SmartSLAM – an efficient smartphone indoor positioning system exploiting machine learning and opportunistic sensing

R. M. Faragher, R. K. Harle.
Digital Technology Group, Computer Laboratory,
University of Cambridge
rmf25@cl.cam.ac.uk

Ramsey Faragher is a Senior Research Associate in the University of Cambridge working on GNSS-denied positioning, sensor fusion and machine learning. He is also an Associate Editor for the Royal Institute of Navigation journal. Prior to this he was a Principal Scientist at the BAE Systems Advanced Technology Centre where he developed the NAVSOP opportunistic positioning system and other GNSS-denied tracking technologies. In 2009 Ramsey won the BAE Systems Early Career Engineer of the Year award for developing NAVSOP.

Robert Harle is a Senior Lecturer in the University of Cambridge with research interests in positioning, sensor fusion, and wireless sensor networks. He has worked on indoor positioning since 2000, developing a series of infrastructure-based and infrastructure-free solutions.

Abstract: **The most promising solution to the ubiquitous positioning problem is the smartphone, and many smartphone-based indoor tracking methods exist today. To ensure consumer acceptance of the technologies, it is critical that these systems do not have a significant effect on the battery life of the device. Methods exploiting signal fingerprinting have been shown to provide good performance with low processing overhead but require prior surveying. Methods exploiting opportunistic sensing and machine learning techniques such as Simultaneous Localization and Mapping (SLAM) need no prior data but at the cost of high computational load. This paper describes a smartphone-based indoor positioning system that exploits a new intelligent filtering approach to reduce this computational load. SmartSLAM moves between different sensor fusion algorithms depending on the current level of certainty in the system, reducing the computational load of the tracking engine, maintaining good positioning performance, improving battery life and freeing CPU cycles for foreground processes.**

*Key words**: Opportunistic radio positioning, SLAM, indoor navigation, smartphone positioning***

I. Introduction

There is a long standing interest in ubiquitous positioning, or the ability to determine a location in any environment, outdoors and indoors. This desire has increased in recent years due to the developments in machine-to-machine interfaces and the "Internet of Things", two new areas of development where accurate and autonomous positioning of devices will be an important feature. We have all become used to the availability and performance of Global Navigation Satellite Systems (GNSS) for accurate outdoor radio positioning with a reasonable degree of reliability and availability. However indoor radio positioning is more challenging since GNSS signals do not penetrate buildings well, and indoor positioning therefore relies typically on local infrastructure and other support. Indoor radio positioning is available today via databases of WiFi or Cellular signal strength fingerprints collected, managed and provided by a third party provider such as Skyhook [2]. These systems have two constraints: the area must already have been surveyed, and the user must have a data connection available to them.

An ideal system would not rely solely on these constraints, but would develop its own database in real time during operation using techniques such as Simultaneous Localization and Mapping. Such a system has been described and demonstrated by Faragher in previous work [3]. The benefits of this system are significant - it can provide situational awareness and asset tracking in new and unknown environments for the military, emergency services, lone workers, security personnel and autonomous vehicles. This method does not require a data link to function, nor any prior surveying of the radio environment, nor any other prior knowledge such as a floor plan or database of signal fingerprint maps (it can, however, incorporate these data if they are available). This method can be used to rapidly survey an area and generate a signal fingerprint database for others users to exploit if periodic data links are available. Its primary disadvantage is resource consumption: a device continuously performing SLAM will typically require significant computation. In this paper we develop a flexible SLAM scheme that can move between different fusion algorithms in order to improve the resource consumption of a smartphone-based indoor positioning system.

II. Indoor Positioning

A. Smartphone positioning

Smartphone positioning is provided today by three main approaches – GNSS, Cell-ID and fingerprinting.

The GNSS chips in smartphones are at the cutting edge of consumer-grade GNSS receiver design. They are typically assisted by the cellular or WiFi communication links to permit rapid acquisitions without downloading ephemerides using the GNSS signals themselves. They also typically contain thousands of correlators per channel to permit parallel signal searches of all possible code phase offsets and Doppler shifts on each satellite. Even so, the weak GNSS signals can be rendered useless by buildings and even dense foliage. Their application in indoor environments is limited.

Cell-ID is the simplest of cellular positioning techniques, whereby the position and approximate coverage area of the cellular base station serving the smartphone at that given moment is provided to the user as their position estimate. The accuracy of this approach varies wildly depending on the power of the serving base station, with accuracies ranging from a few hundred metres to dozens of kilometres [4].

Fingerprinting is currently the most accurate indoor positioning scheme deployed on standard smartphones. Radar [5], Horus [6] and Compass [7] are good examples of WiFi fingerprint schemes. A fingerprint refers to the pattern of radio signal strength measurements that is made at a given location in space and consists of a vector of signal identity information (such as cellular Cell-IDs, or WiFi MAC addresses) and a corresponding vector of Received Signal Strength (RSS) values. Typically WiFi signals alone provide the fingerprints, but cellular measurements and data from a smartphone magnetometer could be stored too. WiFi signals provide greater dynamic range than cellular signals because they are short range transmitters and are generally deployed inside buildings. They therefore exhibit greater anisotropy than the external, and often very distant, cellular signal sources. Magnetometer data provides a high dynamic range on a very fine scale (see Figure 1) but can only provide a single contribution to the fingerprint vectors, which will be dominated by the many WiFi and cellular measurements available in a typical metropolitan indoor environment. As a receiver moves through a complex signal environment, such as a building full of walls and objects, the RSS of any non-line-of-sight signal can vary rapidly on a fine spatial scale (metre level) as that signal penetrates different media and interacts with different objects as it moves along different paths through the building (see Figure 3). Fingerprinting relies on these RSS values varying rapidly on the spatial scale, but only very slowly over time, such that a receiver coming back to a fingerprint location in the future should record the same RSS measurements, within the limitations of measurement noise (see Figure 2).

A database of fingerprints recorded at known locations during a manual survey can be used in the future to estimate the position of a user based on comparisons of the fingerprints they capture and those in the database. In reality fingerprints inevitably degrade over time as inevitable changes occur – the density of people within the building at different times, the positions of furniture, even the positions of walls and partitions. This means that traditional fingerprinting schemes require regular re-surveying to ensure the accuracy of the system. WiFi fingerprints can also be heading dependent [7], as the user's body is typically an excellent attenuator at WiFi frequencies.

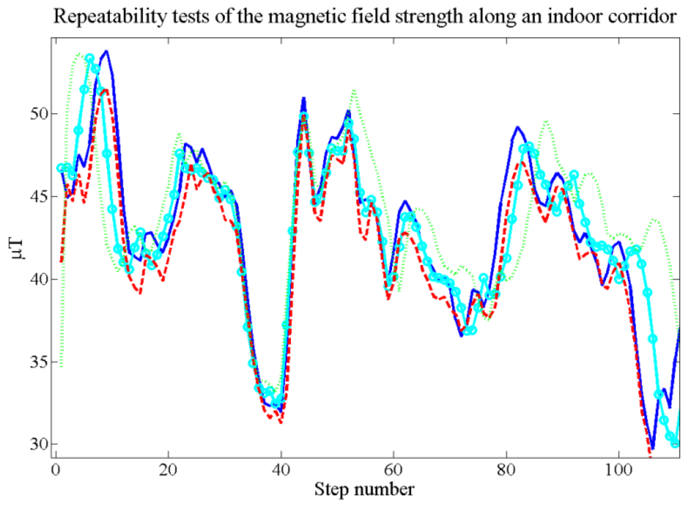

Figure 1. Magnetic field strength along a corridor in an office environment measured on four different days using a smartphone. The repeatability is clear. The slight spreading between traces is caused by slight differences in step length.

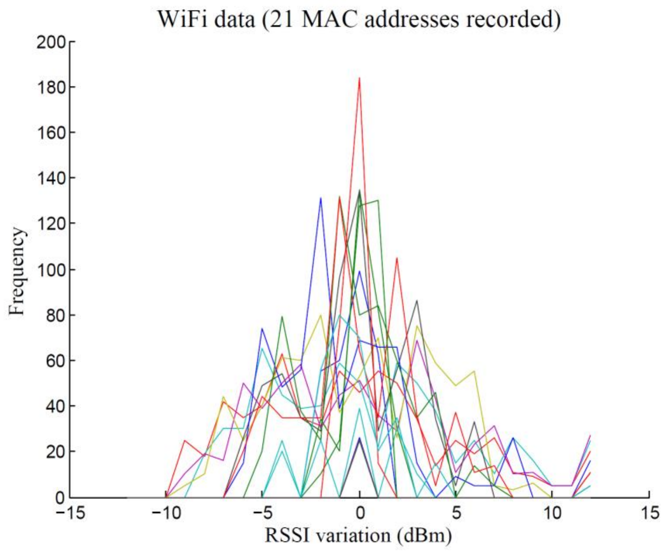

Figure 2. Received signal strength distributions for 21 WiFi access points recorded over ten minutes from a fixed location

B. Alternative indoor positioning approaches

Many indoor positioning solutions exist that have do not need access to signal fingerprint databases, but they introduce their own restrictions. A standard approach is to deploy specific infrastructure to support robust positioning with a known performance and availability envelope [8][9][10]. The drawback of these systems is that they do not scale up to provide ubiquitous positioning to consumers globally. An alternative approach is to use low drift inertial measurement systems to track users during periods of GNSS loss. The cost of inertial measurement units is however inversely proportional to their performance, and the low cost devices deployed in consumer smartphones cannot provide useful tracking via traditional strap down processing [11] for timescales beyond about one second. The application of Zero Velocity Updates for shoe-mounted low cost inertial measurement units can provide excellent (low) drift rates with low cost devices [12], but mounting smartphones to shoes is not a practical solution to the consumer indoor positioning problem. In the future we may see an increase in “smart clothing”, and shoes may contain sensors to assist in a ZUPTs solution for indoor tracking but they are unlikely to be as commonplace as the already-well-established smartphone.

