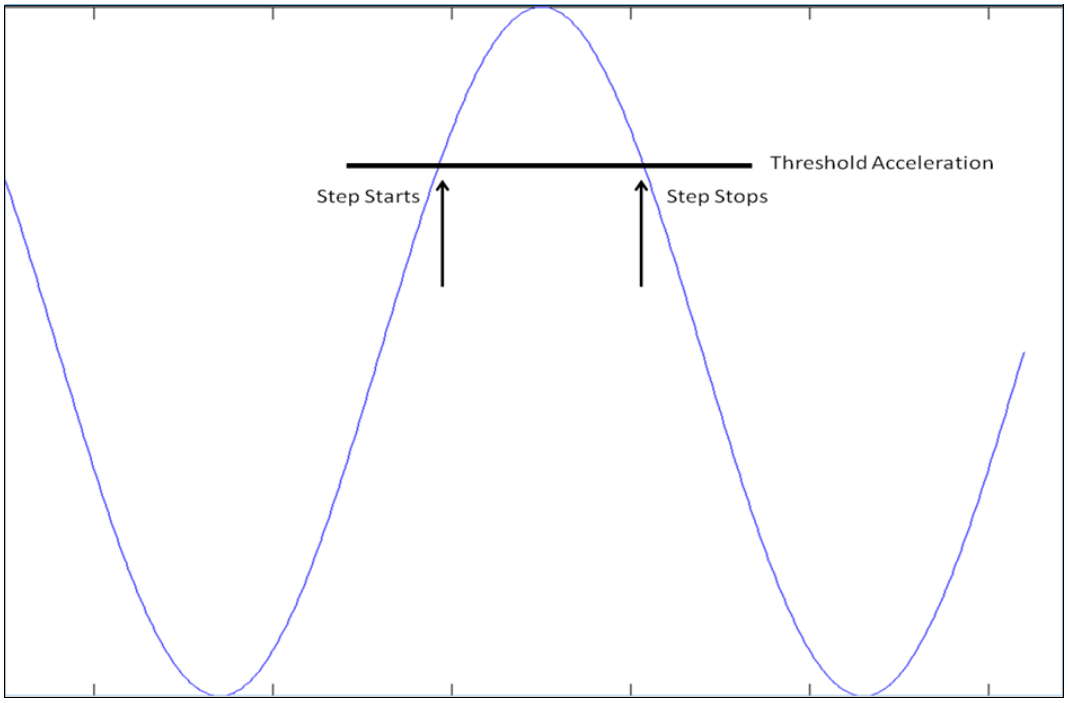

Figure 13 - Step Detection

Initially the algorithm begins in an idle state. An acceleration reading above a certain threshold results in a step being recorded and the state changing to acknowledge the step in progress. Once the acceleration falls back below the threshold the state resets to idle in preparation for the next step. A graphical view of the process is shown in Figure 14.

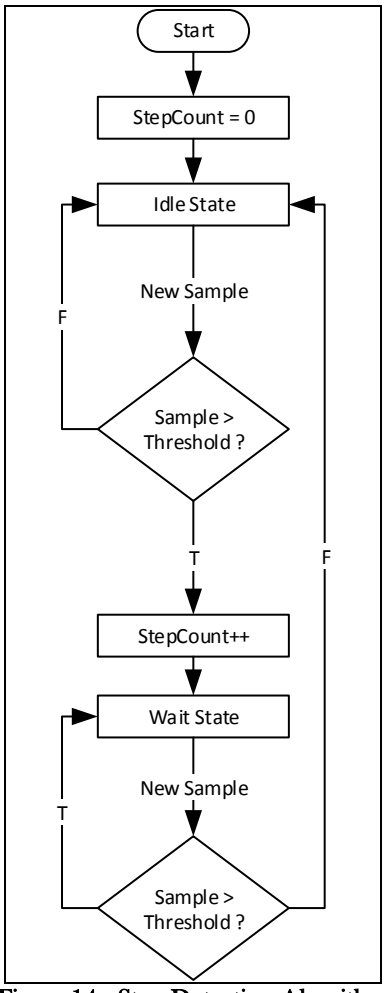

Figure 14 - Step Detection Algorithm

A weakness of this algorithm is that every person has a unique threshold at which they trigger a step event. In order to account for this, a calibration stage is added to the device. To calibrate, a user takes ten steps when instructed and then informs the program of their completion. The program creates a number of step counters each with its own unique threshold value. As the steps

occur, each of these counters record what they believe to be the correct step count. When the user indicates that they have completed the prescribed ten steps the program queries the step counters and deletes any that are outside a certain range around ten steps. For example, those step counters that counted nine, ten or eleven steps might be kept while any others would be removed. After this calibration step the remaining counters would be averaged to determine any subsequent step measurements. The resulting step counter then required a few common filtering techniques to minimize the generation of false steps. These techniques have been left out of this thesis as they do not contribute any new or noteworthy information. The overall result is a step counter with a fairly high accuracy in steps recorded (in the order of +/- 2 steps out of 500 in straight lines) and with limited drift in orientation over time.

4.2 Half -Planes

One of the major drawbacks to using the step counter for positioning is its tendency to record false steps. Through use of wireless internet access points this behavior can be minimized. Attempts to use the RSSI of wireless internet APs is thwarted by the inconsistent nature of wireless internet signals. In order to avoid the pitfalls associated with varying signal strength this thesis attempts to determine a perceived AP location instead of the literal. This is done using a mathematical construct known as negative half space and its respective half-plane. Using these constructs, bounding data is generated each step for each AP in range of the device. Based on the direction of travel half-planes are aligned and used to form a convex hull. Before a convex hull can be formed, redundant half-planes must first be removed using a method put forward by Shamos (Shamos, 1976) and later summarized succinctly by Brown (Brown, 1978). During this process the half-hulls are generated and require fusing to finally reform into a single convex hull. Throughout

the process steps must be taken to ensure that each half-plane measured agrees with other half-planes connected to the AP. When the convex hulls are defined, pertinent averaging is applied based on the situation to construct a perceived location of each AP. The locations are then used to validate future steps. By applying geometric constructs it is possible to minimize the step counter's ability to record false steps.

4.2.1 Wireless Internet Signals

Using wireless internet signals as a means to determine position appears a simple matter. If the assumption can be made that APs have constant signal strength over time and a fixed position then simple triangulation can be used to determine current position. However, as Figure 15 below shows, even at relatively small distances AP signal strength is not at all constant over time.

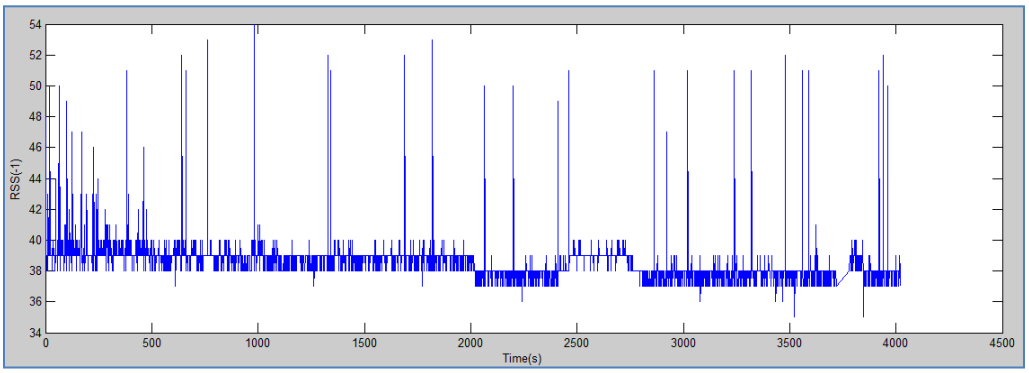

Figure 15 - Wireless Internet RSSI Over Time at 6 ft

The direct effect is that, rather than modeling a router as a sphere, additional information about the surroundings must be taken into account in order to correct the shape into something usable as shown in Figure 16a. Rather than attempting to define a corrected area, this thesis argues that a more effective strategy would be to bound the area and identify a perceived AP location for use in calculations (Figure 16b).

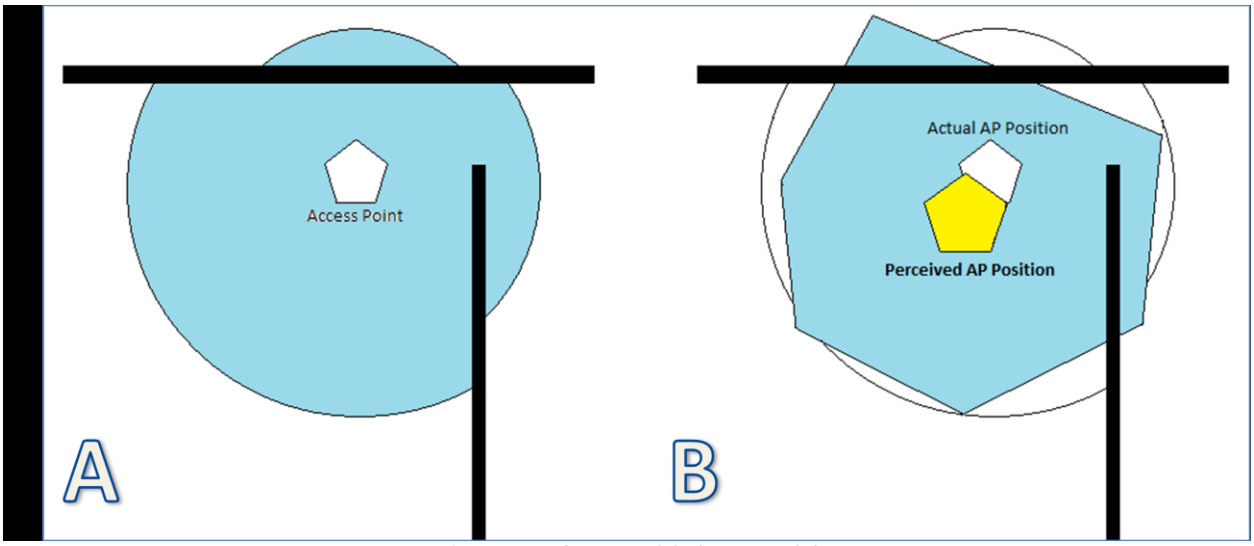

Figure 16 - Corrected Sphere model

4.2.2 Half-Planes and Negative Half Space

A three dimensional plane can be defined mathematically by a point lying on the plane and a normal vector pointing from the face of the plane. A plane can be thought of as a fence separating two half-spaces. The positive half space of a plane is any space lying in the direction of the normal vector while the negative half-space is any space lying "behind" the plane. Michael Shamos (Shamos, 1976), whose intersection algorithms are applied in this thesis refers to planes separating positive and negative half spaces as half-planes and for consistency the same term will be used. Half-spaces use the definition of a plane to quickly determine on which side of a plane a given point lies.

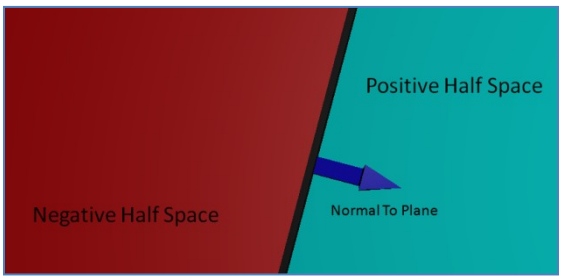

Figure 17 - Planar Half Space

To determine in which half-space a point lies consider the following:

$$\vec{P} = (P_x, P_y, P_z) \equiv Point\ on\ the\ Plane$$

$$\hat{n} = (n_x, n_y, n_z) \equiv Normal\ to\ the\ Plane$$

A point $\vec{X}$ is in the negative-half space of a plane if and only if $(\vec{X} - \vec{P}) \cdot \hat{n} > 0$

This is useful as it can be used to form the edges of a convex polygon. For example, in Figure 18 below a triangle in two dimensions is made up of three outward pointing planes and their respective intersecting negative-half spaces. To determine whether a point $\vec{X}$ lies within the triangle requires that each of the three planes be tested against the point $\vec{X}$ to determine whether the point is within its half-space. If for any plane the point is not within its half-space then the point does not lie within the triangle.

