Automatic Floor Map Construction for Indoor Localization

Xin Luo, Albert Kai-sun Wong, Mu Zhou, Xuning Zhang, and Chin-Tau Lea
Electronic and Computer Engineering Department
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Emails: xluo@connect.ust.hk, eealbert@ust.hk, zhoumu@cqupt.edu.cn, eexuning@ust.hk, and eelea@ece.ust.hk.

***Abstract*—Existing indoor localization systems based on Wi-Fi Received Signal Strength (RSS) fingerprinting often assume the knowledge of a map of the coverage area and involve a tedious manual survey process at a set of sample locations along this map. In this paper, we describe an automatic graphical floor map and radio fingerprints generation system, called the Intelligent Mobility Mapping System (IMMS), which applies the concepts of crowd-sourcing and Simultaneous Localization and Mapping (SLAM) to construct a floor map in support of indoor people localization and tracking. IMMS makes use of high similar patterns in crowd-sourced traces of RSS measurements to identify location segments in the coverage area and to construct a graphical floor map. With IMMS, the elaborate off-line manual data collection process is eliminated.**

Keywords–Wi-Fi SLAM; Indoor Localizations; Graph Theory; Mobility Mapping.

I. INTRODUCTION

Indoor localization based on radio signals has been a focus of research for over 10 years. Kong et al. [1] study the use of CDMA2000 pilot signals to record the fingerprints for radio map construction. The fingerprinting approach for indoor localization typically requires a time consuming off-line survey process for building a radio map. Various proposals have been made to reduce the complexity of this process. Ouyang et al. [2], proposed to use unlabeled Wi-Fi Received Signal Strength (RSS) data to enhance the radio map created by labeled survey data. Zhang et al. [3], used a pedestrian mobility model based on knowledge of the physical indoor map to enhance localization.

Recently, the new concept of Simultaneous Localization and Mapping (SLAM) has been developed with the objective of minimizing the off-line survey effort [4]. A SLAM system gathers location and mapping information at the same time, and requires only a very short time for off-line survey. Typically, SLAM records a user's 'footprint' using Microelectromechanical System (MEMS) devices such as accelerometer, gyroscopes and magnetometers on a mobile device, and labels the RSS measurements with this footprint information to construct the mobility maps. Then, the system can locate users on this constructed indoor map in the on-line stage. For example, the system proposed by Shin et al. [5] constructs a floor plan of a building by integrating the number of walking steps (from pedometer), the walking orientation (from magnetometer), and the RSS values recorded with user movements. Zhou et al. [6] use only Wi-Fi RSS measurements from one or multiple individuals walking around the coverage. Similar RSS measurements are clustered and aligned to construct a map for the coverage area.

A. Main contributions

In this paper, we present a system called the Intelligent Mobility Mapping System (IMMS), which is a crowd-sourced system that sporadically collects traces of RSS measurements from users moving around the indoor coverage area as they carry on their daily routines. IMMS automatically creates graphical floor maps to support Wi-Fi RSS-based indoor localization and tracking. There are three modules in IMMS. The first module is designed to facilitate the subsequent processes by various data pre-processing methods. The second module is designed to find the highly similar pieces of measurements in traces. IMMS estimates similarities of measurements in different traces by correlations. Then, the resemblant measurements in different RSS traces are clustered as High Cross-Trace Correlation Patterns (HCP), and the intersections of these HCPs are used to segment traces into Atomic Location Segments (ALSs). The third module is designed to construct the draft graphical floor map and radio map. The floor map is a simple planar embedding of a drawing that represents the interconnections of ALSs.

Section II overviews the system framework. Section III describes the location segment recognition algorithm and Section IV explains the graph drawing procedures.

B. Notations

Notations to be used in this paper are first summarized in Table I.

TABLE I. IMPORTANT NOTATIONS

Symbol	Meaning
$\mathbf{R^l}$	l^{th} trace
$\nu_{\mathbf{i}}^{\mathbf{l}}$	i^{th} measurement in l^{th} trace
$\mathbf{S}$	Raw data matrix
$\mathcal{N}$	Number of Traces in $\mathbf{S}$
Υ	Number of measurements in $\mathbf{S}$
$\mathcal{M}$	Number of hearable APs in the target area
$\mathbf{C}$	Cross-Trace Correlation (CTC) matrix
$\mathbf{C^{\{f,g\}}}$	Submatrix of CTC matrix corresponds to $\mathbf{R^f}$ and $\mathbf{R^g}$
$C_{i,j}^{\{f,g\}}$	A element in $\mathbf{C^{\{f,g\}}}$
$\mathbf{Q}$	Quantized matrix of $\mathbf{C}$
x_i	Row breaking points
y_i	Column breaking points
$\mathbf{G}$	Geometric map
$\mathbf{U}$	Set of unique vertices/ALS of graph G
$\mathbf{E}$	Set of unique edges of graph G
$d(*)$	Direction of an edge
I	Indication matrix of endpoints

II. SYSTEM OVERVIEW

During the data collection phase, traces of RSS measurements are recorded from mobile phones when users move around the coverage area as they conduct their daily activities

 ISBN: 978-1-61208-360-5

indoors. For a user, the RSS measurements are collected at a regular time interval (1 read/sec by default) when movement is detected. Data collection stops when the user ceases moving. Each measurement is a sample vector that contains the measured RSSs from a set of hearable Wi-Fi APs. The recorded traces are uploaded to IMMS in cloud.

