Improved Distance Estimation with BLE Beacon using Kalman Filter and SVM

Ching Hong Lam, Pai Chet Ng and James She
Department of Electronic and Computer Engineering
HKUST-NIE Social Media Lab, Hong Kong University of Science and Technology
Email: {chlamaq, pcng, eejames}@ust.hk

***Abstract*—Lately, Bluetooth Low Energy (BLE) beacon has attracted a lot of interests for its capabilities in enhancing the interaction between smart things in the Internet of Things (IoT) ecosystem via proximity approach. Even though Proximity sensing is capable of delivering a correct interaction, it might have a problem for explicit interaction when exact distance estimation is required. Considering those interactive applications which are distance-dependent, this paper proposed an optimized support vector machine (O-SVM) on the cloud for distance estimation and a Kalman filter (KF) on the edge to obtain a near true RSS value from a list of RSS measurements. Four benchmark functions (i.e., two from Industries and two Machine Learning Techniques) have been used for performance evaluation. Simulation with real signal samples was conducted to verify the performance of our proposed algorithm. Besides examining the performance gain of our proposed solution over the four benchmark functions, we also implemented the proposed solution on a smartphone for practical testing to demonstrate its feasibility. The proposed solution not only outperforms the rest with significant performance gain, i.e., $> 50\%$ error reduction compared to the benchmark functions. Furthermore, practical implementation verified that our proposed approach is able to return the estimate distance in less than $1s$, such real-time response is desirable for many delay-sensitive applications.**

***Index Terms*—Distance estimation, BLE Beacon, Kalman Filter, Support Vector Machine, Internet of things, interactivity**

I. Introduction

In the Internet of things (IoT) ecosystem, as the things getting smarter and smarter with diverse kinds of embedded sensors and communication capabilities, an accurate interaction becomes the key element towards a better IoT ecosystem, making whole the smartness of the things besides sensing and remote controlling [1] [2] [3]. One vital factor for the smart things to achieve accurate interaction lies in their ability in estimating their distance to other things. Fig. 1 depicts an interactive application that consists of two flying drones: one is mounted with a projector and another one carrying the display [4]. Obviously, these two drones need to estimate their distance to each other to ensure a clear content being displayed on the screen. Unmanned vehicles, on the other hand, can be very dangerous to the pedestrian and other vehicles on the road if they are unable to derive an accurate distance estimation; having said that, good distance estimation is, inevitably, crucial for many interactive applications. Specifically, in reshaping the current IoT ecosystem from only sensing, monitoring and controlling to a smarter interaction between living things as well as non-living things.

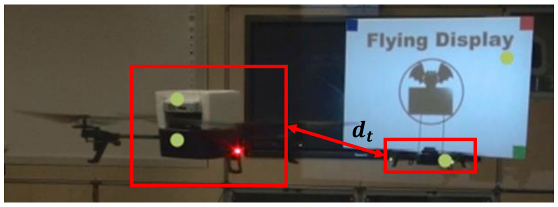

Fig. 1. Interaction between two smart things (i.e., the flying projector and flying display) [4].

In recent years, the ubiquity of wireless communications embedded in current smart things has attracted a vast interest in exploiting Radio Signal Strength (RSS) for distance estimation [5] [6]. RSS-based distance estimation is based on the ideal assumption of a path loss model which states that RSS decays as the distance between two smart things increases, provided that these smart things are within the line-of-sight [7]. While RSS is widely available and can be simply measured by most wireless devices, RSS is known by its unreliability for distance estimation especially in an indoor environment where the signals are subject to serious fluctuation in consequence to shadowing and multipath fading [8] [9]. Furthermore, it might hard to receive the line-of-sight signals due to the presence of various obstacles inside the building. Despite various challenges imposed by the uncertainties in RSS measurements, RSS from RFID [10] and WiFi [11] are, still a widely exploited tool for the distance-dependent applications.

