WarpMap: Accurate and Efficient Indoor Location by Dynamic Warping in Sequence-Type Radio-map

Xuehan Ye, Yongcai Wang
Renmin University
Beijing China

Wei Hu, Lei Song
IIIS, Tsinghua University
Beijing, China

Zhaoquan Gu
University of Hong Kong
Hong Kong, China

Deying Li
Renmin University
Beijing, China

***Abstract*—Radio-map based method has been widely used for indoor location and navigation, but remaining key challenges are: 1) laborious efforts to calibrate a fine-grained radio-map, and 2) the locating result inaccuracy and not robust problems due to random signal strength (RSS) noises. An efficient way to overcome these problems is to collect RSS signatures along indoor paths and utilize sequence matching to enhance the location robustness. But, due to problems of indoor path combinational explosion, random RSS loss during movement, and moving speed disparity during online and offline phases, how to exploit sequence matching in radio-map remains difficult.**

This paper proposes WarpMap, an efficient sequence-type radio-map model and an accurate indoor location method by dynamic warping. Its distinct features include: 1) an undirected graph model (Trace-graph) for efficiently calibrating and storing sequence-type radio-map, which overcomes the path combinational explosion and RSS miss-of-detection problems; 2) an efficient sub-sequence dynamic time warping (SDTW) algorithm for accurate and efficient on-line locating. We show SDTW can tolerate random RSS disparities at discrete points and handle the moving speed differences in on-line and off-line phases. The impacts of different warping distance functions, RSS preprocessing techniques were also investigated. Extensive experiments in office environments verified the efficiency and accuracy of WarpMap, which can calibrated within ten minutes by one person for $1100m^2$ area and provides overall nearly 20% accuracy improvements than the state-of-the-art of radio-map method.

I. Introduction

Radio-map based indoor locating, which characterizes each location-of-interest by the radio signal strength (RSS) signature at that location, has attracted great attentions [1] [2] [3] [4]. Comparing to other locating techniques, it provides key advantages including: 1) purely software-implementable on mobile phones without requiring additional hardware infrastructure; 2) privacy protection for working in navigation mode; and 3) reasonable accuracy with errors around 2-3 meters .

The routine of radio-map based locating generally has an offline site survey and an online locating phases. The offline phase is to calibrate the received signal strengths (RSS) at different known locations, which builds a *radio-map*, in which the RSS signatures at a location can be modeled by mean value, or probabilistic density function of received RSS values, which form deterministic [1] [2] or probabilistic-type [3] [4] radio-map respectively. In the online phase, a mobile target captures on-site RSS and searches in the radio-map to find the location whose RSS signature best matches the measured RSS, thus it determines its real-time location.

Despite of the advantages, several practical issues must be considered for practical applications: 1) it is very laborious to calibrate a fine-grained radio-map, especially for large application areas; 2) the point-type matching at discrete locations is sensitive to random online RSS noises and environment dynamics. To overcome these problems, previous works have devoted deliberated efforts to exploit different kinds of information and methods to reduce radio-map calibration cost and to improve the positioning accuracy and robustness.

A major approach to reduce calibration cost is unsupervised indoor locating method [5], which exploited environment signature including magnetic fluctuation, illumination intensity at specific spots as internal landmarks [6], and used dead reckoning by mobile phones [7] to track the inertial landmarks to conduct locating, so as to avoid the manually radio-map calibration. Another major approach exploited automatic labeling [8], which leveraged dead reckoning by motion sensors to construct a radio-map firstly in the radio space, then stretch and associate the radio-space geometry to physical space by geometrical matching using the path information from the floor-plan. Other approaches also exploited Expectation-Maximisation (EM) and Manifold methods to learn radio-map parameters by training parametric radio-map models [9] [10]. These methods, however, generally require additional sensors or depend heavily on the accuracy of dead reckoning, which is known difficult by using commodity mobile phones [7] [11]. The inaccuracy of radio-map model may also degrade the performance of locating.

Another approach to improve the locating accuracy and robustness is to build fine-grained or probabilistic radio-map to preserve more information [4] [12], but such kind methods requires higher calibration cost. Another major approach is to exploit K-Nearest Neighbor (KNN) [1], Support Vector Machine (SVM) [13] algorithms, etc. to tolerate random noises in the locating process. Some approaches used information fusion techniques, which exploited the motion continuity information, floor map information to design Bayesian filter [2], particle filter [14], and Markov Random Field models [15] to narrow down the search space to posteriorly improve the locating accuracy against noises. Geng et al. [16] proposed hybrid radio-map method to reduce calibration cost and to

978-1-5090-1732-4/16/$31.00 ©2016 IEEE

improve location accuracy. Yang et al. [17] also proposed adaptive radio-map to adapt to online environment dynamics.

A more straightforward way for easing the radio-map calibration and improving the location robustness is to collect RSS sequence along indoor paths to build sequence-type radio-map, which directly embeds the motion constraint and the path constraint information into the sequential RSS signatures. This, on one hand, can reduce the point-by-point calibration cost, and on the other hand can improve the positioning accuracy. But, several practical difficulties have obstructed such a direct approach: 1) the number of indoor paths grow exponentially with the number of path segments, making storage of RSS sequences for all paths inaccessible. Note that a path segment is the path within two adjacent crosses. 2) The random missing of RSS measurements during user movement, causing disparity of RSS sequence signatures for two travels on the same path. 3) The moving speeds and moving patterns of users in online phase and offline phase maybe different, leading to difficulty of accurate sequence matching. As a result, although some practical works collect RSS sequences along paths [8], the collected sequences are generally divided into points to construct point-type radio-map. The path structure and motion constraints are lost in the generated radio-map and the benefits of sequence signature are not fully utilized.