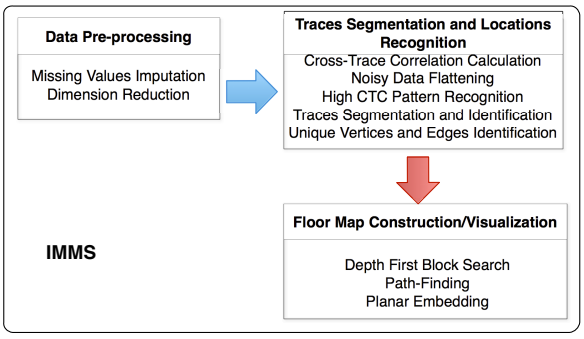


Figure 1. The System Framework

IMMS compares the unlabeled measurements in each pair of traces to see if there are highly similar ones. It is expected that measurements in two traces would exhibit similar distributions if they were collected at nearby places. A string of similar measurements between two traces form what we call a High Correlation Pattern (HCP). Intersecting the HCPs form by a given trace with all other traces allows us to segment HCPs into what we call ALS, which we expect would represent corridor sections between intersections in the physical environment. By observing how the ALSs are connected in the traces, we can then construct a 2D graphical floor map. The architecture of IMMS is shown in Fig.1. It contains three modules: data pre-processing, traces segmentation and locations recognition, and floor map.

The data pre-processing module includes missing value imputation and dimension reduction. Because of the limited coverage of each AP and the random variation of the RSS signals, most of the recorded RSS values are zero. In order to insure the accuracy of the similarity evaluation, we need to find a way to impute these zero, or missing values. Based on recommendation provided by Ouyang et al. [7], we impute missing values with a number that is smaller than the minimum collected RSS measurements. Moreover, because the total number of APs in the coverage area can be very large [6], it is desirable to reduce the data dimensionality to reduce computation complexity. We apply Principal Component Analysis (PCA) [8] to reduce the measurement dimensionality.

The details of the second module ,traces segmentation and locations recognition module, and the third module, floor map construction module, will be given in Section III and Section IV, respectively.

III. Traces Segmentation and Locations Recognition

Our approach is based on the premise that the physical space can be modeled as an interconnection of corridor segments that we call ALSs. Users tend to traverse an ALS in its entirety, and in one of two directions. Each recorded trace of

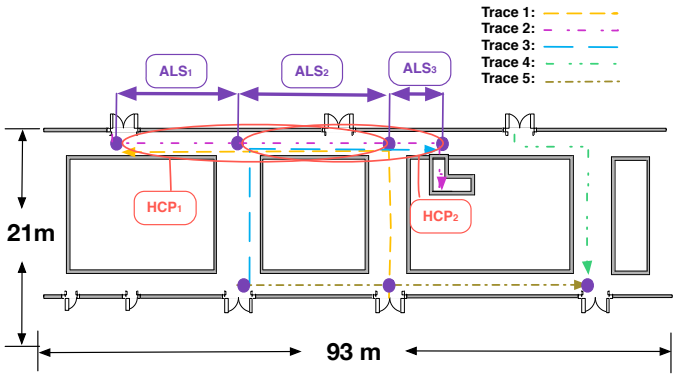


Figure 2. Example of Traces, HCPs and ALSs

Wi-Fi measurements reflects the movement of a user through a number of ALSs.

The objective of traces segmentation is to identify all ALSs in the physical environment using traces from crowdsourcing. The traces segmentation process starts with the recognition of any high similarity pattern, which we call High Cross Trace Correlation (CTC) Pattern, that may exist in any given pair of traces. A High CTC Pattern (HCP) reflects a sequence of overlapping ALSs between two traces. From the starting points and ending points of all HCPs found in many trace pairs, we can identify all the Breaking Points (BPs) which separate the ALSs in all the traces. In other words, we identify all the individual ALSs by intersecting all the HCPs. An example is shown in Fig. 2. Computation of similarity is based on the dimension-reduced measurements after data pre-processing.

