Cluster-Enhanced Techniques for Pattern-Matching Localization Systems

Sheng-Po Kuo, Bing-Jhen Wu, Wen-Chih Peng, and Yu-Chee Tseng*
Department of Computer Science
National Chiao-Tung University
Hsin-Chu, 300, Taiwan
E-mail:{spkuo, sjwu, wcpeng, yctseng}@cs.nctu.edu.tw

Abstract

In location-based services, the response time of location determination is critical, especially for real-time applications. This is especially true for pattern-matching localization methods, which rely on comparing an object's current signal strength pattern against a pre-established location database of signal strength patterns collected in the training phase. In this work, we propose some cluster-enhanced techniques to speed up the positioning process while avoiding the possible positioning errors caused by this accelerated mechanism. Through grouping training locations with similar signal strength patterns together and characterizing them by a single feature vector, we show how to reduce the associated comparison cost so as to accelerate the pattern-matching process. To deal with signal fluctuations, several clustering strategies allowing overlaps are proposed. Extensive simulation studies are conducted. Experimental results show that compared to the pattern-matching systems without clustering techniques, a reduction of more than 90% in computation cost can be obtained in average without degrading the positioning accuracy.

1. Introduction

Location-based services (LBSs) have emerged as one of the killer applications for mobile computing and wireless data services. While providing great market values to business applications, such services are also critical to public safety, transportation, emergency response, and disaster management. Consequently, location estimation is essential to the success of LBSs. In addition to the well-known GPS [7], a lot of techniques have been proposed for indoor localization, such as infrared-based [18], ultrasonic-based [14], and RF-based [3, 16] systems.

Among all localization systems, the RF-based systems are probably most cost-effective because they can rely on existing wireless network infrastructures (such as IEEE 802.11 WLAN, sensor networks, and WiMAX). However, such systems need to handle the characteristic of signal strengths, which may fluctuate frequently. The *pattern-matching* schemes [3, 16, 17, 4], or known as the *fingerprinting* schemes, deal with this problem by involving two phases: *training* and *positioning*. In the training phase, given a set of training locations, the received signal strengths of all base stations (or beacons) at these locations are collected for a sufficient amount of time. Therefore, for each training location, a feature vector is calculated. Then, in the positioning phase, when an object needs to determine its location, it can compare its current received signal strengths against the feature vectors in the location database to check their similarity. The corresponding location with the most similar feature vector is selected as the possible location of the object. Sometimes, interpolation can be conducted to further improve the localization accuracy.

Recently, serval works have discussed how to apply pattern-matching localization methods to large-scale environments, such as a wireless city (typically with a size about a few square kilometers) [11, 6, 10, 8]. However, they may encounter the scalability problem due to the huge calibration efforts required in the training phase and the high comparison cost incurred in the positioning phase. For example, in a wireless city, thousands or millions of training records may have to be collected. Several efforts have been dedicated to reducing the calibration cost [6, 5, 13, 9]. In this paper, we

*Y. C. Tseng's research is co-sponsored by Taiwan MoE ATU Program, by NSC grants 93-2752-E-007-001-PAE, 96-2623-7-009-002-ET, 95-2221-E-009-058-MY3, 95-2221-E-009-060-MY3, 95-2219-E-009-007, 95-2218-E-009-209, and 94-2219-E-007-009, by Realtek Semiconductor Corp., by MOEA under grant number 94-EC-17-A-04-S1-044, by ITRI, Taiwan, by Microsoft Corp., and by Intel Corp.

aim to reduce the computation cost incurred in the positioning phase, where the computation cost is referred to the number of comparisons required for location estimations. This would enable us to support real-time LBSs.

We propose *cluster-enhanced localization techniques* which also consist of two phases. In the training phase, similar to existing pattern-matching approaches, we first collect feature vectors of training locations. Through clustering techniques, those training locations with similar feature vectors are grouped together. This results in a small number of clusters. For each cluster, a representative feature vector is derived. Then, in the positioning phase, given a signal strength vector, we first compare it against all clusters' representative feature vectors and pick the one whose representative feature vector is most similar to the given signal strength vector. Finally, only the training locations in the selected cluster are further evaluated to determine the estimated location of the object.

Although the clustering technique is able to reduce the computation cost, its positioning accuracy may be degraded if the right cluster is not selected. If a false cluster is selected, the final location estimation will be incorrect. We refer to this as the *false cluster selection*. Apparently, the probability of choosing a false cluster should be reduced. In this paper, we first show that the traditional k-means algorithm [12] is not suitable when the effect of noise is not negligible. Then we propose two clustering strategies to enhance the k-means algorithm by allowing clusters to have overlapping members. Although having duplications is redundant, it can effectively reduce the events of false cluster selection due to noises. To verify our results, a simulation model is built and extensive simulation studies are conducted. Experimental results show that our system is able to reduce at least 90% computation cost without sacrificing positioning accuracy compared to the case without clustering techniques.

The rest of this paper is organized as follows. Section 2 reviews some former works. The proposed cluster-enhanced localization system is described in Section 3. Section 4 presents several clustering strategies. Section 5 contains our performance studies. Section 6 concludes this paper.

2. Related Works

Several localization systems have explored the pattern-matching techniques. In [3], the nearest-neighbor algorithm is applied to search the location database for the training location with the shortest Euclidean distance in the signal space. Based on probability theory, [16] presents a probabilistic framework for localization to handle signal strength fluctuations. Reference [17] adopts the similar concept to develop recursive Bayesian filters for localization. In general, the nearest-neighbor approach is not as effective as the probabilistic one [20]. In [4], a more sophisticated network-based classification method is proposed. A neural network, which consists of multiple layers of interconnected neurons, is adopted to model the dependencies among a set of random variables. It has a forward and backward propagation mechanism to adaptively assign suitable weights to neuron connections in the training phase.