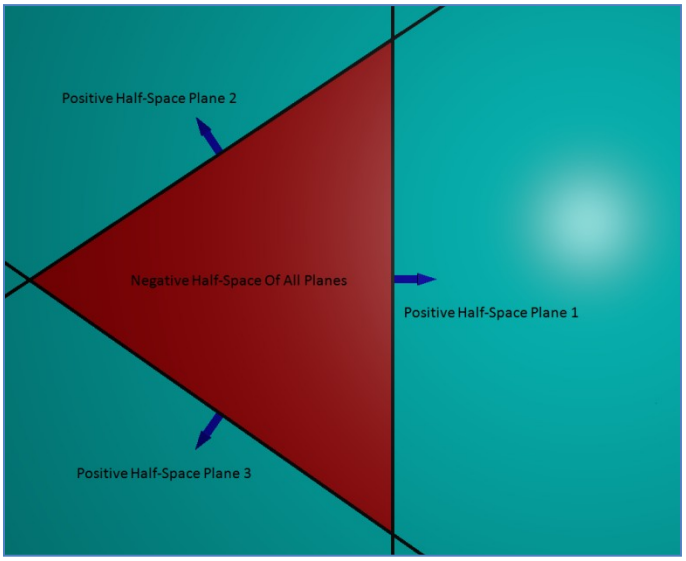

Figure 18 - Intersection of Half-Planes

4.2.3 Application of Half-Planes

The aim of this thesis is to use the concept of negative half-spaces to determine the perceived location of every router detected during movement. At every step a new half-plane is added based on both past user positions and the previous RSSI measurements from the APs themselves. Figure

19 below contains a simple building layout which will be used to demonstrate the general approach to the problem along with Figure 20 which shows the planned path of a user through the building.

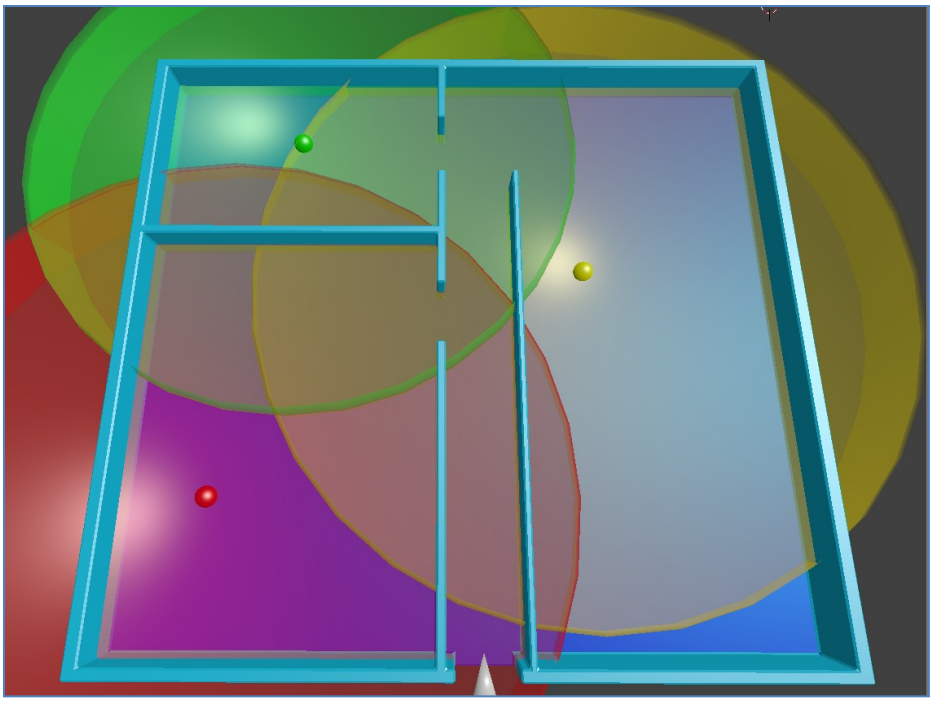

Figure 19 - Simple AP building layout

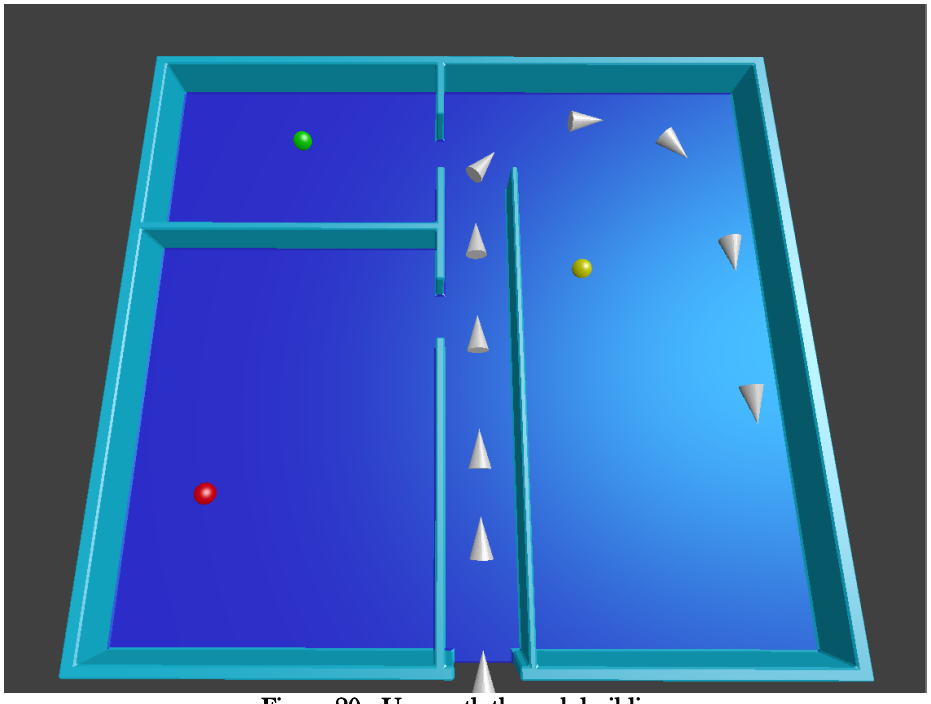

Figure 20 - User path through building

Beginning with the first step from the door as shown in Figure 21a, a half-plane is added, immediately reducing the bound of the lower z position of all three of the AP points from infinity to the end of the first step. As more steps are taken through the interior in Figure 21b-f additional half-planes are added further bounding the locations of the APs until in Figure 21g the user has reached the end of their path.

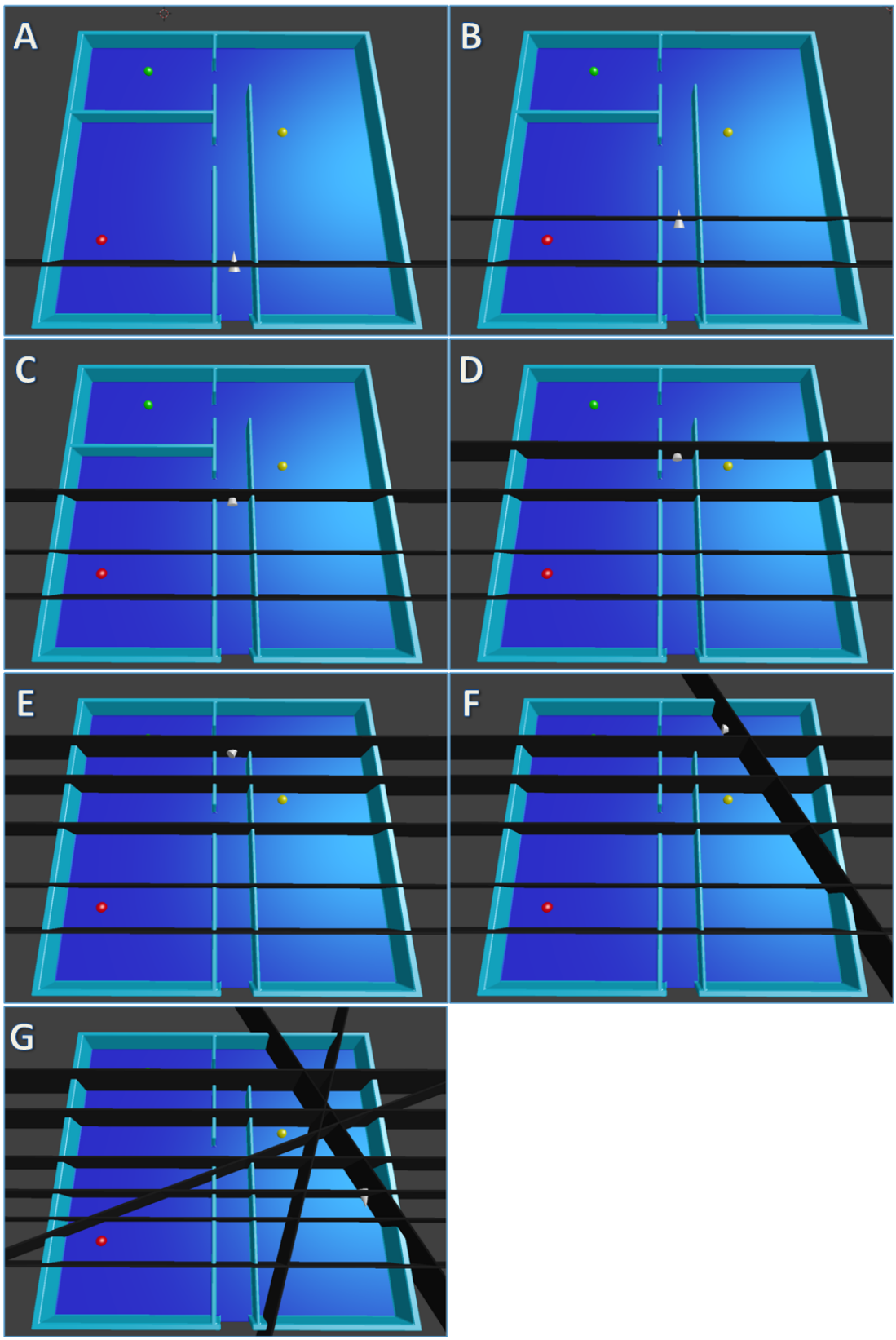

Figure 21 - Progression of a user through a simple building

As the user is progressing through the building the half-planes generated are being intersected and combined to form convex hulls surrounding each AP and Figure 22 shows the final disposition of the convex hulls.