A. Cross-Trace Correlation (CTC)

We measure the Cross-Trace Correlation (CTC) by the Pearson product-moment [9] correlation coefficient between measurements in two different traces. Assume $\mathcal{N}$ traces are collected. Let $\Re^{\mathbf{f}} = (\nu_1^f, ..., \nu_{n^f}^f)$ be the f^{th} $(f = 1, ..., \mathcal{N})$ trace, where ν_i^f is the $i^{th} (i = 1, ..., n^f)$ measurement in trace $\Re^{\mathbf{f}}$, and n^f is the number of measurements in the trace. Each measurement is a row vector $\nu_i^f = [\nu_{i,1}^f ... \nu_{i,\mathcal{M}}^f]$, where $\nu_{i,j}^f$ is the signal strength from the $j^{th} (j = 1, ..., \mathcal{M})$ AP. The raw data of all $\mathcal{N}$ traces can be kept in a $\Upsilon \times \mathcal{M}$ matrix $\mathcal{S} = [\Re^{\mathbf{1}} \Re^{\mathbf{2}} ... \Re^{\mathcal{N}}]^T$, where $\Upsilon = \sum_{f=1}^{\mathcal{N}} n^f$ is the total number of measurements in the data. After dimension reduction, each measurement is reduced to k dimension, denoted as $\mu_{\mathbf{i}}^{\mathbf{l}}$ ($k-$dimension row vector). The dimension reduced data matrix is $\mathbf{S} = [\mathbf{R^1 R^2} ... \mathbf{R}^{\mathcal{N}}]^T$ ($\Upsilon \times k$ matrix). We define CTC matrix, $\mathbf{C}$, as the matrix containing the sub-matrices $\mathbf{C}^{\{\mathbf{f,g}\}}$, where $f, g = \{1, ..., \mathcal{N}\}$. $\mathbf{C}^{\{\mathbf{f,g}\}}$ contains CTC values of measurements in trace $\mathbf{R^f}$ and $\mathbf{R^g}$, denoted as $C_{i,j}^{\{f,g\}}$.

Definition 1: The CTC value of measurements $\mu_i^f \in \mathbf{R^f}$ and $\mu_j^g \in \mathbf{R^g}$ is

$$C_{i,j}^{\{f,g\}} = corr(\mu_{\mathbf{i}}^{\mathbf{f}}, \mu_{\mathbf{j}}^{\mathbf{g}}) = \frac{1}{\sigma_i^f \sigma_j^g} E[(\mu_{\mathbf{i}}^{\mathbf{f}} - \mathbf{m})(\mu_{\mathbf{j}}^{\mathbf{g}} - \mathbf{m})]. \quad (1)$$

The column mean of the dimension reduced sample matrix $\mathbf{S}$ is $\mathbf{m} = [m_1 ... m_k]$. Then, $m_j = \frac{\sum_{f=1}^{N} \sum_{i=1}^{n^f} \mu_{\mathbf{i,j}}^{\mathbf{f}}}{\Upsilon}$. σ_i^f, σ_j^g are

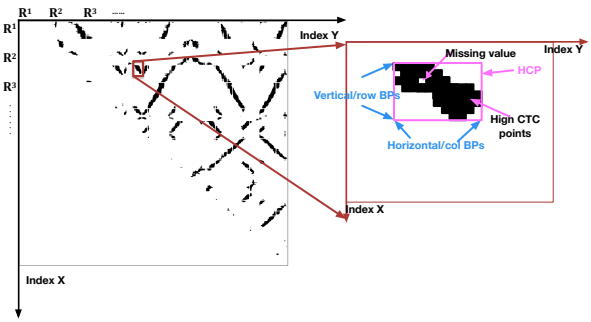
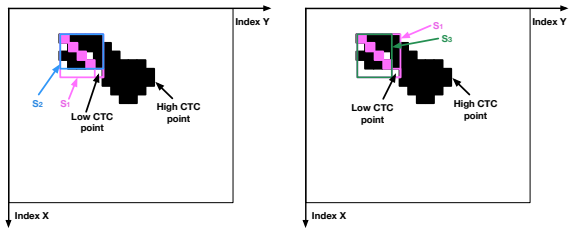


Figure 3. Examples of data structure and HCP



(a) (b)

Figure 4. Example of sub-matrices s_1, s_2, s_3

the standard derivation of measurements $\mu^{\mathbf{f}}_{\mathbf{i}}$ and $\mu^{\mathbf{g}}_{\mathbf{j}}$, calculated as $\sigma_i^f = \sqrt{\sum_{j \epsilon k}(\mu_{i,j}^f - m_j)^2}$.

B. High CTC Pattern Recognition (HCPR)

The High CTC Pattern Recognition (HCPR) algorithm is used to identify groups of correlated measurements in each pair of traces. If two measurements, $\mu^{\mathbf{f}}_{\mathbf{i}} \in \mathbf{R^f}$ and $\mu^{\mathbf{g}}_{\mathbf{j}} \in \mathbf{R^g}$ have high CTC value, we assume they are collected at nearby physical locations in the two traces and we expect that:

- $\mu^{\mathbf{f}}_{\mathbf{\{i+1\}}}$ and $\mu^{\mathbf{g}}_{\mathbf{\{j+1\}}}$ will continue to be a high CTC point if the two users are traversing an ALS in the same direction;
- $\mu^{\mathbf{f}}_{\mathbf{\{i+1\}}}$ and $\mu^{\mathbf{g}}_{\mathbf{\{j-1\}}}$ will continue to be a high CTC point if the two users are traversing an ALS in opposite directions.

Thus, the high CTC points should continue in either the southeasterly direction or southwesterly direction until trace $\mathbf{R^f}$ and trace $\mathbf{R^g}$ diverge to two distinct ALSs. We call the point that breaks the continuity of the high CTC points a *breaking point* (BPs), shown in Fig. 3.