Apart from RFID and WiFi, Bluetooth Low Energy (BLE) beacon has emerged as a promising alternative for IoT-related development [12]. The popularity of BLE beacon is growing exponentially with the introduction of Apple's iBeacon[1] in 2013. Besides being a low power and low-cost device [13], the portability of the beacon which can be associated with any object anytime allow greater flexibility for interaction design as compared to the WiFi infrastructure in which their access point is mostly fixed at a certain location [14]. We chose BLE beacon over RFID simply because of the accessibility of Bluetooth signals in most of the off-the-shelf smart devices (e.g., smartphone, smartwatch, etc.), which enables rapid application development on top of the existing platform [15]. However, Bluetooth signals [16] [17], similar any RF signals,

[1]"iBeacon for Developers", "https://developer.apple.com/ibeacon/"

are subject to severe fluctuation issues as described above. In other words, distance estimation based on beacon also suffers severe performance degradation in the indoor environments as experienced by other RF signals.

In fact, there are a lot of software development kits available freely for the developers to develop beacon-related applications. For example, the *CoreLocation* framework from Apple includes a distance estimation function which can return the estimated distance in $1s$. Radius Network[2], on the other hand, shares their source code openly with their developers. However, both approaches fail to return precise distance estimation, especially when the distance increases, as shown in Fig. 2. In view of the needs of better distance estimation to cater the interaction between the things in the IoT ecosystem, this paper proposes a novel solution to increase the distance estimation performance at both short- and long-range distances. Our novel solution encompasses a Kalman filter (KF) implemented on the edge and an Optimized Support Vector Machine (O-SVM) regression on the cloud. In contrast to the previous work which implemented the KF on the server [18], our approach ensures the real-time performance by transmitting only the filtered RSS (i.e., the near true RSS value estimated from a list of noisy RSS measurements) to the cloud to obtain the estimated distance rather than bombarding the cloud with a bunch of raw RSS measurements. The proposed solution is abbreviated as O-SVM-KF and our contributions are summarized as follows:

- Distributing Kalman filter to the edge and distance computation to the cloud to avoid heavy computation burden on either side, at the same time minimize the data to be exchanged between the edge and the cloud.
- Using four benchmark functions for performance evaluation. Two benchmark functions are widely adopted by industries for iOS and Android application development, respectively. Another two benchmark functions are the machine learning algorithms.
- Verifying the performance through simulation with real RSS samples and demonstrating its feasibility through a real-world implementation over the smartphone.

The rest of the paper is organized as follows. Section II formulates the problem and describes the proposed solution. Section III presents the experiment setup for RSS acquisition. A simulation is conducted to examine the performance of our proposed solution with the collected RSS samples. Section IV describes a simple implementation and discusses the performance of our proposed solution in different environments. Section V concludes the paper.

II. Proposed Solution

This section first provides a mathematical formulation related to RSS-based distance estimation, then follows with a detailed description of our proposed solution. For consistency, we used the bold lowercase letter to represent vector and bold uppercase letter to denote matrix. The related notation used throughout the paper is summarized in Table I.

[2]"AltBeacon: The Open and Interoperable Proximity Beacon", "http://altbeacon.org/"

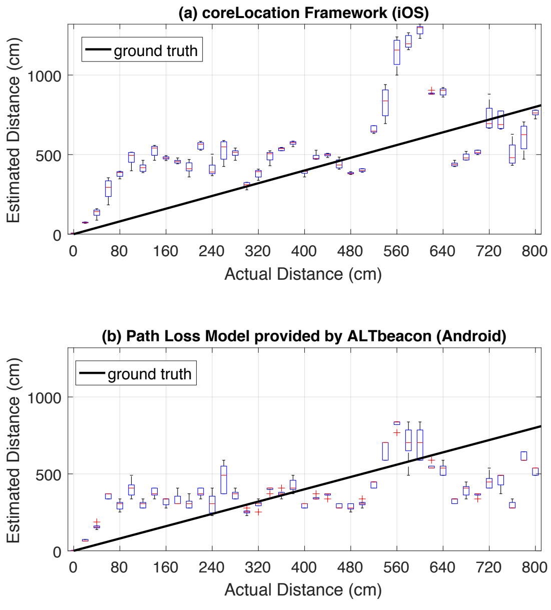

Fig. 2. Beacon-based distance estimation using (a) *CoreLocation* framework and (b) Path Loss Model by ALTBeacon.

