# ขั้นตอนวิธีการระบุตำแหน่งแบบหลายกำลังส่งในเครือข่ายรับรู้ไร้สายโดยใช้
# ซัพพอร์ตเวกเตอร์แมชชีน

นายประกิต เจริญกิตติชัย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2555

# MULTIPLE TRANSMISSION POWER LOCALIZATION ALGORITHMS IN WIRELESS SENSOR NETWORKS BASED ON SUPPORT VECTOR MACHINES

Mr. Prakit Jaroenkittichai

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering Program in Electrical Engineering
Department of Electrical Engineering
Faculty of Engineering
Chulalongkorn University
Academic Year 2012

| | |
|---|---|
| Thesis Title | MULTIPLE TRANSMISSION POWER LOCALIZATION ALGORITHMS IN WIRELESS SENSOR NETWORKS BASED ON SUPPORT VECTOR MACHINES |
| By | Mr. Prakit Jaroenkittichai |
| Field of Study | Electrical Engineering |
| Thesis Advisor | Associate Professor Ekachai Leelarasmee, Ph.D. |

---

Accepted by the Faculty of Engineering, Chulalongkorn University in Partial Fulfillment of the Requirements for the Master's Degree

..............................................Dean of the Faculty of Engineering
(Associate Professor Boonsom Lerdhirunwong, Dr.Ing.)

THESIS COMMITTEE

..............................................Chairman
(Assistant Professor Wanchalerm Pora, Ph.D.)

..............................................Thesis Advisor
(Associate Professor Ekachai Leelarasmee, Ph.D.)

..............................................External Examiner
(Supot Tiarawut, Ph.D.)

..............................................External Examiner
(Pheeradej Nanan, Ph.D.)

ประกิต เจริญกิตติชัย : ขั้นตอนวิธีการระบุตำแหน่งแบบหลายกำลังส่งในเครือข่ายรับรู้ไร้สายโดยใช้ซัพพอร์ตเวกเตอร์แมชชีน (MULTIPLE TRANSMISSION POWER LOCALIZATION ALGORITHMS IN WIRELESS SENSOR NETWORKS BASED ON SUPPORT VECTOR MACHINES) อ. ที่ปรึกษาวิทยานิพนธ์หลัก : รศ.ดร. เอกชัย ลีลารัศมี, 56 หน้า.

การระบุตำแหน่งในเครือข่ายรับรู้ไร้สายคือการประมาณตำแหน่งของโนดรับรู้ไร้สาย งานวิจัยนี้ได้นำเสนอวิธีการใช้ประโยชน์ข้อมูลจากหลายแหล่งข้อมูลสำหรับขั้นตอนวิธีการระบุตำแหน่งโดยใช้ซัพพอร์ตเวกเตอร์แมชชีน วิธีการที่นำเสนอนั้นสามารถใช้ได้ทั้งในกรณีการแบ่งหมวดหมู่และการวิเคราะห์ถดถอยของซัพพอร์ตเวกเตอร์แมชชีน โดยอาศัยเพียงข้อมูลการเชื่อมต่อซึ่งก็คือจำนวนครั้งการส่งผ่านในการส่งข้อมูลระหว่างแต่ละโนด การสร้างข้อมูลจำนวนครั้งการส่งผ่านหลายๆชุดสามารถทำได้โดยการปรับกำลังส่งของโนดรับรู้ไร้สายเพื่อเปลี่ยนระยะการติดต่อสื่อสารระหว่างโนด การเลือกระยะการติดต่อสื่อสารที่ดีที่สุดสามารถทำได้โดยการประเมินค่าสารสนเทศร่วมกันระหว่างข้อมูลจำนวนครั้งการส่งผ่านกับข้อมูลตำแหน่งของโนดรับรู้ไร้สาย เราพิจารณาสองวิธีสำหรับการรวมข้อมูลจากหลายแหล่งข้อมูลเข้าด้วยกันคือวิธีเชื่อมต่อข้อมูลโดยตรงและวิธีการให้น้ำหนักสัมพันธ์กับค่าการจัดวางแนว ผลการทดลองโดยวิธีการจำลองแสดงให้เห็นถึงความแม่นยำในการระบุตำแหน่งที่ดีขึ้นของระบบที่นำเสนอ

ภาควิชา วิศวกรรมไฟฟ้า ลายมือชื่อนิสิต........................................................................

สาขาวิชา วิศวกรรมไฟฟ้า ลายมือชื่อ อ.ที่ปรึกษาวิทยานิพนธ์หลัก..........................

ปีการศึกษา 2555

# # 5270676321 : MAJOR ELECTRICAL ENGINEERING

KEYWORDS : LOCALIZATION / MULTIPLE TRANSMISSION POWER / WIRELESS SENSOR NETWORKS / SUPPORT VECTOR MACHINES

PRAKIT JAROENKITTICHAI : MULTIPLE TRANSMISSION POWER LOCALIZATION ALGORITHMS IN WIRELESS SENSOR NETWORKS BASED ON SUPPORT VECTOR MACHINES. ADVISOR : ASSOC. PROF. EKACHAI LEELARASMEE, Ph.D., 56 pp.

Localization in wireless sensor networks is the problem of estimating the geographical locations of wireless sensor nodes. We propose a framework to utilizing multiple data sources for localization scheme based on Support Vector Machines. The framework can be used with both classification and regression formulation of Support Vector Machines. The proposed method uses only connectivity information, i.e. hop count data. Multiple hop count data sources can be generated by adjusting the transmission power of wireless sensor nodes to change the communication ranges. The optimal choice of communication ranges can be determined by evaluating mutual information between hop count data and node locations. We consider two methods for integrating multiple data sources together; unif method and align method. The improved localization accuracy of the proposed framework is verified by simulation study.

Department : Electrical Engineering ............ Student's Signature ..............................

Field of Study : Electrical Engineering ............ Advisor's Signature ..............................

Academic Year : 2012 ..............................

# ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor, Assoc. Prof. Dr. Ekachai Leelarasmee, for his patient guidance and excellent advice for my thesis research.

Special thanks to my colleagues at ESID Laboratory for the interesting and helpful discussions on many engineering topics.

I would also like to sincerely thank Telecommunications Research and Industrial Development Institute (TRIDI) for providing the scholarship for my Master’s Degree.

Finally, I am grateful to my parents for their support and encouragement to complete my study.

# CONTENTS

Page

Page

Page

# LIST OF TABLES

Page

# LIST OF FIGURES

Page

Page

# CHAPTER I

# INTRODUCTION

## 1.1 Background and Significance of the Problem

Wireless sensor networks (WSNs) are a group of wireless nodes that are connected together to form a network. These nodes are equipped with sensors for sensing the surrounding environmental properties, such as temperature, pressure, humidity, light intensity, acceleration, etc. The main constrains of wireless sensor networks are limited computing capability, limited energy, and low communication bandwidth. To make measured data from those sensors applicable, we often need to know the location.

Localization is the task of identifying the location of an object. There are wide range of applications based on localization, such as automated wireless sensor network deployment, asset tracking, traffic management, surveillance system, location-aware advertising and geographical routing algorithms. An example of well-known localization system is Global Positioning System (GPS). Using GPS in WSNs suffers from high cost, power consumption, and incapability for indoor environment.

Let see some situations that benefit from the localization algorithm. One application for WSNs is environmental monitoring. Thousands of sensor nodes are randomly deployed from the plane in a large area, i.e. a forest. It would be a pleasure to know all these node locations automatically, since using the manpower to manually locate each sensor node location would be impractical in this case. To overcome the problem, one can equip a few sensor nodes with GPS and the rest of the sensor node locations can be found with the localization algorithm. Another application is indoor location tracking. Some nodes are deployed at fixed known locations all over the area of interest. Other nodes are carried by people or attached to the objects. These nodes are moving around and we can find their locations by running the localization algorithm iteratively.

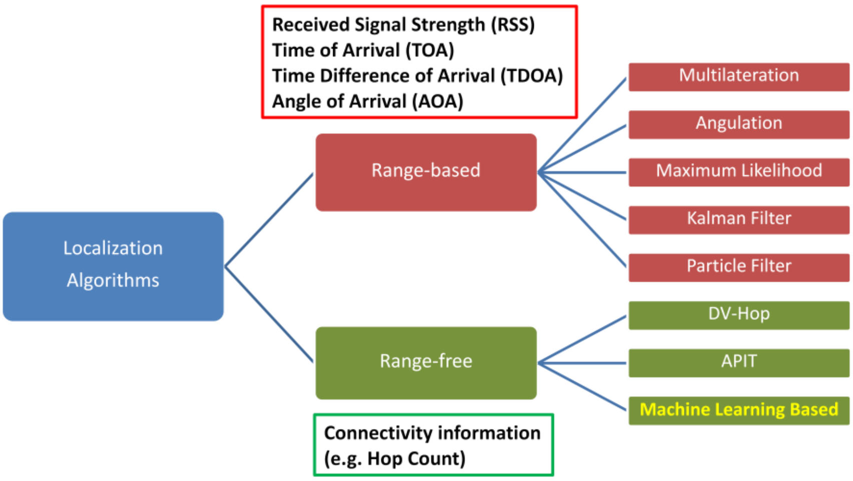


Fig. 1.1 Categorization of localization algorithms.

There are two main categories of localization algorithms; range-based localization and range-free localization. The categorization is illustrated in Fig. 1.1. In range-based localization, we need to measure the distance between two nodes. The internode distance can be estimated from Received Signal Strength Indicator (RSSI), Time of Arrival (ToA), Time Difference of Arrival (TDoA), or Angle of Arrival (AoA). These properties can be measured by several signal types, e.g. radio frequency signal, infrared signal, ultrasound signal, light signal, and ultra-wideband signal. RSSI provides a low cost implementation, but it often suffers from measurement noise. Thus it provides low accuracy. ToA and TDoA require precise time synchronization between wireless nodes. AoA requires additional hardware to perform signal's direction detection. The estimated distances can then be used to estimate location by localization algorithms, such as multilateration [1], particle filter [2], Kalman filter [3], and maximum likelihood [4]. In range-free localization, the distance information is not required. The location is directly approximated from the measured quantity. It is often based on the connectivity information, e.g. hop count. The examples of range-free localization are DV-Hop [5] and APIT [6]. For a review of different techniques for localization in WSNs, please see [7], [8].

Support Vector Machines (SVMs) [9] are among the best off-the-shelf machine learning algorithms and have been used in various field of applications, i.e. handwritten recognition, speaker verification, object detection, bioinformatics, image retrieval, webpage categorization, financial time series prediction and remote sensing data analysis. SVMs can also be applied to the localization problem and provide better accuracy when compared to other localization algorithm [10], [11]. The first work that makes use of SVMs for the localization problem in wireless network is perhaps [11]. By measuring RSSI in wireless LANs, SVMs can be used to classify node location at room level accuracy or performing regression for exact node location. SVMs are then applied to the localization problem in WSNs [12]. The work in [12] uses RSSI from reference nodes as a data input. The algorithm divides the localization area into multiple classification regions that cover the whole sensor network area. The classification regions take the shape of circle. To enable fine-grained localization, the center of multiple regions, in which the target node is predicted to be contained, are averaged together. Nodes in the border area can receive less RSSI information, causing these nodes predicted location to bias toward the center of localization area. Expanded region area coverage is proposed in [13] to avoid the problem of border effect. The SVMs can also be extended to three-dimensional localization problem [14]. The localization classification area can be divided in a hierarchical manner for efficiency [10]. LSVM is proposed and it considers the use of hop count information. Some works [15], [16], [17] also try to tackle the SVM localization problem of mobile node. Combine Sensor Scheduling with SVM localization to perform tracking is considered in [15]. The threshold of RSSI variation [16] is used to determine whether the node has moved or not. In this way, the unnecessary recomputation of static node location can be avoided. In [17], the dual layer particle filter is proposed. The first layer uses SVM to estimate the coarse location, then the second layer uses particle filter to estimate accurate location from coarse location in the first step. For robustness to the large data set, [18] use ensemble support vector regression by dividing the whole network into several subnetworks. Different formulation of regression for SVM localization are used in several works, e.g. LS-SVR for mapping RSSI of acoustic signal to location [19], $\varepsilon$-SVR for mapping RSSI of WLANs [11], and [20] compares $\varepsilon$-SVR, LS-SVR, and complex-valued

$\varepsilon$-SVR. ToA and TDoA information can also be used in SVM localization, as shown in [21], [22].

Advantages of SVMs-based localization are as follows: (1) It is range-free localization. We can easily get hop count or RSSI and let SVMs directly learn mapping function from these data sources to the location. There is no need to find distances between nodes. No extra hardware is required. (2) Comparing to other range-free localization method, SVMs-based method has good performance.

In this thesis, we try to improve the accuracy of SVMs-based localization system. We propose a framework for utilizing multiple data sources. We rely on only connectivity data, i.e. hop count. By varying transmission power of wireless node, we can generate multiple hop count data sources. The proposed method can be applied to both classification and regression formulation of SVMs. We additionally discuss on the topic of selecting optimal communication ranges to minimize the localization error. The simulation study is also performed extensively by varying some network parameters such as reference node density, communication range, coverage hole, reference node location error, and communication model noise. The simulation result demonstrates the advantage of using multiple data sources instead of using single data source.