This paper revisits sequence matching in radio-map, which proposes *WarpMap* to tackle the challenges in sequence-type radio-map construction and online matching. The key idea is to model the RSS sequence signatures of indoor path by an undirected *trace-graph* model. Each vertex in the graph models the RSS sequence of a path segment. Through such a method, the trace-graph can be trained by a user traversing the indoor paths once to capture RSS signatures during movement, which overcomes the path combinatorial explosion problem. The random RSS missing problem is addressed by a collaborative filter, to smooth the measured RSS values, and to fill the missing RSS values. Based on the trace-graph, a *sub-sequence dynamic time warping (SDTW)* algorithm is proposed to conduct sequence matching in online locating. It finds a subsequence in the trace-graph which best matches the online measured RSS sequence in a moving window. SDTW tolerates the moving speed differences between the offline phase and the online phase by sequence warping [18] instead of point-to-point matching. More specifically, our contributions include:

1) A WarpMap model represented by undirected graph $\mathbf{G}=(\mathbf{V},\mathbf{E})$, where $\mathbf{V}$ represents RSS signatures on path segments and $\mathbf{E}$ represents adjacency of path segments.
2) A fast calibration process to build $\mathbf{G}$ by the user traversing indoor routes for once, and a collaborative filtering method to smooth the noisy RSS data.
3) In online phase, a potential path extraction algorithm to extract from $\mathbf{G}$ the potential paths that the target maybe undergoing, based on the real-time detected APs. It prepares a small *candidate sequence set (CSS)* for location determination.
4) A *subsequence dynamic time warping (SDTW)* algorithm to find a subsequence in CSS which has the least warping distance to the online measured RSS sequence within a short time window.
5) Investigations of different warping distance functions and different CSS selection methods.
6) System implementation and extensive experiments in office environments, which verified the efficiency and accuracy improvement than the state-of-the-art (Point radio-map + KNN/particle filter) locating method.

The rest of this paper is organized as follows. Background and problem model are introduced in Section II. The trace-graph construction is introduced in Section III. Online locating by subsequence dynamic time warping is introduced in Section IV. Performance evaluation by a prototype system is presented in Section V. The paper is concluded in Section VI.

II. Background and Problem Model

A. Background

In traditional point radio-map, the radio signatures of a set of location points in the area-of-interest (AOI) are offline calibrated. Let $\mathbf{L} = \{l_1, l_2, \cdots, l_K\}$ denote the calibrated location points. The radio-map can be represented by $\mathbf{M_p} = \{\mathbf{y}_{l_1}, \mathbf{y}_{l_2}, \cdots, \mathbf{y}_{l_K}\}$, where $\mathbf{y}_{l_i}$ is the vector representing the radio-signatures of location l_i. It is generally represented by the mean signal strength vector of different WiFi access points (APs) seen at that location.

In online locating phase, when a target detects its RSS signature $\mathbf{y_t}$ at time t, its location is calculated by finding a location in $\mathbf{M_p}$, whose RSS signature matches best with the online measured RSS signature.

$$l_t^* = \underset{(l):\mathbf{y}_l \in \mathbf{M_p}}{\arg\min} \ \mathrm{Dist}\left(\mathbf{y_t}, \mathbf{y}_l\right). \tag{1}$$

where $\mathrm{Dist}(\mathbf{y_t}, \mathbf{y}_l)$ measures the distance between $\mathbf{y_t}$ and $\mathbf{y}_l$; $\mathbf{y}_t$ and $\mathbf{y}_l$ are generally trimmed or expanded to be the same size vector for distance calculation [19].

A notable problem for point-type matching is that the motion continuity of the user and the indoor path constraint information are not efficiently included in the point matching process. As a result, l_t^*, i.e., the estimated location of a target at time t, may depart far away from l_{t-1}^* due to random RSS noises. Although fusion algorithms such as Bayesian filter [2] or Particle filter [14] can help to correct the location estimation, they need either additional information or additional computing to post-process the matching results. Instead of post-processing, this paper proposes a sequence-matching approach to improve the radio-map matching itself, which can further collaborate with fusion techniques.

B. Sequence Matching

The idea of sequence matching is to utilize the RSS sequence measured by a user along a path, which is treated as the path's radio signature. Let $\mathbf{\Gamma}_i$ be a path. Let $\mathbf{Y}^i = (\mathbf{y}_1^i, \cdots, \mathbf{y}_m^i)$ indicate the path signature, which is the RSS vector sequence captured by a user when he/she walks through $\mathbf{\Gamma}_i$. $\mathbf{y}_j^i$ is the j-th sample of RSS on path $\mathbf{\Gamma}_i$. Let $\mathbf{\Gamma} =$

$\{\mathbf{\Gamma}_1, \cdots, \mathbf{\Gamma}_M\}$ be the path set and $\mathbf{Y} = \{\mathbf{Y}^1, \cdots, \mathbf{Y}^M\}$ be the signature set. M is the number of sequences.

Then in the online phase, a target measures RSS sequences within a moving time window $\{t-w+1, \cdots, t\}$, where t is the current time. This forms a online measured length-w RSS sequence $\mathbf{X}_t = (\mathbf{x}_{t-w+1}, \cdots, \mathbf{x}_t)$. Based on it, sequence matching is carried out to find the optimal match between $\mathbf{X}_t$ and a subsequence $\mathbf{Y}^{s^*}_{[a^*,b^*]}$ from one sequence in set $\mathbf{Y}$.

$$(s^*, a^*, b^*) = \underset{(s,a,b):\mathbf{Y}^s \in \mathbf{Y}, 1 \le a \le b \le m_s}{\arg\min} \text{Dist}\left(\mathbf{X}_t, \mathbf{Y}^s_{[a,b]}\right). \quad (2)$$

where $\mathbf{Y}^s_{[a,b]}$ is a subsequence of $\mathbf{Y}^s \in \mathbf{Y}$. a and b are the start point and the end point of the subsequence$\mathbf{Y}^s_{[a,b]}$; $\mathbf{X}_t$ is matched to a subsequence because $\mathbf{X}_t$ is shorter and $\mathbf{Y}^s$ is generally longer. m_s is the length of $\mathbf{Y}^s$. Dist $\left(\mathbf{X}_t, \mathbf{Y}^s_{[a,b]}\right)$ is a function to measure the distance between $\mathbf{X}_t$ and $\mathbf{Y}^s_{[a,b]}$. The real-time location of the target is given by $\mathbf{\Gamma}_{s^*}(b^*)$, which is the end point of the best matched subpath. Since the motion continuity and the path information are implicitly modeled by the sequence signature, they are efficiently utilized in sequence matching, which provides better accuracy and robustness (Section V).

C. Key Problems

However, above ideal idea of sequence radio-map faces practical challenges. *In training phase*, 1) it is infeasible to calibrate and store the signatures of all indoor paths because there are innumerable number of paths by combination; 2) the detectable AP list changes over time and the phenomenon of RSS miss-of-detection is very general when the user is moving. Therefore, the sequence signatures maybe different even if a user moves on the same path at different times.

In locating phase, 1) the moving speed of user may differ from that of the training phase; 2) different moving directions result in different RSS sequences. Therefore, sequence matching need to tolerate both the speed and the direction differences.

These issues are addressed in Section III and Section IV by trace-graph and SDTW respectively. For clarity of presentation, the notations used in this paper are listed in Table I.

