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

Xuehan Ye, Yongcai Wang
CS, Renmin University
Beijing China
ycw@ruc.edu.cn

Wei Hu, Lei Song
IIIS, Tsinghua University
Beijing, China
huwei12@mails.tsinghua.edu.cn
leisong03@gmail.com

Deying Li
Renmin University
Beijing, China
deyingli@ruc.edu.cn

Abstract—**Radio-map based method has been widely used for indoor location and navigation, but remained 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 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 accurate and efficient indoor location method by dynamic warping in sequence-type radio-map. Its distinct features include 1) an undirected graph model (tracegraph) for storing smoothed RSS sequence in a sequence-type radio-map; and 2) a sub-sequence dynamic time warping (SDTW) algorithm for on-line locating. In particular, in trace-graph, each vertex represents a RSS sequence of a indoor path segment. The graph is constructed off-line by trimming and filtering the noisy RSS sequences collected by user via traversing the indoor paths once. It embeds the path constraint and motion continuity information directly into the radio-map model, while overcoming the indoor path combinational explosion problem. Based on the trace-graph, a subsequence dynamic time warping algorithm (SDTW) for on-line locating was proposed. It includes steps of i) candidate sequence set extraction; ii) warping distance calculation; and iii) location determination by the end-point of the best matched sub-sequence. Sequence warping can tolerate random RSS disparities at discrete points and and can also handle the moving speed differences in on-line and off-line phases. The impacts of different warping distance functions, RSS preprocessing techniques, and fusion extensions were investigated. Property analysis and extensive experiments in office environments verified the efficiency and accuracy of WarpMap. It can be calibrated within several minutes for $1000m^2$ area; provides robust and accurate locating in different settings, and overall nearly 20% accuracy improvements than the state-of-the-art radio-map method.

I. Introduction

Radio-map locating method, which characterizes each location-of-interest by the radio signal strength (RSS) signature, has attracted great attentions for indoor positioning and navigation. It maps the signal space to physical space to tame the indoor radio propagation dynamics; becomes popular nowadays for the broad availability of WiFi coverage and free-of-charge of radio RSS measurement. 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 [1], [2].

More specifically, routine of radio-map method generally has an offline site survey and an online locating phases. The offline site survey is to calibrate the received signal strengths (RSS) at different known locations, which builds a *radio-map*. 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 to determine the real-time location of the target.

Despite of the advantages, several practical issues must be considered for practical applications: (i) it is very laborious to calibrate a fine-grained radio-map point-by-point, especially for large application areas; (ii) the point-type matching at discrete times and locations is sensitive to random online RSS noises and environment dynamics. To overcome these problems, previous work 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 [], which exploited environment signature including magnetic fluctuation, illumination intensity at specific spots as internal landmarks [], and used dead reckoning by mobile phones [] to track the inertial landmarks to conduct locating, so as to avoid the manually radio-map calibration. Another major approach exploited automatic labeling [] [], 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 [] []. 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 [] [].

The inaccuracy of radio-map model may also degrade the performance of locating accuracy.

In the other aspect, for improving the locating accuracy and robustness, a major approach is to build fine-grained or probabilistic radio-map to preserve more information [] [] in the radio-map, but such kind methods requires higher calibration cost. Another major approach is to exploit Support Vector Machine (SVM) or K-Nearest Neighbor algorithms etc to tolerate random noises in the locating process [] []. Other approaches used information fusion techniques, which exploited the motion continuity information, floor map information to design Bayesian filter [], particle filter [], Hidden Markov Model [], and Markov Random Field models [] to narrow down the search space, and posteriorly improve the locating accuracy against noises. These methods depend on information fusion technique to utilize the motion constraint and the path constraint, for these information are not properly contained in the point-type radio-map model.

A more direct and promising way to ease the radio-map calibration and to improve the location robustness as well is to collect RSS sequence along indoor paths to build sequence-type radio-map, and to exploit the motion constraint and path constraint embedded in the sequential RSS signatures to 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 in online phase and offline phase. 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 [], the collected sequences are divided into points to construct point-type radio-map. The path structure and motion constraints are lost in the radio-map, which is however compensated by post-processing via information fusion, such as map-matching and particle filter using additional information [].

This paper revisits sequence matching in radio-map. It proposes *WarpMap* to tackle the sequence radio-map construction and online matching challenges. The key idea is to model the RSS sequence signatures of indoor path by an undirected *tracegraph* model. Each vertex in the graph models the RSS sequence of a path segment. By such a method, the tracegraph can be trained by user traversing the indoor paths once by capturing RSS signatures during movement. The random RSS missing problem is addressed by a collaborative filtering method, which on one hand smooths the measured RSS values, on the other hand complements the missing values. Based on the tracegraph, RSS sequence for each selected path can be easily generated. Then, a sub-sequence dynamic time warping (SDTW) algorithm is proposed to conduct sequence matching in online phase. It finds the best match from the RSS sequence measured online in a short moving window to a subsequence in the tracegraph. It tolerates the moving speed differences by warping [] instead of one-to-one matching. It includes a candidate sequence set extraction step and a location determination phase, which will be detailed in Section III. More specifically, our contributions include:

1) A WarpMap model represented by undirected graph $G = (V, E)$, where V represents RSS signatures on path segments and E represents adjacency of path segments.
2) A fast calibration process to build G by 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 G the potential paths that the target maybe undergoing, based on the list of 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) Property analysis to both the WarpMap construction and the online locating process and extensive experiments in office environments which verified the efficiency and accuracy of the WarpMap method.

The rest of this paper is organized as follows. Related work is summarized in Section **??**. The overview of sequence-type fingerprinting is given in Section **??**. The training and locating methods are detailed in Section III and IV respectively. Performance evaluation by a prototype system is presented in Section V. The paper is concluded in Section VI.