The HCPR algorithm first quantizes all the CTC points in the evaluated submatrix to two quantization levels, setting all CTC values bigger than a given threshold $thres$ as 1 and all CTC values lower than $thres$ as 0. The resulting matrix is denoted as $\mathbf{Q^{\{f,g\}}}$. Then, HCRP groups a contiguous set of high CTC points in the quantized matrix into a high CTC Pattern (HCP).

Missing values and random variations in the raw measurement may lead to unwanted breaks in the high CTC pattern. Additionally, the walking speeds of different users are different, and so the height and width of a HCP may be different. We apply the following criteria to determine the boundary of an HCP:

1) HCPR amounts to finding the row indexes and column indexes of each submatrix in Q that encloses a continuous cluster of 1s.

2) There should be a sufficiently large gap between two different HCPs.

3) Small HCPs should be discarded as noise.

Our algorithm for finding HCPs works as follow. We start from the top row of each sub-matrix $\mathbf{Q^{\{f,g\}}}$ and scan from left to the right for 1s. If none is found, we move to the next row below and scan from left to right again. Once a 1, or a high CTC point, is found, we mark the column and row index of this point as (x, y), and keep searching in the southeasterly or southwesterly direction for 1s. After an HCP is identified, we resume the search for new HCP on the row we left off from. Points covered in a search will be precluded from future search. Starting from a 1, if the point to the southeast is also a 1, then HCPR continues to look for a 1 in the southeast. If a 0 appears, then HCPR calculates the sums of the three sub-matrices indicated in Fig. 4. It compares s_1 and s_2, and s_1 and s_3. If $s_2 = s_1$, it means that there are no new high CTC points is added when the row number is increased (Fig. 4(a)), then the row gap counter: $row_{gap} = row_{gap} + 1$. If $s_2 > s_1$, then one or more new high CTC points are added when the row number is increased. In a similar way, if $s_3 = s_1$, the column gap counter: $col_{gap} = col_{gap} + 1$. If $s_3 > s_1$, then one or more new high CTC points is added because of when the column number is increased (Fig. 4(b)). Once row_{gap} or col_{gap} reaches the defined gap threshold gap_{thres}, HCPR stores the row/column end point of the current HCP, and stops the search of 1s in the southeasterly direction. Next, HCPR starts searching for 1s in the southwesterly direction, using the same search and stop logic as above. Finally, HCPR labels the current HCP using the smallest stored row index and smallest stored column index as well as the largest stored row index and the largest stored column index. The smallest and largest row indexes represent the BPs in trace $\mathbf{R^f}$ and the smallest and largest column indexes represent the BPs in trace $\mathbf{R^g}$. For any possible new HCP between trace $\mathbf{R^f}$ and $\mathbf{R^g}$, HCPR starts searching on row x five entries to the right of the largest column index of the current HCP. The details of HCPR are specified in **Algorithm 1**.

C. Traces Segmentation and Identification Algorithm

From HCPR, we determine the end points of all HCPs. These end points are the BPs in physical paths at which user paths may diverge. Let $\{x_1^1, ..., x_{n_1}^1, x_1^f, ..., x_{n_f}^f x_1^N, ..., x_{n_N}^N\}$ be the set of row indexes in increasing order which are marked as BPs in the vertical direction of $\mathbf{Q}$, where x_i^f is the i-th BP marked for trace R^f and n_f the number of BPs marked for trace R^f. Likewise, let $\{y_1^1, ..., y_{n_1}^1, y_1^f, ..., y_{n_f}^f y_1^N, ..., y_{n_N}^N\}$ be the set of column indexes in increasing order which are marked as BPs in the horizontal direction of $\mathbf{Q}$. Although $\mathbf{Q}$ is symmetric, because of the way we identify the BPs in HCPR, some x_i^f and y_i^f can become different (empirically most of them are the same).

Our algorithm partitions all the traces using the BPs identified and assigns a unique label (ALS ID) to segments

 ISBN: 978-1-61208-360-5

Algorithm 1: High Correlation Patterns Recognition (HCPR)

Initial: $hid := hid + 1 := 1$, $row_{gap} := 0$, $col_{gap} := 0$.
In a submatrix $\mathbf{Q}^{\{f,g\}}$;
For $(x = 1, x <= n^f, x = ++)$
  For $(y = 1, y <= n^g, y = ++)$
  COMMENT: Search high CTC point
  If $\mathbf{Q}^{\{f,g\}}(x, y) = 1$:
    Store row index x and column index y;
    COMMENT: Track the point in the **southeasterly** $\mathcal{Q}^{\{f,g\}}(x + i, y + i)$:
    For $(i = 1, i <= n^f - x, i = ++)$
      If $\mathbf{Q}^{\{f,g\}}(x + i, y + i) = 0$
        $s_1 = sum(\mathbf{Q}^{\{f,g\}}(x : x + i, y : y + i))$;
        $s_2 = sum(\mathbf{Q}^{\{f,g\}}(x : x + i - 1, y : y + i))$;
        $s_3 = sum(\mathbf{Q}^{\{f,g\}}(x : x + i, y : y + i - 1))$;
        COMMENT: Compare the sums
        If $s_2 == s_1$
          $row_{gap} = row_{gap} + 1$;
          If $row_{gap} == gap_{thres}$
            Store row index $x + i - gap_{thres}$;
          END
        END
        If $s_3 == s_1$
          $col_{gap} = col_{gap} + 1$;
          If $col_{gap} == gap_{thres}$
            Store column index $y + i - gap_{thres}$;
    End End End End
    COMMENT: Keep iterating until col_{gap} and row_{gap} reach gap_{thres}
    Track high CTC points in the **southwesterly** direction by the same process
  End
  Label the current HCP with hid.
End End

