DeepLocNet: Deep Observation Classification and Ranging Bias Regression for Radio Positioning Systems

Sahib Singh Dhanjal, Maani Ghaffari, and Ryan M. Eustice

***Abstract*— The WiFi technology has been used pervasively in fine-grained indoor localization, gesture recognition, and adaptive communication. Achieving better performance in these tasks generally boils down to differentiating Line-Of-Sight (LOS) from Non-Line-Of-Sight (NLOS) signal propagation reliably which generally requires expensive/specialized hardware due to the complex nature of indoor environments. Hence, the development of low-cost accurate positioning systems that exploit available infrastructure is not entirely solved. In this paper, we develop a framework for indoor localization and tracking of ubiquitous mobile devices such as smartphones using on-board sensors. We present a novel deep LOS/NLOS classifier which uses the Received Signal Strength Indicator (RSSI), and can classify the input signal with an accuracy of 85%. The proposed algorithm can globally localize and track a smartphone (or robot) with *a priori* unknown location, and without an accurate prior map of the WiFi Access Points (AP). Through simultaneously solving for the trajectory and the map of access points, we recover a trajectory of the device and corrected locations for the access points. Experimental evaluations of the framework show that localization accuracy is increased by using the trained deep network; furthermore, the system becomes robust to any error in the map of APs.**

I. Introduction

Low-cost indoor localization solutions using radio signals such as WiFi and Bluetooth have long been studied. Radio signals are easily distorted by the presence of dynamic objects, the room temperature, dust, and even humidity. Furthermore, shadow fading and multipath propagation severely hinder the reliability of signal strength for ranging. The current state-of-art of radio-based positioning techniques [1], [2] are broadly based on the following four distinct categories: (i) Received Signal Strength Indicator (RSSI), (ii) Angle Of Arrival (AOA), (iii) Time Of Arrival (TOA), and (iv) Physical Layer Information (PHY). For further details of the above approaches please see [1].

Except (i), all of the above methods require specialized hardware for obtaining range measurements from WiFi Access Points (APs). This requirement limits the applicability of these methods in non-commercial applications. Once range measurements are obtained, positioning techniques such as spherical or hyperbolic positioning can be used to localize the device [1]. In order to improve the positioning accuracy, one can combine measurements from multiple sensors such as GPS, magnetometer, and camera using filtering methods such as Kalman filtering or particle filtering [3]. In this work, we are concerned with scenarios where only the RSSI information using a commodity WiFi receiver is available. Such information is ubiquitous and available on even the cheapest smartphones nowadays. We develop a "light-weight" deep neural network for accurately classifying between Line-Of-Sight (LOS) and Non-LOS (NLOS) with the capability of easily being deployed on a smartphone.

Fig. 1: The Fetch Mobile Manipulator and the environment in which we performed our tests. We manually moved it around using a remote control and recorded it's pose and the WiFi signatures of all available APs throughout the trial. The recorded poses were used as ground truth and WiFi signatures used as measurements. Since we knew positions of all APs, we manually labeled all data as LOS/NLOS and calculated the euclidean distances using the floor plan.

The ranging methods mentioned earlier also heavily rely on one of the following two prerequisites about the environment: (i) construction of a radio fingerprint of the entire environment, or (ii) accurate map of the position of each AP. Both of these methods have their associated pros and cons. Radio fingerprinting requires an initial dry run in the environment to record information. The entire area is divided into a grid, and two data points are recorded for each location: (i) the RSSI from each AP, (ii) the standard deviation of each signal. This process is more robust than using a map; however, can be very cumbersome and is not conveniently scalable. Similarly, getting accurate maps of the environment is not always possible.

In this work, we relax the assumption of having an accurate initial map by estimating the location of the APs as well as the trajectory of the robot/device. In addition, we introduce a novel deep learning-based LOS/NLOS observation classification and ranging bias regression that is integrated within the widely used particle filtering-based localization and Simultaneous Localization And Mapping (SLAM) frameworks.

The authors are with the Robotics Institute, University of Michigan, Ann Arbor, MI 48109 USA.

M. Ghaffari and R. Eustice are also with the Department of Naval Architecture and Marine Engineering, University of Michigan, Ann Arbor, MI 48109 USA {sdhanjal, maanigj, eustice}@umich.edu.

Manuscript 1961 submitted to 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2019.

A. Contributions

The main contributions of this paper are as follows. First, we design a deep network to classify LOS/NLOS signal propagation with 85% using the passive RSSI information only. The use of passive RSSI information assures that we will not require any custom hardware and the framework will work on any commodity smartphone/robot. The network is designed in a way that it can easily run on embedded devices such as most modern smartphones, without a lot of memory overhead. Second, we use the WiFi sensor model in a position estimation and mapping framework to provide an accurate map of the access points as well as a trajectory of smartphone or robot. By utilizing our approach, the tedious process of WiFi fingerprinting, or the cost of additional specialized hardware is eliminated. Finally, we make the code for the framework, the architecture of the deep network, and simulation environments for wave propagation open source:
https://github.com/sahibdhanjal/DeepLocNet

B. Outline

A review of related works is given in the following section. Section III describes the problem statement and formulation. An overview of the proposed framework is given in Section IV, followed by experimental methodology and results in Section V. Finally, Section VI concludes the paper by discussing limitations and achievements of the proposed framework and providing ideas for future work.