TABLE I
LIST OF NOTATIONS AND EXPLANATION

Notations	Explanation
$\mathbf{\Gamma}_i$	a path i
$\mathbf{Y}^i$	the RSS vector sequence collected on path $\mathbf{\Gamma}_i$
$\mathbf{Y}^i_j$	the jth RSS vector in sequence $\mathbf{Y}^i$
$\mathbf{X}_t$	RSS sequence in collected online moving window
$\mathbf{Y}^s_{[a,b]}$	subsequence of $\mathbf{Y}^s$ from $\mathbf{Y}^s_a$ to $\mathbf{Y}^s_b$
$\mathbf{C_t}$	the candidate sequence set generated by $\mathbf{X}_t$
m_s	length of RSS sequence $\mathbf{Y}_s$
h_s	height of RSS sequence $\mathbf{Y}_s$
w	length of moving window
N	the number of total APs in AOI
N_{l_i}	the number of APs detectable at a location l_i

III. TRAINING: TRACE-GRAPH

The RSS sequence collection, filtering and the trace-graph construction are introduced in this section.

A. RSS sequence collection and filtering

Note that the number of path segments, which are segments of path between crossing points are limited. So the idea of sequence fingerprint calibration is to firstly collect RSS sequences to cover these path segments; then divide the collected sequences into segments and connect the segments to form a trace-graph. After that, the RSS sequences of any desired path can be generated from the trace-graph, even if it is not calibrated.

1) RSS Sequence Collection: RSS sequence collection can be carried out in many ways using interactive smart phones or indoor navigation systems by inertial sensors [20]. This work used an interactive way, in which an APP was developed on the mobile phone. It renders the floor-plan and gets the scaling factor of the floor-plan, so that each pixel on the screen maps to a location in the AOI. In order to calibrate a path, a user clicks on the map to firstly mark the path. Several clicks on the screen will characterize a path. Then the user walks along the specified path to collect RSS sequence. Since the WiFi-scanner in the APP takes RSS samples in equal intervals, the positions where RSSs are measured on the path can be calculated by assuming the user is moving at constant speed on each segment. A RSS sequence on a path will be a series of $\langle$location, RSS-vector$\rangle$ pairs, which is a mapping from $\mathbf{\Gamma_i}$ to $\mathbf{Y}^i$.

2) Data Clean and Collaborative Filtering: In RSS sequence collection, since each location is generally traversed once, the measured RSS is noisy and unreliable. Notable problems are: 1) the numbers of APs, i.e., the RSS vector sizes at different points on the same path are different; 2) the RSS values of some APs are randomly miss-of-detection and the measured values are noisy. Fig. 2 shows an example of collected raw RSS sequence data. The rows indicate the AP No. and the columns are time index. They form a data matrix, in which data are noisy and many values are missed.

i. Regularize the vector length: We post-processed the collected RSS sequences to regularize the RSS vectors on the same path to the same length. Suppose m_i samples are collected on a path $\mathbf{\Gamma_i}$. Let $\mathbf{B}_{i,j}$ indicate the set of APs detected at the jth sample on $\mathbf{\Gamma_i}$. Then let $\mathbb{B}_i = \mathbf{B}_{i,1} \cup \mathbf{B}_{i,2} \cdots \mathbf{B}_{i,m_i}$ denote the union set of the APs which has been detected on $\mathbf{\Gamma_i}$. Then the size of RSS vectors on $\mathbf{\Gamma_i}$ are all expended to be $h_i = |\mathbb{B}_i|$, which is called **the height of sequence** $\mathbf{Y}_i$. The RSS values of undetectable APs are kept empty for later filling. Thus, the result RSS values on path $\mathbf{\Gamma_i}$ is a size $h_i * m_i$ matrix with noisy and missing values.

ii. Polynomial fitting is not enough. For clarity of presentation, we omit the path index and present the matrix by a $n*m$ matrix $\mathbf{A}$. An intuitive way to fill the missing data is to apply polynomial fitting in each row to train a polynomial function of k coefficients and predict the missed values by the polynomial function. But polynomial fitting using only the row data has

AP No.\Time	1	2	3	4	5	6	7	8	9	10	11	12	13
1			−91	−89	−86	−86	−86	−86	−86	−86			
2		−78	−78	−83	−83	−78		−78	−79	−78	−80	−75	−74
3	−63	−63	−70	−69	−63	−63	−67	−73	−71	−69	−63	−66	−71
4					−76	−76	−76	−76		−81	−81		
5	−80			−80	−80	−81	−81	−81	−81	−85			
6	−73		−72	−73	−69	−76	−73	−79	−76	−72	−72	−72	−72
7						−90	−87	−87	−87	−85			
8	−72	−76	−77	−77	−77	−80	−80	−84	−76	−83	−78	−78	−78
9			−90	−90	−90	−91	−91	−91	−91	−91	−91	−91	−91
10	−81								−86	−86	−86	−86	
11	−84	−85	−84					−85	−85	−84	−84	−81	−81

Fig. 1. Example of collected raw RSS sequence data

a problem. In some rows, which have data only in the middle part, the predicted RSS values at two ends maybe too large or too small, which are inaccurate. To solve this problem, we exploit collaborative filtering to fill the data gradually using both the row and the column information.

iii. Recursive Collaborative Filtering. Collaborative Filtering (CF) [21] is a well known technique to exploit data correlation of to predict unknown data . In this problem, data correlations over time (in row), and across APs (in column) are both considered.

The routine is given in Algorithm 1. For a missing value, one prediction value is calculated in row by polynomial fitting; another is calculated in column by a weighted interpolation based on the APs' RSS sequence similarity. If these two predicted values have difference smaller than a threshold, the average value of these two predictions will be filled to the empty place. Otherwise, the value will be filled in the next round based on the updated matrix. This process repeats a fixed number of times. The left empty values will be filled by polynomial fitting.

