2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN), 24-27 September 2018, Nantes, France

FastGraph – Organic 3D Graph for Unsupervised Location and Mapping

Cristiano Pendão*, Adriano Moreira†
Algoritmi Research Center, University of Minho - Portugal
*cpendao@dsi.uminho.pt, †adriano.moreira@algoritmi.uminho.pt

***Abstract*—It is well-known that fingerprinting-based positioning requires an exhaustive calibration phase to create a radio map, which often requires recalibration. Model-based and geometric approaches try to mitigate this effort at the expense of a lower accuracy or high computational cost. This paper introduces FastGraph, where a 3D graph is used to rapidly model the radio propagation environment. By means of unsupervised techniques, FastGraph is able to operate shortly after its deployment without previous knowledge about the environment. The proposed solution uses a novel algorithm to automatically provide location while simultaneously updating the radio map; and learn the position of the Access Points (APs) and location-specific radio propagation parameters. FastGraph has been evaluated in two real-world environments, a factory-plant and a regular university building, with results comparable to those obtained by conventional radio map-based solutions.**

***Index Terms*—Indoor Positioning, Wi-Fi SLAM, Unsupervised, 3D-Graph, Force-Directed, AP-Position Estimation.**

I. Introduction

The ubiquitousness of WLANs has been opening the opportunity for many different positioning systems, especially in indoor environments. Generally, we can separate Wi-Fi positioning techniques in two main groups: Wi-Fi Fingerprinting Matching and Model-Based/Geometric. In order to achieve low positioning errors, several solutions propose to combine odometry and orientation information with Wi-Fi RSS (Received Signal Strength) measurements. This type of data is also explored in SLAM approaches.

In this paper we present FastGraph. Our core algorithm implements a method inspired in a Force-Directed 3D Graph to solve a set of equations in a way similar to Multidimensional Scaling (MDS), providing unsupervised positioning without relying in a manual calibration, radio maps, or previous knowledge of the APs' positions. In addition, the algorithm is not based on assumptions such as that the propagation parameters are uniform across the space or even for the same AP, or that RSS measurements are unique for a specific area of the space. The only requirement in our solution is the deployment of a limited number of Anchor nodes, that are based on the low-cost Raspberry Pi (RPi). The effort of installing them is much lower than the effort and costs of generating a dense radio map and keep it updated over a long period of time. In the absence of anchors, the system may use a few labelled fingerprints as reference points. Another key advantage of FastGraph is that the system is ready to provide location estimations after just a few minutes after installing the anchors.

II. Wi-Fi Based Positioning

Wi-Fi Positioning techniques are mostly based on Fingerprinting Matching or in Model/Geometric approaches [1].

A. Wi-Fi Fingerprinting

Radar [2] is a pioneer and well-known work in Wi-Fi Fingerprinting. Fingerprinting techniques became popular because they can provide positioning without additional hardware, knowledge about the APs' positions, or the space layout. One advantage of this type of strategy is that it can be applied to heterogeneous indoor spaces, including underground [3]. Fingerprinting solutions rely on scene analysis to map the Wi-Fi radio signatures (fingerprints) at several locations to build a fingerprint database, commonly known as Radio Map. This is commonly known as the offline/calibration phase. In the online/location phase, the position of a device is estimated by comparing the currently observed fingerprints against the radio map. In this approach, the positioning performance is related to the density and distribution of the fingerprints in the radio map, which also becomes outdated due to RSS variations [4], [5]. The radio signature at a specific position can change rapidly due to interference, obstacles being moved, or even due to open/closed doors and different density of people inside the space. For this reason, in addition to the initial calibration, keeping the radio map up to date requires frequent recalibration. More detail about Wi-Fi Fingerprinting can be found in [5], [6], [7].

The requirements of the calibration phase impose limits on the scalability of fingerprinting solutions, being one of the main challenges. Collaborative approaches have been proposed to replace the professional site survey required for calibration. In those systems the radio map is built and maintained with the fingerprints collected and explicit annotated by the users [7], [8], [9]. However, collaborative systems can be affected by the quality of the users' feedback, resulting in poor position accuracy. Also, the required user interaction is inconvenient. For these reasons, some other approaches explore inertial sensors and interfaces integrated in mobile phones to take advantage of the user motion patterns, in order to construct the radio map [8], [10], [11], [12]. However, the site survey dependence on noisy step counters, and the users' tendency to visit the same areas, affect the quality of the resulting radio map. Another aspect to be taken into account in this type of strategy is the energy efficiency, as addressed in [13].

B. Geometric or Model Approach

Model-Based techniques try to minimise the calibration effort by using, for example, radio propagation models, such as the Log-Distance Path Loss Model (LDPL) instead of fingerprinting matching [12], [14], [15], [16], [17]. The LDPL relates the RSS at a given position with the distance to the AP. The difference between the RSS measurements of the same AP in different known positions can be used to approximate the AP's position. Then the user position can be estimated based on the RSS-based weighted centroid, considering the approximated positions of the observed APs. This type of techniques is largely affected by the indoor propagation complexity, for example due to severe multi-path effects. The reduced calibration effort is typically translated in worse positioning performance. In addition, the floor plans or the APs' locations are needed for most of these systems. EZ, a system described in [14], uses the geometric spatial constrains of the Wi-Fi propagation, given by the LDPL, with opportunistic GPS fixes of some devices, and estimate the location based on the relative signal measurements, without requiring previous knowledge of the RF environment, floor plan or APs locations. A genetic algorithm and optimization technique are used to solve a system of LDPL equations. This process requires high computational effort, which can range from minutes to several hours, depending on the space and the problem size, and has to be repeated periodically. Determining the necessary conditions under which a set of LDPL equations has a unique solution is still an open problem. In addition, EZ considers that the path loss exponent η is the same for one AP in all areas of the space while, in practice, η is different from location to location. EZ relies on GPS fixes, which can be a problem in indoor spaces, and only considers a 2D space. The authors report results close to conventional fingerprinting solutions, without requiring extensive calibration.

