Indoor Localization using Wi-Fi

Karan Sanwal
Computer Science Department
University of California, Los Angeles
Los Angeles, California, USA
karansanwal@cs.ucla.edu

Shweta Sood
Computer Science Department
University of California, Los Angeles
Los Angeles, California, USA
shwetasood@cs.ucla.edu

Sonali Garg
Computer Science Department
University of California, Los Angeles
Los Angeles, California, USA
sonaligarg@cs.ucla.edu

***Abstract*— This paper presents a novel smartphone based indoor localization technique that uses Wi-Fi signals to determine the location of a user within a building with minimal effort in fingerprinting. To achieve the same, we propose a self-designed Machine Learning Algorithm, "Optimistic Inverted Comparative KNNs" that handles major challenges faced by indoor positioning systems. The proposed framework doesn't require any external hardware, or make any assumptions about the environment it is deployed in and only relies on Received Signal Strength (RSS) values received on the smartphone. The framework is independent of the phone it is trained on and only requires six seconds to train per location. This is by far the smallest time required for training in the literature. It was deployed in the Computer Science Building at UCLA and achieves an accuracy of over 87%.**

Keywords*— *Optimistic Inverted Comparative KNNs, RSS, Fingerprinting, Calibration, Normalization

I. INTRODUCTION

Indoor Localization is a tricky problem as the Global Positioning Systems can only pinpoint our location to a building and not where we are present in the building. With the growth of ubiquitous computing there is an accelerating demand for a more accurate and granular location prediction. It has various applications such as Smart TVs, or ACs may switch on/off based on user's presence in a room.

Earlier work relied on installation of additional hardware [15-17] on walls or ceilings to perform indoor positioning. Moreover, many Bluetooth and Wi-Fi based techniques use data from wireless hardware, such as Simple Network Management Protocol (SNMP) traps and Bluetooth beacons. These require administrative access to network hardware. SNMP traps are messages sent by a mobile phone to SNMP manager. These traps create privacy concern as they expose user information because they are unencrypted. Not only this, the deployment costs are high and require regular maintenance. Thus, a lot of recent work explores the use of Wi-Fi RSS signals to build a fingerprint of the indoor environment and use this for localization. To this effect, [10] explores the need to reduce the quantity of time and data required to build a fingerprint.
Our proposed framework solves the major traditional Wi-Fi fingerprinting challenges:

1. Multiple Signal Reflection - As the signals may get reflected from various obstacles before reaching the smartphone, it leads to degradation in signal values.
2. Environmental Conditions - The change in environment conditions (humidity, breakdown of an Access Point) can affect the strength of RSS values received on the smartphones. This also calls for the need to build a dynamic system that can expand or shrink based on the availability and addition of new routers, detailed in Section III.H.
3. Calibration - The reception of Wi-Fi Signals varies across smartphones due to differences in chipsets and antenna design. [10] has explored the need for a robust Calibration to make the system independent of smartphones. This paper uses normalization to overcome this problem.
4. Large quantity of time and data required for fingerprinting - The system trains in six seconds for each location. This is a huge speedup over earlier work [10-11] that requires 1-2 minutes to train.
5. Translation, orientation and rotation of smartphone - The smartphone's orientation can affect the reception of RSS signals and hence the fingerprint may get biased to a specific orientation.

Optimistic Inverted Comparative KNNs detailed in Section III solves all the above challenges without compromising the accuracy of the system. It requires only six seconds to train per location, doesn't make any assumptions about the positioning of APs or require any external input. It is not battery hogging for the phone and is dynamic to accommodate new APs and flexible to changes in the environment.

Key Contributions: The algorithm uses randomization of RSS signals, *Random Axial Bagging*, which is perceived as noise by earlier literature to build a robust system. The large variance in RSS signals can be helpful in deciphering location specific variations. Moreover, minimum penalty is associated with every location. The location with least penalty is chosen, making the algorithm *optimistic*. We also show that using a hierarchical structure of floor prediction followed by room prediction improves the performance. We call this, *Hierarchical Granularity Progression*. Another thing that we do differently from literature is that, we use a '*Store-all-technique*' for RSS values instead of only storing median/ mean or any other statistic to summarize RSS values from a router. Our intuition in doing this was that all these RSS values represent information specific to that location - signals bouncing off walls/ ceilings/ furniture. All these are detailed in Section III. In Section VI, we show how the RSS values across different locations correlate to the distances between those locations. The distances between RSS values of closer locations are smaller. This makes KNN based approaches successful. This paper also shows how crowdsourcing (Section III.I.) can be leveraged to update the training data over time across several users and make the system more scalable.