Algorithm 1: Recursive Collaborative Filter

FOR round =1: T
  Update polynomial model of each row;
  FOR each empty value $A_{i,k}$:
    $\hat{A}_{i,k,1} \leftarrow$ poly-fit by row i;
    $\hat{A}_{i,k,2} \leftarrow$ weighted interpolation by column k;
    IF $|\hat{A}_{i,k,1} - \hat{A}_{i,k,2}| <$ Threshold
      $A_{i,k} = (\hat{A}_{i,k,1} + \hat{A}_{i,k,2})/2$
    ENDIF
  ENDFOR
ENDFOR

In line 5 of Algorithm 1, i.e, in the column-based interpolation, Cosine similarity is used to evaluate the similarity between two APs' RSS sequences. Let A_i, A_j denote row i and row j of matrix A, their similarity is evaluated by.

$$S(A_i, A_j) = \frac{(A_i \cdot A_j)}{\|A_i\| \times \|A_j\|}. \tag{3}$$

Then to predict $A_{i,k}$, top-K most similar rows of A_i will be selected, denoted by set $\mathbf{K}$. Their kth column data is used to predict $A_{i,k}$:

$$\hat{A}_{i,k,2} = \frac{\sum_{j \in \mathbf{K}} S(A_i, A_j) \times A_{j,k}}{\sum_{j \in \mathbf{K}} S(A_i, A_j)} \tag{4}$$

B. Construct Trace-graph

The trace-graph is constructed after RSS sequence collection and filtering. The crossing points of these paths are calculated, by path locations on the map. These crossing points divide the the collected RSS sequences into a set of segments. Suppose there are N segments. Each segment is composed by a small set of RSS sequence. We treat each segment as a vertex, denoted by V_i, then an undirected trace-graph can be generated as $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V} = \{V_i, i = 1, 2, \cdots, N\}$ is the segment set; $\mathbf{E} = \{E_{i,j}\}$ represents the connections of these segments. $E_{i,j} = 1$ if segment V_i and V_j share a common crossing point. The graph is stored as sequence-type radio-map.

Remark 1: (scalability) Note that two connected segments maybe calibrated from different training paths. Since the AP union sets in the two paths maybe different, the RSS vectors in two connected segments may have different sizes. This will not affect the later candidate sequence set generation and the online locating process. An advantage of storing trace-graph in this way is to save the storage space and to make the trace-graph scalable for large-area applications. It doesn't need to find the complete AP set of the AOI and doesn't need to regularize all RSS vectors to the same size.

Proposition 1 (degradability): The trace-graph can be degraded to the traditional point-type radio-map by setting the lengths of all vertex segments to 1 and setting the online moving window size to 1.

This is identical to the case when the point-type radio-map is constructed by collecting RSS sequence, which is suitable to the situation that the AOI is a large open area.

IV. Online Locating: Dynamic Sequence Warping

After trace-graph construction, the online locating is carried out by three steps: 1) online RSS sequence collection in a moving window; 2) candidate RSS sequence extraction from the trace-graph; 3) subsequence dynamic time warping for location determination. Notable requirements for online locating is the computing efficiency and the tolerance for the variances of users' moving speeds and moving directions.

A. Online RSS Sequence Collection using a Moving Window

In the online phase, as the target is moving, it captures RSS periodically and stores the measured RSS vectors into a moving window. Let $\mathbf{X}_t = (\mathbf{x}_{t-w+1}, \cdots, \mathbf{x}_t)$ denote the RSS sequence collected in the moving window and w be the window size. x_t is the latest collected RSS sample. Let $\mathbb{B}_x$ be the union set of APs in the moving window. Then the RSS vectors in the moving window are regularized to the length of $n_x = |\mathbb{B}_x|$. Collaborative filtering is then applied to filter the RSS vectors in the moving window. Based on $\mathbb{B}_x$, i.e., the AP list detected in the moving window, the candidate sequence set (CSS) where the target maybe currently locates is extracted from $\mathbf{G}$. This prepares the candidate sequences for online locating. It contains two steps: CVS extraction and CSS generation.

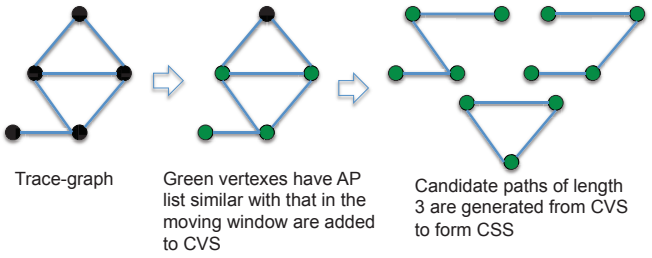

Fig. 2. Generate CVS and CSS from trace-graph

B. Candidate Vertex Set Extraction

The first step is to extract all vertices who has similar AP list with $\mathbf{X}_t$. For every vertex in $\mathbf{G}$, let $\mathbb{B}_i$ denote the AP list of the vertex i, Jaccard similarity score is calculated between $\mathbb{B}_i$ and $\mathbb{B}_x$:

$$J(\mathbb{B}_i, \mathbb{B}_x) = \frac{|\mathbb{B}_i \cap \mathbb{B}_x|}{|\mathbb{B}_x|} \tag{5}$$

Vertex i is added to the candidate vertex set (CVS) if $J(\mathbb{B}_i, \mathbb{B}_x) >$ Threshold, which means the detected AP list in $\mathbb{B}_i$ cover most of the detected APs in $\mathbb{B}_x$. In implementation, we set the Threshold to 0.7.

C. Candidate Sequence Set Generation

Then the candidate sequence set (CSS) is generated based on the vertices in CVS, which online builds the possible sequences that the target may be moving on. Since the moving window is short, it is unnecessary to generate long sequences. This heuristic can limit the size of CSS set. In implementation, we generate possible sequences containing at most c vertices, where $c = 3$ or 4. A direct way for CSS generation is to let each vertex in CVS to conduct Breadth-First-Search c steps to find paths originated at that vertex with length at most c, and then merge the repeated paths returned by different CVS. The remained distinct paths will form the CSS for online locating. Fig.2 illustrates the process of CVS extraction and CSS extraction.

Algorithm 2: *CSS Generation*
Input: $\mathbf{G} = (\mathbf{V}, \mathbf{E}), c$, CVS set.
Output: CSS containing at most c Vertices.
FOR each vertex v in CVS
· Breadth-First-Search c steps to find paths originated at vertex v with length no larger than c
ENDFOR
· Merge the returned paths of different CVS and delete the repeated paths.
· Return the RSS sequences on the remained distinct paths.

We denote the returned CSS by $\{\mathbf{Y}^s : s = 1, \cdots, L\}$, where L is the number of candidate sequences. For each sequence, the union set of AP list is calculated and the vectors in each sequence are expended to have the same height denoted by h_s.

D. Subsequence Dynamic Time Warping

Since $\mathbf{X}_t$ is short and the sequences in the CSS are generally long, the goal of matching is to find in the CSS set a particular sequence $\mathbf{Y}^{s^*}$, whose subsequence $\mathbf{Y}^{s^*}_{[a^*,b^*]}$ satisfies the objective equation (2). The end point of the subsequence $\mathbf{\Gamma}_{s^*}(b^*)$ is the location of the target. This is carried out by subsequence dynamic time warping (SDTW).

