# 國立交通大學

## 資訊科學與工程研究所

## 碩 士 論 文

偵測在無線感測網路下定位系統的感測器移動問題

The Beacon Movement Detection Problem in Wireless Sensor Networks for Localization Applications

研 究 生：郭曉儒

指導教授：曾煜棋　教授

中 華 民 國　九 十 五　年 六 月

# The Beacon Movement Detection Problem in Wireless Sensor Networks for Localization Applications

Student: Hsiao-Ju Kuo
Advisor: Prof. Yu-Chee Tseng

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer Science

July 2006

Hsinchu, Taiwan

# 偵測在無線感測網路下定位系統的感測器移動問題

學生：郭曉儒　　指導教授：曾煜棋教授

國立交通大學資訊科學與工程學系（研究所）碩士班

## 摘　　要

定位是無線感測網路下的關鍵性議題。在大部分的定位架構下，感測區域上會擺放著一些信標感測器以作為決定感測區域上發現的物體或事件的位置的參考位置。一個潛在的假設是這些信標一直都是靜止的。在這篇論文裡，我們定義了一個新的偵測信標移動問題(BMD)。假設在這個系統裡，有一些信標的位置被改變了而且未被注意到，這個問題關心的是要如何自動地監控到這種情況並且識別出這些信標。將這些信標從定位引擎中移除有很大的可能可以提升定位準確度。我們提出了四個架構來解決這個BMD問題。最後，在假設一些信標有未被注意到的移動情形下，我們會估算這些方法可以如何地改善定位準確度。根據模擬結果顯示，我們的方法可以減輕53%的由異常信標移動導致的定位準確度惡化狀況。

**關鍵字**：環境感知，定位服務，普及計算，定位，無線感測網路

# The Beacon Movement Detection Problem in Wireless Sensor Networks for Localization Applications

Student: Hsiao-Ju Kuo Advisor: Prof. Yu-Chee Tseng

Department of Computer Science
National Chiao-Tung University

ABSTRACT

Localization is a critical issue in wireless sensor networks. In most localization schemes, there are beacons being placed as references to determine the positions of objects or events appearing in the sensing field. The underlying assumption is that beacons are always static. In this work, we define a new *Beacon Movement Detection (BMD)* problem. Assuming that there are unnoticed changes of locations of some beacons in the system, this problem is concerned about how to automatically monitor such situations and identify these beacons. Removal of such beacons in the localization engine may improve the localization accuracy. Four schemes are proposed to solve the BMD problem. Finally, we evaluate how these solutions can improve the accuracy of localization schemes in case that there are unnoticed movement of some beacons. Simulation results show that our solutions alleviate $53\%$ the decrease of positioning accuracy caused by the exceptional beacon movement.

**Keywords:** Context Awareness, Location-Based Service, Pervasive Computing, Positioning, Wireless Sensor Network.

# Acknowledgements

My advisor, prof. Yu-Chee Tseng, is the first one I would like to express my gratitude to. With the wonderful research conditions he provided and his attentive instructions, I came to discover the pleasure of research. I am also grateful to my senior, Sheng-Po Kuo. Without his help and suggestions, I would not be able to have this thesis done. Finally, I would like to thank all HSCC members for their generous advice. Discussing with them benefited me in many ways.

Hsiao-Ju at CS, NCTU.

# Contents

# List of Figures

# Chapter 1

# Introduction

Recently, we have seen significant progress in the areas of wireless ad hoc and sensor networks. Ad hoc networking technologies enable quick and flexible deployment of a communication platform. A wireless sensor network typically adopts the ad hoc network architecture and is capable of exploiting context information collected from sensors. Many applications of wireless sensor networks have been proposed [3, 5, 6].

Sensor networks are promising in supporting context-aware and location-aware services. The success of this area may greatly benefit human life. One essential research issue in sensor networks is *localization*, whose purpose is to determine the position of an object or event. For example, the sentient system Bat [2] is composed of a set of sensors for 3D localization purpose. Sensors are installed at known positions such as ceilings, to measure the signal traveling time from a user badge to them. Then, the location of the badge is calculated by a triangulation algorithm. Localization by signal's angle of arrival is addressed in [9, 10]. In [10], ultrasonic sensors are used to estimate the location and orientation of a mobile device with the Cricket compasses. In [1], a distributed positioning system called AHLoS (Ad Hoc Localization System) is proposed, where some beacons are aware of their own locations while others are not. The former are used to determine the positions of the latter. A similar work based on a probability model is proposed in [11]. The RADAR system [4] uses machine learning and pattern-matching techniques to estimate the locations of WiFi-enable mobile devices.

In all the above localization systems, there are a set of *beacon sensors* (or simply *beacons*), which are at fixed locations and periodically send out or receive short broadcast packets to estimate other objects' locations by either triangulation or pattern-matching schemes. Based on such an infrastructure, this paper points out a new *Beacon Movement Detection* (BMD) problem that may happen in most localization systems based on beacons. This problem is concerned about how to automatically determine the unexpected

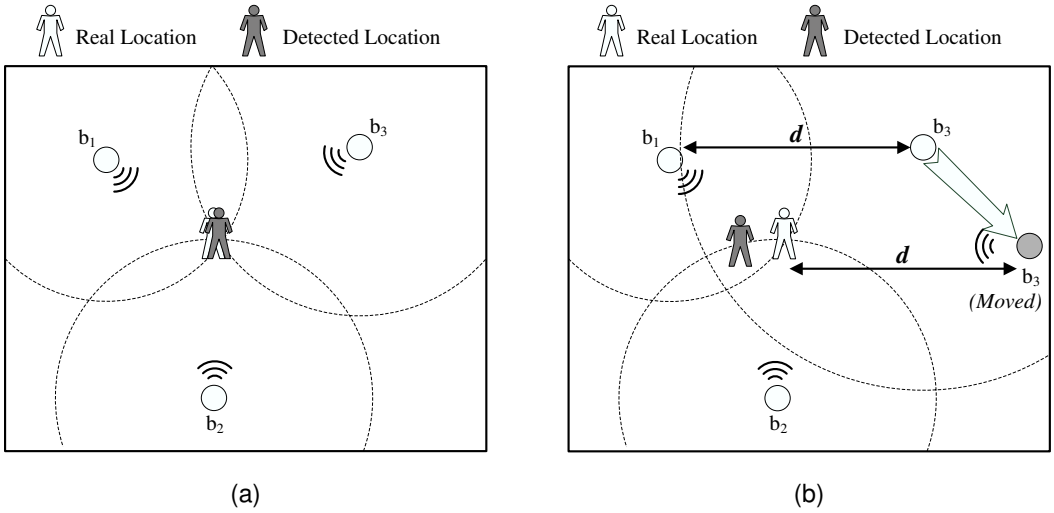



Figure 1.1: An example of the Beacon Movement Detection (BMD) Problem.

change of locations of some beacons in the system. Movement of some beacons may affect the accuracy of the localization results. For example, in Fig. 1.1(a), we show how three beacons determine a target's position in typical triangulation approaches. However, if beacon $b_3$ is moved to the location marked in gray without being noticed, the system may incorrectly estimate the target's location as shown in Fig. 1.1(b). Note that the circle centered at $b_3$ has a radius equal to the distance from the real location of $b_3$ to the target. Also note that the results proposed in this paper is applicable to not only unnoticed movement of beacons, but also unnoticed appearance of interference/obstacles in the sensing field, which may affect the localization results.