Figure 5. High Correlation Patterns Recognition Algorithm

in different traces that recognized as the same ALS. The algorithm is described by the pseudo-code in **Algorithm 2** and **Algorithm 3**. In **Algorithm 2**, each trace R^f is considered as a sequence of ALSs. Each segment, e_i^f, is a cluster of RSS measurements between two row BPs x_i^f and $x_i^f + 1$. If a segment has not yet been labelled, we label it with a new ALS ID, and then consider different traces R^g (for $g > f$) and check whether if any ALSs e_j^g in R^g forms an HCP with e_i^f. This checking is via the test function **TestHCP** as described in **Algorithm 2**. Basically, the algorithm checks whether the average number of "1"s in the submatrix of $\mathbf{Q}$ formed by e_i^f and e_j^g is greater than a given threshold. **TestHCP** also determines whether e_j^g is in the same or opposite direction as e_i^f, based on whether the column indexes and row indexes of the "1"s within the sub-matrix is positively or negatively correlated.

As result, we derive from each trace a sequence of ALSs. Each e_i^f is identified with an ALS ID $l(e_i^f) = r$ and marked with directionality $d(e_i^f) = \pm 1$. A sample of labeled ALSs is shown in Fig. 6.

D. Unique Vertices and Edges Identification Algorithm

Next, we proceed to identify and label all the unique vertices that connect the ALS's. Let s_r represent the starting vertex and t_r the terminating vertex of ALS r in the reference direction. Then, we can represent each segment e_i^f in a trace by a tuple of two vertices as follow:

Assume $l(e_i^f) = r$. If $d(e_i^f) = +1$, then $e_i^f = (s_r, t_r)$; if $d(e_i^f) = -1$, then $e_i^f = (t_r, s_r)$.

Then, we can identify vertices that are the same by examining the sequence of segments in all traces, using an indicator $\mathbf{I}$ to record the result as follows: For trace $\mathbf{R^f}$, if $l(e_i^f) = r$, $l(e_{i+1}^f) = r'$, we set:

Algorithm 2: Traces Segmentation Algorithm

COMMENT: Each trace is now viewed as a sequence of ALSs, where each ALS is bounded by two BPs found in HCPR: $R^f \rightarrow (e_1^f, e_2^f, ..., e_{n_b^f - 1}^f)$;
COMMENT: e_i^f is the i^{th} segment in trace R^f
Initialize ALS ID: $r = 0$;
Initialize labels of all segments as null: $l(e_i^f) = null$ for all f, i.
Row BP: $\{x_i^f; f = 1, ..., 2\mathcal{N}, i = 1, ..., 2n_b^f\}$; Col BP: $\{y_i^f = x_i^f; f = 1, ..., 2\mathcal{N}, i = 1, ..., 2n_b^f\}$;
For $(f = 1, f <= \mathcal{N}, f + +)$; COMMENT: trace R^f
  For $(i = 1, i <= n^f, i + +)$
    If e_i^f not already labelled;
      COMMENT: e_i^f represent the i^{th} segment in trace R^f
      $r = r + 1$;
      $l(e_i^f) = r$;
        COMMENT: Label e_i^f by r;
      $d(e_i^f) = 1$;
      COMMENT: e_i^f becomes the r^{th} reference edge and has a direction of 1
      For $(g = f + 1, g <= \mathcal{N}, g + +)$
        For $(j = 1, j <= n_b^g, j + +)$
          $[Tru, Dir] = HCPrest(e_i^f, e_j^g)$
          COMMENT: Call function TestHCP (**Algorithm 3**)
          COMMENT: Return trueness of whether $\{e_i^f, e_j^g\}$ are highly similar
          COMMENT: Define direction of e_j^g.
          If $Tru == 1$ COMMENT: the subset in an HCP
            $l(e_j^g) = r$;
            COMMENT: Label measurements in e_i^f with c_r.
          End
          $d(e_j^g) = Dir$;
End End End End End

Figure 7. Trace Segmentation Algorithm

Algorithm 3: Function: TestHCP

COMMENT: Tru is the Boolean type variable.
COMMENT: Returns 1 when judgment is true, 0 when judgment is false;
COMMENT: Dir return the direction of e_j^g
Function TestHCP(e_i^f, e_j^g)
  $\mathbf{B} = \mathbf{Q}(x_i^f : x_{i+1}^f, y_j^g : y_{j+1}^g)$;
  COMMENT: Sub-matrix of Q formed by e_i^f and e_j^g
  If $\frac{sum(B)}{(x_{i+1}^f - x_i^f) \times (y_{j+1}^g - y_j^g)} > Sum_{thres}$
    $Tru = 1$;
    COMMENT: The density of 1 in B is greater than threshold
  Else
    $Tru = 0$;
  End
  Let the set (xy) be the row and column indexes of all the 1s in B
  If $(Corr(x, y) > 0)$
    $Dir = 1$;
    COMMENT: HCP extends south-easterly, direction of e_j^g is equal to e_i^f
  Else
    $Dir = -1$;
    COMMENT: Direction of e_j^g is the opposite with e_i^f
  End
Return(Tru,Dir)