For large-scale environments, some literatures have considered the scalability issue incurred in the training phase [5,13,10] and the positioning phase [1,21]. In the training phase, to relieve the labor cost needed for collecting training data, an intuitive idea is to collect less training locations. However, this also implies that the training locations will be coarse-grained, thus decreasing positioning accuracy. Hence, [5] proposes to generate a small number of virtual training locations from the actual ones by interpolation techniques. Similarly, multidimensional regression [13] is used to build a nonlinear mapping between the signal space and the physical space. To compensate the loss of accuracy caused by reduced training locations, [5] suggests to use unlabeled user traces, where an unlabeled user trace is a sequence of continuously received signal strength measurements without location labels. With the help of a hidden Markov model, unlabeled user traces can be used to simplify the training process while keeping a certain degree of positioning accuracy. In [3], authors suggest to establish the training database solely based on a signal propagation model, so the training phase has no human labor. However, such an approach is not appropriate for indoor localization because signals have complex propagation paths.

The issue of reducing the comparison cost in the positioning phase is discussed in [1, 21]. The main idea is to apply some clustering techniques to training locations, so only a subset of them needs to be searched. Reference [1] constructs clusters according to the physical coordinates of training locations. It claims that the estimated locations of two consecutive location queries should be very close in the physical space. Thus, only the training locations close to the previous estimated one need to be searched for the current query. However, it does not consider the actual signal space and its searching range strongly relies on the query interval and the user mobility model. It is more related to object tracking, but in this work we focus more on localization. In [21], training locations that see the q strongest

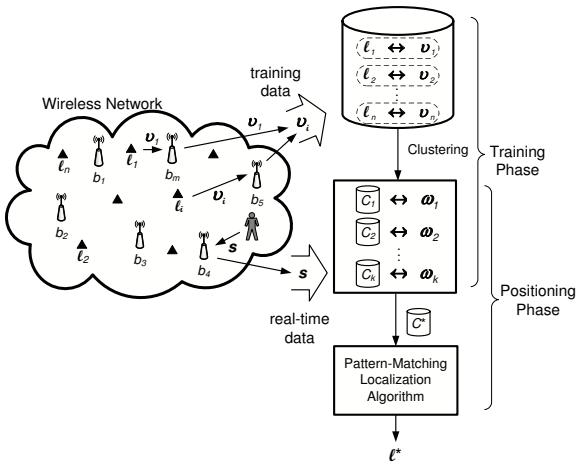

Figure 1. The cluster-based localization system.

signal strengths from the same q access points (APs) are grouped together. This clustering technique is simple but has several drawbacks. First, the top q APs with the strongest signals at a fixed location may vary over time, thus being prone to cause false cluster selection. Second, the number of clusters is not a controllable parameter. In our work, the number of clusters is tunable and the probability of false cluster selection can be effectively reduced.

3. The Cluster-Enhanced Pattern-Matching Localization System

The proposed clustering system can be applied to most pattern-matching localization methods. It also consists of two phases: training and positioning. Fig. 1 depicts the structure of the system.

3.1. The Training Phase

We assume that there is a set of m beacons $\mathcal{B} = \{b_1, b_2, \ldots, b_m\}$ being deployed in the field. These beacons can transmit RF signals for localization purpose. In this field, we define a set of n training locations $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_n\}$. Let the feature space be $\mathcal{F} \in \mathbb{R}^m$, where $\mathbb{R}$ is the set of possible received signal strengths. From each training location ℓ_i, $i = 1..n$, we collect a sufficient number of training samples from beacons and calculate a *feature vector* $\boldsymbol{v}_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,m}] \in \mathcal{F}$ for ℓ_i, where $v_{i,j}$ is the average received signal strength from b_j at ℓ_i. For those training locations with similar feature vectors, we exploit clustering techniques to group them together. Specifically, we will compute k location sets $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$ such that $\mathcal{C}_i \subseteq \mathcal{L}$, $i = 1..k$, and $\bigcup_{i=1}^{k} \mathcal{C}_i = \mathcal{L}$. Associated with each location set $\mathcal{C}_i$, its representative feature vector is expressed by $\boldsymbol{\omega}_i = [\omega_{i,1}, \omega_{i,2}, \ldots, \omega_{i,m}] \in \mathcal{F}$, where $\omega_{i,j} = \frac{\sum_{\forall x:\ell_x \in \mathcal{C}_i} v_{x,j}}{|\mathcal{C}_i|}$ is the average signal strength of beacon b_j. Note that two location sets may have overlaps, i.e., $\mathcal{C}_i \cap \mathcal{C}_j$ is not necessarily an empty set. The detail clustering algorithms will be given in Section 4.