A. Problem Formulation

As shown in Fig. 3, the distance estimation starts will signal acquisition on the edge. The edge can be of any smart devices (e.g., smartphone, smart wearable, embedded microcontroller) as long as they are Bluetooth compatible and possess minimum computing capabilities [19]. Assume that the list of RSS measured by the edge over a particular scanning time t_s is represented with a vector $\mathbf{r} \in \mathbb{R}^n$,

In general, it is always desired to estimate a true RSS value from a list of noisy measurements before proceeding with distance estimation. In this paper, we adopt KF to smooth the RSS measurement. KF has been used by many other RSS-related applications. However, most of them applied the KF on the server which might incur heavy traffic load to the server when multiple smart things transmit their collected RSS samples to the server at the same time. So instead of transmitting all RSS samples, we apply KF on the edge to obtain a near true RSS value by filtering the list of noisy RSS measurements, the processes involved with KF are further discussed in section II-B. Note that, the noise filtering is not limited to only KF, other techniques such as running average can be applied too.

Now, given the received filtered RSS value $\tilde{r}$, the distance computation over the cloud can be expressed as follows:

$$d_t = f(\tilde{r}) \tag{1}$$

where d_t is the distance estimated at time $t > t_s$. $f(\cdot)$ is an estimated function based on the trained model. The trained model can be the path loss model or any model trained through machine learning methods. In this paper, we use support vector machine (SVM) to train the model, and further optimize the

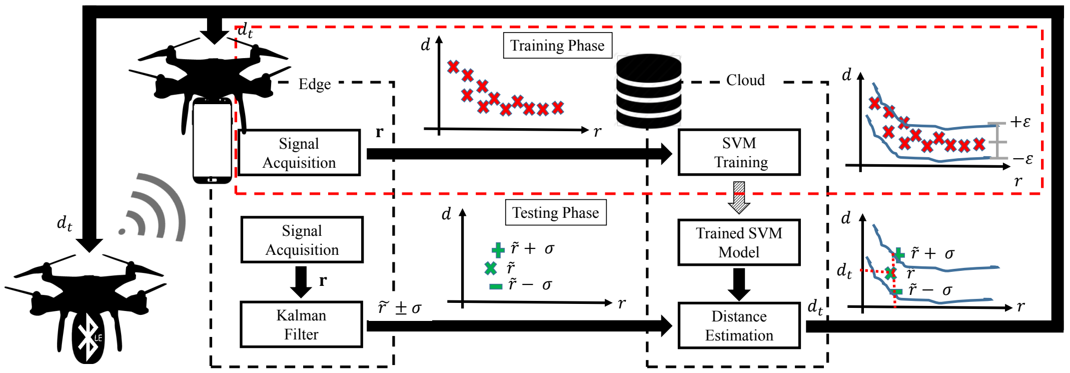

Fig. 3. The proposed solution distributes the RSS processing to (a) edge for signal acquisition and Kalman filter and (b) cloud for distance estimation.

TABLE I
RELATED VARIABLES AND THEIR NOTATIONS

Variable's definition	Notation
scanning duration	t_s
RSS vector	$\mathbf{r}$
filtered rss value	$\tilde{r}$
sampling interval	τ_0
distance at time t	d_t
measurement residual	$\mathbf{z}$
measurement noise covariance	Σ
measurement noise	σ
transformation matrix	$\mathbf{H}$
error covariance	$\mathcal{E}$
Kalman gain	K

trained model to achieve better performance. The description of the SVM training process is provided in section II-C.