II. Problem Model

A. Sequence Radio-map

Let's consider indoor locating in an area of interest (AOI), denoted by A. In traditional point-type radio-map, a vector of mean signal strength values from different WiFi access points (APs) seen at a location l_i can be taken as the WiFi fingerprint for that location, as in []. Note that at that location, only a subset of APs can be detected. For clarity of presentation, let N be the size of the complete AP set; let N_{l_i} be size of AP subset at location l_i. Then the measured RSS vector at l_i can be expanded to a length-N vector, by filling the RSS values of $N - N_{l_i}$ undetected APs by e.g., $-100dbm$ indicating a very weak signal strength. Farshad et al. [] showed that using a smaller AP set with stronger RSS strength could improve positioning accuracy. We show it is more complex to construct and sensitive to AP set size in sequence-type fingerprint.

In sequence-type radio-map construction, instead of storing mean RSS values measured at discrete points the sequences of RSS vectors along different routes are intended to be measured and stored. Let Γ_i be a route a user is moving along. The RSSs are captured periodically during the user movement, so that a sequence $Y^{(i)} = (y_1^{(i)}, \cdots, y_m^{(i)})$ will be measured along

Γ_i, indicating m points on Γ_i. Note that $y_j^{(i)}$ indicates the RSS vector of the j-th point on Γ_i, which contains not mean values of RSS, but instant RSS measurements, so it is exposed to random RSS noises and random RSS miss-of-detection problems []. Other issues should also be considered: 1) the combinations of indoor paths are numerous, so it is not feasible to enumerating all Γ_i to store their sequence signatures; 2) even on a same path segment, different moving directions of user will result at different RSS vector sequences; 3) the set of detectable APs varies as the user is moving along a path.

These issues are addressed in Section III by proposing a trace-graph model $G = (V, E)$, composed by RSS sequence signatures on path segments, which can generate RSS sequence of any selected path. A collaborative filtering method is proposed to address the RSS noises, RSS miss-of-detection, and AP list varying problems. The impact of moving direction is addressed in Section IV by calculating warping distance in both forward and backward directions.

B. Online Locating by Dynamic Warping

In the online phase, the RSS sequences within a moving time window $\{t-w+1, t-w+2, \cdots, t\}$ are measured, where t is the current time. This forms a length-w RSS vector sequence $X_t = (x_{t-w+1}, \cdots, x_t)$. Then, sequence matching algorithms were designed to find an optimal match between X_t and a subsequence in the trace-graph G. The matching algorithm has two steps.

In the first step, based on the list of APs which are detected in X_t, a candidate sequence set (CSS) denoted by $\mathbf{C_t}$ is searched from G, which contains RSS sequences of potential paths that the target maybe on going. Note that the ith path in $\mathbf{C_t}$ is denoted by $\Gamma_i \in \mathbf{C_t}$. Its corresponding RSS vector sequence is denoted by $Y^i \in \mathbf{C_t}$. The length of Y^i is denoted by m_i.

In the second step, a subsequence dynamic time warping algorithm is proposed to find in $\mathbf{C_t}$ a subsequence $Y^{s^*}_{[a^*,b^*]}$, where

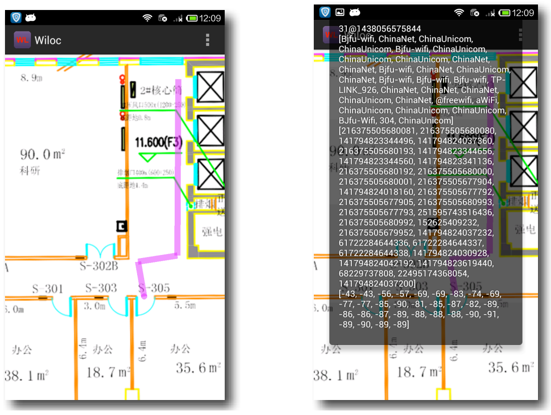
$$(s^*, a^*, b^*) = \operatorname*{arg\,min}_{(s,a,b):Y^s \in \mathbf{C_t}, 1 \le a \le b \le m_s} \text{Dist}\left(X_t, Y^s_{[a,b]}\right). \quad (1)$$

where $Y^s_{[a,b]}$ denotes the subsequence $\{Y^s_a, Y^s_{a+1}, \cdots, Y^s_b\}$ of the sequence Y^s; a and b are the start point and the end point of the subsequence; m_s is the length of $Y^{(s)}$. Dist $\left(X_t, Y^s_{[a,b]}\right)$ is a function to measure the warping distance between X_t and $Y^s_{[a,b]}$. We investigated warping distance functions including 2nd-norm, vector angle in Section IV. The location of the target is then given by $\Gamma_{s^*}(b^*)$, which is the end point of the best matched subsequence. It gives the real-time location of the target.

For clarity of presentation, the notations used in this paper are listed in Table I.

III. Trace-Graph Construction

The input of training problem include: 1) an area of interest (AOI), denoted by $\mathcal{A}$ with its floorplan; 2) a set of APs in the AOI without requiring their locations or total number.



```
31@1438056575844
[Bjfu-wifi, ChinaNet, ChinaUnicom,
ChinaUnicom, Bjfu-wifi, ChinaUnicom,
ChinaUnicom, ChinaUnicom, ChinaNet,
ChinaNet, Bjfu-wifi, ChinaNet, ChinaUnicom,
ChinaNet, Bjfu-wifi, Bjfu-wifi, Bjfu-wifi, TP-
LINK_926, ChinaNet, ChinaNet, ChinaNet,
ChinaUnicom, ChinaNet, @freewifi, aWiFi,
ChinaUnicom, ChinaUnicom, ChinaUnicom,
BJfu-Wifi, 304, ChinaUnicom]
[216375505680081, 216375505680080,
141794823344496, 141794824037360,
216375505680193, 141794823344656,
141794823344560, 141794823341136,
216375505680192, 216375505680000,
216375505680001, 216375505677904,
141794824018160, 216375505677792,
216375505677905, 216375505680993,
216375505677793, 251595743516436,
216375505680992, 152625409232,
216375505679952, 141794824037232,
61722284644336, 61722284644337,
61722284644338, 141794824030928,
141794824042192, 141794823619440,
68229737808, 22495174368054,
141794824037200]
[-43, -43, -56, -57, -69, -69, -83, -74, -69,
-77, -77, -85, -90, -81, -85, -87, -82, -89,
-86, -86, -87, -89, -88, -88, -88, -90, -91,
-89, -90, -89, -89]
```

Fig. 1. Snapshot of trace calibration and RSS collection App

The training phase is separated into three steps: 1) *trace calibration*; 2) *trace-graph generation* ; 3) *CTB generation*.