## 1.2 Research Objectives

1.2.1 Propose a localization algorithm in WSNs based-on SVMs in both classification case and regression case.

1.2.2 Improve localization accuracy by utilizing hop count data from multiple transmission power.

1.2.3 Study the performance of the proposed localization algorithm by simulation.

## 1.3 Research Scope

1.3.1 Design the localization algorithm in WSNs.

1.3.1.1 WSNs contain reference nodes and target nodes.

1.3.1.2 All nodes can function as a router.

1.3.1.3 All node locations are assumed to be static.

1.3.1.4 The localization area is assumed to be a 2-dimensional square area.

1.3.1.5 The localization algorithm is based on SVMs, both classification case and regression case.

1.3.1.6 The SVMs training process is centralized.

1.3.1.7 Use hop count information as features to train SVMs.

1.3.1.8 The transmission power of nodes can be adjusted to vary the communication range.

1.3.1.9 Hop count information is based on shortest path routing.

1.3.2 Simulation study of the proposed localization algorithm.

1.3.2.1 The simulation study is performed in MATLAB.

1.3.2.2 Compare the localization error of classification case and regression case.

1.3.2.3 Study the choice of kernel functions; linear kernel and RBF kernel.

1.3.2.4 Study the effect of reference node density.

1.3.2.5 Study the effect of coverage hole problem.

1.3.2.6 Study the effect of transmission power variation.

1.3.2.7 Study the effect of noise on reference node location error.

1.3.2.8 Study the effect of noise on communication range.

1.3.2.9 Comparing different levels of data sources integration; early level and intermediate level.

1.3.2.10 Choosing the optimal communication ranges by evaluating mutual information criterion.

## 1.4 Expected Benefits of Research

1.4.1 Understand how to apply SVMs to the problem of localization in wireless sensor networks.

1.4.2 Improve the localization accuracy of SVMs-based localization algorithm.

1.4.3 Understand the performance of the proposed localization algorithm.

## 1.5 Thesis Organization

The remaining of this thesis is organized as follows: Chapter II presents a brief review of SVMs. We take a look at both classification and regression formulations of SVMs; SVC and SVR. Chapter III explains how to apply SVMs to the localization problem. Chapter IV shows how to utilize multiple communication ranges in localization. We also use mutual information as criteria to select optimal communication ranges. In chapter V, we analyze the performance of our proposed localization method from simulation study. Chapter VI is the conclusion and some suggestions for future work.

# CHAPTER II

# BACKGROUND KNOWLEDGE

## 2.1 Machine Learning

Machine learning is a process of discovering useful information or knowledge from the given data source. The machine learning process consists of several steps as illustrated in Fig. 2.1. The process begins by collecting all the data. This data can be divided into two data sets, the training data and the testing data. Both two data sets are passed through pre-processing, e.g. filtering noise, removing irrelevant data, normalizing, and scaling data. The training data is then applied to suitable choice of learning algorithms to construct the model that represents some knowledge on the data. The testing data can be applied to the model to get desired output. The post-processing may be used to filter out non-useful discovered knowledge. Depending on the particular learning task, not all of the steps above are required.

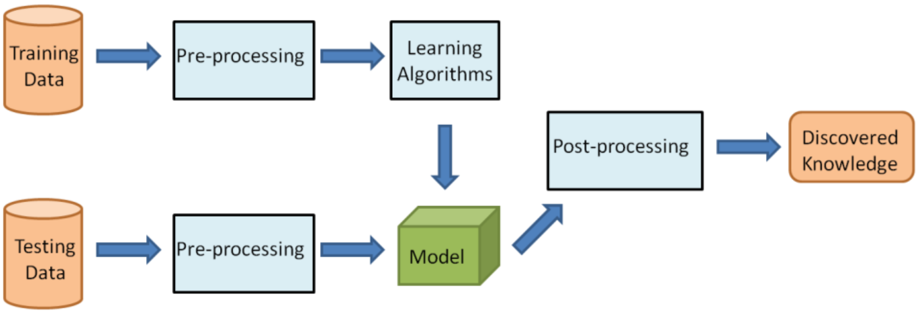


Fig. 2.1 Machine learning process data flow.

We can categorize the learning task into three categories; supervised learning, unsupervised learning, and semi-supervised learning. Supervised learning learns from labeled data, e.g. SVMs, k-nearest neighbor (k-NN), Neural Network, AdaBoost, and Decision Tree. Unsupervised learning learns from unlabeled data, e.g. k-means, k-medoids, mean shift, and Expectation-Maximization (EM) algorithm. Semi-supervised learning learns from both the labeled data and unlabeled data, e.g. TSVM, and one-class SVM.

## 2.2 Statistical Learning Theory

Suppose there are $N$ data pairs $(x_i, y_i)$, which are drawn independently from the identical unknown joint distribution function $P(x, y)$, where $x_i$ denotes the $i^{th}$ data point and $y_i$ denotes the corresponding label of $x_i$ . The first $\ell$ data pairs, $(x_i, y_i)$; $i \in \{1, 2, ..., \ell\}$, are assumed to be training data. The rest of data pairs, $(x_i, y_i)$; $i \in \{\ell + 1, \ell + 2, ..., N\}$, are assumed to be testing data. The training data pairs $(x_i, y_i)$ are given. The testing data points $x_i$ are given, but the testing data labels $y_i$ are unknown. We want to learn the function parameter $\theta$ of the decision function $f(x_i, \theta)$ that predicts the label $y_i$ from its data point $x_i$. In a classification problem, the predicted label is a discrete set of classes, e.g. $y_i \in \{-1, +1\}$. In a regression problem, the predicted label is a continuous real value, e.g. $y_i \in \mathbb{R}$.

The expected risk, which we ultimately want to minimize, is defined as

$$R(\theta) = \int |y - f(x, \theta)| dP(x, y) \tag{2.1}$$

The empirical risk is the mean error of the training data set

$$f(x) = sign\left(\sum_{i=1}^{\ell} \alpha_i y_i K(x, x_i) + b\right) \tag{2.2}$$

Classical learning methods use empirical risk minimization (ERM) principle. By minimizing $R_{emp}$, one hopes to minimize $R$ as well. However, the consistency may not hold [23]. The learned function may overfit the training data set, so that it may not generalize well to the whole data distribution.

If the function $f$ is too complex, the overfitting will occur. If the function $f$ is overly simplified, the underfittting will occur. We must somehow control the capacity of the function $f$. Since the distribution $P(x, y)$ is unknown, we cannot evaluate (2.1) directly. Instead, structural risk minimization (SRM) principle [23] minimizes both $R_{emp}$ and capacity term $\Omega$. The expected risk $R$ is bounded as shown in Fig. 2.2.

$$R(\theta) \leq R_{emp}(\theta) + \lambda\Omega(\theta) \tag{2.3}$$

The parameter $\lambda$ controls the tradeoff between minimizing $R_{emp}$ and minimizing $\Omega$. The capacity term $\Omega$ could be quite difficult to find, but can be estimated from growth function, VC dimension, or fat shattering dimension.

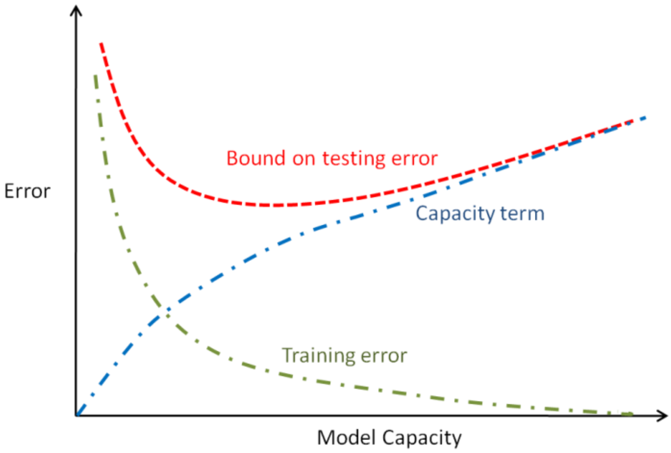


Fig. 2.2 The bound of testing error as the sum of training error and capacity term.

### 2.3 Maximum Margin Hyperplane

Let us begin with the simple separable binary classification problem in 2D space, $x_i \in \mathbb{R}^2$, shown in Fig. 2.3. The circle and rectangle represent different data classes. Let's assume that circle class and rectangle class have label $y_i$ equal to 1 and -1, respectively.

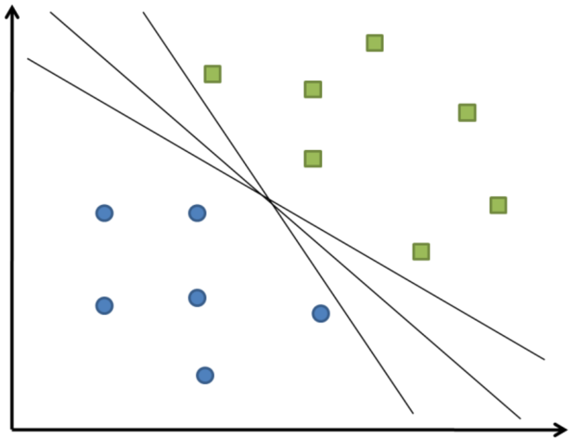


Fig. 2.3 Classification problem with different choices of separating hyperplane in 2D space.

We can draw a line to completely separate these two data classes. For a more general case when the dimension of $x_i$ is more than 2D, this line becomes a hyperplane. The decision function is defined as a hyperplane

$$f(x) = sign(\langle w, x\rangle + b) \tag{2.4}$$

Given that

$$sign(x) = \begin{cases} 1 & if\ x \geq 0 \\ -1 & if\ x < 0 \end{cases} \tag{2.5}$$

The loss function for this binary classification is defined as

$$\mathcal{L}\big(x, y, f(x)\big) = max\big(0, yf(x)\big) = \begin{cases} 0 & if\ y = f(x) \\ 1 & otherwise \end{cases} \tag{2.6}$$

There are many choices for the hyperplane that can perfectly separate the data points. Which hyperplane is the optimum?

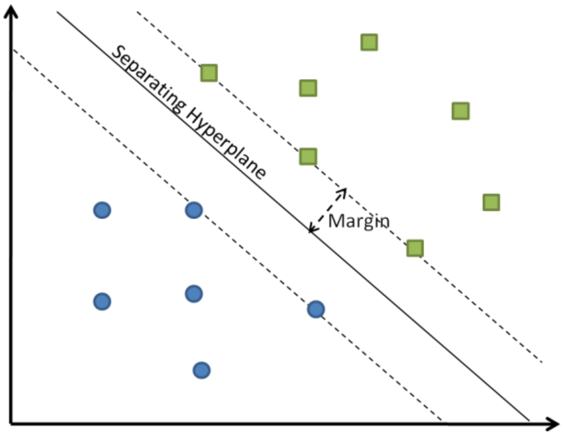


Fig. 2.4 Classification problem with maximum margin hyperplane in 2D space.

The optimum hyperplane must generalize well to be able to classify unseen testing data points correctly. We choose the maximum margin hyperplane. Fig. 2.4 illustrates the concept of maximum margin hyperplane. Assume that the data points can be completely separable by the separating hyperplane $\langle w, x_i\rangle + b = 0$. The data points closest to the separating hyperplane satisfy $|\langle w, x_i\rangle + b| = 1$. Let $\|w\|$ be the norm of $w$. The perpendicular distance from the origin to the hyperplane is $\frac{b}{\|w\|}$. The margin, the perpendicular distance of the closest points to the separating hyperplane, is defined as $\frac{1}{\|w\|}$.

From geometry interpretation, minimizing $\|w\|$ is equivalent to maximizing the margin. Without loss of generality, we consider minimizing $\frac{1}{2}\|w\|^2$ instead of $\|w\|$. We get the following constrained optimization problem.

Minimize $$\frac{1}{2}\|w\|^2, \tag{2.7}$$

subject to $$y_i(\langle w, x_i\rangle + b) \geq 1, \qquad i = 1, \dots, \ell \tag{2.8}$$

The constraint functions are derived from the loss function itself, i.e. no misclassification for all training data points. Refer to (2.3), this means that the empirical error term $R_{emp}(\theta)$ is zero, while minimizing the margin $\|w\|$ is analogous to minimizing capacity term $\Omega(\theta)$.

We can solve this constrained optimization problem by using Lagrange multipliers $\alpha_i \geq 0$ and Lagrangian

$$L_P = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{\ell} \alpha_i(y_i(\langle w, x_i\rangle + b) - 1) \tag{2.9}$$

This is the primal problem, but we can form a dual problem according to the following KKT conditions

$$\frac{\partial}{\partial w} L_P = 0, \tag{2.10}$$

$$\frac{\partial}{\partial b} L_P = 0 \tag{2.11}$$

The conditions above lead us to

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0, \tag{2.12}$$

$$w = \sum_{i=1}^{\ell} \alpha_i y_i x_i \tag{2.13}$$

By substituting (2.12) and (2.13) into (2.9), we get the dual form of the quadratic programming problem

Maximize $$L_D = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle, \tag{2.14}$$

subject to $$\alpha_i \geq 0, \qquad i = 1, \dots, \ell, \tag{2.15}$$

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0 \tag{2.16}$$

We can obtain the decision function that can be express in term of $\alpha_i$ and $\langle x_i, x_j \rangle$ instead of $\langle w, x_i \rangle$. By substituting (2.13) into the original decision function (2.4), the decision function becomes

$$f(x) = sign\left( \sum_{i=1}^{\ell} \alpha_i y_i \langle x, x_i \rangle + b \right) \tag{2.17}$$

For any $\alpha_i > 0$, the corresponding data input $x_i$ is called support vector. These support vectors are used in the calculation of the decision function, while those non-support vectors $(\alpha_i = 0)$ do not involve in the calculation. After the training phase, we only need to store these support vectors and their associated coefficients for the training phase. Also, removing non-support vectors from the training data does not affect the resultant optimal hyperplane. Please note that, for the dual form problem, all input data $x$ only appear in the terms of dot products $\langle x_i, x_j \rangle$. This observation will let us apply the kernel trick to the problem.

