IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 2, 2019, accepted April 19, 2019, date of publication April 30, 2019, date of current version May 13, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2913910

SpringLoc: A Device-Free Localization Technique for Indoor Positioning and Tracking Using Adaptive RSSI Spring Relaxation

DANIEL KONINGS[1], (Member, IEEE), FAKHRUL ALAM[1], (Senior Member, IEEE), FRAZER NOBLE[1], AND EDMUND M-K. LAI[2], (Senior Member, IEEE)

[1]Department of Mechanical & Electronic Engineering, School of Food and Advanced Technology, Massey University, Auckland 0632, New Zealand
[2]School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand

Corresponding author: Daniel Konings (d.konings@massey.ac.nz)

This work was supported in part by Massey University Doctoral Scholarships and in part by Massey University Research Fund (MURF).

ABSTRACT Device-free localization (DFL) algorithms using the received signal strength indicator (RSSI) metrics have become a popular research focus in recent years as they allow for location-based service using commercial-off-the-shelf (COTS) wireless equipment. However, most existing DFL approaches have limited applicability in realistic smart home environments as they typically require extensive offline calibration, large node densities, or use technology that is not readily available in commercial smart homes. In this paper, we introduce SpringLoc and a DFL algorithm that relies on simple parameter tuning and does not require offline measurements. It localizes and tracks an entity using an adaptive spring relaxation approach. The anchor points of the artificial springs are placed in regions containing the links that are affected by the entity. The affected links are determined by comparing the kernel-based histogram distance of successive RSSI values. SpringLoc is benchmarked against existing algorithms in two diverse and realistic environments, showing significant improvement over the state-of-the-art, especially in situations with low-node deployment density.

INDEX TERMS Device-free localization (DFL), histogram distance, indoor positioning systems (IPS), smart homes, spring-relaxation.

I. INTRODUCTION

Device-free Localization (DFL) systems that utilize the Received Signal Strength Indicator (RSSI) metric can track untagged subjects, unlike traditional Device-based/Active Tracking approaches. They can facilitate location-based services such as lighting/music control and intruder detection, based on human presence alone. However, since the tracked entity is untagged, it can be hard to uniquely identify each entity when multiple targets are present. Improved localization accuracy can potentially lead to more accurate entity identification. The purpose of this paper is to provide an improved algorithm for DFL that can be implemented in practical scenarios e.g. smart homes without requiring the deployment of significant additional infrastructure.

Existing indoor DFL systems have three main shortcomings with respect to practical implementation. The first one is that they require many sensors within the target environment. This can lead to high implementation cost due to the large number of sensors required, and the requirement of easily accessible power across the whole environment [1]–[4]. The second shortcoming is that DFL implementations also require a large number of offline measurements to calibrate the system to the target environment [5]–[10]. This restricts their usability, as standard end users cannot be expected to install new power sockets and undertake excessive calibration procedures to facilitate localization within their home. The third shortcoming is that recent attempts at DFL solutions use hardware that is inaccessible to standard end users. For example, Channel State Information (CSI) based DFL has been shown to be quite accurate. However, CSI is not available on the majority of wireless platforms e.g. Zigbee or Bluetooth. Moreover, even with Wi-Fi, CSI is only available on two outdated modules with bespoke modified drivers [11], [12]. Other leading approaches makes use of Frequency Modulated Carrier Wave (FMCW) signals using a software defined

The associate editor coordinating the review of this manuscript and approving it for publication was Byung-Seo Kim.

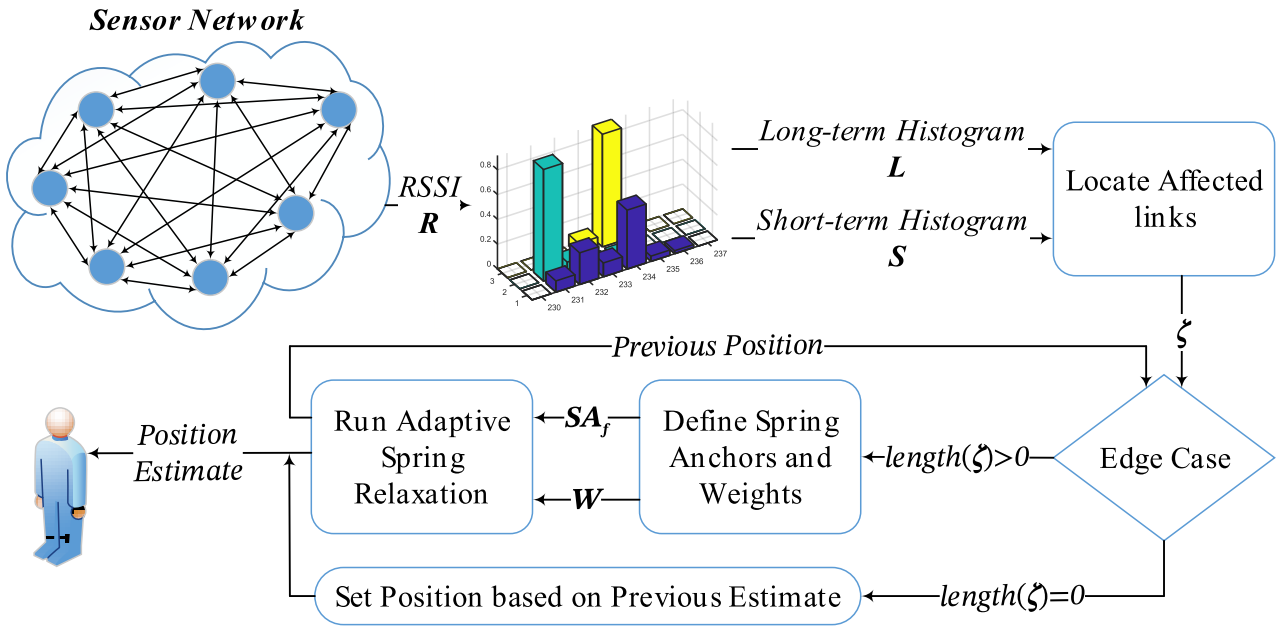

FIGURE 1. Springloc algorithm overview.

radio platform [13], [14]. This limits usability as rather than using pre-existing wireless infrastructure or widely available Commercial Off-The-Shelf (COTS) equipment, these solutions require the deployment of custom designed additional wireless infrastructure for the sole purpose of localization. Camera based pose estimation techniques can also be used for multi-target localization [15], [16], however they have privacy concerns and would likely not be able to access existing infrastructure. To solve these problems, we propose SpringLoc, a new DFL algorithm based off RSSI histogram difference and Spring-relaxation, as shown in Fig. 1. It requires fewer sensor nodes while maintaining an acceptable localization accuracy. Thus, one is able to utilize existing smart home infrastructure, e.g. existing Wi-Fi, Zigbee or Bluetooth smart sensors, to provide indoor localization as a secondary service. SpringLoc also does not require any offline calibration measurements.

SpringLoc records the RSSI values between all transmitting (TX) and receiving (RX) nodes and forms two RSSI histograms for each link. The first histogram is formed by taking a weighted average of recent RSSI values. The second histogram is formed using a long-term weighted average of the RSSI values. At each timestep, the difference between these two histograms is calculated for each link. Links whose histogram difference exceed a predefined threshold are deemed to be 'affected links'. These are the links whose RSSI values have been impacted by the presence of the entity, with the short-term histogram exhibiting significant variation from the long-term one. An artificial spring anchor is defined at the intersection point of the affected links after removing outliers. Each spring acts as an attractive force on the tracked entity, pulling it towards its own anchor, as shown by the springs in Fig. 2. The spring-relaxation algorithm then iteratively localizes the target by equalizing the forces between the set of springs. Each spring has a weight based on the distance between the contributing affected links. The force associated with each spring is defined by this weight and the distance from the previous position estimate. If the intersection between two links does not fall within the localization environment, the closest point from each link is taken as the location of the anchor.

This is how the rest of the paper is organized. Section II covers related DFL techniques, why RSSI has been utilized over CSI and an overview of previous Spring Relaxation approaches. Section III describes the proposed SpringLoc algorithm. Section IV outlines the experimental setup and results. Section V provides a discussion on the experimental results and Section VI concludes the manuscript.

II. RELATED WORKS

DFL has become a popular research topic, as it allows for untagged entities to be tracked, enabling a wide range of usage scenarios. DFL techniques that are based on technologies readily available in a smart home, e.g. Bluetooth, ZigBee or Wi-Fi, can be categorized into three major approaches: 1) Fingerprinting, 2) Link-based or 3) Radio Tomographic Imaging.