III. SLAM ON A SMARTPHONE

Simultaneous Localization and Mapping (SLAM) [13][14][15] is a well-established technique in the robotics industry to allow a platform to map out an environment without a confident absolute knowledge of its own location at any given point. The platform only has good knowledge of its relative motion, and a model of how its relative position estimates are expected to degrade over time (i.e. gyro drift, wheel slip, etc.). Traditionally the positions of landmarks (objects) in the environment are measured using lasers or some other high-resolution sensor and these provide accurate relative measurements of the “anchors” in the world that the system can rely on – a critical assumption is that these landmarks do not change position during the journey.

The key to SLAM is to maintain correlations between measurements of landmarks and the estimated position and pose of the platform, such that when the platform revisits a location it can recognize this fact by recognizing landmarks. It can then observe and correct the relative positioning error that has accumulated throughout the journey. This new information can also be used to correct the historical path that the platform has traced out. The locations of the landmarks in memory also update, as they are positioned in the system relative to the historical path itself. Following these “loop closures”, corrections can be made to the map of the environment and the path taken through it [13].

SLAM can be applied to the smartphone indoor positioning problem, as has been demonstrated by a number of authors [3][16][17]. Here the measurements are not of distances to physical objects, but of the fingerprints recorded by the smartphone at the user location. Most smartphone SLAM approaches rely on an offline SLAM optimization scheme, but the approach taken here builds on the authors’ interests in online schemes that can provide instant feedback to the users.

An offline optimization scheme should always provide the best solution available, as it iterates towards the global minimum in the search space representing the joint estimator of the path taken through space and the underlying ”map” that the measurements have been drawn from. We assume that any large scale deployment of a consumer smartphone indoor positioning system will provide a backend service to provide these offline post-processing stages to ensure that the data gathered by users produces the best fingerprint maps possible for future use, while the users themselves can still benefit from the best positioning possible in real time without needing a live data link at every measurement epoch. The datasets generated by the movement of individual users also merely represent a subsampling of the entire signal coverage map of any WiFi access point or other signal. Amalgamating multiple journeys and using a regression scheme such as Gaussian Processes [18][19] allows this ”training data” to be used to predict the entire coverage map.

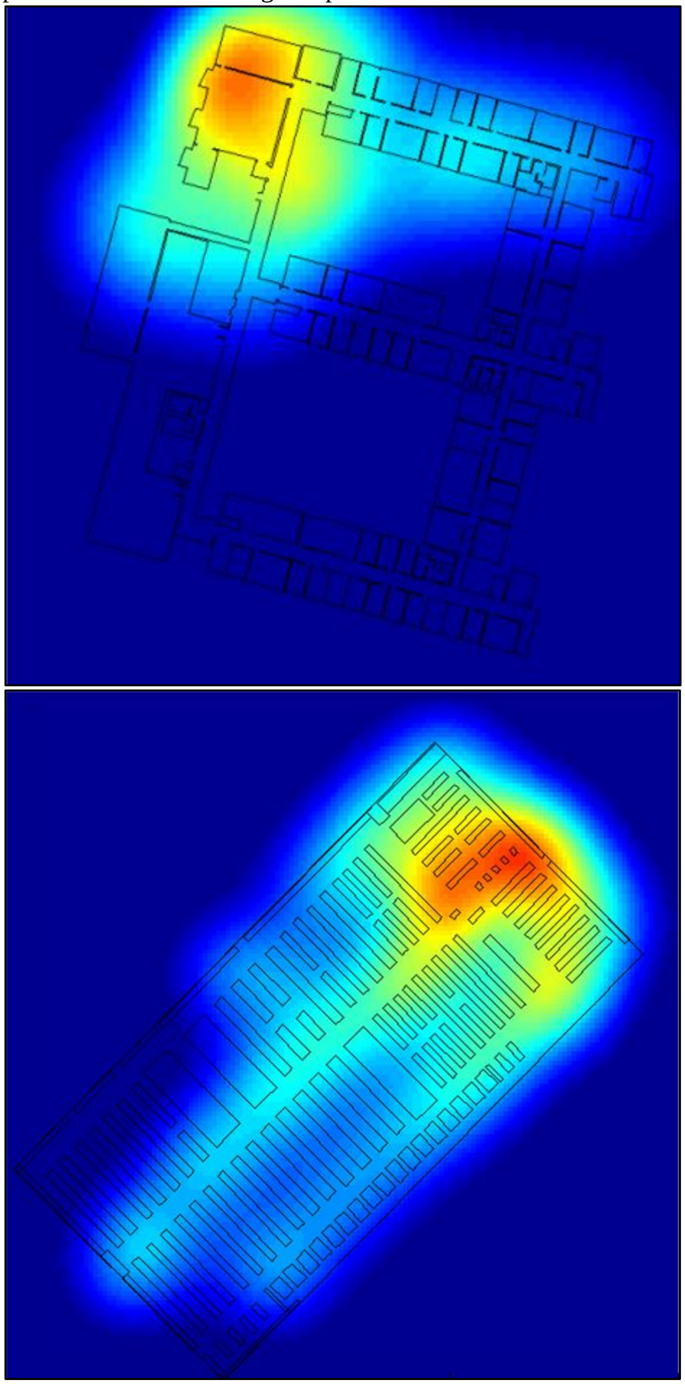
Figure 3 Signal strength coverage maps for a single WiFi access point in the William Gates Computer Laboratory in Cambridge (top) and in a Cambridge supermarket (bottom). The supermarket has roughly double the floor area as the Computer Laboratory. The marked difference in signal coverage area is caused by the large number of internal walls in the laboratory.

IV. Concept of Operation

The user model we assume here is a pedestrian moving from open sky conditions into an indoor environment and losing GNSS coverage. There will be various types of indoor environment, ranging from those with rich existing signal fingerprint databases and digital vector floor plans, through partially-surveyed/crowd-sourced environments, to completely new environments with no prior fingerprint databases available. Ideally an indoor positioning scheme would offer good positioning performance in all of these situations, whilst minimising resource consumption. We assume that a service provider maintains databases of signal fingerprint maps and uses opportunistic measurement data from anonymous users to maintain and update them. This service provider can employ a computationally-intensive SLAM scheme to reprocess crowd-sourced data offline. This ensures that the crowd-sourced fingerprint maps are as accurate as possible without resorting to painstaking manual surveys.

V. SmartSLAM - Reducing the computational load

The drawback of indoor positioning on smartphones exploiting the inertial measurement unit and constant sensor recordings from radio sensors is the effect on battery life. Some IMU chipset manufacturers are helping to reduce this problem by offering low power step detection processing at the hardware level [20][21]. The computational load, and so power consumption, associated with the software processing at the application level is the responsibility of the software application programmer, and the focus of this research.

SmartSLAM moves between four operating regimes as described below.

PDR-only: a step-and-compass Pedestrian Dead Reckoning (PDR) solution

FEKF: a modified Extended Kalman Filter that uses supplied fingerprint maps and PDR

FEKFSLAM: a low-cost online SLAM-like algorithm that uses both PDR and fingerprint measurement sequences when fingerprint maps are not available

DPSLAM: a more costly online SLAM algorithm using a particle filter, PDR, fingerprinting, and magnetic anomaly measurements

The detail of the decision process, which depends on the level of knowledge of both user and environmental parameters, is demonstrated in the flow diagram of Figure 4.

The SmartSLAM state vector tracks the three spatial dimensions (x, y, z), the user's average step length (s), the compass heading (ϕ) and the heading bias (ϕ_b).

$$\hat{\mathbf{x}} = [x, y, z, s, \phi, \phi_b]^{\mathrm{T}}$$

The different states are updated by a series of different algorithms, as discussed below. The step length and heading bias are referred to in this paper as the 'PDR parameters'.

VI. SmartSLAM Operating Regimes

Here we describe the four SmartSLAM operating regimes in more detail. Our discussion is in order of increasing complexity: PDR, FEKF, FEKFSLAM, DPSLAM.

