Enhancing the Accuracy of iBeacons for Indoor Proximity-based Services

Faheem Zafari[*], Ioannis Papapanagiotou[**], Michael Devetsikiotis[†], and Thomas J. Hacker[‡]

[*]Electrical and Electronics Engineering, Imperial College London, faheem16@imperial.ac.uk
[**]Platform Engineering, Netflix, ipapapa@ncsu.edu
[†]Electrical and Computer Engineering, University of New Mexico, mdevets@unm.edu
[‡]Computer and Information Technology, Purdue University, tjhacker@purdue.edu

Abstract—*Proximity-based Services* (PBS) require high detection accuracy, energy efficiency, wide reception range, low cost and availability. However, most existing technologies cannot satisfy all these requirements. Apple's Bluetooth Low Energy (BLE), named *iBeacon*, has emerged as a leading candidate in this domain and has become an almost industry standard for PBS. However, it has several limitations. It suffers from poor proximity detection accuracy due to its reliance on Received Signal Strength Indicator (RSSI). To improve proximity detection accuracy of iBeacons, we present two algorithms that address the inherent flaws in iBeacon's current proximity detection approach. Our first algorithm, *Server-side Running Average* (SRA), uses the path-loss model-based estimated distance for proximity classification. Our second algorithm, *Server-side Kalman Filter* (SKF), uses a Kalman filter in conjunction with SRA. Our experimental results show that SRA and SKF perform better than the current moving average approach utilized by iBeacons. SRA results in about a 29% improvement while SKF results in about a 32% improvement over the current approach in proximity detection accuracy.

Index Terms—Location Based Services, iBeacons, Internet of Things, Proximity Detection, Kalman Filter

I. INTRODUCTION

Location Based Services (LBS) are services provided to users based on their location. Such services require either identifying the user's exact location, known as micro-location, or obtaining a rough estimate of the user's distance to a certain *Point of Interest* (PoI) known as *proximity-based services* (PBS). The basic requirements of PBS are high proximity accuracy, energy efficiency, wide reception range, low cost and availability. For PBS, the estimation error of user's proximity to the POI must be limited within certain bounds, preferably within one meter [1]. Different technologies have been used for proximity detection including WiFi [2], Radio Frequency Identification Devices (RFIDs) [3], and Ultra Wideband (UWB). However, these technologies are not primarily intended for PBS and do not fulfill the aforementioned requirements.

Bluetooth Low Energy (BLE) is a viable technology for PBS. While different BLE-based solutions are available for PBS [4], a reliable solution requires modifying the BLE stack to accommodate proximity specific protocols and optimizations that are currently not part of BLE. Google's *Eddystone* and Apple's *iBeacon* are two such widely used protocols that are based on BLE and optimized to provide PBS. Our paper focuses on iBeacon protocol as it is the industry standard for PBS [5], but our algorithms can also be used with *Eddystone* based system. The BLE devices using iBeacon protocol are also called iBeacons or beacons. iBeacons periodically transmit a universally unique identifier (UUID) that is picked up by the user's mobile device. Once the user device obtains a
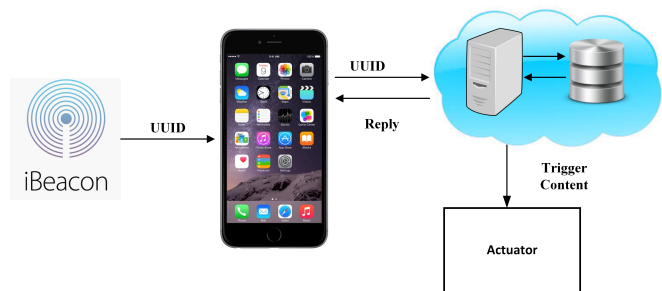

Fig. 1: Working principle of the iBeacon

UUID, it contacts a server to inquire about the UUID and the event associated with the iBeacon. The server responds back with relevant information and can trigger an event such as responding back to the user with a discount coupon or opening a security door based on the user's proximity to the door. Figure 1 shows the working principle of the iBeacon. Since energy efficient and long range BLE is available in modern smart phones, iBeacons would be feasible for PBS if the proximity detection accuracy could be improved to satisfy the requirements for PBS.

In iBeacon-based systems, the strength of the received signal (from iBeacons) at the user device is used as an estimate of how close the user is to the beacons. The user's proximity to the beacon is classified into any of the four zones given in Table I. Different applications could potentially use any of the aforementioned zones to provide PBS. It is therefore fundamental to accurately compute the user's proximity to the iBeacons. A user who enters a store such as 'Starbucks' and is in the 'immediate' zone of the counter, could avoid lengthy queues by leveraging his accurate proximity to the iBeacons. The user can confirm the order through his smartphone and pay for the order based on his proximity. Such service is only possible with accurate proximity estimation and the proximity error being within certain bounds. Because

PBS are primarily provided in indoor environments that are prone to noise due to the presence of obstructions, Apple's *CoreLocation Framework* reports a moving average of RSSI values received from beacons to reduce fluctuation. However, the moving averaged RSSI values fluctuate drastically due to noise and cannot consequently produce an accurate estimate of the actual distance. Therefore, the estimated proximity can be erroneous and not suitable for PBS. Another inherent flaw in iBeacon's current proximity detection approach is that the users are classified into different zones based on specific RSSI values, i.e. it is assumed that all the indoor environments behave identically, but this is unrealistic and further detriorates the performance of beacon based PBS.