3.2. The Positioning Phase

In the positioning phase, when an object needs to determine its location, we can measure its signal strength vector $\boldsymbol{s} = [s_1, s_2, \ldots, s_m]$, where s_j is the signal strength of b_j. Our goal is to determine the object's location in a real-time manner. In typical pattern-matching methods, $\boldsymbol{s}$ will be compared to all n feature vectors in the location database. However, in a large-scale sensing field (such as a wireless city), thousands or millions of vectors may need to be compared. By clustering training locations with similar feature vectors into groups, we only need to compare $\boldsymbol{s}$ against the representative feature vector $\boldsymbol{\omega}_i$ of each $\mathcal{C}_i$ first. As in most works, the similarity between $\boldsymbol{s}$ and $\mathcal{C}_i$ is defined as the reciprocal of the Euclidean distance between $\boldsymbol{s}$ and $\boldsymbol{\omega}_i$, i.e., $\mathrm{sim}(\boldsymbol{s}, \mathcal{C}_i) = \frac{1}{\|\boldsymbol{s}, \boldsymbol{\omega}_i\|} = \frac{1}{\sqrt{\sum_{j=1}^{m}(s_j - \omega_{i,j})^2}}$. Then, the most similar cluster, denoted by $\mathcal{C}^*$, is selected, i.e., $\mathcal{C}^* = \arg\max_{\mathcal{C}_i} \mathrm{sim}(\boldsymbol{s}, \mathcal{C}_i)$. Then, only the training locations in $\mathcal{C}^*$ will be further searched. We will adopt the Nearest Neighbor in Signal Space (NNSS) algorithm [3] to conduct the search. In NNSS, users' locations are estimated by comparing $\boldsymbol{s}$ against the feature vector of each training location ℓ_i in $\mathcal{C}^*$ according to their Euclidean distance. The estimated location is $\ell^* = \arg\max_{\ell_i \in \mathcal{C}^*} \mathrm{sim}(\boldsymbol{s}, \ell_i)$, where $\mathrm{sim}(\boldsymbol{s}, \ell_i) = \frac{1}{\|\boldsymbol{s}, \boldsymbol{v}_i\|} = \frac{1}{\sqrt{\sum_{j=1}^{m}(s_j - v_{i,j})^2}}$. Therefore, the computation cost can be decreased from $O(|\mathcal{L}|)$ (if all locations are searched) to $O(k + \frac{|\mathcal{L}|}{k})$ (if the location database is well partitioned).

4. Clustering Algorithms

Below, we propose several clustering algorithms to partition the location database. We start with the well-known k-means algorithm, followed by two enhanced clustering strategies.

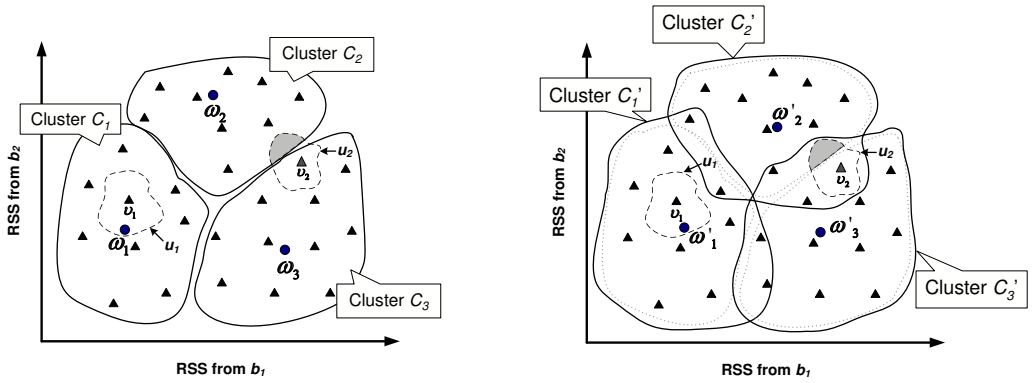

Figure 2. (a) An example of the influence of signal fluctuations on the k-means clustering algorithm (RSS = received signal strength). (b) An example of clusters allowing overlaps.

4.1. k-means Algorithm

The k-means algorithm developed in [12] can be directly applied to our system. The algorithm has multiple iterations. In the x-th iteration, we will form k clusters $\mathcal{C}_1^{(x)}, \mathcal{C}_2^{(x)}, \ldots, \mathcal{C}_k^{(x)}$. Initially, we construct k seeds $\boldsymbol{\omega}_1^{(0)}, \boldsymbol{\omega}_2^{(0)}, \ldots, \boldsymbol{\omega}_k^{(0)} \in \mathcal{F}$, where each seed $\boldsymbol{\omega}_i^{(0)}$, $i = 1..k$, is randomly selected from the set of feature vectors $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\}$, and $\boldsymbol{\omega}_i^{(0)} \neq \boldsymbol{\omega}_j^{(0)}$ for all $i \neq j$. (Other ways to choose these seeds are discussed in [19]. Here we adopt the random strategy.) With these seeds, we define the membership of cluster $\mathcal{C}_i^{(1)}$ in the first iteration as follows:

$$\ell_j \in \mathcal{C}_i^{(1)} \Leftrightarrow \boldsymbol{\omega}_i^{(0)} = \arg\min_{\boldsymbol{\omega}_y^{(0)}} \{\|\boldsymbol{v}_j, \boldsymbol{\omega}_y^{(0)}\|\}.$$

That is, ℓ_j will be categorized as a member of cluster $\mathcal{C}_i^{(1)}$ if $\boldsymbol{v}_j$ is closest to the seed $\boldsymbol{\omega}_i^{(0)}$ among all seeds. For each cluster $\mathcal{C}_i^{(1)}$, $i = 1..k$, we then calculate a new seed $\boldsymbol{\omega}_i^{(1)}$ by averaging the received signal strengths for all $\ell_j \in \mathcal{C}_i^{(1)}$, where for each $y = 1..m$

$$\omega_{i,y}^{(1)} = \frac{\sum_{\forall j: \ell_j \in \mathcal{C}_i^{(1)}} v_{j,y}}{|\mathcal{C}_i^{(1)}|}.$$

In the x-th iteration, for all $x \geq 2$, according to the seeds generated in the $(x-1)$-th iteration, we define cluster $\mathcal{C}_i^{(x)}$ as follows:

$$\ell_j \in \mathcal{C}_i^{(x)} \Leftrightarrow \boldsymbol{\omega}_i^{(x-1)} = \arg\min_{\boldsymbol{\omega}_y^{(x-1)}} \{\|\boldsymbol{v}_j, \boldsymbol{\omega}_y^{(x-1)}\|\}.$$

Similarly, from each cluster $\mathcal{C}_i^{(x)}$, we can calculate another seed $\boldsymbol{\omega}_i^{(x)}$, where $\omega_{i,y}^{(x)} = \frac{\sum_{\forall j: \ell_j \in \mathcal{C}_i^{(x)}} v_{j,y}}{\|\mathcal{C}_i^{(x)}\|}$ for each $y = 1..m$. The regrouping processes will be repeated iteratively until the condition $\mathcal{C}_i^{(x)} = \mathcal{C}_i^{(x+1)}$ is satisfied for all $i = 1..k$. At last, we set $\mathcal{C}_i = \mathcal{C}_i^{(x)}$ and let its feature vector be $\boldsymbol{\omega}_i = \boldsymbol{\omega}_i^{(x)}$.