The rest of the paper is organized as follows: Section II details prior work done in the field of indoor localization. Section III describes the algorithm and how it overcomes the various challenges of indoor localization discussed in this section. The localization algorithm is a self-designed Optimistic Inverted Comparative KNN. The data collection phase and experiments conducted to evaluate the system are described in Section IV. We deployed a client server arc ecture on Boelter Hall of UCLA to test the performance of the system as shown in Fig. 1.

The client-server architecture is described in Section V. We evaluate the performance of our localization technique in various scenarios (rotating and throwing phones, walking while logging, testing at a different time compared to training) through real-world experiments using multiple mobile phones. In Section VI we discuss the results of our experiments. Section VII presents our concluding remarks.

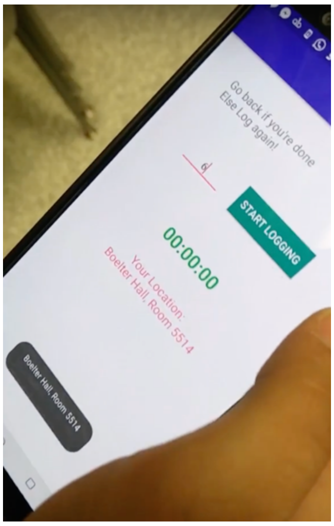


Fig. 1: Client app shows prediction of a location in Boelter Hall, UCLA

II. Related work

Work in this problem domain is often divided into two categories RF fingerprinting based methods and model-based methods. Since, the former tends to perform better, we talk about that in this section.

Early work in the field includes the RADAR [1] system proposed by Microsoft Research that was able to achieve accuracy in the 2-3 m range for indoor localization using WLAN RSS values, and Horus [2], a system by University of Maryland that models the RSS distributions (by identifying reasons for variations in signal) from APs, and improved accuracy considerably.

Further, many studies like PlaceLab [3], ActiveCampus [4] used radio beacon-based approaches to identifying location. However, fingerprinting approaches in general had a major drawback of site-survey being a highly time and effort consuming process. Various papers tried using data from a variety of sensors to mitigate this.

[5] presented a crowdsourcing system based on used Pedestrian Dead Reckoning (PDR) technique as a way to reduce site-survey effort leveraging data from inertial sensors accelerometer, gyroscope etc along with the map of the indoor space in addition to Wi-Fi signals.

HiMLoc [6] uses a particle filter to combine PDR with a crowd-sourced Wi-Fi fingerprinting system. Similarly, a technique WILL [7], relies on the physical floor plan and accelerometer data. [8] also used PDR technique and further, tires to avoid recalibration in updating the RF fingerprint. A more recent paper [9], uses step, map, and light information to improve accuracy to promising results.

However, in these techniques, while the direct manual effort is reduced due to most data collection happening in the background and not depending on input from the user, the actual time taken is not affected much. CAIL [10] just uses Wi-Fi data and tries to reduce the time for site survey by handling device heterogeneity by a cross calibration technique, that requires test data only at some small fixed number of locations. Our paper, overcomes the need for cross calibration by doing normalization of the RSS values. However, the breakthrough was FreeLoc [11], that drastically reduced the time for accurate RSS fingerprints (down to ~1 minute) and proposed a calibration-free method for indoor localization using just Wi-Fi signals. The important observation here was that while absolute Wi-Fi signal strength could vary across devices, their if RSS at location A is greater than that at location B for one device, this trend would be followed across devices. Given enough APs, this could be used to pinpoint location accurately.

This also meant that crowdsourcing data collection became trivial. [12] pointed out that [11] loses fine grained information by just using whether signal strength is greater/less and used the idea of differential RSS fingerprints to solve the problem of device heterogeneity while crowdsourcing. [13] used ensemble-based classifiers on weighted averages (to reduce temporal variation) of RSSI to predict location. [14] proposed integrating human feedback to a baseline Wi-Fi fingerprinting model and showed improvement in performance through the same.

III. Proposed Algorithm

Given that our final ambition was to create an authentic mapping system , it made sense to design a structure that is a map-based-algorithm. Other algorithms – such as Neural Networks or even Gradient Boosted trees would be great only if they are effectively able to create a coordinate system based on Wi-Fi – strengths. To reduce training time – we decided to use a variant of K nearest neighbours to create the final blue-print.