A. FINGERPRINTING

Fingerprinting schemes consist of two phases. In the offline phase, the environment is divided into a grid. An initial measurement is taken when the environment is empty, and successive RSSI measurements are taken with an entity in each known grid location. This measurement set forms the fingerprint database. During the online phase, live RSSI measurements are compared with the fingerprint

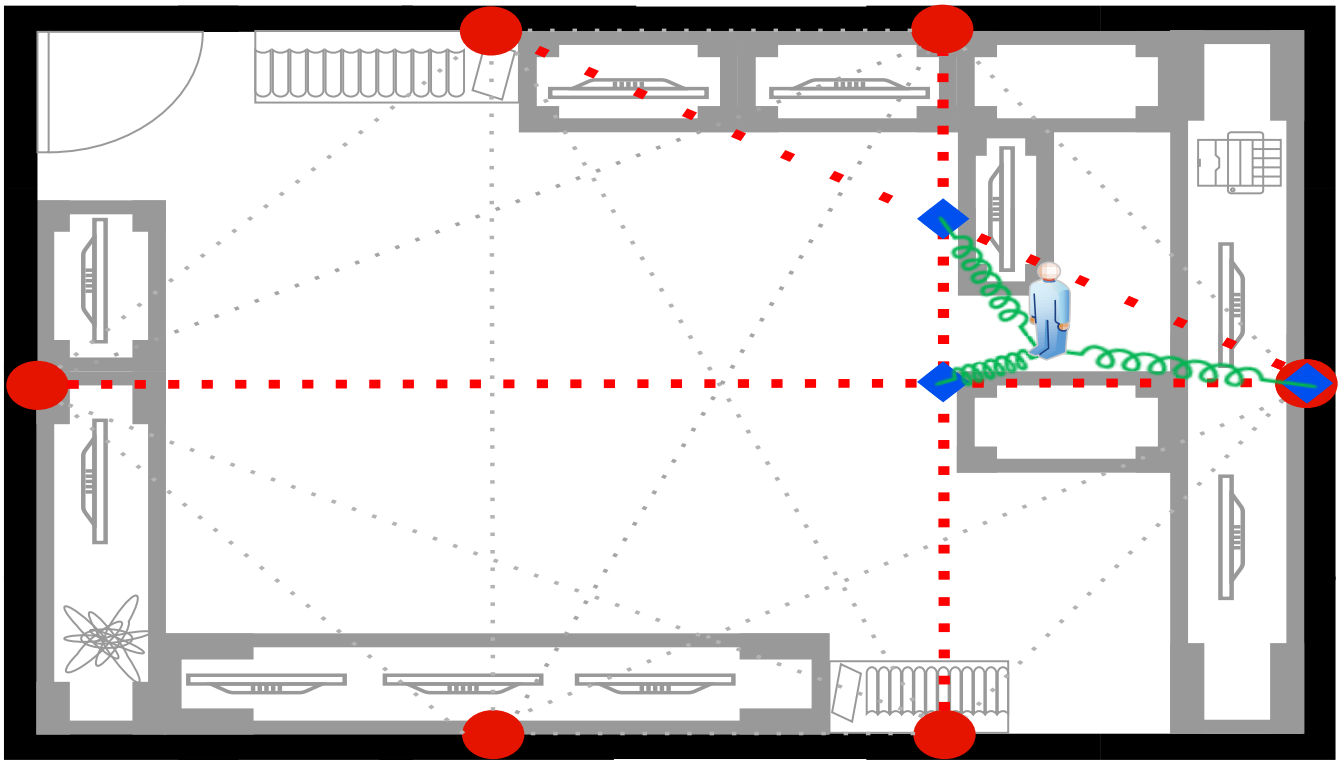

FIGURE 2. Springloc with three affected links.

database, with location estimation performed via classification.

The 'Sequential Counting, Parallel Localization' (SCPL) algorithm first counts the number of subjects in an environment by using successive cancellation to remove the influence of the subjects with the strongest influence each round. Once the number of subjects is known, SCPL incorporates human movement constraints and environmental geometric constraints to track each subject using a conditional random field (CRF) [17]. The 'ACcurate and Efficient' (ACE) localization algorithm incorporates an energy-minimization framework followed by a Markov-based CRF and clustering to smooth transitions between neighboring locations [18]. The 'geometrical localization, fingerprinting device free localization' (GL-FDFL) algorithm improved traditional fingerprinting by reducing the search area of possible fingerprint locations by geometrically restricting it based on the area bounded by shadowed links [19]. The 'Energy-Efficient High-Precision Multi-Target-Adaptive' (E-HIPA) algorithm used compressive sensing and an adaptive orthogonal matching pursuit algorithm to track multiple targets, using a sparse link network [20]. Chiang *et al.* [21] integrated fuzzy logic into a support vector machine (SVM) based DFL approach to improve the classification accuracy of a pure SVM DFL approach by 7.8%. Mager *et al.* [22] sought to improve the accuracy of fingerprint-based approaches as the database degrades due to environmental changes. Experimental results show that Random Forest based classification is more robust to environmental changes than traditional K-Nearest Neighbor (KNN), Linear Discriminant Analysis (LDA) or SVM approaches. Wang *et al.* [8] proposed a novel deep learning approach to reduce the offline training effort by automatically learning features using a sparse autoencoder network. A Soft-Max regression-based classifier is then used to predict a user's location, activity and gesture. The WiDet approach [23] augments the offline training data by resampling some windowed sample sub-sets to simulate different walking speeds. Localization is performed using a Convolutional Neural Network (CNN), which is shown to outperform a traditional approach based on RSSI wavelet features and Bayes classification. Huang *et al.* [5] model DFL as a spare representation problem which they solve using a variant of the iterative shrinkage-thresholding algorithm. Zhang *et al.* [24] implemented a parameterized extreme learning machine (ELM) approach to DFL which was shown to outperform existing WKNN, SVM and RTI techniques.

Though E-HIPA was able to reduce the number of nodes required, and Mager et al reduced the retraining effort, all fingerprinting approaches require extensive offline calibration, and suffer degradation due to any significant environmental changes. This limits their usability in smart homes, where it would be difficult to create a generalized calibration approach that could be followed by a regular end-user. Another difficulty is that calibration must be redone any time the environment changes significantly, which may be untenable in diverse, realistic environments.

B. LINK-BASED

Link-based or model-based schemes work by creating models to analyze the effect a subject has on a TX-RX link. A target is considered present along a link when the model deviates away from its steady-state by a predefined threshold. Particle filters are often used for positioning as they allow for a subject to be localized as the centroid of multiple affected links [2], [25]–[27].

Guo *et al.* [25] developed an Exponential-Rayleigh model for received signal strength (RSS), coupled with a particle filter for multi-target localization and tracking. Zheng and Men [26] represent the RSS model as a Gaussian mixture, with online re-parameterization to ensure correct detection, and a particle filter for localization and tracking. Saeed *et al.* [27] estimates a density function for each link based on a sliding window of RSSI variance. The system detects links as anomalous if they exceed a predetermined critical bound of the density function. The anomalous links are passed to a particle filter which performs localization and tracking [27].

Link-based schemes' accuracy is based on the reliability of their link model. If the model is not updated regularly, environmental change can degrade the system. Moreover, if noisy live measurements are used to update the link models, they may diverge over time. Secondly, most link-based models use a particle filter to solve the localization problem. Particle filters are very computationally expensive and unlikely to be feasibly run on commercial-off-the-shelf (COTS) embedded devices. Furthermore, since parallel particle filters would likely be required for multi-target tracking to ensure convergence in noisy environments, these algorithms do not scale well for realistic environments.

C. RADIO TOMOGRAPHIC IMAGING

Radio Tomographic Imaging (RTI) solves an ill posed linear inverse problem to generate an output image. The brightest pixel within the output image defines a subject's location estimate. RTI approaches are typically coupled with a Kalman filter on the output images to provide subject tracking. The original RTI implementation used RSSI attenuation as a feature [28]. More recent approaches have improved RTI by using variance as a feature and performing subspace decomposition [4], using multiple channels [29], using directional antenna arrays [30], or using a histogram difference feature [3]. Recently, a multi-frequency approach using both 433MHz and 868MHz radios managed to attain sub-meter accuracy in a complex indoor environment of approximately 115m^2, using 39 nodes [31]. Modern RTI approaches require minimal calibration and are less computationally complex (in the online phase) than link-based approaches, while still being able to track multiple targets. However, they require a significant node density to attain their accuracy, making them unsuitable for smart home use, where significant infrastructure modification cannot be justified.

D. CHANNEL STATE INFORMATION

The Channel State Information (CSI) metric has become a popular localization metric over RSSI as it is more immune to the adverse effects of multipath propagation [32] and outperforms RSSI based methods [33]. Since CSI offers more fine-grained information than RSSI, it has been extensively utilized in machine learning based DFL approaches including shapelet learning [34], SVM [9], [35], Random Forest [36], HMM [37], and Deep Learning [38]. A shortcoming of CSI is that it is currently only accessible using modified drivers in legacy Intel 5300 [11], [39], Atheros ath9k [12] based devices, or by using Software Defined Radio (SDR) platforms like USRP [40] or WARP [41]. Even though there has been significant research interest in using the modified drivers since they were released in 2011 and 2015 respectively [42], no vendor has provided access to the CSI metric to end users in any subsequent hardware releases. This means that a CSI based DFL solution cannot be recommended for smart homes, as the metric is not readily available in COTS hardware. Furthermore, smart home networks are commonly implemented using Zigbee, Bluetooth Low Energy (BLE) or Wi-Fi equipment. While the RSSI metric supports Zigbee, BLE and Wi-Fi equipment, the CSI metric only supports the aforementioned legacy Wi-Fi devices. Therefore, it is not suitable for integration into existing smart homes.

E. SPRING-RELAXATION

Spring-relaxation aims to reach equilibrium among a set of artificial springs. It has been used for sensor localization in Wireless Sensor Networks (WSNs) [43]–[45]. A similar energy minimization technique called 'potential fields' has found extensive use in obstacle avoidance and navigation of autonomous robots [46]–[49]. As far as the authors are aware, the concept of spring-relaxation has not been applied to DFL. Spring-relaxation has the benefit of only requiring a few anchors per target, as opposed to potentially thousands of particles used per target in a traditional particle filter. Spring-relaxation techniques require spring anchors, which are not readily apparent in the concept of DFL. Therefore they must first be defined in our calibration-free algorithm, as described in Section III. Spring-relaxation also has the benefit of allowing for adaptively weighted springs, which, with respect to DFL, can ensure high accuracy across a range of different target speeds.

