Multimodal Indoor Device Localization

Joseph Menke

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2016-68
http://www.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-68.html

May 12, 2016

Multimodal Indoor Device Localization

by Joseph Menke

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor A. Zakhor
Research Advisor

Date

* * * * * *

Professor C. Spanos
Second Reader

Date

Multimodal Indoor Device Localization

Joseph Menke
Advisor: Avideh Zakhor
UC Berkeley

Abstract

In this thesis we present a method of indoor localization and tracking that combines multiple sensor measurements to remove dependence on any one information source. A two-step process is proposed that performs an initial localization estimate, followed by particle filter based tracking. Initial localization is performed using WiFi and image observations. For tracking we fuse information from WiFi, magnetic, and inertial sensors. We demonstrate the feasibility of this system using fingerprint maps that are collected with a single walkthrough of the building at normal walking pace. In addition to a smartphone or tablet, only a foot mounted inertial measurement unit (IMU) is needed for database generation. Only a smartphone is needed for positioning after database generation. The positioning method presented uses sensors available on most mobile devices and requires no new infrastructure to be placed in the building. We present results for two locations: the Stoneridge Mall in Pleasanton, California, and the Doe Library at the UC Berkeley campus. We achieve an average location error of 2.6m across both locations.

1 Background

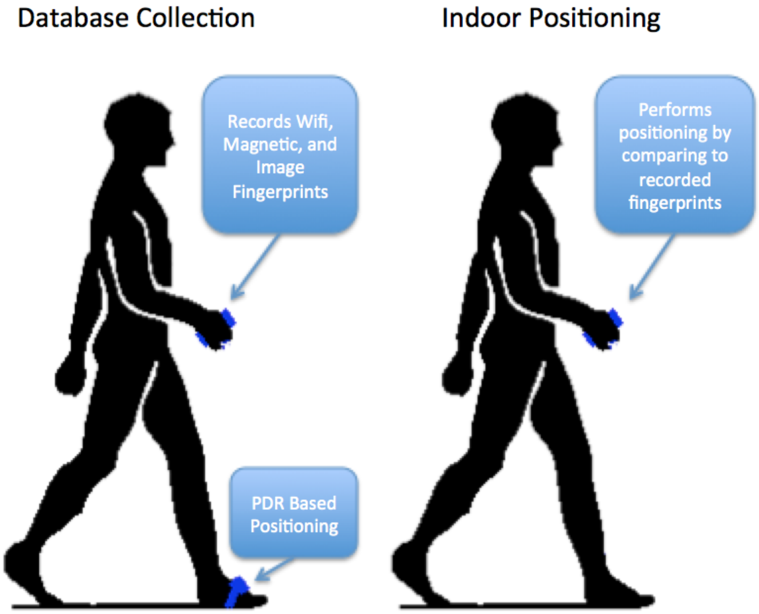

Database Collection
Indoor Positioning
Records Wifi, Magnetic, and Image Fingerprints
PDR Based Positioning
Performs positioning by comparing to recorded fingerprints

Figure 1: System overview. In database collection positioning is performed with the foot mounted device. Once a database is collected a smartphone can be used for positioning.

1.1 Introduction

Global Positioning Systems have allowed for the development of a wide variety of new technology. From personal navigation devices to automated drones, GPS is the primary solution for determining the location of our devices. GPS does have it's limitations however. Arguably, the largest of these limitations is the inability for GPS to work indoors, and unlike for outdoor positioning there is currently no well established solution to the indoor positioning problem.

Despite this, indoor positioning does have some very useful applications. Reliable indoor positioning can help people navigate large conference centers or malls. It can direct people or robots in warehouses to reduce errors and increase productivity. In cases of emergency it can direct people safely and quickly to the exits of the building or help direct first responders to the proper location in need of emergency assistance. These critical moments could provide the extra time needed to reduce incidents and save lives.

While there is the potential for many different solutions to this problem, there are a variety of constraints to consider. One constraint is to develop a system that can run on a mobile device such as a smartphone or tablet. While it would be possible to develop new hardware for indoor positioning, the ubiquity of these mobile devices ensures easy adoption of solutions that support the current technology. Another constraint is to not require any new infrastructure to be placed in the building. This is important as installing new hardware can be both difficult and expensive, reducing the chances these solutions will be adopted. Further we should attempt to minimize the amount of information needed about the building in order to determine position. Similar to infrastructure, if data is difficult or expensive to collect it can limit the number of places where the system is a practical solution. Finally we would like to minimize the burden on the user required by the system. This means having a simple interface that doesn't require users to take too many actions for them to determine their position.

In this thesis we present a solution to the indoor positioning problem that extends the work of Levchev et al. [1] on WiFi and image based localization. We have extended the tracking method to include information from magnetic fingerprints and an onboard gyroscope. Further we split the system into a two-step procedure that decouples localization and tracking. For this we perform an initialization step where a single picture from the user is combined with a WiFi scan to provide an initial location estimate. From there, the tracking step fuses step detection along with WiFi and magnetic measurements to provide a continuous estimate of the user's location. During the tracking we assume that the device stays oriented at a fixed known offset from the direction of motion.

We do not require any new infrastructure to be placed in the building and no new hardware is needed for user positioning. We also introduce a mapping procedure that, given a floorplan, allows all additional information to be collected with a single walkthrough of the building at normal walking pace. For this initial mapping we use a smartphone or tablet along with a single foot mounted Inertial Measurement Unit (IMU). This device is not required for positioning after initial mapping. Fig. 1 shows the devision of these two tasks. We show results for two locations; the Stoneridge Mall in Pleasanton, California, and the Doe Library on the UC Berkeley campus.

1.2 Related Work

One approach to indoor positioning is to apply inertial dead reckoning using an inertial measurement unit consisting of a magnetometer, accelerometer, and gyroscope [3] commonly found in today's mobile devices. Two methodologies arise from this approach. By integrating raw accelerometer and gyroscope readings it is possible to determine displacement information but this integration introduces significant drift error [4]. Alternatively to estimate users movement, a commonly adopted method is to first detect their steps, and then estimate the corresponding step lengths [5, 6, 7]. For mobile devices this can still be a challenge due to the possibility of non-step related movement that can be difficult to detect and classify. A more accurate method is to use a foot-mounted sensor to track movement [8]. This reduces the amount of non-step related motion and utilizes the zero-velocity state of the foot when walking to estimate biases of the sensor. This method provides accurate positioning but requires additional hardware. On its own it also has no way to relate the tracked position to a global coordinate frame.

Another common approach is to use the WiFi infrastructure that is already prevalent inside most modern buildings. Due to unpredictable distortion caused by different building materials and a variety of other sources it is difficult to determine relative position even when the physical locations of access points are known. A popular alternative for utilizing this infrastructure is to construct a database of WiFi Received Signal Strength Indicator (RSSI) fingerprints for the building [11, 12, 13]. The RSSI fingerprint for each location is a vector of decibel values where each entry corresponds to the WiFi signal strength of a particular access point detected at that location. To utilize this database an algorithm such as Redpin [9] is used to relate the observed information to a location in the database. A major advantage of this method is its prevalence as hardware infrastructure and the ubiquity of WiFi scanning capability on mobile phones and consumer electronic devices. A disadvantage is that the location dependency of RSSI is subject to interference and signals can be very similar in wide-open spaces. Several systems have been able to demonstrate room level accuracy e.g. approximately 5 to 10m using this method. At the Microsoft Indoor Localization Competition at IPSM 2015, Wu et al. was able to demonstrate a localization accuracy of 4.22m using RSSI fingerprint methods [14]. Further, Zou et al. demonstrated a localization accuracy of 3.65m using a combination of model and fingerprint based methods [15].

Recently an image-based indoor localization scheme [16, 17, 18] has been proposed for mobile devices with cameras. Usually, a locally georeferenced database of images is constructed via a man portable ambulatory backpack of sensors, although we show later that this database generation is also possible with consumer devices. The Scale Invariant Feature Transform (SIFT) [19] allows accurate matching of images from a client side mobile device with those contained in the database. Even though this method generally achieves higher accuracy than WiFi RSSI matching, its performance is degraded when the query image has few distinguishing features, or when the pictures are of low quality due to out-of-focus and/or motion blur. Liang et al. [17, 18] have shown that when blur-free images are captured with a stationary camera, image-based localization could achieve an accuracy of 2 meters over 80% of the time, and 4 meters over 90% of the time.