The BMD problem involves two issues. First, we need to determine those beacons that are unexpectedly relocated. Second, the result has to be forwarded to the positioning engine to improve the localization accuracy. To solve the first issue, we will allow beacons to monitor each other to determine those moved beacons. We will propose four schemes. In the first *neighbor-based* scheme, beacons will keep track of their nearby beacons and report their observations to the *BMD engine* to determine if some beacons have been moved. In the second *signal-strength-variation* scheme, the change of signal strengths of beacons will be exploited. In the third *signal-strength-summation* scheme, the BMD engine will collect the sum of reported signal strength changes of each beacon to make decisions. The last *location-based* scheme tries to calculate each beacon's current location and compares the result with its predefined location to decide if it has been moved. The first scheme is easy to implement but not very accurate. The second and third schemes also have low complexity but perform much better. The last scheme has high complexity,

but is quite sensitive to noise. We also present some limitations and characteristics of these schemes.

The remainder of this paper is organized as follows: Section 2 gives a formal definition of the BMD problem. Section 3 presents our solutions to the BMD problem. We then evaluate the proposed algorithms and examine their capability to improve the localization accuracy in Section 4. Finally, Section 5 concludes on this work.

# Chapter 2

# Problem Definition

We are given a sensing field, in which a set of beacons $B = \{b_1, b_2, ..., b_n\}$ are deployed for localization purpose. Periodically, each beacon will broadcast a `HELLO` packet. To determine its own location, an object will collect `HELLO` packets from its neighboring beacons and send a signal strength vector $S = \langle s_1, s_2, ..., s_n \rangle$ to an external positioning engine, where $s_i$ is the signal strength of the `HELLO` packet from $b_i$ ($s_i = 0$ if it cannot hear $b_i$). The positioning engine can then estimate the object's location based on $S$ (for example, in the case of RADAR [4], $S$ is compared against a location database obtained in the training phase based on a finger-printing method).

Suppose that a set of unreliable beacons $B_M \subset B$ are moved or blocked by obstacles without being noticed. The *Beacon Movement Detection (BMD)* problem is to compute a detected set $B_D$ that is as similar to $B_M$ as possible. The result $B_D$ may be used to calibrate the positioning engine to reduce the localization error (for example, in the case of RADAR, the entry $s_i$ in $S$ may be ignored if $b_i$ is detected to be unreliable).

To solve the BMD problem, we will enforce beacons to monitor each other from time to time. Let's denote the local *observation vector* of $b_i$ at time $t$ by $O_i^t = \langle o_{i1}^t, o_{i2}^t, ..., o_{in}^t \rangle$, where $o_{ij}^t$ is $b_i$'s observation on $b_j$ at time $t$. The content of an observation will depend on the corresponding BMD scheme (refer to Section 3). We use the observation vector at time $t = 0$ to represent the original observation when no beacon is moved. The *observation matrix* is denoted by $O^t = \langle O_1^t, O_2^t, ..., O_n^t \rangle^T$. Given $O^t$, the BMD engine is responsible of calculating a set $B_D$. The result is then sent to the calibration algorithm in the positioning engine. Fig. 2.1 illustrates our system model.

Considering the following reasons, we define the *tolerable region* $R_i$ of each beacon $b_i$ as the geographic area within which movement is acceptable. First, radio signal tends to fluctuate from time to time. Second, ignoring the data of a beacon in the location database will decrease the localization accuracy due to less beacons helping the localization

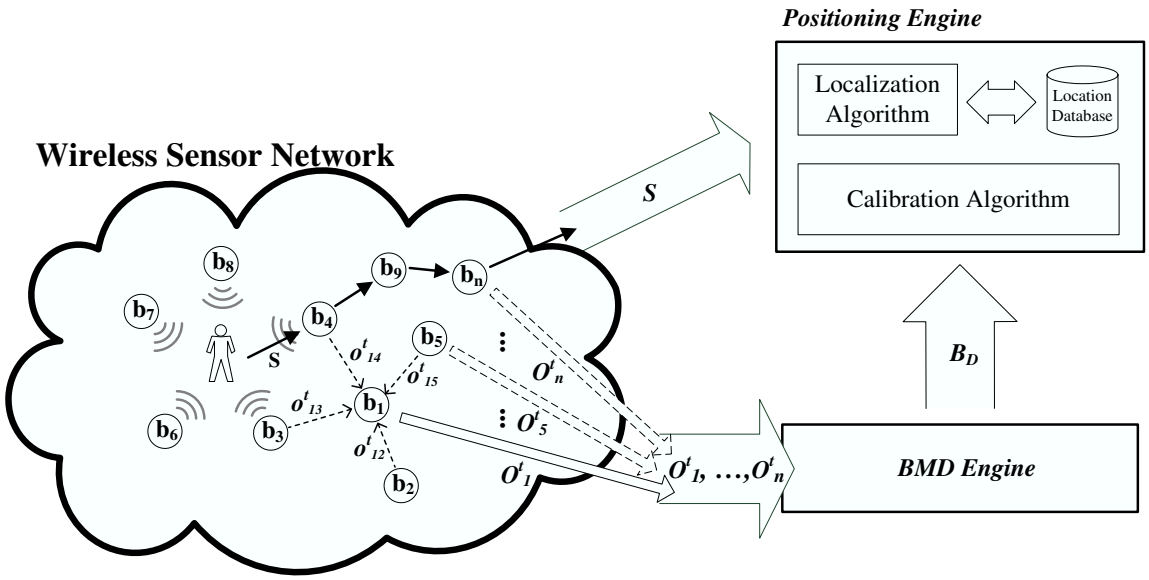



Figure 2.1: The system model.

procedures. So the slight movement activities should be omitted. As a result, the unreliable set $B_M$ only contains those beacons which are moved out of their tolerable regions. The size of tolerable regions is application-dependent, which is beyond the scope of this work. For simplicity, tolerable regions are assumed to be circles centered at beacons of the same radius.

Ideally, we would expect $B_M = B_D$. However, for many reasons this cannot be achieved. For ease of discussion, we define two events. A *hit event* is obtained on a beacon $b_i$ if $b_i \in B_M$ and the BMD engine also determines that $b_i \in B_D$. A false event is obtained on $b_i$ if $b_i \notin B_M$ but $b_i \in B_D$.