## 2.4 Kernel Function

An example of how mapping data to a higher dimensionality can help classification is shown in Fig. 2.5. In this example, the problem is not linearly separable in the original input data space. We map the space of $(X_1, X_2)$ into the polynomial space of degree 2, $\left((X_1)^2, (X_2)^2, \sqrt{2}X_1X_2\right)$, via the mapping function $\Phi()$. The figure on the right side shows that the problem becomes linearly separable in the space of $((X_1)^2, (X_2)^2)$.

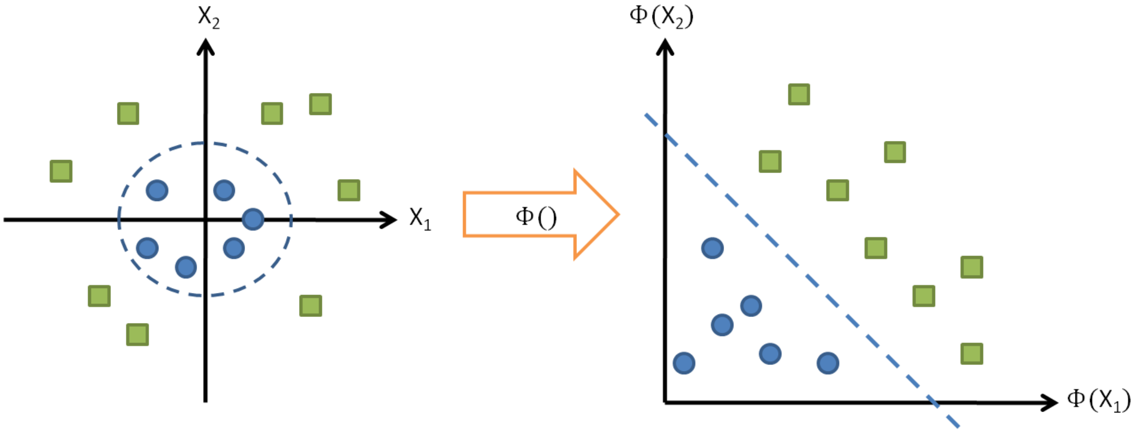


Fig. 2.5 Mapping from original data space into a new feature space that is linearly separable.

The kernel function is defined as

$$k(x,x') = \langle \Phi(x), \Phi(x') \rangle, \tag{2.18}$$

where $\langle\ \rangle$ is the inner product operator such that $\langle x, x' \rangle = x^T x'$.

The function $\Phi(x)$ maps data $x$ in input data space $\chi$ into some feature space $\mathcal{H}$. By using kernel, one doesn't need to compute $\Phi(x)$ explicitly. For the kernel function to be applicable, it must satisfy Mercer's Theorem. The kernel matrix has to be positive semi-definite.

Intuitively, kernel measures the similarity between a pair of data points, $x$ and $x'$. We can apply the kernel trick to maximum separating hyperplane problem by replacing $\langle x, x' \rangle$ with the kernel function $k(x,x')$. This kernel trick can be applied to other learning algorithms as well, such as Kernel Principle Component Analysis (KPCA), and Kernel Fisher Discriminant (KFD).

There are many choices of kernel function. Here are some of the basic kernel functions found in most literature.

- Linear kernel :

$$k(x,x') = \langle x, x' \rangle \tag{2.19}$$

This linear kernel is equivalent to learning in the original input data space $\chi$. No mapping is performed.

- $d^{th}$ degree polynomial kernel :

$$k(x,x') = (\gamma + \langle x, x' \rangle)^d \tag{2.20}$$

The parameter $d$ is the order of polynomials.

- Radial basis function (RBF) kernel :

$$k(x,x') = exp\left(-\frac{\|x - x'\|^2}{\sigma}\right) \tag{2.21}$$

The RFB kernel is sometimes referred to as Gaussian kernel. The parameter $\sigma$ controls the width of the kernel function.

- Sigmoid kernel :

$$k(x,x') = tanh(\gamma\langle x,x'\rangle + r) \tag{2.22}$$

Using the sigmoid kernel in SVMs is equivalent to two-layer neural network learning. Note that, this kernel doesn't satisfy the condition of Mercer's theorem.

There are also types of kernel for specific applications, such as string kernel, locality improvement kernel, and graph kernel.

The valid kernel function $k(x,x')$ must have some feature mapping space $\mathcal{H}$. According to Mercer's theorem, It is a necessary and sufficient condition that the kernel matrix $K$ is symmetry positive semi-definite (PSD) for the valid kernel function.

A kernel matrix $K$ on the set of $N$ input data $x_i$ is defined as

$$K = \begin{bmatrix} k(x_1,x_1) & \cdots & k(x_1,x_N) \\ \vdots & \ddots & \vdots \\ k(x_N,x_1) & \cdots & k(x_N,x_N) \end{bmatrix} \tag{2.23}$$

This kernel matrix $K$ is PSD if

$$z^T K z \geq 0, \qquad \forall z \in \mathbb{R}^N \tag{2.24}$$

Note that, weighted linear combination of PSD kernel matrices is also PSD.

## 2.5 Support Vector Machines

In this section, we explain different formulations of SVMs in both classification and regression problem. The derivation here is primary based on [9].

### 2.5.1 Support Vector Classification

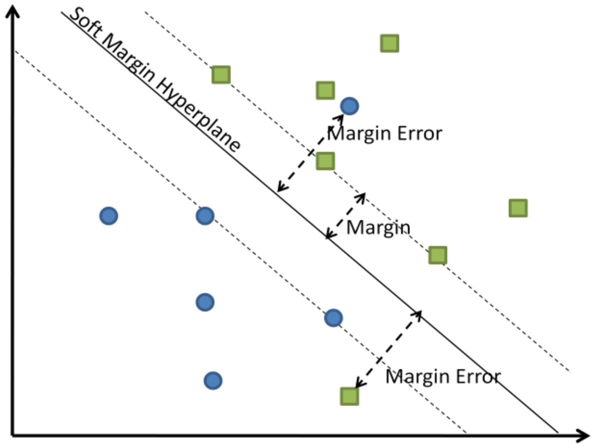


Fig. 2.6 Classification problem with soft margin hyperplane in 2D space.

In real problem, we cannot always find a hyperplane that can completely separate two data classes. From Fig. 2.6, those misclassified data points have some margin error. This means that constraint functions (2.8) cannot be satisfied at those data points. Thus, we need to introduce slack variable $\xi_i$ to relax the constraint functions.

The loss function for this soft margin classification is defined as

$$\begin{aligned}\mathcal{L}\big(x,y,f(x)\big) &= max\big(0,1-y(\langle w,x\rangle+b)\big)=\xi \\ &= \begin{cases} 0 & if\ y(\langle w,x\rangle+b)\geq 1 \\ 1-y(\langle w,x\rangle+b) & otherwise \end{cases}\end{aligned} \tag{2.25}$$

We can formulate constrained optimization for $C$-Support Vector Classification ($C$-SVC) as follows

Minimize
$$\frac{1}{2}\|w\|^2+C\sum_{i=1}^{\ell}\xi_i \tag{2.26}$$

subject to
$$y_i(\langle w,x_i\rangle+b)\geq 1-\xi_i, \qquad i=1,\dots,\ell, \tag{2.27}$$

$$\xi_i\geq 0, \qquad i=1,\dots,\ell \tag{2.28}$$

Refer to (2.3), minimizing the slack variable $\xi_i$ is analogous to minimize empirical error term $R_{emp}(\theta)$, while minimizing the margin $\|w\|$ is analogous to minimizing capacity term $\Omega(\theta)$. The parameter $C$ is equivalent to the term $\lambda$ which controls tradeoff between minimizing capacity and empirical error.

We arrive at the dual form of optimization problem

maximize
$$L_D=\sum_{i=1}^{\ell}\alpha_i-\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\alpha_i\alpha_j y_i y_j k(x_i,x_j) \tag{2.29}$$

subject to
$$0\leq\alpha_i\leq C, \tag{2.30}$$

$$\sum_{i=1}^{\ell}\alpha_i y_i=0 \tag{2.31}$$

Decision Function of SVC is

$$f(x)=sign\left(\sum_{i=1}^{\ell}\alpha_i y_i k(x,x_i)+b\right) \tag{2.32}$$

### 2.5.2 Support Vector Regression

SVMs are not only limited to the classification problem, they can also be used for regression problem. The regression function is defined as

$$f(x) = \langle w, x \rangle + b \tag{2.33}$$

Here we introduce the $\varepsilon$-insensitive loss function defined as

$$\mathcal{L}\big(x, y, f(x)\big) = |y - f(x)|_{\varepsilon} = max\{0, |y - f(x)| - \varepsilon\} = \xi \tag{2.34}$$

As shown in Fig. 2.7, this loss function ignores any errors of size less than $\varepsilon$.

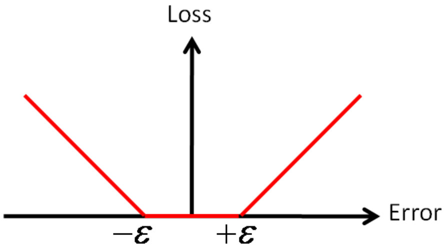


Fig. 2.7 $\boldsymbol{\varepsilon}$-insensitive loss function.

Those data points, which have more error than $\varepsilon$, will introduce slack error. Now, we need two slack variables, $\xi_i$ and $\xi_i^*$, for two cases of $f(x_i) - y_i > \varepsilon$ and $y_i - f(x_i) > \varepsilon$, respectively.

We can formulate constrained optimization for $\varepsilon$-Support Vector Regression ($\varepsilon$-SVR) as follows

minimize
$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{\ell}(\xi_i + \xi_i^*) \tag{2.35}$$

subject to
$$(\langle w, x_i \rangle + b) - y_i \leq \varepsilon + \xi_i, \qquad i = 1, \ldots, \ell, \tag{2.36}$$

$$y_i - (\langle w, x_i \rangle + b) \leq \varepsilon + \xi_i^*, \qquad i = 1, \ldots, \ell, \tag{2.37}$$

$$\xi_i, \xi_i^* \geq 0, \qquad i = 1, \ldots, \ell \tag{2.38}$$

We arrive at the dual form of optimization problem

Maximize
$$L_D = -\frac{1}{2}\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}(\alpha_i^* - \alpha_i)\big(\alpha_j^* - \alpha_j\big)k\big(x_i, x_j\big) \tag{2.39}$$

$$-\varepsilon\sum_{i=1}^{\ell}(\alpha_i^*-\alpha_i)+\sum_{i=1}^{\ell}y_i(\alpha_i^*-\alpha_i)$$

subject to $$0\le\alpha_i,\alpha_i^*\le C, \tag{2.40}$$

$$\sum_{i=1}^{\ell}(\alpha_i^*-\alpha_i)=0 \tag{2.41}$$

The decision function of SVR is

$$f(x)=\sum_{i=1}^{\ell}(\alpha_i^*-\alpha_i)k(x_i,x_j)+b \tag{2.42}$$

## 2.6 Error Bound

### 2.6.1 Leave-one-out Error

The leave-one-out error is an almost unbiased estimate of the expected test error. We define leave-one-out error as

$$Ep_{error}^{\ell-1}=\frac{1}{\ell}E\big(\mathcal{L}(x_1,y_1,\dots,x_\ell,y_\ell,f(x))\big) \tag{2.43}$$

The term "almost" is from the fact that leave-one-error is the expected test error for the sample size of $\ell-1$ instead of $\ell$.

While leave-one-out error is a good estimator of the probability of test error, the direct computation of leave-one-out error is very computationally intensive. We need to select one sample data as testing data and the rest of sample data as training data. We repeated this process until all sample data get used as test data. It requires $\ell$ times of training. In case of SVMs, it is possible to speed up the computation of leave-one-out error. For example, we only need to test leave-one-out error for support vectors, since removing non support vector does not change the decision function. Alternatively, we can find the bound for leave-one-out error effectively. We can derive this bound from span.

### 2.6.2 Span Bound for Classification

For any support vector $x_p$, we define the set $\Lambda_p$ as a constrained linear combinations of the other support vectors $\{x_i: \forall i\neq p\}$.

$$\mathit{\Lambda}_p = \left\{ \sum_{i=1, i\neq p}^{\ell} \lambda_i x_i : \sum_{i=1, i\neq p}^{\ell} \lambda_i = 1, and\ \forall i \neq p, \alpha_i^0 + y_i y_p \alpha_p^0 \lambda_i \geq 0 \right\} \tag{2.44}$$

We then define span $S_p$ of the support vector $x_p$ as the minimum distance between $x_p$ and the set $\mathit{\Lambda}_p$

$$S_p^2 = \min_{x\in\mathit{\Lambda}_p} \left(x_p - x\right)^2 \tag{2.45}$$

The leave-one-out error is bounded as

$$Ep_{error}^{\ell-1} \leq \frac{1}{\ell} max\ (\frac{S_p D}{\rho^2}) \tag{2.46}$$

where $D$ is a smallest diameter of sphere containing all training data points and $\rho$ is the margin.

Under assumption that the set of support vectors remains the same during leave-one-out error procedure, the following equality is true for any support vector $x_p$

$$y_p(f^0\left(x_p\right) - f^p(x_p)) = \alpha_p^0 S_p^2 \tag{2.47}$$

and the leave-one-out error can be computed as

$$Ep_{error}^{\ell-1} = \frac{1}{\ell} Card\left\{p \colon \alpha_p^0 S_p^2 \geq y_p f^0(x_p)\right\} \tag{2.48}$$

where $f^0$is a decision function trained on a whole training set and $f^p$ is a decision function trained without $x_p$.