Lastly there have been recent efforts to perform localization using distortions in the earths magnetic field caused by metal structures in buildings. Traditionally magnetometers are used as digital compasses to determine the orientation with respect to magnetic north. This has been shown to be problematic indoors due to the distortions caused by iron or steel in buildings. However it has been determined that if these distortions are mapped, it is possible to use them for localization [20, 21, 22]. Haverinen et al. show that these fields could be used with a particle filter to localize a mobile device. The advantages to using magnetic data are the potential to use high sampling rates, the prevalence of these sensors in mobile devices, and that no new hardware infrastructure is required. The magnetic reading at a single location however is not enough to uniquely determine position. In general it takes about 10m of walking to generate a reasonable estimate of position, which is undesirable for many applications. However once a reasonable estimate is achieved, tracking can be performed with high accuracy.

In contrast to previous works this thesis fuses multiple sources of information to provide a more reliable estimate. In addition we de-couple initialization and tracking in order to provide a more immediate location estimate, while still retaining the benefits of the various methods. Further we demonstrate a data collection method that greatly reduces the amount of time and additional hardware necessary to obtain the required information for positioning, while still retaining the same position accuracy. We show that these maps provide accurate localization results for months after collection, and that the method can be applied to locations with very different properties.

1.3 Outline

The structure of this thesis is as follows. In section 2 we discuss how the data is collected and processed for use in our system. In section 3 we explain the initialization step that is used to determine the initial states of particles for our filter. The processes of propagating and weighting particles as the user walks is described in section 4. In section 5 we explain how we compute error to validate our system. In section 6 we review the results of our experiments and finish with a conclusion in section 7.

1.4 Coordinate Frame

We define a right hand coordinate frame for our system such that the user walks on the xy-plane. Yaw is measured as rotation about the z-axis with 0-degrees pointing along the x-axis increasing positively as the user rotates counterclockwise towards the y-axis.

2 Database Generation

2.1 Path

To generate our fingerprint database a smartphone is carried throughout the building in a single walkthrough at normal walking pace. A method of positioning is needed to determine where the smartphone is located during data collection. While any method of positioning can be used, we opted for a single foot mounted IMU that is capable of tracking movement over long distances. The IMU reports position information to the smartphone via bluetooth where it is logged and timestamped. The phone is held in front of the user with a constant orientation so a known offset with respect to the direction of walk can be determined. Positions recorded from this system are improved using manual loop closures and landmark locations. The positions are then optimized using these constraints and manually aligned with a floor plan to provide fingerprint locations in the common coordinate frame. To determine these constraints we manually align the path to either the floorplan or an overhead satellite view of the building from google maps. We start by defining a loop closure at the start and end of the walk and then add additional constraints as necessary to get a reasonable path.

2.2 Image

The image database is made up of images, camera positions, and sparse depth maps. To collect images during the walkthrough the phone is held vertically with the camera pointing perpendicular to the direction of walk. If the user is walking along a wall the phone should be pointed towards the far wall to capture as much information as possible. Images are collected at a frequency of roughly 1Hz. As with the WiFi database, the images are then associated with a position from the foot mounted IMU in order to determine the camera pose. A sparse depth map is then computed for each image using the method proposed in [18]. For this we use the Scale Invariant Feature Transform (SIFT) algorithm [19] to first extract features in each image. In every pair of subsequent images we determine matching feature correspondence through nearest neighbor search. We compute the fundamental matrix that relates the two sets of SIFT features and removes any feature pairs which do not satisfy epipolar constraints. For each pair of SIFT correspondences, we calculate the 3D vectors that connect the camera centers of the images to the respective pixel locations of their SIFT features. We use the following equation to determine the intersection point or the point mutually closest to the two vectors:

$$x = \left(\sum_{i=1}^{2} I - v_i v_i^T \right)^{-1} \left(\sum_{i=1}^{2} (I - v_i v_i^T) p_i \right)^{-1} \tag{1}$$

where x is the intersection point, v_i is the normalized direction of the i^{th} vector, and p_i is a point located on the i^{th} vector.

2.3 WiFi

As the fingerprinting smartphone is carried through the building we record scans that consist of access point MAC addresses and the observed signal strength for each access point. A scan contains measurements for all access points viewable by the device during the duration of the scan. Each scan is associated with a single location in the map by matching with the recorded position at the time of the scan. Since scans are not instantaneous the average time stamp of the scanned access points is used to determine the appropriate position to associate with. This results in a fingerprint which is stored in a SQL database on a local server.

2.4 Magnetic

A magnetic database consists of observed magnetic vectors that have been rotated back to a global coordinate frame. The magnetometer on a typical smartphone collects at a much faster rate than images or WiFi scans can be collected. This sensor, however, is also very noisy and so in generating our database we average sensor readings around positions reported by the foot mounted IMU in order to obtain the observation at that position. At this point the observations are with respect to the local coordinate frame of the phone. In order to obtain a useful fingerprint we need to rotate this vector using the rotation matrix between the phone coordinate frame and the database coordinate frame. First we perform tilt-compensation using the onboard accelerometer. Fig. 2 shows a visualization of this procedure. The solid box represents what we think the orientation of the phone is before tilt compensation. However we observe an accelerometer reading that informs us that the device is not perfectly flat. This allows us to estimate the true orientation of the phone, represented by the dashed box, and produce a compensated magnetic reading **c** from the original magnetic reading **m**. Tilt compensation removes the effects of minor changes in pitch and roll that can occur while the user walks through the building.

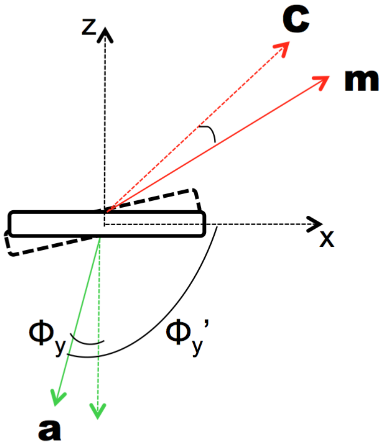


Figure 2: An illustration of the tilt compensation procedure.

In Figure 2, ϕ_y is the angle of rotation of the device about the y-axis. It is equivalently the angle between the observed gravity vector and the gravity vector we would observe if the device was perfectly flat. ϕ'_y is the angle between the observed gravity vector and the x-axis. Here we make the approximation that the reading from the accelerometer is dominated by gravity. Under this assumption we can compute the roll and pitch of the phone. We compute the rotation about the y-axis, ϕ_y as:

$$\begin{aligned}
\phi_y &= \pi/2 - \phi'_y && (2)\\
a_x &= cos(\phi'_y) && (3)\\
a_x &\approx \pi/2 - \phi'_y && (4)\\
\phi'_y &\approx \pi/2 - a_x && (5)\\
\phi_y &\approx \pi/2 - (\pi/2 - a_x) \\
&= a_x && (6)
\end{aligned}$$

where a_x is the x-component of the normalized gravity vector. Since we assume the device will be held mostly flat we can use the small angle approximation to get from equation 3 to equation 4. The rotation about the x-axis is computed in the same way resulting in $\phi_x = a_y$. We now just need to undo these rotations by multiplying the magnetic vector by the rotation matrices computed from the negative of these angles:

$$\begin{aligned}
\vec{c} &= R_y(-\phi_y) * R_x(-\phi_x) * \vec{m} \\
&= R_y(-a_x) * R_x(-a_y) * \vec{m} && (7)
\end{aligned}$$

where $\vec{m} = [m_x, m_y, m_z]^T$ is the original magnetic vector, $\vec{c} = [c_x, c_y, c_z]^T$ is the tilt compensated magnetic vector. $R_x(\phi)$ and $R_y(\phi)$ are computed as:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & sin(\phi) & cos(\phi) \end{bmatrix} \tag{8}$$

$$R_y(\phi) = \begin{bmatrix} cos(\phi) & 0 & sin(\phi) \\ 0 & 1 & 0 \\ -sin(\phi) & 0 & cos(\phi) \end{bmatrix} \tag{9}$$