A. Pedestrian Dead Reckoning (PDR)

A PDR scheme is a core part of this concept. In a smartphone-based SLAM system, odometry is provided by the inertial sensors: accelerometers, gyroscopes, magnetometers and barometer. The quality of these sensors is not high enough to allow a free-running strapdown inertial navigation system without resulting to ZUPTs or some other method of observing and correcting the drifting sensor biases. Typically, smartphone pedestrian navigation systems exploit a "step-and-compass" odometry approach where the accelerometers are used to detect walking motion and the user's position estimate is updated one step at a time along the heading provided by the magnetometers, accelerometers and gyroscopes [3].

For these systems, the step length of the user needs to be calibrated somehow, either using confident GNSS fixes when they are available [3] or by direct user input. It has been shown that the step length typically varies with stepping frequency [25]. However, given a calibrated floor plan and coarse initial position estimate within that map, the user step length and any compass bias can easily be estimated using a wall-sensitive particle filter [26].

The user heading can be determined using a tilt-compensated three-axis magnetometer and smoothed using a thee-axis gyroscope. This smoothing can be provided using a complementary filter, Kalman filter, or other fusion scheme. The magnetic declination may be unknown at the user's location, and so this bias can be determined in the same manner as the user step length – i.e. by exploiting GNSS measurements, floor plans, or existing signal strength maps whenever they are available.

It is possible to process the accelerometer and barometric data within a moving window to assess the likelihood of a user traversing floors by staircases, escalators and elevators [22][23].

Humans may undertake a variety of motions such as backward steps, side steps, crawling, running, jogging, jumping, shuffling, etc. Recognising these different gaits is a well-researched aspect of pedestrian dead reckoning and there are many established methods based on machine learning or signal processing [27][29]. For the purposes of this work, which concentrates on SLAM methods and opportunistic positioning, the user attempted to always walk forwards in the direction they were facing in order to reduce the effects of gait recognition algorithms on the experimental results presented here. In the context of SmartSLAM, a PDR-only scheme is applied when GNSS is first lost to account for the possibility that the user is only briefly denied GPS by moving into a building for the purposes of a simple rapid task like making an enquiry at a reception desk or buying an item in a small shop. Good modern PDR schemes can operate with drift rates of a few percent of distance travelled [24]. We propose therefore that once the position estimate error has grown by 2-3 metres (which may only occur after around fifty steps or more), the system moves to a new regime, as it is clear that the user is undergoing extended indoor operations.

B. Fingerprint Extended Kalman Filter (FEKF)

The Fingerprint Extended Kalman Filter can be applied when a trusted database of fingerprints is available. It incorporates signal strength maps and measurements into the update step of an Extended Kalman Filter (EKF). We define a *fingerprint* to be a vector of *N* Received Signal Strength (RSS) measurements received from *N* distinct WiFi access points.

$$\mathbf{w} = [RSS_1, RSS_2 \dots RSS_N]^{\mathrm{T}}.$$

The FEKF uses a standard EKF prediction stage [30] with a process model provided by the step and compass PDR scheme discussed above. In the standard EKF formulation, the measurement update stage at time *t* requires the population of a measurement vector $\mathbf{z}_t$, a measurement prediction vector $\mathbf{h}_t$, and a Jacobian matrix $\mathbf{H}_t$. In the FEKF, when a WiFi scan returns a set of RSS measurements, the measurement vector $\mathbf{z}_t$ is populated accordingly:

$$\mathbf{z}_t = \mathbf{w}.$$

The variances σ_R^2 of these noisy RSS measurements populate the measurement noise matrix

$$\mathbf{R} = \left[\sigma_{R\,1}^2, \sigma_{R\,2}^2, \dots, \sigma_{R\,N}^2\right]\mathbf{I}.$$

In the standard EKF scheme, predictions of the values in the measurement vector are calculated using an analytic function and the current values of the state vector. Here this could naively be provided using the radio free space path loss equation, but only if the locations of the WiFi routers and their transmit powers are known, which is rarely the case. A radial free space path loss model is also expected to be a poor estimator of WiFi signal strength in cluttered indoor environments. Therefore, no analytic measurement model is available in the FEKF scheme to populate **h**. Predicted values of each RSS measurement are instead drawn from the signal strength maps **M** previously generated by manual surveys or SLAM.

$$\mathbf{h}_t = [\mathbf{M}_1(x,y), \mathbf{M}_2(x,y) \dots \mathbf{M}_N(x,y)]^{\mathrm{T}};$$

These predicted values do of course have some error associated with them which must be accounted for within the Kalman Filter's probabilistic framework. Conveniently, generating the signal strength maps using Gaussian Processes regression [18] ensures that each position in the signal strength map has associated with it both a predicted value and a variance providing a measure of uncertainty on that value. The variances σ_V^2 associated with the predicted signal strength values extracted from each map are adding to the **R** matrix, in effect decreasing the confidence on the radio measurements.

$$\mathbf{R} = \left[\sigma_{R\,1}^2 + \sigma_{V\,1}^2, \sigma_{R\,2}^2 + \sigma_{V\,2}^2, \dots, \sigma_{R\,N}^2 + \sigma_{V\,N}^2\right]\mathbf{I}.$$

Alternatively, the **R** matrix can be constructed as a full covariance matrix by calculating the correlation coefficients for all pairs of WiFi signal strength maps. This will allow any correlated behaviour between WiFi routers (due to their relative locations) to be captured and accounted for within the filter update stage. These values can all be calculated by the service provider and passed to the users as needed along with the prior maps.

Within an EKF estimation scheme a Jacobian matrix is required in order to allow any differences between the measurements and predictions to provide a weighted update to the states of interest. The Jacobian is normally generated using the derivative of a mathematical model describing the relationship between the measurement and the system's states of interest but here we have no such function.

To solve this problem we introduce the concept of the *Jacobian map*. Given a dense WiFi signal strength map for each access point (for example generated using Gaussian Processes regression following an offline SLAM calculation using a rich set of training data gathered from journeys of previous users in the building of interest) the derivatives of these maps with respect to the states of interest (for example Cartesian *x* and *y* coordinates) can be generated and stored along with the normal fingerprint maps. The FEKF scheme can therefore use lookup routines to generate the values for the **h** vector and **H** matrix. Not only does this provide the Jacobian matrix terms we need, but it reduces the computational load of the simple EKF algorithm even further. For each WiFi signal strength map, **M,** the spatial Jacobian maps can be generated in advance using

$$\mathbf{J}_{i,x} = \frac{\boldsymbol{d}}{\boldsymbol{dx}}\left(\mathbf{M}_i(x,y)\right)$$

$$\mathbf{J}_{i,y} = \frac{\boldsymbol{d}}{\boldsymbol{dy}}\left(\mathbf{M}_i(x,y)\right)$$

And provided to the user such that the spatial *x* and *y* terms of the **H** matrix can be populated accordingly using the same database lookup coordinates that were used to extract the fingerprints and their variances.

$$\mathbf{H}(i,\mathbf{1}) = \mathbf{J}_{i,x}(x,y)$$
$$\mathbf{H}(i,\mathbf{2}) = \mathbf{J}_{i,y}(x,y)$$

Traditionally a scheme such as K-Nearest Neighbour [5] classification would be used to provide a user with position estimates if a rich fingerprint map is available. The advantage of an EKF-based scheme is the use of the process model to restrict unrealistic large jumps in position as the user moves through an environment (the KNN classification returns position estimates that are uncorrelated with the previous state of the system unless further constraints are applied to the KNN output). An EKF scheme should inherently produce a smoother user path through the environment by fusing PDR data with successive WiFi measurements.

The performance and reliability of the FEKF scheme is dependent on the validity of its assumptions and the accuracy of the prior maps (mean, variance, and gradients for each WiFi access point). The consistency and stability of the filter can be monitored using the normalised innovation squared, and the system can switch to a more flexible algorithm (DPSLAM) if necessary.

If the prior WiFi maps are generated using a regression or interpolation scheme, then their values will be correlated over some spatial scale. Passing correlated data into the FEKF scheme can result in overconfident terms in the covariance matrix. However the typical correlation length scale generated during Gaussian Processes optimisation is around 2-3 metres for indoor WiFi maps (we have observed this ourselves and it is also confirmed by Ferris [16] and Huang [17]). Therefore if we assume typical walking speeds of around 1 ms^{-1}, then limiting the WiFi scan update rate to around 5 seconds between scans will help to ensure that fingerprints extracted from the prior maps have uncorrelated errors.

C. FEKFSLAM

If there are no fingerprints available but the user's PDR parameters are well known (e.g. the step length and any compass bias were recently calibrated during a period of GNSS availability) then we propose the use of the Fingerprint Extended Kalman Filter Simultaneous Localization and Mapping (FEKFSLAM) scheme. This is a highly efficient *approximation* of the full SLAM solution which ensures low computational load.