# Chapter 3

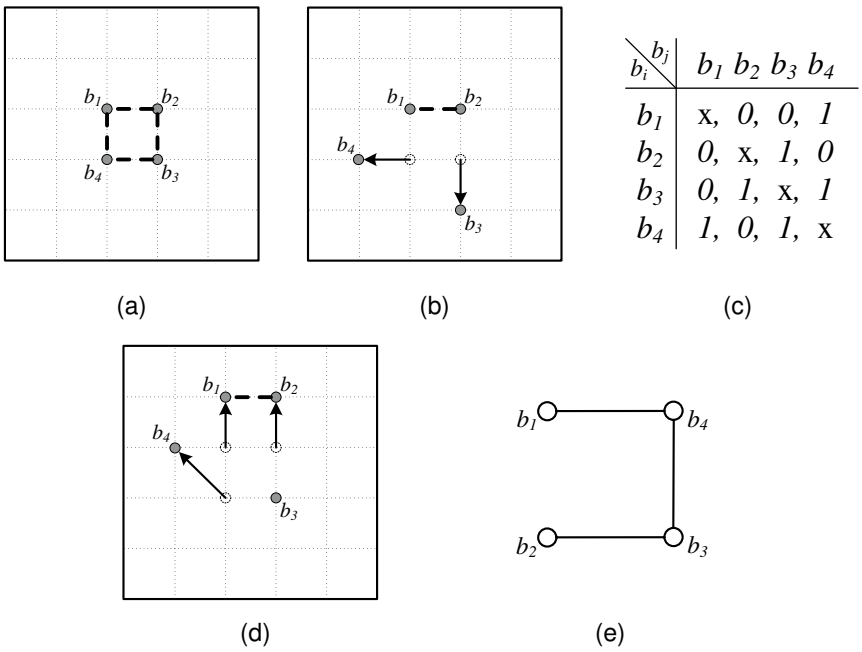
# Beacon Movement Detection Algorithms

To solve the BMD problem, we propose four detection schemes, namely *neighbor-based*, *signal-strength-variation*, *signal-strength-summation*, and *location-based* schemes. These schemes differ in the local processing rule of beacons and the decision algorithm at the BMD engine. In the neighbor-based scheme, each beacon locally decides if some neighboring beacons have moved into or out of their communication coverage range and reports its observation to the BMD engine. The signal-strength-variation scheme is similar to the neighbor-based scheme, but the definition of movement is according to a threshold of signal strength change. In the signal-strength-summation scheme, a beacon does not try to determine whether a neighboring beacon has been moved or not. Instead, each beacon reports the amount of signal strength change of each neighbor; the sums of all reported values are used by the BMD engine to make a global decision. In the location-based scheme, each beacon reports its actually received signal strengths, which are used by the BMD engine to compute each beacon's current location and to compare against its original location.

## 3.1 Neighbor-Based (NB) Scheme

In this scheme, each beacon $b_i$ monitors the change of neighborhood relations with other beacons in its coverage area. The neighborhood relation of $b_i$ at time $t$ is defined as

$$n_{ij}^t = \begin{cases} 1, & \text{if } b_i \text{ can hear } b_j \\ 0, & \text{otherwise.} \end{cases}$$




| $b_i$ \ $b_j$ | $b_1$ | $b_2$ | $b_3$ | $b_4$ |
|---|---|---|---|---|
| $b_1$ | x, | 0, | 0, | 1 |
| $b_2$ | 0, | x, | 1, | 0 |
| $b_3$ | 0, | 1, | x, | 1 |
| $b_4$ | 1, | 0, | 1, | x |

Figure 3.1: An example of BMD problem in the neighbor-based scheme. (a) the original relation, (b) a movement scenario, (c) observation matrix $O^t$, (d) another movement scenario, and (e) the observation graph $G_O$.

Let $n_{ij}^0$ be the original neighborhood relation when the system was first configured. Then the observation $o_{ij}^t$ of $b_i$ on $b_j$ at time $t$ is

$$o_{ij}^t = n_{ij}^t \otimes n_{ij}^0 = \begin{cases} 1, & b_j \text{ moves into or} \\ & \text{out of } b_i\text{'s coverage} \\ 0, & \text{otherwise,} \end{cases}$$

where $\otimes$ is the "exclusive or" operator. An example with four beacons is shown in Fig. 3.1(a), where the coverage of each beacon is a circle of radius 1. Initially, each beacon is in the coverage of two neighboring beacons. Suppose that at time $t$ beacons $b_3$ and $b_4$ are moved as shown in Fig. 3.1(b). Suppose that the tolerable region defines that each beacon can not move more than 1 grid length. Then, the observation matrix $O^t$ is as shown in Fig. 3.1(c).

Unfortunately, given an observation matrix $O^t$, it is possible to come up with multiple beacon movement scenarios that result in the same $O^t$. For example, the movement scenario in Fig. 3.1(d) also has the same observation matrix in Fig. 3.1(c). In fact, we can prove a stronger result that such ambiguity always exists.

**Definition 1** *An observation matrix $O^t$ obtained in the NB scheme is* ambiguous *if there exist two different movement scenarios $B_M$ and $B'_M$ such that (i) both $B_M$ and $B'_M$ result*

*in the same $O^t$ and (ii) $B_M \cap C(O^t) \neq B'_M \cap C(O^t)$, where $C(O^t)$ is the* candidate set *such that $C(O^t) = \{b_j | O^t[i,j] = 1$ or $O^t[j,i] = 1, 1 \leq i \leq n, 1 \leq j \leq n\}$ and $C(O^t) \neq \emptyset$.*

The above condition (ii) is to ensure that there is non-trivial difference between $B_M$ and $B'_M$. Each beacon in $C(O^t)$ is detected by at least one other beacon.

**Theorem 1** *Given any movement scenario $B_M$ and its corresponding observation matrix $O^t$ obtained in the NB scheme, we can always find another movement scenario $B'_M$ such that $O^t$ is ambiguous.*

*Proof:* Given any $B_M$ and its corresponding $O^t$, we can easily compute $B_M \cap C(O^t)$. To construct another $B'_M$, we first pick any beacon $b_k \in B_M \cap C(O^t)$ and move all beacons in $B_M - \{b_k\}$ to their new locations as specified in the movement scenario $B_M$. Let the corresponding observation matrix of yet-to-be-constructed movement scenario $B'_M$ be $\hat{O}^t$. We shall show that $O^t = \hat{O}^t$. For the time being, for any beacons $b_i$ and $b_j \in B$ such that $b_i \neq b_k$ and $b_j \neq b_k$, we can derive that $\hat{O}^t[i,j] = O^t[i,j]$.

Next, suppose that in the movement scenario $B_M$, beacon $b_k$ is moved from location $l_1$ to $l_2$. Let the moving vector $\vec{v} = l_2 - l_1$. Then, we move all beacons except $b_k$ (i.e., $B - \{b_k\}$) by the vector $-\vec{v}$. Such movements will not change the entries $O^t[i,j]$ and $\hat{O}^t[i,j]$ for all $i \neq k$ and $j \neq k$. Also, these movements will not change the relative locations of $b_i$ and $b_k$ for all $b_i \in B - \{b_k\}$, i.e., $\hat{O}^t[k,i] = O^t[k,i]$ and $\hat{O}^t[i,k] = O^t[i,k]$ for all $i$. Clearly, the new movement scenario will lead to $\hat{O}^t = O^t$. Furthermore, $b_k \in B_M \cap C(O^t)$ and $b_k \notin B'_M$, which implies that $b_k \notin B'_M \cap C(\hat{O}^t)$, so this theorem is proved. ■

An example of the proof of Theorem 1 is in Fig. 3.1(d). Let $B_M$ be 3the movement scenario in Fig. 3.1(b). To construct $B'_M$, $b_3$ is kept unchanged and $b_4$ is moved as scheduled. Then $b_1$, $b_2$, and $b_4$ are moved in the direction $(0,1)$ (the reverse of $b_3$'s moving vector, $(0,-1)$). This shows that the matrix $O^t$ in Fig. 3.1(c) is ambiguous.

Clearly, the above ambiguity property prohibits us from finding the exact $B_M$ given any $O^t$. The following derivation will rely on the assumption that the unreliable beacons are only a small proportion of all beacons. This is reasonable in practice. Hence, we try to select a set $B_D$ that contains as few beacons as possible. First, we transform matrix $O^t$ to an undirected *observation graph* $G_O = (V, E)$, where $V = C(O^t)$ and $E = \{(b_i, b_j) | O^t[i,j] = 1$ and $O^t[j,i] = 1\}$. Second, observe that if $O^t[i,j] = 1$ and $O^t[j,i] = 1$, then at least one of $b_i$ and $b_j$ has been moved. Therefore, the problem can be regarded as a *vertex cover* problem [7], whose goal is to find the smallest set $V' \subseteq V$ such that for each $(b_i, b_j) \in E$, $b_i \in V'$ or $b_j \in V'$. For example, Fig. 3.1(e) represents the observation graph of the $O^t$ in Fig. 3.1(c).