### 2.6.3 Span Bound for Regression

For any support vector $x_p$ , we define the set $\mathit{\Lambda}_p$ as a constrained linear combinations of the other support vectors $\{x_i \colon \forall i \neq p\}$.

$$\mathit{\Lambda}_p = \left\{ \sum_{i=1, i\neq p}^{\ell} \lambda_i x_i : \sum_{i=1, i\neq p}^{\ell} \lambda_i = 1, and\ \forall i \neq p, \alpha_i^0 + \alpha_i^{*0} \geq 0 \right\} \tag{2.49}$$

We then define span $S_p$ of the support vector $x_p$ as the minimum distance between $x_p$ and the set $\mathit{\Lambda}_p$

$$S_p^2 = \min_{x\in\mathit{\Lambda}_p} \left(x_p - x\right)^2 \tag{2.50}$$

Under assumption that the set of support vectors remains the same during leave-one-out error procedure, the leave-one-out error is bounded as

$$Ep_{error}^{\ell-1} \leq \sum_{p=1}^{\ell} \left(\alpha_p + \alpha_p^*\right) S_p^2 + \ell\varepsilon \tag{2.51}$$

# CHAPTER III

# LOCALIZATION USING SUPPORT VECTOR MACHINES

## 3.1 Problem definitions for Localization using SVMs

In order to apply SVMs to any particular problem, we need to identify what is training data set and testing data set. We need to define the format of the data point $x_i$and the label $y_i$. We also need a procedure for acquiring them. For our proposed localization method, we learn a mapping function from hop count data to location of the sensor node. In this chapter, we describe how to apply SVMs to the problem of localization in wireless sensor network.

Suppose there is a wireless sensor network that consists of $N$ wireless sensor nodes $S_1, S_2, \ldots, S_N$. These nodes are deployed in 2D space along the x-axis and y-axis. Let $\left(S_{x,i}, S_{y,i}\right)$ be the location of $S_i$ node. Also assume that the first $\ell$ node locations are known. These first $\ell$ nodes are called reference nodes. The rest of the nodes are target nodes, whose locations are unknown. The perfect disk communication model is adopted. This means a particular node can directly communicate with any nodes that are in the communication range $r$. All of the nodes can act as a router, forwarding the message in multi-hop fashion. We try to estimate the locations of target nodes.

## 3.2 Algorithms Overview

The proposed localization approach consists of three main stages; data collecting stage, training stage, and localization stage.

(1) Data collecting stage – We gather the training data set and testing data set within WSNs. The data set in this case is the hop count information. The hop count data is generated from shortest-path routing. First, all reference nodes broadcast their own ID to all nodes in the network. The hop count information are collected and sent to the concentrator node of the network. This concentrator node is assumed to have high processing power. This stage can be repeated with varied transmission power to obtain additional hop count information.

(2) Training stage – Concentrator node builds the training data set and testing data set from the hop count information collected in the previous stage. The training data set is used to train the SVMs. Both classification and regression case are considered in this research. In classification case, we use binary SVMs with the multiclass strategy that adopts the idea of hierarchical classification in LSVM [10]. The two coordinates $(S_{x,i}, S_{y,i})$ are trained independently with different SVMs models.

(3) Localization stage – The testing data set is applied to the trained SVMs in order to predict the locations of target nodes. This stage can be done in centralized manner (concentrator node compute all the locations of target nodes) or in distributed manner (the trained SVM models are sent to the target nodes and let them compute the location for themselves). The output becomes predicted node area in case of classification and predicted node location in case of regression.

### 3.3 Regression Case

Because of the simplicity in applying the regression formulation of SVMs to the problem of localization, we discuss this case first. Although we stated before that the location in x-axis and y-axis have to be applied to the SVMs separately, the process for any axis is similar. Without any loss of generality, we discuss for the location along the x-axis.

Let us define $H_{[i,j]}$ is the hop count between node $S_i$ and node $S_j$ . By broadcasting of all reference nodes, we can generate hop count between any particular node $S_i$ to all reference nodes, i.e. $H_{[i,j]} : i \in \{1, \dots, N\}, j \in \{1, \dots, \ell\}$.

For the training data, we know hop count matrix between all reference nodes

$$\begin{bmatrix} H_{[1,1]} & \cdots & H_{[1,\ell]} \\ \vdots & \ddots & \vdots \\ H_{[\ell,1]} & \cdots & H_{[\ell,\ell]} \end{bmatrix} \tag{3.1}$$

where each row $i$ of this matrix is the data point $x_i$ of reference node $S_i$.

We also have location matrix of all reference nodes

$$\begin{bmatrix} S_{x,1} \\ \vdots \\ S_{x,\ell} \end{bmatrix} \tag{3.2}$$

where each row $i$ of this matrix is the label $y_i$ of reference node $S_i$.

Because we know completely the data point and label of all reference nodes, we can use this training data set to train SVMs, in this case SVR. As a result, we get a trained SVMs model that mapping the hop count data to the location.

For the testing data set, we have hop count matrix between target nodes and reference nodes

$$\begin{bmatrix} H_{[\ell+1,1]} & \cdots & H_{[\ell+1,\ell]} \\ \vdots & \ddots & \vdots \\ H_{[N,1]} & \cdots & H_{[N,\ell]} \end{bmatrix} \tag{3.3}$$

where each row $i$ of this matrix is the data point $x_i$ of target node $S_i$.

We need to find the unknown location matrix of all target nodes

$$\begin{bmatrix} S_{x,\ell+1} \\ \vdots \\ S_{x,N} \end{bmatrix} \tag{3.4}$$

where each row $i$ of this matrix is the label $y_i$ of target node $S_i$.

By feeding hop count data of each target node to the trained SVMs model, we can predict its location along the x-axis.

There are two obvious ways to improve the performance of machine learning algorithms, in this case SVMs. One way is increasing number of training samples or another way is increasing number of features in each data point. For this localization procedure, as we increase the number of reference nodes, we simultaneously increase number of both training samples and data point features. But these reference nodes are costly to deploy, we want to minimize their amount as much as possible. Alternatively, we can vary radio transmission power in these wireless nodes to generate multiple hop count matrices, hence we can get more data point features. The topic of how to utilize multiple transmission power is discussed in chapter IV.

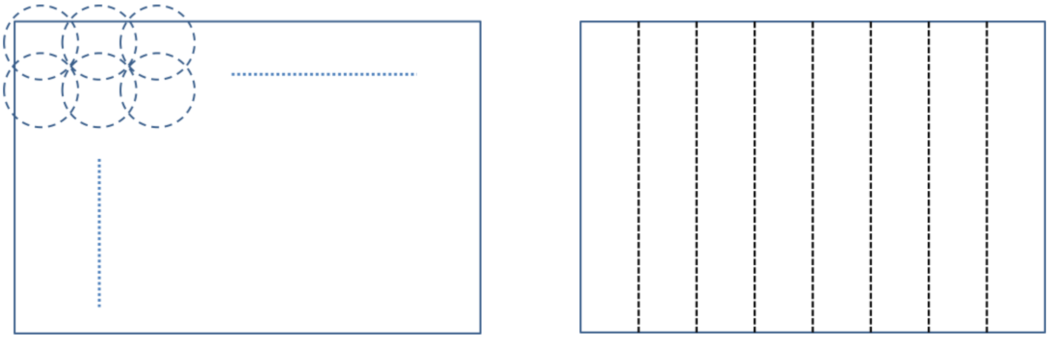
For the error bound in regression case, we can simply use span bound described in previous chapter II.

The error in localization is defined as norm of error in x-axis and y axis

$$e = \sqrt{e_x^2 + e_y^2} \tag{3.5}$$

## 3.4 Classification Case

Apply classification formulation of SVMs to localization is similar to regression case except the format of label $y_i$. The label is defined to be an area that particular node reside. Fig. 3.1 show some examples of how can we define the classification area; (a) circular shape and (b) rectangle shape. We can simultaneously classify along both x-axis and y-axis or classify along each axis separately.



(a) Area classification along 2-axis

(b) Area classification along 1-axis

Fig. 3.1 Shape of classification area.

In this thesis, we adopt LSVM method, which divides the classification area in decision tree manner illustrated in Fig. 3.2 for three level depths.

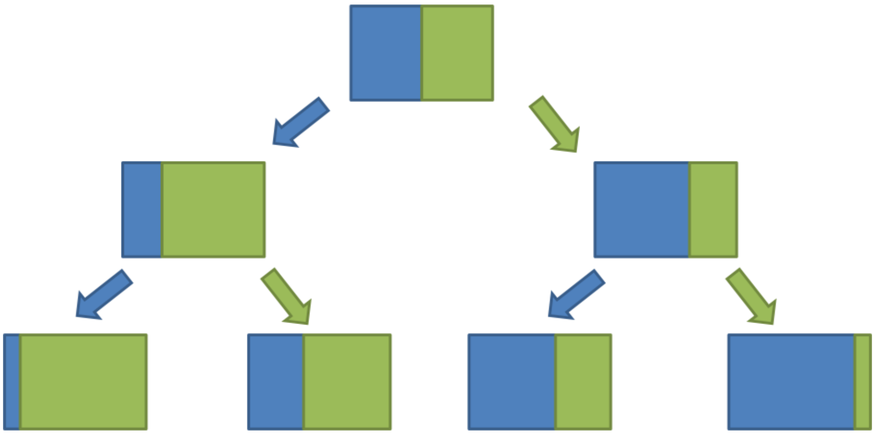
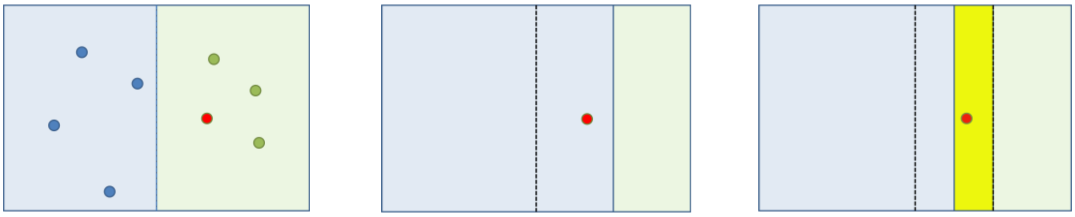


Fig. 3.2 Decision tree classification.

Let us consider classification along the x-axis. An example of classification process is shown in Fig. 3.3. We want to classify the red node. At first level depth in decision tree, the area is divided in half and the node is on the right side. At second level depth, the area on the right is further divided in half and the node is on the left side. At third level depth, the area on the left is further divided in half and the node is on the right side, so this red node is classified to be in the yellow area. For 2D classification, we can also do the same process along the y-axis and find the interception of the area from both axes. We can simply use the center of the classified area to be the location of the node. The depth of decision tree can be increased to get finer resolution of localization.



(a) First level depth.

(b )Second level depth.

(c) Third level depth.

Fig. 3.3 Example of classification in LSVM.

### 3.4.1 Error Bound for LSVM

In this section, we derive the error bound for LSVM. First, we consider only along x-dimension without any loss for generality. The x-dimension length $D$ is divided into equally $2^m$ sections, where $m$ is the depth of tree in LSVM. Let $X_r$ be the real class index and $X_p$ be the predicted class index for LSVM. $e(X_r, X_p)$ is defined as average distance between class $X_r$ and $X_p$. In the case of uniform node distribution, $e(X_r, X_p)$ becomes

$$e(X_r, X_p) = |X_r - X_p|\frac{D}{2^m} \tag{3.6}$$

The expected error of LSVM becomes

$$Error = \sum_{X_p=1}^{2^m}\sum_{X_r=1}^{2^m} e(X_r, X_p)p(X_r, X_p, \boldsymbol{\epsilon}) \tag{3.7}$$

Definition of $\boldsymbol{\epsilon}$ is expected classification error at each level of decision tree. We can use span bound to estimate its value.

By using Bayes' rule, we can rewrite (3.7) as

$$Error \ = \sum_{X_p=1}^{2^m} \sum_{X_r=1}^{2^m} e(X_r, X_p) p(X_p | X_r, \boldsymbol{\epsilon}) p(X_r) \tag{3.8}$$

The probability density function $p(X_r)$ is according to the pattern of node deployment. If the node is uniformly deployed across the field, we get

$$p(X_r) = \frac{1}{2^m} \tag{3.9}$$

Now we need to find the conditional probability density function $p(X_p | X_r, \boldsymbol{\epsilon})$. Assume that $\boldsymbol{\epsilon}$ is equal at all level of decision tree, we get

$$p(X_p | X_r, \boldsymbol{\epsilon}) = \ \epsilon^n (1 - \epsilon)^{m-n} \tag{3.10}$$

where $n$ is the number of misclassification between class $X_p$ and $X_r$.

# CHAPTER IV

# UTILIZING MULTIPLE TRANSMISSION POWER FOR LOCALIZATION

In this chapter, we discuss how to generate multiple additional hop count data sources and integrate them together to increase localization accuracy. By adjusting transmission power, we can inherently change the hop count data. We also discuss on the topic of using mutual information to select optimal choice of communication ranges to minimize the localization error.

## 4.1 Learning with Multiple Data Representation

When we consider learning from multiple data sources for kernel learning method, there are three choices regarding level of integration [24]; early integration, intermediate integration, and late integration. The levels of integration are shown in Fig. 4.1. For the data source $i$, let $X_i$, $Y_i$, and $K_i$ be the data matrix, label matrix, and kernel matrix, respectively. Let $K_S$ be the kernel matrix that is combined from multiple kernels matrices. Let $Y_S$ be the label matrix that is combined from multiple label matrices.