We couldn't use naïve K – Means for the following reasons :

1) Multiple RSS values at the same time .
2) Do not have signal strengths for all routers at the same time, so cannot create a coordinate point.
3) Cannot be a dynamic scalable system
4) Unable to tackle the various issues related with the Wi-Fi signals :
    a) Volatile changing signals
    b) Dependence on environmental conditions
    c) Multiple reflection points
    d) Dying and new routers getting added every day.

e) High volume of data.
f) Changing granularity requirement
g) Hardware dependency
h) Translation and rotational variance.

For the above mentioned reasons, we had to design a different variant of K-Nearest Neighbours, and the realization of that was built on the following principles :
.

A. *Using Relative Signal Strengths instead of Absolute RSS Values*

This solves the problem of changing RSS values in the same position and hardware and environmental dependencies. Since the RSS values might change, using Nearest Neighbour algorithm with the absolute RSS values is guaranteed to perform poorly.

Instead of seeing the absolute RSS values, we instead analyse the relative comparisons between them. To not loose information by plain comparisons alone, we instead normalize all the received signals in the location. So, now we not only the order of the signal strengths but also know the spacing between them. This is used to create a much more effective blueprint, and was shown to decrease the fluctuations in map coordinates.

B. *Using the 'Store-all-technique'*

Most papers in literature were focussed on techniques to reduce the various RSS values received from one router at one location to just one value. To produce such a statistical measure, they had to wait and measure data for a long time while training and testing. This was a slow process and also lead to a loss in information as reduction was taking place. We overcame this by using a simple solution – storing all points and not carrying out reduction. However, this created new complications. Example, if we have 30 values from 5 routers, then we have 24300000 coordinates to map in our Nearest neighbour train-set. This would slow down both test and train time. We tackled this by using a concept that we call '*optimism*', and this required us to add only 150 values to the train-set.

C. *Using Optimism*

Optimism is added to the Nearest Neighbour algorithm as follows: To find your coordinate in the system, you go to every axis and choose the axial point in train that is the closest to the axial point in the test coordinate. How the test coordinate is created is decided by a technique we call randomized axial bagging and we would cover that next. But once this is done, a penalty of how far the closest point was, with respect to axial domains is calculated. The location with the least penalty is outputted as answer.

D. *Using Random Axial Bagging*

The 'Store-all' solves the problem of multiple RSS values in train set, but does not elaborate how the same problem would be dealt with during test time. During test time, a random point from each axis (in this case, every axis is actually a router) will be chosen. A coordinate system will be created and using 'optimism' a prediction would be made on location. This process would be repeated multiple times, so coordinates would be sampled from the test data and the location with the majority of the votes would be predicted as the final location.

E. *Using Hierarchical Granularity Progression*

A major issue with KNNs is that they cannot be ensembled together because it is almost impossible to introduce diversity between two KNNs created from the same data. We solve this problem by changing the domain space within the same data. So now two KNNs work on different domains. These domains are decided based on the required granularity. Example, in our case, we used two KNNs - one for floor and the other for room. Once the first KNN gave output of what floor the predicted location is in, the second KNN only worked between the rooms of that floor alone. This gave rise to more confident outputs.

F. *Using Neighborhood matching based Confidence*

For maximal use of our application, we must not only output a prediction, but also a confidence measure on how good we think the prediction is. This is useful in multiple scenarios, e.g., if the confidence value is low, the client could re-take the sensor data before making a prediction.

To incorporate confidence in our algorithm, we predict not just the room , but also the top 5 closest rooms to it. Given our algorithm, the hierarchical structure guarantees the closest rooms to be all from the same floor. Confidence is then reported as how greatly is the predicted neighbourhood overlapping with the actual neighbourhood of the phone.

This removes anomalous matched test points and further increases the accuracy. However, this increase in performance comes at a cost of possible increased test time, which is why we leave '*the re-do confidence*' as a hyper-parameter to be decided at the client's end.

G. Using Hit-ratios to Limit the training size

It is quite easy for us to determine which parts of our algorithm are good training points and which are just pure noise. All we have to do is a keep a score of which training points are actually matched more frequently to the test points compared to others. We can hence trim off the points which aren't matched much. This correlates closely with the way cache is maintained using 'Hit-ratios' in a Computer Memory system.