To improve the detection accuracy of the iBeacon-based proximity services, we developed two new server-based algorithms that incorporate moving average and Kalman filter to improve the proximity detection accuracy of an iBeacon based system. We propose to leverage the computational power of the server, and not the device, for running the algorithms to reduce the energy consumption of user device, and to exploit the greater computing power of servers. Our first algorithm, the *Server-side Running Average* (SRA), suitable for environments with less interference noise (in less crowded places such as coffee shops or small stores with fewer WiFi Access Points (AP)), improves the proximity detection accuracy of iBeacons by 29% when compared with the current moving average based approach used today by Apple's *CoreLocation Framework*. Our second algorithm, *Server-side Kalman Filter* (SKF), is suitable for large spaces with greater interference noise (typically in more crowded space with a higher number of WiFi APs) and is a modified version of SRA, improves proximity detection accuracy by 32% over Apple's current approach. We have open sourced the aforementioned algorithms along with an end to end micro-location framework[1] to solicit feedback, and to enable collaborations and contributions from the research and corporate community.

The paper is structured as follows: Section II discusses related work. Section III describes our proposed server-based algorithms. Section IV presents our experimental setup and obtained results. Section V presents a detailed discussion of our obtained results. Section VI presents our conclusions.

II. RELATED WORK

A. Related Work

Kumar et al. [6] presented an indoor localization system called Ubicarse that emulates large antenna arrays on user devices through a novel formulation of Synthetic Aperture Radar (SAR). Ubicarse is favorable in terms of accuracy for both micro-location and proximity based services. However, the energy consumption of the Ubicarse localization is high and it requires its users to rotate their devices for localization purposes [6]. Furthermore, the user device must have at least two antennas to emulate large antenna arrays. Klokmose

---

[1] https://github.com/ipapapa/IoT-MicroLocation

TABLE I: The classification of proximity zones based on distance between the user and the iBeacon

Zone	Distance
Immediate	<1 m
Near	1-3 m
Far	>3 m
Unknown	Device not ranged

et al. [2] proposed a WiFi proximity detection system for mobile web applications that is based on proximity adaptive HTTP responses. This approach, despite its low cost, can only provide proximity based services if the user device is generating traffic [2]. Furthermore, WiFi also lacks the necessary accuracy required for proximity detection as described by Ghose [7], and is not energy efficient. Bolic et al. [3] proposed a novel RFID device called *"Sense-a-Tag" (ST)* for detecting and decoding backscatter signals from different tags in its vicinity. The proposed ST can be incorporated in a standard RFID system to improve proximity detection accuracy for IoT. However, the range of RFIDs is a major challenge particularly in large spaces. Ghose et al. [7] proposed a mobile-based system that leverages the Bluetooth capabilities of mobiles for proximity detection. A new path-loss model that takes the mobile orientation into account to improve the system performance is described in [7].

The drawbacks of these technologies is that they are not primarily focused on accurate and energy efficient PBS. In contrast, iBeacon technology is more suited for proximity detection. In our prior work [8], we used iBeacons to provide indoor localization services to any user. We used Particle filtering to track the location of a user with a localization error as low as 0.97 meters. In this paper, we describe two novel server-based proximity detection algorithms that respectively leverage moving average and Kalman filter to improve the accuracy of an iBeacon-based proximity detection system. To the best of our knowledge, this is the first attempt to improve the proximity detection accuracy of *iBeacons* using Kalman filters.

III. OPTIMIZING THE PROXIMITY DETECTION ACCURACY OF IBEACONS

To overcome the problems with the current approach used by iBeacons for proximity classification, we propose two server-based alternatives. Below we describe our algorithms.

A. Server-side Running Average (SRA)

In our first algorithm, *Server-side Running Average*, we collect the RSSI values from the beacons using the user device and report them to a server. Rather than using RSSI directly as a measure of the user's proximity to any specific beacon, we relate it with distance using the path-loss model as described by Kumar in [9] and given in Equation 1. In this equation, n represents a path-loss exponent that varies in value depending on the environment, d is the distance between the user and the beacon, d_0 is the reference distance which is 1 meter in our case, and C is the average RSSI value at d_0.

$$RSSI = -10\,n\,log_{10}(d/d_0) + C \qquad (1)$$

Once the path-loss model is obtained, it efficiently characterizes the behavior of RSSI at different distances resulting in an accurate distance estimate. We believe that using a path-loss model that reflects the characteristics of the environment will improve the proximity detection accuracy as compared with the current approach. To account for the drastic fluctuations in the RSSI, the user is classified in a proximity zone only if three consecutive measurements obtained from the iBeacons classify him in that position through the estimated distance obtained using path-loss model. Algorithm 1 shows the SRA algorithm.