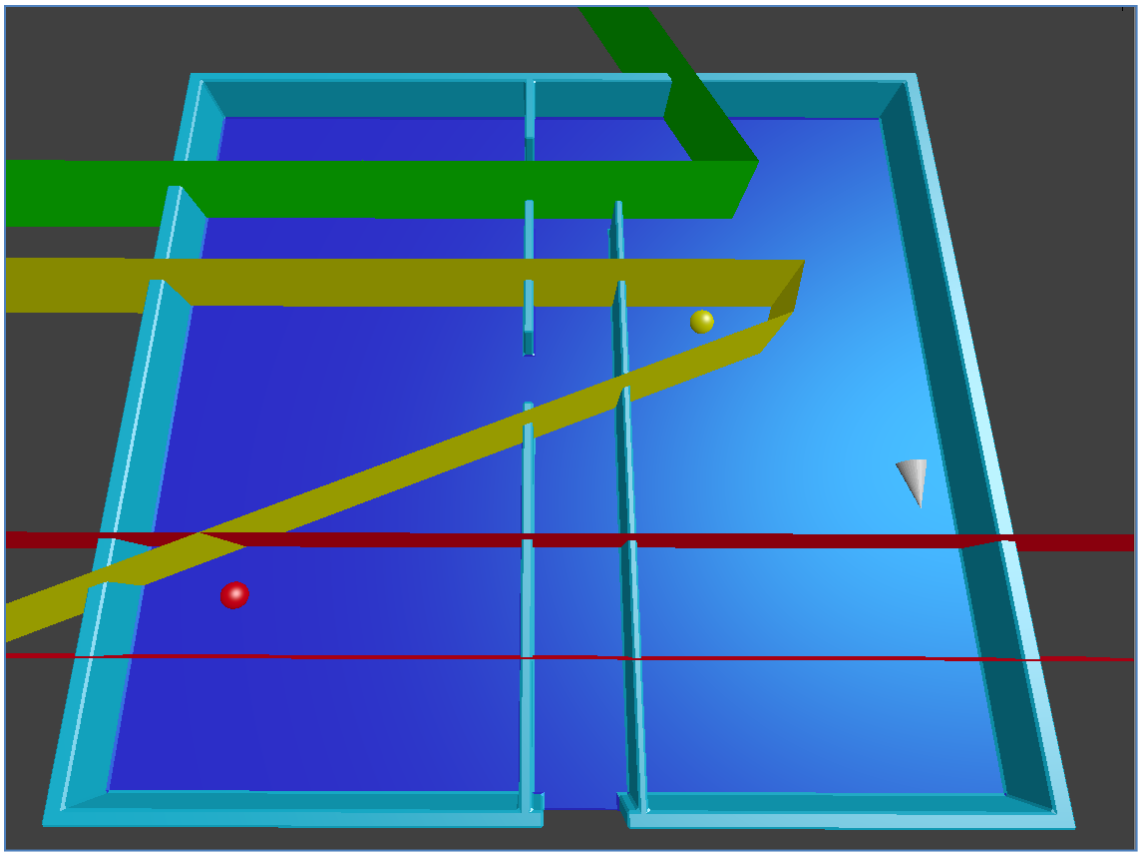

Figure 22 - Final bounding convex hulls

At first glance the resulting bounds are too unrestrictive to be of use. However, assume that on the way out of the building the user took one additional step off of the original path as shown in Figure 23 below. This one small change in direction has instantly transformed the previously unhelpful hulls into boundaries that are very accurate considering no prior data was known about the APs upon entering the building. The existence of this phenomena indicates that further

research may be helpful in determining what movement patterns, if any, result in improved AP position detection.

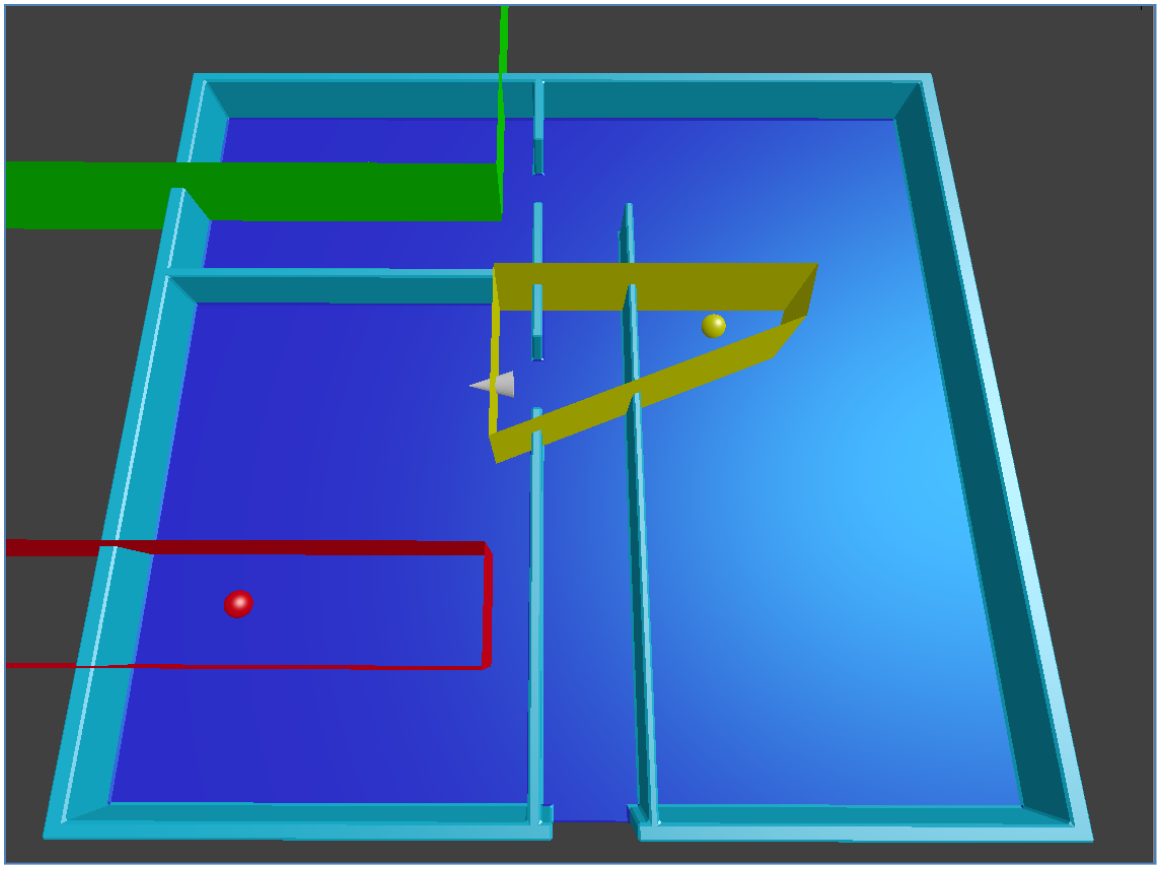

Figure 23 - Additional step

In fact, given almost any additional information about the surroundings radically increases the accuracy of the bounding hulls. Figure 24 below demonstrates the original situation from Figure 22 if it were possible to obtain a layout of the building interior.

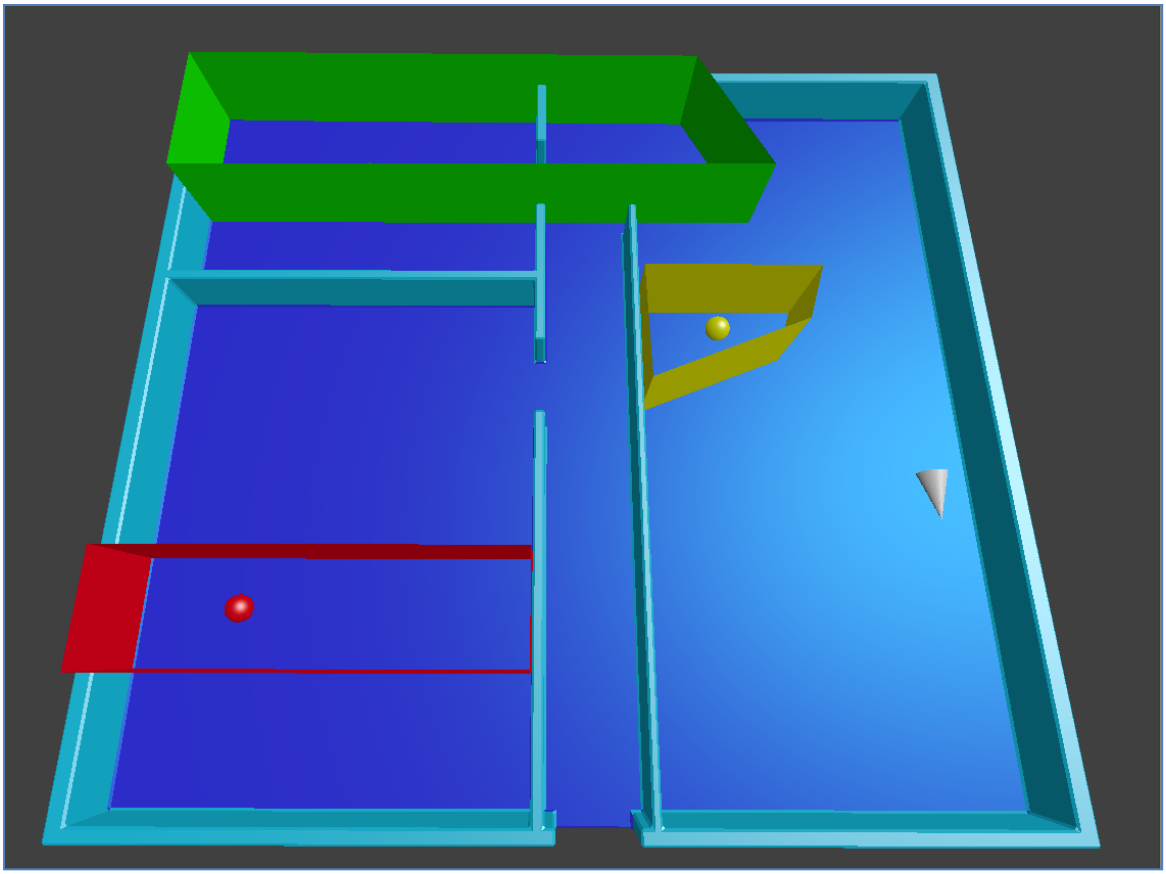

Figure 24 - Bounding hulls given prior building layout information

4.2.4 Determining Plane Normal Vectors

The application of half-spaces is fairly simple. At every new location the signal strength from the current point and the previous point is compared. Based upon the direction of travel and the change in strength a half-plane is added, bounding the position of each AP. This course of action necessitates two conditions for adding a new plane. A valid step must occur as must a change in signal strength. Without both, it is impossible to differentiate an error from a user simply walking around the outer edge of the AP. Given these conditions, the only remaining question is the direction of the normal vector of each half-plane. In the event of the router's signal strength increasing, the application responds by placing a half-plane pointing against the direction of travel.

This indicates that the AP is somewhere ahead of the user as shown in Figure 25. In contrast Figure 26 shows a decrease in signal strength. As a result a half-plane is placed pointing in the same direction as the direction of travel. This indicates that the router is now somewhere behind the user. Over time, the culmination of half spaces forms a stable outer bound for an AP's perceived location.

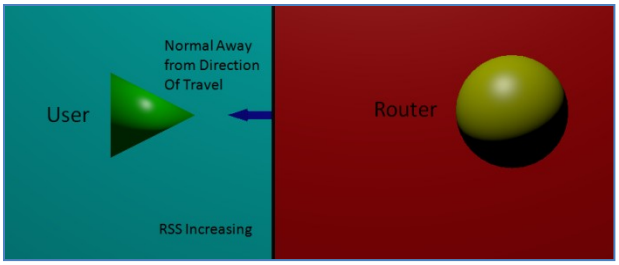

Figure 25 - Signal Strength Increases

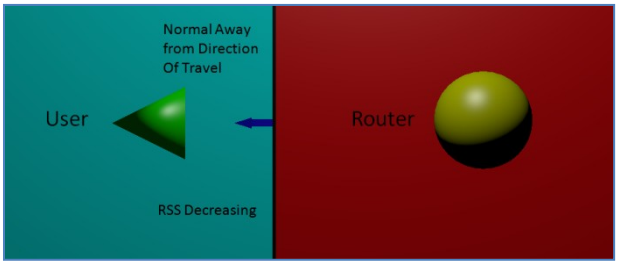

Figure 26 - Signal Strength Decreases

4.2.5 Redundant Half-Planes

Before the method above can be used, however, it bears considering the intersection. Given that every time a step is taken there is a high probability of at least one new half-space being added, in very short order there are far more planes than can easily be sifted through. To further complicate matters there are many cases in which the addition of a half-plane provides no new information and can be considered redundant. Figure 27 below shows a few of the possible redundancies.