Figure 8. TestHCP function

$$\begin{cases} \mathbf{I}(t_r, s_{r'}) = 1 & \text{if} \quad d(e_i^f) = +1, d(e_{i+1}^f) = +1; \\ \mathbf{I}(t_r, t_{r'}) = 1 & \text{if} \quad d(e_i^f) = +1, d(e_{i+1}^f) = -1; \\ \mathbf{I}(s_r, s_{r'}) = 1 & \text{if} \quad d(e_i^f) = -1, d(e_{i+1}^f) = +1; \\ \mathbf{I}(s_r, t_{r'}) = 1 & \text{if} \quad d(e_i^f) = -1, d(e_{i+1}^f) = -1. \end{cases} \tag{2}$$

Assume all traces contain a total of $\mathcal{N}^a$ segments. That means there are $2\mathcal{N}^a$ vertices and the indicator matrix $\mathbf{I}$ is a $2\mathcal{N}^a \times 2\mathcal{N}^a$. As described in **Algorithm 4**, we examine the $2\mathcal{N}^a$ vertices one by one. If a vertex i has not yet been labelled, we label it as well as all other vertices with $\mathbf{I}(i, j) = 1$ with a new vertex ID. The result is a set of unique vertices $\mathbf{U} = \{U_1, ..., U_{\mathcal{N}^u}\}$. Knowing the set of unique vertices, we can further verify an unique edge as the edge connecting two distinct unique vertices. The set of unique edges is $\mathbf{E} = \{E_1, ..., E_{\mathcal{N}^e}\}$.

 ISBN: 978-1-61208-360-5

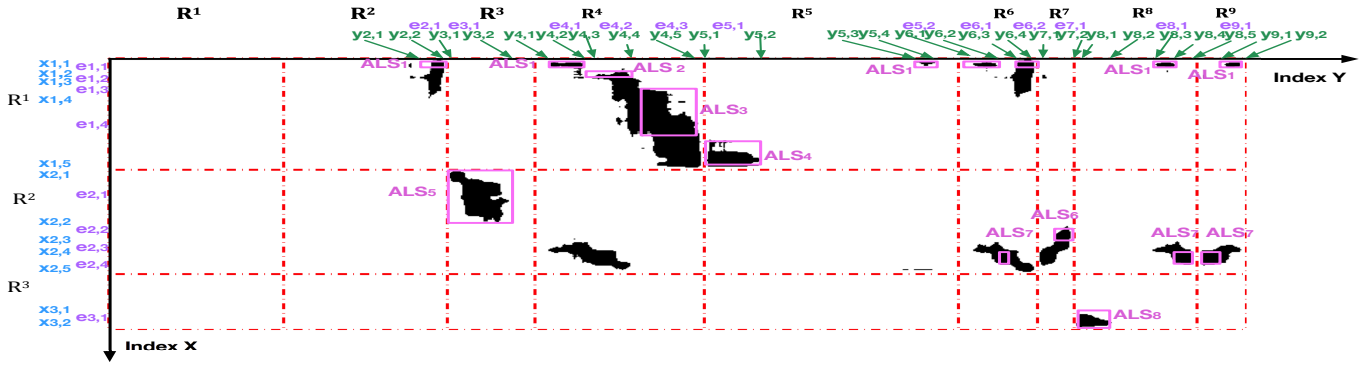


Figure 6. Relationship of BPs of HCPs and Trace Segments

Algorithm 4: Unique Vertices Identification Algorithm

```
COMMENT: Search the indication matrix to verify the unique vertices
Initialize vertex label as v := 0; vertex label is a 1 × 2N^a zero vector;
For (i = 1, i <= 2N^a, i++)
  If l(v_i) is unlabeled
    v = v + 1;
    For (j = j + 1, j <= 2N^a, j++)
      If I(i, j) == 1
        Label vertices l(v_i) = l(v_j) = v;
    END; END
    Elseif l(v_i) is unlabeled
      For (j = j + 1, j <= 2N^a, j++)
      If I(i, j) == 1
        Label vertices l(v_j) = l(v_i);
End End End End
Return N^u = v;
```

Figure 9. Unique Vertices Identification Algorithm

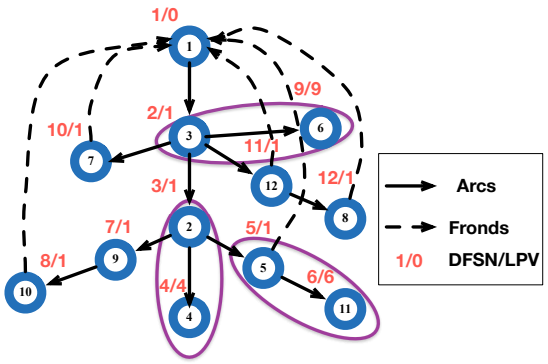


Figure 10. A sample of spanning tree

IV. Floor Map Construction/Visualization

With the set of vertices $\mathbf{U}$ and the set of unique edges $\mathbf{E}$, we create graph $\mathbf{G} = (\mathbf{E}, \mathbf{U})$. The graph is described by an adjacency matrix $\mathbf{A}$. The draft floor map is an embedding of $\mathbf{G}$ which is drawn according to the adjacency matrix.