Algorithm 1 Server-side Running Average

1: **procedure** SERVER-SIDE RUNNING AVERAGE
2:     Obtain a path-loss model P_L using site survey
3:     $D_0 \leftarrow 0$ ▷ Initial distance
4:     $P_0 \leftarrow Unknown$ ▷ Initial proximity
5:     Load $RSSI_{recv}$ ▷ RSSI values received from sensors
6:     $RSSI_{filt} \leftarrow RSSI_{recv}$ ▷ iOS filtered RSSI values
7:     $D_i \leftarrow D_0$ ▷ Distance at sample i
8:     $P_i \leftarrow P_0$ ▷ Proximity at sample i
9:     $P \leftarrow P_0$ ▷ Classified proximity
10:    **while** $RSSI_{filt} \neq 0$ **do**
11:       $D_i \leftarrow P_L(RSSI_{filt})$
12:       $P_i \leftarrow Proximity(D_i)$ ▷ Zones using Table I
13:       **if** P_i is in zone x for i= t-2,t-1,t **then**
14:          $P \leftarrow P_i$
15:       **else** $P \leftarrow P$

B. Server Side Kalman Filter (SKF)

Our second algorithm, *Server-side Kalman Filter*, is a modified version of SRA and utilizes Kalman Filtering [10], [11] to reduce the fluctuation in the RSSI as shown in Figure 2. Since proximity is a mere estimation of location rather than exact position, we chose a Kalman filter over a particle filter due to reduced complexity. Using this approach, RSSI values from the beacons are received by the user device which are then forwarded to the server that uses Kalman filtering to reduce signal fluctuations. The smoothed RSSI values are then converted into a distance value using the path-loss model. Like SRA, the proximity is reported in a particular zone only if three consecutive measurements obtained from the iBeacons indicate the proximity of the user to the beacon to be in that zone. This means that the proximity decision for any specific samples depends on the two samples preceding it.

Our Kalman filter based RSSI smoother is based on the work of Guvenc [11]. The state x_i that in our case consists of RSSI and rate of change of RSSI, at time i is a function of the state at time $i - 1$ and the process noise w_{i-1} which is given mathematically by Equation 2. The obtained RSSI measurements z_i at instant i from the iBeacons is a function of the state at $i-1$ and the measurement noise v_i as given by Equation 3 as described in Arulampalam [10].

$$x_i = f(x_{i-1}, w_{i-1}) \tag{2}$$

$$z_i = h(x_{i-1}, v_i) \tag{3}$$

TABLE II: Kalman filter parameter notation

Symbol	Meaning
x	State vector
z	Measurement/observation vector
F	State transition matrix
P	State vector estimate covariance or Error covariance
Q	Process noise covariance
R	Measurement noise covariance
H	Observation matrix
K	Kalman Gain
w	Process noise
v	Measurement noise

The traditional Bayesian based approach consists of the prediction and update stage as described by Guvenc [11] and is given below:

1) Prediction Stage
$$p(x_i|z_{1:i-1}) = \int p(x_i|x_{i-1})p(x_{i-1}|z_{1:i-1})dx_{i-1} \tag{4}$$
2) Update Stage:
$$p(x_i|z_{1:i}) = \frac{p(z_i|x_i)p(x_i|z_{1:i-1})}{p(z_i|z_{1:i-1})} \tag{5}$$
where
$$p(z_i|z_{1:i-1}) = \int p(z_i|x_i)p(x_i|z_{1:i-1})dx_i \tag{6}$$

We assume that both the process noise and measurement noise are Gaussian and the functions f and h in Equations 2 and 3 are linear. Because of the linearity assumption, we can apply a Kalman filter since it is the optimal linear filter.

Due to the aforementioned assumptions, Equations 2 and 3 can be rewritten as described by Guvenc [11]

$$x_i = Fx_{i-1} + w_i \tag{7}$$

$$z_i = Hx_i + v_i \tag{8}$$

where $w_i \sim N(0, Q)$ and $v_i \sim N(0, R)$. Table II lists the parameters of a Kalman filter. The prediction and update stages for the Kalman filter as described by Guvenc [11] are