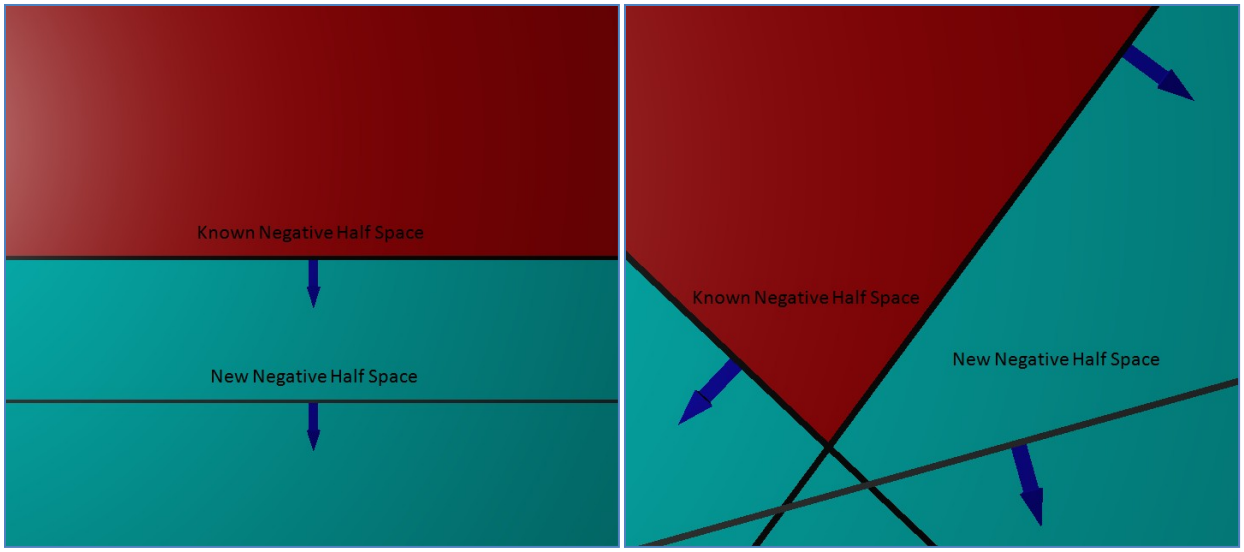

Figure 27 - Redundant Half Spaces

According to Brown, an efficient manner of determining redundant half-spaces (with the eventual goal of determining the intersection) begins by categorizing half-planes into two distinct groups. Brown arbitrarily selects a two dimensional axis direction to act as the up vector and from here defines a half-plane as either Upper or Lower. Upper half-planes are those whose boundary line is above the area bounded by the half-plane. Lower half-planes are those whose bounded area is above the line (Figure 28). In the case of this application the axis chosen as the "up" axis was the negative z axis in order to maintain the axis consistency of OpenGl for later rendering.

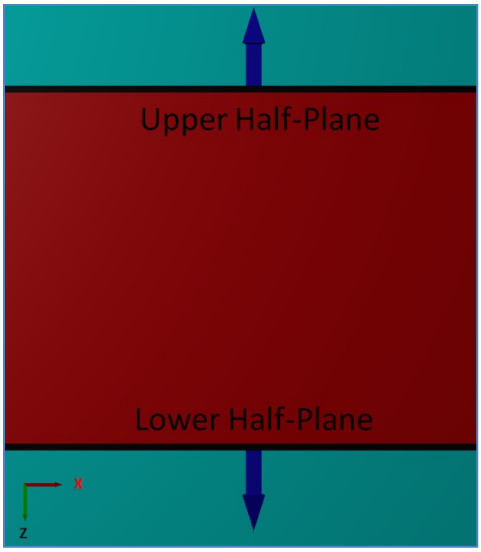

Figure 28 - Upper and Lower Half-Planes

Using a technique that Preparata and Muller (Preparata & Muller, 1979) refer to as "dualizing the half-spaces" it is possible to reduce the complexity of the $O(n^3)$ operation above. The final result of the dualization is that the problem of upper half-plane intersection reduces to simply finding the lower half of the convex hull formed by the dualized half-space points. Finding the convex hull of a set of n points can be done in $O(nlog(n))$ time through the use of a modified Graham Scan algorithm. Once the lower convex hull has been identified, any points that are not a part of the hull are discarded as redundant and the hull is transformed back into half-planes. The process is then repeated for the set of lower half-planes.

4.2.6 Determining the Convex Hull

Once dualized, Brown demonstrates how non-redundant half-planes meeting the conditions above form either an upper or a lower convex hull. In general a convex hull of a set of points is the smallest envelope formed by sequential points such that any point within the envelope can be connected to any other by a straight line that does not cross the envelope. Consequently an upper

convex half-hull consists of any points forming the convex hull in which the area contained by the hull does not appear above and lower those for which the area is not below. In the case of UPPER half-planes the points forming the LOWER convex hull were non-redundant while the reverse was true of the lower half-planes. Graham Scan, a fairly well known algorithm, was applied in order to determine the convex hull of a set of points. In Graham Scan the first step is to determine the root - the point with the lowest y value or the lowest x-y pair if multiple values containing the lowest y are present. Each point is compared to the root and the angle formed between point, root, and the x-axis is calculated. The points are then sorted in ascending order based on their angles. The sorted values are traversed in order and at each point a test is performed to determine whether the current three points in question form a right turn or a left turn. Any three points that form a right turn result in the second of the three points being removed from the hull. The final result is an algorithm which quickly (in $O(nlog(n))$ time due to sorting) computes the convex hull described by a set of points.

Brown's next step is to remove the redundant points and their respective half-planes by keeping only those segments of the convex hull that form the upper (if considering lower half-planes) or lower edge of the convex hull depending on the half-planes begin considered. With the assumption that the convex hulls are sorted in counter-clockwise order, finding the upper half-hull becomes a matter of finding the point farthest left and the point farthest to the right. Then, starting at the left most point, proceed clockwise around the edge of the half-plane adding points until the right-most point is reached. Similarly the lower half-hull can be found by traversing the convex hull in a counter-clockwise manner. Having isolated the non-redundant half-planes, all that remains is to dualize the points back into their respective half-planes and perform a simple line-line intersection algorithm on the resulting segments to form the corresponding half-hull in planar

space. An important step that seems to be implied by Brown's algorithm but is not clearly stated is the need to extend the edge segments in order to facilitate later fusing of the two half hulls. To this end, the points at either end were allowed to go to infinity.

The result of the redundancy test is two sets of ordered half-planes forming the upper and lower halves of a single convex hull. In order to treat the AP as a single body it was then necessary to fuse the two halves. Fusing the upper and lower half-hulls into a cohesive convex hull uses a procedure similar to that found in the merge associated with the merge-sort algorithm and so can be done in $O(nlog(n))$ time. Any difficulty arising from the process is due more to identification and subsequent processing of the edge cases.

4.2.7 Handling Invalid Access Point Data

A key weakness in this half-space algorithm is the tendency for APs to lie. Many conditions prompt invalid data such as movement directly adjacent to the actual location of the AP in which one half-space is placed to one side and another half space is placed on the other. In fact testing shows that apart from passing an AP the only other main source of error in this algorithm occurs when the path of travel is changed abruptly. The reason that invalid values are a problem is that they force a separate hull to form. This new hull is either not associated with the current hull, as it cannot be reliably intersected, or worse pulls the hull into a strange and completely invalid shape. In an attempt to reduce the number of errors, two filtering measures were added. The first such measure involves the averaging of data. Firstly, note that for a particular step many RSSI measurements may be taken and from these measurements multiple half-spaces (often in disagreement) may be added. Regardless of position, every half-plane added will have a normal vector facing in one of only two directions: towards the direction of travel or away from the

direction of travel. The averaging technique used compares the number of half-spaces with normal vectors towards the direction of travel to those in the opposite direction and adds a single new half-plane per step with its normal vector in agreement with the majority. The second filtering technique relies on the fact that while there may be the occasional misinterpreted step, but there will very rarely be two. Instead of relying only on the last known location to determine direction of travel, the second to last location was also considered. If the direction of travel for both the last step and the second to last step were within a certain angle of one another then the resulting half-space was considered valid. To complete the filter the average of the half-planes was derived as described previously and a new half-space was added.

4.2.8 Using Convex Hulls To Determine Position

After establishing a perceived location for each AP it is then possible to revert back to the spherical model in order to determine location. Conceptually, each AP is assigned a radius equal to the distance between the AP and current location. Furthermore a radius of movement is added around the user to establish a maximum possible range. This range serves as an outer bound to the movement shape constructed from the overlapping spheres projected by the APs as seen in Figure 29 below.

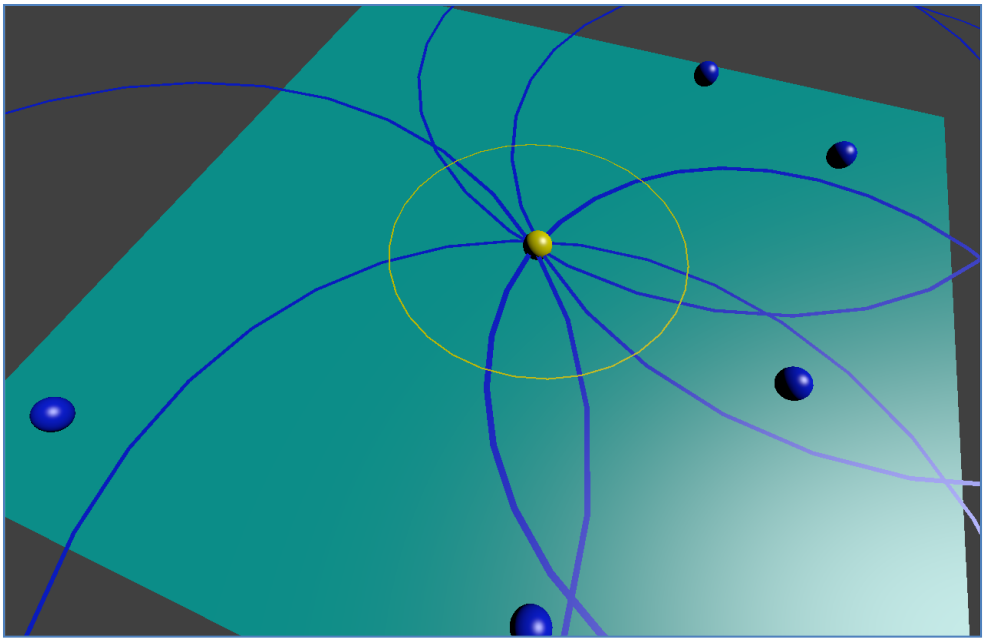

Figure 29 - Spherical Model (AP = blue, User = yellow)

When a user moves to a new location, each AP with a known, valid position is queried to determine the change in strength. Of the APs queried, all of those APs who report an increase in signal strength - meaning a change in position towards them - are considered relevant for continued processing. All circles associated with these relevant APs are then intersected with each other and with the predefined circle defined by the radius of travel. Intersection points are considered valid if they lie either within or on the boundary of a relevant AP's circle and upon the central circle of travel. Due to the nature of the intersections the result of these operations will be a set of two points which, when combined with the current position of the user form a triangle loosely describing the possible new position of the user. An example of such a situation is shown below in Figure 30 where the green portion represents the possible area in which the user must be located based on the signal strength.