H. Using axis ensembling for handling new and dead routers

One of the key problems with Wi-Fi signals is that their source, the routers, are constantly increasing and upgrading. Consequently, it may happen that previously known router is no longer available and a new router is introduced into the system.

For missing router, we assign it an RSS value of -150 dB which is representative of a very weak value. This makes sense as a router not available can be assumed to be available with a negligible strength.

When a previously unknown router is seen by the system, it makes its predictions based on the other router and adds the recorded RSS values to the location training set.

I. Enhancing performance using Crowd-Sourcing

With more and more people relying on smartphones for their day to day activities we have the opportunity to use this huge data and customer support to make our product better. The basic idea is that anyone using the app will also contribute to increasing its performance. With a possibility of a large customer base, our last challenge was to design a crowd sourcing protocol.

The protocol works by superimposing a clustering algorithm over our KNN map. Users in the same cluster will be prompted to enter their location or verify the prediction. This would then add on data in large values.

IV. Data collection and experiments

For evaluation of the proposed algorithm, a total of 3 phones have been used as shown in Table I. The data was collected for 33 uniformly distributed locations inside the Computer Science building of UCLA. Locations included places, such as canteen, classrooms that have furniture, such as chairs and desks, to open areas like corridors. The building contains APs catering to a 1000+ student body, administrative staff, and faculty. These locations were uniformly distributed across three floors of the building. Fig 3. illustrates the floor map of the third floor of the building. The logging varied from 10 AM in the morning to 11 PM night. The RSS values in dBm were logged at a frequency of ⅙ Hz, i.e., 1 Wi-Fi scan per 6 seconds, multiple times at all the locations. Each data log collected at a location spanned 6 seconds, logged at the 33 locations, at different times, over multiple days.

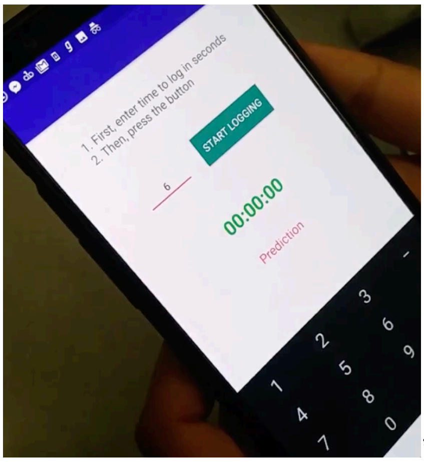
To find the performance, we tested the phone in various conditions: different times of day, different orientations of phone, over multiple phones. We also tried shaking, throwing the phone and walking, running while logging signals. The results in Section VI show that our system was invariant to all these conditions.

Model	CPU
OnePlus 5T	Octa-core (4x2.45 GHz Kryo & 4x1.9 GHz Kryo)
OnePlus2	Octa-core (4x1.56 GHz Cortex-A53 & 4x1.82 GHz Cortex-A57)
Gionee F103 Pro	Quad-core 1.3 GHz

Table I: Configuration of mobile phones used in the experiments

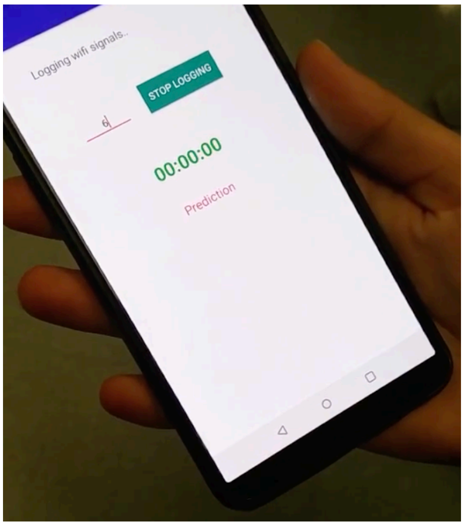
V. Demo

To test the efficacy of the system, we deployed a client server architecture. The client is an Android Application, created using Android Studio, that creates a background service to log Wi-Fi RSS signals at ⅙ Hz. Once the user starts the app, he specifies the number of seconds for which logging has to be done and then starts logging. Now, the user can either wait for the specified seconds or stop logging midway. The BSSIDs and their respective RSS values are written to a csv file. This file is sent to the server via Asynchronous HTTP push request. The server is a Xampp server written in PHP. The server runs the proposed algorithm and produces a prediction that it pushes to the client. The client queries the server for the prediction and then the server responds with the result. The app finally displays the result. The illustration of the architecture is depicted in Fig 2.