1) Dynamic Time Warping: SDTW is an extension of Dynamic Time Warping (DTW) [18], an algorithm for measuring similarity between two temporal sequences which may vary in time or speed. Consider two sequences $X = (x_1, \cdots, x_n)$ and $Y = (y_1, \cdots, y_m)$. DTW calculates a warping path $P = (p_1, \cdots, p_l)$ where $p_k = (i_k, j_k)$; $i_k \in [1 : n]$; $j_k \in [1 : m]$; and $k \in [1 : l]$ that satisfies the following alignment conditions:

$$\begin{cases} p_1 = (1,1), p_l = (n,m) \\ p_{k+1} - p_k \in \{(0,1),(1,0),(1,1)\}, \forall k \in [1 : l-1] \end{cases} \tag{6}$$

The first condition enforces the first elements of X and Y as well as the last elements of X and Y are aligned to each other. The second condition reflects no elements in X and Y can be omitted and there are no replications in alignment. An alignment satisfies above conditions is called a warping path. The *warping distance* between X and Y is defined as the summation of local distances along the warping path.

$$\text{DTW}(X, Y) = \sum_{k=1}^{l} d(x_{i_k}, y_{j_k}). \tag{7}$$

The warping path is not unique. Based on (7), let $X(1:i) = (x_1, \cdots, x_i)$ for $i \in [1:n]$ and $Y(1:j) = (y_1, \cdots, y_i)$ for $j \in [1:m]$. Then an warping distance matrix $D(i,j)$ is defined for evaluating all warping paths [18]:

$$D(i,j) = \text{DTW}\left(X(1:i), Y(1:j)\right) \tag{8}$$

The optimal warping path, i.e., P^* that minimizes $\text{DTW}(X, Y)$ can be calculated by dynamic programming [18].

2) Subsequence Dynamic Time Warping (SDTW): SDTW relaxes the boundary condition in (6). It allows X match a subsequence $Y_{[a,b]} = (y_a, y_{a+1}, \cdots, y_b)$ $(1 \le a \le b \le m)$ of Y [18], such that

$$(a^*, b^*) = \underset{(a,b):1 \le a \le b \le m}{\arg\min} \text{DTW}(X, Y_{[a,b]}). \tag{9}$$

But the locating problem has some differences from the traditional SDTW.

i) It needs only to determine the end point of the matched subsequence, i.e., b^* without the need to determine the start point a^*. This feature is utilized to design an efficient subsequence matching algorithm which doesn't need the cost of back-trace by dynamic programming, as detailed in Algorithm 3.

ii) It needs to search over multiple candidate sequences $\{\mathbf{Y}^s : s = 1, \cdots, L\}$ to find the subsequence with the overall minimum warping distance.

iii) It needs to find in $\mathbf{X}_t$ several representative APs (with strongest RSS values) and extract corresponding RSS sequences from both $\mathbf{X}_t$ and $\mathbf{Y}_s$, so that the sequence matching can be conducted between sequences of the representative APs, which is verified as an efficient way to avoid noises from weakly detectable APs in point-type radio-map [19].

iv) We also need to consider the situation that the user may walk in the reversed direction of the training direction of the candidate sequences. These problems are addressed in Algorithm 3.

Algorithm 3: *SDTW for Sequence-Based Locating*

STEP1: Representative AP Set Selection from $\mathbf{X}_t$

· Find in $\mathbf{X}_t$ the top n APs with the strongest mean RSS.

· Extract the RSS sequences of representative APs from $\mathbf{X}_t$ and replace $\mathbf{X}_t$ by the extracted represented sequences.

STEP2: Extract Representative Sequence for $\{\mathbf{Y}_s, s=1,\cdots,L\}$

· Extract the RSS sequences of representative APs from $\mathbf{Y}^s$. In case an representative AP is not in $\mathbf{Y}^s$, the corresponding row is filled by a very week signal strength, i.e., -95dbm.

· Replace $\mathbf{Y}^s$ by the extracted representative sequences.

STEP3: SDTW between $\mathbf{X}_t$ *and* $\{\mathbf{Y}_s, s=1,\cdots,L\}$

· Set the boundary conditions of warping distance matrix:

$$\begin{aligned} D_s(i,1) &= \sum_{k=1}^{i} d(x_k, y_{1,s}), \quad i \in [1:n] \\ D_s(1,j) &= d(x_1, y_{j,s}), \quad j \in [1:m_s] \end{aligned} \tag{10}$$

· Then the warping distance matrix is produced as:

$$D_s(i,j) = \min\{D_s(i\text{–}1, j\text{–}1), D_s(i, j\text{–}1), D_s(i\text{–}1, j)\} + d(x_i, y_{j,s}) \tag{11}$$

· The matched end point between $\mathbf{X}_t$ and $\mathbf{Y}^s$ is determined by D_s:

$$b_s^* = \underset{b_s \in [1:m_s]}{\arg\min} \; D_s(n, b_s), \tag{12}$$

· The optimal warping distance with $\mathbf{Y}^s$ is given by:

$$D(\mathbf{X}_t, \mathbf{Y}^s) = D(n, b_s^*) \tag{13}$$

STEP4: Revert sequence $\mathbf{X}_t$ *to match with* $\{\mathbf{Y}_s, s=1,\cdots,L\}$

· Reverse the direction of $\mathbf{X}_t$ and recalculate (10) - (13) to obtain $\overleftarrow{b_s^*}$ and $\overleftarrow{D}\left(n, \overleftarrow{b_s^*}\right)$ for $s=1,\cdots,L$.

STEP5: Determine the target location and the minimum warping distance

· For all the candidate sequences, the overall best matched sequence is the one with the overall least warping distance:

$$s^* = \underset{s \in [1:L]}{\arg\min} \; \min\left\{D(n, b_s^*), \overleftarrow{D}\left(n, \overleftarrow{b_s^*}\right)\right\} \tag{14}$$

· The location of the target is determined as the end matching point in the matched subsequence that has the overall least warping distance.

Note that in (12), since the ending point and the optimal warping distance $D(n, b_s^*)$ have already been provided by (12), it is not necessary to calculate a_s^* by reversely tracing. This saves the cost of dynamic programming than the traditional SDTW [18].

Fig.3 further illustrates the warping distance matrix in Step 3. The values represented by "+" is given as the boundary values. They produce the other entries of distance matrix by (11). After getting the matrix, the entry with the smallest distance in the nth row can be found, i.e., the star. Its column index is b_s^*, and its value is $D(n, b_s^*)$. In locating problem, we don't need to back-trace to find a_s^*, and the warping path (represented by the dashed line), although they have already been determined by the distance matrix. This saves computation cost than traditional SDTW [18]. It is easy to verify that the computation cost is $O(mn)$ in this step. The overall computation cost is $O(Lmn)$, which is efficient.