II. Related Work

One of the earliest Radio Frequency (RF)-based localization methods, *RADAR* [4], used fingerprinting and environment profiling with commodity hardware to provide for indoor localization. Building on top of that, *Horus* [5] introduced a client-based probabilistic technique aiming to identify and address channel loss in a light-weight package. Even though accurate up to $0.6\,\mathrm{m}$, the major drawback of this method is that localization occurs in two phases: (i) offline map building and clustering phase, (ii) online localization phase.

Similarly, [6] describes a method based on Gaussian process regression and develops a press-to-go package, where an initial run is made for training the model and then online localization takes place. Going a step further, [7] tracks footstep data and integrates it with RSSI based range measurements, IMU, and magnetometer to provide an accuracy of up to $0.9\,\mathrm{m}$. In [8], the authors use a Gaussian processing-based method to classify RSSI data from Bluetooth Low Energy (BLE) beacons into LOS/NLOS which is then used in conjunction with an IMU and a particle filter for localization.

Most recent methods have started using the Physical Layer (PHY) information instead of the MAC Layer RSSI information. In [9], the authors devise a new method called *PinLoc*, which is able to localize a device within a $1\,\mathrm{m} \times 1\,\mathrm{m}$ box. The main observation was that dynamic obstructions in the environment can be statistically reproduced. Then this fact was used to detect LOS signals using Bayesian Inference.

A similar method is described in [10], where the authors extract phase, transmission time, and strength information from the PHY layer and incorporate a classifier to localize. Most of the above methods are based on clustering approaches, which limits the capabilities of the system as there is no guarantee that the dataset will inherently contain clusters. To tackle this, the authors in [11] came up with a unique statistic called the *Hopkins Statistic*, which measures the clustering tendency to recognize environments. Based on this clustering tendency, they were able to model the environments better, and in turn the location of the APs, leading to improved ranging. However, the method heavily relies on Multiple-Input-Multiple-Output (MIMO) techniques, with a number of antennas, and hence suffer from considerable hardware modification, limiting the practical applications.

Other methods such as AmpN [12], leverage the use of amplitude information of the Channel State Information (CSI) from the PHY Layer. These methods measure standard properties such as the *kurtosis*, *Rician K-factor*, *skewness*, and variation among others for each of the CSI amplitudes, and then train a neural network for dynamic classification and recognition of LOS/NLOS signals. The authors of [13] add another layer of filtering to better understand the property of LOS/NLOS propagation. In their work, *Bi-Loc*, they use phase errors in conjunction with the the CSI amplitude data, to propose a deep learning approach for fingerprinting. Using this, they have two modalities: Angle of Arrival and CSI, which is then used for fingerprinting. The authors of [14] go as far as visualizing the CSI heatmap as an image, and then running a deep convolutional network on it to differentiate between LOS and NLOS.

Although the use of PHY level information has now provided means for more accurate LOS/NLOS classification, and in turn localization, the use of custom hardware *such as Network Interface Cards (NICs)* limits its application to truly mobile and ubiquitous devices (such as smartphones). Some of the issues identified to enable real-time LOS identification are as listed below:

- Commodity WiFi devices fail to support precise Channel Impulse Response measurements due to limited operating bandwidth.
- Existing channel statistics-based features require large amount of samples, impeding real-time performance.
- Most LOS identification schemes are designed for stationary scenarios. Even those incorporating slight mobility fail in truly mobile cases.
- Requirement of custom hardware limits the application of these algorithms for truly mobile cases.

In this work, we bring the advances in SLAM to efficiently solve the indoor localization and tracking problem using sensors available in commonly used mobile devices. The main features that distinguishes this work from the available radio signal-based indoor positioning literature are as follows. We develop a deep neural network that can classify between LOS/NLOS using only RSSI information. Our framework

Manuscript 1961 submitted to 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2019.

also does not require the tedious process of fingerprinting (site survey), hence is more scalable. Moreover, the system is robust to the error in the map of access points due to the usage of FastSLAM to localize the device, as well as obtain a map of the environment.

III. Problem Formulation and Preliminaries

We now define the problems we study in this paper and then briefly explain required preliminaries to solve these problems. Let $x_t \in \mathbb{R}^3$ be the device position at time t. The device is initially located at x_0 which is unknown and can only receive the RSSI of a broadcasted signal. Let Z_t be the set of possible range measurements obtained from converting RSSI at time t. Given the set of known APs, we wish to solve the following problems:

Problem 1 (Measurement Model): Let $Z_t \in \mathbb{R}^+$ be the set of possible range measurements at time t that is calculated through a nonlinear mapping $s_t \mapsto z_t$. The measurement model $p(z_t|x_t)$ is a conditional probability distribution that represents the likelihood of range measurements. We are interested in finding the mapping from signal, s_t, to range measurements, z_t, and the likelihood function that describes the measurement noise.

Problem 2 (Positioning): Let $z_{1:t} = \{z_1, \ldots, z_t\}$ be a sequence of range measurements up to time t. Let x_t be a Markov process of initial distribution $p(x_0)$ and transition equation $p(x_t|x_{t-1})$. Given $p(z_t|x_t)$, estimate recursively in time the posterior distribution $p(x_{0:t}|z_{1:t})$.