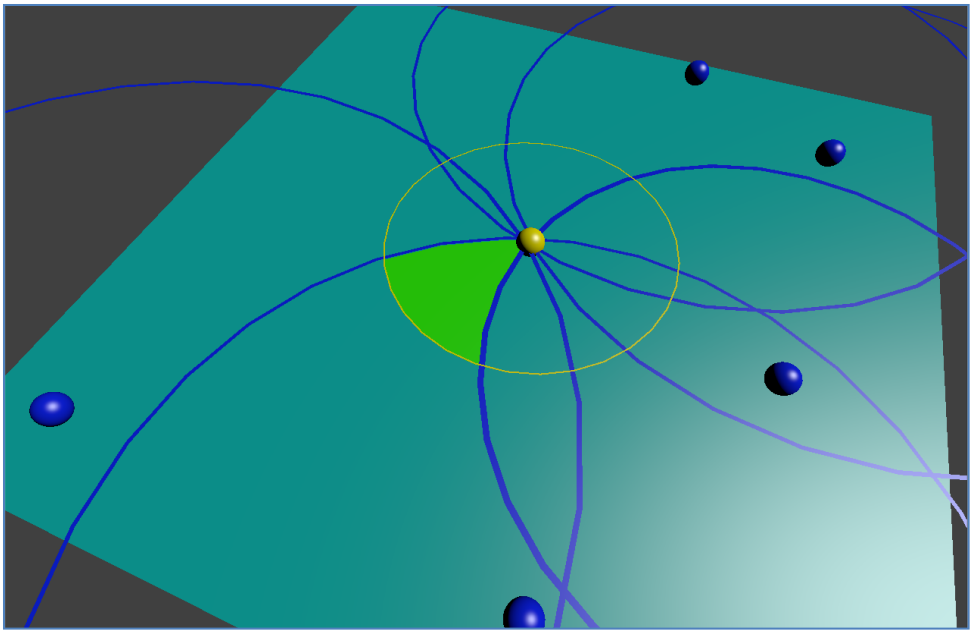

Figure 30 - Shape of Travel

At this point the computations for AP positioning for this step are complete and the application uses this as a verification for any step counters claims. Ideally the next step to be taken would do so not with a circle of travel but with a scaled version of the shape of travel projected outward to include the possible area. However, given the limitations and complexity of curve scaling, an approximation was determined as a substitute. Once the shape of travel has been determined for the current step, a circumcircle is circumscribed through the triangle's three vertices and is then scaled by the maximum possible step range defined above. This forms the new circle of travel (see Figure 31). From there the algorithm resets in preparation for the next step.

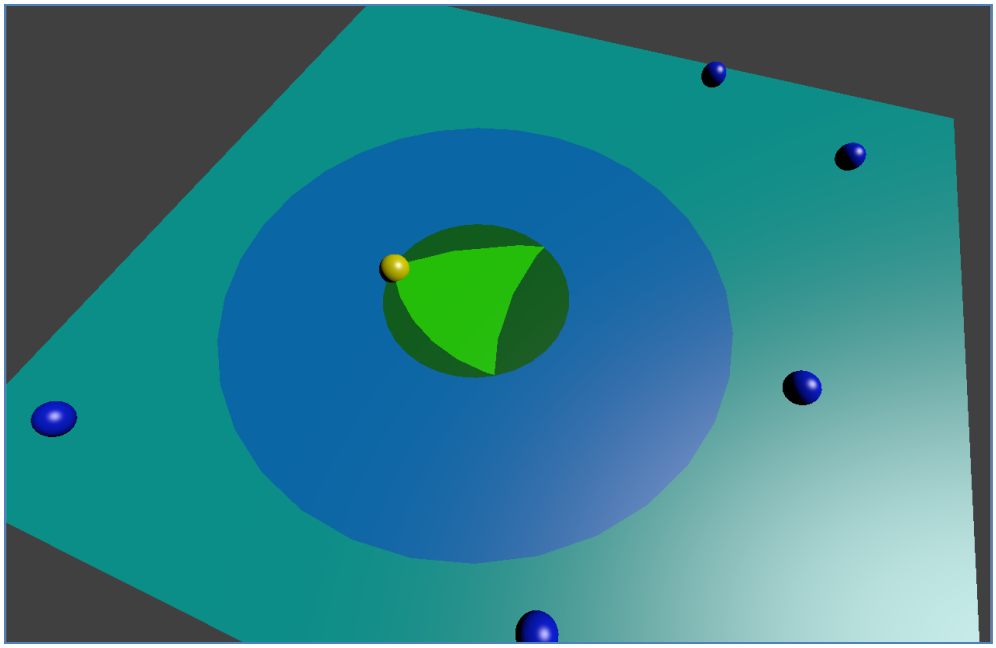

Figure 31 - New Circle of Travel

An apparent weakness of this method is the ever increasing radius of travel as shown in Figure 32 A-F. However as Figure 32G shows, there are cases in which the possible region shrinks dramatically which should be sufficient to prevent an unmanageable increase in bounds.

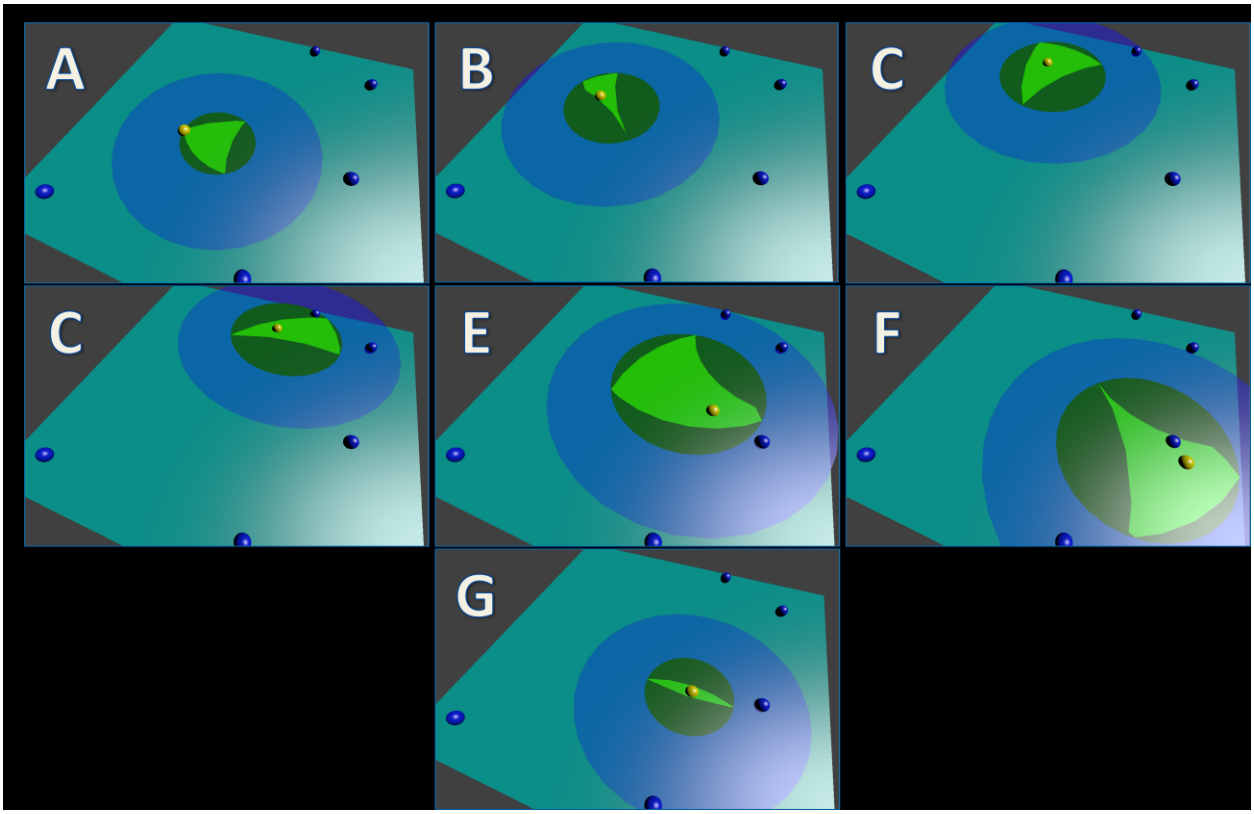

Figure 32 - Progression of Shape of Travel over several sequential steps

Even with the many flaws in using wireless internet for positioning it can still be a valuable tool. By using half-planes defined during travel, individual APs can be bounded over time until they form an effective position regardless of their actual position. Getting to the point where it is possible to evaluate the convex hulls formed by the half-planes is not computationally cheap. By splitting the half-planes into half-hulls and then fusing them back together the computational complexity is reduced to a manageable level. Throughout the process care is taken to ensure valid half-planes and the resulting convex hulls can then be used to determine position from the changing RSSI induced through movement.

4.3 Visualization

No matter how great the algorithm, unless the information it provides can be used and understood, it's of little use. In the case of this thesis a user had to be able to understand where they were in relation to where they started. Initial visualization began by utilizing the forms commonly available to the android programming environment in the form of simple textboxes, buttons, and sliders. Over time more complexity was required and work began on the 2D canvas with limited success. Despite the inherent 2D environment, many situations are much simpler to understand and debug from a 3D environment using OpenGl. Progress into 3D graphics was slow but worthwhile as it offered a few unforeseen methods of displaying the limited visible data. Each display approach is extremely useful in a different area of application. Detailed explanations of rendering techniques used are left out of this thesis for the sake of relevancy.

5 Evaluation

Summary evaluation of such a vast interlocking system requires a few considerations. Having chosen to implement the application on an Android device forced a few early restrictions onto the testing approach. In the first case the device space of a handheld device was found to be fairly restrictive, forcing any testing procedures performed to either remove the test files after completion or perform a compression step to minimize the footprint. Another device-imposed challenge was the Java garbage collector. While the amount of time it required to operate was minute it was certainly large enough to interrupt sampling, especially in the early stages of development. A third limitation on testing was the finite battery life of the device coupled to the incredibly high power consumption required by the array of sensors. Presiding over all other concerns, however, was the final matter of complexity. With so many separate and complex subsystems contributing to the final product, care was taken to isolate each part ensuring that any errors generated were contained in small, manageable pieces.

As a result of the restrictions in testing the general approach was to divide the project into its component pieces and on each piece perform the same series of tests. Initial tests of each system required an on-device proof of concept test in order to minimally demonstrate that each component was able to perform its given tasks. During the course of the tests explicit effort was made to prevent computation from occurring on the device itself. Reason dictated that higher reliability of data in the early stages was of greater concern than efficient real-time device processing which may inversely affect incoming data. Once proof of concept was complete the device was temporarily removed from the picture and further testing for refinement of the algorithms was performed in simulated environments. At regular intervals throughout the simulation process, attention would return to the device and the code base would be transferred in order to test the

progress in a real-time context. Periodical testing in this manner served to confirm any assumptions made as well as to fine tune general parameters for use in the target platform. It also functioned as a means to reveal unforeseen conditions in which the simulations had not yet been designed to model. The iterative approach to testing minimized many of the problems related to the android device while still exposing testers to real world edge cases and unhandled circumstances. Merging of the pieces into a final product was left out of this thesis as the process is largely one of repetitive debugging - ensuring that each system performs identically before and after connection, followed by gradual replacement of simulation components with their real counterparts.