Ideally, in the positioning phase, when an object provides its current signal strength vector $\boldsymbol{s}$, we would expect that a correct cluster with the most similar feature vector can be selected. However, due to the fluctuation of radio signal, this cannot always be achieved. Fig. 2(a) shows an example with three clusters in a feature space. Each triangle represents the feature vector of a training location. Due to signal fluctuation, the signal strength vector $\boldsymbol{s}$ of an object may appear in a wide range of area, such as the dotted shapes u_1 and u_2, which represent the areas of the signal strengths possibly received at locations ℓ_1 and ℓ_2, respectively. It is shown that the range of u_1 falls safely in the scope of the cluster $\mathcal{C}_1$ in $\mathcal{F}$, but the range of u_2 falls in both the scope of clusters $\mathcal{C}_2$ and $\mathcal{C}_3$ in $\mathcal{F}$. If unfortunately the signal strength vector $\boldsymbol{s}$ received at ℓ_2 falls in the shaded region, a false cluster selection happens. Clearly, this should be avoided, especially when an object is near the boundaries of clusters.

4.2. Clustering Techniques Allowing Overlaps

The problem in Fig. 2(a) calls for the design of clusters with certain degrees of overlapping. For example, three new clusters $\mathcal{C}_1, \mathcal{C}_2$, and $\mathcal{C}_3$ which overlap each other are shown in Fig. 2(b). Below, we propose two clustering schemes extended from the k-means algorithm to allow a training location to join multiple clusters. The training location ℓ_2 in Fig. 2(b), for instance,

will join both clusters $\mathcal{C}'_2$ and $\mathcal{C}'_3$. We define *overlapping degree* of a training location to be the number of clusters that it can join. Clearly, this will increase the searching complexity in the positioning phase to $O(k + \lambda \times \frac{|\mathcal{L}|}{k})$ in average, where λ is the average overlapping degree.

After performing the k-means algorithm, we will apply our overlapping strategies to $\mathcal{L}$, which will generate k new sets with a certain degree of overlapping. The main difference between these overlapping strategies is the way to determine which clusters a training location should join. In the first *multi-nearest-neighbor* strategy, each training location can join a fixed number of clusters with higher similarity. In the second *Voronoi-based* strategy, the overlapping degree of a training location will be determined by its geometric relation with nearby clusters.

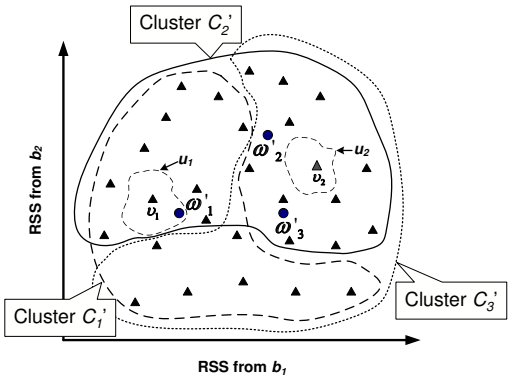

Figure 3. An example of muti-nearest-neighbor overlapping strategy with $\phi_N = 2$.

4.2.1. Multi-Nearest-Neighbor Overlapping Strategy

The multi-nearest-neighbor overlapping strategy assigns a constant overlapping degree ϕ_N to all training locations. As mentioned before, k clusters of training locations $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$ are obtained by the k-means clustering algorithm with representative feature vectors $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2 \ldots, \boldsymbol{\omega}_k$, respectively. Then, for each training location ℓ_i, it will join the top ϕ_N similar clusters. Specifically, we will construct new sets $\mathcal{C}'_j$, $j = 1..k$, such that $\ell_i \in \mathcal{C}'_j$ if and only if $\mathrm{sim}(\ell_i, \mathcal{C}_j)$ ranks in the top ϕ_N among all clusters, where $\mathrm{sim}(\ell_i, \mathcal{C}_j) = 1/\|\boldsymbol{\upsilon}_i, \boldsymbol{\omega}_j\|$. Then, the representative feature vector ω'_j of each $\mathcal{C}'_j$ is generated by averaging the received signal strengths for all $\ell_i \in \mathcal{C}'_j$. In average, the searching space is increased from $O(k + \frac{|\mathcal{L}|}{k})$ to $O(k + \frac{\phi_N \times |\mathcal{L}|}{k})$ compared to the k-means clustering algorithm.

Intuitively, this strategy takes the result of the earlier k-means scheme and allows each training location to join multiple clusters. For the example illustrated in Fig. 2(a), we can avoid incorrect location estimations caused by false cluster selection if the ℓ_2 is allowed to join two closest clusters $\mathcal{C}_2$ and $\mathcal{C}_3$ simultaneously. The result of the multi-nearest-neighbor strategy with $\phi_N = 2$ is shown in Fig. 3. Because both $\mathcal{C}'_2$ and $\mathcal{C}'_3$ contain ℓ_2, we can avoid false cluster selection whenever the signal strength vector $\boldsymbol{s}$ received at ℓ_2 is more similar to $\mathcal{C}'_2$ or $\mathcal{C}'_3$.