,

Step 1: Enter the amount of time user wants to log and press ‘Start Logging’ button



Step 2: Wait for the designated time to get over or press ‘Stop Logging’ if you’re done

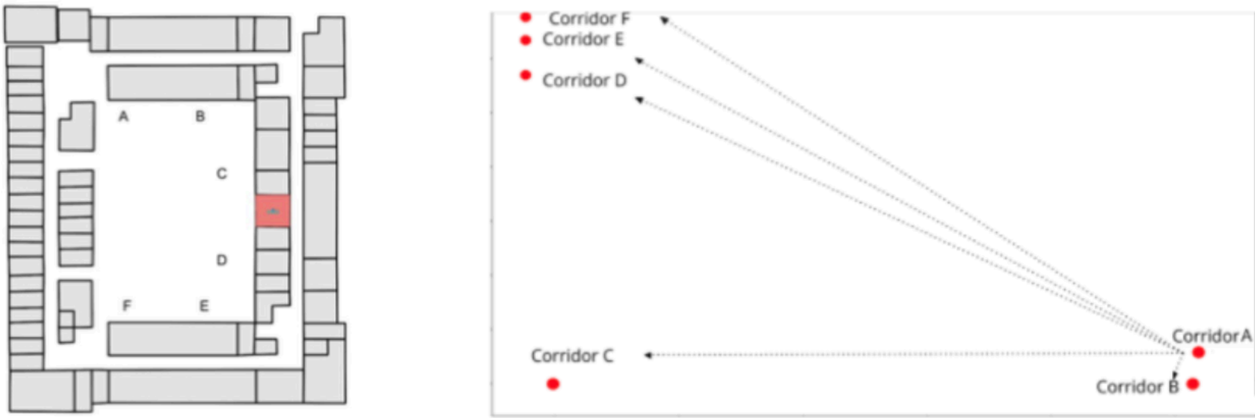
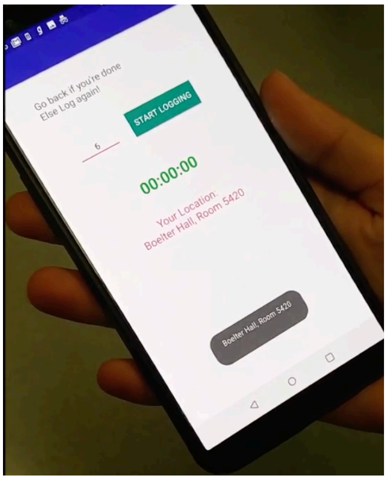


Fig. 3: RSS values at corridor A and B are closer to each other as compared to those at C,D,E,F. This also correlates to the distances shown in the adjacent map. Thus, RSS can be used to capture distances between locations



Step 3: Server sends location prediction which is displayed on the app

Fig. 2: Step 1 to 3 provide walkthrough of the app, from starting logging to location prediction

VI. Results

We did pessimistic testing for our system. Everything that could go wrong with localization due to variations in Wi-Fi signals was tried including but not limited to testing in crowded as well as empty classrooms, testing by rotating and even throwing phones, walking while logging, testing at a different time compared to training, collecting data and testing data at different coordinates in a room. A client - server (Android – PHP/Windows) based application was created for evaluation of the frameworks.

We conducted experiments with 33 locations, with around thirty 6-seconds samples per location. Given the nature of the problem, training and testing were done on different phone models, with testing done 20 days after training, and distributed throughout the day at different times.

We achieved an exact prediction accuracy of 87.58% i.e. we were able to predict the exact room with an accuracy of 87.58%. Our one-block prediction accuracy i.e. with maximum 1 block error as per neighborhood defined earlier was 97.04%.