5.1 Step Counter

As the first component tested, the step counter represents the weakest part of the overall system design. Sensor testing is extremely time consuming and requires incredible effort to produce reasonable results. Given the existence of other, more sophisticated approaches to step counting, effort was put into ensuring that the step counter was able to produce initial results with high accuracy while ignoring the need for accuracy over time. The figures below show the results of testing the step counter in Eaton Hall on the Lawrence campus of the University of Kansas. As expected the paths describing the motion through the hallways are generally smooth, although over time the sensors drift so that turning corners can produce strange results. As the wireless internet components exist to correct any errors within the step counter, the tradeoff of long term orientation accuracy is an acceptable price for higher accuracy data on long, straight stretches where the wireless internet techniques tend to struggle.

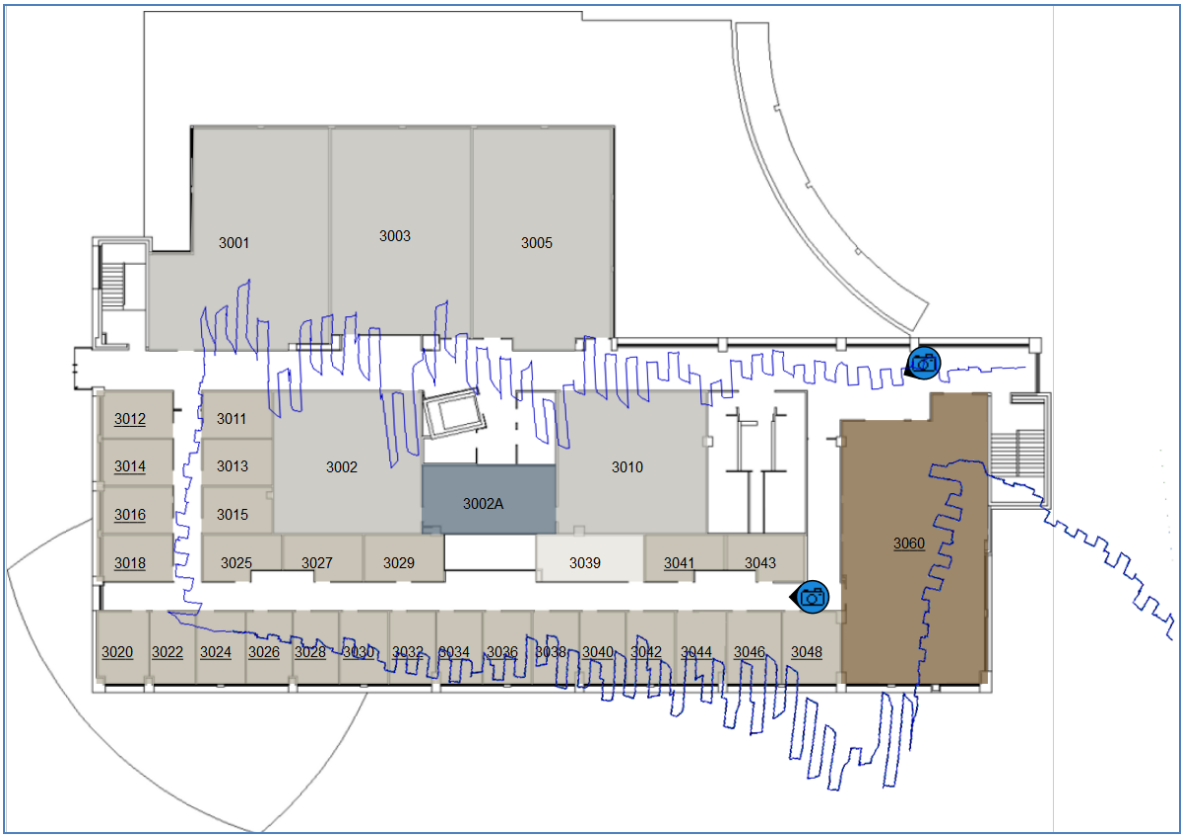

Figure 33 - Step Counter Testing In Eaton Hall

5.2 Router Location Determination

The proof of concept stage for the second system - router location determination - was a simple test involving a single router and a small area. If operation on a small, tightly controlled scale was successful it was deemed likely that a larger implementation would enjoy similar results. In order to isolate the system from the rest of the project, the step counter was disabled and distances were measured and marked by hand to guarantee that any discrepancies in the data would originate entirely from the location determination. The test required that the user stand at each point in sequence and push a button on the device to begin recording RSSI data. Once

sufficient samples were taken a second button stopped the recording and the user moved to the next step. Figure 34 below shows the initial setup of the proof of concept.

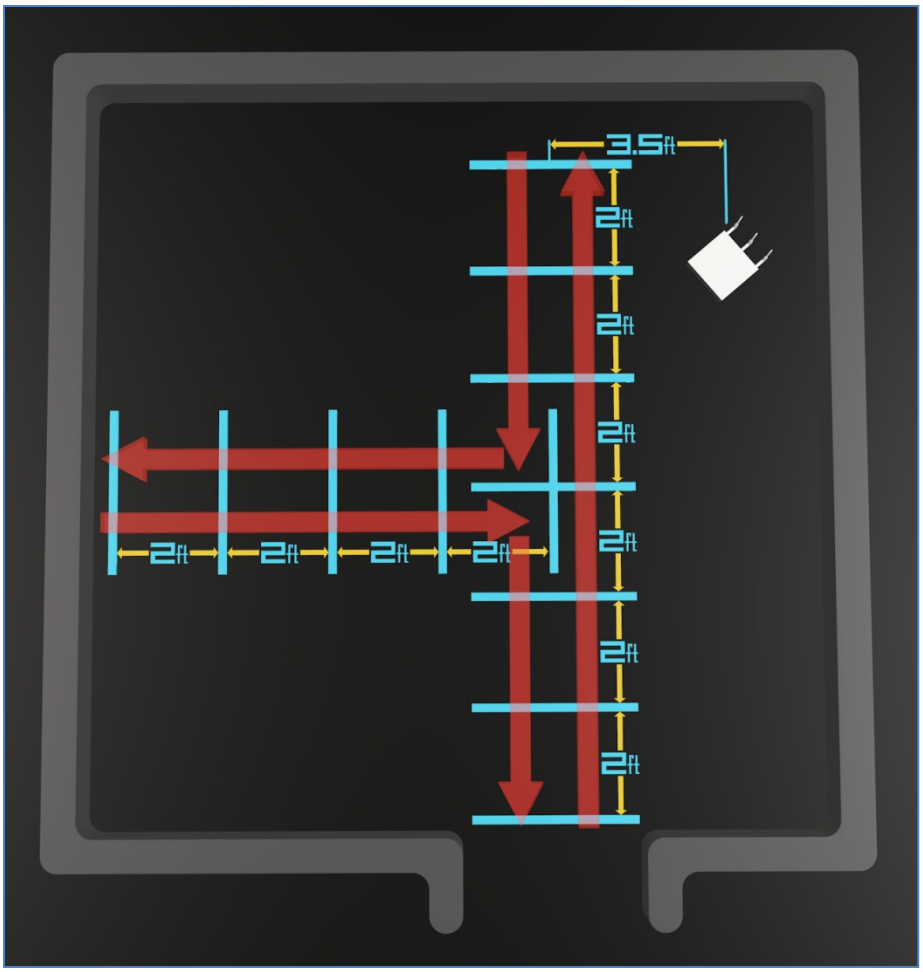

Figure 34 - Location Determination Proof of Concept Layout

Initial testing provided a cacophony of conflicting data and Figure 35A-C below demonstrate the application of three different filtering techniques on the half-plane data. In A the wifi data is simply averaged at each point and the resulting value used to determine the direction of the half-plane. Figure 35B demonstrates the aforementioned technique in which averaging is applied to the normal vectors of the planes. The last figure shows the results of applying directional averaging

after which the second to last step is used as a verification function on every current step. In each stage conflicting half planes are displayed in red and the possible area in which the router can exist is marked in yellow.

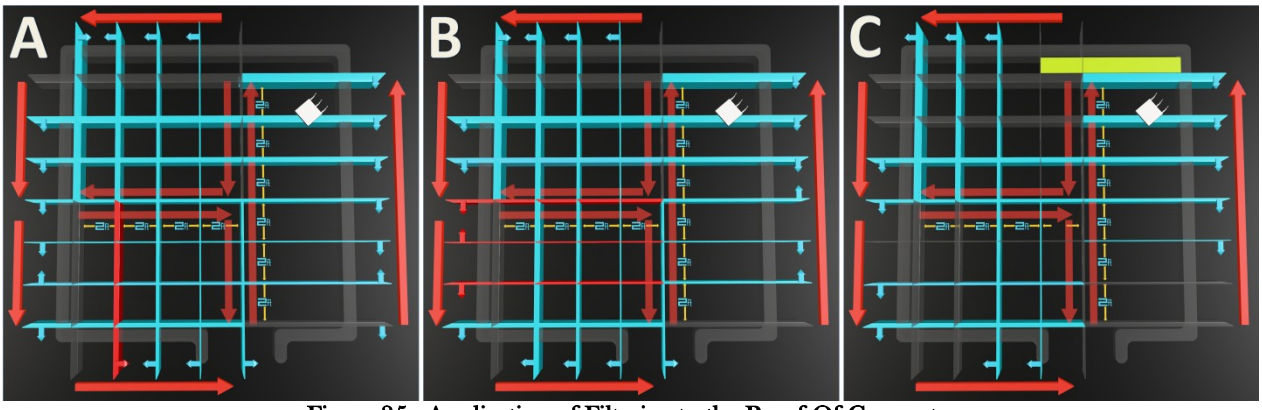

Figure 35 - Application of Filtering to the Proof Of Concept

Having managed to isolate the position of the router to the expected area with minimal conflicting cases, work began on simulating a more taxing environment for the algorithm to process. The goal of the initial simulation was to push forward development to incorporate convex hulls and multiple APs. A sequence of positions was generated and for each step the APs within the simulation were simply assumed to be relaying a RSSI equal to the distance between the AP and the user.

Figure 36 shows the results of the initial stage of simulation testing. During testing a number of observations were made. The first was that in ideal situations the algorithm is capable of making position estimations that fall within ten steps of the actual AP position as shown in R1,R2, and R4. A second observation is that the algorithm is highly path dependent. R3 was bound poorly by the half-planes as a simple reaction to the path's inability to provide it sufficient lower bounds on the z axis.