Another approach is to use the attributes of the received signals, such as the time that the signal takes from the sender to the receiver or the angle at which the signal is received. Examples of this type of methods can be found in [10]. However, in most of these methods the position of the APs must be known a priori, which is difficult due to the uncoordinated deployment of WLANs. The propagation effects, due to obstacles, signal fluctuations, and interference also affect this type of solutions.

Despite the fact that indoor propagation modeling is one of the most complicated tasks in this field, it is highly relevant as it may be used to replace or complement the site survey techniques [1].

C. Wi-Fi SLAM

In robotics, the Simultaneous Localization and Mapping (SLAM) approach is used to construct the space map while locating the robot in the space. Some works have been done to apply this approach to Wi-Fi. WiFi-SLAM [18] uses the Gaussian Process Latent Variable Model (GP-LVM) to build Wi-Fi RSS Maps without requiring Inertial Measurements Units (IMU) data. The limitations of the WiFi-SLAM are the computational efficiency, and relying on assumptions on a signature uniqueness and human walking patterns. Wi-Fi GraphSLAM [19] improves the techniques based on GP-LVM, in terms of computational efficiency and the assumption on signature uniqueness. In robotics, the GraphSLAM technique is used for building the space map while estimating the trajectory. Wi-Fi GraphSLAM uses gyroscope and pedometer data, and similar Wi-Fi RSS observations to detect that the user has returned to a previously surveyed physical location. In WiSLAM [20] data from foot mounted IMU is combined with the Wi-Fi RSS. The RSS measurements are translated into distance using a log-distance propagation model. The application scenarios are limited by the required foot-mounted sensor and by fixing the path loss exponent to 2. However, for indoor it is not realistic to assume a fixed and homogenous path loss exponent value for all areas of the space.

III. The FastGraph solution

FastGraph core can be seen as a sophisticated trilateration problem, where RSS values are used to obtain the distances to the detected APs using the LDPL model and where a 3D graph is used to compute the estimated position. The 3D graph is continuously updated with the latest fingerprints using and algorithm inspired on the Force-Directed method to draw graphs. Fingerprints can either come from crowdsourcing (unlabelled) of from anchor nodes (georeferenced). We argue that the more gathered data through the whole scenario, the more consistent will the graph be and, therefore, the lower will be the positioning error.

A. The fundamental principles

The fundamental idea is based on creating a 3D Graph that evolves to be a representation of the physical and radio environment. In this Graph (see Fig.1), nodes represent APs, Anchors, or Samples from Moving Devices. Only the position of Anchor nodes is assumed to be known a priori. After each new sample is added, the Graph is adjusted to a minimum energy state. Each sample contains Wi-Fi RSS measurements, collected at a specific position, of the nearby APs. In addition, these samples can contain motion and orientation information. A sample s_i is described as:

$s_i = (device_{ID}, t, ((AP_1, RSS_1), (AP_2, RSS_2), ...), dis, he)$, where $device_{ID}$ is the unique identifier of the device that collected the sample, t is the timestamp when the sample was collected. The list of (AP_i, RSS_i) pairs is the RSS measurements for the visible APs, dis is the linear displacement since the previous sample and he is the current heading.

1) The Graph: The Graph has different types of nodes and different types of edges that connect the nodes (see Fig.1).

A node on the Graph can be:

- **AP node:** $node_i = (AP_i, (x_i, y_i, z_i))$, represents an Access Point$_i$ with unknown initial position (x_i, y_i, z_i).
- **Anchor node:** $node_i = (A_i, (x_i, y_i, z_i))$, represents Anchor$_i$ with fixed and known position.

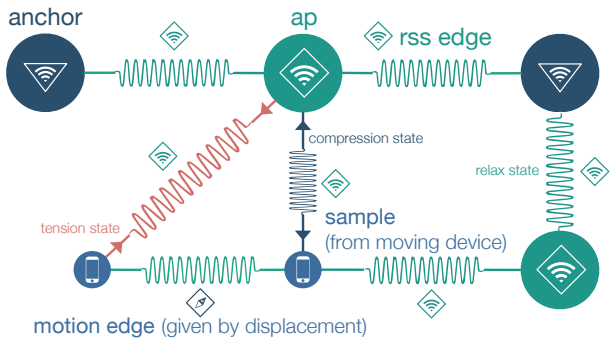

Fig. 1. Graph Concept (2D View)

- **Samples node:** $node_i = (S_i, (x_i, y_i, z_i))$, each sample collected by a moving device originates a new sample node in the Graph, with unknown initial position.

The nodes' positions (p_{node}) on the Graph are subject to constrains:

- **Anchor nodes:** Known and fixed positions.
- **AP nodes and Sample nodes:** p_{node} must be within the 3D space and not complex (e.g. p_{node} cannot be underground or APs positions cannot be above the celling, and xi, yi, zi must be real numbers).