Problem 3 (Access Point Locations): Let $M = \{m^{[j]}|j = 1, \ldots, n_m\}$ be a set of unknown and partially observable features whose elements, $m^{[j]} \in \mathbb{R}^3$ represent WiFi access point locations with respect to a global frame of reference. Given $p(z_t|x_t)$ and $p(x_{0:t}|z_{1:t})$ recursively estimate $p(m^{[j]}|x_t, z_t)$.

In the first problem, we try to characterize the received signal and through an appropriate model transform it to a range measurement. Furthermore, we need to find a likelihood function that describes the measurement noise. The second problem can be seen as a range-only self-localization problem. Finally the last problem is to estimate the locations of the access points given that we have accurate location of the device. We now state the main assumptions we use to solve the defined problems:

Assumption 1 (Known Data Association): Each access point has a unique hardware identifier that is available to the receiver device. This assumption is usually satisfied in practice as each device has a unique MAC-address that is broadcasted together with the RSSI.

Assumption 2 (only RSSI available): We assume that the only available information to the receiver is the RSSI. This is the common case for existing wireless routers and commercial NICs.

A. WiFi Technology

WiFi is a technology for radio wireless local area networking of devices based on the IEEE 802.11 standards. It most commonly operates on the $2.4\,\mathrm{GHz}$ Ultra-High Frequency (UHF) and $5.8\,\mathrm{GHz}$ Super-High Frequency (SHF) Industrial, Scientific and Medical (ISM) radio bands. These wavelengths work best for line-of-sight. Many common materials absorb or reflect them, which further restricts the range, but can minimize interference between different networks in crowded environments.

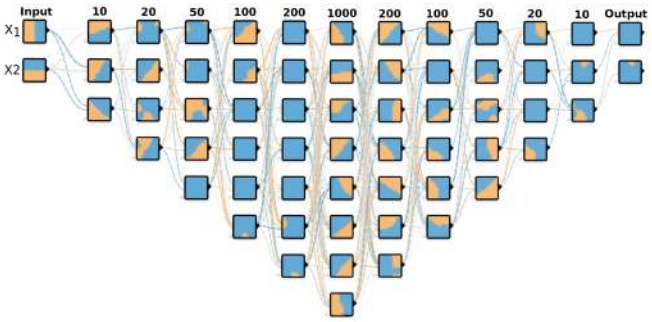


Fig. 2: Neural Network Architecture. x_1 and x_2 represent the Euclidean distance and the distance calculated using Free Space Path Loss (FSPL) formula, respectively. ReLU [15] activation is used between each of the hidden layers. As the number of neurons increases with network depth, the network learns the underlying representation of the data which it then compresses to learn the underlying structure. The final layer consists of two neurons and the SoftMax activation which gives the probabilities of LOS/NLOS signal propagation as output.

B. Particle Filtering

In the problem of localization using RSSI, the observation space is nonlinear, and the posterior density is often multi-modal. Particle filters are a non-parametric implementation of the Bayes filter that are suitable for tracking and localization problems where dealing with global uncertainty is crucial. Hence, in this work, we use a particle filter and its extension FastSLAM [3] to solve the position and mapping problems.

IV. The DeepLocNet Framework

The proposed system framework can be divided into two distinct parts as follows:

A. LOS/NLOS Classifier

The first part of the framework comprises of a deep neural network which is trained to classify between LOS and NLOS signal propagation. The network includes a total of 12 fully connected (FCN) layers which first expand and then contract. The architecture of the network we are using can be seen in Fig. 2. The structure of the neural network is inspired by an autoencoder, where we learn the low-level features representing the underlying data by scaling up, and then scaling down to compress these low-level features into output dimensions. The only difference here is instead of using the input as the output, we are using the class as the output to compute classification loss. `ReLU` [15] activation is used between every layer and is found out to work better than `tanh` activation during the training and experimentation phases. One of the reasons is that the `tanh` activation function causes saturation of multiple neurons during the training process. The network is trained using the cross entropy loss function given by

Manuscript 1961 submitted to 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2019.

$$H(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}), \quad (1)$$

where y is the probability of the true label being 1 *(that is,* $p_{label=1} = y$*)* and $\hat{y}$ is the probability of the label predicted by the network being 1. Cross entropy was chosen over other losses such as MSE or Hinge loss as they are mathematically ill-defined for the classification problem. Before finalizing the network architecture, we tried several different architectures in order to learn a valid representation of the signal propagation. We also tried going deeper than the current depth, but that lead to overfitting on the training data, causing a substantially less testing accuracy than what we obtained using this network. Another reason going deeper did not help is that the number of parameters in the network increased without a substantial increase in classification accuracy.

The number of neurons used per hidden layer were decided on the basis of trial-and-error and the complexity of the signal propagation we wanted to represent. The input data is highly non-linear and signal propagation with shadowing and multi-path effects causing further non-linearity. Because of this complexity, we increased the number of hidden layers until we obtained an increase in both training and testing accuracy. A summary of few of the well performing architectures, number of network parameters, and classification accuracies are shown in Table I.