F. CONTRIBUTIONS

As far as the authors are aware, the concept of spring-relaxation has never been practically implemented with respect to wireless DFL. This leads to the following novel contributions:

1) Apply the concept of spring-relaxation to wireless DFL as a form of localization and tracking
2) Provide a calibration-free way of providing the DFL spring-relaxation algorithm with artificial anchor points, during live operations

3) Provide experimental results across two diverse and realistic environments which show that spring-relaxation can outperform existing state-of-the-art RSSI-based DFL approaches, under both high and low node densities, for varying walking trajectories

III. ALGORITHM

DFL systems that use RSSI values must choose a feature to determine whether an entity is influencing the propagation of any specific link. A commonly utilized feature has been either RSSI difference or absolute difference (also termed as RSSI attenuation) where the current RSSI value is subtracted from one taken during offline measurements when no one is present. Unfortunately, this metric does not work well in through-wall environments and requires offline measurements. RSSI variance is another commonly utilized feature that works better in through-wall environments. However, it cannot track stationary targets.

RSSI histogram difference, as featured in [3] has the benefit of incorporating both mean and variance RSSI features, with neither of their limitations. This is beneficial as it allows for a feature detector that does not require an offline calibration phase. Another benefit of this feature is that it looks for change in the RSSI values caused by movement, irrespective of whether the change increases or decreases the RSSI values. This allows for the metric to work in multipath rich environments, where the magnitude RSSI change cannot be predicted in advance. This allows the metric to work with both stationary and moving targets, and in both open and through-wall environments.

The SpringLoc approach is broken into five modules which are described below and shown in Fig. 1. The first module forms the long-term and short-term histograms of each link, required for calculating the RSSI histogram difference feature. Module two extracts the most prominently affected links. Module three handles cases when too few affected links were detected. Module four defines the spring anchor points and their weights and module five performs an iterative adaptive spring relaxation approach that localizes and tracks a subject.

A. HISTOGRAM FORMATION

The RSSI difference feature is formulated by arranging incoming RSSI values into histograms averaged over either a short or a long period of time. The histogram difference for each link can then be found by computing an empirical histogram distance between the long-term histogram (L) and short-term histogram (S) for each link. Using an exponentially weighted moving average (EWMA) weighting scheme, the histograms can be defined as:

$$h_l^t = (1-\alpha)\, h_l^{t-1} + \alpha f_R\left(R_l^t\right) \tag{1}$$

where h_l^t is the histogram of link l at time t and R_l^t is the RSSI value of link l at time t. α, the forgetting factor, has a value between 0 and 1 and governs how much recent RSSI values contribute to the histogram. Hence a large value will help formulate the S while a low value will formulate the L. Assuming RSSI values are quantized with a step-size of one, and have a range between 1 and N, f_R is a function that given an RSSI value will return a vector of length N, with a value of 1 at the index of the RSSI value, and 0 elsewhere.

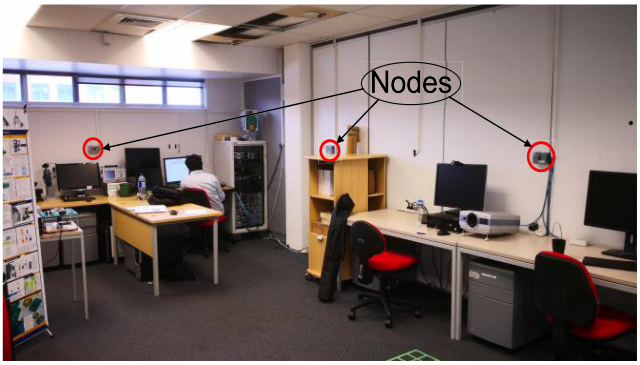

FIGURE 3. Cluttered office space environment.

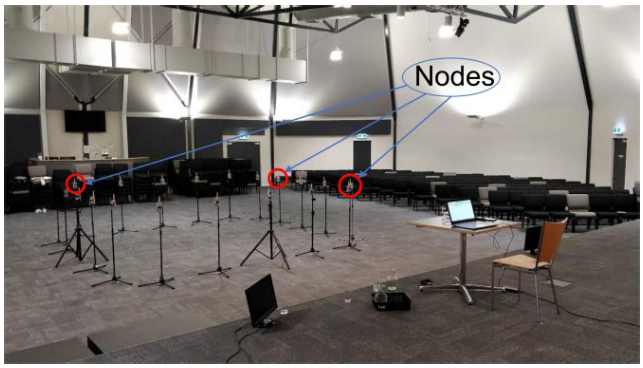

FIGURE 4. Church hall environment.

The difference, $KD(S, L)$, between the two histograms S and L, are computed using the Epanechnikov kernel distance in accordance with the literature [3], and can be defined as:

$$KD(S, L) = S^T KS + L^T KL - 2S^T KL \tag{2}$$

where T represents a transpose operation and K is an NxN matrix defined by:

$$K(i, j) = \begin{cases} \frac{3}{4}\left(1 - \frac{|i-j|^2}{\varpi}\right), & |i-j| \le \varpi \\ 0, & otherwise \end{cases} \tag{3}$$

where ϖ is a kernel smoothing parameter.

B. LOCATE AFFECTED LINKS

After computing the histogram difference for each link, the system must determine which links have been triggered by a target's presence. Since the presence of a target will cause an increased difference between the short-term (S) and long-term (L) histograms, a link threshold is defined as:

$$\zeta^t = f\left(KD(S, L)_{1:l}^t\right) \tag{4}$$

$$f(x) = \begin{cases} x, & x > \beta \\ nothing, & x \le \beta \end{cases} \tag{5}$$

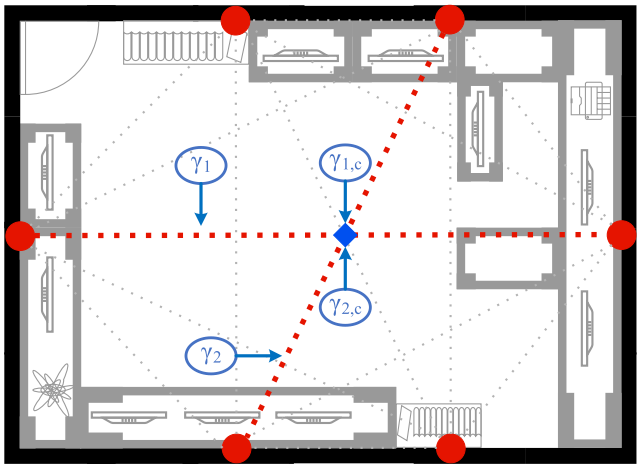

FIGURE 5. Affected link pair with intersection point.

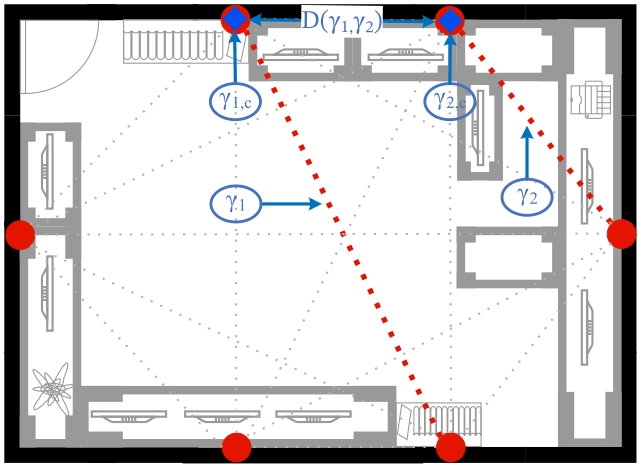

FIGURE 6. Affected Link pair with no intersection point.

where function f iterates through each link l in KD, adding them to an array, ζ, if they exceed predefined link threshold constant, β.

After locating the affected links, SpringLoc defines appropriate spring anchors, and their weights. This requires the successful detection of at least two affected links. If less than two affected links are detected, the algorithm utilizes the approach outlined in Section E.

C. DEFINE SPRING ANCHORS / WEIGHTS

Once all the affected links have been located, the system needs to translate this into the target's location. Intuitively, the subject is more likely to be in a region where there is a higher density of affected links. To help define this region, we locate a set of points that reside within the unknown region. For each link l in ζ, we define the coordinate of the transmitting node as TX_l, the coordinate of the receiving node as RX_l, and the line segment formed between TX_l and RX_l as γ_l. Sunday's geometric method [50] was used to either: find the intersection point of each pair of line segments, or the minimum distance between them within the environment. This can be clearly observed in Fig. 5 and Fig. 6. Fig. 5 shows two intersecting affected links γ_1 and γ_2. By defining the closest point in line segment one as $\gamma_{1,c}$ and the closest point in line segment two as $\gamma_{2,c}$, we know that $\gamma_{1,c} = \gamma_{2,c}$ as the links intersect. These points are represented by the blue diamond in Fig. 5. This also means that the minimum distance between the two links, $D(\gamma_1, \gamma_2) = 0$. In Fig. 6, the affected link pair does not intersect within the test environment. This means that the $\gamma_{1,c}$ and $\gamma_{2,c}$ points are defined by the node locations as shown in Fig. 6. In this case since $\gamma_{1,c} \neq \gamma_{2,c}$, $D(\gamma_1, \gamma_2) > 0$.