The first algorithm, called *Enumerate-NB*, is only presented here for reference purpose. From graph $G_O$, we first construct all minimum vertex covers (since this problem is NP-complete, this step could be very costly). Among all solutions, the one with the lowest *stability* is selected, where the stability of a beacon $b_i$ is

$$stability(b_i) = \frac{NPos(i)}{Pos(i)},$$

where $NPos(i)$ is the set of neighboring beacons of $b_i$ that positively report that $b_i$ has not been moved, i.e., $NPos(i) = \{b_j | n^0_{ji} = 1, o^t_{ji} = 0\}$, and $Pos(i)$ is the set of all beacons that positively report that $b_i$ has been moved, i.e., $Pos(i) = \{b_j | o^t_{ji} = 1\}$. The stability level of a vertex cover is the sum of stability values of all beacons in the cover set. Then the vertex cover with the lowest stability level is selected as our $B_D$.

Considering the above algorithm is quite costly when the problem scales up, the second algorithm *Greedy-NB* adopts a heuristic approach as follows. If a beacon $b_i$'s degree in $G_O$ is higher, it is more suspicious to be moved. So the algorithm sorts the vertices in $G_O$ according to their degrees of uncovered edges in a descending order, and then examines them one by one. A node is included in $B_D$ if any edge incident to it has not been covered.

## 3.2 Signal-Strength-Variation (SSV) Scheme

In the neighbor-based scheme, we only consider the neighborhood relations between beacons. Assuming that beacons can measure the signal strengths of `HELLO` packets from their neighbors, the signal-strength-variation scheme asks each beacon $b_i$ to evaluate the amount of signal strength change of each neighboring beacon $b_j$. Let the observed signal strength by $b_i$ on $b_j$ at time $t$ be $s^t_{ij}$ ($t = 0$ means the initial observed signal strength). The observation $o^t_{ij}$ of $b_i$ on $b_j$ is

$$o^t_{ij} = \begin{cases} 1, & \text{if } (s^t_{ij} - s^0_{ij} \geq \delta^+_{ij} \text{ or } s^0_{ij} - s^t_{ij} \geq \delta^-_{ij}) \text{ or} \\ & (b_j \text{ moves into or out of } b_i\text{'s coverage}) \\ 0, & \text{otherwise,} \end{cases}$$

where $\delta^+_{ij}$ and $\delta^-_{ij}$ are the pre-defined thresholds of signal strength variations.

The thresholds $\delta^+_{ij}$ and $\delta^-_{ij}$ of each beacon $b_i$ can be determined by the tolerable region $R_j$ of $b_j$. Let locations $p$ and $q$ be the farthest and nearest locations in $R_j$ with respect to $b_i$ (refer to Fig. 3.2). If the expected signal strengths of `HELLO` packets from a beacon at $p$ and $q$ are $s_p$ and $s_q$, respectively, then $\delta^-_{ij} = s^0_{ij} - s_p$ and $\delta^+_{ij} = s_q - s^0_{ij}$. As can be seen, as long as $b_j$ moves within the belt-like gray region, $b_i$ will not report a movement event.

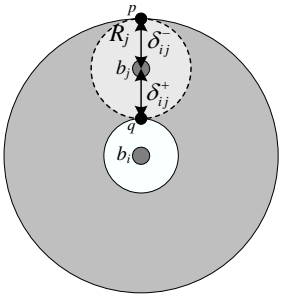



Figure 3.2: Determining thresholds $\delta_{ij}^+$ and $\delta_{ij}^-$ by the tolerable region $R_j$ of $b_j$ in the signal-strength-variation scheme.

The major difference between the neighbor-based scheme and the signal-strength-variation scheme is the calculation of local observation. However, the ambiguity property still holds.

**Definition 2** *An observation matrix $O^t$ obtained in the SSV scheme is* ambiguous *if there exist two different movement scenarios $B_M$ and $B'_M$ such that (i) both $B_M$ and $B'_M$ result in the same $O^t$ and (ii) $B_M \cap C(O^t) \neq B'_M \cap C(O^t)$, where $C(O^t)$ is the* candidate set *such that $C(O^t) = \{b_j | O^t[i,j] = 1$ or $O^t[j,i] = 1, 1 \leq i \leq n, 1 \leq j \leq n\}$ and $C(O^t) \neq \emptyset$.*

**Theorem 2** *Given any movement scenario $B_M$ and its corresponding observation matrix $O^t$ obtained in the SSV scheme, we can always find another movement scenario $B'_M$ such that $O^t$ is ambiguous.*

*Proof:* The proof is similar to that of Theorem 1. Given $B_M$, we can construct another movement scenario $B'_M$ in a similar way. Still, we can prove that (i) for any beacons $b_i$ and $b_j \in B$ such that $i \neq k$ and $j \neq k$, $\hat{O}^t[i,j] = O^t[i,j]$, and (ii) for all $i \neq k$, we can derive that $\hat{O}^t[k,i] = O^t[k,i]$ and $\hat{O}^t[i,k] = O^t[i,k]$. To prove (i), we move all beacons in $B_M - \{b_k\}$ to their new locations as specified in the original movement scenario. To prove (ii), we move all beacons except $b_k$ by an opposite moving vector of the original moving vector of $b_k$. After these movements, the relative positions of beacons are the same as that in the movement scenario $B_M$. Hence, $s_{ij}^t$ equals to the new observed signal strength $s_{ij}^{t\prime}$ in $B'_M$. Besides, the threshold $\delta_{ij}^+$ and $\delta_{ij}^-$ for each pairs $b_i$ and $b_j$ only depend on their tolerable regions and the initial deployment, so their observation matrices will be identical. ■

Based on changes of signal strengths, we can also develop two BMD algorithms called *Enumerate-SSV* and *Greedy-SSV*, which perform exactly the same as *Enumerate-NB* and *Greedy-NB*, respectively, except that the observations are computed by each beacon by a

different criteria. So we omit the details. However, with more accurate information, these algorithms are expected to be more accurate than the earlier ones in the neighbor-based scheme.

## 3.3 Signal-Strength-Summation (SSS) Scheme

Similar to the signal-strength-variation scheme, the signal-strength-summation scheme also assumes that beacons can measure the signal strengths from their neighboring beacons. However, in this scheme, the values of signal strength variations observed by a beacon will be reported to the BMD engine directly without any further processing. Specifically, the observation $o_{ij}^t$ of $b_i$ on $b_j$ at time $t$ is

$$o_{ij}^t = \left| s_{ij}^t - s_{ij}^0 \right|.$$

To avoid the effect of slight signal fluctuation problem, we will first filter out those small values in the observation, i.e., if $O^t[i,j] < \xi$ and $O^t[j,i] < \xi$, we will let $O^t[i,j] = O^t[j,i] = 0$, where $\xi$ is a threshold value. Further, we will filter out observations on a beacon $b_i$ if the summation of signal strength changes observed by other beacons is below a threshold. That is, if $\sum_{j=1}^{n} O^t[j,i] < \eta_i$, we will set $O^t[j,i] = 0$ for all $j$, where $\eta_i$ is a threshold related to the tolerable region $R_i$ of $b_i$.

Next, we will convert the problem to the *minimum weight vertex cover problem* [8]. We define an undirected weighted observation graph $G_O = (V, E)$, where $V = \{b_i | \sum_{j=1}^{n} O^t[j,i] \neq 0\}$ and $E = \{(b_i, b_j) | O^t[i,j] \neq 0 \text{ or } O^t[j,i] \neq 0\}$. The *suspicion degree* of beacon $b_i$ is defined as $w_s(b_i) = \sum_{j=1}^{n} O^t[j,i]$. The maximum suspicious degree is $w_s^* = \max_{i=1..n}\{w_s(b_i)\}$. A weight function $w : V \rightarrow R^+$ is then defined for each $b_i \in V$ such that $w(b_i) = w_s^* - w_s(b_i)$. The minimum weight vertex cover problem is to find a vertex cover $V' \subseteq V$ such that if $(b_i, b_j) \in E$, then $b_i \in V'$ or $b_j \in V'$ or both, and the sum $\sum_{b_i \in V'} w(b_i)$ is minimized.