A. Interactive RSS trace calibration

By using smart phones, RSS trace calibration can be easily carried out, which is implemented by developing a trace calibration App. It performs three tasks: 1) rendering the floorplan, 2) defining routes by clicking on the floorplan shown on the screen, and 3) recording the RSS sequences as user moves along the defined routes.

More specifically, the App loads the floorplan and sets the scaling factor specified by the floorplan so that each pixel on the rendered map are mapped to a location in the AOI. Then for each route to calibrate, the user clicks on the map to mark the route. One route is often obtained by connecting several clicked points on the screen. Then the user walks along the specified routes to collect a RSS sequence. Since the WiFi-scanner in the App takes RSS samples in equal intervals, the concrete positions where RSSs are measured on the route can be calculated accordingly, assuming that the user is moving at constant speed on each designed segment. A trained RSS trace on a route will be a series of $\langle$location, RSS-vector$\rangle$ pairs. For a trained trace, let T_i be its trajectory and $Y^{(i)}$ be the corresponding RSS sequence. Note that the length of different traces maybe different and the vector length in each pair maybe different for WiFi coverage dynamics. We suppose that there are n trained traces. Snapshots of the route and RSS trace calibration App are shown in Fig. 1.

TABLE I
List of Notations and Explanation

Notations	Explanation
Γ_i	a path i
Y^i	the RSS vector sequence collected on path Γ_i
Y^i_j	the jth RSS vector in sequence Y^i
X_t	RSS sequence in collected online moving window
$Y^s_{[a,b]}$	subsequence of Y^s from Y^s_a to Y^s_b
$\mathbf{C_t}$	the candidate sequence set generated by X_t
m_s	length of RSS sequence 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

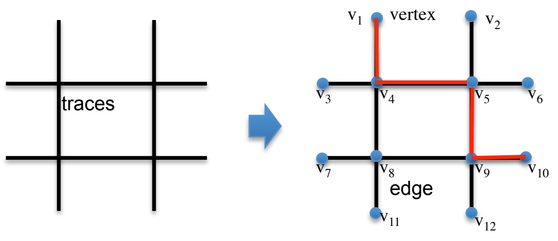


Fig. 2. Construct trace graph from calibrated traces.

B. Construct trace-graph

By simplify RSS trace collection along long routes, in next step, we want to generate RSS traces for all the possible routes. As shown in Fig. 2, suppose we have trained the four routes, the problem is how to generate RSS traces for other possible routes. Training them one-by-one is still laborious, because there maybe many combinatorial routes in the AOI. But note that, some of the routes have common parts, so it is not necessary to train these routes separately. We propose a graph representation for the RSS traces on the indoor routes, which provides an efficient and flexible way to generate the possible candidate routes.

Automatic graph construction: The basic idea is to construct a graph $G = (V, E)$ from the calibrated traces. The construction is composed of a segmentation step and a connection step. In segmentation, if two routes intersect at a common point v, v will be recognized and added as a vertex to the graph. Since the trajectories of all calibrated routes are known, we can find all the intersections formed by these routes. The graph vertex set V include the intersection points and the terminal points (i.e., starting and ending points) of all calibrated routes. Then each route is broken into edges by the vertices on it; these edges form the edge set E. The graph G is called a *trace-graph*. An example of how a trace-graph is formed by the calibrated traces is shown in Fig. 2.

C. Generate CTD

Based on the trace-graph, candidate trace database, which worked as online reference for trace matching is generated. Note that the input sequence in the online phase is a short RSS sequence within a given time window, so the offline generated candidate routes do not need to be long, as long as they are confidentially longer than the size of the moving window. This can greatly reduce the route generation and storage complexity. Hence, we will generate routes by enumerating all paths in G of lengths no more than c, i.e., paths that contain at most c edges. In our implementation, we use $c = 3$ or 4. The route generation algorithm is summarized in Algorithm 1. Here $\mathcal{P}_k$ contains all paths of length k. Let $\mathcal{P}_k(i)$ be the ith path in $\mathcal{P}_k$, and $E = \{e_1, e_2, \cdots, e_{|E|}\}$. Note that if a path is generated by Algorithm 1, its subpaths will not be included.

Note that the result CTD contains a set of moderate-size RSS traces, labeled by the routes where they are collected, in the form of $\{(\Gamma_i, Y^{(i)}) : i = 1, \cdots, L\}$, where L is the total number of traces.

Algorithm 1 c-edge RSS-trace generation

```
Require: $G = (V, E), c$
Ensure: Paths containing at most $c$ edges: $\mathcal{P}$
  Add all edges in $E$ to $\mathcal{P}_1$
  for $k = 1$ to $c - 1$ do
    for $i = 1$ to $|\mathcal{P}_k|$ do
      for $j = 1$ to $|E|$ do
        if $\mathcal{P}_k(i)$ and $e_j$ have a common terminal point and
        $e_j$ is not part of $\mathcal{P}_k(i)$ then
          Connect $\mathcal{P}_k(i)$ and $e_j$, and add the connected
          route to $\mathcal{P}_{k+1}$
          Mark $\mathcal{P}_k(i)$ as "used"
        end if
      end for
    end for
    Delete all "used" paths from $\mathcal{P}_k$
  end for
  Output $\mathcal{P} = \mathcal{P}_1 \cup \cdots \cup \mathcal{P}_c$
```

Theorem 1 (computational complexity): The time complexity of Algorithm 1 is at most $O(N^c)$, where $N = |E|$ is the number of edges and c is a small constant.