Neural Network Configurations			
A	**B**	**C**	**D**
5 weight layers	**8 weight layers**	**9 weight layers**	**12 weight layers**
Input Layer (2 x 1 matrix)			
FCN(2,10)	FCN(2,10)	FCN(2,10)	FCN(2,10)
RELU Layer			
FCN(10,20)	FCN(10,20)	FCN(10,20)	FCN(10,20)
RELU Layer			
FCN(20,50)	FCN(20,50)	FCN(20,50)	FCN(20,50)
RELU Layer			
FCN(50,100)	FCN(50,100)	FCN(50,100)	FCN(50,100)
RELU Layer			
FCN(100,2)	FCN(100,50)	FCN(100,200)	FCN(100,200)
	RELU Layer		
	FCN(50,20)	FCN(200,500)	FCN(200,1000)
	RELU Layer		
	FCN(20,10)	FCN(500,1000)	FCN(1000,200)
	RELU Layer		
	FCN(10,2)	FCN(1000,2000)	FCN(200,100)
		RELU Layer	
		FCN(2000,2)	FCN(100,50)
			RELU Layer
			FCN(50,20)
			RELU Layer
			FCN(20,10)
			RELU Layer
			FCN(10,2)
SoftMax Layer			
Classification Accuracies			
63.24%	79.65%	73.88%	**85.25%**
Number of Parameters			
182	262	3882	1762

TABLE I: Neural Network Architectures. FCN(x,y) represents a Fully Connected layer taking x inputs and mapping it to y outputs. RELU LAyer is the ReLU activation [15] function and SoftMax Layer represents the SoftMax activation function.

B. Localization Framework

The localization framework is responsible for the positioning of the device given an estimate of the motion and the received signal strengths of the WiFi signal. We divide this framework into two categories as follows.

1) Map accurately known: In the case where the map is accurately known, we use a particle filter [3] to estimate the position of the robot/device. The Sample Importance Resampling (SIR) particle filter we use consists of three main modules: (i) Motion model (sample), (ii) Measurement model (importance), and (iii) resampling.

Motion model is responsible for generating a set of hypothesis for the current position, based on the previous position and the action taken. More specifically, it specifies a posterior probability $p(x_t|x_{t-1}, u_t)$, that action u_t carries the robot from state x_{t-1} to x_t . Let the number of particles generated be n_p, then the action model generates a position hypothesis for each of the particles.

Measurement model is responsible for assigning the weights (or *importance weights*) to each of the particles sampled from the motion model. Since we only obtain range measurements from the sensor, the measurement function can be given as the distance z_t between the current position x_t and the location $m^{[j]}$ of the j^{th} access point as calculated using the Free Space Path Loss (FSPL) equation. Hence, our measurement model using only WiFi can be given as follows:

$$d_{Euc}(x_t, m^{[j]}) = \left((x_t - m^{[j]})^\intercal (x_t - m^{[j]})\right)^{\frac{1}{2}} \quad (2)$$

$$d_{rssi}(RSSI^{[j]}) = \frac{1}{20} 10^{|RSSI^{[j]}| - K - 20 \log_{10}(f)}, \quad (3)$$

where d_{Euc} is the euclidean distance between the current position x_t and the position $m^{[j]}$ of the j^{th} access point. Similarly, d_{rssi} is the distance of the j^{th} access point based on the RSSI value that the device receives at position x_t. f is the frequency of the signal in MHz. K is a constant that depends on the units for d_{rssi} and f. For f in MHz and d in km, $K = 32.44$ [16].

We implement three methods in the measurement model: (i) No Classification (NC), (ii) Hard (acceptance/rejection) Classification (HC), and (iii) Soft (probabilistic) Classification (SC). The first case represents the naive implementation of the measurement model. In the second case, we only consider LOS signal propagation for ranging (hence hard classification), whereas in the third case, we use probabilities of the signal being LOS or NLOS to calculate an importance weight for the particle. For soft classification, σ can be given as the standard deviation in the maximum range the device is able to sense. We assign it an arbitrary value of 3 assuming that the changes in power transmission would not change the distance(d_{Euc}) more than 3m. We use 2 functions which access the classifier, `getLabel()` and `getProbs()`. Both of them take the euclidean distance and RSSI based distance using FSPL as inputs. While `getLabel()` returns the

Algorithm 1 `DeepLocNet-Measurement-Model`

Require: Set of particles X_t generated by motion model, RSSI value $RSSI_j$ from each AP $m^{[j]}$ at each waypoint in the path, measurement noise sz_j for each AP $m^{[j]}$;

for $i \in X_t$ **do**
  for j in access points **do**
    `Given` x_t, $m^{[j]}$
    Let R be the sensing range of the device
    $\sigma = 3$, $w_{total} = 0$
    $de = d_{Euc}(x_t, m^{[j]})$, $dr = d_{rssi}(RSSI_j)$
    if $de \leq R$ **then**
      if use classifier **then**
        if hard classification **then** ▷ hard classification case
          $label = getLabel(de, dr)$
          if $label$ = `LOS` **then**
            $dz = abs(dr - de)$
          end if
        else ▷ soft classification case
          $pL, pNL = getProbs(de, dr)$
          $dz = pL * |dr - de|$
          $dz += pNL * |dr - N(R, \sigma)|$
        end if
      else ▷ no classification case
        $dz = |dr - de|$
      end if
    end if
  end for
  $w^{[i]} = w^{[i]} * normpdf(dz, 0, sz)$
  $w_{total} += w^{[i]}$
end for

for i in particles **do** ▷ weight normalization
  $w^{[i]} = w^{[i]} / w_{total}$
end for

return X_t

label predicted by the network, `getProbs()` returns the probabilities that the given inputs are either LOS or NLOS. The overall measurement model we use for the particle filter is be given as in Algorithm 1. It shows all the 3 cases we use in classification: *(i)* No Classification, *(i)* Hard Classification, and *(i)* Soft Classification.