To summarize, we have converted our BMD problem to the minimum weight vertex cover problem. We then define two algorithms: *Enumerate-SSS* and *Greedy-SSS*. The first *Enumerate-SSS* algorithm adopts a brute-force search strategy to enumerate all vertex covers in $G_O$ and picks the solution with the minimum weight. The *Greedy-SSS* is a heuristic algorithm. For each beacon $b_i$, we define a cost metric $c_i = w(b_i)/UE(b_i)$, where $UE(b_i)$ is the number of uncovered edges of $b_i$. Then, the beacon with the minimum cost metric is included in our solution. Then we recompute the cost metrics of these beacons that are affected due to the selection of the above beacon and pick the next beacon with the minimum cost metric. This is repeated until all edges are covered.

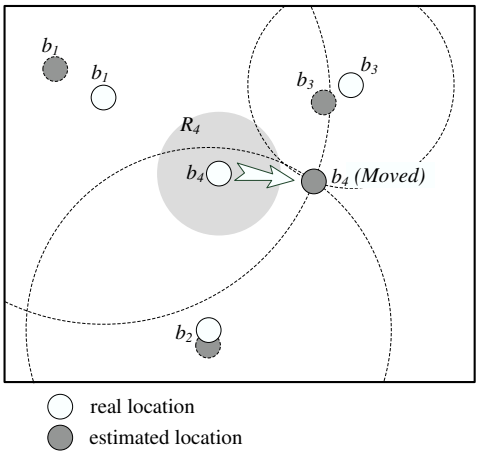



Figure 3.3: An example of movement detection in the location-based scheme where $b_4$ is the only beacon being moved. A trilateration technique is used in this example.

Note that $G_O$ may have some isolated beacons with no edges incident to them. This may happen because their neighboring beacons are not considered being moved. In this case, we will consider that these isolated beacons are likely to be moved and include them in our solution $B_D$.

## 3.4 Location-Based (LB) Scheme

The location-based scheme utilizes localization techniques to monitor the locations of beacons. Techniques such as trilateration or fingerprinting can be used in the BMD engine. Each beacon is in charge of reporting their observed signal strength values to the BMD engine. A location error threshold will be used to decide whether beacons are moved. The observation $o_{ij}^t$ of a beacon $b_i$ on $b_j$ at time $t$ is defined as $o_{ij}^t = s_{ij}^t$. We assume the BMD engine knows the initial location of each beacon. The engine then estimates the position of each beacon through any localization technique. Let the estimated location of $b_j$ at current time $t$ be $L_j^t$. If $L_j^t$ is out of the tolerable region $R_j$, then $b_j$ is determined to be unreliable.

An example using the trilateration technique is shown in Fig. 3.3. Beacon $b_4$ is moved out of its tolerable region $R_4$. Since beacons $b_1$, $b_2$, and $b_3$ are unmoved, they can help to determine $b_4$'s new location. One thing worthy of mentioning is that because of $b_4$'s movement, the estimated location of $b_1$, $b_2$, and $b_3$ may also be changed by a certain degree. So the tolerable regions need to be defined carefully. As shown by our simulation results, the location-based scheme is too sensitive to any movement, and thus does not perform well.

# Chapter 4

# Simulation Results

In this section, we present our simulation results to evaluate the proposed schemes. The performance metrics include the probabilities of hit and false events. We also use the results to calibrate the positioning engine and measure the localization error when a localization scheme is applied (refer to our system model in Fig. 2.1). Experiments are conducted under different conditions, such as the ratio of moved beacons, the maximum movement distance, and the noise degree of the environment.

## 4.1 Simulation Model

The sensing field is a 500m×500m square area. Beacons are randomly deployed on the field. Two kinds of deployment scenarios are simulated. In the *dense scenario*, $100$ beacons are randomly placed with the restriction that the distance between any two beacons is at least $5$ meters[1]. In the *sparse scenario*, $25$ beacons are randomly placed with a distance restriction of at least $20$ meters. Moved beacons are chosen randomly and a parameter *moved ratio (MR)* is used to control the number of moved beacons. The moving distance is uniformly distributed between 0 and a parameter *moved degree (MD)*. The tolerable region of each beacon is a circle centered at the beacon with a radius of $20$ and $50$ meters in dense and sparse scenarios, respectively. Note that due to the definitions of tolerable regions, only part of the moved beacons will be considered moved.

The signal propagation of `HELLO` packets are modeled by a *log-distance path loss model* [12], where the path loss of a distance $d$ is

$$PL(d) = PL(d_0) + 10\alpha log_{10}(\frac{d}{d_0}) + X_\sigma,$$

[1]The restriction is to avoid some beacons being placed too crowded, thus reducing the detection capability of the network. When a scenario is generated not satisfying the restriction, it will be discarded and we will regenerate another scenario.

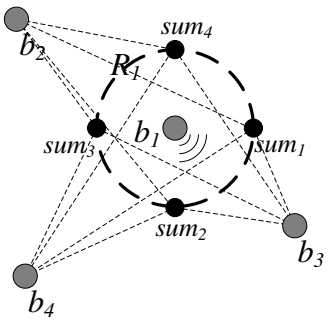



Figure 4.1: Calculation of the threshold $\eta_i$ in the signal-strength-summation scheme in our simulation.

where $d_0$ is a reference distance, $\alpha$ is the path loss exponent typically ranging from $2$ to $6$, and $X_\sigma$ is a zero-mean Gaussian random variable with a standard deviation $\sigma$. Also, the receiver sensitivity is $-100$dBm (signal lower than this value is not detectable by a receiver). The default parameter settings are: transmit power $P_t = 17$ dBm, reference path loss $PL(d_0) = 41.5$ dBm, path loss exponent $\alpha = 3.5$, and $\sigma = 2.13$.

All results are from the average of 50 experiments. To reduce the influence of noise, signal strength is calculated from the average of 10 `HELLO` packets. Therefore, the Gaussian random variable $X_{\sigma'}$ in the sampling distribution of signal strength is still zero-mean but with a standard deviation $\sigma' = \sigma/\sqrt{10}$. The noise threshold $\xi$ in the SSS scheme is set to $\xi = 2\sigma' = 2\sigma/\sqrt{10}$.