1) Prediction Stage:
$$\hat{x}_{\bar{i}} = F\hat{x}_i \tag{9}$$

$$P_{\bar{i}} = FP_{i-1}F^T + Q \tag{10}$$
2) Update State:
$$K_i = P_{\bar{i}}H^T(HP_{\bar{i}}H^T + R)^{-1} \tag{11}$$
$$\hat{x}_i = \hat{x}_{\bar{i}} + K_i(z_i - H\hat{x}_{\bar{i}}) \tag{12}$$
$$P_i = (I - K_iH)P_{\bar{i}} \tag{13}$$

The higher the Kalman gain, the higher will be the influence of the measurements on the state. The prediction and update steps are recursive in nature. For the purpose of filtering the RSSI values, we utilize a state vector x_i that consists of the RSSI value y_i and the rate of change of RSSI Δy_{i-1} as given below.

$$x_i = \begin{bmatrix} y_i \\ \Delta y_i \end{bmatrix}$$

Δy_i is dependent on the environment and signifies how drastically RSSI value fluctuates. The higher the noise in the environment, the higher will be the fluctuation. The current value of RSSI y_i is assumed to be the previous RSSI y_{i-1} plus the change Δy_i and process noise w_i^y. Hence Equation 7
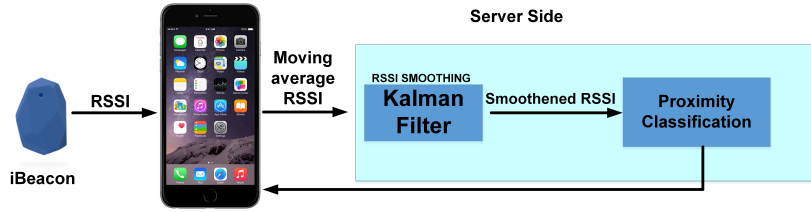
Fig. 2: Proposed Kalman filter-based proximity detection

can be written as

$$\begin{bmatrix} y_i \\ \Delta y_i \end{bmatrix} = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_{i-1} \\ \Delta y_{i-1} \end{bmatrix} + \begin{bmatrix} w_i^y \\ w_i^{\Delta y} \end{bmatrix} \qquad (14)$$

which means that the state transition matrix F is given by

$$F = \begin{bmatrix} 1 & \delta t \\ 0 & 1 \end{bmatrix}$$

The parameter δt is to be adjusted as per the variation in RSSI which depends on the environment. For our set of experiments, δt was taken as 0.2 (using trial and error). Similarly, Equation 8 can be rewritten as

$$\begin{bmatrix} z_i \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y_i \\ \Delta y_i \end{bmatrix} + \begin{bmatrix} v_i^y \end{bmatrix} \qquad (15)$$

The observation matrix H is given by

$$H = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

Parameters P, Q and R used in the experiments were obtained using trial and error, and are given below.

$$P = 100\mathbf{I}_{22}, \quad Q = 0.001\mathbf{I}_{22}, \quad R = \begin{bmatrix} 0.10 \end{bmatrix}$$

The Kalman filter, once calibrated, effectively smooths the RSSI values. The smoothed RSSI values were then input into the path-loss model to obtain distances between the iBeacons and the user, and the user's proximity to the beacon was classified in any of the aforementioned zones. Algorithm 2 shows SKF.

IV. EXPERIMENTAL RESULTS

To evaluate the detection accuracy of our two algorithms, we placed a Gimbal Series 10 beacon in two different rooms (for cross validation) which are 11m × 6m (environment 1 or e_1) and 8m × 4m (environment 2 or e_2) in dimension. The rooms, due to the infrastructure inside, replicate a typical real world scenario in which beacons are utilized. The transmission interval for the beacons was set at 100ms. We used an iPhone 6s plus running the latest iOS version 9.2 and Bluetooth V4.2 as the user device. The iPhone, placed horizontally on a tripod stand, was loaded with our prototype application. Our prototype application primarily has two main functionalities; micro-location (used in our prior work [8]) and proximity-based services. This application can listen to several beacons in its vicinity and then classify them into locations based on the RSSI value. An Intel core-i5 Macbook-pro with 8 gigabytes of RAM, running Apache Tomcat 8.0 and Java 1.8 was used as the server to run the SRA and SKF algorithms.

To obtain the path-loss models for our environments, we put the beacon in a fixed position and noted the average RSSI values on a user hand held device for a number of distances starting from 0 meter up to 7 meters. We collected and averaged 22 RSSI samples at each location. As shown

---

Algorithm 2 Server-side Kalman Filter

1: **procedure** SERVER-SIDE KALMAN FILTER
2:     Obtain a path-loss model P_L using site survey
3:     $D_0 \leftarrow 0$ ▷ Initial distance
4:     $P_0 \leftarrow Unknown$ ▷ Initial proximity
5:     Load $RSSI_{recv}$ ▷ RSSI values received from sensors
6:     $RSSI_{filt} \leftarrow RSSI_{recv}$ ▷ iOS filtered RSSI values
7:     $D_i \leftarrow D_0$ ▷ Distance at sample i
8:     $P_i \leftarrow P_0$ ▷ Proximity at sample i
9:     $P \leftarrow P_0$ ▷ Classified proximity
10:     **while** $RSSI_{filt} \neq 0$ **do**
11:         $RSSI_{filt} \leftarrow KalmanFilter(RSSI_{filt})$
12:         $D_i \leftarrow P_L(RSSI_{filt})$
13:         $P_i \leftarrow Proximity(D_i)$ ▷ Zones using Table I
14:         **if** P_i is zone x for i= t-2,t-1,t **then**
15:             $P \leftarrow P_i$
16:         **else** $P \leftarrow P$