4.2.2. Voronoi-based Overlapping Strategy

Although the multi-nearest-neighbor overlapping strategy is simple to be implemented and easy to control the average searching space, the parameter ϕ_N is hard to determine. If ϕ_N is too small, it may not compensate for the effect of signal fluctuations. Thus, the problem of false cluster selection remains. On the other hand, if ϕ_N is too large, some training locations may join unnecessary clusters, thus causing redundancy. We can observe that when $\boldsymbol{\upsilon}_i$ of a training location is close to the center of a cluster in $\mathcal{F}$, the number of clusters it joins should not the same as another training location whose feature vector $\boldsymbol{\upsilon}_j$ is near the periphery of a cluster.

For this consideration, we next propose the Voronoi-based overlapping strategy. After performing the k-means algorithm, $\mathcal{F}$ is decomposed into k partitions centered at $\boldsymbol{\omega}_j$, $1 \leq j \leq k$. It can be observed that the distance between the cluster center $\boldsymbol{\omega}_j$ and any of its members $\boldsymbol{\upsilon}_i$, $\ell_i \in \mathcal{C}_j$, is shorter than other clusters. This property is equivalent to a Voronoi diagram [2], where all points in a Voronoi cell are closest to the Voronoi vertex in the same cell. Thus, the members of a cluster $\mathcal{C}_j$ are contained in a Voronoi cell V_j with a Voronoi vertex at $\boldsymbol{\omega}_i$. If we let a training location which is close to the Voronoi edge join more clusters and oppositely let the others join less clusters, we can improve the effectiveness of the overlapping technique.

Motivated by the observation above, we propose the Voronoi-based overlapping strategy. Specifically, a feature vector close to the Voronoi edge joins more than one cluster. For each pair of neighboring Voronoi cells V_x and V_y, we formally define an overlapping region $R_{x,y}$ ($x < y$) in which any training location whose feature vector located joins both $\mathcal{C}_x$ and $\mathcal{C}_y$. For example, in Fig. 4(a), there are three Voronoi cells V_1, V_2, and V_3, separated by three Voronoi edges $e_{1,2}$, $e_{2,3}$, and $e_{1,3}$, and three overlapping regions $R_{1,2}$, $R_{1,3}$, and $R_{2,3}$, shaded. The feature vector $\boldsymbol{\upsilon}_1$ is inside $R_{2,3}$ and

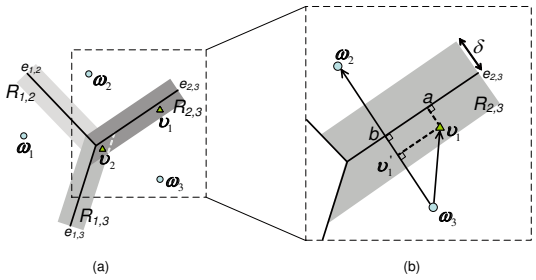

Figure 4. An example of Voronoi-based overlapping strategy.

$\boldsymbol{\upsilon}_2$ is located both in $R_{2,3}$ and $R_{1,3}$. As a result, ℓ_1 joins $\mathcal{C}_2$ and $\mathcal{C}_3$; while ℓ_2 joins $\mathcal{C}_1, \mathcal{C}_2$, and $\mathcal{C}_3$. An overlapping region $R_{x,y}$ can be regarded as an expansion of an Voronoi edge $e_{x,y}$ along the edges incident to the endpoints of $e_{x,y}$, like the gray region $R_{2,3}$ shown in Fig. 4(b). The expansion range δ is used to control the size of $R_{x,y}$ by expanding both sides from $e_{x,y}$.

To determine which overlapping regions where a feature vector $\boldsymbol{\upsilon}_i$ located, we have to determine the Voronoi cell V_x and V_y such that $\ell_i \in \mathcal{C}_x$ and V_x and V_y are neighbors. Let $dist(\boldsymbol{\upsilon}_i, e_{x,y})$ be the vertical distance between $\boldsymbol{\upsilon}_i$ and the Voronoi edge $e_{x,y}$. Then, if $dist(\boldsymbol{\upsilon}_i, e_{x,y}) < \delta$, then $\boldsymbol{\upsilon}_i$ is definitely in $R_{x,y}$. Hence, ℓ_i will join $\mathcal{C}_y$ in the Voronoi-based overlapping strategy.

However, due to the costly computation of $e_{x,y}$ in high dimension feature space [2], we do not calculate $dist(\boldsymbol{\upsilon}_i, e_{x,y})$ directly. Instead, we use the projection of the vector $\overrightarrow{\boldsymbol{\omega}_x\boldsymbol{\upsilon}_i}$ on the line $\overline{\boldsymbol{\omega}_x\boldsymbol{\omega}_y}$ to obtain $dist(\boldsymbol{\upsilon}_i, e_{x,y})$. Again in Fig. 4(b), we want to determine $dist(\boldsymbol{\upsilon}_1, e_{2,3}) = \overline{\boldsymbol{\upsilon}_1 a}$. First, $\overrightarrow{\boldsymbol{\omega}_3\boldsymbol{\upsilon}_1}$ is projected on $\overline{\boldsymbol{\omega}_3\boldsymbol{\omega}_2}$ as $\overrightarrow{\boldsymbol{\omega}_3\boldsymbol{\upsilon}_1'}$. Let b be an intersection point of $\overline{\boldsymbol{\omega}_3\boldsymbol{\omega}_2}$ and $e_{2,3}$. The edge $e_{2,3}$ and $\overline{\boldsymbol{\omega}_3\boldsymbol{\omega}_2}$ are mutually orthogonal, which is a property of a Voronoi diagram. Therefore, the points $\boldsymbol{\upsilon}_1$, $\boldsymbol{\upsilon}_1'$, b, and a form a rectangle so $\overline{\boldsymbol{\upsilon}_1' b} = \overline{\boldsymbol{\upsilon}_1 a}$. Finally, $\overline{\boldsymbol{\upsilon}_1' b}$ can be obtained by $\overline{\boldsymbol{\omega}_3 b} - \overline{\boldsymbol{\omega}_3\boldsymbol{\upsilon}_1'} = \overline{\boldsymbol{\omega}_2\boldsymbol{\omega}_3}/2 - \overrightarrow{\boldsymbol{\omega}_3\boldsymbol{\upsilon}_1} \cdot \overrightarrow{\boldsymbol{\omega}_3\boldsymbol{\omega}_2}/\|\overrightarrow{\boldsymbol{\omega}_3\boldsymbol{\omega}_2}\|$. Compared with finding $\overline{\boldsymbol{\upsilon}_1 a}$ directly, this method saves more computation cost.