Once all intersection points have been calculated, the system needs to define each spring anchor, and its associated weight. The spring anchor points set (SA), are defined as:

$$SA = g(\gamma_{1:end}) \tag{6}$$

$$g(x_i, x_j) = \begin{cases} [\gamma_{i,c}\gamma_{j,c}], & D(x_i, x_j) < \eta \\ nothing, & D(x_i, x_j) \geq \eta \end{cases} \tag{7}$$

where g is a function that iterates through each pair of intersection points, adding them to the array of spring anchor points, SA, if the distance between them,$D(\gamma_i, \gamma_j)$, as shown in Fig. 6, is less than the distance constant, η. This is done to exclude link pairs that do not share close proximity.

After selecting the initial anchor points, a filter is applied to only keep points surrounding the median coordinate values in both x and y directions. The final set of spring anchor points (SA_f) is defined as:

$$SA_f = m(SA_{1:end}) = \begin{cases} SA_i, (\tilde{x} - \rho\sigma_x) \leq SA_{i,x} \leq (\tilde{x} + \rho\sigma_x) \\ (\tilde{y} - \rho\sigma_y) \leq SA_{i,y} \leq (\tilde{y} + \rho\sigma_y) \\ nothing, \quad otherwise \end{cases} \tag{8}$$

where m is a function that iterates through each spring anchor, only returning ones that are close to the median x and y value. $SA_{i,x}$ and $SA_{i,y}$ represent the x and y coordinate of spring anchor i respectively, $\tilde{x}$ is the median x coordinate from the SA point set, ρ is a SA selection constant, σ_x is the standard deviation of the x values from the SA point set, $SA_{i,x}$ is the x coordinate for point i in SA, $\tilde{y}$ is the median y coordinate value in SA, $SA_{i,y}$ is the y coordinate for point i in SA, and σ_y is the standard deviation of the y values from the s the x coordinate for point i in SA point set.

Once the final spring anchor points have been defined at timestep t, SA_f^t, they need to be weighted. Each spring in SA_f^t receives a weight defined by:

$$W_k^t = KD_i^t * KD_j^t \tag{9}$$

where indexes i and j were used by (7) for defining a SA point, now stored in $SA_{f,k}^t$. The weights are then normalized between 0 and 1 using:

$$W^t = \frac{(W^t - min(W^t))}{max(W^t) - min(W^t)} \tag{10}$$

TABLE 1. Springloc parameters.

Symbol	Description	Value
α_S	short-term histogram forgetting factor	0.9
α_L	long-term histogram forgetting factor	0.05
ϖ	kernel smoothing parameter	30
β_{20}	affected link threshold (20 nodes)	1.4
β_6	affected link threshold (6 nodes)	0.63
η	affected link proximity constant	0.5
ρ	spring anchor selection constant	2
ψ	maximum spring iterations per timestep	8
τ	spring step size scaling parameter	0.05
δ	Spring early breakout parameter	0.015

D. ADAPTIVE SPRING RELAXATION

The iterative spring-relaxation approach takes the final anchor points set, spring weights, and the previous position estimate as arguments. It is defined by parameters including a max number of iterations (ψ), a step size (τ), and breakout parameter (δ) which stops the algorithm early if convergence is reached. In a single iteration, the distance vector between the previous location estimate, *prevPos*, and each spring anchor is calculated as:

$$\overrightarrow{dis_k} = prevPos - SA_{f,k} \tag{11}$$

with the force defined as:

$$f_k = \overrightarrow{dis_k} * W_k \tag{12}$$

Assuming there are n springs, the net force over all springs is defined as:

$$\sum_{k=1}^{n} (netf = netf - f_k) \tag{13}$$

with the position estimate defined as:

$$Pos = prevPos + \tau * \frac{netf}{n} \tag{14}$$

A full pseudocode breakdown of the SpringLoc algorithm is included in Algorithm 1 and Algorithm 2 and all parameter values used are given in Table 1.

E. ALGORITHM EDGE CASES

These are the scenarios that may cause the algorithm to either not converge correctly or perform suboptimally. The first case arises when no affected links, ζ^t, are triggered in module two. This can occur if the target is walking through a temporary blind-spot or has stood relatively motionless for a considerable duration of time. To resolve this, we implement two cases.

If there are no affected links across multiple timesteps, ($length(\zeta^t) = 0$) and ($length(\zeta^{t-1}) = 0$), we assume the subject is currently stationary and set the current position estimate to the previous prediction estimate ($posEstimate = prevPos$). This ensures the subject stays located at the last known spring convergence target. However, if the previous timestep had affected links, $length(\zeta^{t-1}) > 0$, we assume that the subject is moving through a momentary blind-spot. Since no location information is available in the current timestep, the subject is assumed to be maintaining the same velocity and heading as their previous timestep. Their position estimate is given as:

$$posEstimate = prevPos + \overrightarrow{project} \tag{15}$$

where $\overrightarrow{project}$ is a vector defined by the previous trajectory:

$$\overrightarrow{project}_t = posEstimate_{t-1} - prevPos_{t-1} \tag{16}$$

The other edge case occurs when only one affected link is detected, $length(\zeta^t) = 1$. Since module four requires at least two affected links to calculate spring anchors, an artificial link is inserted into the affected links array. The artificial 'injected' link is defined by:

$$\begin{aligned} \mathbf{TX_{injected}} &= prevPos \\ \mathbf{RX_{injected}} &= prevPos + \vec{project} \\ \zeta^t_{injected} &= [\mathbf{TX}_{injected}, \mathbf{RX}_{injected}] \end{aligned} \tag{17}$$

IV. EXPERIMENT AND RESULTS

SpringLoc infers a subject's location by analyzing the changes in RSSI values across a network of wireless links. The network consisted of 20 Texas Instruments CC2530 Zigbee radios for the first experiment, and six radios for the second and third experiments. The radios operated at maximum power and were set to channel 26. The network was set up with a token ring protocol, where each node takes turns sending a broadcast packet. The broadcast packet's are received by every other node within range, which would record the ID of the transmitting (TX) node and the packets RSSI value. Each transmitted broadcast packet would contain its own ID, followed by a list of the last round's received RSSI values. A master node listens to all network traffic and sends all link RSSI values back to a PC for live processing. The network was set up to run at 5Hz (i.e. 5 RSSI values for every network link, every second). If the master node detects that a node has missed incoming packets

(as its broadcast RSSI list only contained a few values), it would fill in the missing RSSI values with known dummy values to keep the data structure sizes consistent. This also allows for the PC to know which packets were dropped for each node, for each time frame.

Experiments were conducted in two diverse environments. The first environment consisted of a church hall, which had the chairs removed from the center of the room, giving approximately 120m^2 of open space. The sensors were mounted on stands 1.2m above the ground and placed in a square encompassing 25m^2 in the center of the open space. The second environment consisted of a cluttered office space of approximately 44m^2, where computers and laboratory equipment were set up around the perimeter of the room. The nodes were wall-mounted around the perimeter of the room at 1.4m above the ground. In both environments, the Wi-Fi was turned off and a Rohde & Schwarz Spectrum Rider

Algorithm 1 SpringLoc Algorithm

for $t = 1 : t_{end}$ — *At time instant t*
  for $i = 1 : l$ — *Iterate through each link*
    $S_i^t = (1-\alpha_s)\, S_i^{t-1} + \alpha_s \mathbf{f_R}\left(R_i^t\right)$ — *Compute the short-term histogram (S) using (1)*
    $L_i^t = (1-\alpha_L)\, L_i^{t-1} + \alpha_L f_R\left(R_i^t\right)$ — *Compute the long-term histogram (L) using (1)*
    $KD_i^t = S_i^{tT} K S_i^t + L_i^{tT} K L_i^t - 2\, S_i^{tT} K L_i^t$ — *Compute the kernel-distance between S and L using (2)*
    if $KD_i^t > \beta$ — *Check if link is affected by subject's presence*
      $\zeta^t = [\zeta^t KD_i^t]$ — *Add affected links to the end of the link set using (4)*
    end
  end
  if $(length(\zeta^t) == 0)\&\&(length(\zeta^{t-1}) > 0)$ — *Check for edge cases*
    $\mathbf{posEstimate} = \mathbf{prevPos}+\mathbf{project}$ — *Accounting for an edge case, as in (15)*
  elseif $(\mathbf{length}(\zeta^t) == 0)\&\&(length(\zeta^{t-1}) == 0)$
    $\mathbf{posEstimate} = \mathbf{prevPos}$
  else
    if $length(\zeta^t) == 1$ — *Fix edge case if only one affected link present using (17)*
      $\mathbf{TX}_{injected} = \mathbf{prevPos}$
      $\mathbf{RX}_{injected} = \mathbf{prevPos}+\vec{project}$
      $\zeta^t = [\zeta^t(\mathbf{TX}_{injected}, \mathbf{RX}_{injected})]$
    end
    for $j = 1 : length(\zeta^t)$-1 — *Iterate through each pair of affected links*
      for $k = 2 : length(\zeta^t)$
        $[\gamma_{j,c}, \gamma_{k,c}] = ClsPoints(TX_j, RX_j, TX_k, RX_k)$ — *Calculate the closest point pairs using Sunday's method [50]*
        $D\left(\gamma_{j,c}, \gamma_{k,c}\right) = DisBetSeg(TX_j, RX_j, TX_k, RX_k)$ — *Calculate the distance between each affected pair using Sunday's method [50]*
        if $D\left(\gamma_{j,c}, \gamma_{k,c}\right) < \eta$ — *If affected link pairs points are within close proximity, add to spring anchor set using (7)*
          $SA = [SA\gamma_{j,c}\gamma_{k,c}]$
          $W_{tmp} = [W_{tmp}(KD_j^t * KD_k^t)(KD_j^t * KD_k^t)]$ — *Calculate weights with same indexing as SA*
        end
      end
    end
    for $i = 1 : length(SA)$
      if $((\tilde{x} - \rho\,\sigma_x) \leq SA_{i,x} \leq (\tilde{x} + \rho\sigma_x))\&\&$ $((\tilde{y} + \rho\sigma_y) \leq SA_{i,y} \leq (\tilde{y} + \rho\sigma_y))$ — *If both x and y coordinates of SA fall within bounds set around the medians, include SA in the final SA_f using (8)*
        $SA_f^t = [SA_f^t SA_i]$
        $W^t = [W^t W_{tmp_i}]$ — *Create final weight set as in (9)*
      end
    end
    for $i = 1 : length(W^t)$ — *Iterate through each final weight*
      $W_i^t = \frac{W_i^t - min(W^t)}{max(W^t) - min(W^t)}$ — *Normalize final weight set to between 0-1, as in (10)*
    end
    $posEstimate = ASR(SA_f^t, W^t, prevPos)$ — *Apply adaptive spring-relaxation, for t=1, set the previous position estimate as the environments entrance coordinate*
  end
  $\overrightarrow{project} = posEstimate - prevPos$ — *Create projection vector for edge cases, as in (16)*
  $prevPos = posEstimate$ — *Update previous position estimate*
end