FEKFSLAM maintains only a *single* hypothesis of the state vector (unlike the DPSLAM scheme discussed below) and requires a loop closure detection step at *every* measurement epoch. We use the 3-sigma covariance ellipse available from applying the FEKF scheme to restrict the amount of the user's history that must be searched for a loop closure. When historical positions do lie within the covariance ellipse of the current location, the age of the historical location is tested. Any locations that are less than a few steps old are ignored (else the routine will constantly be testing the previous few steps of the journey). The old (from times t_{old_i}) and new (at time t) positions are tested by generating a distance metric between their associated WiFi fingerprints. Here we use the normalised Euclidean distance,

$$D_i = \sqrt{\left(\mathbf{w}_t - \mathbf{w}_{t_{old_i}}\right)^{\mathrm{T}} N^{-1} \left(\mathbf{w}_t - \mathbf{w}_{t_{old_i}}\right)}$$

Where $\mathbf{w}_t$ and $\mathbf{w}_{t_{old_i}}$ are the WiFi fingerprint vectors captured at times t and t_{old_i} and N is the number of WiFi measurements in the fingerprint. If the two fingerprints being compared do not contain identical WiFi access point measurements, then only the matching pairs of measurements are used in the fingerprint comparison. If the fingerprint vector contains fewer than four radio measurements it is considered too weak a fingerprint to be used for a position update [28].

In our two test environments, we found that the Euclidean distance between WiFi fingerprints increased linearly with the spatial separation (Figure 5). This provides a useful empirical measurement model for loop closure of the form

$$\Upsilon = A + B \cdot r,$$

where Υ is the expected Euclidean distance, A and B are the typical best fit parameters of the linear empirical model and r is the spatial separation between the current position estimate and the historical position under test. If the user position is accurately known and this test is applied to a nearby historical location, then the fingerprint distance should be predictable (within the bounds of measurement noise) using the linear measurement model. If the calculated Euclidean distance is consistently lower than this prediction, then the true spatial separation is likely to actually be smaller, and vice versa. This test can be performed for each historical location that lies within the current spatial covariance ellipse, and so provide values for the **z**, **h** and **H** terms of a Kalman Filter update. Note that the plots of Euclidean distance with spatial separation exhibit noise about the lines of best fit. This noise is however white for moderate and short separations (see Figure 6) and so this approach lends itself well to the Kalman-based filtering scheme we describe.

It is interesting to note that the gradient of the measurement model was shallower in the supermarket environment than the office environment. This is likely caused by the office environment being denser with many walls spanning the full floor to ceiling distance (the supermarket was characterised by aisles with large gaps to the ceiling). The best-fit parameters for this linear model are therefore expected to vary slightly for different buildings (due to wall thicknesses, density of objects, etc.), but can be easily gathered on-the-fly during a journey. To do this a moving window processing the most recent 15 metres or so can be used to gradually build up a dense scatter plot of the type shown in Figure 5. The two parameters can then be tracked as extra states within the FEKFSLAM state vector.

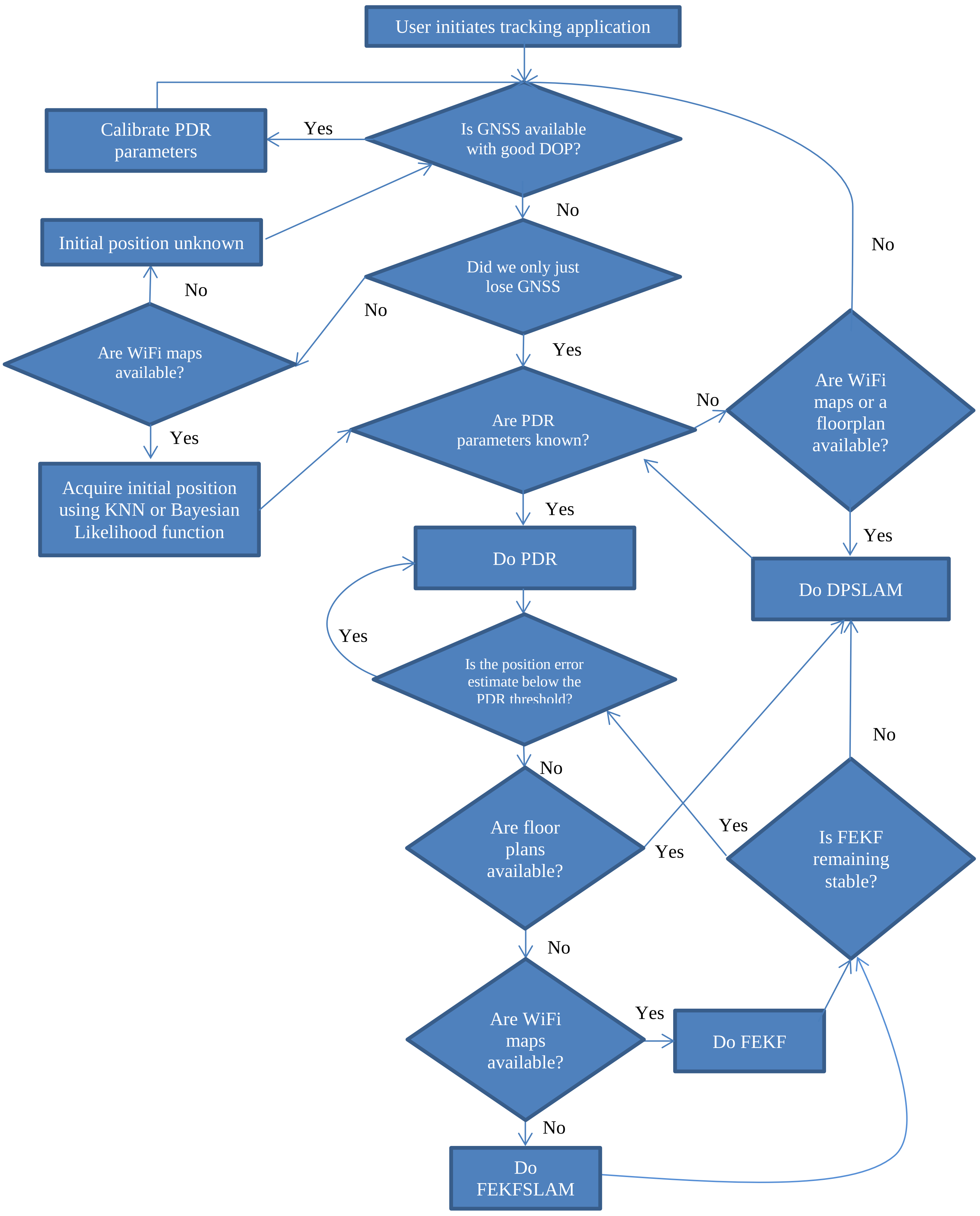
User initiates tracking application
Calibrate PDR parameters
Yes
Is GNSS available with good DOP?
No
Initial position unknown
Did we only just lose GNSS
No
No
No
Are WiFi maps available?
Yes
Are PDR parameters known?
No
Are WiFi maps or a floorplan available?
Yes
Acquire initial position using KNN or Bayesian Likelihood function
Yes
Yes
Do PDR
Do DPSLAM
Yes
Is the position error estimate below the PDR threshold?
No
No
Yes
Are floor plans available?
Yes
Is FEKF remaining stable?
No
Are WiFi maps available?
Yes
Do FEKF
No
Do FEKFSLAM

Figure 4. The decision process for SmartSLAM (see Section VII)

Figure 7 shows the evolution of the estimates of the measurement model parameters as the user moved through the environment. A moving batch window processing the fingerprint data over the most recent 30 steps was used to calculate pairs of Euclidean distance and actual separation in order to build up scatter plots like those shown in Figure 6.

The first order polynomial fit to this growing dataset was calculated at each step, and the best fit parameters are plotted in Figure 7 for ten experiments involving two distinct handsets: a Nexus 4 (solid blue lines in the office, red lines in the supermarket) and a Galaxy Nexus (dashed green lines). The results show that the FEK-FSLAM measurement model parameters are well estimated within the first 100 steps of the journey (equivalent to moving around 70 metres). They also demonstrate the variation in best fit parameters between devices, environments, and over time, and show that maintaining an online estimate of the FEKFSLAM measurement model parameters ensures optimal performance.

Once the measurement model parameters have been determined, the measurement vector $\mathbf{z}_t$ is populated whenever a set of historical positions is overlapped by the FEKFSLAM covariance ellipse. This vector contains the Euclidean distances D_i calculated using the WiFi RSS measurements at the current location, and those from each of the L historical locations under test

$$\mathbf{z}_t = [D_1, D_2 \dots D_L].$$

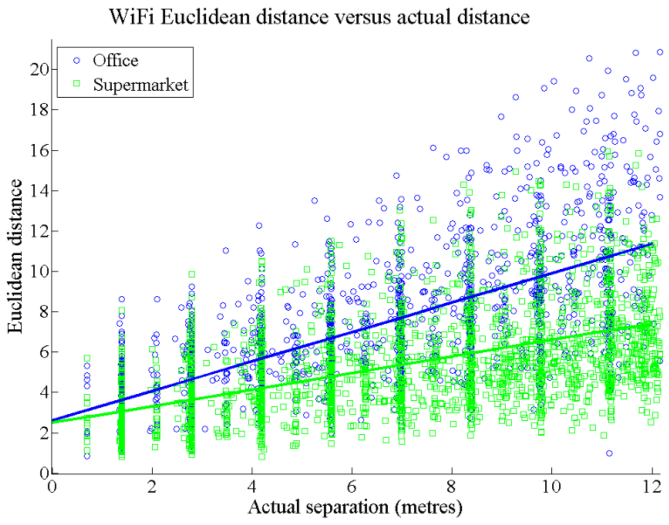

Figure 5 demonstrates that the Euclidean distance between WiFi fingerprints increases linearly with actual separation for short distances. The shallower gradient evident in the supermarket environment is to be expected as this environment is more open than an office environment. At zero physical separation the Euclidean distance is not zero because of measurement noise, with standard deviation around 3dBm (see Figure 2).