The floor map construction algorithm aims to embed the graph $\mathbf{G}$ on a plane in a way that is visually more intuitive to a human observer. Three steps are involved: Depth First Block Search (DFBS), path finding, and straight-line embedding.

A. Depth First Block Search

This step is based on Tarjan's DFS block search algorithm [10]. The purpose is to label each vertex with a DFS number $DFSN(v)$, create a spanning tree, and identify blocks and fronds in the graph in order to enable path finding. DFS starts from the vertex with the highest node degree, and iteratively searches for unexplored descendants. Each new vertex is numbered by a DFS number according to the order in which it is explored. DFS stops when all the edges are explored. As result, the edges are separated into a set of arcs making up a spanning tree $\mathbf{T} = v \rightarrow w$, where $DFSN(v) < DFSN(w)$, and a set of fronds $\mathbf{F} = \mathbf{E} - \mathbf{T} = v \dashrightarrow w$, where $DFSN(v) > DFSN(w)$. In addition, DFS assigns an important parameter called the *low point value* to each vertex, and identifies the "blocks", which are the biconnected components of the graph. For vertex v, its *low point value* is defined as $LPV(v) = min(\{DFSN(v)\} \cup \{LPV(w)|v \rightarrow w\} \cup \{DFSN(w)|v \dashrightarrow w\})$ where initially all low point values are set to be the corresponding DFS number: $LPV(v) = DFSN(v)$. After the low point values are calculated, we look for all vertices such that $DFSN(v) \leqslant LPV(v)$. The edge leading to such a vertex v is a bridge, which is an edge whose deletion would partition the graph. The bridge and all edges whose connectivity to the graph depends on this bridge are grouped into a sub-graph called a block. All remaining edges are also grouped into a block. An example of a spanning tree is shown in Fig. 10. The numbers next to each node are the nodes DFS number and low point value respectively. There are four blocks in the example, and the largest block is the set of edges that exclude edges $(3, 6)$, $(2, 4)$, $(5, 11)$.

B. Path-Finding Algorithm

This algorithm searches for circle paths block-by-block, starting from the largest block. Initially, all vertices and edges in the block are marked as unexplored. We start from the vertex with the smallest DFS number in the block, which is the root vertex in the subgraph and mark it as explored. In each iteration, we extend the path to a neighbor vertex with unexplored edges. If the neighbor contains an unexplored frond, the path is outputted as a circle path. We repeat until all vertexes and edges in the block are explored. If there is only one edge which is in T in the block, we also output the edge as a path.

C. Planar Embedding Algorithm

This algorithm is based on a straight-line planar drawing algorithm [11]. Two main constraints are considered: i) Each edge must be a straight-line; ii) The angle between edges

 ISBN: 978-1-61208-360-5

must have good angle resolution. The path-finding algorithm produces a set of distinct circle paths. The direction of the paths is ignored. We first draw the circle paths with four edges, followed by those with three and then five or more edges. Finally, the non-circle paths are drawn. The final result is a draft floor map, which would enable us to visualize the logical relationship of the vertices and edges on a plane.

V. Experiment and Results

The experiment took place at the lab area on the third floor of our academic building. The actual floor map of the area is shown in Fig. 2. The total survey area is 93 meters by 21 meters. In the experiment, a student is equipped with SAMSUNG GALAXY Tab 2 (7.0 version) and walks at a relatively constant speed around the area. A total of 20 traces are recorded with a total of 1468 measurements containing RSS values from 267 APs.

Following the framework of IMMS (shown in Fig. 1), the dimension of the measurement is reduced to 28. The traces segmentation and locations recognition algorithm identifies 38 ALSs in total. Then, the unique vertices identification algorithm produces 11 unique vertices and 13 unique edges.

Fig. 11 is the resulting draft floor map, and Table II shows the relationship of unique edges and ALSs. The two numbers next to the edge index are the number of times the edge appears in different traces and the total number of measurements corresponding to the edge. The black solid arrows are edges that appear more frequently and they match the corridors in the physical floor map quite well. The dash arrows are noise vertices and edges, which do not match the physical floor map. They apparently arise because of variability in the RSS signals of the APs. In the future, we may need to conduct more extensive experiments to determine how we may eliminate the noise vertices and edges or how we may merge them with those that match the physical map.