The above procedure functions well based on the assumption that each Voronoi cell knows the neighborhood information. For example, V_3 knows V_1 and V_2 are its neighbors in Fig. 4(b). Unfortunately, we cannot obtain this information until the relationship between Voronoi cells is completely discovered. This is as hard as finding Voronoi edges. Note that the k-means clustering algorithm only finds out the Voronoi vertex of each cell. Here, we propose a simple speculation technique, called *neighborhood speculation*, to guess the neighborhood relationship. It is based on an observation that if two Voronoi cells V_x and V_y are neighbors, then the midpoint of $\overline{\boldsymbol{\omega}_x\boldsymbol{\omega}_y}$ is *usually* closer to V_x and V_y than any other cell. Therefore, we use the position of the midpoint of V_x and V_y to speculate the relationship between them. If the midpoint is inside other cells except for V_x or V_y, we tend to believe that V_x and V_y are not neighbors.

5. Simulations

In this section, we conduct some experiments to evaluate the performance of the cluster-enhanced system. We study the impact of varying parameters used in our system.

5.1. Simulation Model

We consider a 100×100 square meters sensing field. Since larger environment results in larger differences between different methods, we simply perform reasonably scaled simulations to observe the trend of the results. Eight beacons are placed at $(0, 0)$, $(0, 99)$, $(99, 0)$, $(0, 50)$, $(50, 0)$, $(50, 99)$, $(99, 50)$, and $(99, 99)$, respectively. As to other 9992 grid points, we collect 200 training samples at each of them in the training phase. The *log-distance path loss model* is exploited to model the signal propagation given by [15]:

$$PL(d) = PL(d_0) + 10\alpha log(\frac{d}{d_0}) + N(0, \sigma), \quad (1)$$

where $d_0 = 1$ is the reference distance, and d is the distance between the transmitter and the receiver. The term α denotes the path loss exponent, typically from 2 to 6, and $N(0, \sigma)$ is a zero-mean normal distributed random variable with a standard deviation σ. The received signal strengths are generated by $P_t - PL(d)$, where P_t denotes the transmit power. Also, P_t is set to be 15 dBm, $PL(d_0) = 37.3$, $\alpha = 2$, and $\sigma = 4$.

To evaluate the system performance, three performance metrics are employed:

- *Positioning error*: The error distance between the estimated location and the true location is the positioning error. We will use this metric to evaluate our cluster-enhanced system with other pattern-matching methods.
- *Hit rate*: To get insight into the impact of clustering on localization, the hit rate is defined as the probability of accurately predicting the cluster containing the true location. Obviously, the higher the hit rate, the smaller the positioning error.

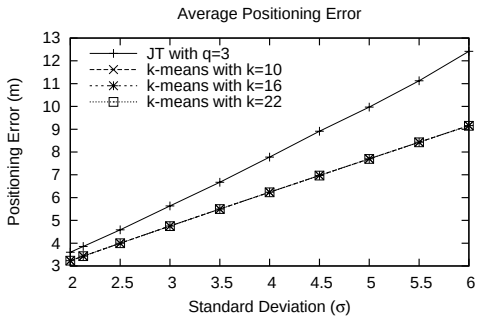

Figure 5. The comparison of the average positioning error under different standard deviation σ.

- *Average cluster size*: This metric stands for the improvement on computation reduction. According to our cluster-enhanced localization system, the average number of comparison would be $O(k + \phi \times |\mathcal{L}|/k)$. Since our proposed clustering strategies have the same number of clusters, we should further derive the average number of training locations in clusters (i.e., $\sum_{i=1}^{k} |\mathcal{C}_i|/k$).

We evaluate the following clustering techniques: k-means algorithm, the Joint Clustering (abbreviated as *JC*) technique in [21], the multi-nearest-neighbor (abbreviated as *MNN*) strategy and the Voronoi-based (abbreviated as *Voronoi*) strategy. As to the *LOCATOR* proposed in [1], because it requires user mobility model and is more applicable to the tracking systems, we do not compare it in our simulation. A good clustering strategy should have a higher hit rate, a smaller average cluster size, and a lower positioning error.

The above performance metrics are measured by the following steps. First, we use the the signal propagation model mentioned in Eq. (1) to simulate 200 test samples of each grid point and then apply a clustering technique for each sample to determine its nearest cluster. A hit event occurs when the cluster contains the corresponding location of the sample.