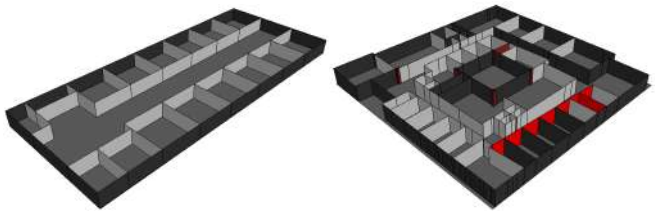
Importance Sampling draws, with replacement, n_p particles from the set X_t of generated and weighted particles using the above two steps. The probability of drawing each particle is given by its importance weight. The resampling essentially transforms the particle set of size n_p into another particle set of the same size by replicating particles with higher weights and, in the end, setting all weights uniformly. The resulting sample set usually possesses many duplicates, since particles are drawn with replacement.

2) Map partially known: : Particle filtering is not effective when the locations of the access points are partially known. The reason is that the there is an added layer of complexity in this problem, where we are not sure of the access point locations giving us inaccurate ranging using the measurement model as discussed in the previous section. To tackle such cases, the FastSLAM [3] algorithm is used to provide us with an effective means of localization, as well as a method of rectifying the locations of the access points using the sensor measurements we have.

The FastSLAM algorithm, in essence, is a particle filter where each particle comprises of a map of the locations of each detected access point in addition to the weight and the position of the device. The access point locations are tracked using an Extended Kalman Filter [3], whereas the robot position is tracked using Particle Filtering. Analogous to the previous case, we divide the measurement model into 3 cases here as well. We refer the reader to [3] for implementation and other details.

2D Localization for the Office Environment			
Alg.	Classifier (Y/N)	Hard/Soft Class. (H/S)	Loc. RMSE (in m)
PF	N	N/A	6.3581
PF	Y	H	1.8491
PF	**Y**	S	**1.5912**
FS	N	N/A	6.9249
FS	Y	H	1.9071
FS	**Y**	S	**1.8169**
3D Localization for the Office Environment			
Algorithm	Classifier (Y/N)	Hard/Soft Class. (H/S)	Loc. RMSE (in m)
PF	N	N/A	8.6115
PF	**Y**	H	**2.2447**
PF	Y	S	2.4827
FS	N	N/A	10.7775
FS	Y	H	3.5645
FS	**Y**	S	**3.3193**

TABLE II: Results for 2D/3D Simulation Experiments in the Office Environment for a single trial. We use 2 algorithms - (i) Particle Filter (PF), (ii) FastSLAM (FS) with the measurement models as described in the DeepLocNet Framework. For each experiment we calculate the RMSE (root mean square error) between each of the waypoints of the ground truth and the localized path.



(a) (b)

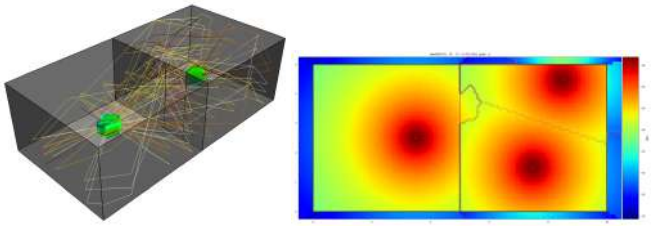
Fig. 3: *(top to bottom)* Environments [16]: (i) Office ($52m$ x $9.5m$), (ii) W2PTIN ($49m$ x $49m$)

V. Experimentation and Evaluation

In this section, we define our experimentation apparatus and methods. We divide this section into the following subsections:

A. Deep Neural Network Training:

The focus of this part was to collect data to train our network. The training was done in 2 phases: *(i)* initial training on simulated data, and *(ii)* network weight refinements on real-world data. Since obtaining floor plans and blueprints with access point locations may not always be a feasible task, we used *Pylayers* [16], an open-source tool, to simulate wave propagation in complex and dense environments. Using this tool, we defined 30+ environments with varying temperatures, humidity, AP locations, AP characteristics (transmission power, antenna directions, etc) and floor plans. Some of the environments are shown in Fig. 3.



(a) (b)

Fig. 4: Simulated wave propagation and Motley Keenan Path Loss in the Defstr ($13m$ x $8m$) environment with 2 and 3 access points respectively.

Results of Motley Keenan Path Loss and 3D Ray tracing for one of the environments can be seen in Fig. 4.

A random walk algorithm was implemented which was able to navigate through a given environment in three dimensions given random start and goal points. We used this algorithm to generate waypoints and calculate the RSSI, euclidean distance, and label (LOS/NLOS) of each AP at each waypoint in the path. We used Bresenham's line algorithm to calculate if there was any obstruction between the said AP and the current waypoint. If there is an obstruction, we label it as NLOS, otherwise LOS. We simulated over 10 million data points and initially trained our neural network on this data.