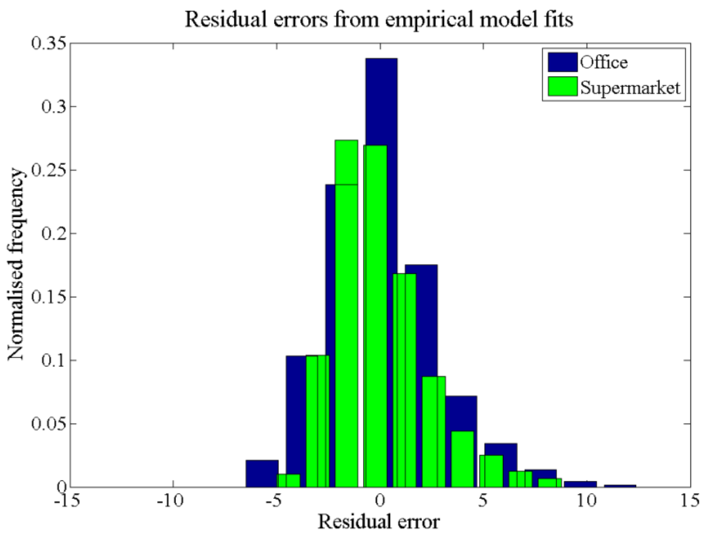

Figure 6 shows the residuals of the data compared to the lines of best fit given in the previous figure. The distribution is roughly Gaussian, suggesting that an EKF approach is well suited to fusing data from this proposed empirical measurement model. The noticeable tail of positive error is attributed to the simple empirical model used starting to break down for large separations.

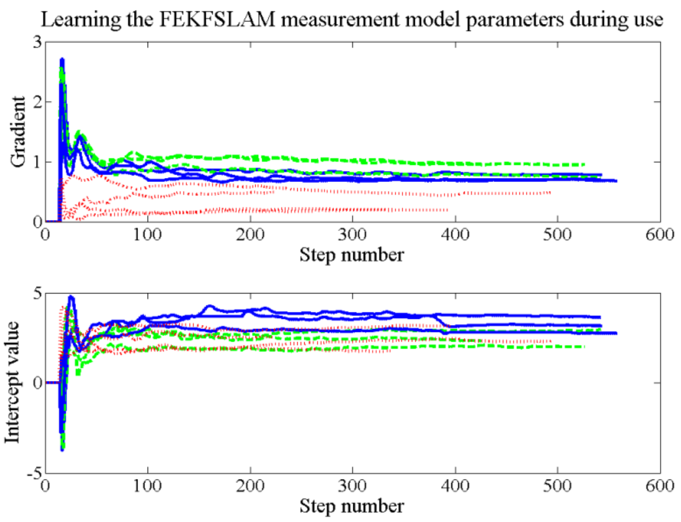

Figure 7 demonstrates the ability to learn the FEKFSLAM measurement model parameters (the best fit line intercept and gradient) during the user journey.

The corresponding predicted values Υ_i are calculated using the empirical measurement model and the Euclidean distances between the physical locations of the current user position estimate and each of the historical locations under test

$$\mathbf{h}_t = [\Upsilon_1, \Upsilon_2 \dots \Upsilon_L].$$

The Jacobian matrix is populated using the first derivatives of the measurement model with respect to the states of interest (spatial coordinates x and y), as usual for an EKF.

$$\mathbf{H}(i, \mathbf{1}) = -B \cdot \frac{x_{t_{old_i}} - x_t}{r_i}$$

$$\mathbf{H}(i, \mathbf{2}) = -B \cdot \frac{y_{t_{old_i}} - y_t}{r_i}$$

Where the i^{th} historical position under test is labelled as $\left(x_{t_{old_i}}, y_{t_{old_i}}\right)$ and B is the current estimate of the slope of the first order polynomial measurement model. Following this update to the EKF , three outcomes can occur: the current position state estimates are unchanged (the predicted and measured Euclidean distances were very similar); the position state estimates move towards the historical positions (the measured Euclidean distances were consistently lower than predicted); or they may be pushed away (the measured Euclidean distances were consistently higher than predicted). A SLAM-like update can at this point be propagated backwards through the journey to the historical positions marking the start of the loop, updating the estimated positions of the section of historical track.

A traditional EKF-SLAM scheme would allow the state vector and covariance ellipse to grow without bound as the number of landmark observations increases. This is to allow the correlations between any historical journey points to be maintained via the landmarks observations they share, thereby allowing accurate historical corrections to the path. In our FEKF framework, however, we can assume our PDR scheme is well modelled and our drift characteristics are well understood[1]. This means we can avoid maintaining a large history of correlations and allowing our state vector and covariance ellipse to grow dynamically. Instead we assume that the PDR drift that has occurred during this loop in the journey can be well estimated by a linear error growth with time in the x and y spatial dimensions. When the position update is provided to the current state vector by the Euclidean distance comparison, a weighted component of this update is driven back through the loop, providing a very simple approximation to SLAM. In this way we trade high resource consumption for lower (but acceptable) accuracy. The user will still benefit from useful corrections to their current position estimate as they journey through the environment, and the historical positions and associated fingerprints will improved too, although the quality of these updates (and the resulting final overall journey track accuracy) will not be as high as for schemes such as DPSLAM or GPLVSLAM, which have much higher computational load. We have chosen to sacrifice a more exact SLAM scheme for an approximation with the benefits of reduced computational load.

If the current location and the historical location really are displaced (e.g. two parallel corridors within a building separated by rooms), then the resulting Euclidean distances predicted by the model are expected to be consistently similar to those resulting from processing the real measurements. In this case the user track will not erroneously collapse onto the old track via a loop closure SLAM update as there will not be a consistent gradient generated in the Jacobian to drive the path back towards the historical locations.

D. DPSLAM

The most computationally-intensive algorithm in this toolbox is the Distributed Particle SLAM scheme we described in [3]. SmartSLAM applies this algorithm if a building floor plan is available for PDR parameter calibration; one of the FEKF-based schemes begins to diverge; or when the PDR gait recognition scheme is returning inconclusive information regarding the motion of the user. DPSLAM maintains multiple hypotheses can operate on highly non-linear and non-Gaussian systems through its use of a particle filter. Other schemes such as FASTSLAM, GRAPHSLAM or GPLVSLAM could be used, although DPSLAM can easily be implemented to run in real time on a smartphone

In order to minimize the computation load, the DPSLAM filter should make use of a dynamic particle number based on the size of the spatial area that the particle cloud covers and a minimum particle density [31].

VII. Bringing it all together

A decision tree based on the flow diagram of Figure 4 can be used to move between the different fusion algorithms. The initial position of the user must be provided by either GNSS fixes or prior WiFi signal fingerprint maps. If the latter is required then a traditional fingerprint look up scheme such as KNN can provide a useable initial position fix. Given the availability of confident GNSS, or rich prior fingerprint maps, the user calibration parameters can be determined as discussed in Section VI.

The system is then free to move through the PDR-only and FEKF schemes to the DPSLAM scheme if necessary. Once the particle cloud has confidently collapsed into a small region below a trigger threshold the system can switch back to the FEKF algorithm.

When the SLAM methods apply a loop closure correction and update the current position estimate, the corresponding spatial error ellipse (in FEKFSLAM) or particle cloud (in DPSLAM) will contract accordingly. It is important however to note that the historic position that is being used to drive the correction itself carries some spatial uncertainty in terms of the global position coordinate. We use the same approach proposed in the previous work on DPSLAM [3] to account for this problem by storing the level of uncertainty (particle cloud size or FEKFSLAM covariance ellipse spatial terms) along with the coordinates of each point on the historical track. When loop closure occurs the global spatial uncertainty can then be accounted for. In the DPSLAM solution this can be provided by an error ellipse based on the distribution of the particle cloud from the historical, rather than the current, user position. In the FEKFSLAM solution the spatial covariance terms from

[1] The validity of this assumption can be monitored by running a gait recognition scheme to classify each step [29].

the historical position can be incorporated into the current covariance matrix.

VIII. Resource Consumption

Here we assess the costs associated with each of the schemes described above. We do not consider the costs of inertial and WiFi sampling; floor determination; or PDR algorithms because these are common to all schemes and result in a common overhead.