Proof 1: We always have $|\mathcal{P}_k| \leq N^k$ throughout the algorithm. Note that it takes $O(1)$ time to add a new path to the database since c is a constant. Therefore the time used by Algorithm 1 is at most $O(\sum_{k=1}^{c-1} N^k \cdot N) = O\left(\frac{N^2(N^{c-1}-1)}{N-1}\right) = O(N^c)$.

Theorem 2 (size of route database): The number of offline generated routes, i.e., $|\mathcal{P}|$, is at most $O(N^c)$, where $N = |E|$ is the number of edges.

Proof 2: Since $|\mathcal{P}_k| \leq N^k$, the number of routes is at most $\sum_{k=1}^{c} N^k = \frac{N(N^c-1)}{N-1} = O(N^c)$.

Remark. As will be shown in our experiments, the moving window in the online phase is generally required to contain RSS readings when moving for 5-10 meters in order to achieve good accuracy. In most buildings, 10 meters can definitely cover 1 to 4 route edges, so choosing $c = 4$ in our experiments is a good balance between locating accuracy and computational complexity. Further, the offline training phase needs only to be run once, so it is acceptable even if c is large.

IV. Trace-Type Fingerprinting: Locating Phase

In this section we present the locating method in trace-type fingerprinting. The basic idea is that RSS is online measured by user's device as the user is moving, which feeds into a moving window containing the latest n measurements of the RSS vectors, denoted by $X = (x_1, \cdots, x_w)$. Note that x_w indicates the current RSS measurement. Then X is searched in the offline trained CTD. The a segment of trace in CTD that best matches with X is detected as the matched subsequence, and the end point of the subsequence is detected as the current position of the user. The key challenge is that the moving speed of a user in online phase may differ greatly than that in offline phase. We propose a subsequence dynamic time wrapping (SDTW) algorithm to measure similarity between two temporal sequences that can tolerate the variances in time or speed.

A. Dynamic time warping

SDTW is an extension of Dynamic Time Wrapping (DTW) [5], [19]. We briefly review DTW at first. For convenience, we use $[n]$ to represent the set $\{1, 2, \cdots, n\}$ for a positive integer n.

Classical DTW: Consider two sequences $X = (x_1, \cdots, x_n)$ and $Y = (y_1, \cdots, y_m)$ whose elements $x_i, y_j (i \in [n], j \in [m])$ are all in the same metric space $\mathcal{X}$ with a distance function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$. Let $P = (p_1, \cdots, p_l)$ be an *alignment* between X and Y, where $p_k = (i_k, j_k) \in [n] \times [m]$ for each $k \in [l]$. We say that P is a *warping path* if it satisfies the following conditions.

- Boundary condition:

$$p_1 = (1, 1), p_l = (n, m). \tag{2}$$

- Step size condition:

$$p_{k+1} - p_k \in \{(0, 1), (1, 0), (1, 1)\} \quad \forall k \in [l - 1]. \tag{3}$$

The distance between X and Y given the warping path P is defined as

$$d_P(X, Y) = \sum_{k=1}^{l} d(x_{i_k}, y_{j_k}). \tag{4}$$

A warping path P^* between X and Y is optimal if it achieves the minimum distance among all possible warping paths. This minimum distance is called the *DTW distance* between X and Y, denoted by

$$\text{DTW}(X, Y) = \min\{d_P(X, Y) : P \text{ is a warping path}\}. \tag{5}$$

To find an optimal warping path, there is a simple dynamic programming algorithm. Consider the following procedure:

$$\begin{aligned} D(i, 1) &= \sum_{i'=1}^{i} d(x_{i'}, y_1), \quad i \in [n] \\ D(1, j) &= \sum_{j'=1}^{j} d(x_1, y_{j'}), \quad j \in [m] \end{aligned} \tag{6}$$

$$\begin{aligned} D(i, j) =& \min\{D(i - 1, j - 1), D(i - 1, j), D(i, j - 1)\} \\ &+ d(x_i, y_j), \quad 1 < i \leq n, 1 < j \leq m. \end{aligned} \tag{7}$$

It is easy to see that $D(i, j)$ stores the DTW distance between $X_{[1,i]} = (x_1, \cdots, x_i)$ and $Y_{[1,j]} = (y_1, \cdots, y_j)$. Thus we have $\text{DTW}(X, Y) = D(n, m)$. The optimal warping path can be found by tracing all the iterations. The time complexity of this dynamic programming algorithm is $O(mn)$ [5].

Subsequence DTW (SDTW): In this problem, the online measured RSS sequence X is generally much shorter than the offline trained RSS traces. Let $Y = (y_1, \cdots, y_m)$ $(n \ll m)$ be one of the offline trained RSS traces. Instead of matching X with whole sequence Y, the goal is to find a subsequence $Y_{[a,b]} = (y_a, y_{a+1}, \cdots, y_b)$ $(1 \leq a \leq b \leq m)$ of Y that can best match with X, such that

$$(a^*, b^*) = \underset{(a,b):1 \leq a \leq b \leq m}{\arg\min} \text{DTW}(X, Y_{[a,b]}). \tag{8}$$

The solution to (8) can also be found using dynamic programming. In fact, it suffices to change the initial values (6) to

$$\begin{aligned} D(i, 1) &= \sum_{i'=1}^{i} d(x_{i'}, y_1), \quad i \in [n] \\ D(1, j) &= d(x_1, y_{j'}), \quad j \in [m]. \end{aligned} \tag{9}$$

Then the same recursion (7) is used. The index b^* can be found by

$$b^* = \underset{b \in [m]}{\arg\min} D(n, b), \tag{10}$$

and then the index a^* can be determined by tracing the iterations. The computational complexity of this algorithm is also $O(mn)$.