To obtain the magnetic vector in the database coordinate frame we also need to determine the yaw of the phone. While the phone reports an orientation from its IMU, this orientation is dependent on the magnetic vector at that location, making it an inappropriate choice for determining the pose with respect to the database coordinate frame. Rather we opt to infer the yaw of the phone from successive positions reported by the foot mounted IMU. Specifically:

$$\Delta\vec{a_i} = \vec{a}_{i+1} - \vec{a_i} \tag{10}$$

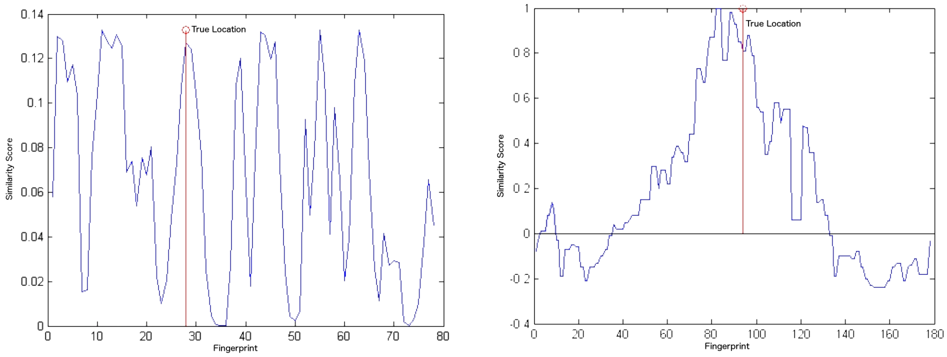
$$\hat{\theta} = tan^{-1}((\Delta\vec{a_i} \times \vec{x})/(\Delta\vec{a_i} \cdot \vec{x})) \tag{11}$$

where $\vec{a_i}$ is the position at time i, $\vec{x}$ is a unit vector pointing along the x-axis and $\hat{\theta}$ is the estimate of our yaw at that position.

3 Initialization

In order to track a user they must first be localized inside a known coordinate frame. While this has been shown to be possible with a particle filter alone [29], it has significant pitfalls. The first is that a much larger number of particles must be used which can exceed the computational abilities of the average smartphone. The second is that the user must walk a certain distance before the filter is able to localize them accurately. This is impractical for navigation purposes. Rather we implement methods that allow for reasonable initial estimates of location rapidly and without requiring the user to walk any distance.

One method for this would be to generate particles at all locations on the map and then weight them using the same methods we use later in the tracking step. In Fig. 3 we show example similarity scores computed for fingerprints for both magnetic and WiFi observations. This can be thought of as a distribution of the probability that the measurement was taken at a given fingerprint. What we observe is that a magnetic observation results in a multimodal distribution with many narrow peaks. This means that a magnetic observation provides good local information but very little information about the global position of the user. Without walking the magnetic readings do not contain enough information to determine the global position of the user. A WiFi observation results in a distribution that is nearly unimodal but with a very wide variance. Simply weighting the particles by this score will not eliminate enough particles to give an accurate location estimate. In theory it may be possible to wait for multiple WiFi observations to remove enough particles for a precise location estimate, however due to the infrequency of WiFi measurements the amount of time this would take would be unreasonable and, due to the sparse nature of our database, may still not converge to the correct estimate.



(a) Magnetic observation similarity

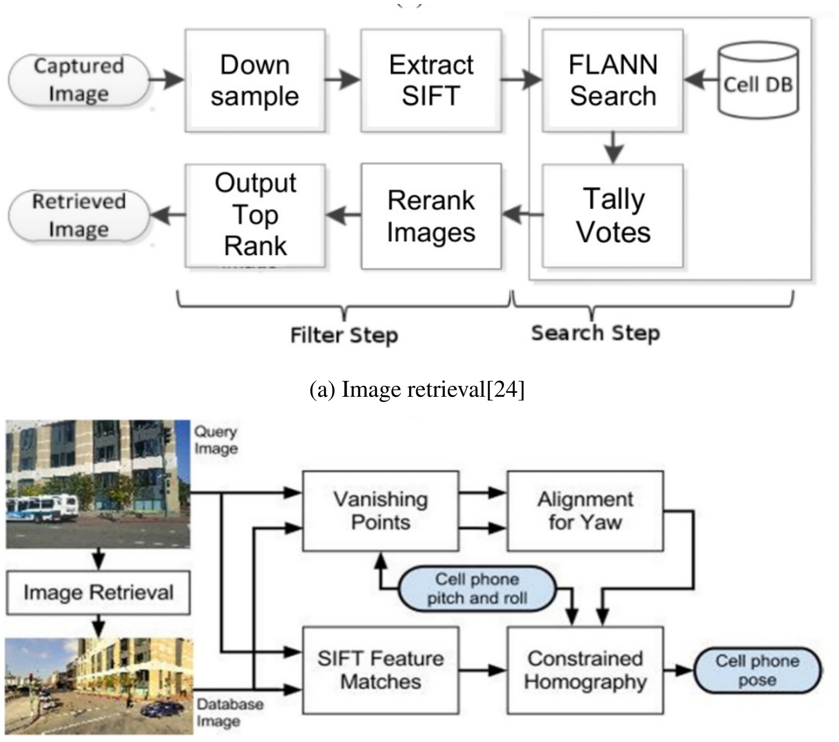
(b) WiFi observation similarity

Figure 3: Example observation similarity to fingerprints in a database

Instead we use a clustering method to compute an initial guess of position from WiFi measurements. This allows us to give a low-variance estimate of the users position. However, this estimate can sometimes be erroneous, or report a bad confidence meaning it is unusable. To account for this we include image based initialization. Images provide very accurate estimates of location, but can also be erroneous due to incorrect matches or fail due to changes in the environment. Together they provide a more robust estimate of initial location.

3.1 Image

To perform image based localization the user takes a picture of the area around them. This image is then sent to our server for pose computation using the method presented in [18]. The first step in this is to retrieve an image from the database that shares features with the query image (Fig. 4a). To do this our indoor image retrieval system loads the SIFT features of every database image into a single FLANN kd-tree [23]. Next, we extract SIFT features from the query image and for each SIFT vector extracted, we lookup its top N neighbors in the kd-tree. For each closest neighbor found, we assign a vote to the database image that this feature vector belongs to. This is repeated this for all the SIFT features in the query image. The database images are ranked by the number of matching SIFT features they share with the query image. We check geometric consistency and filter out mismatched SIFT features. Further we solve for the fundamental matrix between the database and query images and eliminate feature matches that do not satisfy epipolar constraints [24]. We also remove SIFT feature matches where the angle of SIFT features differ by more than 0.2 radians. At this point the highest ranking image is chosen for computing the pose of the query image.



(a) Image retrieval[24]

(b) Pose computation[24]

Figure 4: Method for computing initial pose estimates from images.

To compose pose, pitch and roll estimates from cell phone sensors are used in vanishing point analysis to compute yaw of the query image [16] (Fig. 4b). Once we estimate orientation, SIFT matches are used to solve a constrained homography problem within RANSAC [25] to recover translation between query and database images. The scale factor is given by the depth of the SIFT features in the database image computed during database generation. A confidence value for the location estimate is also computed from a logistic regression classifier. A confidence value of .5 or higher indicates high probability that this is a good location estimate.

3.2 WiFi

The WiFi initial estimate is performed using a method proposed in [26]. In this method we compute scores for each fingerprint in the database using a method proposed in [9]. This score, which we have called the Redpin score after the paper that proposed it, is a combination of the number of common APs, the number of non-common APs, and the differences in RSSI values for the common APs. In [10] it was shown that a k-nearest neighbors search with k=5 on the Redpin results, with the location chosen through a majority vote, produces more accurate results than standard Redpin. In [26] a similar algorithm was proposed noting that the important aspect was to find an area with similar fingerprints, but these may not be the fingerprints immediately nearest to each other. In this algorithm clusters are built starting with the fingerprint with the highest score. Fingerprints

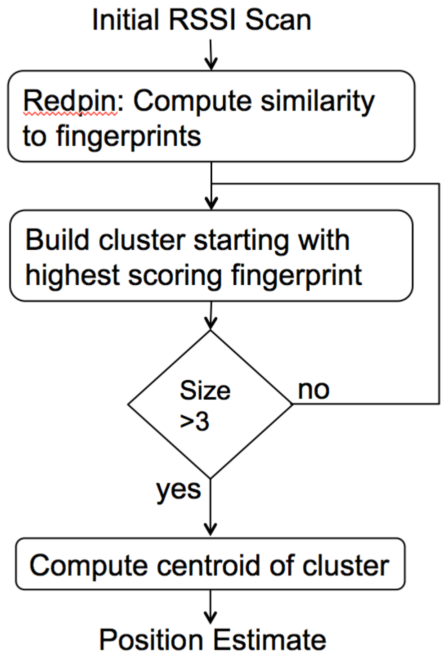

Initial RSSI Scan
Redpin: Compute similarity to fingerprints
Build cluster starting with highest scoring fingerprint
Size >3
no
yes
Compute centroid of cluster
Position Estimate

Figure 5: A flowchart of the method of computing an initial position estimate by comparing a single WiFi scan to all the fingerprints in the database

are evaluated in order of decreasing score. A fingerprint is added to a cluster if it is within a fixed distance from any fingerprint in that cluster. Otherwise a new cluster is formed. If the fingerprint is within this distance of multiple clusters, it is assigned to the one with the element with the highest score. Once a cluster of size 3 is created, the centroid of the cluster is returned representing an estimate of the position of the mobile device. Fig. 5 shows a flow chart of the clustering algorithm. Confidence in this value is determined by the spread of clusters and ratio of the max Redpin score to the average Redpin score. Confidence ranges from 0-1 with a value greater than 0.5 indicating a reasonable estimate.