5.2. Impact of Clustering on the Average Positioning Error

We first investigate the impact of clustering on the average positioning error. To demonstrate that clustering will also guarantee the accuracy of location estimation, we only compare the baseline clustering method (i.e. k-means algorithm) with the JC algorithm. We vary the standard deviation σ in the path loss model and show the effectiveness of clustering. Note that JC only generates $14 \sim 18$ clusters in our simulation and thus we set the number of clusters for k-means algorithms to 10, 16 and 22, respectively. Fig. 5 shows the experimental results. In Fig. 5, JC incurs larger average positioning error under different noise levels. On the other hand, our system is able to provide better performance than that of JC in terms of average positioning error. Note that under the same σ, the average positioning errors of k-means algorithms ($k = 10, 16, 22$) are very close to each other.

5.3. Sensitive Performance Study for Clustering Strategies

From the above experiment, our system with the baseline clustering algorithm (i.e., k-mean algorithm) outperforms the existing algorithm (i.e., JC). Before comparing k-means with other proposed strategies, we first conduct sensitive performance study for clustering strategies so as to determine the optimal parameter for each one. The number of clusters for each clustering strategy is set to 50, 100 and 200, respectively. Fig. 6 shows the performance study of the two clustering strategies with their parameters varied (ϕ_N in MNN and δ in Voronoi). Note that to show their difference, we will compare these clustering strategies in terms of the hit rate and the average cluster size. It can be seen in Fig. 6(a) and Fig. 6(c), the hit rates of MNN and Voronoi have similar trend. However, Fig. 6(b) and Fig. 6(d) reveal that the Voronoi strategy can have smaller average cluster size in a lower density environment ($k = 50$). In other words, Voronoi is more suitable for a sparse environment. This agrees with our claim that Voronoi can effectively avoid training locations joining unnecessary clusters.

5.4. Performance Comparison of Clustering Strategies

In light of sensitive performance studies in Section 5.3, we select MNN with $\phi_N = 4$ and Voronoi with $\delta = 5$ to further compare their performance with the varied noise degree. The performance study of these clustering strategies is shown in Fig. 7. Fig. 7(a) shows that the hit rates of these two strategies decrease as noise degree increases. However, the hit rates of MNN and Voronoi are very high when $\sigma \leq 3.5$. It is worth mentioning that without the overlapping technique, k-means performs worst in terms of hit rates. However, from the result in Fig. 7(b), k-mean has the smallest average cluster size. If the latency caused by positioning is more important, this algorithm is a good choice because smaller

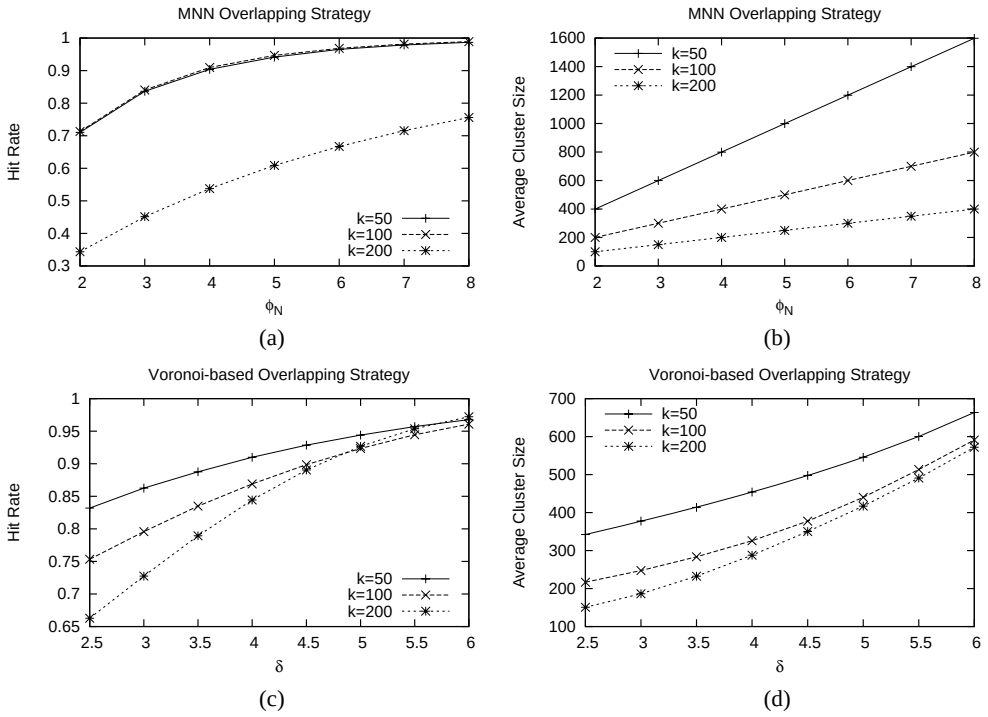

Figure 6. Sensitivity performance studies for proposed clustering strategies.

average cluster size implying shorter latency. Besides, both MNN and Voronoi have reasonable average cluster size in the environment with larger noise degree.

5.5. Performance Study of Reduced Comparison Cost

The number of clusters will impact on the reduced computation cost compared to the pattern-matching localization system without clustering techniques. Hence, we further conduct some experiments by varying the number of clusters. The experimental result is shown in Fig. 8, where the reduced computation cost is represented by the percentage of the reduced number of comparisons in average compared to the system without clustering techniques (total number of comparisons $= 10,000$). In this experiment, each strategy guarantees that the hit rate is larger than 0.85. From Fig. 8, when $k = 100$, the reduced cost of Voronoi and is more than that of MNN. It also shows that when $k \geq 100$, the reduced costs of these two methods can be more than 90%.

6. Conclusion

In this paper, we presented efficient cluster-enhanced techniques to speed up the pattern-matching localization algorithms. With the aid of clustering techniques, the training data can be divided into several groups based on their similarity defined in the specific feature space. Then, we select the cluster which is most similar to the real-time received sample and only search locations in it. Considering the problem of potential location errors caused by false cluster selection, two overlapping clustering strategies are proposed. Our performance evaluation shows that the proposed overlapping strategies can greatly improve the hit rate of the clustering technique and reduce at least 90% computation cost compared to the pattern-matching systems without clustering techniques.