B. Kalman Filter on the Edge

This paper adopts KF to obtain a filtered RSS value from a list of RSS samples collected over time t_s. KF is popular for its light computational requirement which enables us to implement the filter directly on the edge [20] [21]. Furthermore, KF is capable of inferring the true value of RSS from a set of noisy and random observation, provided the observation exhibits a Gaussian distribution [22]. Given the RSS vector $\mathbf{r}$, KF iterates through each element in $\mathbf{r}$ to obtain the filtered RSS value $\tilde{r}$. In general, the RSS measurements filtering via KF can be modeled as follows:

$$\mathbf{r}_\tau = \tilde{r}_\tau \mathbf{H}_\tau \pm \sigma_\tau \tag{2}$$

where the transformation matrix $\mathbf{H}_\tau$ maps $\tilde{r}_\tau$ to the observable domain, and σ_τ is the measurement noise vector. τ is the time where the RSS is being sampled. Note that $\tau < t_s$ and multiple RSS values might be measured at time τ due to the reflection from multipath, and $\mathbf{r}_\tau \subseteq \mathbf{r}$. The objective here is to infer $\tilde{r}_{\tau+\tau_0}$ from $\tilde{r}_\tau$, that is,

$$\tilde{r}_{\tau+\tau_0} = \Phi \tilde{r}_\tau \pm \sigma_\tau \tag{3}$$

where Φ is the transition matrix from $\tau + \tau_0$ and τ_0 is the sampling interval.

The noise filtering process continues until $\tilde{r}$ converges or until reaching the maximum scanning time t_s. At each iteration, KF will keep updating their current estimate and the error covariance. Here, we list down the three equations required by KF during the updating phase, the derivation of these equations can be found in [22].

- Update current estimate: $\tilde{r}^+_\tau = \tilde{r}^-_\tau + K_\tau(\mathbf{z}_\tau - \mathbf{H}_\tau \tilde{r}^-_\tau)$
- Update error covariance: $\mathcal{E}^+_\tau = (I - K_\tau \mathbf{H}_\tau)\mathcal{E}^-_\tau$
- Kalman gain: $K_\tau = \mathcal{E}^-_\tau \mathbf{H}^T_\tau (\Sigma_\tau + \mathbf{H}_\tau \mathcal{E}^-_\tau \mathbf{H}^T_\tau)^{-1}$

Refer to Table I for the definition of the related notations.

Once $\tilde{r}$ is obtained, it will be sent to the cloud for distance computation. According to to [23], 10 RSS samples per second are sufficient to obtain a true average RSS value through running average approach. In fact, with KF, we observed that $\tilde{r}$ converges after 3 to 4 iterations and before reaching the maximum scanning duration t_s. In other words, we only need less than 10 RSS samples to obtain the filtered RSS value in less than $1s$. Such fast convergence rate ensures a near real-time distance estimation for interactive applications.

C. Distance Computation with SVM over the Cloud

The received $\tilde{r}$ is used by the trained SVM model for distance estimation. The SVM is a common machine learning technique which uses a kernel function to train the model. In this paper, a non-linear kernel function - Gaussian kernel is used, this Gaussian kernel $\mathcal{G}$ can be expressed as follows:

$$\mathcal{G} = e^{-\|\mathbf{r}\|^2} \tag{4}$$

The corresponding cost function is:

$$\begin{aligned} C(\alpha) = &\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\mathcal{G} \\ &+ \epsilon \sum_{i=1}^{n}(\alpha_i + \alpha_i^*) - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) \\ \text{subject to} \quad &\sum_{i=1}^{k}(\alpha_k - \alpha_k^*) = 0 \\ &\forall n : 0 \leq \alpha_n \leq C \\ &\forall n : 0 \leq \alpha_n^* \leq C \end{aligned} \tag{5}$$

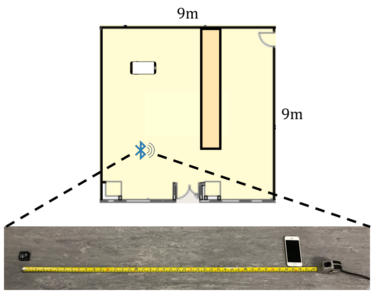

Fig. 4. Experimental setup

where C is a constant that controls the penalty imposed on the RSS samples that lie outside the margin ϵ. The above optimization formula was derived via the Lagrange dual formulation, which is clearly described in [24]. To solve the above optimization problem, we performed a series of two-point optimization via the sequential minimal optimization method [25], which is provided within the Matlab toolbox.