Algorithm 2 $[posEstimate] = ASR(SA_f^t, W^t, prevPos)$

```
for i = 1 : ψ                                  Iterate until hitting the max number of iterations
  for j = 1 : n                                Iterate through each spring anchor (n = length(SA_f^t))
    dis⃗ = prevPos − SA_f,j                     Create a distance vector between the previous location estimate
                                                 and current spring anchor using (11)
    f = dis⃗ ∗ W_j^t                            Define the force for spring anchor j as in (12)
    netf = netf − f                            The net force is defined equivalent to (13)
  end
  posEstimate 𝒟 prevPos 𝒞 τ ∗ netf/n           Define the current iterations position estimate, as in (14)
  prevPos = posEstimate                        Update the previous position estimate
  if (mag(netf) ∗ τ)/n < δ                     Return early if position estimate has reached predicted location
      break
  end
end
```

FPH spectrum analyzer was used to ensure that there was no significant 2.4GHz interference present. Multiple walking trajectories were used per subject in both environments to ensure that we measured the algorithm's performance across the entire test space. For the test involving 20 nodes, the red and blue nodes were used, as shown in Fig. 7. For tests involving only 6 nodes, only the blue nodes were used.

We compared SpringLoc with three well cited DFL approaches from literature: Fingerprinting-based SCPL, link-based Ichnaea and RTI-based KRTI. For SCPL's fingerprinting, the office space was divided into 21 cells, and the church hall was divided into 25 cells. Though SpringLoc can run in real time, for these experiments, all RSSI values received by the processing pc were stored to a file. This allows for SpringLoc to be fairly compared with existing approaches across both environments, using the exact same readings. During testing, SpringLoc took an average of 8 milliseconds to process each timesteps RSSI values. Since this is significantly faster than the network rate of 5Hz, it confirms real-time viability. For each test, two sets of data were recorded. The first set of data included three fingerprint subsets. The first subset recorded RSSI readings when no subject was present in the test environment. The second subset consisted of labeled RSSI readings when a single roaming target was standing stationary at the center of each known cell. The third subset included labeled RSSI readings from when a single subject was moving randomly within the confines of a single known cell. This dataset was used by any algorithm that required them for offline calibration. The second dataset consisted of the RSSI values recorded at 5Hz intervals when a subject was walking at a known constant speed, through a known trajectory. Care was taken to ensure that the walking speed remained constant for each subject. For each of the two environments, three walking trajectories were explored. The subject would either walk near the perimeter in a clockwise or anticlockwise route, or the subject would follow a zig-zag trajectory covering the whole test environment. The trajectories traversed are outlined in Fig. 7 and were the same in both the church hall and cluttered office space environments. This allowed us to compare whether the trajectory had any influence on a DFL systems accuracy, and whether the effect of walking along different trajectories affected DFL algorithms differently.

For the first experiment, we utilized 20 nodes positioned in a square, in the church hall, as shown in Fig. 7. The church hall has no walls, support pillars or furniture within the immediate vicinity of the test environment. This minimizes the likelihood of dominant multipath propagation components, providing an opportunity to compare SpringLoc with other approaches under reasonably ideal conditions.

The second experiment, undertaken in the church hall, and third experiment, undertaken in the office, utilized only six nodes each. This was done to benchmark SpringLoc against other RSSI based DFL approaches with a node density that would be realistically found within a Smart Home deployment. We have represented the results with two Cumulative Distribution Function (CDF) plots for each experiment, (Fig. 8/Fig. 9 for experiment 1, Fig. 10/Fig. 11 for experiment 2 and Fig. 12/Fig. 13 for experiment 3). The first plot shows the accuracy of each separate walking trajectory (clockwise/anticlockwise/zig-zag) within an experiment. The second CDF plot combines the data from each trajectory into one dataset and shows the overall performance for a given experiment.

A. EXPERIMENT 1

Figure 8 clearly shows that a subject's trajectory can greatly influence the tracking ability of the traditional approaches. This is not an issue with SpringLoc as shown by the consistent shape of the empirical CDF error curve. SpringLoc also outperformed all other benchmarked approaches, achieving better median and 90^{th} percentile errors across every trajectory. When the data across trajectories is combined, as seen in Fig. 9, the effect of this becomes more pronounced, with SpringLoc outperforming existing approaches by a significant margin. This suggests that SpringLoc is more robust to

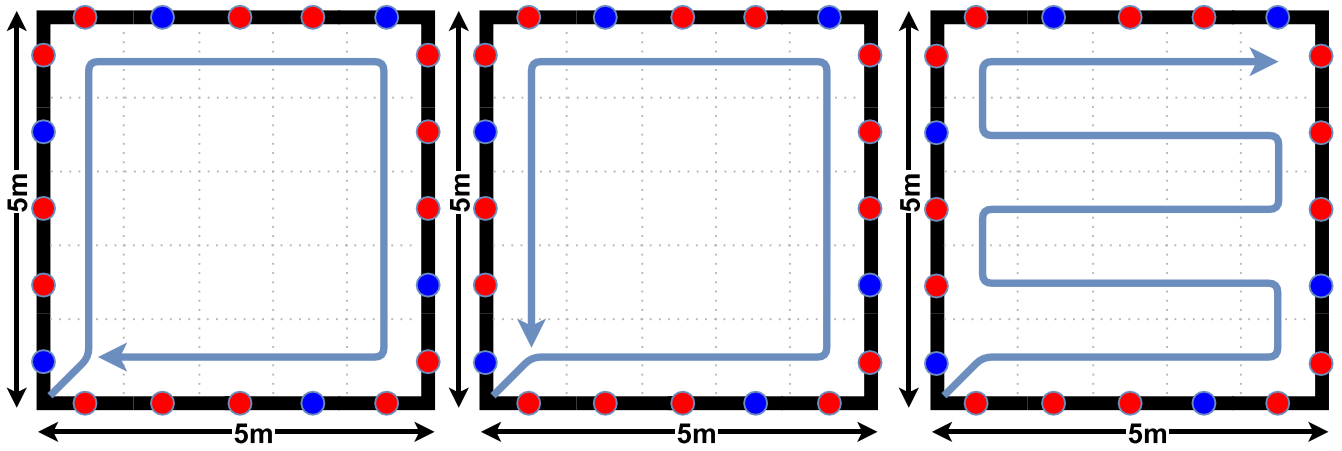

FIGURE 7. Church hall test environment–node placement and walking trajectories.

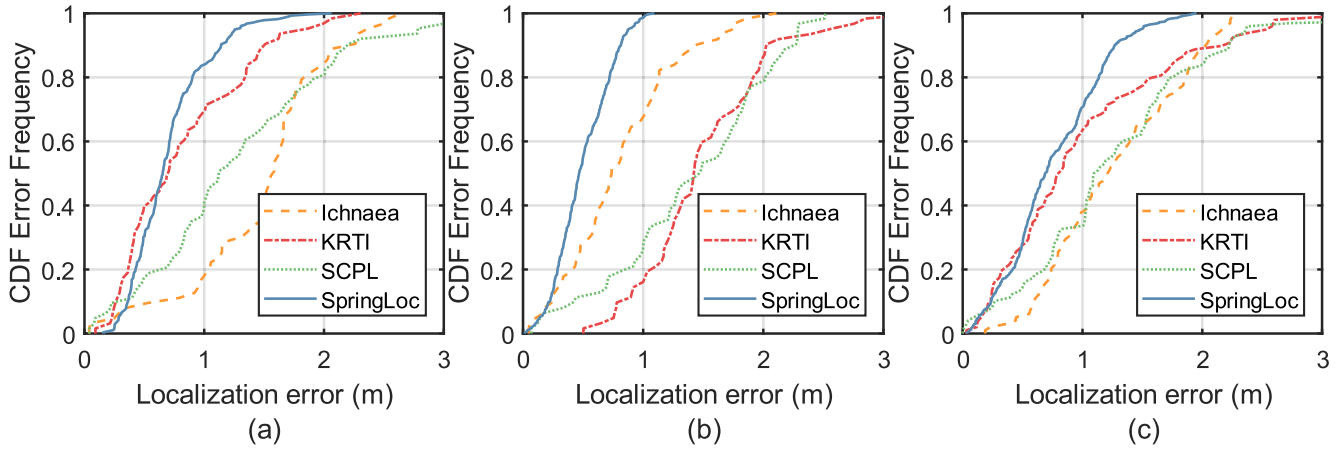

FIGURE 8. Church hall 20 node results–(a) clockwise trajectory, (b) anticlockwise trajectory, and (c) zigzag trajectory.