Assuming the DPSLAM scheme makes use of P particles, there is an O(P) cost to updating the states. We have found that P needs to be at least few hundred for reliable results. Testing the proximity to historical positions requires some form of spatial indexing, which can cost O(1) if a simple lookup grid is used, or a logarithmic form if a more space-efficient structure is preferred. However, the space cost is dominated by the need to store the particle histories. The actual space requirement at any given moment is dependent on the environment and the route taken through it. As a guideline, however, we have found it necessary to retain multiple hypotheses for the last 100 steps or so.

The FEKF scheme requires the lookup of nearby values within a signal strength map: as with DPSLAM and history lookups, this can be achieved in O(1) time or similar. It has a constant size state vector (3D position and, if required, the step length and compass bias) and the Kalman update step involves inverting an MxS matrix where M is the number of measurements used in the fingerprint (typically around 4-6 for WiFi) and S is the number of states in the state vector to be updated (two spatial dimensions, or two spatial dimensions and two PDR parameters if these need to be calibrated/tracked). Assuming that M>S in general, the associated matrix inversion is O(M^3) or better (depending on the method used). This is significantly faster than an O(PM) update for DPSLAM. Note that during epochs with no measurements available the FEKF update complexity is O(1) whereas the DPSLAM complexity is still O(P) if all particles are propagated with some additional random noise in their spatial states. In addition there is no need to store historical state, making the space requirements much smaller. When a loop closure occurs and the particle weights change, driving this update through the historical path of N steps is O(PN).

The FEKFSLAM scheme is an approximation to the SLAM problem with the key benefit that we do not track all historical landmarks/positions within the state vector and covariance matrix. In traditional EKF SLAM schemes these structures grow without bound, resulting in the need to invert sparse matrices with hundreds of rows and columns. Here however the SLAM scheme maintains a state vector of fixed size and a covariance matrix of size equal to that in the FEKF approach, regardless of the size of the SLAM loop. A SLAM loop closure and position update results in a simple linear correction being applied to each historical position in the corresponding section of the journey. The resulting historical path is not expected to be corrected as well as is possible for the DPSLAM scheme, or any of the offline SLAM schemes, but the FEKFSLAM approximation is much more efficient and represents an excellent compromise between accuracy and resource consumption. Given an historical path consisting of N positions, the FEKFSLAM correction step following a loop closure detection is O(N) and is not iterative, compared to the iterative optimisation schemes of GRAPHSLAM (O(N^2) per iteration) or GPLVSLAM (O(N^3) per iteration) [17].

IX. Experimental results

The SmartSLAM scheme was tested using Galaxy Nexus and LG Nexus 4 smartphones. The indoor navigation smartphone application processed the sensor data from the accelerometers, gyroscopes, magnetometers and barometer within a step detection scheme. When steps were detected the gyro-smoothed compass heading was recorded for that epoch, along with the magnetometer readings and WiFi scan results. All experiments could therefore be rerun offline for further testing and analysis.

In order to compare the errors associated with each algorithm, a ground truth reference is required. The Cambridge Computer Laboratory is still home to the Active Bat indoor positioning system [32], although it is only functional in a very small section of the building. Active Bat provides 3D positioning with an accuracy of 3cm. The output of the DPSLAM algorithm when constrained by the vector floor plan and making use of a large particle number (1000) was validated using the Active Bat system. The accuracy was determined to be around 1.6m with 66% confidence, and 2.7m with 95% confidence. An example validation journey is shown in Figure 8. A DPSLAM solution constrained by the floor plan was therefore used as the ground truth for longer journey experiments exploring the whole building where Active Bat was unavailable.

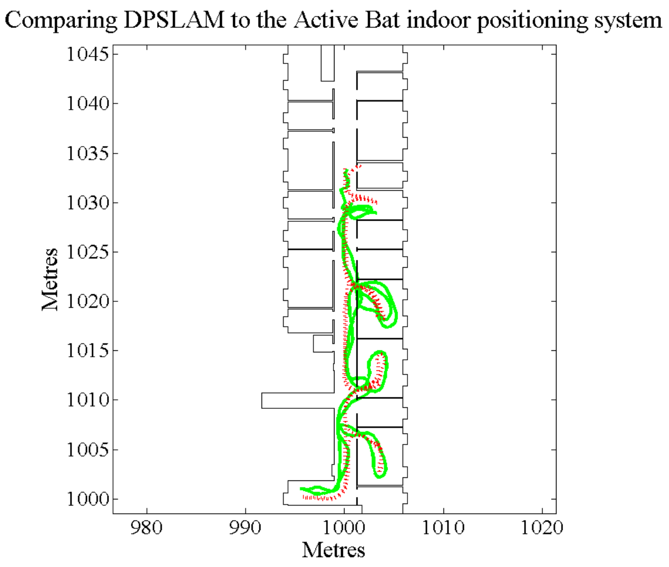

Figure 8 shows a comparison of the accuracy of DPSLAM (green solid line) constrained by the floor plan and the Active Bat indoor positioning system (red dotted line).

Figure 9 shows an example of a figure-of-eight journey through a large office building. The top row demonstrates the performance of two extremes – the PDR solution (red) and a DPSLAM particle filter solution with both prior signal strength maps and the vector floor plan available (green). The PDR solution exhibits noticeable drift over this five minute walk (the final PDR position estimate carries an error of 15 metres). The second row demonstrates the performance of the FEKF scheme (blue) and DPSLAM schemes (green) for the same journey when the prior WiFi coverage maps (but *no* vector floor plan) are available. The third row demonstrates the performance of the FEKFSLAM (blue) and DPSLAM schemes (green) with no prior information whatsoever available. The differences between each of these tracks and the baseline track are given in Figure 10. Note that for the SLAM schemes the corresponding error plot is calculated using the full track at the end of the journey, which has been corrected at multiple stages as the journey progressed. The particle filters used for Figure 9 used a fixed particle number of 500 for each algorithm under test.

Figure 11 shows the online errors during a series of SLAM tests (with no prior information whatsoever) in the Computer Laboratory, each test consists of a journey around the laboratory consisting of at least one loop closure after around 200-300 steps, and a further loop closure at the end of the journey. This time the error traces are representative of the instantaneous position estimate at each step, i.e. the SLAM back-corrections are not applied to earlier position estimates in this figure. The plots therefore demonstrate the range in current-position errors that a user could observe in real time during an indoor journey. Each plot shows the mean of the traces from the whole set of journeys, with the standard deviation providing error bars. The distribution of errors are not expected to follow Gaussian statistics, the error bars and choice of mean have merely been used to combine the data to permit a qualitative analysis of the range in errors for each method, and the effect of varying the DPSLAM particle size. The PDR errors are shown in red. The FEKFSLAM scheme (blue plots) exhibits excellent performance compared to DPSLAM (green plots). This is due to a number of factors.

Firstly, the user motion was valid under the FEKFSLAM assumptions, i.e. the user mostly walked at a steady pace along their heading. At times the gait changed, such as when interacting with security doors, and false steps are likely to have been recorded by the system, but these errors can be corrected to some degree by the SLAM corrections. Secondly, the linear Euclidean measurement model proposed in this paper has proven to be reliable in cluttered indoor environments. Thirdly, there is an issue with DPSLAM which does not affect FEKFSLAM. When a user approaches part of their historical track to begin a loop closure, only the extreme particles at the edge of the particle cloud can be tested against their histories at first. As this small subset of particles begins to overlap with their past positions weightings begin to be applied to the cloud. This can bias the DPSLAM loops to start to close slightly early, and while this can be rectified later as further measurements and comparisons occur, it can result in a reduction in the real time positioning accuracy as each loop closure occurs. FEKFSLAM on the other hand does not suffer such a problem as it is based on a measurement model with continuous variation with range from the historical track positions. Under the current formulation, DPSLAM particle updates can only provide a probability that a particle is back in a historical location, not that it is a certain distance from a historical location. However, given the availability of a floor plan, DPSLAM has a strong advantage over FEKFSLAM, as it can easily deal with the highly discontinuous nature of the geometrical constraints of walls and passageways. DPSLAM can also make use of magnetic anomaly measurements, which have not been included yet in the FEKFSLAM or FEKF frameworks because they vary on a very rapid spatial scale.

X. CONCLUSIONS AND FURTHER WORK

We have presented a new smartphone indoor positioning scheme that uses different sensor fusion algorithms depending in the level of prior knowledge available, in order to improve battery life while maintaining good performance. We have also introduced two new sensor fusion concepts: the Jacobian Map, which allows WiFi signal strength maps and fingerprints to be incorporated directly into an Extended Kalman Filter; and a fast self-correcting extension to this Kalman-filter based scheme which provides an approximation to a SLAM solution without any of the computational scalability issues of traditional SLAM schemes. This FEKFSLAM scheme is enabled by the use of a novel measurement model based on the linear increase with spatial separation of Euclidean distances for WiFi fingerprints in indoor environments. The computational benefit of this new scheme is significant, with FEKFSLAM loop closure scaling as O(N), where N is the number of steps taken on the journey. DPSLAM scales as O(PN) during a loop closure, where P is the particle number (typically a few hundred or thousand). GraphSLAM and GPLVSLAM scale as $O(N^2)$ and $O(N^3)$ per iteration of their optimisation routines.