---

in Figure 3a and 3b respectively, we plotted distance vs. average RSSI and then used Matlab's curve fitting function to estimate a curve for distance vs RSSI in both environment 1 (e_1) and environment 2 (e_2). The path loss parameters with 95% confidence interval (CI), obtained through curve fitting for both e_1 and e_2, are listed in Table III.

TABLE III: Path loss parameters for environment 1 (e_1) and environment 2 (e_2)

Parameter	e_1		e_2	
	Value	**95% CI**	**Value**	**95% CI**
n	0.9116	(0.827,0.996)	1.246	(1.139,1.354)
C (dBm)	-62.78	(-64.07,-61.05)	-60.95	(-62.24,-59.66)
R^2	0.9915		0.9926	

Using the above values in Equation 1, we obtained Equation 16 to obtain the distance from the beacons using RSSI values in environment 1. Similarly for environment 2, we obtained Equation 17 to obtain the distance from the beacons using RSSI values. Table IV lists the average RSSI values at different distances from the beacons along with the actual distance, computed distance and the estimation error for environment 1 and 2 respectively.

$$d = 10^{\left(\frac{62.78+RSSI}{-9.116}\right)} \qquad (16)$$

$$d = 10^{\left(\frac{60.95+RSSI}{-12.46}\right)} \qquad (17)$$

Using these models, we evaluated the performance of SRA and SKF and used the current approach of moving averaging of RSSI values as the benchmark. To evaluate the performance of the current approach, we put the beacon in a fixed position and

TABLE IV: An insight into the estimation error of the fitted curve for environment 1 and 2

| Average RSSI		Actual Dist.	Computed Dist. (m)		Error (m)	
e_1	e_2		e_1	e_2	e_1	e_2
-26.8692	-23.1034	0.0001	0.0001	0.009	0	0.0008
-59.9565	-61	1	0.4901	1.0093	0.5099	0.0093
-64.4782	-67.3448	2	1.5357	3.2601	0.4643	1.2601
-67.6086	-67.9655	3	3.3861	3.6563	0.3861	0.6563
-68.4347	-68.5	4	4.1717	4.0359	0.1717	0.0359
-69.437	-69	5	5.3705	4.4266	0.3705	0.5734
-70.5652	-69.9310	6	7.1452	5.2576	1.1452	0.7424
-72.2173	-69.4827	7	10.8457	4.8396	3.8457	2.1604
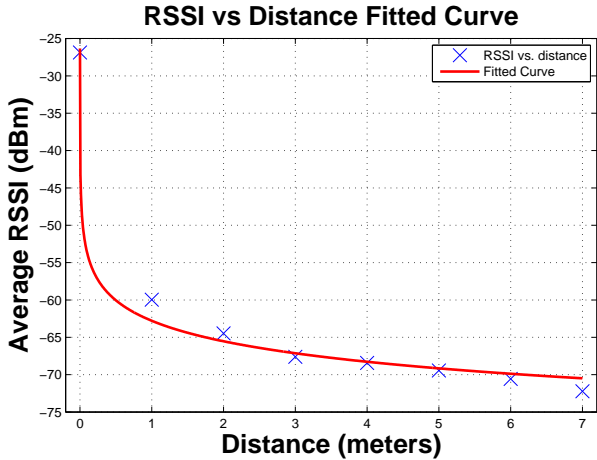
Fig. 3a: Curve fitting for RSSI values at distances from 0 to 7 meters in Environment 1
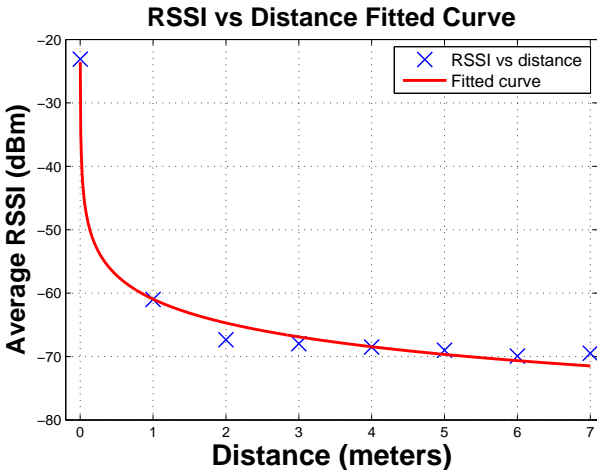