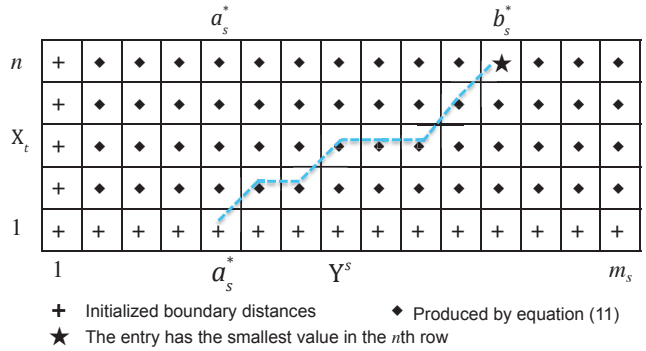

Fig. 3. Illustration of the warping distance matrix in SDTW.

E. Different distance functions

We also investigated different distance functions, i.e., $d(x_{i_k}, y_{j_k})$ to check their impacts to the locating accuracy. For different functions were investigated.

1) Euclidean distance: $d_{xy} = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2}$
2) Cityblock distance: $d_{xy} = \sum_{k=1}^{n} |x_k - y_k|$
3) Chebyshev distance: $d_{xy} = \max_k |x_k - y_k|$
4) Cosine distance: $d_{xy} = \frac{dot(x,y)}{\|x\| \times \|y\|}$

In Section.V-D, we will concretely analyze the effect of basic warping distance functions on locating accuracy for SDTW.

V. Implementation and Performance Evaluation

A. Experiment setup

1) Implementation: A sequence-based location APP was developed by Android platform and the location determination is carried out by a location server. The App carries out route calibration, RSS traces collection, data transmission to server, and location rendering in the online phase. The server constructs the trace-graph, generates CSS based on real-time detected AP list, and conducts online locating.

2) Experiment area: Experiments were conducted in NEC Laboratories China, located in Innovation Plaza No.1 of Tsinghua Science Park. The office area where experiments are carried out is about $1100m^2$ and more than 90 APs can be detected in the area. Experiments have last almost one month, by training trace-graph on one day, and conducted online locating using RSS sequences collected on other days, using different mobile phones, and tested by different users. In particular, for the training phase, the calibration was finished

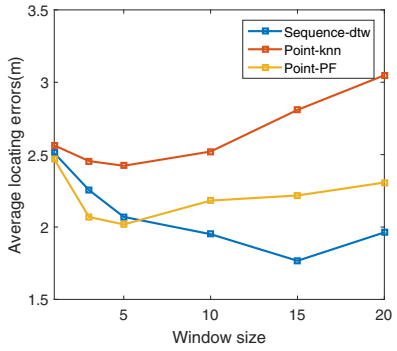

Fig. 4. Mean locating error as a function of moving window size w.

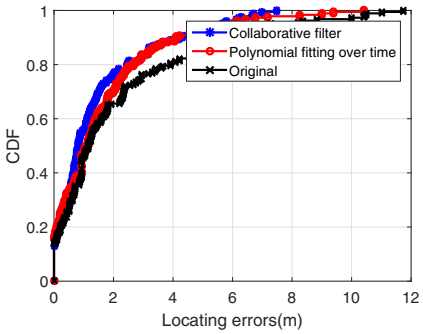

Fig. 5. CDF of locating error as a function of data clean methods.

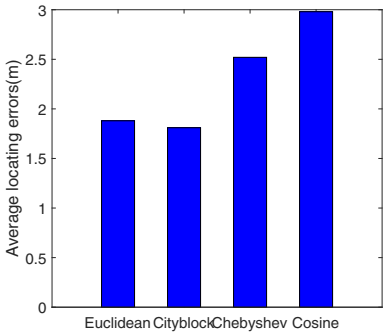

Fig. 6. CDF of locating error as a function of warping distance methods.

within almost 10 minutes by one person, by collecting 17 long RSS sequences to cover almost all indoor path segments.

3) Comparing methods: To compare with existing methods, two traditional point-based locating methods were implemented: 1) point radio-map using K-nearest Neighbor, abbreviated as *Point-KNN*; and 2) point radio-map using particle filter locating method, abbreviated by *Point-PF*. They are state-of-the-art of point-type radio-map locating method, which are implemented on the point-type radio-map, calibrated by dividing the collected RSS sequence into points.

4) Categories of experiments: Experiments are carried out in two categories: one is to investigate the effects of inner factors (e.g. the length of the moving window, different data clean methods); the other is to compare accuracy of the sequence-type locating method. In particular, we mainly evaluate:

1) The locating accuracy. We concretely analyzed the effect of four inner factors on SDTW, including the length of moving window, different data clean methods, different distance functions, the size of RSS vectors.After choosing the best values for above four factors, we compared general locating accuracy of SDTW with two pointed-based methods.
2) The locating robustness. We evaluated SDTW's robustness to the changes of environments, devices, walking speed, walking pattern and compared it with the point-based methods.

B. Accuracy vs. Length of Moving Window

We investigate the impact of moving window length on the locating accuracy of SDTW. Experiments were conducted by varying the length of moving window from 1, 3, 5, 10, 15, to 20. The average locating accuracies of point-KNN and point-PF were also calculated for comparing. Note that the average locating accuracies were calculated based on 135 locating tests. The average locating error as a function of the length of moving window w is plotted in Fig. 4. When w=1, SDTW is degraded to point-KNN. Compared with point-PF, SDTW works worse when $w < 5$. However, as w increases, SDTW shows better locating accuracy than both point-KNN and point-PF.

When the moving window length is larger than 15, the locating accuracy tends to decrease a little bit. The reason is that when w is long, the sequence matching accuracy decreases at some boundary locations, because it is hard to form a long sequence. So a window size between 10 and 15 is appropriate. Note that when RSS sampling rate is 1 Hz, user moves about 10-15 meters during this period, which is acceptable in real applications.

C. Accuracy vs. Data Clean Methods

In this section, the effect of data clean methods on the locating accuracy of SDTW is accessed. The collaborative filtering methods have been introduced in Section.III-A2. It was compared it with 1) polynomial fitting using purely row data, and 2) using the original data without filtering. Locating tests were conducted for each data clean method for 135 times and the CDF of locating errors were shown in Fig. 5. It can be seen that the collaborative filtering method improves the locating accuracy most efficiently, which is better than polynomial fitting based method. They both perform much better than using original data without filtering.