An edge in the Graph connects two nodes and can be seen as a spring with an elastic factor. The Natural Length (NL) of an edge can be obtained based on different types of information, depending on the two nodes that the edge connects. The edges can be:

- **RSS-based edge:** A RSS-based edge can be seen as the 3D radio communication channel with an AP.
  - Anchor $\leftrightarrow$ AP: $e = ((node_i, node_j), NL, CL, ke1, \eta)$
  - Sample $\leftrightarrow$ AP: $e = ((node_i, node_j), NL, CL, ke2, \eta)$

  where NL is the Natural Length of the edge estimated based on the measured RSS using a propagation model, CL is the Current Length of the edge, $ke1$ and $ke2$ are the elastic constants, and η is the path loss factor.
- **Motion edge:** Connects two consecutive Sample nodes from the same moving device. The NL of this type of edge can be estimated through:
  - **Time-based:** $e = ((node_i, node_j), NL, CL, ke3)$, where the NL is the maximum displacement based on the time difference between the two samples.
  - **Odometry/Inertial-based:** $e = ((node_i, node_j), NL, CL, ke4)$, where NL is based on more accurate displacement measurements between consecutive samples.

The edges based on accurate motion data, such as odometry, can be used to improve the natural length estimations, as they provide significantly more accurate displacement information, which results in better spatial constrains between samples.

An edge can be in one of three different states (like a spring):

- **Relax state:** $CL = NL$, means that the two nodes are at the correct distance.
- **Compression state:** $CL < NL$, means that the two nodes should be farther from each other.
- **Tension state:** $CL > NL$, means that the two nodes should be closer.

2) Building the Graph: The Graph is built iteratively, based on new samples, one sample at a time.

If the new sample is from an anchor:

i. Create an Anchor node to represent the Anchor (if not already exists).
ii. Create a AP node for each new AP, visible in the sample.
iii. Add or update the edges from the Anchor node to the APs visible in the sample.

Otherwise if the new sample is from a moving device:

i. Create a new Sample node.
ii. Create a AP node for each new AP, visible in the sample.
iii. Create one edge from each AP visible in the sample.
iv. Create an edge to previous Sample nodes of the same device (from time or odometry information).

3) Adjust the Graph to a Minimum Energy State: Update the position of all nodes (except Anchor nodes), using a Force-Directed inspired method. This means that each node is subject to a Combined Force vector $c\vec{F}$, resulting from the individual forces applied by each edge connected to the node.

B. Building the Graph Iteratively

After placing the anchors, the system starts processing the samples collected by the anchors to initiate the 3D Graph, ignoring the samples from moving devices. The anchors' data is used to automatically estimate an initial position of the AP nodes and to adjust the graph to the real-world space configuration. When the algorithm detects that the position of the APs has stabilised, which normally takes only a few minutes, then the system begins to process also the samples of the moving devices, providing positioning. When the system is already operating, the additional samples provided by the anchor nodes and the moving devices make the Graph to automatically evolve and keep the radio map always updated for the whole environment.

Anchor nodes are created at known positions when the first sample from the anchor is processed. AP nodes are created when they are observed for the first time, and they are initially placed according to the first detected RSS measurements. Sample nodes are created for every new sample from moving devices, and are placed on an initial position given by the weighted centroid in relation to the APs detected on the sample, and that are already on the Graph.

With the exception of the Anchor nodes, the position of the nodes on the Graph will be influenced by the spatial constrains defined by the Natural Length (NL) of the edges.

Sample to AP edges are created when a new sample from a moving device is processed and the Sample node is created. These edges are RSS-based, therefore the Natural Length (NL) is set to the distance estimated from the measured RSS value. To obtain a distance from the RSS value the LDPL propagation model is used. The LDPL model is frequently used to model

Wi-Fi signals propagation [10], [12], [14], [15], [21], [22]. Using the LDPL model the RSS at a given position p at a distance d from an AP is given by:

$$RSS_p = RSS_0 - \left(10 \times \eta \times log10\left(\frac{d}{d_0}\right)\right) \quad (1)$$

RSS_p is dependent on the RSS_0 (measured at d_0, generally 1 meter), the distance to the transmitter (d) and the path loss exponent (η) that reflects the fall rate of the RSS signal from an AP at a specific position.

Therefore, the NL of an edge e can be given by:

$$e_{NL} = 10^{\left(\frac{1}{10\times\eta}\times(RSS0-RSS_p)\right)} \quad (2)$$

In contrast to many works in the field, where the path loss exponent of each AP is empirically set, we consider that this exponent not only depends on the AP, but also in the location of the sample. It is well-known that getting the exact sources of noise in radio propagation is not an easy task, as it involves many factors such as the presence of metallic objects. For that reason, we consider that the communication channel from a given position to a specific AP, that is represented by an edge in the graph, has singular propagation characteristics, and therefore a particular value of η.

In the proposed solution η has a strong influence in the estimated natural length of the edges and by consequence in the nodes' positions. For this reason, after adding a new Sample node to the Graph, an initial estimation of the value of η is done for each RSS-based edge of the Sample node. This initial estimation is obtained after running a Gradient Descent Algorithm and it is good enough in most of cases, in spite of the fact that it might not be the optimal exponent. That initial exponent is subject to improvement as more samples are processed and the Graph evolves.

Anchor to AP edges are also RSS-based, and are created when an AP is observed for the first time from an Anchor. Existing edges are updated as new samples involving the attached nodes are gathered. In contrast to the Sample nodes, the Anchor nodes represent a set of observations. Therefore, the RSS measurements of the same AP can be combined, for example by averaging a given number of previous RSS values. Using the combined RSS values leads to more robust estimation of the NL with the LDPL model.