The threshold $\eta_i$ of beacon $b_i$ is calculated by an approximation as follows. On the tolerable region $R_i$, we pick four sampling points on the east, west, south, and north sides of the boundary of $R_i$. For each sampling point $p$, we measure the sum of signal strength changes observed by other beacons assuming that $b_i$ is moved to location $p$. The sum of the sampling point with the smallest value is selected as the value of $\eta_i$. The idea is illustrated in Fig. 4.1

## 4.2 Probabilities of Hit and False Events

We first make a general comparison on the *Enumerate* and *Greedy* algorithms of the first three schemes. The sparse scenario is used. The result is in Fig. 4.2. Generally, the *Enumerate* algorithms perform better than the *Greedy* algorithms, except for *Enumerate-SSS*. This is because we assume that only a small number of beacons are moved, and thus the *Enumerate* algorithms will try to find the smallest $B_D$. However, this assumption is not true any more when a lot of beacons are moved. Hence, we see that the *Enumerate-SSS* algorithm performs slightly worse than the *Greedy-SSS* algorithm. Furthermore, consid-

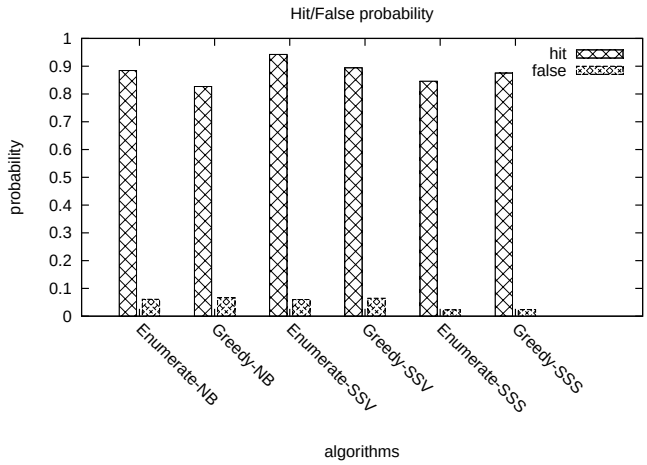



Figure 4.2: Comparison of the *Enumerate* and *Greedy* algorithms of each scheme (*MR*=0.1, *MD*=150m, sparse).

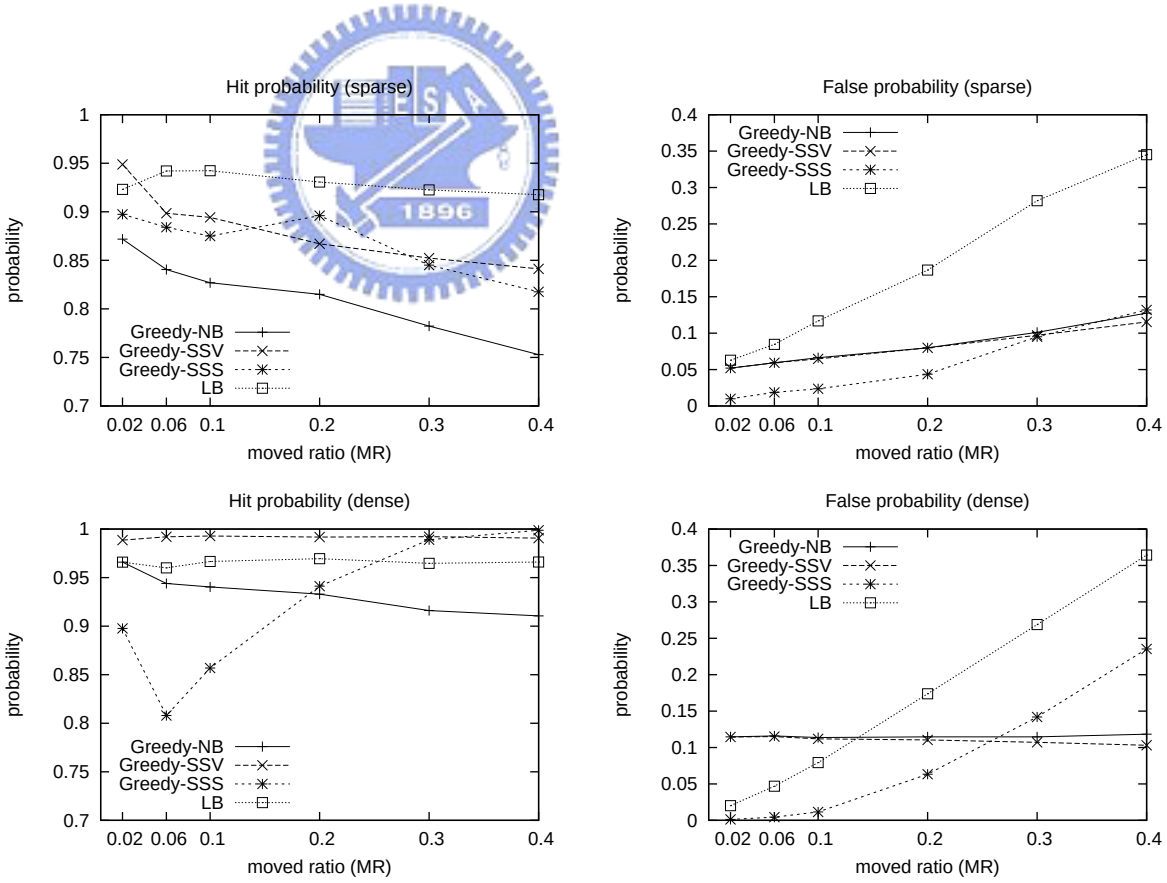



Figure 4.3: Comparison of hit and false probabilities by varying the *MR* value (*MD*=150m).

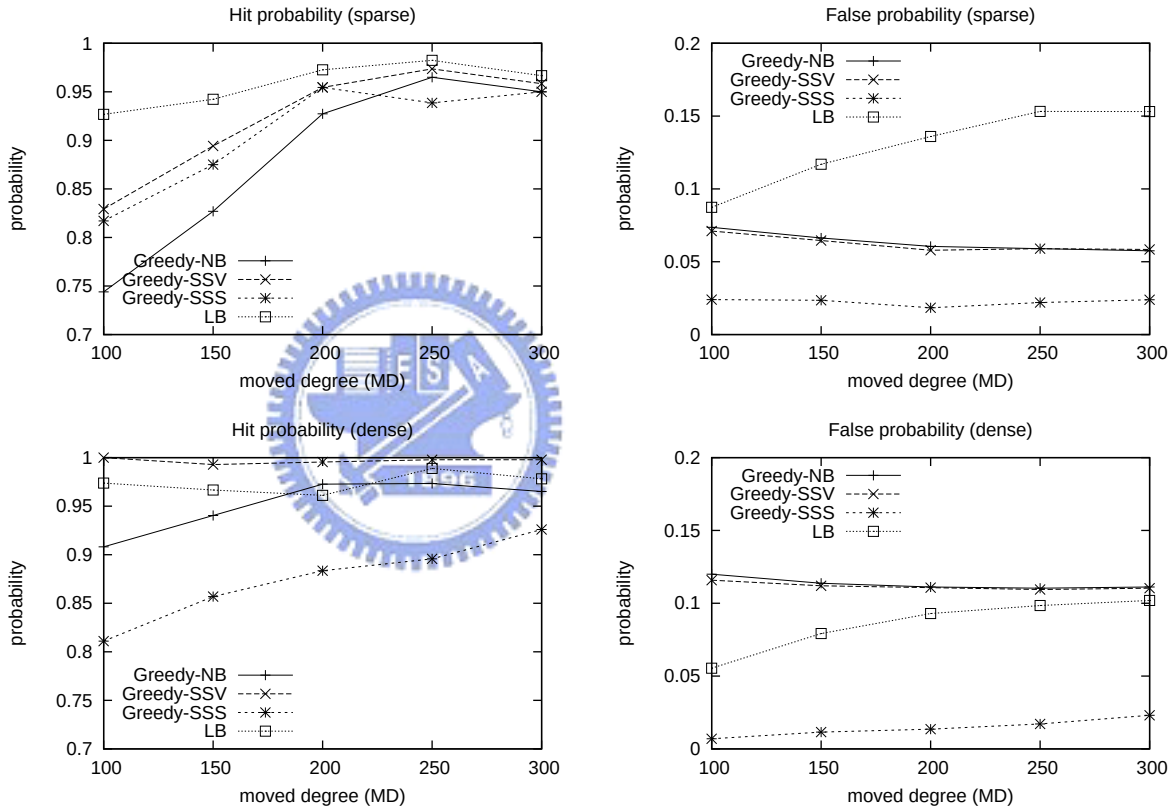



Figure 4.4: Comparison of hit and false probabilities by varying the *MD* value (*MR*=0.1).