D. Accuracy vs. Different Distance Functions

In Section.IV-E, we have introduced four different distance functions for $d(x_{i_k}, y_{j_k})$. We applied them separately in SDTW algorithm and the average locating accuracies over 135 tests were evaluated . The results are shown in Fig. 6. Euclidean distance and Cityblock distance works better than the others. It is reasonable as these two distance functions give larger values (all four distance functions give positive values) for the same difference between two vectors. It means these two functions distinguish similar vectors better. Therefore, we use Euclidean distance for SDTW for it is also intuitive.

E. Accuracy vs. the Size of RSS Vectors

As discussed in Algorithm 3, in SDTW, n representative APs' sequences are extracted from both $\mathbf{X}_t$ and $\mathbf{Y}^s$ for online matching. We tested the locating accuracy by varying the size of n from 1 to 30 and evaluating the locating accuracy in each case. The results are shown in Fig.7. We find that SDTW works well only if the size of n is larger than 10. Therefore,

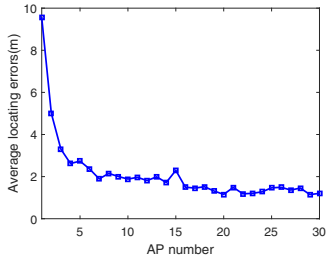

Fig. 7. Mean locating error as a function of the size of RSS vectors.

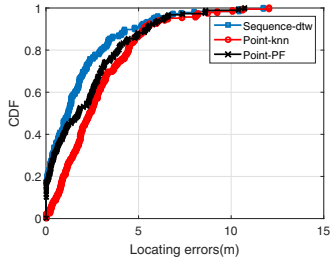

Fig. 8. CDF of locating errors for SDTW, point-KNN, point-PF.

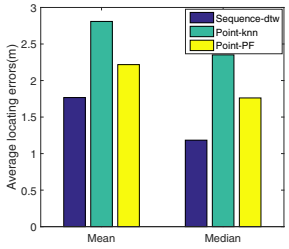

Fig. 9. Mean and median of locating errors.

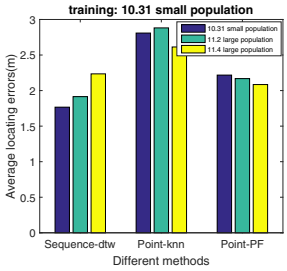

Fig. 10. Mean locating errors when environment changes.

in our experiments, n was chosen to be 10. That is, in each warping distance calculation, the RSS sequences of 10 APs of stronger values are used.

F. Accuracy vs. Point-type Radio-map Methods

After analyzing effect of four inner factors on locating accuracy, we (i) chose the size of moving window w=15, (ii) used collaborative filtering methods for data clean, (iii) used Euclidean distance as basic warping distance function and (iv) chose the size of representative AP set as 10. Then the locating accuracy of SDTW was compared with that of point-KNN and point-PF. The the cumulative distribution function (CDF) of locating error is shown in in Fig.8, which is the statistical results of 135 test points. It can be seen that SDTW works much better than both the two point-based methods. The mean and median of location errors for the same test sets are plotted in Fig.9. WarpMap reduces the average positioning error more than 20% than point-PF, which is the state-of-the-art of the point-based indoor locating. Further, SDTW method can still be further improved to collaborate with particle filter, which still has space for performance improvement.

G. Robustness vs. Environment Changes

In this section, we assess the effect of environment changes on the locating accuracy. It is evident that the calibration effort is the major cost of using fingerprinting method. If the locating performance can be robust to the environment changes, frequent recalibration will not be needed.

To test the impacts of environment changes, we firstly trained the undirected graph G on Oct. 31, when there were only a few people in the office in the morning. Then, we collected three sets of testing traces at three different times: 1) in the evening of Oct. 31 when there were few people in the office; 2) on Nov. 2 when many people are working; 3) at noon time of Nov 4 when many people were working and moving. SDTW, point-based KNN, point-based PF were evaluated and their mean locating errors are shown in Fig. 10. It can be seen that the three locating methods all keep reasonable robustness when the testing environments differ from the training environment. But overall, SDTW has much better locating accuracy than the other two point-based methods.

H. Robustness vs. Walking Speed and Walking Pattern

To evaluate SDTW's tolerance to the walking speed and walking pattern variances, two sets of experiments were designed.

The effects of different walking speeds to the locating accuracy were firstly evaluated. The training traces were collected by walking in almost constant moderate speed, about 1m/s. Then in the online phase, three speeds were tested for online locating: 1) high speed about 2.5m/s; moderate speed about 1m/s, and low speed about 0.3m/s. The CDF of locating errors for different experiments is shown in Fig.12. We find that when the training and testing traces are collected in the same speed, the locating error is the smallest. And when the speeds are different in the training and locating phases, the locating accuracy become worse, but SDTW still keeps a good accuracy.

Then we evaluated SDTW's robustness to different walking patterns. We conducted the experiments in which the tester moved in a walk-stop pattern, i.e., walks, stops, and then walks again repeatedly. We compare SDTW with an sequence-based equal-length matching algorithm in which the distance is calculated point by point between sequences without warping.

The matching result of an instance in SDTW is shown in Fig. 11. SDTW matches the moving window with indices 10-20 to a subsequence with indices 16-19 in CSS. The stopping sequence was efficiently warped. However, equal-length matching cannot warp the RSSs at the stopping time. Furthermore, the CDF of locating error in the series of walk-stop experiments is plotted in Fig. 13. It shows that SDTW and point-PF perform much better than the other two methods. We can also find the sequence-based equal-length matching algorithm works the worst and its sensitivity to walk patterns may be the main reason why sequence-type fingerprinting is not widely used before.

VI. Conclusion

This paper investigated feasibility, algorithms and performances of indoor locating by using sequence-type radio-map. It firstly proposed a trace-graph model to efficiently calibrate and store the RSS sequences signature of indoor paths, which can overcome the indoor path combinatorial explosion problem. A recursive collaborative filter has been developed for filling the missed value and smoothing the noisy values in

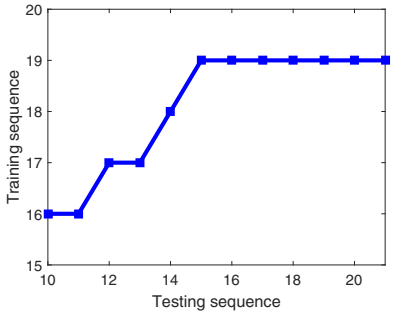

Fig. 11. Warping path of in an instance of walk-stop experiment.