B. Locating by SDTW

As introduced in Section III-C, after the offline training phase, a candidate trace database is constructed, which generates a set of L candidate routes, denoted as $\Gamma^{(1)}, \cdots, \Gamma^{(L)}$. For a route $s \in [L]$, let $Y^{(s)} = (y_1^{(s)}, \cdots, y_{m_s}^{(s)})$ be the sequence of RSS vectors along the route $\Gamma^{(s)}$.

In the online phase, a person walks along a route Γ' while measuring RSSs. Let $X = (x_1, \cdots, x_w)$ be the RSS vector measured in the moving window, then online matching is to align X to a subsequence of some $Y^{(s)}$, i.e., $Y_{[a,b]}^{(s)}$ of $Y^{(s)}$. The current location of the target will then be given as $Y_b^{(s)}$.

A distance function on the space of RSS vectors is defined as the l_2-norm of the RSS vector:

$$d(X, Y) = \|X - Y\|_2 \tag{11}$$

where X and Y are the RSS vectors with the same dimension. A preprocessing is used when the measured RSS vector and the trained RSS have different lengths; details are given in Section V.

Since we do not know in advance which route $\Gamma^{(s)}$ is the one Γ' belongs to, we need to find $s^* \in [L]$ such that X can be optimally matched to a subsequence of $Y^{(s^*)}$. Therefore we want to find

$$(s^*, a^*, b^*) = \underset{(s,a,b):s \in [L], 1 \leq a \leq b \leq m_s}{\arg\min} \text{DTW}(X, Y_{[a,b]}^{(s)}). \tag{12}$$

To solve (12), we use SDTW to find for every $s \in [L]$ the optimal alignment between X and a subsequence of $Y^{(s)}$, and choose s^* as the one that achieves the minimum DTW distance. Then the location of $Y_{b^*}^{(s^*)}$ is recognized as the current location of the target. For each $s \in [L]$, finding the optimal alignment between X and a subsequence of $Y^{(s)}$ by SDTW requires $O(w \cdot m_s)$ time. Therefore the total time used for solving (12) is $O(w \sum_{s=1}^{L} m_s) = O(wL \max_{s \in [L]} m_s)$. Combining this with Theorem 2, we have the following theorem.

Theorem 3: Suppose that the number of RSS measurements in the online phase is w. Then the time complexity of online locating by SDTW is $O(c_0 w |E|^c)$, where c_0 is the maximum number of RSS measurements in any c-edge route in $G = (V, E)$.

Proof 3: By Theorem 2, The total number of RSS measurements in CTD is at most $c_0 \cdot |E|^c$. Then the time complexity of using SDTW to determine the current location is bounded by $O(c_0 w |E|^c)$.

V. Implementation and Performance Evaluation

A. Implementation

We develop the trace-type fingerprinting framework using Android platform and the location server is developed by Matlab 2014. The android App carries out the functions including route calibration, RSS traces collection, data transmission to server, and location rendering in the online phase. The server based on Matlab is responsible for conducting trace-map construction and the online localization.

1) Data preprocessing: In trace-type fingerprinting implementation, the collected RSS sequence information has to be preprocessed, because the number and order of detected APs at different locations are rather different. In the preprocessing step, we represent the measured RSS vectors in the same length and rearrange the RSS values in each vector so that each coordinate corresponds to a fixed AP.

In particular, the distinct MAC addresses of APs are counted to form a AP vector. Then, every measured RSS vector is projected to a vector having the same length of the AP vector. Since RSS from some APs cannot be detected at certain locations, a constant is filled as the RSS of the out-of-the-range AP. We test how the filled constant value affects the locating performances, which is presented in Section V-G.

2) Implementing of the locating phase: Locating phase includes collecting the real-time RSS sequences during movement, and finding one optimal matched subsequences of the training set to infer the concrete user location. For data collection, the user just holds the mobile phone and collects real-time sequence of his movement. The latest w RSS vectors forms the moving window. Similar to the training phase, the collected RSS sequences are also preprocessed. We study the impact of the size of the moving window on the locating accuracy. The details are given in Section V-C.

B. Experiment setup

Experiments are mainly conducted on the third floor of Meng Mingwei Science and Technology (South) Building at Tsinghua Universally, which has size $17 \times 60 m^2$. In the offline phase, we walk through the corridor and offices to generate the trace-graph and the candidate route database. In the online phase, we take 16 testing traces.The results are compared with the traditional K-nearest Neighbor (KNN) locating method. In KNN, the online matching is done by finding K most similar RSS vectors from the whole training set and calculating the mean of these K locations as the estimated location of the target. We mainly evaluate:

1) The locating accuracy performance, i.e. how trace-type fingerprinting affects the locating accuracy compared with point-type fingerprinting.
2) The locating robustness, i.e. how the changes of environments and devices affect the locating accuracy of trace-type fingerprinting.

C. Accuracy vs. length of moving window

A key point to evaluate is how trace-type fingerprinting improves the locating accuracy. We evaluate this by varying the length of the moving window and testing how the size of trace affects the locating accuracy.

We test the locating accuracy performance by varying the size of moving window as 3, 5, 10, 15, and 20, and evaluating the locating accuracy in each case. In each setting of the moving window size, we independently generate 500 random testing traces from the 16 long testing traces. The generation method is: (i) randomly select one trace from 16 training traces; (ii) randomly choose a length-w window from the chosen trace. For example, when $w = 10$, 500 random testing traces, each containing 10 RSS vectors are generated on the 16 long testing traces. We use SDTW and KNN to estimate the end point locations of these 500 traces. The locating error is the Euclidean distance from the ground truth position to the estimated position.

Fig. 3 to Fig. 7 show the empirical cumulative distribution functions (CDF) of the locating error when the moving window size varies from 3 to 20. The locating errors of KNN ($k = 2$ and 3) are also plotted for comparison. We can find a general trend that, the longer the moving window is, the better the locating accuracy is. When $w = 3$, SDTW is almost degraded to KNN. When $w > 5$, SDTW shows remarkable locating accuracy improvement than KNN. In all the experiments with $w \geq 5$, SDTW shows better locating accuracy than KNN.