For comparison, Samsung's paper on Wi-Fi localization [13] published in March 2018 achieves an accuracy of 86% in just 4 locations. [Current System deployed on Boelter Hall - 3rd, 4th, 5th Floor]

Hence, in this paper, we are able to tackle the challenges presented by this problem. We only using Wi-Fi signals, hence there is no extra installation cost etc. We are able to solve the multiple reflection problem because our algorithm considers all possible RSS values for that location for that router and matches optimistically against the closest one. Since we keep multiple values (possibly obtained from different reflections), this seems to automatically handle the issue of translation, rotation and orientation of the smartphone. This also helps us reduce the time required drastically as now we do not need to take average over a large time window to get a good approximation of the RSS value. We are able to remove the need for calibration by normalization for each location individually, which basically gives us a RF fingerprint for the location. Hence, we are able to use relative RSS values like [11] while at the same time retain their proportions which leads to more granularity. Further, we propose a crowdsourcing algorithm to deal with issues due to change in environmental conditions.

VII. Conclusion and Future Work

The proposed algorithm hence performed excellent when tested on the Boelter Hall of UCLA. To the best of our knowledge, the 6 second test time is the fastest in current literature whilst maintaining such a high accuracy. The proposed algorithm also seems to be accommodating dynamism and scalability features that are highly appreciated when the research is put on production. As a part of future work, we would like to implement strict crowd sourcing protocol so that the crowd source data is clean and noise free. Also, taking into account other sensors, like accelerometer, gyroscope combined with environmental information like the weather and state of art machine learning algorithms would be a great boost to performance maintenance at time of anomalous conditions.

References

[1] Bahl, Paramvir, and Venkata N. Padmanabhan. "RADAR: An in-building RF-based user location and tracking system." *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*. Vol. 2. Ieee, 2000.

[2] Youssef, Moustafa, and Ashok Agrawala. "The Horus WLAN location determination system." *Proceedings of the 3rd international conference on Mobile systems, applications, and services*. ACM, 2005.

[3] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter et al., "Place lab: Device positioning using radio beacons in the wild," in International Conference on Pervasive Computing. Springer, 2005, pp. 116–133.

[4] W. G. Griswold, P. Shanahan, S. W. Brown, R. Boyer, M. Ratto, R. B. Shapiro, and T. M. Truong, "Activecampus: experiments in community-oriented ubiquitous computing," Computer, vol. 37, no. 10, pp. 73–81, 2004

[5] Rai, Anshul, et al. "Zee: Zero-effort crowdsourcing for indoor localization." *Proceedings of the 18th annual international conference on Mobile computing and networking*. ACM, 2012.

[6] Radu, Valentin, and Mahesh K. Marina. "Himloc: Indoor smartphone localization via activity aware pedestrian dead reckoning with selective crowdsourced wifi fingerprinting." *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*. IEEE, 2013.

[7] Wu, Chenshu, et al. "WILL: Wireless indoor localization without site survey." *IEEE Transactions on Parallel and Distributed Systems* 24.4 (2013): 839-848.

[8] Chang, Kyungmin, and Dongsoo Han. "Crowdsourcing-based radio map update automation for Wi-Fi positioning systems." *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Crowdsourced and Volunteered Geographic Information*. ACM, 2014.

[9] Cheng, Hui-Chun, et al. "Smartphone-based Pedestrian localization algorithm using inertial and light sensors." *Applied System Innovation (ICASI), 2017 International Conference on*. IEEE, 2017.

[10] Tyagi, Geetali, et al. "CAIL: Cross-calibration for accurate indoor localization without extensive site survey." *Communications (NCC), 2017 Twenty-third National Conference on*. IEEE, 2017.

[11] Yang, Sungwon, et al. "FreeLoc: Calibration-free crowdsourced indoor localization." *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013.

[12] Laoudias, Christos, Demetrios Zeinalipour-Yazti, and Christos G. Panayiotou. "Crowdsourced indoor localization for diverse devices through radiomap fusion." *Indoor Positioning and Indoor Navigation (IPIN), 2013 International Conference on*. IEEE, 2013.

[13] Singh, Vishal, Gorish Aggarwal, and B. V. S. Ujwal. "Ensemble based real-time indoor localization using stray WiFi signal." *Consumer Electronics (ICCE), 2018 IEEE International Conference on*. IEEE, 2018.

[14] Luo, Yan, Yuanzhu Peter Chen, and Orland Hoeber. "Wi-Fi-based indoor positioning using human-centric collaborative feedback." *Communications (ICC), 2011 IEEE International Conference on*. IEEE, 2011.

[15] R. Want, A. Hopper, V. Falcao, and J. Gibbons, "The active badge location system," ACM Trans. Inf. Syst., vol. 10.

[16] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The anatomy of a context-aware application," in Proceedings of MobiCom '99.

[17] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in Proceedings of MobiCom '00.