References

[1] A. Agiwal, P. Khandpur, and H. Saran. LOCATOR: Location Estimation System for Wireless LANs. In *ACM WMASH*, pages 102–109, 2004.

[2] F. Aurenhammer. Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An In-building RF-based User Location and Tracking System. In *IEEE INFOCOM*, volume 2, pages 775–784, 2000.

[4] R. Battiti, T. L. Nhat, and A. Villani. Location-aware Computing: A Neural Network Mmodel for Determining Location in Wireless LANs. Technical Report DIT-

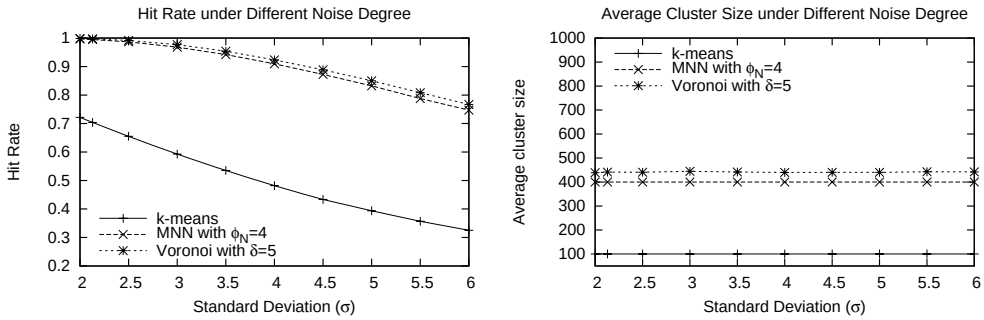

Figure 7. Effect of the standard deviation σ on the (a) hit rate and (b) average cluster size.

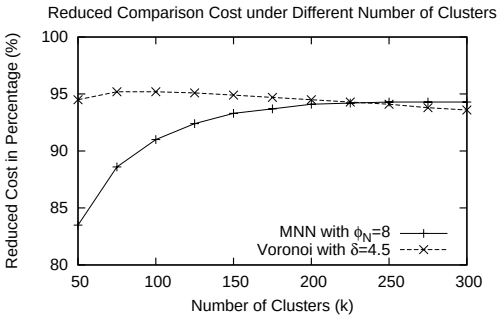

Figure 8. The reduced comparison cost compared to the pattern-matching localization systems without clustering.

02-0083, Universita di Trento, Dipartimento di Informatica e Telecomunicazioni, 2002.

[5] X. Chai and Q. Yang. Reducing the Calibration Effort for Location Estimation Using Unlabeled Samples. In *IEEE PERCOM*, pages 95–104, 2005.

[6] Y.-C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm. Accuracy Characterization for Metropolitan-scale Wi-Fi Localization. In *ACM MOBISYS*, volume 5, pages 233–245, 2005.

[7] P. Enge and P. Misra. Special Issue on Global Positioning System. *Proc. IEEE*, 87(1):3–15, 1999.

[8] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical Robust Localization over Large-scale 802.11 Wireless Networks. In *IEEE/ACM MOBICOM*, pages 70–84, 2004.

[9] P. Krishnan, A. S. Krishnakumar, W.-H. Ju, C. Mallows, and S. Ganu. A System for LEASE: Location Estimation Assisted by Stationary Emitters for Indoor RF Wireless Networks. In *IEEE INFOCOM*, volume 2, pages 1001–1011, 2004.

[10] A. LaMarca, J. Hightower, I. Smith, and S. Consolvo. Self-Mapping in 802.11 Location Systems. In *Proc. 7th Int'l Conf. on Ubiquitous Computing (UBICOMP)*, pages 87–104. Springer, 2005.

[11] J. Letchner, D. Fox, and A. LaMarca. Large-Scale Localization from Wireless Signal Strength. In *Proc. of the Nat'l Conf. on Artificial Intelligence (AAAI)*, pages 15–20, 2005.

[12] J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297, 1967.

[13] J. J. Pan, J. T. Kwok, Q. Yang, and Y. Chen. Multidimensional Vector Regression for Accurate and Low-Cost Location Estimation in Pervasive Computing. *IEEE Trans. on Knowledge and Data Engineering*, 18(9):1181–1193, 2006.

[14] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-support System. In *IEEE/ACM MOBICOM*, pages 32–43. ACM Press New York, NY, USA, 2000.

[15] T. S. Rappaport. *Wireless Communications: Principles and Practice*. IEEE Press Piscataway, NJ, USA, 1996.

[16] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen. A Probabilistic Approach to WLAN User Location Estimation. *Int'l Journal of Wireless Information Networks*, 9(3):155–164, 2002.

[17] V. Seshadri, G. V. Záruba, and M. Huber. A Bayesian Sampling Approach to In-door Localization of Wireless Devices Using Received Signal Strength Indication. In *IEEE PERCOM*, pages 75– 84, 2005.

[18] R. Want, A. Hopper, V. Falcão, and J. Gibbons. The Active Badge Location System. *ACM Trans. on Information Systems (TOIS)*, 10(1):91–102, 1992.

[19] R. Xu and D. W. II. Survey of Clustering Algorithms. *IEEE Trans. on Neural Networks*, 16(3):645–678, 2005.

[20] M. Youssef and A. Agrawala. On the Optimality of WLAN Location Determination Systems. In *Comm. Networks and Dist. Syst. Modeling and Simulation Conf.*, 2004.

[21] M. Youssef, A. Agrawala, and U. Shankar. WLAN Location Determination via Clustering and Probability Distributions. In *IEEE PERCOM*, pages 143–150, 2003.