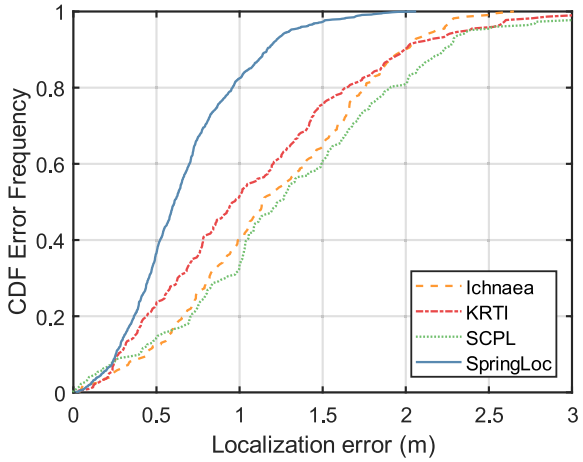

FIGURE 9. Church hall 20 node results–combined trajectories.

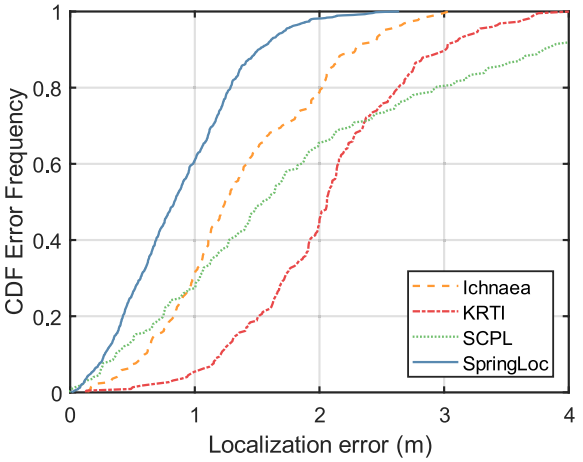

FIGURE 10. Church hall 6 node results–combined trajectories.

spurious large errors than other approaches. This can also be clearly seen by the maximum error in Fig. 9, where SpringLoc's maximum error is more than 1m lower than any other approach.

B. EXPERIMENT 2

When the number of deployed nodes was reduced to six, the performance of all algorithms degraded as expected. However, unlike the other approaches, SpringLoc managed

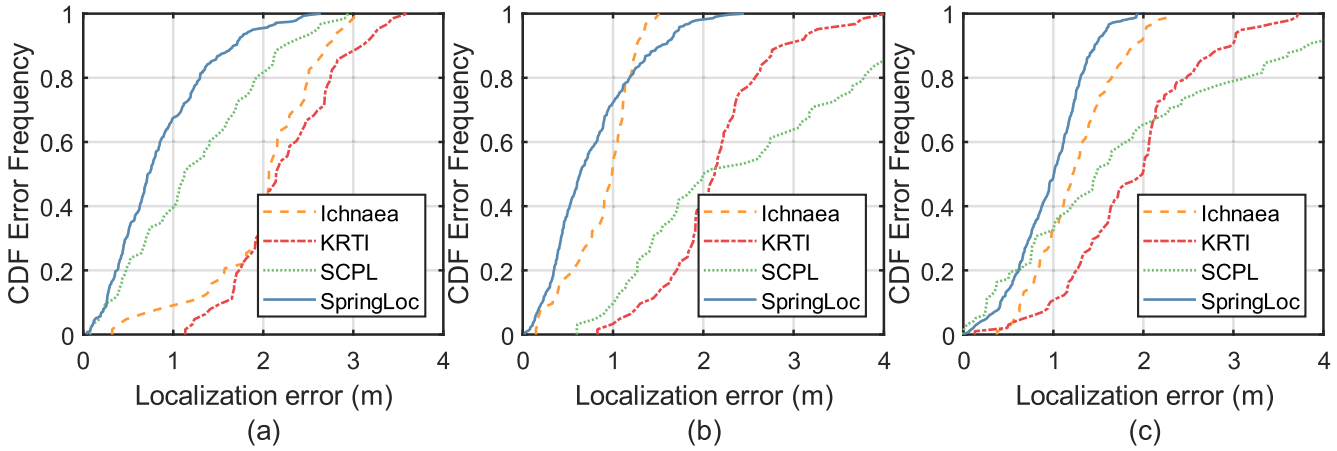

FIGURE 11. Church hall 6 node results (a) clockwise trajectory, (b) anticlockwise trajectory, and (c) zigzag trajectory.

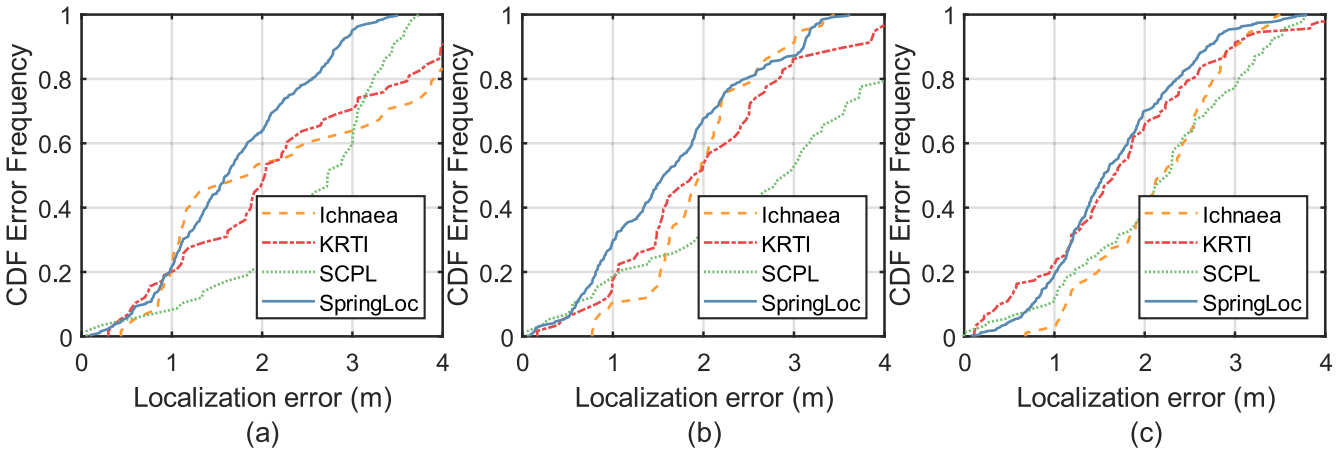

FIGURE 12. Office space results (a) clockwise trajectory, (b) anticlockwise trajectory, and (c) zigzag trajectory.

to maintain a sub-meter overall median error, as shown in Fig. 10, proving its viability even under a low node density. It was also relatively unaffected by the subject's trajectory as shown in Fig. 11. This shows that SpringLoc is resilient to varying walking trajectories and is also able to maintain a superior accuracy to existing approaches, even when the number of deployed nodes is reduced from 20 to 6.

C. EXPERIMENT 3

Experiment 3 utilized 6 nodes in a cluttered office environment with potential for significant multipath components, as shown in Fig. 3. SpringLoc outperformed all other benchmarked algorithms and maintained consistent performance across all trajectories as shown in Fig. 12. This suggests that SpringLoc's superior accuracy is less susceptible to environmental variations, as it maintained the best localization error across multiple indoor test locations. It was also the only approach that did not suffer from errors exceeding 4m, as shown in Fig. 13.

V. DISCUSSION

The CDF plots have been provided for SpringLoc as they allow for algorithms performance to be compared over the whole quartile range, rather than using a singular numerical metric. To provide consistency with existing literature, numerical results are also provided in Table 2. Since literature does not have an agreed numerical standard for benchmarking DFL algorithms, we use metrics standardised by indoor active tracking. The EvAAL framework recommends using the 'third quartile of point Euclidean error', equivalent to the 75^{th} percentile error [51]. The formal standard ISO/IEC 18305 has been recently introduced to standardize indoor localization scoring, however it does not provide explicit guidelines for DFL [52]. ISO/IEC 18305 mentions three scoring metrics that can be utilized by a DFL approach. It recommends using the Root-Mean-Square-Error (RMSE), Circular Error Probable (CEP), and Circular Error 95% (CE95). CEP is equivalent to the 2d 50^{th} percentile Euclidean error, and CE95 is equivalent to the 2d 95^{th} percentile Euclidean error. We have incorporated these, alongside the

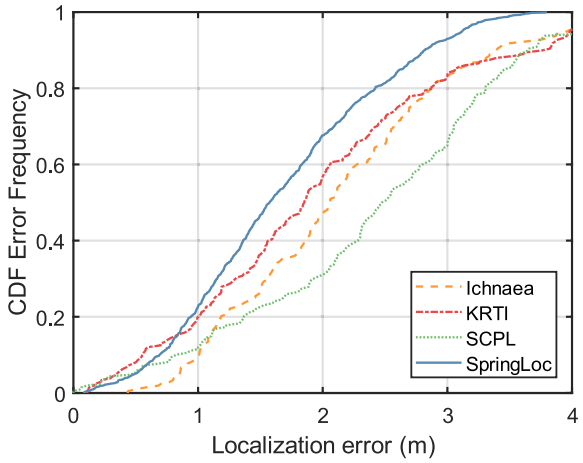

FIGURE 13. Office space results combined trajectories.