Sample to Sample edges are Motion edges, and are created between samples of the same moving device, when possible. The NL is given by the displacement obtained from odometry or inertial data, or based on time elapsed between samples.

C. Adjusting the Graph

The 3D Graph evolves when a new sample is processed and the Graph is adjusted. The position of the APs and the position of the moving devices (Sample nodes) is estimated by adjusting the Graph in order to respect the 3D spatial constrains defined by the edges. The Anchor nodes provide spatial references, in order for the Graph to keep the correct rotation and scale. To adjust the Graph to a minimum energy state a force-directed approach is used [23].

Following the spring analogy, each edge has associated a force vector that is calculated based on the difference between the natural length (NL) and the current length (CL) of the edge, that is then multiplied by a elastic constant (ke), that gives the edge more a less elasticity. As explained before, each edge can be in relax state, in tension or in compression, depending on the current values of NL and CL. CL is the actual 3D distance between the two nodes on the Graph. The elastic constant (ke) is different for each type of edge, allowing each type of edge to have different influence in the nodes' positions. For example, edges connecting Anchors and APs are less affected by signal fluctuations in the RSS measurements and, for this reason, have higher value of ke.

The tension magnitude (te) applied by an edge e on the connected nodes is given by:

$$te = ke \times (CL - NL) \quad (3)$$

The force vector ($\vec{eF}$) is then given by:

$$\vec{eF} = (\vec{distance} \times \hat{u}) \times signum(tei) \times log\,(1 + ||te||) \quad (4)$$

where, $\vec{distance} \times \hat{u}$ is the distance vector between the two nodes times the unit vector, $signum(te)$ is the signal of the tension magnitude, and $log\,(1 + ||te||)$ is the logarithmic of the tension module.

The total force applied to a $node_j$ ($\vec{cF}_j$) is the vectorial sum of all the forces associated to each edge ($\vec{eF}_i$) connected to it:

$$\vec{cF}_j = \sum_{i=1}^{n} \vec{eF}_i, \text{for all node edges} \quad (5)$$

The Graph adjustment can affect all nodes or a specific sub-set of nodes. The Graph adjustment is an iterative process that ends when all the nodes on the Graph move less than a defined threshold, meaning that the Graph reached the minimum energy state. In each iteration the nodes are moved based on a velocity vector ($\vec{vel}_j$) that is the linear combination between the current $\vec{cF}_j$ and $\vec{cF}_{j-1}$, in order to preserve part of the momentum from the previous iteration. Hence the velocity vector of a node in an iteration j is given by:

$$\vec{vel}_j = \beta \times \vec{cF}_j + (1 - \beta) \times \vec{cF}_{j-1} \quad (6)$$

Each node is moved in order to minimise the forces applied to it. The new position is calculated by adding to the current position the velocity vector multiplied by a factor T:

$$\vec{pos} = \vec{pos} + \vec{vel} \times T \quad (7)$$

The T factor defines how much the node moves in a single iteration in the direction defined by the velocity vector. This controls how fast the node will converge to the position that minimises all forces. In order to avoid oscillatory motion on some nodes, the value of T is adjusted at each iteration. First

the position variation for all nodes is calculated:

$$\Delta pos_{nodes} = \frac{1}{n} \times \sum_{i=1}^{n} \Delta pos_{\vec{node_i}} \tag{8}$$

The new value of T to be used in the next adjustment iteration is then given by:

$$T = \frac{2}{\gamma + \Delta pos_{nodes} \times \Delta_t} \tag{9}$$

Where γ is the amortisation factor, and Δ_t is the time taken by the previous iteration. With this solution the graph converges to the minimum energy state faster, which improves the performance of the algorithm in real time applications.

IV. Real World Deployment

A. Deployment Spaces

To test our solution we conducted experiments in two very distinct spaces.

1) PIEP: The first deployment space is a polymers engineering building PIEP at University of Minho. The PIEP is very similar to a factory-plant, with an area of ≈1000 m^2. This space combines large open areas and smaller areas obstructed by walls. There are several metallic elements, industrial machinery and exposed beams, that result in several reflections and multi-path effects. This is a challenging space for signal propagation, and is also very dynamic, because machines and cargo are moved from place to place, and people are frequently moving around. The space was mapped and a XYZ referential defined. A ground truth grid, based on floor tags spaced 1 meter from each other, was added to the space, to define paths with known positions. Six anchors were installed to cover the area.

2) DSI-DEP: The second deployment space was a building at the University of Minho. A floor of this building has a combined square footage of around 4638m^2. The DSI-DEP is a space with characteristics very distinct from PIEP. The layout of the DSI-DEP is complex with several offices, rooms and corridors, which result in different propagation characteristics in relation to PIEP, where the main obstacles are large industrial metallic machines. At the DSI-DEP the strategy for the solution deployment was different from that used at PIEP. Instead of having several anchors installed at fixed positions, a unit that can be used as moveable anchor (Tripod Testing Unit - TTU) was designed and developed. In the DSI-DEP we collected Anchor data at 13 distinct locations.

B. Experiment Devices

We developed different devices to deploy and evaluate our solution in real world environments. The Raspberry Pi 3 (RPi) was chosen as base for the solution implementation. The Wi-Fi interface allows to collect Wi-Fi data, and the IO interfaces (USB, GPIO) allow to use external sensors and devices when necessary. Software modules were developed to collect different types of data such as Wi-Fi, orientation and distance, as well as to control and monitoring the devices.