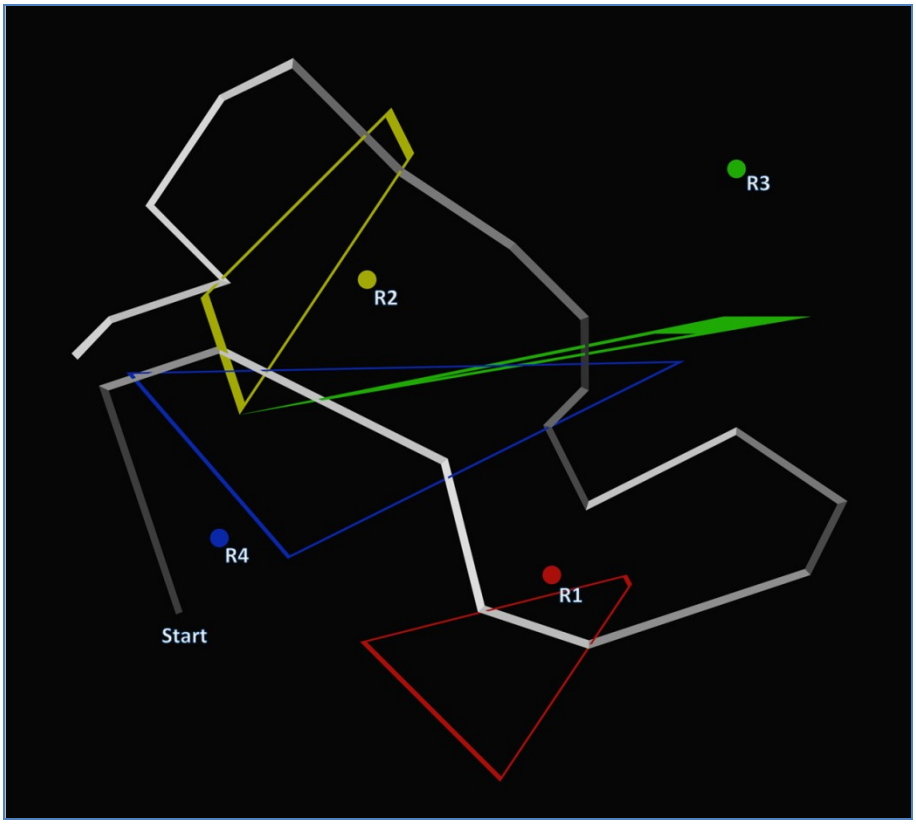

Figure 36 - Convex Hull Testing

A third test was performed in an attempt to both determine what comprises an effective path and what (if any) indications exist which may suggest to the algorithm that a given AP may be insufficiently bound. The results of the tests are shown below. The test involved generation of up to fifty steps which were taken at random orientation and distance amid a field of hundreds of active APs displayed as X's. In the leftmost graph the user's path through the APs is shown as a connected line beginning at the solid blue circle. In the second graph the data is augmented with visual attributes to assist in understanding the results of the algorithm. The size of the X's in the second frame represent the area contained within the convex hull bounding the given AP. The area, obtained through a quick triangulation of the hull and a subsequent application of Heron's formula provides one measure of how well each AP is bound by the half-planes. The thickness of

each X is relative to the error associated with the AP. The error is merely the measured distance between where the AP really was and where the algorithm determined it to be. As such Xs that are thicker and therefore bolder are considered to be poorly defined and represent a need for further refinement of the algorithm. The third graph compares the minimum RSSI obtained from each AP to its error as described above. In the third graph each AP is represented as a circle. Those circles that are filled have been confirmed as being fully bounded on all sides by their convex. Empty circles are ones in which one or more of the points forming the hull exist at infinity implying an unbounded hull.

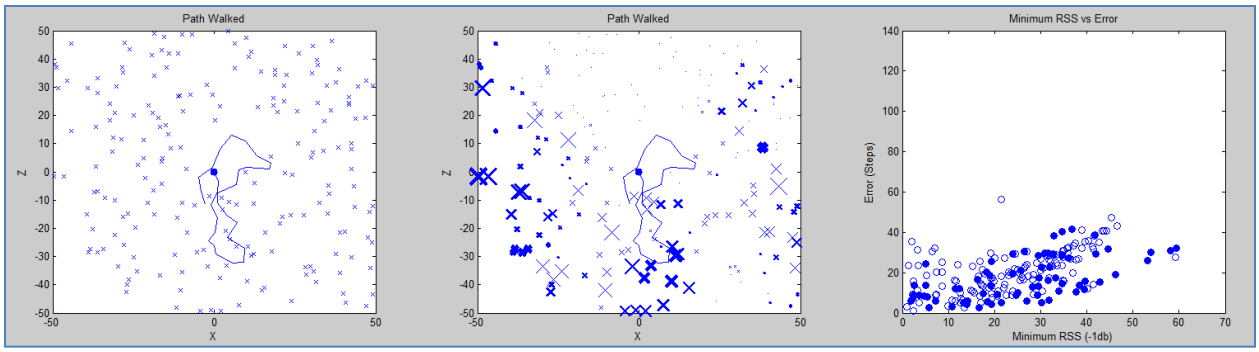

Figure 37 - Single Loop

.

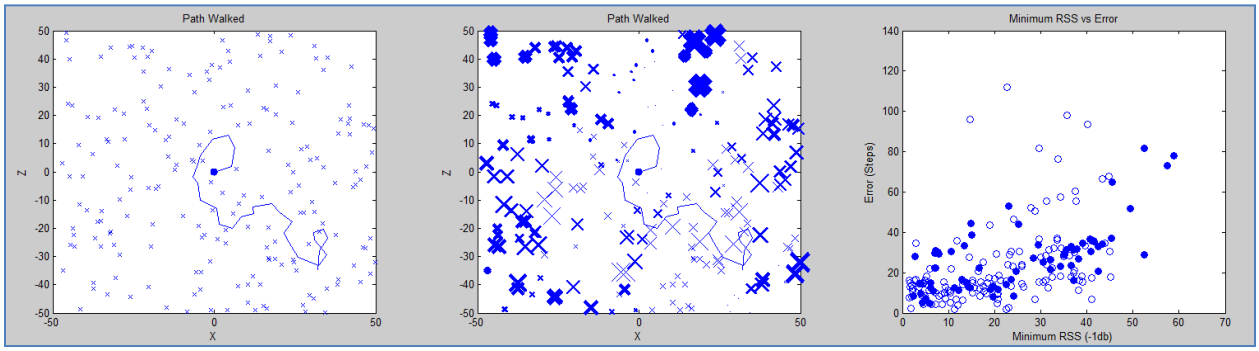

Figure 38 - Double Loop

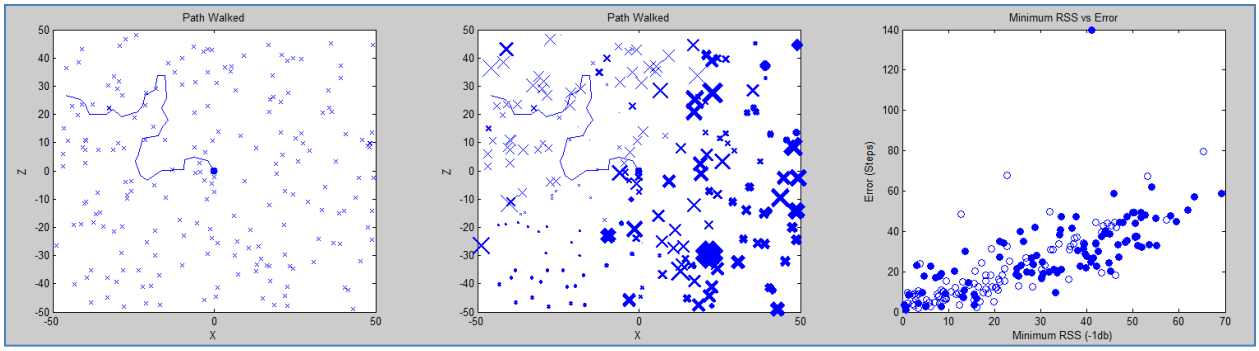

Figure 39 - No loop

The graphs above were selected to demonstrate a few phenomena observed during testing. The first such observation regards the occurrence of loops in the path of travel. Figure 37 and Figure 38 contain a single and a double loop respectively. Examination of the third graph in each case shows that even points very close to the path of travel have a tendency to drift from their real locations. In contrast, Figure 39 contains no loops and has comparatively well behaved AP values near their path of travel. Situations do exist in which paths containing no loops will occasionally produce invalid nearby values. Paths that contain loops, however, almost always produce invalid values.

Despite the outliers, graph 3 in each case demonstrates a pervading relationship between the nearness of the AP and its usefulness as a data point. Further work is needed to determine whether APs farther away can be improved. Yet even with given the requirement that an AP be within a certain RSSI range to be valid, the resulting positions of the routers have been established to within twenty steps from an initially unknown position.

The histograms shown in Figure 40 reinforce this finding by correlating the results of twenty simulations. Recall that, for simplicity, the simulation assumes the RSSI of an AP measured at the actual location of the AP is 0dBm. The top graphs describe the error present in access points, both fully and partially bounded with a minimum RSSI of -10 and -20 dBm respectively. The lower

figures describe the error present only in fully bounded access points also within the respective minimum -10 and -20 dBm.

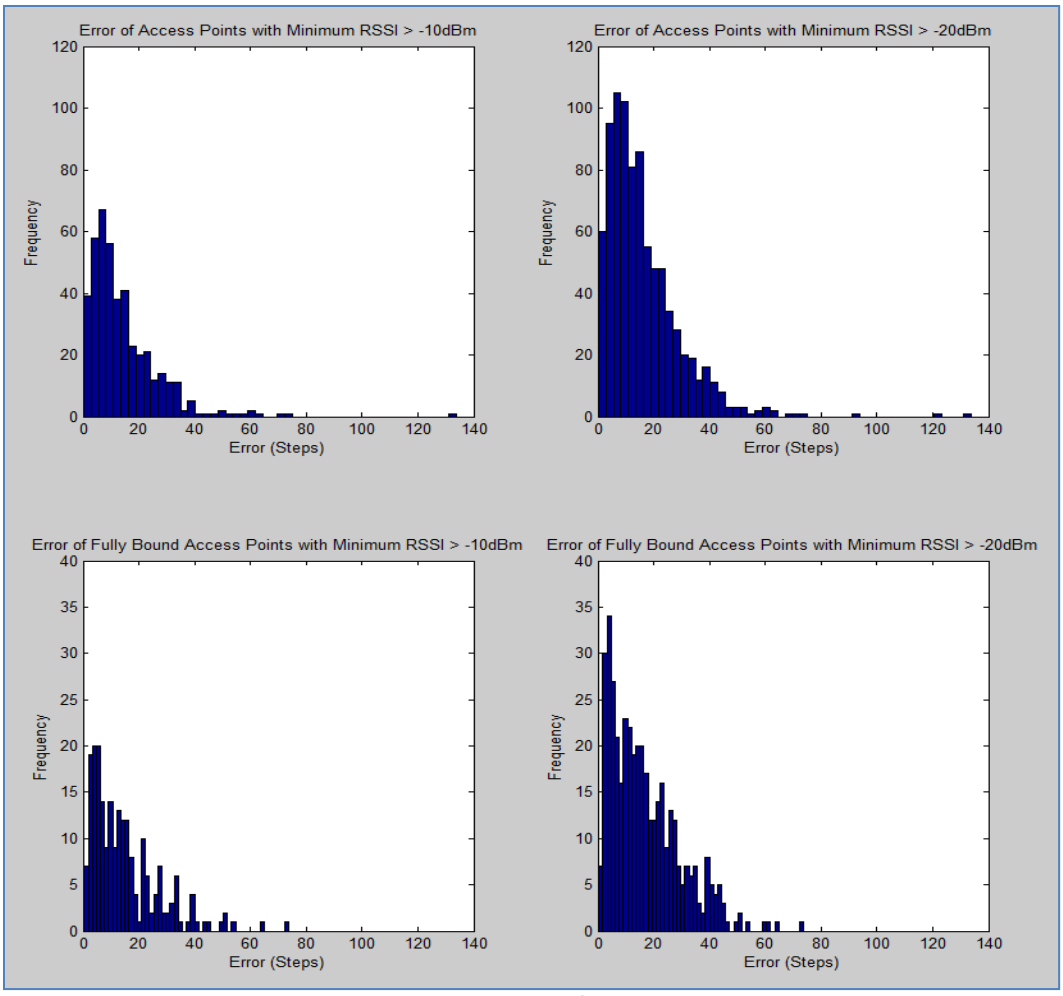

Figure 40 - Error of Access Points

5.3 Induced Position

The existence of techniques capable of determining user position from known AP positions meant that less focus was spent in inducing user position than was spent in determining the effective positions of the APs. That being said, care was taken to ensure that the approach described above was tested and proved viable, though isolating the subsystem was not possible given its complete

dependence on the processed data provided by the other systems. Figure 41 is generated by providing the AP positions and initial RSSI strengths to the verifier. Next the verifier component receives a list of RSSI strengths indicating the occurrence of a change in position. A series of points generated from the parametric equation of a circle centered at the origin are then sent to the verifier which determines whether they fall in the possible region of movement. Any points which fall within the region are then displayed proving that given router strengths and positions, an area can be described to bound a user's motion. Figure 42, in contrast, passes the verifier all the information of Figure 41 but instead of testing a point it simply requests the area enclosed by the next central circle. Continuing to repeat this process results in a path that very closely follows that actually taken by the user as shown below.