With the trained coefficient α, the optimized SVM (O-SVM) model can be obtained. The resultant O-SVM model for distance estimation can be expressed as follows:

$$d_t = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) e^{-\|\mathbf{r}\|^2} + \gamma \tag{6}$$

Once d_t is obtained, the cloud will send the estimated distance back to the edge, as shown in Fig. 1. Note that the data size to be exchanged between the server and the edge is just a few bytes, which mean the time taken for the data exchanging and distance computation over the cloud is less than $100ms$. Given the processing on the edge is also less than $1s$, the total time taken to obtain the final estimated distance is less than $1s$, which is desirable for most interactive applications. Section IV verified the real-time performance of our proposed approach through a practical implementation.

III. Experiment and Results

This section describes the setup of the experiment for signal acquisition, the collected samples are used for latter simulation. Section III-B lists the benchmark functions used in the simulation. The simulation is described in section III-C, and the simulation results are presented in section III-D.

A. Experimental Setup

The experiment was conducted in a $9{\times}9m$ laboratory room with a beacon placed on the floor, as indicated in Fig. 4. This beacon is configured to have an advertising interval of $100ms$ and a transmit power of $0dBm$. A customized RSS measuring app is installed in an iPhone for signals acquisition at every distance. We measured and collected the RSS samples from 0 to $8m$ with $0.2m$ increment each step. The collected samples were stored in the phone and exported as .CSV file for latter simulation. Besides testing the RSS samples with our proposed solution, we also adopted four popular approaches as benchmark functions.

B. Benchmark Functions

Among the four adopted approaches for benchmarking, two are widely used methods for iOS and Android development, another two is based on the machine learning techniques.

1) CoreLocation Framework (CL): Together with their proprietary iBeacon, Apple provides their developers the *Core-Location* framework[3] for app development. The *CoreLocation* framework provides a method called "accuracy" to return the estimated distance in $1s$. More specifically, the method applies the running average over the collected RSS samples in $1s$ and return the estimated distance. However, Apple did not explicitly disclose their method in estimating the distance.

2) Open ALTBeacon Standard (ALTPLM): As opposed to *CoreLocation* framework, ALTBeacon provides a set of open source libraries for Android app development. The path loss model used for distance estimation is clearly described in their documentation. The path loss model provided by ALTBeacon can be expressed as follows:

$$d_t = \alpha(\tilde{r}/r_0)^{\beta} + \gamma \tag{7}$$

where the corresponding coefficients α, β and γ are provided explicitly, and can be found in their Github page[4]. r_0 is the reference RSS value at distance equals to $1m$ and it is set to $-66dBm$ as according to their documentation.

3) Linear Regression (LRM): Since the trained model returns the negative distance, which is meaningless in our case as we only consider scalar distance estimation without the directional information. Hence, we further modified the trained model with a unit step function, that is:

$$d_t = (\alpha\tilde{r} + \beta)u(t) \tag{8}$$

where coefficient α is the slope of the linear model and β is the interception.

4) Non-linear Regression (PLRM): The non-linear regression uses the same path loss equation described in Eq. (7) to obtain the regression model. However, for the parameter d_0, instead of using the one provided by ALTBeacon, we computed the average value of RSS at $1m$ distance and substitute the computed value as d_0.