4 Tracking

For location tracking we implement a particle filter based tracking method to fuse accelerometer, gyroscope, magnetic, and WiFi measurements. While it would be possible to also use images in the tracking step, we choose not to for several reasons. If we want to capture images continuously, it requires that the user hold the phone in an awkward position, incurs large amounts of additional computation, and since images are captured automatically, will often result in unusable images. Alternatively, we could require that the user takes pictures manually as they walk. This, however, would be difficult in many practical scenarios and incurs unnecessary burden on the user.

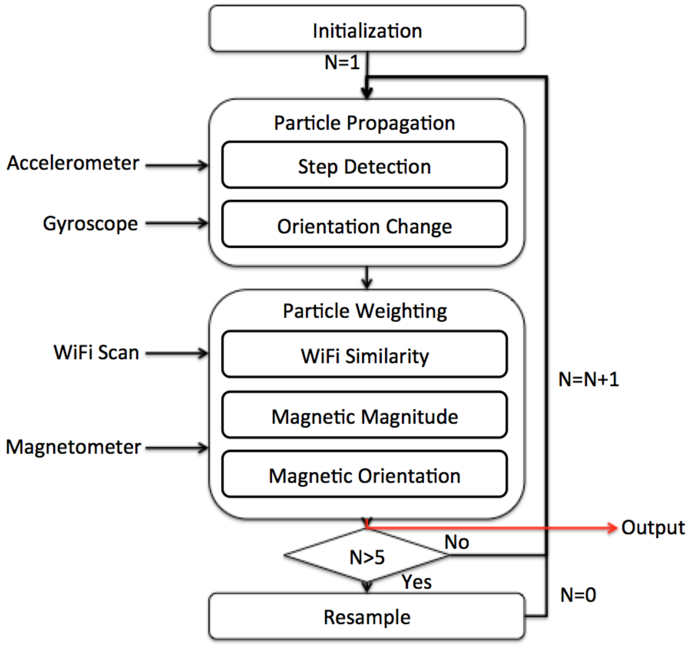


Figure 6: Block diagram of tracking procedure.

An overview of the tracking method is given in Fig. 6. Particles are represented by a three dimensional state vector consisting of the x and y position and yaw orientation of the particle. The initial locations of the particles are samples from a 2D Gaussian distribution with a mean equal to that of the location estimate provided by either WiFi or image based initialization. If both WiFi and image initialization report reasonable confidence values then half the particles are sampled from each distribution. If neither reports reasonable confidence the user will be required to retry the initialization process with a new image and WiFi scan. The user also has the ability to use only one of the two methods for initialization if they notice one method is resulting in poor estimates in a location. Initial yaw estimates are sampled uniformly in the range [0,360). The location reported by the filter at each interval is the weighted average position of all particles.

4.1 Steps

For the propagation of the particle filter we perform step detection on the mobile device and propagate the particles at each step. Algorithms presented in [27, 28] are used to detect steps and estimate their lengths from accelerometer readings. A change in yaw is determined from gyroscope measurements. It is assumed that the device stays oriented at a fixed known offset from the direction of motion. For each particle, we add a random noise to both the rotation and translation of each step. This noise is assumed to be Gaussian with a mean and standard deviation that have been determined empirically. The movements of the particles are constrained to be within a known floor plan of the building. If a step causes a particle to cross a wall then it is assumed that the particle cannot have represented a true location and it is eliminated. Fig. 7 shows an example step propagation. In Fig.

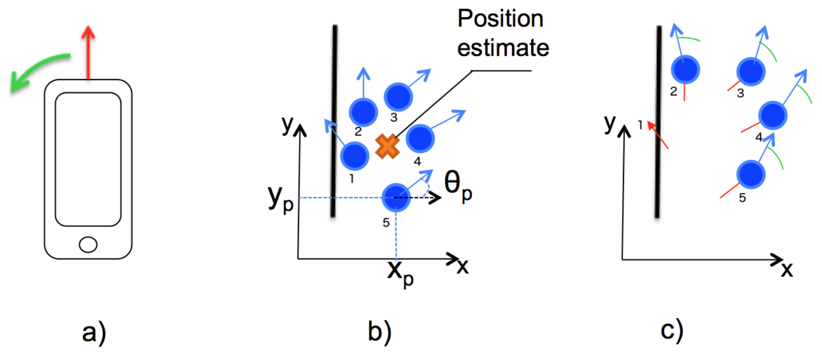


Figure 7: The step propagation procedure. a) Device observations of forward movement and change in orientation. b) The state of particles prior to a step update. c) The state of the particles after a step update.

7a the red arrow represents the measurement of forward movement as determined by step detection while the green arrow represents the measurement of change in rotation as determined by the phones gyroscope. In Fig. 7b we show the state of the particles prior to a step update. Each particle (shown as a blue dot with an arrow) is represented by three values: x_p and y_p which represent the particle's position, and θ_p which represents the particle's orientation. The estimated position/orientation of the device is the weighted average of the positions and orientations of the particles. The dark black line represents a wall in the floor plan. In Fig. 7c we show the state of the particles after a step update. Each particle has moved forward by the amount measured by step detection, plus some random noise. Similarly, each particle has changed orientation by the amount measured by the gyroscope, plus some random noise. The particle on the far left (labeled "1") has been eliminated as it crossed a wall, which is not a valid trajectory.

4.2 WiFi

WiFi scans are collected continuously as the user walks. At the completion of each scan the observations are sent to the server where the normalized Redpin score for each fingerprint is computed. The time at which the scan was recorded is saved on the device. Since scans are not instantaneous the average time stamp of the scanned access points is recorded. When a response is received by the server particles are weighted by the score of the fingerprint closest to their position at the time at which the scan was recorded. It is also be possible to interpolate between nearby fingerprint locations to provide a weight based on the particles exact position rather than just the closest fingerprint [30], but this is not used in our implementation.

4.3 Magnetic

The onboard digital compass of the mobile device is used to record the magnetic vector. While this sensor reports magnetic readings at a very high frequency we only perform updates after each time the user walks one step. This constraint is necessary as the particle filter assumes that the noise is independent in successive measurements. If multiple readings are taken in a location that is not correctly modeled by the closest fingerprint then the error that results from this will be correlated for all measurements at that location. By only using one magnetic reading at each location we help preserve the idea that the noise is independent in each observation. Since the weight is computed after the step propagation, we use only the most recent magnetic reading at the time of computation as this will be indicative of the orientation at the end of the step.

While we assume the phone is held at a fairly constant orientation, we know that it would be impossible to keep the phone perfectly stationary and so we perform tilt compensation on the magnetic reading using the same method as in the database generation.

For each particle we then find the closest fingerprint in the database using a quad-tree search. Probabilities are computed for observations of the magnitude and orientation of the magnetic vector separately. The magnitude is compared to the magnitude of the closest fingerprint as shown in Fig. 8b. The probability is assumed to be Gaussian centered at the magnitude of the fingerprint, with a standard deviation that is determined empirically. This represents the probability that a particle is at the correct position.

To determine the probability that a particle is at the correct orientation we compute an "observed orientation" of the particle from the measurement. That is, assuming the particle is at the correct location, we can compute an observed orientation by comparing our magnetic vector to the closest database vector. For this we project both the observed magnetic vector and the database magnetic vector to the X-Y plane. Since these values have already been tilt compensated, this projections are merely the X and Y components of the tilt-compensated vectors as shown in Fig. 8a. From these values we then compute the observed orientation θ_o as the angle between the projections of observed magnetic vector and the database magnetic vector:

$$\theta_o = tan^{-1}((\vec{v_o} \times \vec{v_d})/(\vec{v_o} \cdot \vec{v_d})) \tag{12}$$

where $\vec{v_o}$ and $\vec{v_d}$ are the projections of the observed and database magnetic vectors respectively. As seen in Fig. 8c, we compute the orientation weight by comparing the particle orientation to the observed orientation. The probability is assumed to be Gaussian with a standard deviation that is determined empirically. The Gaussian is centered at the orientation of the particle and evaluated at the observed orientation. The particle is weighted by the product of the magnitude and orientation probabilities.