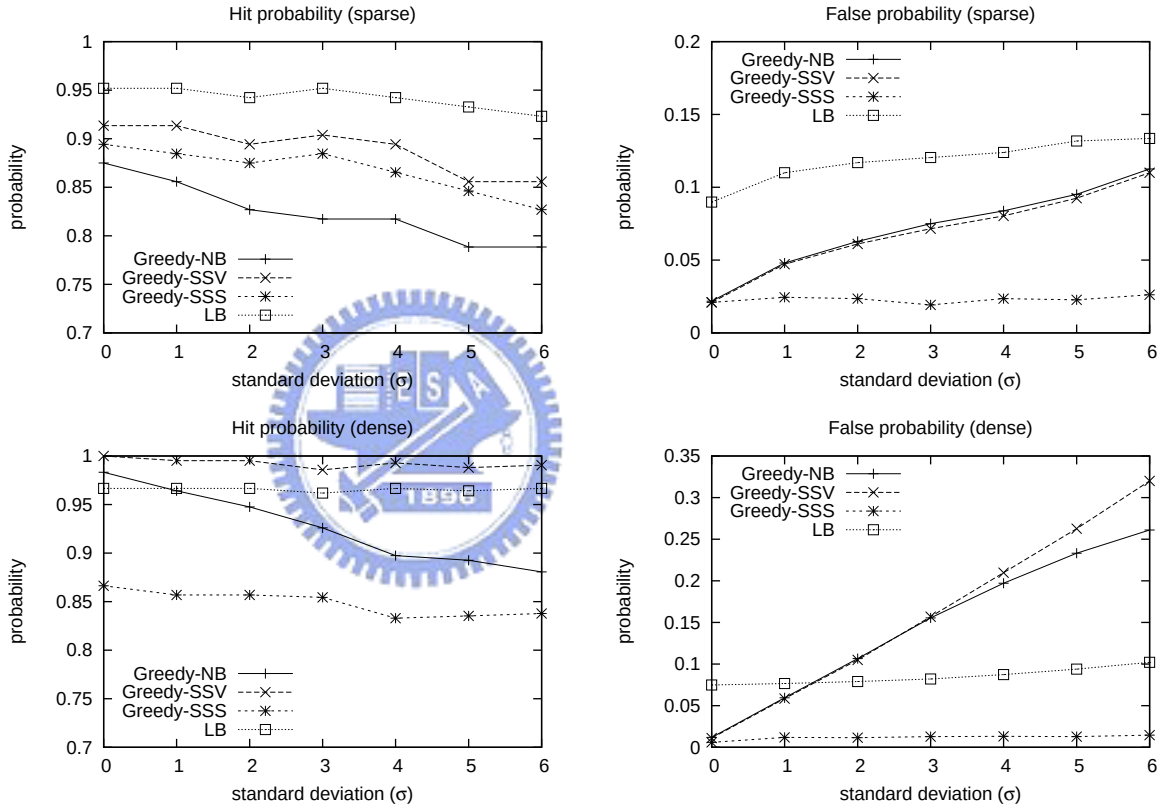



Figure 4.5: Comparison of hit and false probabilities by varying the standard deviation $\sigma$ (*MR*=0.1, *MD*=150m).

ering that the *Enumerate* algorithms are computationally infeasible, we will only adopt the *Greedy* algorithms in the rest of the discussion.

In Fig. 4.3, we vary the *moved ratio (MR)* between 0.02 and 0.4 to make the comparison. In terms of the hit probability, the *LB* algorithm performs the best, followed by the *Greedy-SSV* algorithm, the *Greedy-SSS* algorithm, and then the *Greedy-NB* algorithm. However, the *LB* algorithm also induces the highest false probability. As a result, the *Greedy-SSV* algorithm and the *Greedy-SSS* are considered the best, which provide a hit probability over 0.82 and a false probability under 0.15 even when the *MR* is 0.4. The *Greedy-NB* algorithm always has the worst hit probability due to its over-simplified detection model. The high false probability of the *LB* algorithm can be explained by its high sensitivity. Since beacons will all report their observations, a movement can easily propagate errors to its neighboring beacons, especially when the scenario is sparse. Thus a lot of reliable beacons will be reported as unreliable. The same phenomenon can also be seen for the *Greedy-SSS* algorithm when the *MR* gets higher. However, its false probability is much less than that of the *LB* algorithm. When the scenario is dense, the false probability of the *LB* algorithm can be significantly decreased due to more beacons helping the mutual detection task. Dense scenario also makes the threshold $\eta_i$ in the SSS scheme become higher and thus reduce the hit probability of the *Greedy-SSS* algorithm.

In Fig. 4.4, we vary the *moved degree (MD)*. Generally, because a larger *MD* means that each movement is more dramatic, this is beneficial for our detecting work. Therefore, we see increases of hit probabilities and decreases of false probabilities as *MD* increases in all schemes except the *LB* scheme. Again, this demonstrates that the *LB* algorithm is too sensitive. When the scenario is dense, we have similar results.

In Fig. 4.5, we vary the noise level by adjusting the standard deviation $\sigma$ from $0$ and $6$. As expected, all schemes are affected as noise level increases. Overall, the *Greedy-SSV* and the *Greedy-SSS* algorithms perform the best considering all the above factors, which is followed by the *Greedy-NB* algorithm. The *LB* algorithm is only practical when the network is dense.

## 4.3 Evaluation of Localization Accuracy

After determining the moved set $B_D$, the set will be sent to the positioning engine to calibrate the location database. In the following, we will assume the fingerprinting-based localization algorithm [4], where the location database contains the signal vector $\langle v_1, v_2, \ldots, v_n \rangle$ of each training location $l$ in the sensing field, where $v_i$ is the averaged signal strength of beacon $b_i$ observed at location $l$, $i = 1, 2, \ldots, n$. For calibration pur-