C. Simulation

The simulation is conducted with a set of real RSS samples collected during the experimental phase. We used the Statistics and Machine Learning Toolbox available in Matlab to train the LRM, PLRM and also the O-SVM model. The RSS samples collected at each distance is divided into two portions: 70% are for training, and 30% for testing. The absolute distance error is used as the performance measure. This error measure is a function of distance which can be expressed as

$$e(d) = \|d - d_t\| \tag{9}$$

[3]"CLBeacon", "https://developer.apple.com/documentation/corelocation/clbeacon"

[4]"AltBeacon: The Open and Interoperable Proximity Beacon specification", "https://github.com/AltBeacon/android-beacon-library/blob/master/src/main/resources/model-distance-calculations.json"

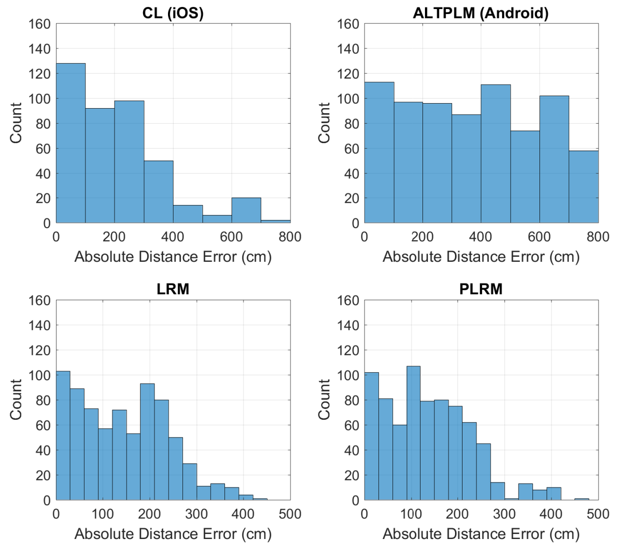

Fig. 5. Histogram is used to capture the absolute distance errors return by the four benchmark functions.

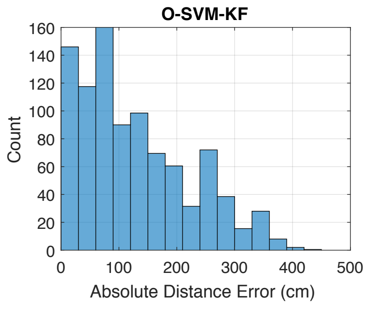

Fig. 6. The absolute distance errors returned by the proposed solution.

where d is the actual distance and d_t is the estimated distance at time t. According to Eq. (1), the estimated distance d_t is dependent on the filtered RSS value $\tilde{r}$ and the applied distance estimation model. The performance achieved by each model is presented in the next section.

D. Results

The absolute distance errors returned by the four benchmark functions are illustrated with error histogram, as shown in Fig. 5. The maximum error class by both CL and ALTPLM can go up to $700-800cm$, in particular, ALTPLM produces a large number of errors in the $700-800cm$ error class. On the other hand, better results are achieved by well-trained models (e.g., LRM and PLRM), in which the maximum error class is almost half than CL and ALTPLM. As for our proposed solution, the maximum error class is at $400-450cm$, as shown in Fig. 6. Furthermore, the absolute distance error produces by O-SVM-KF is concentrated at the lower error classes.

Fig. 7 summarizes the absolute distance errors in the format of cumulative distribution function. Obviously, O-SVM-KF

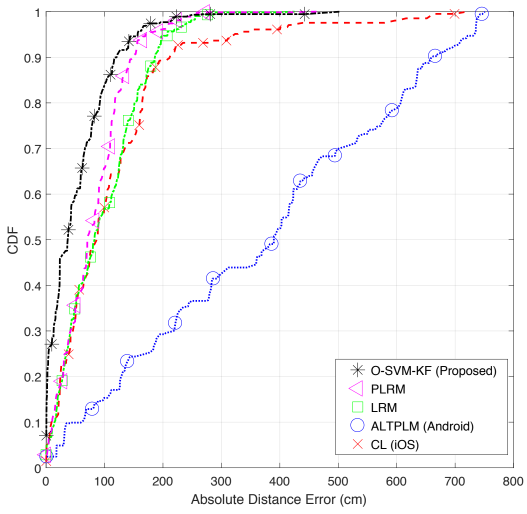

Fig. 7. The cumulative distribution function of absolute distance errors.

outperforms the rest with less than $1m$ distance error 80% of the time. One possible reason that ALTPLM fails to produce good performance might be the coefficients provided is environmental-dependent, which means those coefficients might able to produce a good distance estimation in their tested environment but the performance might drop when the environment changes. Hence, it is valid to assume that the good performance of PLRM (used similar path loss model as ALTPLM) is simply because the coefficients were trained in the same environment. To verify the feasibility of O-SVM-KF for different environments, we further implement the model on a smartphone for practical distance testing.