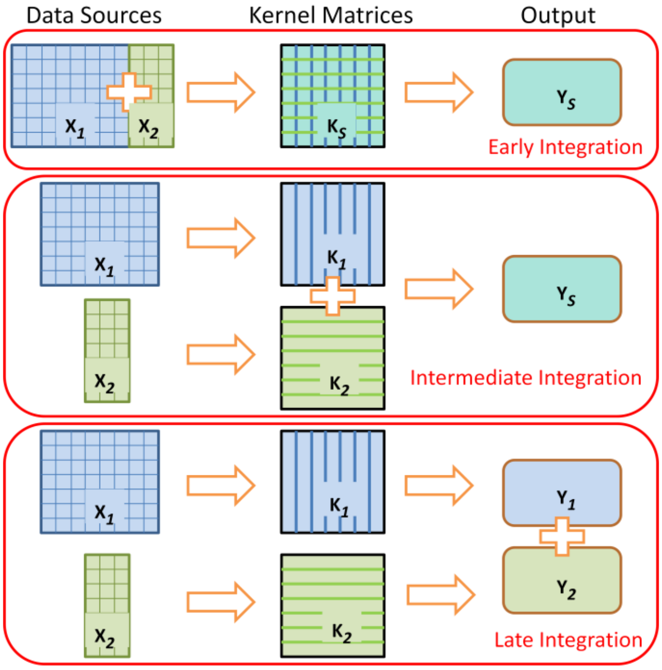


Fig. 4.1 Different level of integration for learning with multiple data sources.

Early integration simply concatenates the data matrices from different data sources together. The early integration is probably the easiest to implement, but it is not suitable if data sources are heterogeneous, e.g. combining the string data with the image data.

Intermediate integration uses different kernel functions for each data source and the resulting kernel matrices are linearly combined into a single kernel matrix. Then, the single kernel matrix gets passed to a single classifier. The intermediate integration gives the flexibility to select appropriate kernel for each data source, so we can combine heterogeneous data sources.

Late integration uses different classifiers for each data source and then applies ensemble learning method to combine the results. Late integration has the same flexibility at kernel level and it also gives additional flexibility to select appropriate classifier for each kernel matrix. However, it could be quite difficult to find the optimal combination choice of different kernels and classifiers. Otherwise, the final result may not be good.

In this research, we consider two levels of integration, i.e. early and intermediate level. For the early integration case, we simply concatenate hop count matrices from different communication ranges together and use it as a training data matrix for SVMs. We will refer to this as “unif method”. For the intermediate integration case, we use independent alignment-based method [25]. We will refer to this as “align method”. The kernel matrices are linearly combined as

$$K_S = \sum_{i=1}^{N} w_i K_i \tag{4.1}$$

with weight $w_i$ proportional to the centered alignment of each kernel matrix with the training data label.

$$w_i \propto \hat{\rho}(K_i, yy^T) \tag{4.2}$$

Alignment is a quantity to measure how much two kernels correlate to each other. Good kernel matrix $K$ should correlate well to ideal kernel $yy^T$. The centered alignment of two kernel functions is defined as follows

$$\rho(K,K') = \frac{E[K_cK_c']}{\sqrt{E[K_c^2]E[K_c'^2]}} \tag{4.3}$$

where $K_c$ is the centered kernel matrix. Essentially, this centered kernel matrix means centering in feature space. The centered kernel function becomes

$$k_c(x,x') = (\Phi(x) - E[\Phi(x)])^T(\Phi(x') - E[\Phi(x)]) \tag{4.4}$$

However, we do not need to know the mapping function $\Phi(x)$ to find centered kernel matrix. For kernel matrix $K$, we can directly compute the center kernel $K_c$ for all element $[i,j]$ in the matrix by

$$K_{c[i,j]} = K_{[i,j]} - \frac{1}{m}\sum_{i=1}^{m} K_{[i,j]} - \frac{1}{m}\sum_{j=1}^{m} K_{[i,j]} + \frac{1}{m^2}\sum_{i,j=1}^{m} K_{[i,j]} \tag{4.5}$$

The empirical alignment of two kernel functions on the sample data is defined as

$$\hat{\rho}(K,K') = \frac{\langle K_c, K_c'\rangle_F}{\sqrt{\langle K_c, K_c\rangle_F \langle K_c', K_c'\rangle_F}} \tag{4.6}$$

where $\langle .,.\rangle_F$ is the Frobenius product defined as

$$\langle A,B\rangle_F = Tr[A^TB] \tag{4.7}$$

In fact, unif method is a special case of intermediate level integration, as it has all weight $w_i$ set to 1 and the same kernel function is used for all data set.

## 4.2 Relationship between Transmission Power and Communication Range

We can generate multiple hop count data sources by adjusting transmission power of sensor nodes. This adjustment causes the communication range to be changed, so routing path is also changed and we get a new hop count data source. The probability density function of each hop count data source is different. Thus, we get new information to learn from.

In wireless radio communication, the received signal power falls off as the distance between transmitter node and receiver node increases. We can use a simple path-loss propagation model to describe the relationship between the received signal power $P_i$ and the distance $d_i$ as

$$P_i = P_0 - 10\eta log\left(\frac{d_i}{d_0}\right) \tag{4.8}$$

where $P_0$ is the reference power (dBm) at the reference distance $d_0$ and $\eta$ is a path-loss exponent.

Path-loss for indoor environment can be affected by obstruction from walls, floors, and objects. We can modify (4.8) to account for the attenuation factor $L$ and measurement noise $\mathcal{N}(\mu, \sigma^2)$, which is assumed to be Gaussian noise with mean $\mu$ and variance $\sigma^2$.

$$P_i = P_0 - 10\eta log\left(\frac{d_i}{d_0}\right) + L + \mathcal{N}(\mu, \sigma^2) \tag{4.9}$$

The value of $\eta$ and $L$ can be varied by surrounding environment [26]. Typical obstruction losses $L$, depending on the material, can vary from 1.4 dB for cloth to 26 dB for metal. Typical path-loss exponent $\eta$ can vary from 1.6 to 6.5. In real environment, the measured RSSI might not follow the path-loss propagation model due to multipath fading and shadowing effect, as illustrated in Fig. 4.2. The blue circles represent measured RSSI at the given distances. The red line shows fitted curve for path-loss model. The mean of RSSI tends to decrease with distance and the variance of RSSI tends to increase with distance. At farther distance, the measured RSSI becomes less reliable.

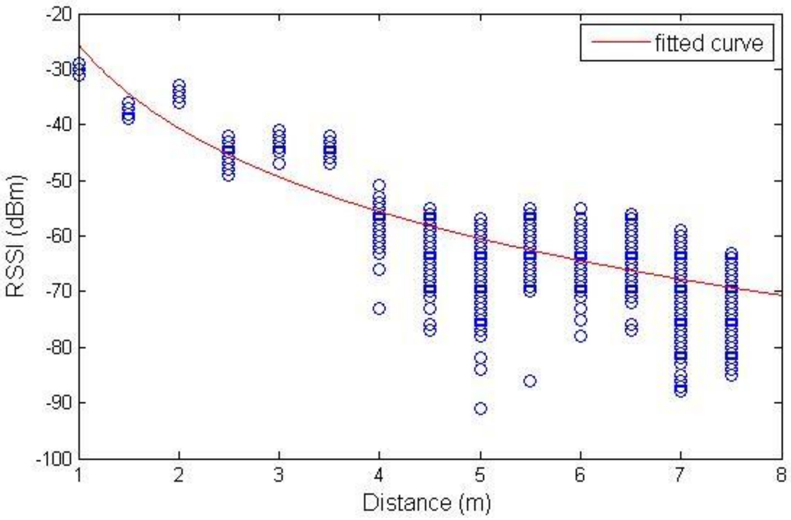


Fig. 4.2 Measured RSSI of wireless signal in an indoor environment.

Typically, the transmission power of wireless node can be adjusted by changing the according register value in transceiver IC. For example, we consider TI CC2530 SOC [27]. From its datasheet, the typical receiver sensitivity is –97 dBm and the recommended transmission power can be adjusted from 4 dBm down to -28 dBm. At a particular indoor environment, we make a measurement with transmission power set to 0 dBm at several distances. We do a curve fit for the measured RSSI data to the path-loss model in (4.8). With reference distance $d_0$ at 1 m, we get the value of $P_0$ and $\eta$, -30 dBm and 3.5, respectively. By rearranging (4.8), we get

$$r = d_0 exp\left(\frac{P_0 + P_i + Tx}{10\eta}\right) \tag{4.10}$$

We can now predict the communication range $r$ of the corresponding transmission power $Tx$ by using (4.10). The prediction of this example is shown in Fig. 4.3. In this case, the communication range can be varied around 12 m to 110 m.

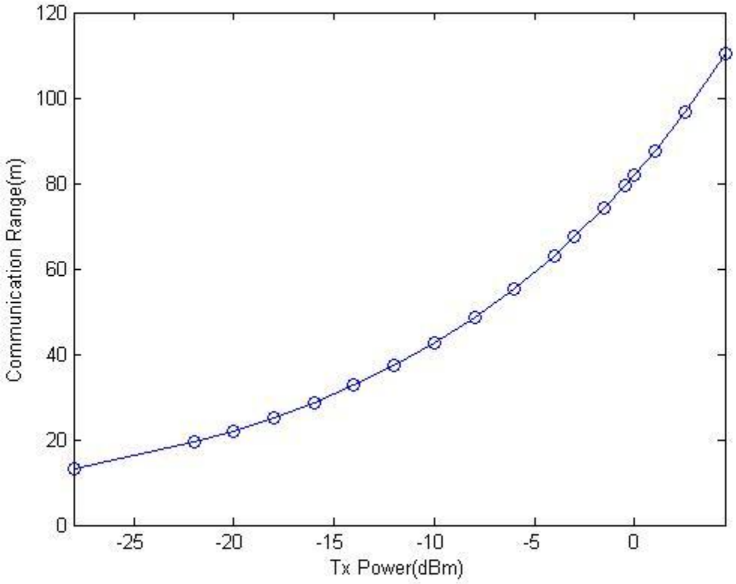


Fig. 4.3 Relationship between communication range and transmission power.

Note that, apart from the estimated lower and upper bound limit of communication range adjustment, the resolution of communication range adjustment is limited. Since we can only vary transmission power with a discrete step size, this implies that the communication range can be adjusted only in a discrete step size too.

## 4.3 Mutual Information

From information theory, we can measure the amount of information of a random variable by its entropy. The entropy of the discrete random variable $X$ is defined as

$$H(X) = -\sum_{x} p(x)\, log\, p(x) \tag{4.11}$$

The conditional entropy of the random variable $Y$ given the random variable $X$ is defined as

$$H(Y|X) = -\sum_{x,y} p(x,y)\, log\, p(y|x) \tag{4.12}$$

We can measure the amount of information that is shared by the random variable $X$ and $Y$. We call it the mutual information. The mutual information is defined as

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{4.13}$$

Intuitively, we can interpret the mutual information as information gain, as it measures how much knowing one of the random variables can reduce the uncertainty of another random variable. This mutual information can be applied to the problem of feature selection [28] in machine learning field by choosing the feature $X$ that give the highest mutual information on the output $Y$.

Note that the variables in the above discussion, $X$ and $Y$, are not restricted to be single random variable. They can be jointly distributed random variables, e.g. $I(X_1, X_2; Y_1, Y_2, Y_3)$ . The comma operator denotes the jointly distributed random variables.

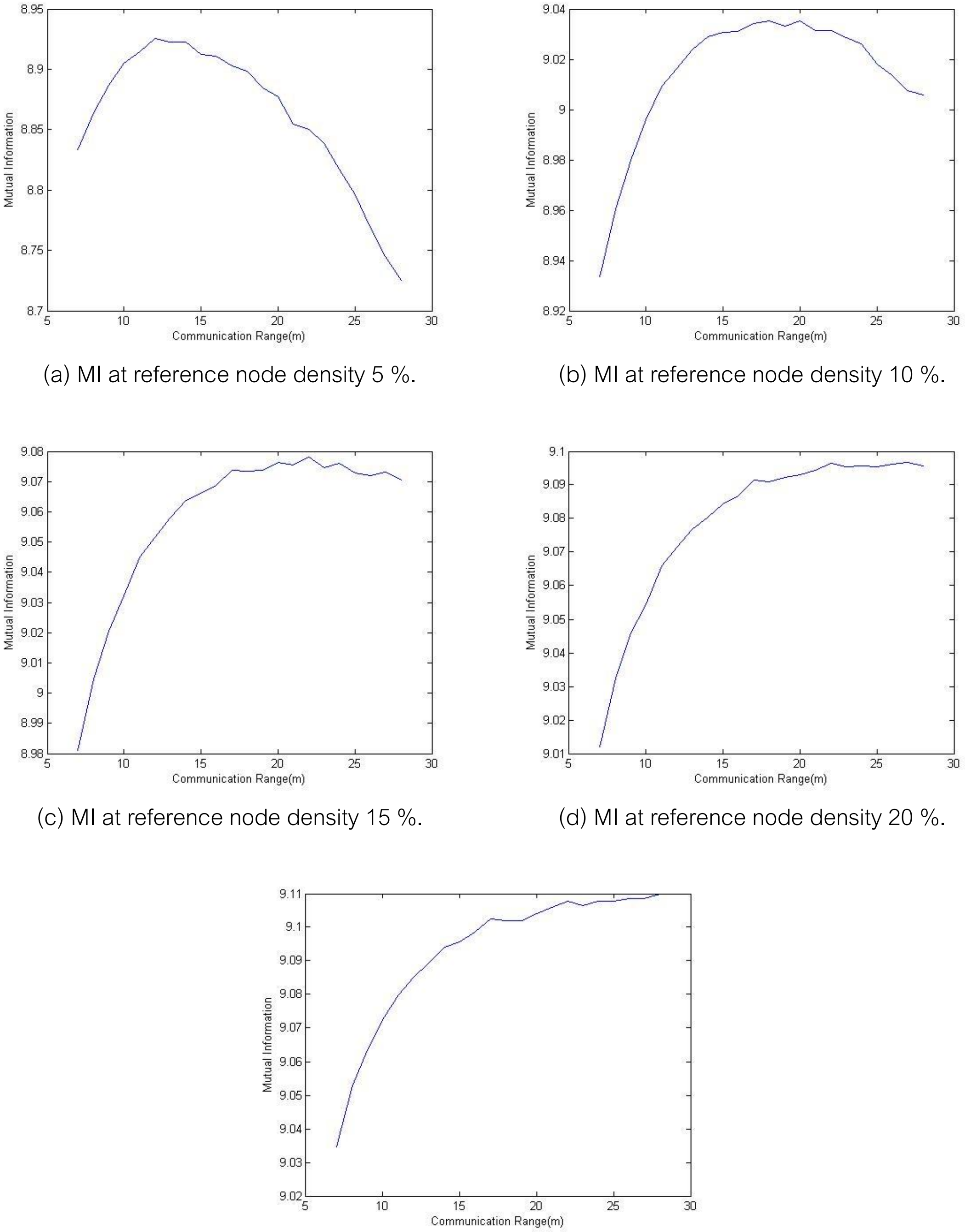
## 4.4 Selection of Optimal Communication Range

Communication range is one of the important network parameters. Selecting an optimal communication range can get involved with many criteria, such as power saving optimization and reliability of the connection. For the case of too short communication range, we have the problem of not enough coverage, unreliable connectivity, or some nodes become unreachable. For the case of too far communication range, the transmission power is wasted and the resolution of hop count is lost, i.e. farther or nearer nodes have the same hop count. In this section we choose the optimal combination of communication ranges that minimize localization error.