4.4 Resampling

We resample our particles after every five steps the user takes. The frequency of resampling is a calibrated value that worked well for our system. To resample particles we sample with replacement a new set of particles from our current set of particles. This sampling occurs until we have a new

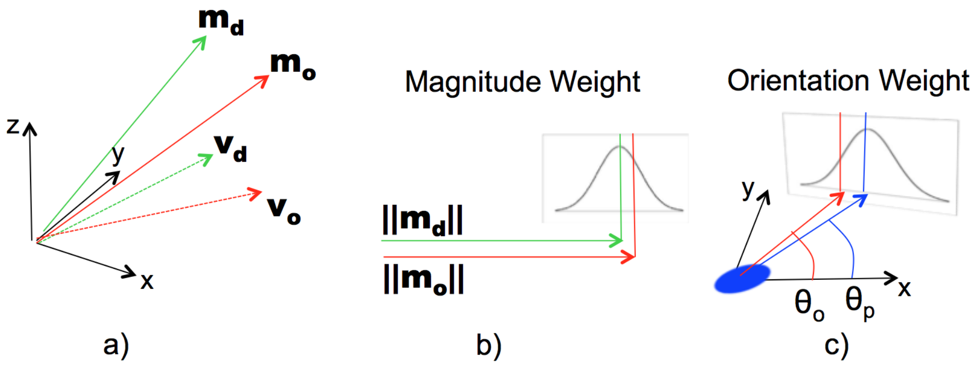


Figure 8: Method for weighting particles based on magnetic observation. a) the observed magnetic vector m_o and the database magnetic vector m_d, and their projections on the xy-plane: v_o and v_d respectively. b) the computation of the magnitude weight of a particle. c) the computation of the orientation weight of the particle. θ_o is the estimated orientation of the particle computed from v_o and v_d in eq. (12).

dataset of particles with a number of particles equal to our maximum number of particles. For our system the maximum number of particles is set at 300. The probability of a particle being selected is proportional to the weight the particle received over the last five steps. After resampling the old set of particles is deleted and the new set replaces it. The weights of all the particles are reset to 1 after resampling.

5 Verification

In order to characterize the performance of our system we need to determine the error of the reported position of our system to the true position of the evaluator. For this the evaluator wears the same foot mounted PDR device that was used in data collection. This allows us to compare the locations reported by our system to the "ground truth" reported by the PDR device. The PDR device reports position at a rate of approximately 1Hz. Whenever the PDR device reports a position, we compute the error based on the current estimate of our system. This ensures that our system is accurate both spatially and temporally. The PDR positions used are again those optimized utilizing manual loop closures and landmarks.

While we denote the location from the PDR device as a "ground truth", it should be noted that the location reported by the device can have errors as well despite corrections provided by manual loop closures. The locations reported by this device will even occasionally go outside our floorplan, increasing, possibly significantly, the estimation of our error. Despite this we felt that the benefits of a continuous error estimate made it a vast improvement compared to other error comparisons we could have used.

6 Results

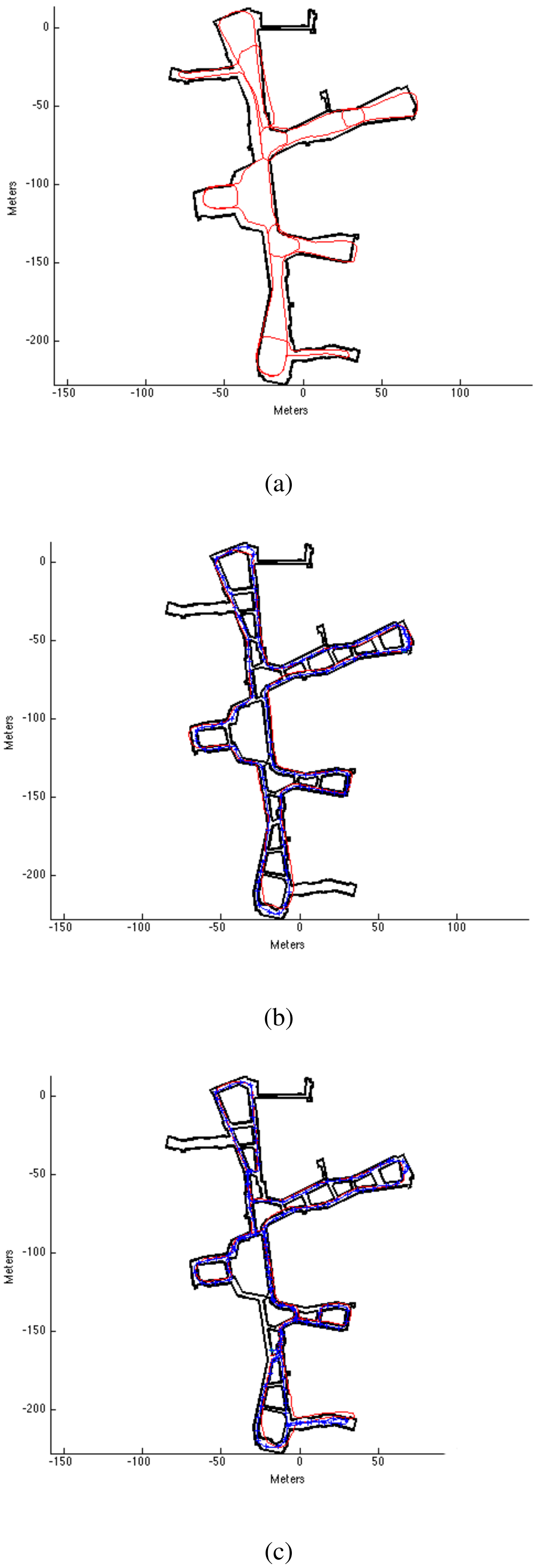
We test our system in two locations. The first is Stoneridge Mall in Pleasanton, California. The second test scenario is the Doe Library on campus at UC Berkeley. For both test cases fingerprint data is collected using a smartphone or tablet and a foot mounted PDR device. Data collection is performed by one person in a single walkthrough of the building at normal walking pace.

6.1 Stoneridge Mall

Table 1: Experimental results for the Stoneridge Mall.

	Err. Mean(m)	Err. Std.(m)	90% Err.(m)	Length(m)
D1	2.15	1.21	3.7	830
D2	2.19	1.45	3.8	827
D3	2.92	1.41	5.0	835
D4	2.70	1.21	4.3	838
R1	2.64	2.69	4.0	921
R2	3.03	2.98	5.4	909
R3	2.61	1.51	4.6	892
R4	2.63	1.39	4.6	1022
R5	2.74	2.13	5.8	1006
R6	2.04	1.42	3.9	711
Avg.	2.56	1.74	4.51	879

For the Stoneridge Mall data for the database is collected using a Samsung Galaxy S4 smartphone. The collection path is 1417m as shown in Fig. 9a. For testing a Google Nexus 5 phone is used to walk ten paths, each of which spanned a majority of the mall. Table 1 shows results of the individual trials. Note that the average error computed in Table 1 is not weighted by the length of the paths. Of these ten paths, four are identical with the same start location and path walked as shown in Fig. 9b. These paths are labeled “D” for deterministic. The remaining six paths each have different start locations and trajectories. These are labeled “R” for random. Fig. 9c shows an example of a “R” path. Average length of paths is about 880m with an average position error of about 2.6m. Additionally Fig. 10 shows the error PDF for the system computed across all ten tests, followed by a breakdown of the error CDF in Table 2.



(a)

(b)

(c)

Figure 9: Example test paths in Stoneridge Mall. Red is the "ground truth" as measured by the foot mounted IMU. Blue is the path estimated by our system. 'x's indicate locations where a WiFi update is received. (a) Database collection path, (b) Path D4 from Nexus 5, (c) Path R5 from Nexus 5.