We also had access to the blueprint of one of the buildings on our campus. We implemented a software which aided us in data-collection given these blueprints as the robot was moved through the environment. We collected data from this building and used it to refine the weights of our deep neural network.

From the several architectures we tested, network D in Table I performed the best both in simulation as well as hardware experiments. The classifier was able to classify LOS from NLOS signal propagation with an accuracy of $\approx 85\%$. From multiple experiments, we found out that the maximum misclassification occurs when the ground truth is NLOS but the prediction is LOS. The reason for this happening is that in cases when the device is near the access point (distance < 10 map units or $2.5m$), the signal propagation might be NLOS because of the presence of an obstruction in between the AP and the device (for instance a wall), the euclidean distance and the distance calculated using the path-loss formula is approximately the same. Hence, though the signal is NLOS, it is predicted as LOS.

B. Experimentation in Simulation:

We developed a simulator for testing the performance of the deep network. The random walk algorithm used in the training phase is used to generate ground truth data between a start and goal point. We start simulating the motion of the robot along this path. We then calculate the RSSI value at each of the waypoints in the path, as stated in the previous section. We also calculate the euclidean distance of the estimated position of the robot from each of the access points (as we know the map within reasonable accuracy). We input these 2 values to the classifier, which then gives us the label (or probabilities) of the signal. We then use this data to update the measurement model accordingly. An example of the localization obtained with the FastSLAM algorithm and soft classification can be seen in Fig. 5. We also attach results for 2D localization in a different environment using FastSLAM and the 3 cases of classifier usage in Fig. 6. We refer the reader to our Github repository for similar results in both 2D and 3D for more complex environments *(such as the Office and W2PTIN)*.

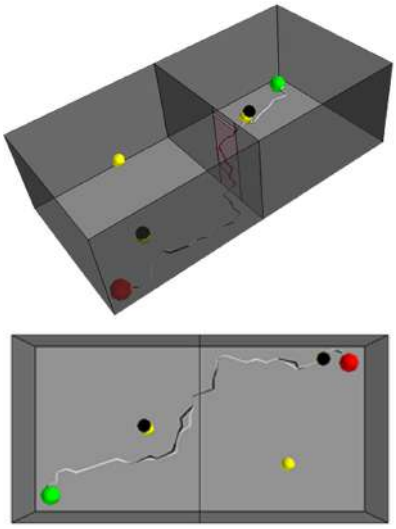


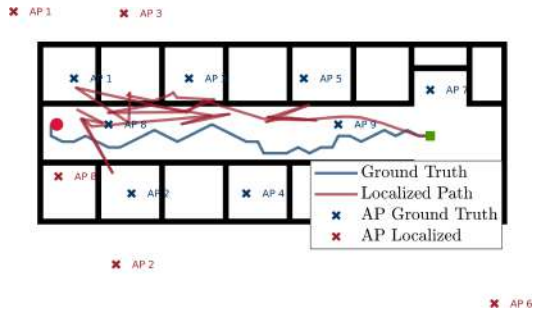
Fig. 5: 3D Localization in the Defstr environment [16] using FastSLAM and soft classification (top to bottom - orthographic view, top view). The yellow spheres represent actual location of access points, black spheres the localized estimates of the access points, the white pipe the ground truth path and the black pipe the localized path. The green and red spheres represent the start and goal positions respectively. One AP is not detected in this case because it is out of sensing range (2m).The red wall is a door, hence the device is allowed to pass through it.

Parameter Name	**Value**
Number of particles	3000
Step size	2m
Motion standard deviation	[rand()*0.8m, rand()*0.8m, rand()*0.8m]
Measurement standard deviation	random value between $[10m - 100m]$
Sensing range	15m
AP location noise	[rand()*10m, rand()*10m, rand()*10m]

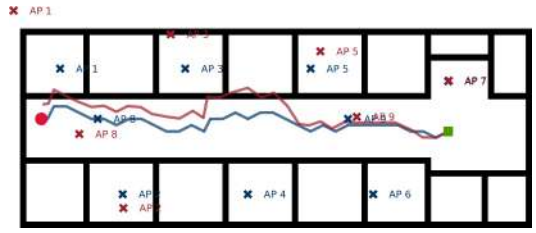
TABLE III: Parameters used for Simulation Experiments. Please refer Section *V-B* for details on each parameter. Ignore the last dimension in case of 2D Localization.

The results obtained in simulation for both the 2D and 3D case for the office environment are given in Table II. A set of 50 experiments each was run for each of the scenarios presented in the given table and the mean root mean square error (RMSE) was calculated. As can be seen in Fig. 7, the RMSE shows substantial improvement for localization using both FastSLAM and Particle Filter.