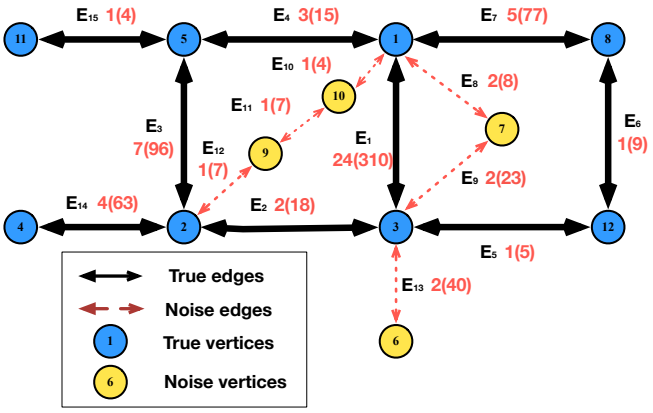


Figure 11. The resulting graphical floor map

VI. Conclusion and Future Work

In this paper, an automatic floor mapping system, IMMS, for draft floor map construction was presented. IMMS uses unlabeled crowd-sourced RSS measurements to construct the floor map of a building. Unlike existing fingerprinting methods, no elaborate manual off-line data collection process at fixed location is required. Accelerometers or other MEMS devices for measuring heading directions and distances are also not used. IMMS is an unsupervised system and is time efficient. The frequently found ALSs can be correctly correlated to corridor segments in the physical environment. Some noise vertices and edges are produced because of variability in the RSS signals. We need to conduct more extensive experiments and to enhance our algorithms so that these noise vertices and edges can be eliminated or merged with other ones.

The next step of our work is to construct a radio map on top of the draft floor map. The radio map can then be used in on-line localization and tracking application of individuals.

TABLE II. Adjacency List of Unique Vertices

Vertex Inx.	Incident Edges	ALS Inx.
Node U_1	$\mathbf{E_1}$	1, 3, 22, 24
	$\mathbf{E_4}$	21
	$\mathbf{E_8}$	4
	$\mathbf{E_7}$	6
	$\mathbf{E_{10}}$	14
Node U_2	$\mathbf{E_2}$	17
	$\mathbf{E_{14}}$	10
	$\mathbf{E_3}$	11, 18
	$\mathbf{E_{12}}$	12
Node U_3	$\mathbf{E_{13}}$	29
	$\mathbf{E_9}$	2
	$\mathbf{E_5}$	24
Node U_5	$\mathbf{E_{15}}$	32
Node U_8	$\mathbf{E_6}$	35
Node U_9	$\mathbf{E_{11}}$	13

References

[1] Y. Kong, Z. Zhong, G. Yang, X. Luo, A. K. S. Wong, and H. Zhai, "A non-parametric kernel method for CDMA2000 network indoor localization using multiple observations," in Proc. Int. Conf. Inform Netwrking., 2012, pp. 97–101.

[2] R. W. Ouyang, A. K. Wong, C. T. Lea, and M. Chiang, "Indoor location estimation with reduced calibration exploiting unlabeled data via hybrid generative discriminative learning," IEEE Trans. Mobile Comput., vol. 11, Sep. 2011, pp. 1613–1626.

[3] V. Y. Zhang, A. K. Wong, and K. T. Woo, "Histogram based particle filtering with online adaptation for indoor tracking in WLANs," Int. J. Wireless Inform. Networks, vol. 19, no. 3, 2012, pp. 239–253.

[4] M. Panzarino. What exactly Wi-Fi SLAM is, and why apple acquired it. URL: http://thenextweb.com/apple/2013/03/26/what-exactly-wifislam-is-and-why-apple-acquired-it/#!p5wMH [accessed: 2014-05-20]. (Mar. 2013)

[5] H. Shin, Y. Chon, and H. Cha, "Unsupervised construction of an indoor floor plan using a smartphone," IEEE Trans. Systems, Man, and Cybernetics-Part C: Applications andReviews, vol. 42, no. 6, Nov. 2012, pp. 889–898.

[6] M. Zhou, A. K. S. Wong, Z. Tian, Y. Zhang, X. Yu, and X. Luo, "Adaptive mobility mapping for people tracking using unlabelled Wi-Fi shotgun reads," IEEE Commun. Lett., vol. 17, no. 1, 2013, pp. 87–90.

[7] R. W. T. Ouyang, A. K. S. Wong, M. Chiang, K. T. Woo, V. Y. Zhang, H. S. Kim, and X. M. Xiao, "Energy efficient assisted GPS measurement and path reconstruction for people tracking," in Proc. IEEE GLOBECOM, 2010, pp. 1–5.

[8] E. Alpaydin, Introduction to Machine Learning, 2nd ed. Cambridge, Massachusetts, London, England: The MIT Press, 2010.

[9] K. Pearson, "Notes on regression and inheritance in the case of two parents," in the Royal Society of London, vol. 58, Jun. 1895, p. 240?242.

[10] R. Tarjan, "Depth-first search and linear graph algorithms," SIAM J. Comput., vol. 1, Jun. 1972, pp. 146–160.

[11] P. Rosenstiehl and R. E. Tarjan, "Rectilinear planar layouts and bipolar orientations of planar graphs," Discrete Comput. Geom, vol. 1, 1986, pp. 343–353.

 ISBN: 978-1-61208-360-5