TABLE 2. Numerical results.

Experiment	Algorithm	RMSE	50% Error	75% Error	90% Error	95% Error
Experiment 1 Church Hall 20 Nodes	Ichnaea	1.34	1.138	1.66	2.01	2.20
	KRTI	1.29	0.97	1.48	1.99	2.39
	SCPL	1.54	1.23	1.78	2.23	2.42
	SpringLoc	**0.75**	**0.60**	**0.86**	**1.16**	**1.30**
Experiment 2 Church Hall 6 Nodes	Ichnaea	1.51	1.24	1.92	2.29	2.53
	KRTI	2.18	2.06	2.48	3.01	3.27
	SCPL	2.19	1.55	2.60	3.80	4.27
	SpringLoc	**1.00**	**0.83**	**1.21**	**1.51**	**1.72**
Experiment 3 Office Space 6 Nodes	Ichnaea	2.34	2.05	2.71	3.39	3.98
	KRTI	2.32	1.85	2.60	3.80	3.98
	SCPL	2.70	2.48	3.18	3.65	4.58
	SpringLoc	**1.85**	**1.57**	**2.22**	**2.82**	**3.12**

90^{th} percentile error in Table 2. As can be observed, SpringLoc significantly outperforms the other algorithms in every scenario. As expected, when the node density decreased to 6 nodes in experiment 2, from 20 in experiment 1, localization performance decreased with every algorithm. With the lower node density, SpringLoc was the only algorithm that managed to maintain a sub-meter median error. SpringLoc also outperformed other approaches in experiment 3 which featured both a low node density, and complex multipath propagation paths caused by the environment itself. It was the only approach that managed to maintain an RMSE below 2m, and a 90th percentile error below 3m.

Though SpringLoc outperformed all other approaches, it did experience a significant increase in median error from 0.83m in the open church hall, to 1.57m in the cluttered office space. SpringLoc utilizes EWMA weighted histograms, which actively dampen the effect of any spurious outlier values. However, it still suffers increased error if the multipath propagation introduces significant non-transient fading. This shows that while SpringLoc does not presume the noise will follow a zero mean gaussian distribution, it is still detrimentally affected if the complex propagation environment interferes with the average variance that a person introduces to the environment. If an accurate noise model could be attained for complex indoor environments, the accuracy of SpringLoc could improve by better understanding how the attenuation introduced by a target fluctuates due to noise.

Since SpringLoc can form its long-term histogram during live operation, it does not require offline measurements. Therefore, its parameters can be optimized in advance and deployed at an unknown environment without a significant setup cost. Parameters were tuned empirically within a test room, before being deployed in both the Church Hall and Office Space shown. Only the affected link threshold parameter (β) optimized for Church Hall and Office Space respectively. With the other parameters being kept unchanged from their initial test room tuning, SpringLoc still managed to outperform all existing approaches. This shows that the parameters are largely transferable between varying environments, while maintaining an acceptable localization accuracy. Since the affected link threshold is largely coupled with the number of nodes deployed, this could be further generalized based on a given number of nodes and deployment area. An end user only needs to know the number of deployed nodes, and approximate localization area to set up a SpringLoc based system. For example, based on our empirical testing, the maximum β value can be heuristically estimated as: $2 \times \left(\sqrt{\text{Nodes}}/\sqrt{\text{Area}\,(\text{m}^2)}\right)$. Thus, a 140m^2 house utilizing 8 nodes would have an estimated maximum β value of 0.53. When a large number of links are available, a large value can be set for β, which causes the system to only react to strongly affected links, resulting in increased accuracy. For low node deployments, there may be multiple areas with very sparse link density. Since this reduces the likelihood that moving subjects will trigger strong link interactions, the threshold is reduced. This acts to reduce the number of potential blind spots within an environment.

VI. CONCLUSIONS AND FUTURE WORKS

SpringLoc has shown that careful 'affected link' selection based on histogram difference, coupled with spring-relaxation can increase the performance of an RSSI based DFL approach in real life scenarios. In the initial 20 node benchmark, SpringLoc surpassed all other algorithms, achieving median and 90th percentile errors of 0.60m and 1.16m respectively. SpringLoc achieved a median localization accuracy of 0.83m in the church hall and 1.57m in the office space, surpassing existing approaches median error by up to 59%, when the node density was reduced.

Though SpringLoc does not require any offline measurements, it does need to know the location of the transceiver nodes. Future work could include performing self-localization [53-55] on the nodes themselves, while following a deployment strategy that could be realistically followed by end users. This paper only investigated single entity localization; multi-entities is left for future work. This could be accomplished by using a separate set of springs for each detected entity, after accurately counting the number of subject's present, and the affected links for each. Furthermore, RSSI is a coarse metric when compared to Wi-Fi CSI. If CSI ever became readily accessible in COTS equipment, SpringLoc could be implemented using CSI to further improve the performance.

ACKNOWLEDGMENT

The authors thank Baden Parr for his suggestions on how to formulate DFL as a spring-relaxation problem and Nathaniel Faulkner and Gene Hopkins for their help with data collection. The authors would also like to thank the anonymous reviewers for their insightful comments that helped improve the quality of this paper.

REFERENCES

[1] Q. Lei, H. Zhang, H. Sun, and L. Tang, "Fingerprint-based device-free localization in changing environments using enhanced channel selection and logistic regression," *IEEE Access*, vol. 6, pp. 2569–2577, 2018.
[2] Z. Wang, H. Liu, S. Xu, X. Bu, and J. An, "Bayesian device-free localization and tracking in a binary RF sensor network," *Sensors*, vol. 17, no. 5, p. 969, Apr. 2017.
[3] Y. Zhao, N. Patwari, J. M. Phillips, and S. Venkatasubramanian, "Radio tomographic imaging and tracking of stationary and moving people via kernel distance," in *Proc. 12th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, Philadelphia, PA, USA, Apr. 2013, pp. 229–240.
[4] Y. Zhao and N. Patwari, "Robust estimators for variance-based device-free localization and tracking," *IEEE Trans. Mobile Comput.*, vol. 14, no. 10, pp. 2116–2129, Oct. 2015.
[5] H. Huang, H. Zhao, X. Li, S. Ding, L. Zhao, and Z. Li, "An accurate and efficient device-free localization approach based on sparse coding in subspace," *IEEE Access*, vol. 6, pp. 61782–61799, 2018.
[6] D. Yu, Y. Guo, N. Li, and D. Fang, "Dictionary refinement for compressive sensing based device-free localization via the variational EM algorithm," (in English), *IEEE Access*, vol. 4, pp. 9743–9757, 2016.
[7] R. Zhou, M. Hao, X. Lu, M. Tang, and Y. Fu, "Device-free localization based on CSI fingerprints and deep neural networks," in *Proc. IEEE 15th Annu. Int. Conf. Sens., Commun., Netw. (SECON)*, Jun. 2018, pp. 1–9.
[8] J. Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang, "Device-free wireless localization and activity recognition: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6258–6267, Jul. 2017.
[9] R. Zhou, X. Lu, P. Zhao, and J. Chen, "Device-free presence detection and localization with SVM and CSI fingerprinting," *IEEE Sensors J.*, vol. 17, no. 23, pp. 7990–7999, Dec. 2017.
[10] X. Chen, C. Ma, M. Allegue, and X. Liu, "Taming the inconsistency of Wi-Fi fingerprints for device-free passive indoor localization," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.
[11] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 1, p. 53, 2011.
[12] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity Wi-Fi," in *Proc. 21st Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Paris, France, 2015, pp. 53–64.
[13] M. Zhao *et al.*, "Through-wall human pose estimation using radio signals," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7356–7365.
[14] M. Zhao *et al.*, "RF-based 3D skeletons," presented at the Conf. ACM Special Interest Group Data Commun. (SIGCOMM), Budapest, Hungary, 2018.
[15] G. Papandreou *et al.*, "Towards accurate multi-person pose estimation in the wild," in *Proc. CVPR*, Jul. 2017, vol. 3, no. 4, pp. 4903–4911.
[16] R. Girdhar, G. Gkioxari, L. Torresani, M. Paluri, and D. Tran, "Detect-and-track: Efficient pose estimation in videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 350–359.
[17] C. Xu *et al.*, "SCPL: Indoor device-free multi-subject counting and localization using radio signal strength," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, 2013, pp. 79–90.
[18] I. Sabek, M. Youssef, and A. V. Vasilakos, "ACE: An accurate and efficient multi-entity device-free WLAN localization system," *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 261–273, Feb. 2015.
[19] B. Han, Z. Wang, H. Liu, S. Xu, X. Bu, and J. An, "Shadow fading assisted device-free localization for indoor environments," in *Proc. 8th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2016, pp. 1–5.
[20] J. Wang *et al.*, "E-HIPA: An energy-efficient framework for high-precision multi-target-adaptive device-free localization," *IEEE Trans. Mobile Comput.*, vol. 16, no. 3, pp. 716–729, Mar. 2017.
[21] Y.-Y. Chiang, W. Hsu, S. Yeh, Y. Li, and J. Wu, "Fuzzy support vector machines for device-free localization," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf.*, May 2012, pp. 2169–2172.
[22] B. Mager, P. Lundrigan, and N. Patwari, "Fingerprint-based device-free localization performance in changing environments," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 11, pp. 2429–2438, Nov. 2015.
[23] H. Huang and S. Lin, "WiDet," presented at the 21st ACM Int. Conf. Modeling, Anal. Simulation Wireless Mobile Syst. (MSWIM), Montreal, QC, Canada, 2018.
[24] J. Zhang, W. Xiao, S. Zhang, and S. Huang, "Device-free localization via an extreme learning machine with parameterized geometrical feature extraction," *Sensors*, vol. 17, no. 4, p. 879, Apr. 2017.
[25] Y. Guo, K. Huang, N. Jiang, X. Guo, Y. Li, and G. Wang, "An exponential-Rayleigh model for RSS-based device-free localization and tracking," *IEEE Trans. Mobile Comput.*, vol. 14, no. 3, pp. 484–494, Mar. 2015.
[26] Y. Zheng and A. Men, "Through-wall tracking with radio tomography networks using foreground detection," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, 2012, pp. 3278–3283.
[27] A. Saeed, A. E. Kosba, and M. Youssef, "ICHNAEA: A low-overhead robust WLAN device-free passive localization system," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 1, pp. 5–15, Feb. 2014.
[28] J. Wilson and N. Patwari, "Radio tomographic imaging with wireless networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 5, pp. 621–632, May 2010.
[29] O. Kaltiokallio, M. Bocca, and N. Patwari, "Enhancing the accuracy of radio tomographic imaging using channel diversity," in *Proc. IEEE 9th Int. Conf. Mobile Ad-Hoc Sensor Syst. (MASS)*, Oct. 2012, pp. 254–262.
[30] B. Wei, A. Varshney, N. Patwari, W. Hu, T. Voigt, and C. T. Chou, "dRTI," presented at the 14th Int. Conf. Inf. Process. Sensor Netw. (IPSN), Seattle, WA, USA, 2015.
[31] S. Denis, R. Berkvens, G. Ergeerts, and M. Weyn, "Multi-frequency sub-1 GHz radio tomographic imaging in a complex indoor environment," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2017, pp. 1–8.
[32] Z. Yang, Z. Zhou, and Y. Liu, "From RSSI to CSI," *ACM Comput. Surv.*, vol. 46, no. 2, pp. 1–32, 2013.
[33] J. Xiao, K. Wu, Y. Yi, L. Wang, and L. M. Ni, "Pilot: Passive device-free indoor localization using channel state information," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2013, pp. 236–245.
[34] H. Zou, Y. Zhou, J. Yang, W. Gu, L. Xie, and C. J. Spanos, "WiFi-based human identification via convex tensor shapelet learning," in *Proc. AAAI 32nd Conf. Artif. Intell.*, 2018, pp. 1711–1719.