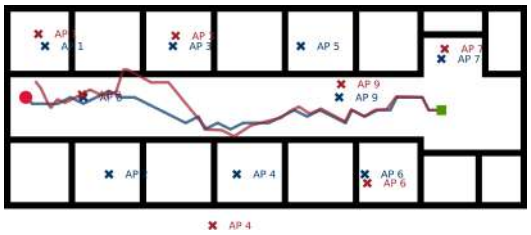
The parameters used for generating these results are given in Table III. The number of particles defines n_p used in the particle filter/FastSLAM algorithm. The step size is the length of step taken during the random walk. The motion standard deviation is used to produce white gaussian noise *(zero-centered gaussian distribution with given standard de-*



(a) No Classification



(b) Hard Classification



(c) Soft Classification

Fig. 6: 2D Localization in the Office environment using FastSLAM: (i) No Classification *(RMSE=2.5737 m)*, (ii) Soft Classification *(RMSE=0.5258 m)*, (iii) Hard Classification *(RMSE=0.5377 m)*. All access points have been detected in all 3 cases, however they're not shown as they are out of the field of view.

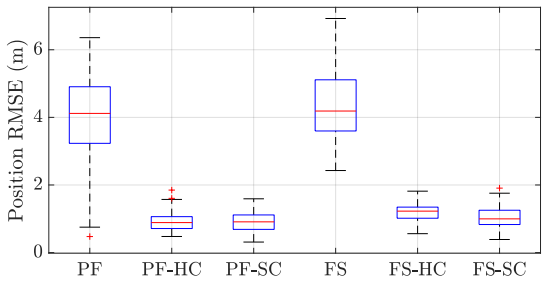
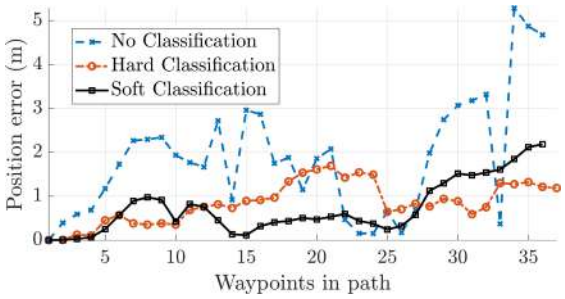
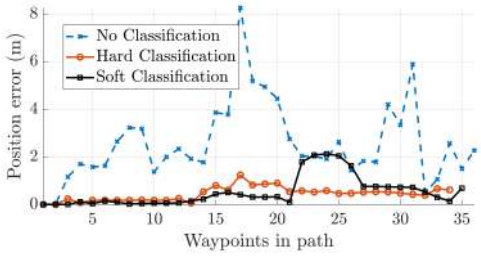


Fig. 7: RMSE for 2D localization over 50 trials in the office environment.

viation) [3] in each dimension (x,y,z) which is taken to be a percentage of the step size. The higher the percentage, the noisier the motion model is considered and a more scattered distribution of hypothesis is produced. Similarly, the measurement standard deviation is used to produce white gaussian noise in the range measurements obtained using the RSSI values of access points. Since RSSI based ranging is very inaccurate, we consider the noise to be anywhere between 10m to 100m. In practice, the noise can be higher than 100m, however, in that case, the localization algorithm would



(a) Particle filter



(b) FastSLAM

Fig. 8: Error of localized path from ground truth at each waypoint in the Office Environment for a single run using (i) Particle Filter and (ii) FastSLAM (corresponds to Fig. 6).

totally rely on the motion model for tracking the location rendering the use of a measurement model ineffective. The sensing range defines the maximum range the device is able to sense for WiFi APs. We take the sensing range of the device to be a safe estimate of 15m. Lastly, the AP location noise determines the noise in the location of every AP in each dimension *(only valid for FastSLAM)*.

C. Hardware Experiments

We used the Fetch Mobile Manipulator for our experiments in one of the buildings of our campus, whose floor plan we had access to. The environment and the robot is as given in Fig. 1. We manually moved the robot around using a remote controller. While in motion, we used an open-source implementation of Adaptive Monte Carlo Localization [17] for obtaining the ground truth of the robot. We broke down the entire continuous path into waypoints with each segment at least 2.5m apart. At each waypoint, we recorded the WiFi signatures (includes signal strength and MAC address) of all the access points present.

Once we gathered all the WiFi and waypoint information, we ran the FastSLAM algorithm for all 3 cases. Particle Filter wasn't run as we had a certain measure of ambiguity in the location of all the access points. Localization is only performed in two dimensions because of the sensing capabilities of the robot.

We assumed the motion standard deviation of the robot to be $1m$ in both x and y axes. All other parameters were same as in the case of simulation. Step size and sensing range are not applicable in this case. The access point locations as well weren't accurate as the floor plan wasn't to scale. Hence we assume a certain error in their locations as well.

Manuscript 1961 submitted to 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2019.

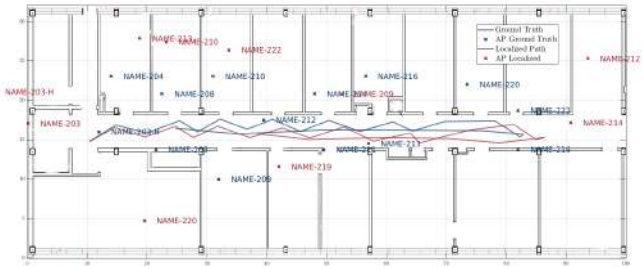


Fig. 9: 2D Localization in the the University of Michigan Naval Architecture and Marine Engineering (NAME) building using FastSLAM and Soft Classification in a single run.