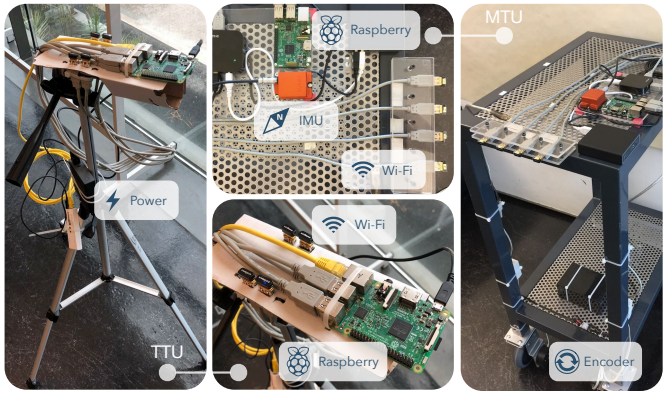

Fig. 2. Moving Testing Unit (MTU) and Tripod Testing Unit (TTU)

1) Anchors: Each anchor is based on a single RPi3 and is able to send data to a central server and be controlled remotely.

2) Moving Testing Unit (MTU): The Moving Testing Unit (MTU) (Fig. 2) was developed in order to simulate a moving device, such as an autonomous robot or machine moving inside a factory. The MTU core is also a RPi3, with the USB interfaces allowing to integrate external sensors. To measure travelled distance between samples a Magnetic Encoder attached to one wheel is used, and an Internal Measurement Unit (IMU) is used to track the direction of moving (heading). A smartphone keeps track of the ground truth tags, using the camera, and also collects data. Four additional 2.4 GHz Wi-Fi interfaces were connected to the MTU to collect additional data, that can be used to improve the position estimation, as described in [24]. In the experiments reported in this paper we only use Wi-Fi data, and from a single interface, this way a simple smartphone can be used. All of the additional data, such as the motion/orientation and cellular data is still being used to test the other possible applications of the proposed solution.

3) Tripod Testing Unit (TTU): This unit has also a RPi as base, with four additional external Wi-Fi interfaces (Fig. 2). The TTU is light and can be easily carried and moved from place to place, therefore can be used not only to work as an anchor buts also to work as a moving device, in applications where the magnetic encoder and the IMU is not used. This simple approach, can be easily placed at any location, and the height and direction can be precisely set.

V. Experimental Results

A. Experiments at PIEP

In a first experiment we tested the performance in estimating the position of the APs using only the data collected by the anchors (Fig.3). The plot on top, shows the sum of the position change, after each iteration, as we add more data (for all eleven APs), and the plot on the bottom shows the average positioning error in the same conditions. These results show that, as expected, with more samples from the anchors, the average error of the APs positions decreases.

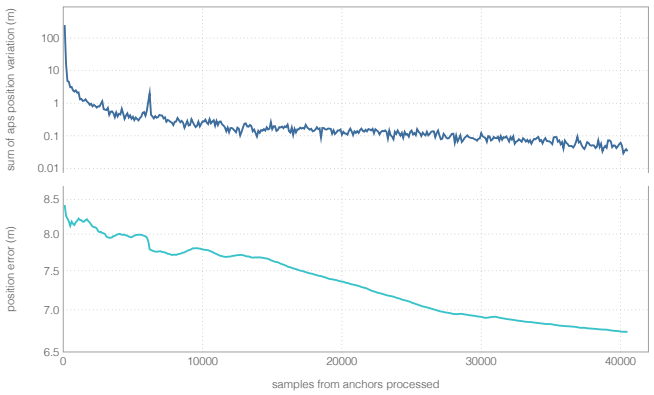

Fig. 3. APs positions variations over samples processed (40500 samples)

TABLE I
ANCHORS ONLY: APS POSITION ERROR IN METERS

AP	XY Error	Z Error	XYZ Error
AP1	4.79	1.54	5.04
AP2	7.20	1.54	7.36
AP3	3.73	1.24	3.93
AP4	4.33	0.44	4.36
AP5	4.17	0.0	4.17
AP6	5.0	2.87	5.78
AP7	8.82	0.0	8.82
AP8	3.67	2.96	4.71
AP9	5.26	1.33	5.43
AP10	6.90	0.5	6.92
AP11	17.47	1.33	17.52
Average	**6.49**	**1.26**	**6.73**

Despite the variation in the APs position becomes lower, the position estimation error keeps dropping, even though at a slower rate, as the APs position variation decrease with more samples processed. Table I shows each AP position estimation error after the 40500 samples. The average error in AP position estimation was 6.73 meters in XYZ, using only the data provided by the anchors.

After testing the APs position estimation we tested the device positioning estimation using only Wi-Fi data. Our algorithm begins by processing data from the anchors to create the 3D Graph, and uses that anchors data to estimate initial positions for the APs. After only 500 samples processed from each anchor (around 8 min of sampling), the system starts locating the MTU by processing the MTU Wi-Fi samples. In Fig.4 we can see the real time positioning error for each sample processed from the MTU, along four different paths.

The plot in Fig.5 shows the samples at the position where they have been collected, where the colour is used to represent the error in the estimated position for that sample. We can see two peaks, almost identical, with a maximum error of around 24m. These two peaks are identified as low coverage and low line of sight (LOS) area. This samples were collected in an isolated area, below a metal bridge. The plot in Fig.5, shows the samples at the position where they have been collected.