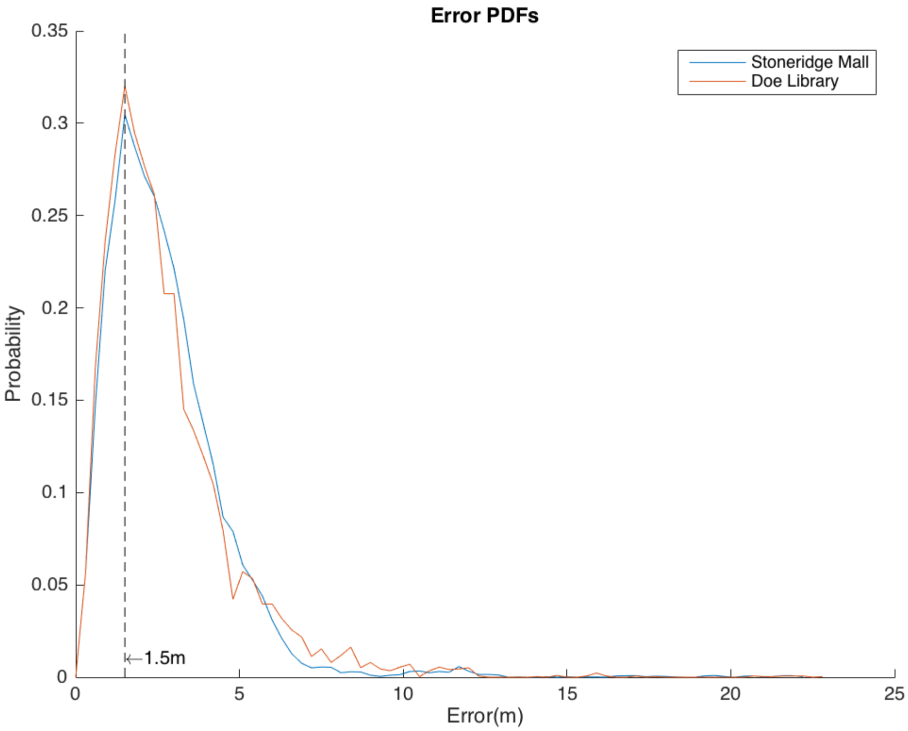


Figure 10: The error probability density function for Stoneridge Mall (blue) and the Doe Library (red).

Table 2: Breakdown of error

	50% (m)	75% (m)	90% (m)	95% (m)	99% (m)
Stoneridge Mall	2.25	3.37	4.66	5.5	10.34
Doe Library	2.14	3.39	5.13	6.42	10.19

For this experiment the database was collected over a month prior to final testing. This speaks to the stability of our maps, showing that a single walkthrough is capable of generating maps that are useable long after the collection was performed. Additionally tests were performed on a Saturday, when the mall was very busy, over the course of an entire day. This is important because it shows that our system is robust to various perturbations, such as inconsistent step lengths to avoid people, WiFi hotspots, or security guards on segways. In Fig. 11 we see that our system is able to provide repeatable results for similar paths despite these perturbations.

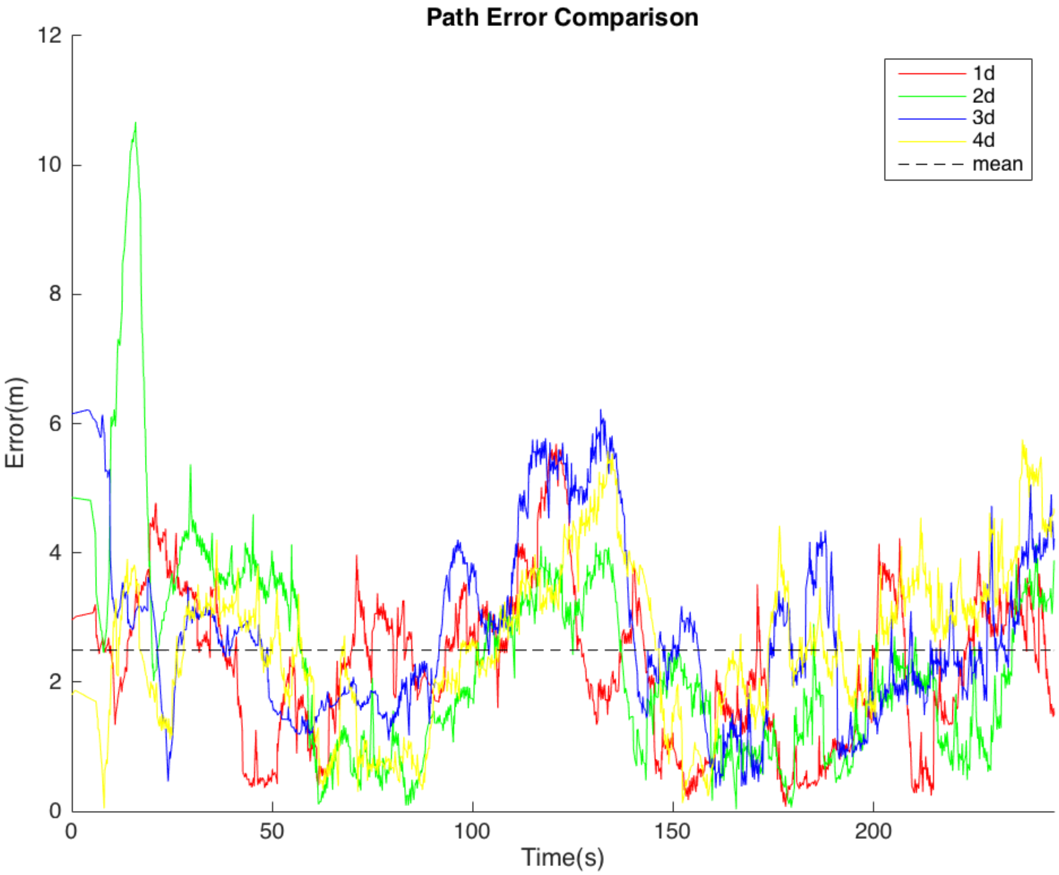


Figure 11: The error over time for deterministic paths in the Stoneridge Mall. This path is shown in Fig. 9b.

There are two main contributors of error for this experiment. The first is that one of the two initialization methods may return a location that is sometimes erroneous. This results in a position being reported between the two initialization locations until the incorrect particles die off. This can be seen in Fig. 12 where the error starts around 6m and then rapidly drops below 2m. As we see in Fig. 11 this variation in initial error does not affect our overall ability to track the user. The other significant contributor to error in this experiment is that our "ground truth" method is not quite accurate for such a large area. In Fig. 12 the areas of large error around 100s and 600s are both caused by the "ground truth" being reported in areas outside our floorplan. This type of error is also probably the largest contributor to the error discrepancy across the runs.

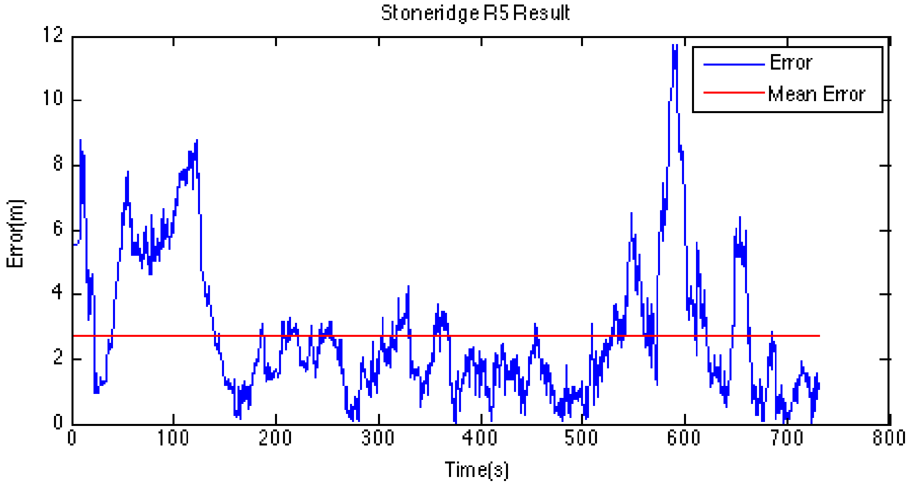
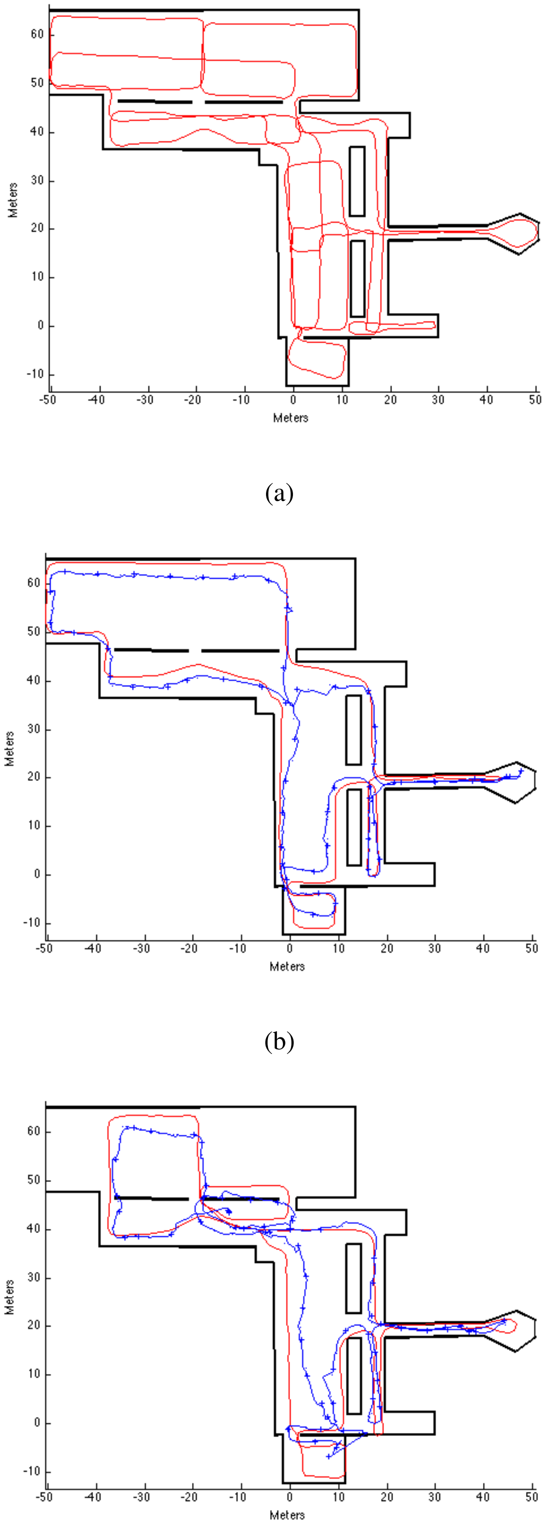