While it is well known that the magnetic signal strength patterns recorded indoors provide rich and fine scale fingerprints, they have not yet been incorporated into the FEKF, as they vary on too fine a spatial scale for the linearization approximation of the Jacobian Map to be valid. Future work will involve incorporating magnetic measurements into FEKF and using them to assist in FEKFSLAM loop closure. More advanced loop closure schemes for the FEKFSLAM scheme could also provide improved performance, especially in situations where the user's motion is not ideal, and where the heading error is dominated by gyro drift rather than magnetic compass noise and spurious readings. Finally, a method to fuse vector floor plan data directly within the FEKF and FEKFSLAM schemes would make a further important contribution.

Acknowledgements

The authors would like to thank Ian Sheret for his constructive comments.

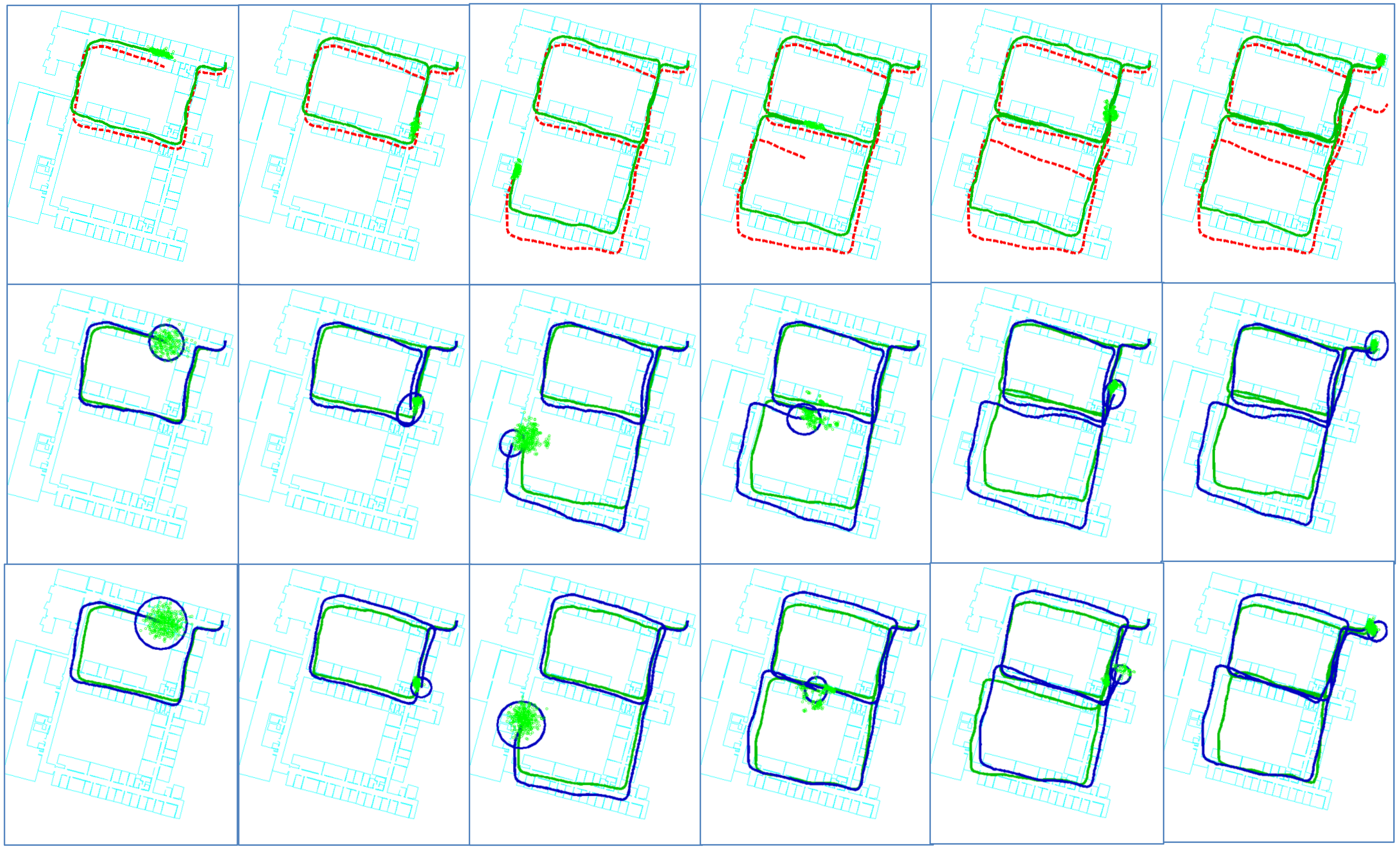

Figure 9. A comparison of the different tracking algorithms available in the SmartSLAM system. The top row of images shows six stages through a figure-of-eight journey in the Cambridge Computer Laboratory. The red dashed line is the PDR solution. The solid green line is the DPSLAM solution with prior WiFi maps and a floor plan available. The second row shows the FEKF (blue line and error ellipse) and DPSLAM (green line and particle cloud) solutions when prior WiFi maps are available to the user. The bottom row of images shows the performance of the FEKFSLAM (blue) and DPSLAM (green) algorithms when no prior knowledge at all is available and SLAM alone constrains the user error growth over time.

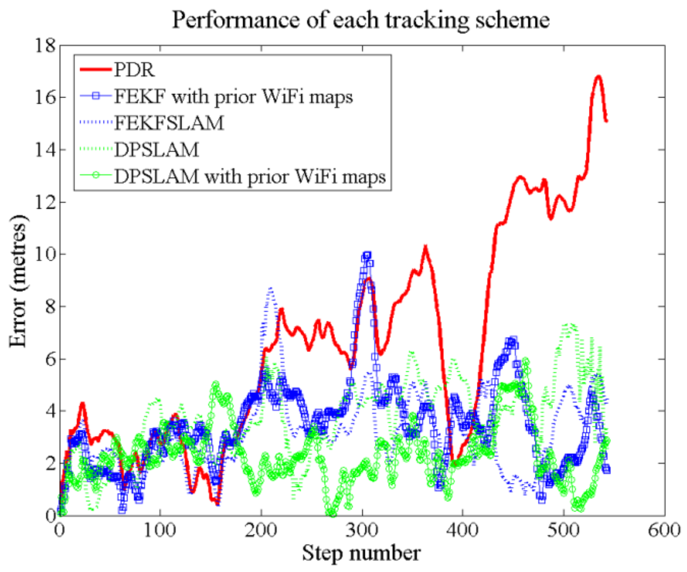

Figure 10. The positioning performance of each tracking scheme relative to the baseline scheme for the journey shown in Figure 9. The PDR error increases approximately linearly with distance travelled and represents a 2% error with distance travelled.