Indoor localization using the Fetch Mobile Manipulator and FastSLAM (FS)			
Alg.	Classifier (Y/N)	Hard/Soft Class. (H/S)	Loc. RMSE (in m)
FS	N	N/A	6.5642
FS	Y	H	2.2361
FS	**Y**	S	**1.4070**

TABLE IV: Results for Hardware Experiments in the University of Michigan Naval Architecture and Marine Engineering building, Ann Arbor for a single trial. We only use FastSLAM (FS) with the measurement models as described in the DeepLocNet Framework. For each experiment we calculate the RMSE (root mean square error) between each of the waypoints of the ground truth and the localized path.

The results obtained using FastSLAM for one of our trials is given in Table IV. The result for 2D localization using soft classification in FastSLAM can be seen in Fig. 9

VI. Conclusion and Future Work

We presented a deep learning-based approach for classifying LOS/NLOS signal propagation, aiding in better localization estimates solely on the based of signal strength measurements. In particular, the proposed method can provide effective localization using the available infrastructure without the use of any custom hardware. We further incorporated the proposed deep learning-based classifier into the measurement model of particle filtering and FastSLAM. Our experiments show that the resulting system can track a person moving through a large and highly- structured indoor environment with accuracy within $2\,\mathrm{m}$. We also present results in large scale indoor environments using signal strength from multiple access points. The presented results in this work show that the DeepLocNet framework performs promisingly better that the available wireless-based positioning systems which have an accuracy of $1-10\,\mathrm{m}$ [18].

Future work includes modifying the network to incorporate information from the PHY layer as well, further improving the accuracy of localization. However, for this to happen, mobile devices would have to be fitted with the required hardware.

Acknowledgment

This work was partially supported by the Toyota Research Institute (TRI), partly under award number N021515, however, this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. We also thank Dr. Corina Barbalata and Deep Robot Optical Perception Lab for providing us with the help and support for all hardware experiments.

References

[1] L. D. Nardis, "3D indoor positioning and navigation: Theory and implementation," 2017, morgner@uni-muenster.de.

[2] Tektronix. (2014) Wi-Fi: Overview of the 802.11 physical layer and transmitter measurements. [Online]. Available: https://www.cnrood.com/en/media/solutions/Wi-Fi_Overview_of_the_802.11_Physical_Layer.pdf

[3] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[4] P. Bahl, V. N. Padmanabhan, *et al.*, "Radar: An in-building rf-based user location and tracking system," in *INFOCOM*, vol. 2, no. 2000. IEEE, 2000, pp. 775–784.

[5] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *Proceedings of the 3rd international conference on Mobile systems, applications, and services*. ACM, 2005, pp. 205–218.

[6] X. He, S. Badiei, D. Aloi, and J. Li, "Wifi ilocate: Wifi based indoor localization for smartphone," in *Wireless Telecommunications Symposium*. IEEE, 2014, pp. 1–7.

[7] J. L. Carrera, Z. Li, Z. Zhao, T. Braun, and A. Neto, "A real-time indoor tracking system in smartphones," in *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2016, pp. 292–301.

[8] M. Ghaffari Jadidi, M. Patel, and J. Valls Miro, "Gaussian processes online observation classification for rssi-based low-cost indoor positioning systems," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2017, pp. 6269–6275.

[9] S. Sen, R. R. Choudhury, B. Radunovic, and T. Minka, "Precise indoor localization using phy layer information," in *Proceedings of the 10th ACM Workshop on hot topics in networks*. ACM, 2011, p. 18.

[10] Z. Zhou, Z. Yang, C. Wu, L. Shangguan, H. Cai, Y. Liu, and L. M. Ni, "Wifi-based indoor line-of-sight identification," *IEEE Transactions on Wireless Communications*, vol. 14, no. 11, pp. 6125–6136, 2015.

[11] Z. Li, Z. Tian, M. Zhou, and Y. Jin, "Wi-vision: An accurate and robust los/nlos identification system using hopkins statistic," in *Global Communications Conference*. IEEE, 2017, pp. 1–6.

[12] F. Xiao, Z. Guo, H. Zhu, X. Xie, and R. Wang, "Ampn: Real-time los/nlos identification with wifi," in *IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–7.

[13] X. Wang, L. Gao, and S. Mao, "Biloc: Bi-modal deep learning for indoor localization with commodity 5ghz wifi," *IEEE Access*, vol. 5, pp. 4209–4220, 2017.

[14] X. Wang, X. Wang, and S. Mao, "Deep convolutional neural networks for indoor localization with csi images," *IEEE Transactions on Network Science and Engineering*, 2018.

[15] A. F. Agarap, "Deep learning using rectified linear units (relu)," *arXiv preprint arXiv:1803.08375*, 2018.

[16] N. Amiot, M. Laaraiedh, and B. Uguen, "Pylayers: An open source dynamic simulator for indoor propagation and localization," in *Communications Workshops (ICC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 84–88.

[17] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization: Efficient position estimation for mobile robots," *AAAI/IAAI*, vol. 1999, no. 343-349, pp. 2–2, 1999.

[18] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Trans. on Systems, Man, and Cybernetics, Part C*, vol. 37, no. 6, pp. 1067 – 1080, 2007.