Fig. 3b: Curve fitting for RSSI values at distances from 0 to 7 meters in Environment 2

noted the estimated proximity at different distances. During our experiments, we tested the models only in the 'immediate', 'near' and 'far' regions since the 'unknown' region is of no practical use. We obtained the user's proximity using the three different approaches at a distance of 0, 0.6, 1.8, 2.4, 4.3, and 5.5 meters. These distances, in contrast with Table IV, are chosen such that we have two evaluation points in each proximity zone. We took 20 RSSI samples at each physical location where each RSSI sample consists of the running
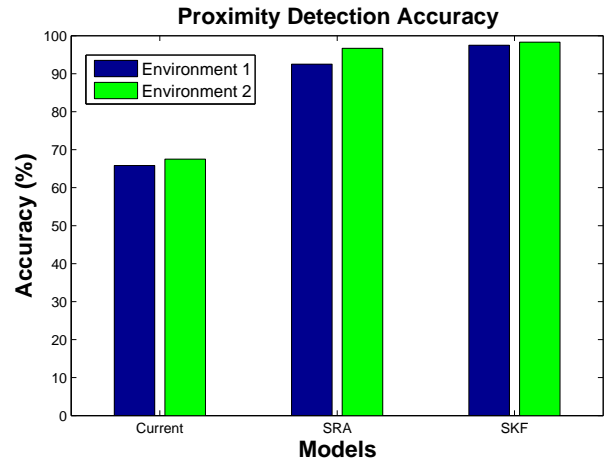

Fig. 4: Average Proximity Detection Accuracy of three algorithms

average of 10 RSSI values (Apple reports RSSI after every 1 second while our gimbal beacon transmitted every 100ms). For every proximity zone, we took 40 samples (20 samples × 2 distances) resulting in 120 (3 zones × 40 samples for each zone) RSSI samples for each approach.

To compare SRA, SKF, and the current approach, we used a three class *confusion matrix* which is one of the popular methods used for evaluating the performance of classification models [12]. The matrix compares the actual classification with the predicted or estimated classification. Table V lists the parameters of the utilized confusion matrix as described by Fawcett in [13] and provides their description in context of our experiment. Table VI contains confusion matrix statistics obtained for SRA and SKF along with the current approach used as the benchmark for both environment 1 and environment 2. Table VII shows the proximity error at different distances in different proximity zones for environment 1 and environment 2 respectively. The SRA and SKF have 0% error in the 'immediate' zone compared to the 47.5% and 37.5% observed using the current approach in environment 1 and environment 2 respectively. Figure 4 shows the average proximity detection accuracy of the two proposed approaches in both environments in comparison with the current approach used as the benchmark. Our algorithms, SRA and SKF, outperform Apple's current approach for proximity detection both in environment 1 and environment 2.

V. DISCUSSION

Figures 3a and 3b show that our curve-fitted path-loss model in environment 1 and 2 respectively can accurately estimate the distance between any user and beacon using the RSSI values. The R^2 value of 0.9915 and 0.9926 highlight the accuracy of the fitted models. These results are also confirmed by Table IV. The average error between the actual distance and estimated distance is 86.14 cm and 67.98 cm for environment 1 and environment 2 respectively.

The results in Table VI highlight the improvements that are attained using SRA and SKF in comparison with the current

TABLE V: Different parameters used in confusion matrix

Parameter	Description
True Positive (TP)	When the user is in zone 'x' and is classified in zone 'x'
True Negative (TN)	When the user is not in zone 'x' and is not classified in zone 'x'
False Positive (FP)	When the user is not in zone 'x' but is classified in zone 'x'
False Negative (FN)	When the user is in zone 'x' but is not classified in zone 'x'
Precision/ Positive Prediction Value (PPV)	The fractions of samples classified in zone 'x'. Mathematically, precision = $\frac{TP_i}{TP_i+FP_i}$ where i is any zone.
Sensitivity/Recall	The fraction of samples correctly classified in zone 'x'. Mathematically, sensitivity = $\frac{TP_i}{TP_i+FN_i}$. The higher the sensitivity, the better will be the algorithm.
Specificity	The fraction of samples correctly classified in any zone other than zone 'x'. The higher the specificity, the better will be the algorithm. Mathematically, specificity = $\frac{TN_i}{TN_i+FP_i}$.
Fall out/False Positive Rate (FPR)	Mathematically, FPR = 1-specificity = $\frac{FP_i}{FP_i+TN_i}$. The lower the FPR value, the better will be the algorithm.
False Negative Rate (FNR)	Mathematically, FNR = $\frac{FN_i}{FN_i+TP_i}$. The lower the FNR value, the better will be the algorithm.
False Discovery Rate (FDR)	A good indicator for conceptualizing the rate of type I error. Mathematically, FDR = 1 - PPV = $\frac{FP_i}{FP_i+TP_i}$. The lower the FDR value, the better will be the algorithm.
Accuracy	The fraction of samples correctly classified. Mathematically, accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$ $\forall$ zones.