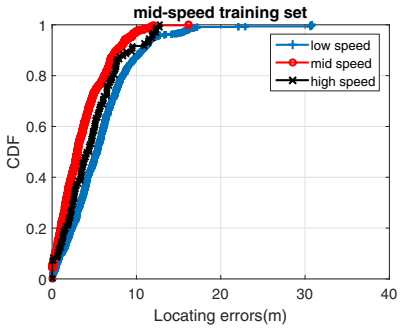

Fig. 12. CDF of locating error when locating speed changes.

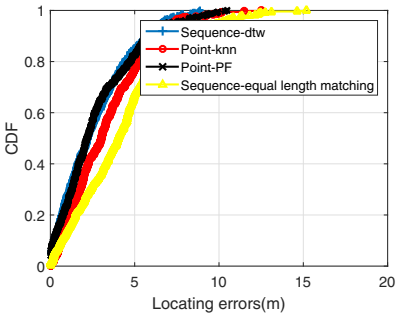

Fig. 13. CDF of locating error in walk-stop experiment.

the collected RSS sequence. Then in online locating phase, RSS sequences are collected by a target using a short moving window, based on which, the potential paths that the target maybe undergoing is extracted to form a candidate sequence set (CSS). Then, an efficient sub-sequence dynamic time warping algorithm was proposed to determine the location of the target, which can tolerate the differences of moving speeds, moving patterns, and moving directions between the training and locating phases.

Extensive experiments in office environment shows that WarpMap can be calibrated within ten minutes by one person in a 1100 m^2 office area. It reduces the mean locating error almost 20% than the traditional point-type locating method with particle filter. These features show that it is efficient and easy to use.

Since Warpmap works as a basic block for RSS fingerprint matching, it can be widely collaborated with fusion algorithms and additional information. In future work, we will exploit the fusion of Warpmap with particle filter, dead reckoning algorithm using inertial sensors, and digital floor map information. It can be foreseen that by combining these information, sequence-based indoor location can achieve better locating accuracy.

VII. Acknowledgement

This paper was partially supported by the Fundamental Research Funds for the Central University, and the Research Funds of Remin University of China, 2015030273; The National Natural Science Foundation of China, No. 61202360; the Hong Kong Scholars Program.

References

[1] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *IEEE INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, vol. 2, 2000, pp. 775–784 vol.2.

[2] I. Haque and C. Assi, "Profiling-Based Indoor Localization Schemes," *IEEE Systems Journal*, vol. Early Access Online, 2013.

[3] M. Youssef and A. Agrawala, "The horus wlan location determination system," in *MobiSys'05*, New York, NY, USA: ACM, 2005, pp. 205–218.

[4] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg, "Compass: A probabilistic indoor positioning system based on 802.11 and digital compasses," in *WiNTECH '06*. New York, NY, USA: ACM, 2006, pp. 34–40.

[5] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to war-drive: Unsupervised indoor localization," in *MobiSys '12*. New York, NY, USA: ACM, 2012, pp. 197–210.

[6] Y. Tian, R. Gao, K. Bian, F. Ye, T. Wang, Y. Wang, and X. Li, "Towards ubiquitous indoor localization service leveraging environmental physical features," in *INFOCOM'14*, Apr. 2014, pp. 55–63.

[7] R. Harle, "A Survey of Indoor Inertial Positioning Systems for Pedestrians," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1281–1293, 2013.

[8] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: Wireless indoor localization with little human intervention," in *Mobicom '12*. New York, NY, USA: ACM, 2012, pp. 269–280.

[9] S. J. Pan, J. T. Kwok, Q. Yang, and J. J. Pan, "Adaptive Localization in a Dynamic WiFi Environment Through Multi-view Learning," in *AAAI'07*. Vancouver, British Columbia, 2007, pp. 1108–1113.

[10] J. Yin, Q. Yang, and L. Ni, "Learning Adaptive Temporal Radio Maps for Signal-Strength-Based Location Estimation," *IEEE Transactions on Mobile Computing*, vol. 7, no. 7, pp. 869–883, 2008.

[11] J.-O. Nilsson, I. Skog, P. Handel, and K. Hari, "Foot-mounted INS for everybody - an open-source embedded implementation," in *Position Location and Navigation Symposium* PLANS'12*, 2012 IEEE/ION*, Apr. 2012, pp. 140–145.

[12] M. B. Kjærgaard, "A taxonomy for radio location fingerprinting," in *LoCA'07*. Berlin, Heidelberg: Springer-Verlag, 2007, pp. 139–156.

[13] Z. l. Wu, C. h. Li, J. K. y. Ng, and K. R. p. h. Leung, "Location Estimation via Support Vector Regression," *IEEE Transactions on Mobile Computing*, vol. 6, no. 3, pp. 311–321, Mar. 2007.

[14] J. Hightower and G. Borriello, Particle Filters for Location Estimation in Ubiquitous Computing: A Case Study, in *UbiComp'04*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 88–106. 2004

[15] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, "Lightweight map matching for indoor localisation using conditional random fields," in *IPSN'14*, Apr. 2014, pp. 131–142.

[16] X. Geng, Y. Wang, H. Feng, and Z. Chen, "Hybrid radio-map for noise tolerant wireless indoor localization," in *ICNSC'14*, Apr. 2014, pp. 233–238.

[17] Z. Yang, Y. Wang, and L. Song, "Adamap: Adaptive radiomap for indoor localization," in *ADHOC-NOW'15*, Athens, Greece, 2015, pp. 134–147.

[18] S. Z. Li and A. Jain, Eds., *Encyclopedia of Biometrics*. Boston, MA: Springer US, 2009, ch. Dynamic Time Warping (DTW), pp. 231–231.

[19] A. Farshad, J. Li, M. K. Marina, and F. J. Garcia, "A microscopic look at WiFi fingerprinting for indoor mobile phone localization in diverse environments," in *IPIN'13*, Oct. 2013, pp. 1–10.

[20] J.-O. Nilsson, A. K. Gupta, and P. Händel, "Foot-mounted inertial navigation made easy," in *IPIN'14*, vol. 27, 2014, p. 30th.

[21] X. Su, T. M. Khoshgoftaar, X. Su, and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," *Advances in Artificial Intelligence*, vol. 2009, p. e421425, Oct. 2009.

[22] M. Müller, *Information Retrieval for Music and Motion*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[23] H. Sakoe, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, pp. 43–49, 1978.