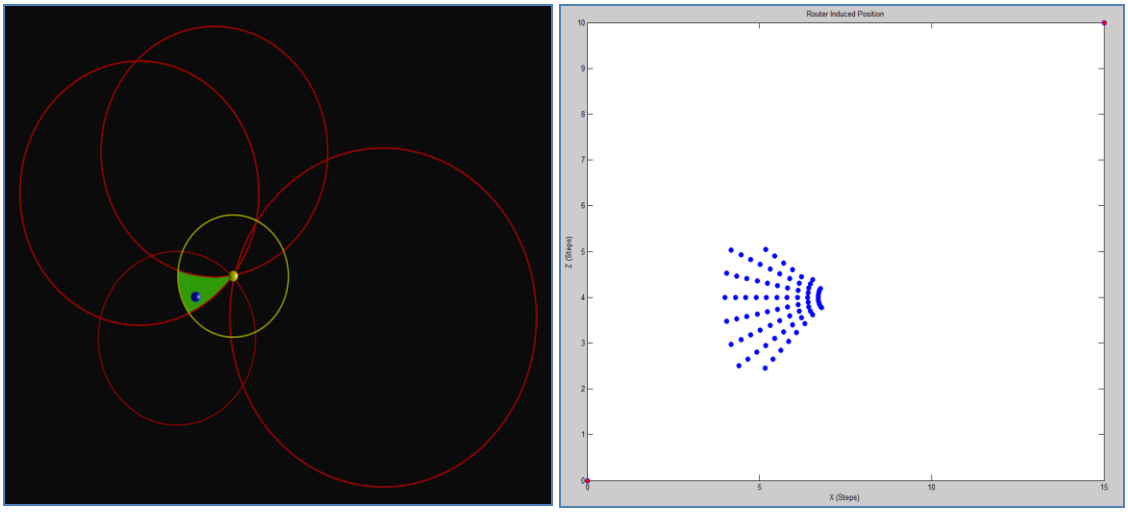

Figure 41 - Possible Movement Area

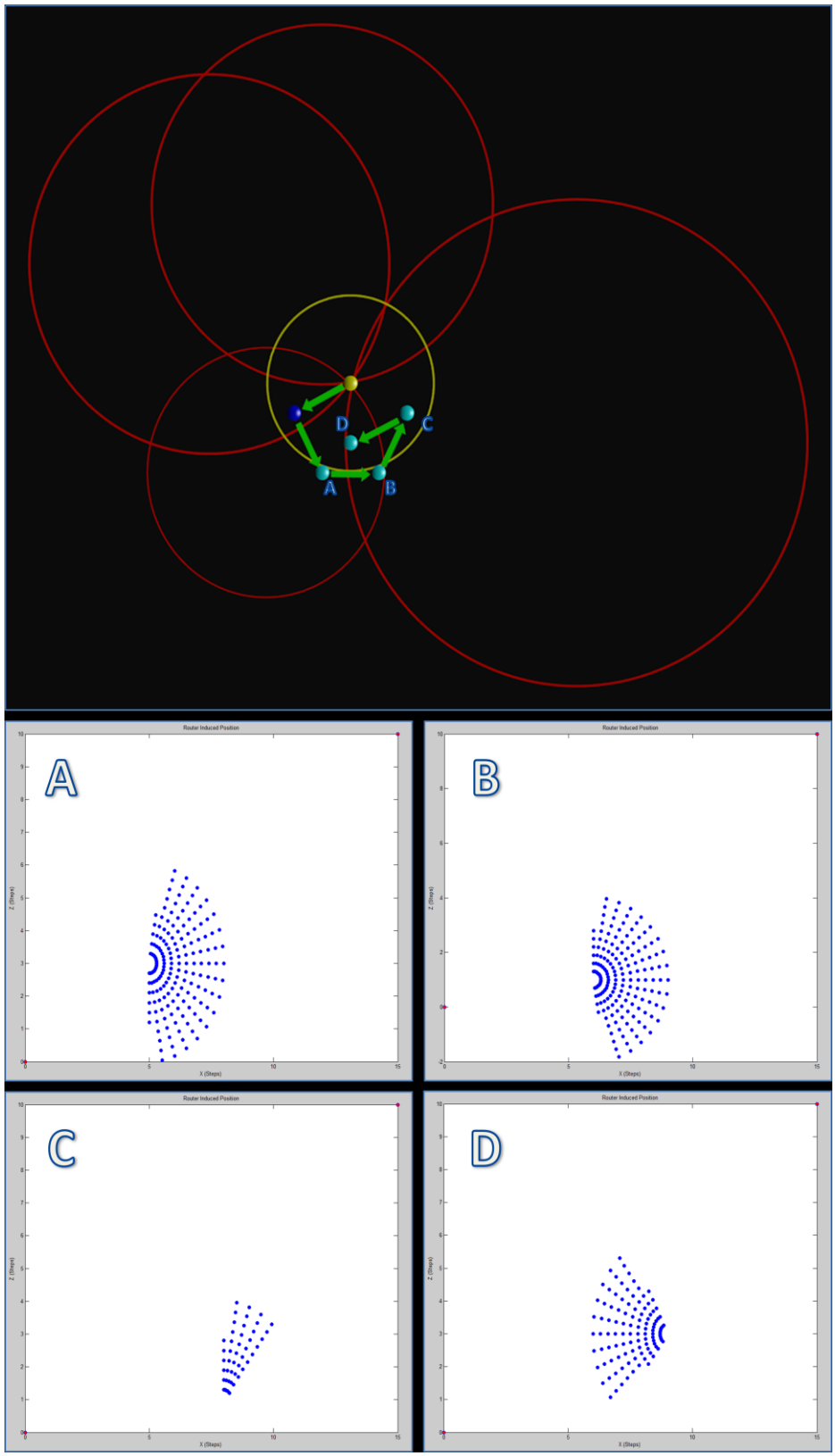

Figure 42 - Induced Position

6 Conclusions

While the contributions from this thesis include all components necessary for a complete indoor positioning system its key contribution is the manner in which it determines, with reasonable accuracy, the positions of nearby access points. Analysis of the data used to build the histograms in Figure 40 is shown below Table 1.

Table 1 - Statistical Data

Condition	Mean Error (Step)	Standard Deviation (Step)
Minimum RSSI > -10dBm	14.3	13.0
Minimum RSSI > -20dBm	16.1	13.4
Fully Bound Minimum RSSI > -10dBm	14.8	12.4
Fully Bound Minimum RSSI > -20dBm	16.9	12.7

Conclusions drawn from the table above indicate that for access points within some RSSI range of the user's path of travel it is possible to determine an effective position which, free of obstruction and interference lie in close proximity to their actual positions. Although difficult to test reliably without a shielded environment, preliminary field testing agrees with the findings above albeit with slightly different ranges in the RSSI than are proffered by the simulated tests.

Of note is that the approach presented in this thesis has a significant reliance on highly accurate initial measurements of relative motion to calibrate nearby access points. A second limitation to the algorithm presented is its apparent reliance on path. The manner in which bounds are placed upon the access points means that the algorithm performs well in situations in which a mix of straight lines and weaving are blended. The exact extent to which the two are blended as well as

determining metrics describing the effectiveness of a given path are both areas in which future research is needed.

7 Future Work

The contributions from this thesis provide a foundation whereby information is drawn onto a previously blank slate. The addition of almost any form of information which further refines the bounding hulls will result in higher accuracy results. Of the common approaches a few stand out.

As previously stated research into optimized user movement paths and their effect on accuracy would both shed light on further optimizations within the algorithm itself and provide possible ways in which the usability of the overall application could be improved. In a similar manner, it may prove useful to investigate the speed at which the user moves and determine the extent to which it effects accuracy.

As described in Chapter 2.3 the addition of maps, either interior or simply exterior bounds would likely improve results in response to the addition of reliable outer bounds.

Integration of modern approaches such as the utilization of multiple users in simultaneous movement through an environment or alternatively a database containing previously recorded data also represents a huge area for further research. Such techniques would also compliment the initial intention as a means of positioning for emergency response teams.

Bibliography

AmBrSoft. (2012, March). *Analytical Geometry.* Retrieved April 3, 2015, from AmBrSoft: http://www.ambrsoft.com/TrigoCalc/Circles2/Circle2.htm

Ambuhl, M., & Reinthaler, N. (2011, December 19). *Distrubuted Computing Group at ETH Zurich.* Retrieved March 16, 2015, from Path Tracking - Guiding Smartphones Based on Recorded Tracks: http://disco.ethz.ch/theses/hs11/PathTracking_Report.pdf

Android. (n.d.). *Location and Sensors APIs.* Retrieved April 6, 2015, from Android Developers: http://developer.android.com/guide/topics/sensors/sensors_overview.html

Bolliger, P. (2008). *Redpin - Adaptive, Zero-Configuration Indoor Localization.* Retrieved March 16, 2015, from Distributed Systems Group at ETH Zurich: http://www.vs.inf.ethz.ch/publ/papers/bolligph-redpin2008.pdf

Brown, K. Q. (1978). *Carnegie Mellon University Research Showcase.* Retrieved March 16, 2015, from Fast intersection of half spaces: http://repository.cmu.edu/cgi/viewcontent.cgi?article=3351&context=compsci

Cho, D.-K., Mun, M., Lee, U., Kaiser, W. J., & Gerla, M. (2010). *Network Research Lab UCLA.* Retrieved March 16, 2015, from AutoGait: A Mobile Platform that Accurately Estimates the Distance Walked: http://nrlweb.cs.ucla.edu/publication/download/533/autogait.pdf

Graham, R. L. (1972). An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set. *Information Processing Letters , 1*, 132-133.

Karlsson, F., & Karlsson, M. (2014). *Lund University Publications.* Retrieved March 16, 2015, from Sensor Fused Indoor Positioning Using Dual Band Wifi Signal Measurements: http://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=4317328&fileOId=4317329

KU School Of Engineering - Electrical Engineering and Computer Science. (n.d.). Retrieved March 16, 2015, from About the Eaton Hall Map: http://www.eecs.ku.edu/facilities/eaton_hall

Li, F., Zhao, C., Guanzhong, D., Gong, J., Chenxing, L., & Zhao, F. (2012). A reliable and accurate indoor localization method using phone inertial sensors. *ACM Conference on Ubiquitous Computing* (pp. 421-430). New York: ACM New York.

Preparata, F. P., & Muller, D. E. (1979). FINDING THE INTERSECTION OF n HALF-SPACES IN TIME O(n log n). *Theoretical Computer Science* (8), 44-55.

Shala, U., & Fredrik, F. (2011, September). *Indoor Position using Sensor-fusion in Android Devices.* Retrieved March 16, 2015, from http://hkr.diva-portal.org/smash/get/diva2:475619/FULLTEXT02.pdf

Shamos, M. I. (1976). Geometric Intersection Problems. *Foundations of Computer Science* (pp. 208-215). Houston: IEEE.

Zhao, N. (2010). *Full-Featured Pedometer Design Realized with 3-Axis Digital Accelerometer.* Retrieved March 16, 2015, from Analog Dialog: http://www.analog.com/media/en/technical-documentation/analog-dialogue/pedometer.pdf