TABLE VI: Statistical metrics for the current approach, SRA and SKF in environment 1 (e_1) and environment 2 (e_2)

Metrics	Immediate						Near						Far					
	Current		SRA		SKF		Current		SRA		SKF		Current		SRA		SKF	
	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2
TP	21	25	40	40	40	40	18	16	38	37	39	38	40	40	33	39	38	40
TN	80	80	78	80	79	80	61	65	73	79	78	80	58	56	80	77	80	78
FP	0	0	2	0	1	0	19	15	7	1	2	0	22	24	0	3	0	2
FN	19	15	0	0	0	0	22	24	2	3	1	2	0	0	7	1	2	0
PPV	1	1	0.952	1	0.975	1	0.486	0.516	0.844	0.973	0.951	1	0.645	0.625	1	0.928	1	0.952
Sensitivity	0.525	0.625	1	1	1	1	0.45	0.4	0.95	0.925	0.975	0.95	1	1	0.825	0.975	0.95	1
Specificity	1	1	0.975	1	0.987	1	0.762	0.812	0.912	0.987	0.975	1	0.725	0.7	1	0.962	1	0.975
FPR	0	0	0.025	0	0.012	0	0.237	0.187	0.087	0.012	0.025	0	0.275	0.3	0	0.037	0	0.025
FDR	0	0	0.047	0	0.024	0	0.513	0.483	0.155	0.026	0.048	0	0.354	0.375	0	0.071	0	0.047
FNR	0.475	0.375	0	0	0	0	0.55	0.6	0.05	0.075	0.025	0.05	0	0	0.175	0.025	0.05	0

TABLE VII: Comparison of proximity detection error of SRA and SKF in comparison with current approach in environment 1 (e_1) and environment 2 (e_2)

Actual	Distance (m)	Error at Different Distances (%)						Error in Different Zones (%)					
		Current		SRA		SKF		Current		SRA		SKF	
		e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2	e_1	e_2
Immediate	0	0	0	0	0	0	0	47.5	37.5	0	0	0	0
	0.6	95	75	0	0	0	0						
Near	1.8	10	20	10	0	5	0	55	60	5	7.5	2.5	5.0
	2.4	100	100	0	15	0	10						
Far	4.3	0	0	5	5	0	0	0	0	17.5	2.5	5	0
	5.5	0	0	30	0	10	0						

approach. The higher value of the true positives and true negatives for both SRA and SKF indicates that our algorithms can accurately detect the user's location in any particular zone. The lower value of false positives and false negatives for both SRA and SKF in comparison with the current approach means that our algorithms do not falsely detect or reject a user in a particular zone. Similarly, the sensitivity values for SRA and SKF are higher than the current approach in both the 'immediate' and 'near' zones. The higher sensitivity values mean that both SRA and SKF are more sensitive and able to detect the user in a particular zone as compared to the current approach. SKF performs better than SRA as well. The improved proximity detection of SRA and SKF is also highlighted by the lower FDR, FNR, and FPR values. The high FNR value for the current approach in the 'immediate' and 'near' zone means that the current approach is not suitable for these zones. Furthermore, the high FDR value for the current approach in the 'far' zone means that it is more likely to incorrectly classify the user's proximity in the 'far' zone. This is why the zero error in 'far' zone for current approach, given in Table VII, is not significant as it is due to the inherent flaw in the current approach.

In Table VII, a proximity error of 95% and 75% respectively for the current approach at 0.6 meters (which falls in the immediate region) means that for 19 (environment 1) and 15 (environment 2) out of 20 samples collected at this distance, the current approach was not able to accurately classify them into the 'immediate' zone. The average error of 47.5% and 37.5% for the current approach in the 'immediate' region of environment 1 and 2 respectively, means that for the 40 samples collected in this region, only 21 samples in environment 1 and 25 samples in environment 2 were correctly classified.

This shows the current approach is not favorable for PBS. SRA and SKF, on the other hand, have 100% accuracy in the 'immediate' zone in both environments. In the 'near' region, error for the current approach is 55% and 60% for environment 1 and 2 respectively while it is 5% and 2.5% for SRA and SKF respectively in environment 1 and 7.5% and 5% in environment 2. This means that out of 40 samples, the current approach accurately classified only 18 samples and 16 samples in environment 1 and environment 2 respectively, which is far less than the 38 and 39 samples accurately classified by SRA and SKF respectively in environment 1 and 37 and 38 samples accurately classified by SRA and SKF respectively in environment 2.