We adopt mutual information between node locations and hop count data sources as a criterion to select the optimal communication ranges. As we use SVM to learn node location from hop count information. The simulation setup is similar to chapter V. The network setup is 1,000 nodes deployed uniformly across the area of 100 m x 100 m. At various reference node density (5%, 10%, 15%, 20%, 25%), the communication range is varied incrementally from 7 m to 28 m in a step size of 1 m. Mutual information for the case of single communication range and double communication ranges are shown in Fig. 4.4 and Fig. 4.5, respectively.

Let's discuss for the case of single communication range first. As reference node density is increased, the optimal communication range becomes larger. Also, the value of mutual information is increased with reference node density. The estimated optimal communication ranges for reference node density of 5%, 10%, 15%, and 20% are 12 m, 18 m, 22 m, and 27 m, respectively. For the case of reference node density 25%, the optimal communication range is actually greater than 28 m.

For the case of double communication ranges, the mutual information in this case is greater than the single case. Similar to the case of single communication range, the mutual information and optimal communication ranges become greater at high reference node density. The best combination of double communication range is the area around the single optimal communication range. For example, at reference node density 5%, the optimal combination is 12 m and 13 m. We also test with the case of three communication ranges. The best combination is around 12m, 13m, and 14m.



(a) MI at reference node density 5 %.

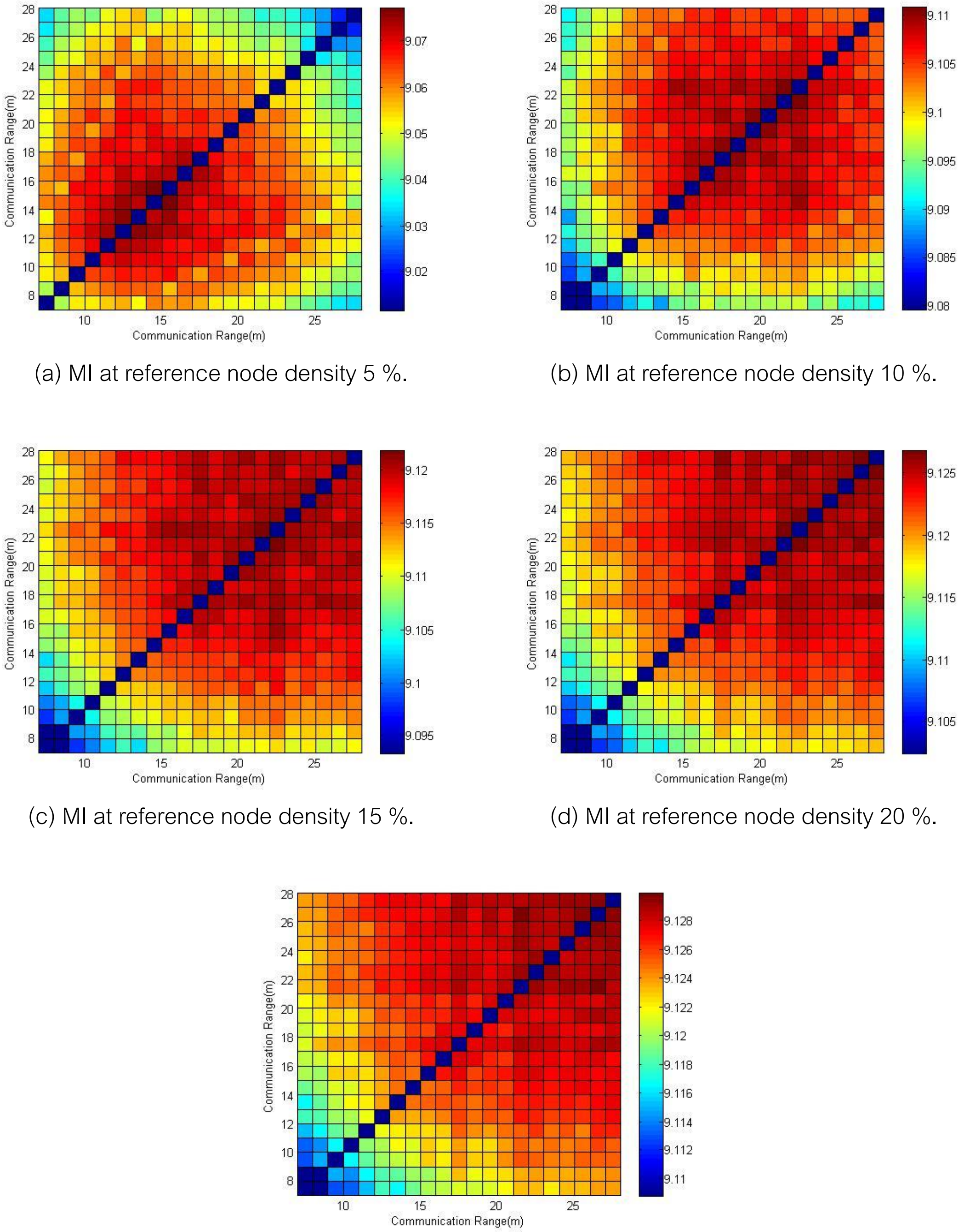
(b) MI at reference node density 10 %.

(c) MI at reference node density 15 %.

(d) MI at reference node density 20 %.

(e) MI at reference node density 25 %.

Fig. 4.4 Mutual information between hop count information and location in x-axis at different reference node density for the case of single communication range.



(a) MI at reference node density 5 %.

(b) MI at reference node density 10 %.

(c) MI at reference node density 15 %.

(d) MI at reference node density 20 %.

(e) MI at reference node density 25 %.

Fig. 4.5 Mutual information between hop count information and location in x-axis at different reference node density for the case of double communication ranges.

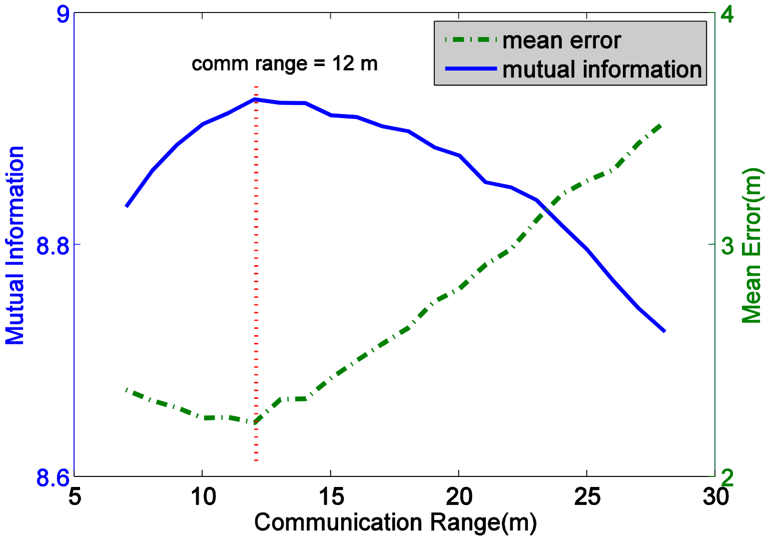
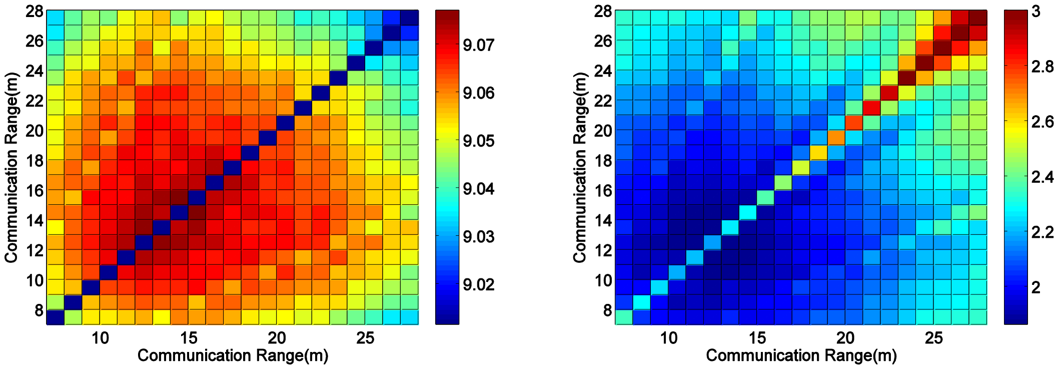


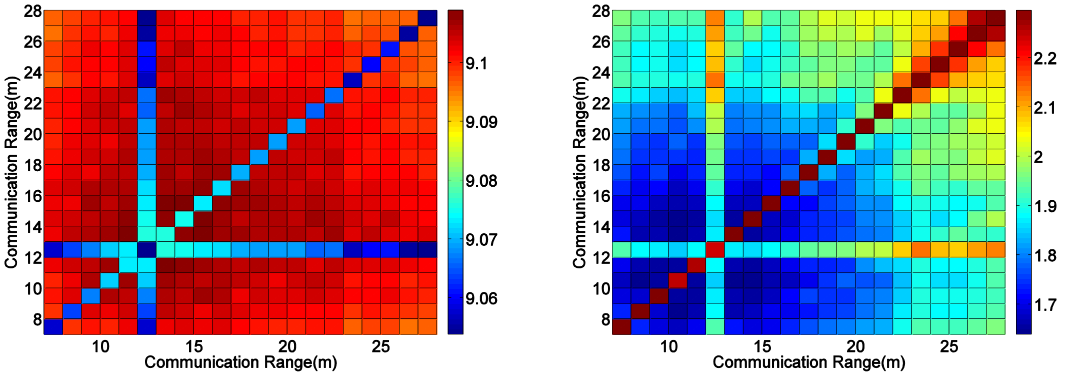
Fig. 4.6 Comparison of mutual information and mean localization error in case of single communication range.



(a) Mutual information.

(b) Mean localization error.

Fig. 4.7 Comparison of mutual information and mean localization error in case of double communication ranges at reference node density 5%.



(a) Mutual information.

(b) Mean localization error.

Fig. 4.8 Comparison of mutual information and mean localization error in case of triple communication ranges at reference node density 5%.

Now, we want to validate how well mutual information corresponds to the localization error. We use unif-SVR localization method with various number of communication ranges at reference node density of 5%. The RBF kernel is used. In the case of triple communication ranges, we fix the third communication range to 12 m, while varying the first and second communication ranges from 7 m to 28 m. The comparison between mutual information and mean localization error in the case of single, double, and triple communication ranges are shown in Fig. 4.6, Fig. 4.7, and Fig. 4.8, respectively.

The result shows that mutual information is consistent with localization mean. High mutual information corresponds to low error rate. The highest point of mutual information corresponds to the lowest point of mean error. For the case of single communication range, Fig. 4.6 the lowest error is at communication range 12 m. For the case of double and triple communication ranges, the lowest error is at the area around 12 m.

We should be aware that even though the optimal data set is selected, it may not be processed in the optimal way by the machine learning algorithm [29]. For example, SVR with linear kernel may not have enough model complexity to learn the data set. Fig. 4.9 shows a comparison between mutual information and mean localization error in the case of single communication range. Linear kernel yields a larger localization error when compared to RBF kernel. The point of lowest localization error also does not correspond to the optimal mutual information.

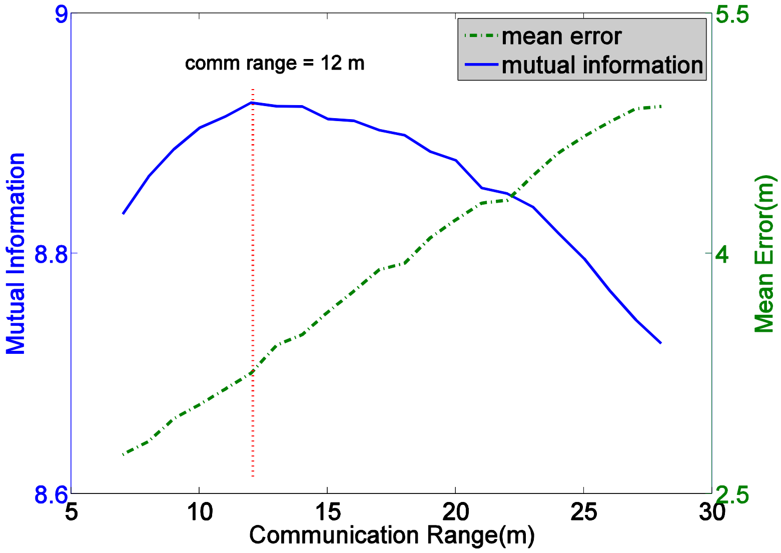


Fig. 4.9 Comparison of mutual information and mean localization error in case of single communication range.