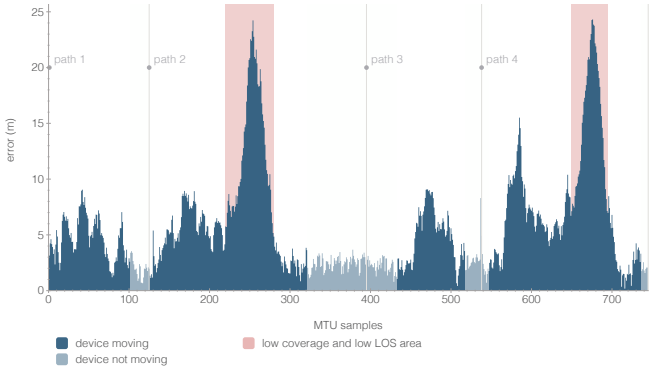

Fig. 4. Real Time Positioning Error: Wi-Fi Only PIEP

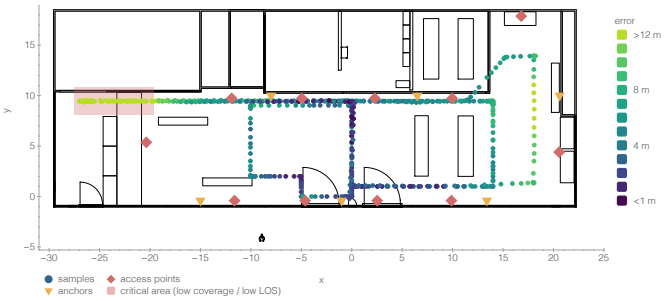

Fig. 5. PIEP Critical Area

The critical area that generated the two peaks is identified by the red rectangle and by the light green colour of the samples. As the map shows, we do not have any AP at the left of that area, and we have the bridge, at lower level than the APs, obstructing the signals. In Fig.4, we can observe that the error increases and decreases progressively before and after the peaks. In the map, we can see the same pattern in the colour of samples, becoming lighter (higher error) when going further left and then becoming darker when coming back. Each one of the peaks represents going to that area and coming back, since in two paths (2 and 4) the MTU travelled to that area.

Even with the high errors resulting from the critical area, our solution obtained an average error of 5.65m. PIEP is a very difficult environment due to metallic elements all over the space, with the critical area below a metallic bridge. However, the problem in that area can be eliminated by installing new access points. Without consider the samples collected at the critical area, the average error is 4.26m, 1.39m less, and the max error drops to 15.51m. Fig.8 shows the CDF of the position error for all paths, with and without considering the samples collected at the critical area.

We also tested our solution in the same conditions but giving the APs truth positions to the algorithm. As expected the mean error dropped significantly (from 5.65 to 3.98 meters). These results suggest that the positioning performance can be improved in applications where the APs positions are known.

B. Experiments at DSI-DEP

The second set of experiments were performed at DSI-DEP, where there are a large number of visible APs, some of them inside offices or in nearby buildings. In order to evaluate the algorithm ability to find the position of the APs, we mapped the real position of the APs located in the corridors and some inside offices in the first and second floor. With around only 6 minutes of samples, at the first floor, where data samples were collected, the algorithm estimated the APs positions with an average XYZ error of 8.21 meters and a max error of 15.88 meters. At the second floor the average error in the APs positions estimation was 15.58 meters, almost the double of the error in first floor, and the max error was 23.26 meters. The larger error is expected since we only installed anchors in the first floor, therefore the position estimations of the APs at the second floor are based on the data provided by the anchors in the first floor, which is not ideal. We compare these results with Serendipity [17], which finds the APs position in an unsupervised manner, using the dissimilarities between pairs of APs and reference scans at known positions. The authors assume homogeneous characteristics for the APs and indoor space. The algorithm cannot locate APs individually, requiring a dense AP coverage with at least two APs coverage overlapping, and scans in all floors. The number of floors is required to cluster the APs into different floors, then a 2D position is estimated. The authors report an error ranging from 3.5m to 6.7m, in two buildings with floor area of 1000m^2 and 1750m^2. We compare this with our 3D error of 6.73m in PIEP (in an area of 1000m^2) and 8.2m in the first floor of the DSI-DEP (with area of 4638m^2). We do not compare with the results at the second floor of the DSI-DEP, since the positions of those APs were estimated without any scan in that floor.

Since our AP position estimation is in 3D, Fig.6 shows the XZ plan, with the estimated position of the APs in the first and second floor. As we can see, with the exception of two APs of the second floor, all other APs are placed in the correct floor, even though no anchors were installed in the second floor. With anchors in the second floor we can expect results similar to the AP's in the first floor, where the Z position error is very small. It is out of the scope of this paper to further discuss multi-floor support, but these results strongly suggest that the proposed algorithm can automatically support multiple floors even without anchors in all floors.

The positioning results at DSI-DEP are shown in Fig.7 and Fig.8. The errors were measured at ten different known positions. The average error was 5.08m and the max error was 14.86m. The APs at the second floor, with higher error, were also used in the position estimation process. Considering this, and as already explained, we expect that by adding anchors to the second floor the APs position estimation for that floor will improve, improving also the positioning performance.

C. Comparing with other methods

We compare our solution with the results of two Tracks of the 2017 IPIN competition [25]: Track 1 Smartphone-based (on- site) and Track 3 Smartphone-based (off-site).