Based on the results in Tables VI, VII, and Figure 4, our SRA and SKF approaches outperform the current approach in the 'immediate' and 'near' zones, which is primarily used for triggering PBS in most of the beacon-based applications. The current approach, due to the inherent flaw of not taking the environmental factors into account, classifies most of the samples in the 'far' region, which is why there is no detection error in the 'far' zone for the current approach (the high FDR value in Table VI proves this fact). This is also the cause of higher detection error in the 'immediate' and 'near' regions for the current approach. As shown in Figure 4, the current approach achieved a proximity detection accuracy of 65.83% and 67.5% in environment 1 and environment 2 respectively. SRA achieved 92.5% and 96.6% proximity detection accuracy which is 26.7% and 29.1% improvement over the current approach in environment 1 and 2 respectively. SKF achieved a proximity detection accuracy of 97.5% and 98.3%, which improves proximity detection accuracy by almost 31.6% and 30.8% over the current approach in environment 1 and environment 2 respectively. The figure indicates that out of 120 samples, only 79 and 81 were properly classified by the current approach in environment 1 and environment 2 respectively, which is much lower than the 111 and 116 correctly classified by our approach SRA in environment 1 and environment 2 respectively, and 117 and 118 samples correctly classified by our approach SKF in environment 1 and environment 2 respectively.

The increased accuracy of both SRA and SKF in two different environments makes it a viable alternative to the current approach. The improved performance of current approach, SRA and SKF in environment 2, when compared with environment 1, is due to less noise in environment 2 as evident from Figure 3b and lower values of C in Table III. Based on the obtained results, it is evident that use of the proposed algorithms SRA and SKF improves the proximity detection accuracy of iBeacons and enhances their performance for providing reliable proximity based services.

VI. CONCLUSIONS

Proximity-based services can be leveraged in different locations including airports, retail stores, hospitals and stadiums etc. However, the current technologies cannot fulfill the accuracy, energy consumption, range, cost and availability requirements for PBS. iBeacon, the industry standard for PBS, currently lack the accuracy to be utilized for efficient and accurate PBS. In this paper, we proposed two server-based algorithms that improved the proximity detection accuracy of an iBeacon based proximity solution by handling the inherent limitations of the current approach utilized by Apple for proximity classification. We used the current approach as a benchmark that resulted in proximity detection accuracy of 65.83% and 67.5% in environment 1 and environment 2 respectively, while the Server-side Running Average (SRA) achieved 92.5% and 96.6% proximity detection accuracy in environment 1 and environment 2 respectively, which is 26.7% and 29.1% improvement over the current approach. The Server-side Kalman Filter (SKF) achieved proximity accuracy of 97.5% and 98.3%, that is 31.67% and 30.8% improvement over the current approach in environment 1 and environment 2 respectively.

REFERENCES

[1] F. Zafari, I. Papapanagiotou, and K. Christidis, "Microlocation for internet-of-things-equipped smart buildings," *IEEE Internet of Things Journal*, vol. 3, pp. 96–112, Feb 2016.

[2] C. N. Klokmose, M. Korn, and H. Blunck, "WiFi proximity detection in mobile web applications," in *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems*, pp. 123–128, ACM, 2014.

[3] M. Bolic, M. Rostamian, and P. M. Djuric, "Proximity detection with RFID: A step toward the internet of things," *IEEE Pervasive Computing*, no. 2, pp. 70–76, 2015.

[4] Bluevibe, "Proximity Marketing-made interactive." http://www.bluevibe.net/. [Online; accessed 31-Mar-2016].

[5] R. Kallas, "Google is going after Apple." http://unacast.com/googles-eddystone-is-killing-apples-ibeacon/. [Online; accessed 01-Apr-2016].

[6] S. Kumar, S. Gil, D. Katabi, and D. Rus, "Accurate indoor localization with zero start-up cost," in *Proceedings of the 20th annual international conference on Mobile computing and networking*, pp. 483–494, ACM, 2014.

[7] A. Ghose, C. Bhaumik, and T. Chakravarty, "Blueeye: A system for proximity detection using bluetooth on mobile phones," in *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*, pp. 1135–1142, ACM, 2013.

[8] F. Zafari and I. Papapanagiotou, "Enhancing ibeacon based micro-location with particle filtering," in *2015 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, Dec 2015.

[9] P. Kumar, L. Reddy, and S. Varma, "Distance measurement and error estimation scheme for RSSI based localization in wireless sensor networks," in *Fifth IEEE Conference on Wireless Communication and Sensor Networks (WCSN)*, pp. 1–4, IEEE, 2009.

[10] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[11] I. Guvenc, C. Abdallah, R. Jordan, and O. Dedoglu, "Enhancements to RSS based indoor tracking systems using kalman filters," 2003.

[12] V. M. Patro and M. R. Patra, "A novel approach to compute confusion matrix for classification of n-class attributes with feature selection," *Transactions on Machine Learning and Artificial Intelligence*, vol. 3, no. 2, p. 52, 2015.

[13] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.