[35] X. Dang, Y. Huang, Z. Hao, and X. Si, "PCA-Kalman: Device-free indoor human behavior detection with commodity Wi-Fi," *EURASIP J. Wireless Commun. Netw.*, vol. 2018, no. 1, p. 214, 2018.
[36] H. Zou, Y. Zhou, J. Yang, W. Gu, L. Xie, and C. Spanos, "FreeDetector: Device-free occupancy detection with commodity WiFi," in *Proc. IEEE Int. Conf. Sens., Commun. Netw. (SECON)*, Jun. 2017, pp. 1–5.
[37] H. Zhang *et al.*, "Wireless non-invasive motion tracking of functional behavior," *Pervasive Mobile Comput.*, vol. 54, pp. 29–44, Mar. 2019.
[38] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, Jan. 2016.
[39] S. Kumar, D. Cifuentes, S. Gollakota, and D. Katabi, "Bringing cross-layer MIMO to today's wireless LANs," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 43, no. 4, pp. 387–398, 2013.
[40] (2019). *USRP Software Defined Radio Device*. [Online]. Available: https://www.ettus.com
[41] (2019). *WARP Project*. [Online]. Available: http://warpproject.org
[42] Y. Ma, G. Zhou, and S. Wang, "WiFi sensing with channel state information: A survey," *ACM Comput. Surv.*, vol. 52, no. 3, p. 35, 2019.
[43] W.-T. Yu, J.-W. Choi, Y. Kim, W.-H. Lee, and S.-C. Kim, "Self-organizing localization with adaptive weights for wireless sensor networks," *IEEE Sensors J.*, vol. 18, no. 20, pp. 8484–8492, Oct. 2018.
[44] B.-C. Seet, Q. Zhang, C. H. Foh, and A. C. M. Fong, "Hybrid RF mapping and Kalman filtered spring relaxation for sensor network localization," *IEEE Sensors J.*, vol. 12, no. 5, pp. 1427–1435, May 2012.
[45] J. Eckert, F. Villanueva, R. German, and F. Dressler, "Distributed mass-spring-relaxation for anchor-free self-localization in sensor and actor networks," in *Proc. 20th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul./Aug. 2011, pp. 1–8.
[46] J. Stipes, R. Hawthorne, D. Scheidt, and D. Pacifico, "Cooperative localization and mapping," in *Proc. IEEE Int. Conf. Netw., Sens. Control*, Apr. 2006, pp. 596–601.
[47] A. S. Paul and E. A. Wan, "RSSI-based indoor localization and tracking using sigma-point Kalman smoothers," *IEEE J. Sel. Topics Signal Process.*, vol. 3, no. 5, pp. 860–873, Oct. 2009.
[48] B. Li, H. Du, and W. Li, "A potential field approach-based trajectory control for autonomous electric vehicles with in-wheel motors," *IEEE Trans. Intell. Transp.*, vol. 18, no. 8, pp. 2044–2055, Aug. 2017.
[49] W. Li, C. Yang, Y. Jiang, X. Liu, and C.-Y. Su, "Motion planning for omnidirectional wheeled mobile robot by potential field method," *J. Adv. Transp.*, vol. 2017, Mar. 2017, Art no. 4961383.
[50] D. Sunday. (2012). Distance Between Lines, Segments and Their CPA (2D & 3D). Geomalgorithms.com. Accessed: May 2, 2019. [Online]. Available: http://geomalgorithms.com/a07-_distance.html
[51] F. Potortí *et al.*, "Comparing the performance of indoor localization systems through the EvAAL framework," *Sensors*, vol. 17, no. 10, p. 2327, 2017.
[52] *ISO/IEC JT, Information Technology—Real Time Locating Systems—Test and Evaluation of Localization and Tracking Systems*, document ISO/IEC 18305:2016, 2016. [Online]. Available: https://www.iso.org/standard/62090.html
[53] M. Al-Hattab, M. Takruri, H. Attia, and H. Al-Omari, "Decentralized localization in wireless sensor networks," in *Proc. Int. Conf. Elect. Comput. Technol. Appl. (ICECTA)*, Nov. 2017, pp. 1–5.
[54] C. Alippi and G. Vanini, "A RSSI-based and calibrated centralized localization technique for wireless sensor networks," in *Proc. IEEE 4th Annu. Int. Conf. Pervasive Comput. Commun. Workshops (PERCOMW)*, Mar. 2006, p. 5 and 305.
[55] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.

DANIEL KONINGS (M'17) received the B.E. degree (Hons.) in computer systems and electronics Engineering from Massey University, Auckland, New Zealand, in 2014, where he is currently pursuing the Ph.D. degree with the School of Food and Advanced Technology. His current research interests include indoor localization, digital signal processing, machine learning, the IoT, and wireless sensor networks.

FAKHRUL ALAM (M'17–SM'19) received the B.Sc. degree (Hons.) in electrical and electronic engineering from BUET, Bangladesh, and the M.S. and Ph.D. degrees in electrical engineering from Virginia Tech., USA. He is currently a Senior Lecturer with the School of Food and Advanced Technology, Massey University, New Zealand. His research interests include indoor localization, 5G and visible light communication, disaster management with 5G, the IoT, and wireless sensor networks. He is a Senior Member of the Institute of Engineering and Technology (IET).

FRAZER NOBLE received the B.E. degree (Hons.) in mechatronics, the M.E. degree (Hons.) in engineering, and the Ph.D. degree in engineering from Massey University, New Zealand, where he is currently a Lecturer with the School of Food and Advanced Technology. His research interests include mechatronics, robotics and automation, machine vision, and machine learning.

EDMUND M-K. LAI (M'82–SM'95) received the B.E. (Hons.) and Ph.D. degrees in electrical engineering from The University of Western Australia, in 1982 and 1991, respectively. He is currently a Professor in information engineering with the Auckland University of Technology, New Zealand. He has previously held Faculty Member positions at universities in Australia, Hong Kong, and Singapore. His research interests include digital signal processing, digital communications and networks, computational and swarm intelligence, and artificial neural networks. He is a Fellow of the Institution of Engineering and Technology (IET).