Figure 12: The error over time for run R5 in the Stoneridge Mall. This path is shown in Fig. 9c.



(a)

(b)

(c)

Figure 13: Example test paths in Doe Library. Red is the “ground truth” as measured by the foot mounted IMU. Blue is the path estimated by our system. ‘x’s indicate locations where a WiFi update is received. (a) Database collection path, (b) Path D2 from Nexus 5, (c) Path R2 from Galaxy S4.

6.2 Doe Library

Table 3: Experimental results for the Doe Library.

	Err. Mean(m)	Err. Std.(m)	90% Err.(m)	Length(m)
D1 N5	2.09	1.11	3.8	383
D2 N5	2.17	1.45	3.8	381
R1 N5	2.42	1.68	5.1	418
R2 N5	3.11	2.42	6.6	361
R3 N5	2.11	1.57	3.6	372
D1 S4	2.45	1.33	4.2	382
D2 S4	3.50	2.55	7.7	380
R1 S4	2.62	1.89	4.3	373
R2 S4	2.81	2.09	4.8	408
R3 S4	3.18	3.25	5.7	352
Avg.	2.65	1.93	4.96	381

For the Doe library data for the database is collected using a Google Nexus 7 tablet. The collection path is 880m as shown in Fig. 13a. For testing a Google Nexus 5 phone and a Samsung Galaxy S4 phone were each used to walk five paths for a total of ten paths. Table 3 shows the results of the ten trials. Note that the average error computed in Table 3 is not weighted by the length of the paths. Trials taken with the Google Nexus 5 and Samsung Galaxy S4 are labeled "N5" and "S4" respectively. All paths labeled "D" were taken along the same route as shown in Fig. 13b. Paths labeled "R" are all different with different starting points as shown in Fig. 13c. Average length of paths is about 381m with an average position error of about 2.65m. Additionally Fig. 10 shows the error PDF for the system computed across all ten tests.

Despite the smaller area of the library compared to the mall, it is actually the more difficult of the two test cases. There are several factors that contribute to this. First there are very few WiFi access points and very large open spaces. This means that the WiFi signal strengths are fairly uniform for wide areas making it difficult to use them for localization. Additionally there is significantly less metal in library compared to the Stoneridge Mall, reducing magnetic perturbation. Finally compared to the mall where the heterogeneity of stores provides significant visual differences between locations, the library has many repeated patterns that make it difficult to visually differentiate between areas.

While Table 2 shows the errors for the library are in line with those obtained at the Stoneridge Mall, it is unlikely that the "ground truth" system is a large contributor of error for this location. A larger contributor to error in this location is the drift of particles towards the center of the room due to the low magnetic and WiFi signal variation. This can be seen in Fig. 14 where there are two areas, starting at 120s and 200s, of large variation between paths. These areas correspond to the two large rooms in the Doe Library where the low variation makes it difficult to discern the correct position of the user.

Fig. 15 shows a comparison of results performed using the individual sensors compared to the combined system. These results are taken across identical paths in the Doe Library. The average error for magnetic alone is 6.2m, for WiFi alone is 9.8m, and for magnetic and WiFi combined is 3.0m, illustrating the benefit of sensor fusion.

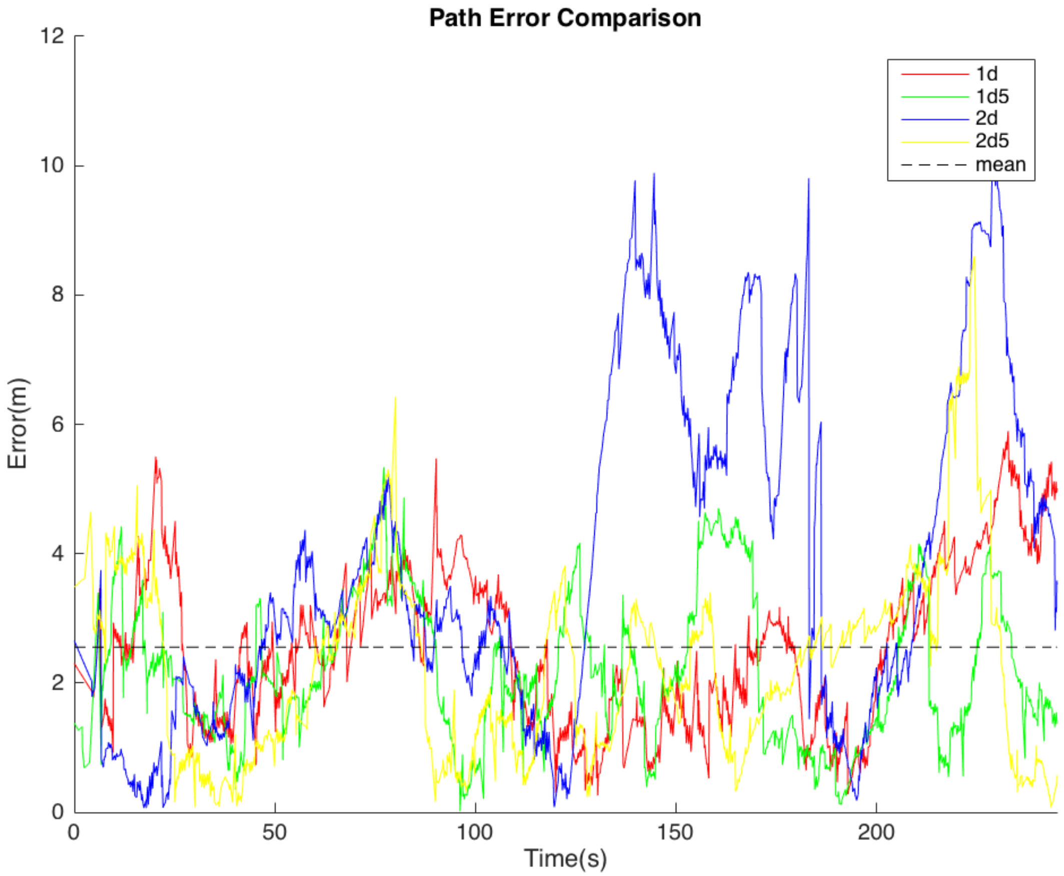


Figure 14: The error over time for deterministic paths in the Doe Library This path is shown in Fig. 13b.