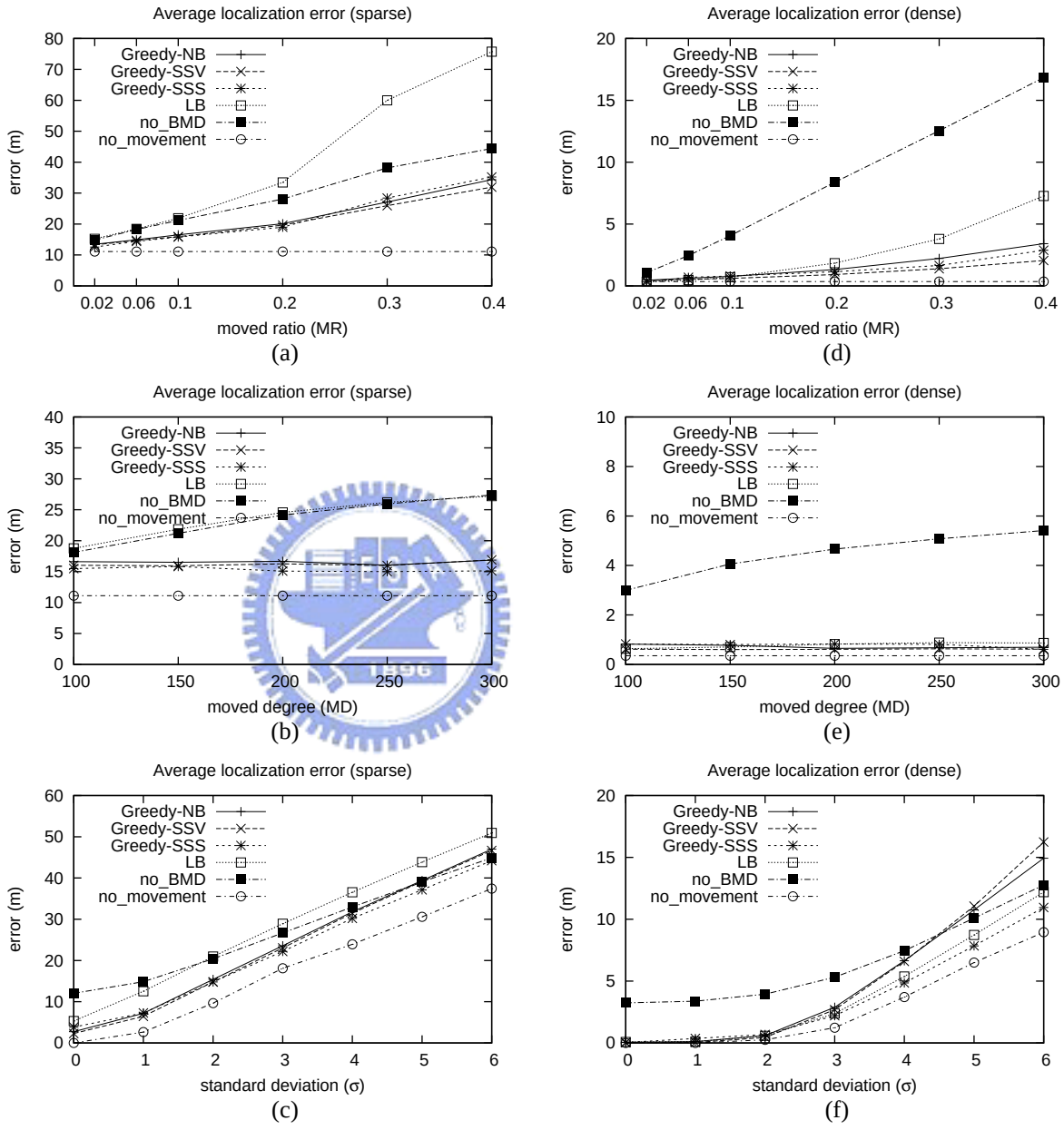



Figure 4.6: Comparison of average localization errors by varying: (a) *MR* (*MD*=150m, sparse), (b) *MD* (*MR*=0.1, sparse), (c) $\sigma$ (*MD*=150m, *MR*=0.1, sparse), (d) *MR* (*MD*=150m, dense), (e) *MD* (*MR*=0.1, dense), and (f) $\sigma$ (*MD*=150m, *MR*=0.1, dense).

pose, we will ignore the element $v_i$ for each $b_i$ that is determined to be in $B_D$ during the localization procedure. Clearly, this will reduce the accuracy of localization. However, if the moved beacons are not ignored, the error will be even higher. In the following, we will evaluate how our schemes can reduce localization errors due to moved beacons. In our experiment, there are 25×25 training points and 25×25 test points. Then, the average positioning error of the 625 test points is recorded.

We compare our results against the *no_movement* case, where no beacon is moved, and the *no_BMD* case, where there are unnoticed movements of some beacons but no special action is taken. The former is only used as a reference.

Fig. 4.6(a), (b), and (c) show the average localization errors under different *MR*, *MD*, and $\sigma$, respectively. The results demonstrate that the *Greedy-SSS* algorithm incurs errors closest to those of the *no_movement* case. The *Greedy-SSV* and *Greedy-NB* algorithms are slightly worse than the *Greedy-SSS* algorithm. Surprisingly, due to its high false probability, the *LB* algorithm's errors are quite unacceptable, sometimes even worse than the *no_BMD* case. Fig. 4.6(d), (e), and (f) show the similar simulations in the dense scenario. As the beacon density is increased a lot, we see that the *LB* and *Greedy-SSS* algorithms perform quite closely in most cases.

To model the error recovery capability of a BMD algorithm, we propose the following metric:

$$ERC(\textit{algorithm}) = \frac{\text{error}_{\text{no\_BMD}} - \text{error}_{\textit{algorithm}}}{\text{error}_{\text{no\_BMD}} - \text{error}_{\text{no\_movement}}} \times 100\%.$$

Ideally, an *ERC* of 100% is expected. However, this is unlikely to be achieved because some data are ignored in the location database. For example, when $MR = 0.1$, $MD = 150$, and $\sigma = 2.13$, the *ERC* values are 53.28%, 52.10%, and 46.47% for *Greedy-SSS*, *Greedy-SSV*, and *Greedy-NB*, respectively.

From above simulations, we can see that the *Greedy-SSS* algorithm performs well under most situations. However, its parameters $\xi$ and $\eta_i$ need to be set carefully. In some cases, *Greedy-SSV* has slightly better *ERC* than *Greedy-SSS*, but it is more sensitive to environment noise. To summarize, both the *Greedy-SSV* and *Greedy-SSS* algorithms are good choices to solve the BMD problem.

# Chapter 5

# Conclusions

The reason In this paper, we define a new beacon movement detection (BMD) problem in wireless sensor networks for localization applications. This problem describes a situation that some beacon sensors which participate in the localization procedure are moved unexpectedly. The result is a reduced localization accuracy if we disregard this situation. We propose to allow beacons to monitor each other to identify the moved beacons. Four schemes are presented to solve the BMD problem. Moreover, we have proven some impossibility theorems which will make the BMD problem unsolvable under some situations. Some heuristics are proposed by mapping the BMD problem to the vertex-cover problem. Hit and false probabilities of these heuristics are obtained through simulations. It is shown that the best heuristic, *Enumerate-SSS*, has an error recovery capability of $53\%$ in general case. As to future work, it deserves to further investigate the BMD problem if there is some trust model among beacons.

# Bibliography


[1] A. Savvides, C.C. Han, and M.B. Srivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *ACM/IEEE MOBICOM*, 2001.

[2] M. Addlesee, R. Curwen, S. Hodges, J. Newman, P. Steggles, A. Ward, and A. Hopper. Implementing a sentient computing system. *Computer*, 34(8):50 – 56, 2001.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, 2002.

[4] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *IEEE INFOCOM*, pages 775–784, 2000.

[5] J. Burrell, T. Brooke, and R. Beckwith. Vineyard computing: Sensor networks in agricultural production. *IEEE Pervasive Computing*, 3(1):38 – 45, 2004.

[6] A. Cerpa, J. Elson, D. Estrin, L. Girod, and M. Hamilton. Habitat monitoring: Application driver for wireless communications technology. In *ACM SIGCOMM Workshop on Data Communications*, Apr. 2001.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press/McGraw-Hill, 1990.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co. New York, NY, USA, 1979.

[9] D. Niculescu and B. Nath. Ad Hoc Positioning System (APS) using AoA. In *IEEE INFOCOM, San Francisco, CA.*, 2003.

[10] N. B. Priyantha, A. K. L. Miu, H. Balakrishnan, and S. J. Teller. The cricket compass for context-aware mobile applications. In *ACM/IEEE MOBICOM*, pages 1–14, 2001.

[11] V. Ramadurai and M. L. Sichitiu. Localization in wireless sensor networks: A probabilistic approach. In *Int't Conf. on Wireless Networks (ICWN)*, pages 275–281, June 2003.

[12] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall PTR, 1996.

# Curriculum Vita

Hsiao-Ju Kuo (srkuo@csie.nctu.edu.tw) received her B.S. degree in Computer Science from National Chiao-Tung University, Taiwan, in 2004. Her research interests include wireless networks and wireless sensor networks.