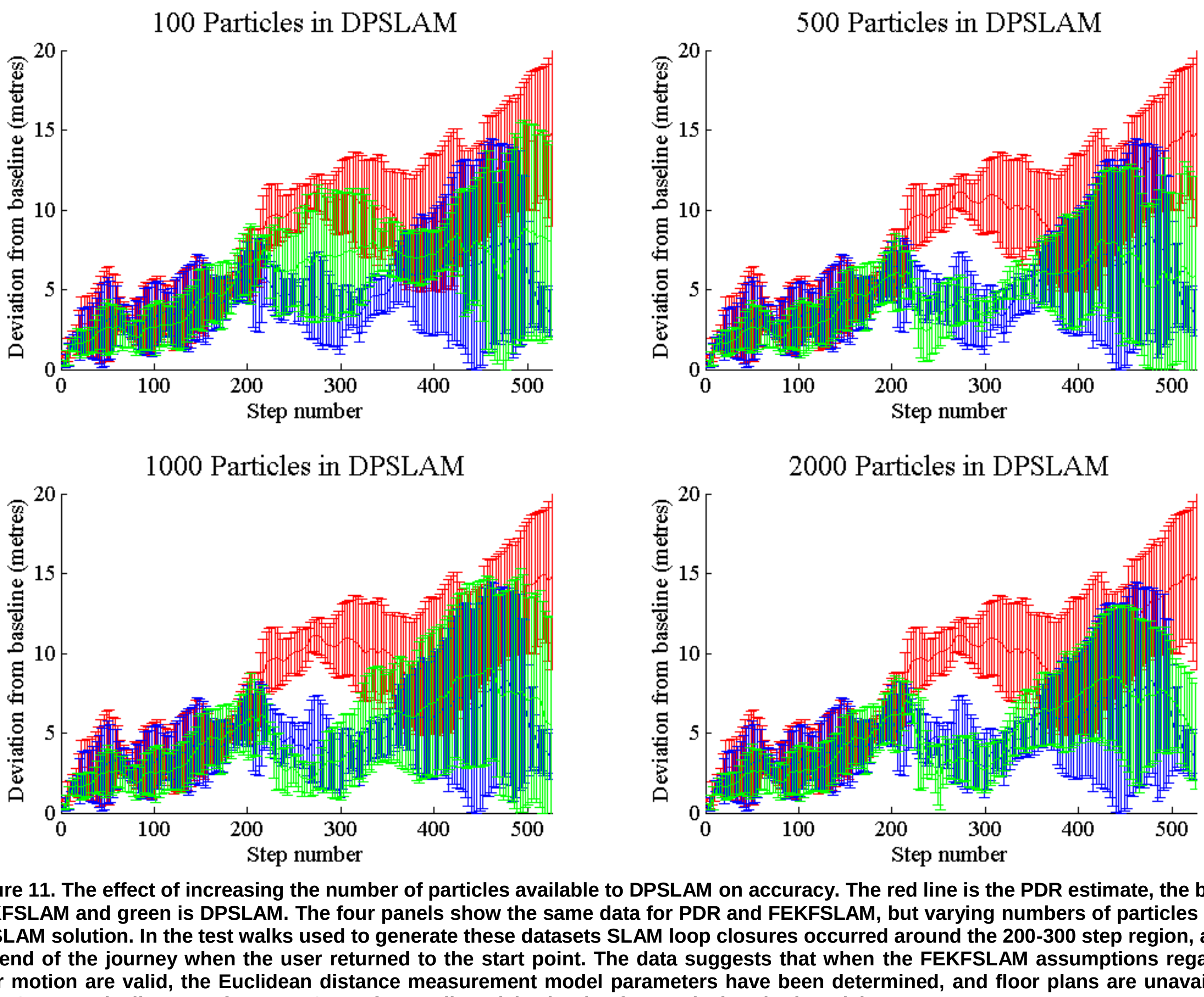

Figure 11. The effect of increasing the number of particles available to DPSLAM on accuracy. The red line is the PDR estimate, the blue is FEKFSLAM and green is DPSLAM. The four panels show the same data for PDR and FEKFSLAM, but varying numbers of particles in the DPSLAM solution. In the test walks used to generate these datasets SLAM loop closures occurred around the 200-300 step region, and at the end of the journey when the user returned to the start point. The data suggests that when the FEKFSLAM assumptions regarding user motion are valid, the Euclidean distance measurement model parameters have been determined, and floor plans are unavailable, FEKFSLAM typically outperforms DPSLAM for small particle clouds of around a hundred particles.

REFERENCES

[1] Niedermeier, H.; Eissfeller, B.; Winkel, J.; Pany, T.; Riedl, B.; Wörz, T.; Schweikert, R.; Lagrasta, S.; Lopez-Risueno, G.; Jiminez-Banos, D., "DINGPOS: High sensitivity GNSS platform for deep indoor scenarios," Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on , vol., no., pp.1,10, 15-17 Sept. 2010

[2] http://www.skyhookwireless.com/

[3] Opportunistic radio SLAM for Indoor Navigation using Smartphone Sensors R. Faragher, C. Sarno, M. Newman. IEEE/ION PLANS 2012 – April 24-26, Myrtle Beach, SC, April 24, 2012

[4] Trevisani, E.; Vitaletti, A., "Cell-ID location technique, limits and benefits: an experimental study," *Mobile Computing Systems and Applications, 2004. WMCSA 2004. Sixth IEEE Workshop on* , vol., no., pp.51,60, 2-3 Dec. 2004

[5] Bahl, P., Padmanabhan, V., Balachandran, A., 2000. Enhancements to the RADAR user location and tracking system, Technical Report MSR-TR-00-12, Microsoft Research, Feb. 2000.

[6] Youssef, M., 2004. HORUS: A WLAN-Based indoor location determination system, Ph.D. Dissertation, University of Maryland, 2004.

[7] King, T., Kopf, S., Haenselmann, T., Lubberger, C., Effelsberg, W., 2006. COMPASS: A Probabilistic Indoor Positioning System Based on 802.11 and Digital Compasses, 1st WiNTECH, Sept 2006, 34-40.

[8] http://www.ubisense.net/en/

[9] http://www.q-track.com/

[10] http://www.omnisense.co.uk/

[11] D. H. Titterton and J. L. Weston, Strapdown Inertial Navigation Technology. Stevenage, U.K.: Peregrinus, 1997.

[12] Foxlin, E., "Pedestrian tracking with shoe-mounted inertial sensors," *Computer Graphics and Applications, IEEE* , vol.25, no.6, pp.38,46, Nov.-Dec. 2005

[13] Durrant-Whyte, H.; Bailey, Tim, "Simultaneous localization and mapping: part I," *Robotics & Automation Magazine, IEEE* , vol.13, no.2, pp.99,110, June 2006

[14] Grisetti, G.; Kümmerle, R.; Stachniss, C.; Burgard, W., "A Tutorial on Graph-Based SLAM," Intelligent Transportation Systems Magazine, IEEE , vol.2, no.4, pp.31,43, winter 2010

[15] http://openslam.org/

[16] B. Ferris, D. Fox, and N. Lawrence, "WiFi-SLAM using Gaussian process latent variable models," Proceedings of IJCAI 2007, pp. 2480–2485, 2007.

[17] Huang, J.; Millman, D.; Quigley, M.; Stavens, D.; Thrun, S.; Aggarwal, A., "Efficient, generalized indoor WiFi GraphSLAM," *Robotics and Automation (ICRA), 2011 IEEE International Conference on* , vol., no., pp.1038,1043, 9-13 May 2011

[18] B. Ferris, D. H¨ahnel, and D. Fox. Gaussian processes for signal strength-based location estimation. In Proc. Of Robotics Science and Systems, 2006.

[19] Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian processes for machine learning. The MIT Press, 2006.

[20] http://www.invensense.com/mems/gyro/documents/PS-MPU-6000A.pdf

[21] Bullock, J. Blake, Chowdhary, Mahesh, Rubin, Dimitri, Leimer, Donald, Turetzky, Greg, Jarvis, Murray, "Continuous Indoor Positioning Using GNSS, Wi-Fi, and MEMS Dead Reckoning," Proceedings of the 25th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2012), Nashville, TN, September 2012, pp. 2408-2416.

[22] Vanini, S.; Giordano, S., "Adaptive context-agnostic floor transition detection on smart mobile devices," *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on* , vol., no., pp.2,7, 18-22 March 2013

[23] Wonsang. S.; Lee, J.; Schulzrinne, Henning G.; Lee, B. "Finding 9-1-1 Callers in Tall Buildings" Columbia University Computer Science Technical Reports http://hdl.handle.net/10022/AC:P:18852

[24] Pratama, A.R.; Widyawan; Hidayat, R., "Smartphone-based Pedestrian Dead Reckoning as an indoor positioning system," *System Engineering and Technology (ICSET), 2012 International Conference on* , vol., no., pp.1,6, 11-12 Sept. 2012

[25] S. H. Shin, C. Park, J. W. Kim, H. Hong, and J. M. Lee. Adaptive step length estimation algorithm using low-cost mems inertial sensors. In Sensors Applications Symposium, 2007. SAS '07. IEEE, pages 1–5, Feb.

[26] O. Woodman and R. Harle. Pedestrian localisation for indoor environments. In Proceedings of the 10th international conference on Ubiquitous computing, pages 114–123. ACM, 2008.

[27] Pei, Ling, Chen, Ruizhi, Liu, Jingbin, Kuusniemi, Heidi, Chen, Yuwei, Tenhunen, Tomi, "Using Motion-Awareness for the 3D Indoor Personal Navigation on a Smartphone," Proceedings of the 24th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2011), Portland, OR, September 2011, pp. 2906-2913.

[28] Ville Honkavirta, Tommi Perälä, Simo Ali-Löytty, and Robert Piché. A comparative survey of WLAN location fingerprinting methods. In Proceedings of the 6th Workshop on Positioning, Navigation and Communication 2009 (WPNC'09), pages 243-251, March 2009.

[29] Avci, Akin; Bosch, Stephan; Marin-Perianu, Mihai; Marin-Perianu, Raluca; Havinga, Paul, "Activity Recognition Using Inertial Sensing for Healthcare, Wellbeing and Sports Applications: A Survey," Architecture of Computing Systems (ARCS), 2010 23rd International Conference on, vol., no., pp.1,10, 22-23 Feb. 2010

[30] Understanding the Basis of the Kalman Filter via a Simple and Intuitive Derivation [Lecture Notes] R. Faragher. Signal Processing Magazine, IEEE , vol.29, no.5, pp.128-132, Sept. 2012 doi: 10.1109/MSP.2012.2203621

[31] Fox, Dieter. "KLD-sampling: Adaptive particle filters." In Advances in neural information processing systems, pp. 713-720. 2001.

[32] Harter, Andy; Hopper, Andy; Steggles, Pete; Ward, Andy; Webster, Paul (2002), "The anatomy of a Context-Aware Application", Wireless Networks 8: 187–197, doi:10.1023/A:1013767926256