We know that increasing number of communication ranges can improve the accuracy of localization, but at a cost of increased complexity. The question here is: How many communication ranges should we use?

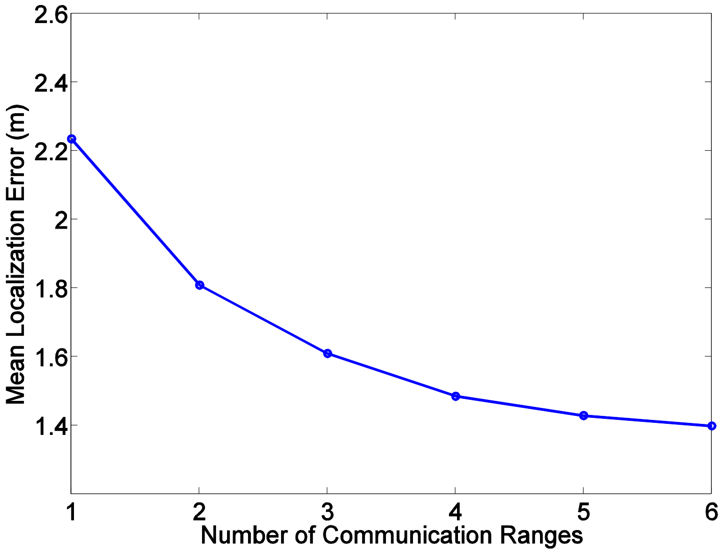


Fig. 4.10 Mean localization error with varied number of communication ranges at reference node density 5%.

We use unif-SVR method with various number of communication ranges at reference node density of 5%. Fig. 4.10 shows optimal mean localization error of different combination of communication ranges. Obviously, increasing number of communication ranges decreases the localization error, but at diminishing return rate. With more than four communication ranges, the improvement becomes negligible. We suggest using around two or three communication ranges would be the most cost effective.

# CHAPTER V

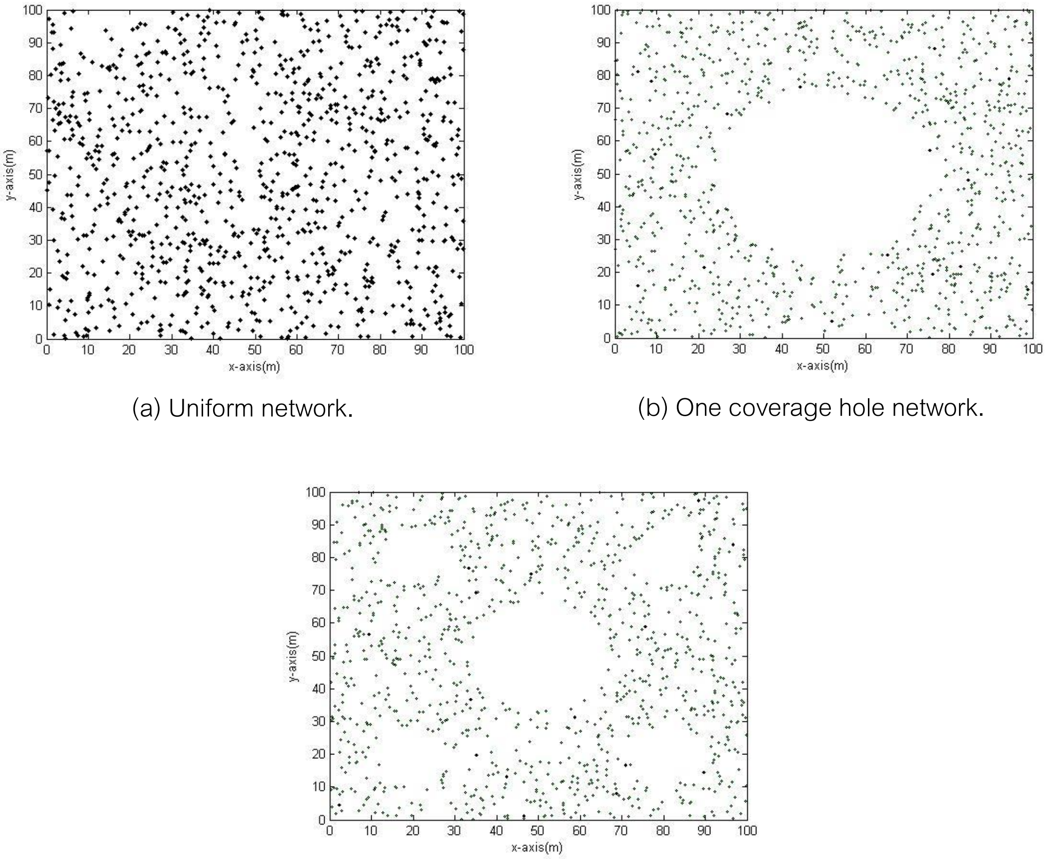
# SIMULATION STUDY

In this chapter, we run simulation to verify the performance of the proposed SVM-based localization method. Both classification (LSVM) and regression (SVR) formulation are considered here. We show how utilizing multiple communication ranges, unif and align method, can improve the result when compared to the case of single communication range. We also try to study how various network parameters affect the localization accuracy.

## 5.1 Simulation Setup

The network setup in this simulation is 1,000 sensor nodes deployed in an area of 100 m x 100 m area. The communication range is set at three different levels ($r = 7$ m, 10 m, and 14 m). The reference node density is set at five different levels; 5%, 10%, 15%, 20%, and 25% ($\ell = 50$, 100, 150, 200, and 250 nodes). The network deployment is considered at three different patterns; uniform network, one coverage hole network, and five coverage holes network. Examples of these network patterns are shown in Fig. 5.1. The simulation is run in MATLAB. LIBSVM [30] library is used for SVC and SVR computation. The kernel function is RBF kernel. For the case of LSVM, the number of classes in each dimension is set to 128. The tuning of SVM hyperparameter ($C$, $\varepsilon$, and $\sigma$) is done by grid search with 5-fold cross-validation [31]. Each simulation setting is run for 50 times and the results are averaged together.

Mean and standard deviation of localization error are shown in the tables for the case of uniform network (Table 1), one coverage hole network (Table 2), and five coverage holes network (Table 3).



(a) Uniform network.

(b) One coverage hole network.

(c) Five coverage holes network.

Fig. 5.1 Node deployment patterns.

## 5.2 Effect of Coverage Hole

In real network, it is not unusual that the sensor nodes are not fully deployed in the area, i.e. there may be some coverage holes. This coverage holes affects the hop count data of the network, as some routing path must avoid these coverage holes. With the occurrence of coverage hole, the localization error is increased. The uniform network has the least localization error and one coverage hole network has medium localization error. Although five coverage holes network has the most localization error, it still does not differ much from the uniform network case.

## 5.3 Effect of Reference Node Density

At higher reference node density, the localization error is decreased. This is as expected, since the number of training samples is directly proportional to the number of reference nodes. However, these reference nodes are costly, so we would prefer to deploy a minimum number of reference nodes. We notice that the localization error is rapidly decreased when the reference node density rises from 5% to 10% and also from 10% to 15%. The improvement at higher reference node density becomes smaller. The most cost effective reference node density is around 10% and 15%.

## 5.4 Effect of Communication Range

With long communication range, the node connectivity becomes greater. When the communication range is increased, the localization error is decreased. The only exception is when reference node density is 5%, communication range of 10 m gives better localization error than 7 m and 14 m. This result corresponds with the mutual information criteria result in chapter IV, which shows that the best communication range depends on the reference node density.

## 5.5 Effectiveness of Proposed Method

By utilizing data from all three communication ranges together, it gives better localization error than using individual communication range. It appears that align method mostly has lower localization error than unif method, except at reference node density of 5%. This can be possible because the align method needs to approximate the kernel align from training data. At low reference node density, the approximation from low number of training data may be not accurate enough. Thus, we suggest using unif method for low reference node density and align method for high reference node density.

SVR outperforms LSVM in all cases. This is reasonable as the location itself is a continuous value. If the fine-grained location is needed, SVR is recommended. LSVM may still be useful in coarse-grained localization, such as room level.

Table 5.1

Average localization error (mean/standard deviation) in uniform network

| Method | Reference Node Density | | | | |
|---|---|---|---|---|---|
| | 5% | 10% | 15% | 20% | 25% |
| LSVM – 7m | 5.08/3.30 | 3.35/2.11 | 2.75/1.72 | 2.44/1.50 | 2.25/1.38 |
| LSVM – 10m | 4.97/3.18 | 3.21/2.02 | 2.58/1.58 | 2.28/1.38 | 2.10/1.24 |
| LSVM – 14m | 5.03/3.20 | 3.22/1.99 | 2.58/1.53 | 2.27/1.33 | 2.06/1.20 |
| LSVM – unif | 4.58/3.05 | 2.89/1.88 | 2.28/1.42 | 1.97/1.19 | 1.78/1.05 |
| LSVM – align | 4.65/3.10 | 2.85/1.82 | 2.23/1.37 | 1.92/1.15 | 1.73/1.01 |
| SVR – 7m | 2.38/1.96 | 1.82/1.26 | 1.63/1.08 | 1.51/0.98 | 1.42/0.91 |
| SVR – 10m | 2.25/1.86 | 1.63/1.11 | 1.43/0.96 | 1.30/0.85 | 1.23/0.80 |
| SVR – 14m | 2.34/1.84 | 1.57/1.10 | 1.32/0.89 | 1.20/0.79 | 1.10/0.73 |
| SVR – unif | 1.68/1.99 | 1.13/1.03 | 0.96/0.81 | 0.87/0.68 | 0.81/0.64 |
| SVR – align | 1.74/1.84 | 1.06/0.81 | 0.92/0.68 | 0.85/0.62 | 0.80/0.56 |

Table 5.2

Average localization error (mean/standard deviation) in one coverage hole network

| Method | Reference Node Density | | | | |
|---|---|---|---|---|---|
| | 5% | 10% | 15% | 20% | 25% |
| LSVM – 7m | 5.12/3.02 | 3.57/2.06 | 2.98/1.73 | 2.62/1.51 | 2.42/1.41 |
| LSVM – 10m | 5.09/2.92 | 3.56/2.06 | 2.96/1.71 | 2.60/1.49 | 2.33/1.30 |
| LSVM – 14m | 5.13/2.98 | 3.54/2.01 | 2.91/1.67 | 2.58/1.46 | 2.31/1.29 |
| LSVM – unif | 4.61/2.72 | 3.15/1.85 | 2.52/1.45 | 2.18/1.24 | 1.97/1.13 |
| LSVM – align | 4.69/2.77 | 3.07/1.79 | 2.47/1.42 | 2.13/1.22 | 1.92/1.08 |
| SVR – 7m | 2.51/1.76 | 1.96/1.29 | 1.70/1.10 | 1.52/0.97 | 1.37/0.84 |
| SVR – 10m | 2.48/1.70 | 1.89/1.22 | 1.65/1.05 | 1.47/0.94 | 1.32/0.84 |
| SVR – 14m | 2.55/1.68 | 1.85/1.19 | 1.61/1.02 | 1.45/0.91 | 1.31/0.82 |
| SVR – unif | 1.90/1.63 | 1.43/1.23 | 1.26/1.09 | 1.13/0.95 | 1.01/0.77 |
| SVR – align | 2.34/1.84 | 1.34/0.98 | 1.19/0.89 | 1.07/0.73 | 0.97/0.64 |

Table 5.3

Average localization error (mean/standard deviation) in five coverage holes network

| Method | Reference Node Density | | | | |
|---|---|---|---|---|---|
| | 5% | 10% | 15% | 20% | 25% |
| LSVM – 7m | 5.35/3.31 | 3.60/2.16 | 2.97/1.74 | 2.60/1.49 | 2.44/1.41 |
| LSVM – 10m | 5.24/3.18 | 3.50/2.07 | 2.87/1.68 | 2.53/1.46 | 2.38/1.38 |
| LSVM – 14m | 5.23/3.17 | 3.49/2.06 | 2.88/1.67 | 2.50/1.42 | 2.34/1.29 |
| LSVM – unif | 4.84/3.04 | 3.14/1.92 | 2.48/1.43 | 2.18/1.27 | 2.02/1.16 |
| LSVM – align | 4.91/3.05 | 3.08/1.86 | 2.46/1.44 | 2.13/1.21 | 1.98/1.13 |
| SVR – 7m | 2.62/1.84 | 2.13/1.49 | 1.83/1.23 | 1.63/1.06 | 1.50/0.94 |
| SVR – 10m | 2.50/1.72 | 1.98/1.34 | 1.74/1.15 | 1.54/1.06 | 1.41/0.91 |
| SVR – 14m | 2.52/1.71 | 1.93/1.34 | 1.66/1.13 | 1.50/0.99 | 1.39/0.90 |
| SVR – unif | 1.93/1.75 | 1.51/1.42 | 1.28/1.13 | 1.16/1.04 | 1.02/0.81 |
| SVR – align | 2.15/1.71 | 1.47/1.18 | 1.27/0.97 | 1.15/0.88 | 1.01/0.70 |