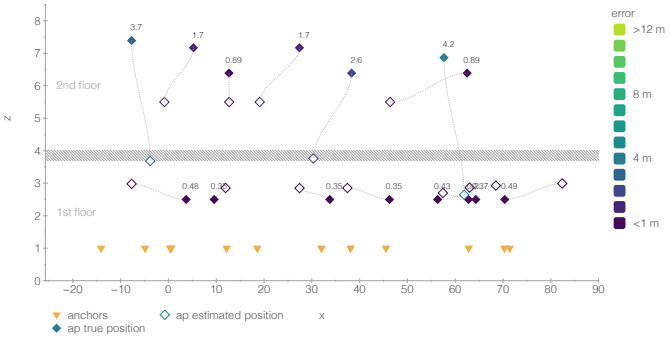

Fig. 6. APs Z Error DSI 1st and 2nd Floors

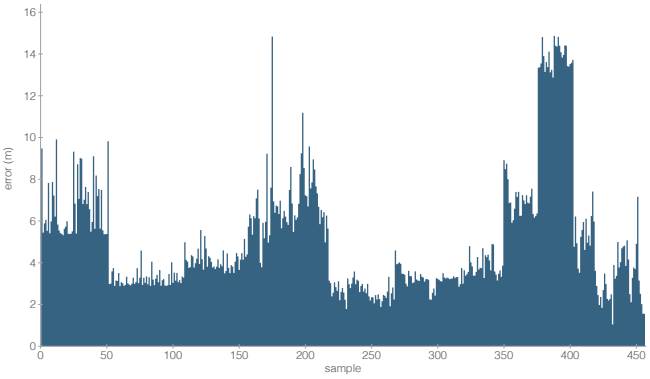

Fig. 7. RT Position Error: WiFi Only at DSI-DEP

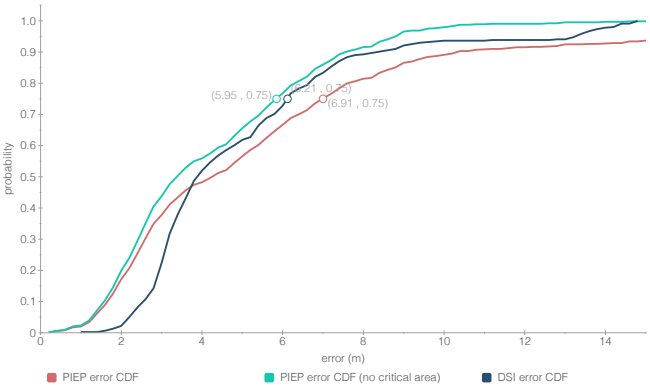

Fig. 8. Wi-Fi Only Position Error CDF

The objective of Track 1 is to evaluate the competing system on-site and in real-time. The competing teams are not allowed to install any instrumentation in the competition area. Our solution require data from the anchors, however this would not be a problem since the competing teams have at least a full day before the competition to survey the area. We could simply use our TPU device, to collect the anchors data, since we do not need these data to be available in real-time. In this track, competing teams are given the coordinates of a starting point, from where they must start walking over a reference path. Any sensor available can be used to track the user trajectory. In our results we do not use any information besides Wi-Fi, neither

TABLE II
COMPARING WITH IPIN'17 TRACK 1 AND TRACK3

Track 1 (on-site)	**Mean**	**3/4 error**
SNU-NESL PDR	6.2m	8.8m
MCL	12.6m	16.8m
XMC	23.0m	30.8m
Track 3 (off-site)	**Mean**	**3/4 error**
UMinho Team	3.00	3.48m
AraraDS Team	3.74	3.53m
Yai Team	3.51	4.41m
HFTS Team	3.52	4.45m
Our Solution (PIEP)	**5.65m**	**6.91m**
Our Solution (DSI-DEP)	**5.08m**	**6.21m**

we know the starting point.

Track 3 is an off-site competition, where all the data for calibration is provided, including Wi-Fi, data from all kind of sensors in a mobile device and some ground-truth positions. In this track the competing teams can calibrate their algorithms with several ground-truth databases provided.

As we can see in Table II, our solution performance is better than the results of the Track 1 (on-site), and somewhat worse than the Track 3 (off-site). These are encouraging results, since we are comparing with state of the art solutions. In addition, our results are based in Wi-Fi data only, and our system is unsupervised without requiring extensive calibration. In addition, our solution performance is very similar in two environments that are completely different, in propagation characteristics, layout, build materials, AP coverage, and dimensions.

VI. CONCLUSION

In this paper we have presented FastGraph, a solution that can be used in several applications. It allows to provide real-time unsupervised positioning, just a few minutes after its deployment. It is able to automatically generate a radio map of the space, keep it updated over the time, and estimate the position of the APs. The proposed solution has been validated in two different environments, showing promising results in Wi-Fi based positioning and AP-location estimation. We argue that the full potential of this solution is yet to be explored. The radio maps and the APs positions estimations can be used in many other applications, such as Wi-Fi networks maintenance or evaluation. The presented results used Wi-Fi data only. As future work, our short-term objective is to extend FastGraph to include data from other sensors (odometry, cellular and 5G networks, etc.). Our long-term goal is targeting a seamless indoor/outdoor positioning system and a solution for autonomous robots and machines in a factory environment, achieving an average error below 1 meter.

ACKNOWLEDGMENT

This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT – Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/00319/2013 and the PhD fellowship PD/BD/105865/2014.