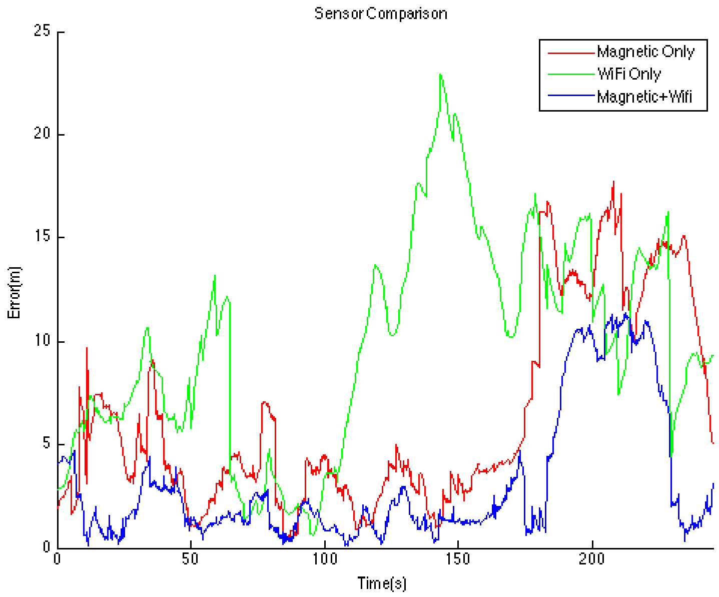


Figure 15: A comparison between using magnetic or WiFi sensors alone and fusing magnetic and WiFi measurements.

7 Conclusion

We have presented a complete system for performing indoor localization using a database collected by a smartphone in a single walkthrough of the building. Our system uses little additional hardware and provides a cheap and easy solution for indoor localization in a variety of environments. We have verified localization capabilities in two very different environments, both of which have practical use cases for this technology. We have been able to demonstrate an average localization error of 2.6m in both environments and showed that our databases are valid for multiple devices.

In the future we plan to examine the possabilty of removing the foot-IMU from the system using vision based SLAM. Vision based SLAM could be used to reconstruct a path from recorded data, allowing a database to be built with only a smart phone. Further we would like to investigate using interpolation between fingerprints to provide more accurate localization accuracy. Richter et al. showed that using a Gaussian Process with a Matérn kernal is able to provide improved positioning results for sparse RSSI fingerprint maps [30]. Since we generate maps with only a single walkthrough of the building our fingerprint maps are generally very sparse. Using a Gaussian Process for interpolation could help to fill in the gaps created by this sparsity.

References

[1] P. Levchev, M. Krishnan, C. Yu, J. Menke, and A. Zakhor, ”Simultaneous Fingerprinting and Mapping for Multimodal Image and WiFi Indoor Positioning” *IPIN2014, Busan, Korea* October 2014

[2] H. Liu, ”Survey of Wireless Indoor Positioning Techniques and Systems”, *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE*, Volume:37 , Issue: 6, 2007

[3] R. Harle, ”A Survey of Indoor Inertial Positioning Systems for Pedestrians”, *Communications Surveys & Tutorials, IEEE* Volume: 15, Issue 3, 2013

[4] A. R. Pratama, ”Smartphone-based Pedestrian Dead Reckoning as an indoor positioning system”,*International Conference on System Engineering and Technology (ICSET)*, 2012

[5] M. Holcik, ”Indoor Navigation for Android,” *Masters Thesis, Faculty of Informatics, Masaryk University*, 2012.

[6] Y. Jin, H.Toh, W. Soh, and W. Wong, ”A robust dead-reckoning pedestrian tracking system with low cost sensors,” *IEEE international Conference on Pervasice Computing and Communications (PerCom), Seattle, WA, USA*, 2011

[7] F.H.T. Pinto, ”An indoor localization solution for mobile devices”, *Master’s thesis, FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO*,2011

[8] E. Foxlin, ”Pedestrian tracking with shoe-mounted inertial sensors”, *Computer Graphics and Applications, IEEE* , Volume:25 , Issue: 6, pp 38-46, 2005

[9] P. Bolliger, ”Redpin - Adaptive zero configuration indoor localization through user collaboration”,*Workshop on Mobile Entity Localization and Tracking in GPS-less Environment Computing and Communications Systems (MELT), San Francisco* 2008

[10] H. Lin, Y. Zhang, M. Griss, and I. Landa ?WASP: An Enhanced Indoor Locationing Algorithm for a Congested WiFi Environment?, *The Second International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments (MELT) , Orlando, FL* 2009

[11] B. Ferris, D. Fox, N. Lawrence, ”WiFi-SLAM Using Gaussian Process Latent Variable Models”, *IJCAI* 2007

[12] P. Bolliger, K. Partridge, M. Chu, and M. Langheinrich, ”Improving location fingerprinting through motion detection and asynchronous interval labeling”,*Lecture Notes in Computer Science* Volume 5561, pp 37-51 2009

[13] P. Bahl, V. Padmanabhan, ”RADAR: an in-building rd-based user location and tracking system”,*INFOCOM*, 2000

[14] Z. Wu, E. Jedari, B. Liu, R. Rahidzadeh, M. Ahmadi, ”Particle Filter and Extreme Learning Machine Based Indoor Localization System”, IPSN 2015, Seattle, WA

[15] H. Zou, H. Jiang, X. Lu, Z. Chen, J. Chen, J. Zhu, Y. Luo, L. Xie, Y. Chai Soh, M. Jin, and C. Spanos, ”WiFiGenius: An Accurate and Reliable WiFi-based Indoor Localization and Navigation System” IPSN 2015, Seattle, WA

[16] A. Hallquist and A. Zakhor, ”Single view pose estimation of mobile devices in urban environments,” *IEEE Workshop on the Applications of Computer Vision (WACV), Clearwater, FL, USA*, 2013

[17] J. Z. Liang, N. Corse, E. Turner, and A. Zakhor, ”Image based localization in indoor environments,” *International Conference on Computing for Geospatial Research and Application (COM. Geo) ’13, San Jose, CA, USA*, 2013

[18] J. Z. Liang, N. Corse, E. Turner, and A. Zakhor, ”Reduced-complexity data acquisition system for image-based localization in indoor environments,” *International Conference on Indoor Positioning and Indoor Navigation (IPIN ’13), Montbeliard-Belfort, France*, 2013

[19] D. G. Lowe, ”Distinctive image features from scale invariant key points”, *International Journal of Computer Vision*, pp 91-110 2004

[20] J. Haverinen, ”Global indoor self-localization based on the ambient magnetic field”,*Journal of Robotics and Autonomous Systems*, Volume 57 Issue 10, pp 1028-1035 2009

[21] J. Chung, M. Donahoe, C. Schmandt, I. Kim, P. Razavai and M. Wiseman, ”Indoor location sensing using geo-magnetism”, *9th international conference on Mobile systems, applications, and services* 2011

[22] W. Storms, J. Shockley, J. Raquet, ”Magnetic field navigation in an indoor environment”, *Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS)*, 2010

[23] M. Muja and D. G. Lowe, ”Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration,” *in International Conference on Computer Vision Theory and Applications*, 2009.

[24] J. Zhang, A. Hallquist, E. Liang, and A. Zakhor, ”Location-Based Image Retrieval for Urban Environments,” *in International Conference on Image Processing*, 2011.

[25] P. Prichett, A. Zisserman, ”Wide baseline stereo matching” *Sixth international conference on computer vision*, 1998

[26] P. Levchev, C. Yu, M. Krishnan, A. Zakhor, ”Indoor WiFi Localization with a Dense Fingerprint Model” UC Berkeley, Tech. Rep. Apr. 2014 [Online]. Available: http://www-video.eecs.berkeley.edu/papers/plamen/globe14-localization-0401.pdf

[27] A. Serra, T. Dessi, D. Carboni, V. Popescu, and L. At- zori, ”inertial navigation systems for user-centric indoor applications,”*Networked and Electronic Media (NEM ’10) Summit, Barcelona, Spain*, 2010.

[28] O. J. Woodman, ”An introduction to inertial navigation,” University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-696, Aug. 2007. [Online]. Available: http://www.cl.cam.ac.uk/techreports/UCAM- CL-TR-696.pdf

[29] A. Rai, K. Chintalapudi, V. Padmanabhan, R. Sen, ”Zee: Zero-Effort Crowdsourcing for Indoor Localization”, *international conference on Mobile computing and networking(Mobicom)*, 2012

[30] P. Richter, A. Pena-Torres and M. Toledano-Ayala, ”A Rigorous Evaluation of Gaussian Process Models for WLAN Fingerprinting”, *IPIN2015, Alberta, Canada*, October 2015