IV. Implementation and Discussion

The proposed solution was implemented on an iPhone for practical testing, Fig. 8 shows the App interface and the testing. Since *CoreLocation* only return the running averaged of beacons' RSS values each second, we used *CoreBluetooth* to measure the raw RSS values. With the raw RSS samples, we applied KF on the phone to obtain a near true RSS value from a list of noisy measurements. The filtered RSS is then be sent to the cloud using TCP/IP socket. Upon receiving the filtered RSS, the cloud retrieves the trained SVM model to estimate the distance and return to the mobile phone through the same socket connection.

The implemented App was tested in another laboratory room with multiple beacons present at the same time. The estimated distance is obtained almost immediately on the App, which is indeed desirable for most interactive applications which are delay-sensitive. Furthermore, since only a filtered RSS and an estimated distance are exchanged between the cloud and the edge, the network traffic is greatly reduced as compared to the previous work proposed by [18]. However, we also notice estimation deviation from the results obtained in simulation. One possible reason might be the environmental

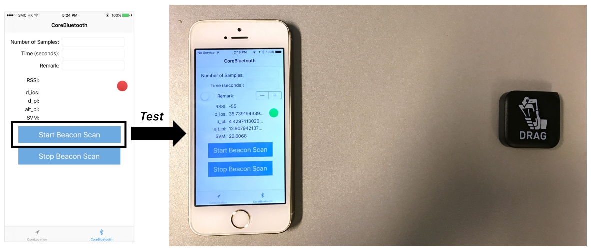

Fig. 8. The feasibility of the proposed approach is demonstrated with a practical implementation.

factor that causes different signal variation in a different room. In other words, when the signal is blocked by objects [26], a drop in estimation performance is observed compared to the environment where the smartphone and beacon are within the line-of-sight. Despite the estimation variation caused by different environmental factors, the variation of our proposed solution is still relatively lower compared to the rest.

V. Conclusion

Improved distance estimation is definitely beneficial in enhancing the smart things' interaction in the IoT ecosystem. While RSS-based distance estimation is cost-effective, its signals suffer a severe fluctuation issue, especially in the indoor environment. The fluctuation issue affects the reliability of good estimation. In this paper, we introduced a novel solution by implementing a Kalman filter on the edge to deal with the noisy RSS measurements and an optimized SVM on the cloud for distance estimation. The performance of proposed solution is verified through both simulation and real-world implementation. The results indicate the superiority of our proposed solution with double error reduction in distance estimation. Furthermore, the real-world implementation proves the real-time performance of our proposed solution in which the distance is returned almost immediately. In other words, the delay is minimal and could not be detected with the naked human eye. Such a real-time performance is desirable for most interactive applications which are delay-sensitive.

Acknowledgment

This work was supported by HKUST-NIE Social Media Laboratory.

References

[1] P. C. Ng, J. She, and S. Park, "Notify-and-interact: A beacon-smartphone interaction for user engagement in galleries," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, July 2017, pp. 1069–1074.

[2] Y. Xiao, Z. Xiong, D. Niyato, Z. Han, and L. A. DaSilva, "Full-duplex machine-to-machine communication for wireless-powered internet-of-things," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[3] P. C. Ng, J. She, K. E. Jeon, and M. Baldauf, "When smart devices interact with pervasive screens: A survey," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 13, no. 4, pp. 55:1–55:23, Aug. 2017. [Online]. Available: http://doi.acm.org/10.1145/3115933

[4] H. Nozaki, "Flying display: a movable display pairing projector and screen in the air," in *CHI'14 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2014, pp. 909–914.