REFERENCES

[1] S. Zvanovec, "Wireless LAN networks design: site survey or propagation modeling?," *Radioengineering*, pp. 42–49, 2003.
[2] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," *Proc. IEEE INFOCOM 2000. Conf. Comput. Commun. Ninet. Annu. Jt. Conf. IEEE Comput. Commun. Soc. (Cat. No.00CH37064)*, vol. 2, 2000.
[3] S. Conceição, C. Pendão, A. Moreira, and M. Ricardo, "Evaluation of medium access and a positioning system in wireless underground sensor networks," in *IFIP Wirel. Days*, vol. 2016-April, 2016.
[4] S. Yiu, M. Dashti, H. Claussen, and F. Perez-Cruz, "Locating user equipments and access points using RSSI fingerprints: A Gaussian process approach," in *2016 IEEE Int. Conf. Commun. ICC 2016*, 2016.
[5] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," *Proc. 3rd Int. Conf. Mob. Syst. Appl. Serv. - MobiSys '05*, p. 205, 2005.
[6] A. K. M. M. Hossain and W.-s. S. Soh, "A Survey of Calibration-free Indoor Positioning Systems," *Comput. Commun.*, pp. 1–18, 2015.
[7] S. He and S.-H. H. G. Chan, "Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons," *IEEE Commun. Surv. Tutorials*, vol. 18, no. 1, pp. 466–490, 2016.
[8] J. Ledlie, J.-g. Park, D. Curtis, A. Cavalcante, L. Camara, A. Costa, and R. Vieira, "Mole: A scalable, user-generated WiFi positioning engine," in *Indoor Position. Indoor Navig. 2011 Int. Conf.*, pp. 1–10, sep 2011.
[9] Y. Luo, O. Hoeber, and Y. Chen, "Enhancing Wi-Fi fingerprinting for indoor positioning using human-centric collaborative feedback," *Human-centric Comput. Inf. Sci.*, vol. 3, no. 1, p. 2, 2013.
[10] C. Wu, Z. Yang, and Y. Liu, "Smartphones based Crowdsourcing for Indoor Localization," *IEEE Trans. Mob. Comput.*, vol. 14, no. 2, pp. 444–457, 2015.
[11] S. Hilsenbeck, D. Bobkov, G. Schroth, R. Huitl, and E. Steinbach, "Graph-based Data Fusion of Pedometer and WiFi Measurements for Mobile Indoor Positioning," *Proc. 2014 ACM Int. Jt. Conf. Pervasive Ubiquitous Comput. - UbiComp '14 Adjun.*, pp. 147–158, 2014.
[12] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: Zero-Effort Crowdsourcing for Indoor Localization," *Proc. 18th Annu. Int. Conf. Mob. Comput. Netw. - Mobicom '12*, p. 293, 2012.
[13] C. G. Pendao, A. C. Moreira, and H. Rodrigues, "Energy consumption in personal mobile devices sensing applications," *2014 7th IFIP Wirel. Mob. Netw. Conf.*, pp. 1–8, 2014.
[14] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," *16th Annu. Int. Conf. Mob. Comput. Netw. - (MobiCom '10)*, p. 173, 2010.
[15] H. Lim, L. C. Kung, J. C. Hou, and H. Luo, "Zero-configuration indoor localization over IEEE 802.11 wireless infrastructure," *Wirel. Networks*, vol. 16, no. 2, pp. 405–420, 2010.
[16] A. Goswami, L. E. Ortiz, and S. R. Das, "WiGEM : A Learning-Based Approach for Indoor Localization," *CoNEXT*, 2011.
[17] J. Koo and H. Cha, "Unsupervised Locating of WiFi Access Points Using Smartphones," vol. 42, no. 6, pp. 1341–1353, 2012.
[18] B. Ferris, D. Fox, and N. Lawrence, "WiFi-SLAM using Gaussian process latent variable models," *Proc. 20th Int. Jt. Conf. Artifical Intell.*, pp. 2480–2485, 2007.
[19] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor WiFi GraphSLAM," *Proc. - IEEE Int. Conf. Robot. Autom.*, pp. 1038–1043, 2011.
[20] L. Bruno and P. Robertson, "WiSLAM: Improving FootSLAM with WiFi," *2011 Int. Conf. Indoor Position. Indoor Navig. IPIN 2011*, no. September, pp. 21–23, 2011.
[21] A. R. Sandeep, Y. Shreyas, S. Seth, R. Agarwal, and G. Sadashivappa, "Wireless Network Visualization and Indoor Empirical Propagation Model for a Campus WI-FI Network," pp. 730–734, 2008.
[22] B. Li et al., "Hybrid method for localization using WLAN," *Spat. Sci. Conf.*, no. August, pp. 12–16, 2005.
[23] Y. Hu, "Efficient, High-Quality Force-Directed Graph Drawing," *Math. J.*, vol. 10, no. 1, pp. 37–71, 2005.
[24] A. Moreira, I. Silva, F. Meneses, M. J. Nicolau, C. Pendão, and J. Torres-Sospedra, "Multiple simultaneous Wi-Fi measurements in fingerprinting indoor positioning," in *2017 Int. Conf. Indoor Position. Indoor Navig. IPIN 2017*, vol. 2017-Janua, pp. 1–8, 2017.
[25] J. Torres-Sospedra et al., "Off-Line Evaluation of Mobile-Centric Indoor Positioning Systems: The Experiences from the 2017 IPIN Competition," *Sensors*, vol. 18, no. 2, p. 487, 2018.