The average locating error as a function of the size of moving window w is plotted in Fig. 8. It clearly shows how the moving window size affects the locating error. We observe that SDTW provides better accuracy when the moving window size is larger than 5, and that its accuracy sharply outperforms KNN in such settings. When the length is too short, SDTW is almost degraded to KNN. After $w > 10$, when the moving window size increases, the locating accuracy tends to decrease a little bit. The reason might be that when w is longer, the accuracy decreases at some boundary locations, because it is hard to form a long sequence. So a window size between 5 and 10 is appropriate. Note that when RSS sampling rate is 1 Hz, user moves about 5-10 meters during this period, which is acceptable in real applications.

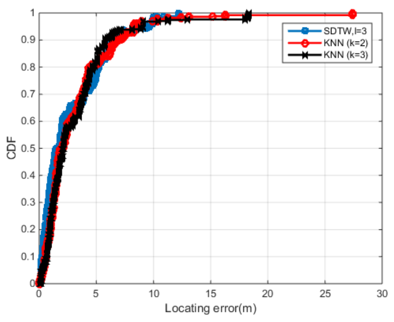


Fig. 3. CDF of locating error when $w = 3$

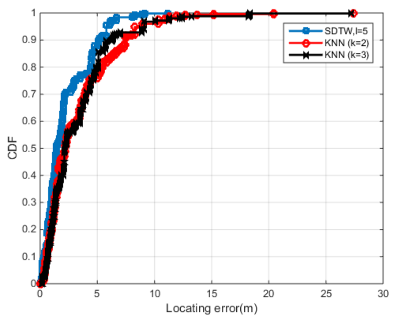


Fig. 4. CDF of locating error when $w = 5$

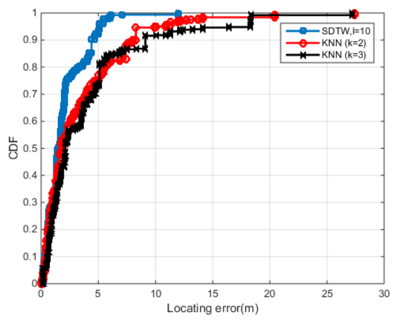


Fig. 5. CDF of locating error when $w = 10$

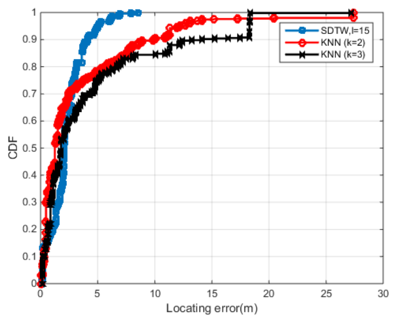


Fig. 6. CDF of locating error when $w = 15$

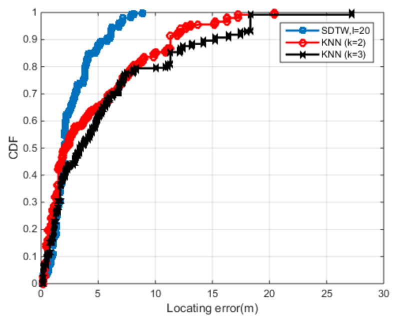


Fig. 7. CDF of locating error when $w = 20$

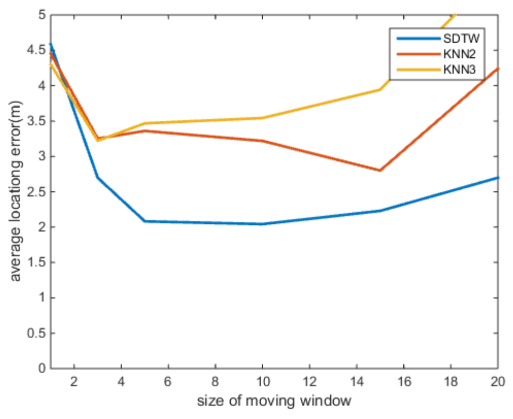


Fig. 8. Mean locating error as a function of w

D. Locating robustness vs. environment changes

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

For testing the impacts of environment changes, we collected one set of training records at 3:00-4:00pm, July 23, 2015 and use it to construct the trace-graph and candidate route database. Then, we collected three sets of testing traces at three different times. Set 1 was collected at 4:00-5:00pm, July 23, 2015. Set 2 was collected at 10:00-11:00am, July 25, 2015. Set 3 was collected at 2:00-3:00pm, July 31, 2015. The temperature when collecting Set 3 is most close to that when collecting the training set. We use SDTW and KNN to calculate the locating errors of these three sets. We use window size $w = 10$ in SDTW in these experiments. The mean locating errors of the three experiments are shown in Fig. 9.

We can see that both SDTW and KNN keep reasonable locating accuracy when the test times differ from the training time. One reason may be that the indoor environment was kept almost the same by the air conditioner. But overall, STDW still has much better locating accuracy than KNN. This shows that we need not frequently update CTD.

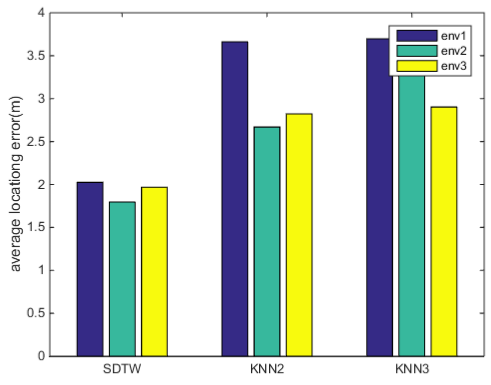


Fig. 9. Mean locating errors in three sets of experiments investigating the impacts of environment changes.