[5] Y. Liu, D. Zhang, X. Guo, M. Gao, Z. Ming, L. Yang, and L. M. Ni, "Rss-based ranging by leveraging frequency diversity to distinguish the multiple radio paths," *IEEE Transactions on Mobile Computing*, vol. 16, no. 4, pp. 1121–1135, April 2017.

[6] F. Yin, Y. Zhao, F. Gunnarsson, and F. Gustafsson, "Received-signal-strength threshold optimization using gaussian processes," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 2164–2177, April 2017.

[7] F. Zafari, I. Papapanagiotou, and K. Christidis, "Microlocation for internet-of-things-equipped smart buildings," *IEEE Internet of Things Journal*, vol. 3, no. 1, pp. 96–112, Feb 2016.

[8] M. Ayadi and A. B. Zineb, "Body shadowing and furniture effects for accuracy improvement of indoor wave propagation models," *IEEE Transactions on Wireless Communications*, vol. 13, no. 11, pp. 5999–6006, Nov 2014.

[9] J. Yang, X. Wang, S. I. Park, and H. M. Kim, "Optimal direct path detection for positioning with communication signals in indoor environments," in *2012 IEEE International Conference on Communications (ICC)*, June 2012, pp. 4798–4802.

[10] K. Deepika and J. Usha, "Design development of location identification using rfid with wifi positioning systems," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2017, pp. 488–493.

[11] Y. Agata, J. Hong, and T. Ohtsuki, "Room-level proximity detection based on rss of dual-band wi-fi signals," in *2016 IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–6.

[12] P. C. Ng, L. Zhu, J. She, R. Ran, and S. Park, "Beacon-based proximity detection using compressive sensing for sparse deployment," in *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, June 2017, pp. 1–6.

[13] K. E. Jeon, J. She, P. Soonsawad, and P. C. Ng, "Ble beacons for internet of things applications: Survey, challenges and opportunities," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2018.

[14] G. Solmaz and F. J. Wu, "Together or alone: Detecting group mobility with wireless fingerprints," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–7.

[15] Y. Guo, Y. Sun, T. Y. Wu, M. S. Obaidat, and W. T. Lee, "Monitoring the status of ibeacons with crowd sensing," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–7.

[16] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," *Sensors*, vol. 12, no. 9, pp. 11 734–11 753, 2012.

[17] P. C. Ng, J. She, and S. Park, "High resolution beacon-based proximity detection for dense deployment," *IEEE Transactions on Mobile Computing*, vol. PP, no. 99, pp. 1–1, 2017.

[18] F. Zafari, I. Papapanagiotou, M. Devetsikiotis, and T. J. Hacker, "Enhancing the accuracy of ibeacons for indoor proximity-based services," in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–7.

[19] X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, December 2016.

[20] A. Abusara and M. Hassan, "Error reduction in distance estimation of rss propagation models using kalman filters," in *2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO)*, May 2015, pp. 1–5.

[21] H. Ma and K. Wang, "Fusion of rss and phase shift using the kalman filter for rfid tracking," *IEEE Sensors Journal*, vol. 17, no. 11, pp. 3551–3558, June 2017.

[22] R. Faragher, "Understanding the basis of the kalman filter via a simple and intuitive derivation [lecture notes]," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 128–132, Sept 2012.

[23] R. Faragher and R. Harle, "An analysis of the accuracy of bluetooth low energy for indoor positioning applications," vol. 1, pp. 201–210, 01 2014.

[24] P. Bouboulis, S. Theodoridis, C. Mavroforakis, and L. Evaggelatou-Dalla, "Complex support vector machines for regression and quaternary classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 6, pp. 1260–1274, June 2015.

[25] "A study on smo-type decomposition methods for support vector machines," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 893–908, July 2006.

[26] M. Kohne and J. Sieck, "Location-based services with ibeacon technology," in *2014 2nd International Conference on Artificial Intelligence, Modelling and Simulation*, Nov 2014, pp. 315–321.