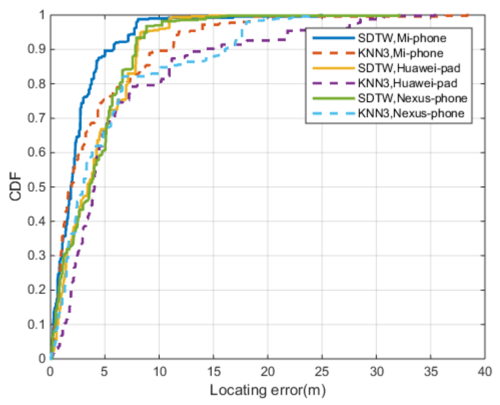


Fig. 10. CDF of locating error when locating device changes.

E. Locating robustness vs. device changes

We also evaluate how the change of locating devices affect the locating accuracy.

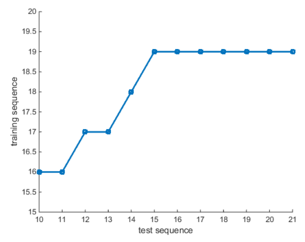
In this experiment, we use Mi-phone to collect the training traces and to construct the trace-graph and the candidate route database. Then, we choose three different mobile phones, i.e., Mi-phone, Huawei-Pad and Nexus-phone to conduct online testing. We calculate the locating errors in each group of testing. The empirical CDFs of the locating error are shown in Fig. 10. We find that both SDTW and KNN are worse when the training device is different from the online locating device, because different devices may have different sensitivity to the nearby AP signal strengths. This difference leads to the accuracy performance degradation when changing to a different device in the online phase.

Nevertheless, we observe that SDTW is still more robust than KNN when the testing device is different from the training device. We can see that some serious errors appear in KNN when the testing devices are changed. For example, for the Nexus group, about 10% of locating errors are beyond 10m. Such performance degradation is not acceptable. But SDTW still provides reasonable accuracy, with mean locating error less than 3 meters, which shows its robustness to device changes.

F. Robustness vs. walking pattern dynamics

In the training phase, the RSS traces are collected while user moves in almost constant speed, but we cannot expect the users in online phase to walk in constant speed. Users may occasionally halt and then start walking again, or may change their walking speed frequently.

To evaluate SDTW's tolerance to the walking speed and walking pattern variances, we design a set of experiments. The tester moves in a walk-stop pattern, i.e., walks, stops, and then walks again repeatedly. The collected window of RSS sequences are searched in CTD using STDW to locate the user. We compare SDTW with an equal-length matching algorithm in which the subsequence in CTD that has the same length w and the has the shortest distance to the measured trace is detected as the location route.



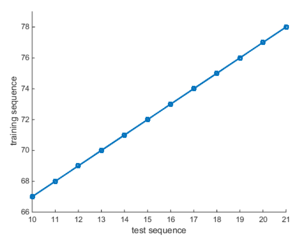


Fig. 11. Sequence matching by SDTW

Fig. 12. Equal-length sequence matching

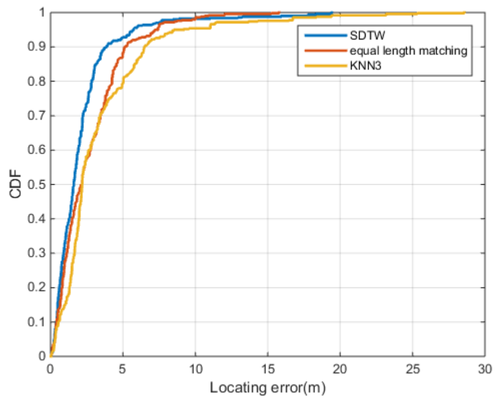


Fig. 13. CDF of locating error in walk-stop experiment.

The matching result of one instance is shown in Fig. 11 and Fig. 12. SDTW matches the moving window with indices 10-20 to a subsequence with indices 16-19 in CTD. The stopping sequence was efficiently wrapped. However, equal-length matching cannot wrap the RSSs at the stopping time. The figure shows the adaptivity of SDTW. Furthermore, the empirical CDF of locating error in the series of walk-stop experiments is plotted in Fig. 13, which shows that SDTW performs the best. It is interesting to see that equal-length matching is a little better than KNN, which shows the accuracy improvement of trace-type matching over point-to-point matching.

G. Robustness vs. filled RSS value for out-of-the-range APs

As mentioned in the implementation section, in data pre-processing, the RSSs of the out-of-the-range APs are filled by a constant value in order to turn the RSS vectors into equal length. We study the impact of this filled value to the locating accuracy.

We evaluated three cases by setting the filled value to 0, -100 and -200. The empirical CDFs of locating errors are plotted in Fig. 14. It shows that the difference of the filling values has little impact on the locating accuracy. The reason is that the out-of-the-range APs in the offline training phase highly consistent with the out-of-the-range APs in the online phase. This shows that the DTW distance is mainly determined by the RSSs of the nearby APs.

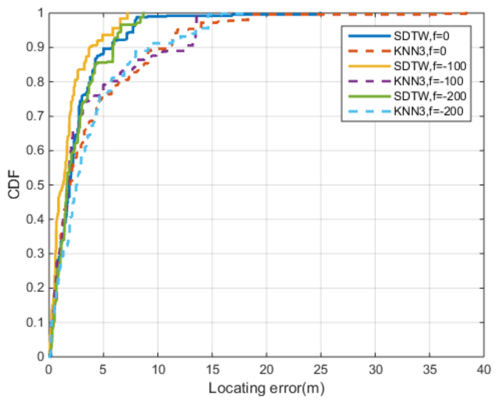


Fig. 14. CDF of locating error when filling different values to the RSSs of out-of-the-range APs.

VI. Conclusion, Discussion, and Future work

This paper presents design methodologies and performance evaluations of a novel trace-type fingerprint method for indoor locating. It is designed to implicitly model the user movement continuity and indoor path constraint into the trace-type fingerprint, while making the training phase easier and the online matching phase more robust. By proposing CTD and STDW, the trace-type fingerprinting method is designed to have low complexity while being very effective. The analysis and performance evaluations show comprehensive merits over the traditional point-type radio map methods.

A. Merits

1) Easier to train: Training can be easily carried out by holding a mobile device and walking along marked routes. The trace-graph generation algorithm is responsible for generating CTD for online reference. This process is much easier than the point-by-point RSS fingerprint calibration.

2) Better accuracy: Trace-type fingerprinting provides better locating accuracy than point-type fingerprinting.

3) Better robustness: It is more robust under environment changes and device changes.

B. Limitations

A limitation of trace-type fingerprinting method is that it is a little complex to be used in large hall-like areas, because there may be innumerable number of routes. An approach is to calibrate routes in grid topology and generate grid topology trace-graph. Although the complexity to generate candidate routes from the dense grid topology trace-graph is high, because it needs to be run only once, it is still possible to use. Further, it might be possible to combine point-type fingerprinting and trace-type fingerprinting in hall-like areas. In this paper we focus on trace-type fingerprinting and leave the combination work into future.

C. Future work

There are still many problems to explore in the future.

1) *How to design hierarchical method to partition large-scale environment?* Since each AP has limited coverage area, an idea of hierarchical locating is to firstly narrow down the location space by matching AP lists, and then to conduct sequence matching within the smaller area. We leave this to future work.

2) *Sensor fusion to construct traces of multiple features.* Today's smart phones contain a set of sensors including inertial sensors, compass, light sensors, acoustic sensors, etc. By training trace-type fingerprint containing multi-type sensor features, the locating accuracy and robustness can be further improved by sensor fusion and sequence matching.

3) *Energy efficiency in online matching is also an important problem.* Since mobile phones have limited battery, how to save energy while using the locating service is an important issue. It can be explored by taking samples more sparsely and integrating with the user moving model.

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 *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '05. New York, NY, USA: ACM, 2005, pp. 205–218. [Online]. Available: http://doi.acm.org/10.1145/1067170.1067193

[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 *Proceedings of the 1st International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, ser. WiNTECH '06. New York, NY, USA: ACM, 2006, pp. 34–40. [Online]. Available: http://doi.acm.org/10.1145/1160987.1160995

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

[6] F. Zampella, A. Jimenez R, and F. Seco, "Robust indoor positioning fusing PDR and RF technologies: The RFID and UWB case," in *2013 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct. 2013, pp. 1–10.

[7] J. C. Aguilar Herrera, P. G. Plöger, A. Hinkenjann, J. Maiero, M. Flores, and A. Ramos, "Pedestrian Indoor Positioning Using Smartphone Multi-sensing, Radio Beacons, User Positions Probability Map and IndoorOSM Floor Plan Representation," 2014. [Online]. Available: https://opus.bib.hochschule-bonn-rhein-sieg.de/frontdoor/index/index/docId/1281

[8] V. Honkavirta, T. Perälä, S. Ali-Löytty, and R. Piché, "A comparative survey of wlan location fingerprinting methods." in *WPNC*, 2009, pp. 243–251.

[9] P. Scholl, S. Kohlbrecher, V. Sachidananda, and K. Van Laerhoven, "Fast indoor radio-map building for RSSI-based localization systems," in *2012 Ninth International Conference on Networked Sensing Systems (INSS)*, 2012, pp. 1–2.

[10] X. Geng, Y. Wang, H. Feng, and Z. Chen, "Hybrid radio-map for noise tolerant wireless indoor localization," in *2014 IEEE 11th International Conference on Networking, Sensing and Control (ICNSC)*, Apr. 2014, pp. 233–238.

[11] M. Molina-García, J. Calle-Sánchez, J. I. Alonso, A. Fernández-Durán, and F. B. Barba, "Enhanced In-Building Fingerprint Positioning Using Femtocell Networks," *Bell Labs Technical Journal*, vol. 18, no. 2, pp. 195–211, Sep. 2013. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/bltj.21613/abstract

[12] Y. Ji, S. Biaz, S. Wu, and B. Qi, "Impact of Building Environment on the Performance of Dynamic Indoor Localization," in *Wireless and Microwave Technology Conference, 2006. WAMICON '06. IEEE Annual*, Dec. 2006, pp. 1–5.

[13] L. Ni, Y. Liu, Y. C. Lau, and A. Patil, "LANDMARC: indoor location sensing using active RFID," in *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003)*, 2003, pp. 407–415.

[14] J. Yin, Q. Yang, and L. M. Ni, "Adaptive temporal radio maps for indoor location estimation," in *3rd IEEE International Conference on Pervasive Computing and Communications (PerCom 2005), 8-12 March 2005, Kauai Island, HI, USA*, 2005, pp. 85–94. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/PERCOM.2005.7

[15] 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.

[16] S. J. Pan, J. T. Kwok, Q. Yang, and J. J. Pan, "Adaptive Localization in a Dynamic WiFi Environment Through Multi-view Learning," in *Proceedings of the 22Nd National Conference on Artificial Intelligence - Volume 2*, ser. AAAI'07. Vancouver, British Columbia, Canada: AAAI Press, 2007, pp. 1108–1113. [Online]. Available: http://dl.acm.org/citation.cfm?id=1619797.1619824

[17] C.-C. Lo, L.-Y. Hsu, and Y.-C. Tseng, "Adaptive radio maps for pattern-matching localization via inter-beacon co-calibration," *Pervasive Mob. Comput.*, vol. 8, no. 2, pp. 282–291, Apr. 2012. [Online]. Available: http://dx.doi.org/10.1016/j.pmcj.2012.01.001

[18] Z. Yang, Y. Wang, and L. Song, "Adamap: Adaptive radiomap for indoor localization," in *Ad-hoc, Mobile, and Wireless Networks - 14th International Conference, ADHOC-NOW 2015, Athens, Greece, June 29 - July 1, 2015, Proceedings*, 2015, pp. 134–147.

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