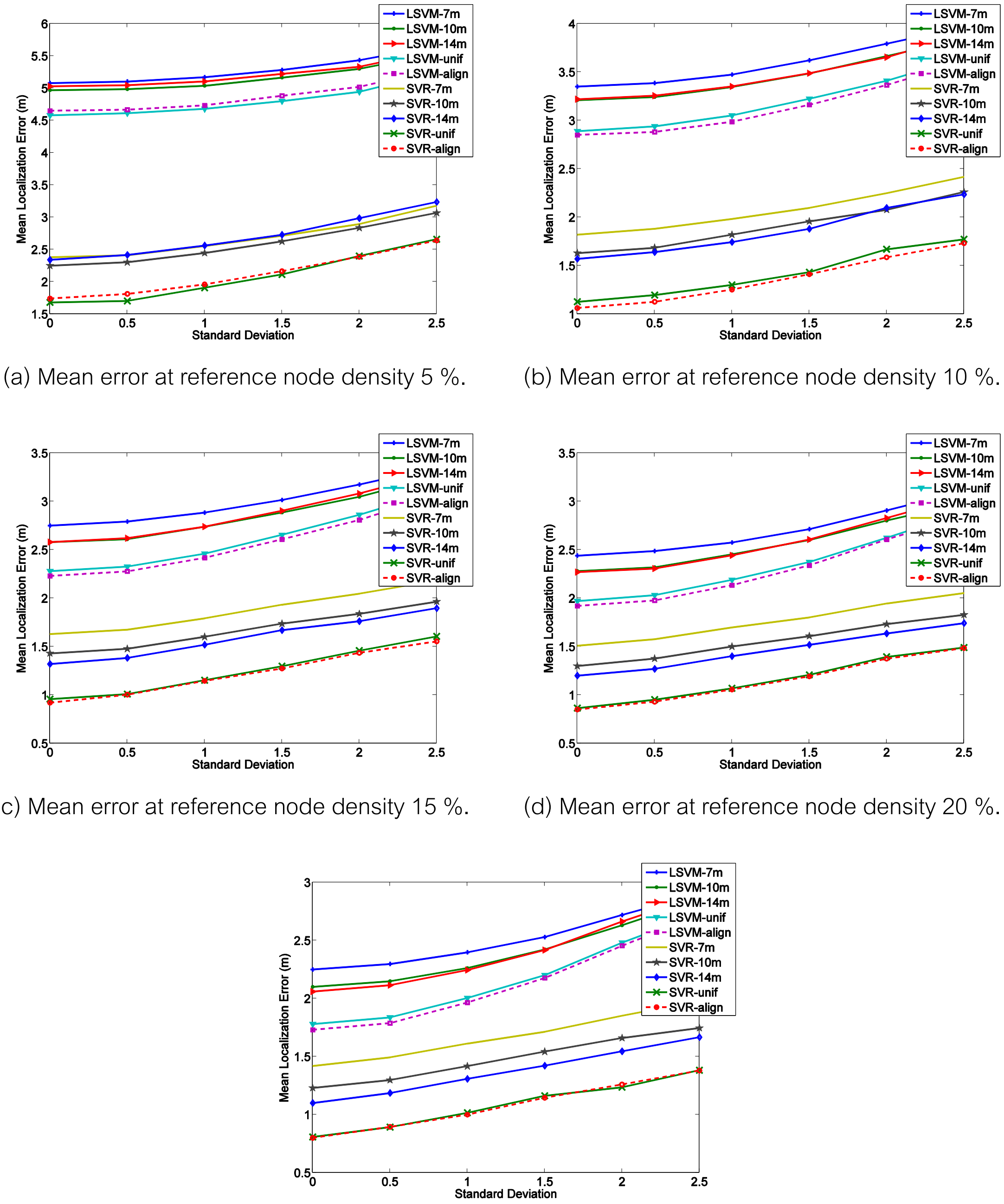
## 5.6 Effect of Reference Node Location Error

In previous simulation, we assume that the known location of reference node is absolutely correct, but in practice there might be some error in the location of reference node itself, e.g. error in human placement or GPS location error. In this simulation, we model this error by adding some Gaussian noise to the reference node location as

$$\tilde{S}_{x,i} = S_{x,i} + \mathcal{N}(\mu, \sigma^2) \qquad (5.1)$$

$$\tilde{S}_{y,i} = S_{y,i} + \mathcal{N}(\mu, \sigma^2) \qquad (5.2)$$

We run the simulation in uniform network with varying standard deviation of zero mean Gaussian noise ($\mu = 0$ and $\sigma = 0$ m, 0.5 m, 1 m, 1.5 m, 2 m, and 2.5 m). Mean and standard deviation of localization error are shown in Fig. 5.2 and Fig. 5.3, respectively.



(a) Mean error at reference node density 5 %.

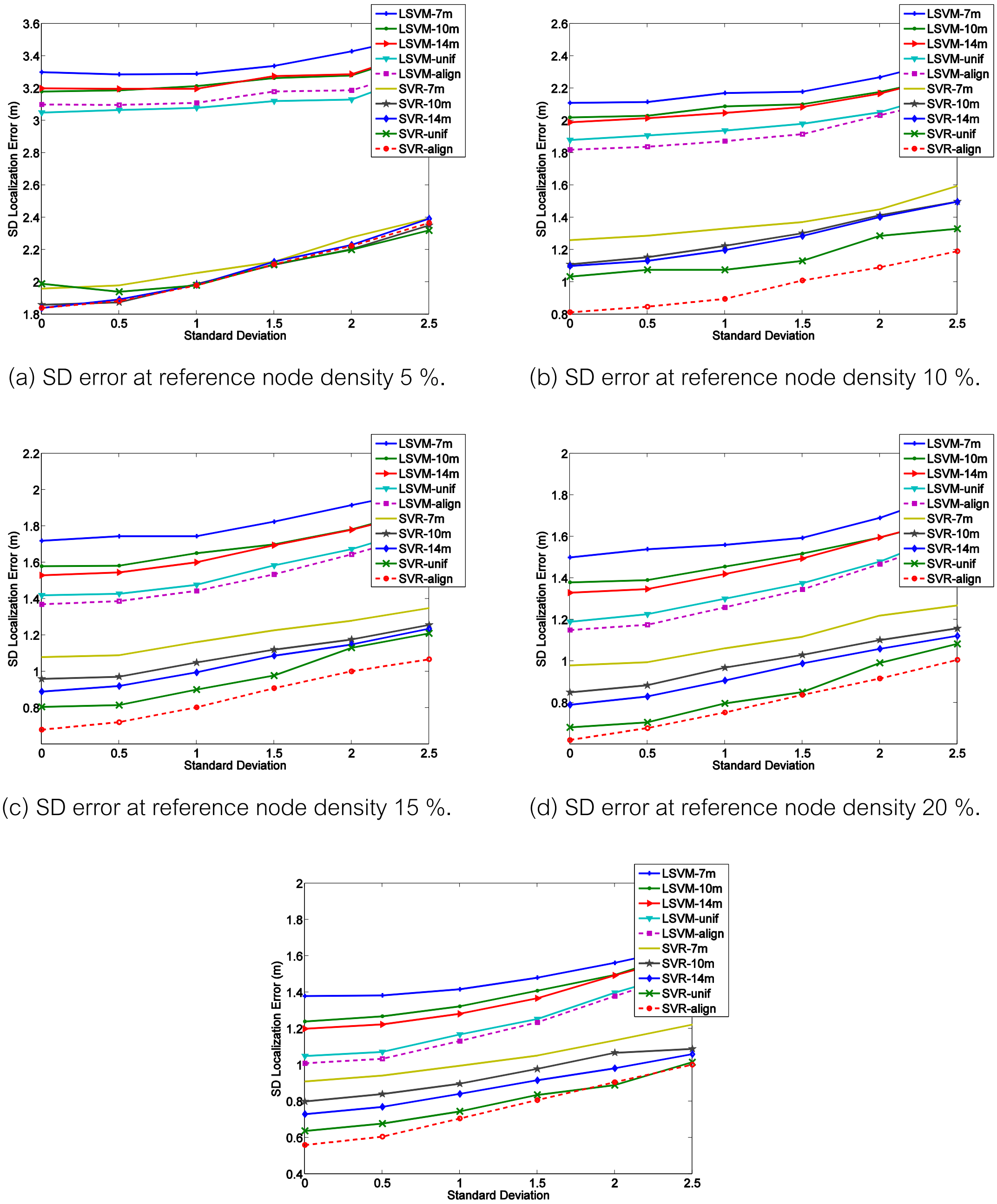
(b) Mean error at reference node density 10 %.

(c) Mean error at reference node density 15 %.

(d) Mean error at reference node density 20 %.

(e) Mean error at reference node density 25 %.

Fig. 5.2 Mean of localization error under the effect of reference node location error at various reference node density.



(a) SD error at reference node density 5 %.

(b) SD error at reference node density 10 %.

(c) SD error at reference node density 15 %.

(d) SD error at reference node density 20 %.

(e) SD error at reference node density 25 %.

Fig. 5.3 Standard deviation of localization error under the effect of reference node location error at various reference node density.

On all reference node density, localization performance is gradually degraded as standard deviation is increased. Both unif and align method still have good localization performance when compared to individual communication range case. Align method is slightly better than unif method. SVR seems to be more robust to the reference node location error than LSVM, as the localization error for SVR rises slower than LSVM. Note that in machine learning field, this kind of error on reference node location is actually noise on the label of training data set.

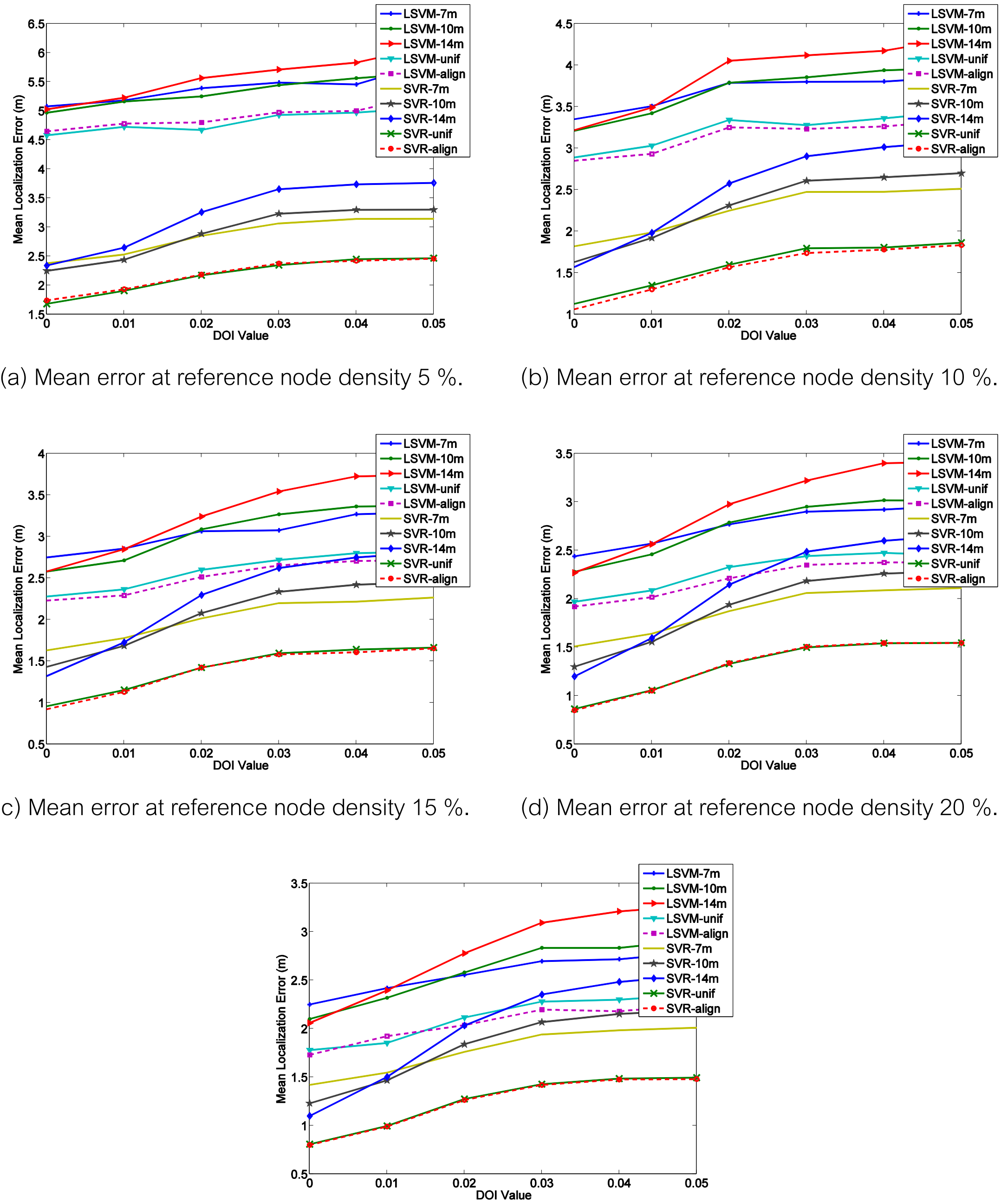
### 5.7 Effect of Noise on Communication Range

In previous simulation, we assume the unit disk communication model; the communication range is constant at all direction. In practice, the wireless signal doesn't follow the unit disk communication model. It can be disturbed by many factors, e.g. signal interference, multipath fading effect, shadowing effect, obstruction from object in the field, and anisotropic antenna pattern. In this section, we try to investigate the effect of the non-ideal communication range. We adopt DOI communication model similar to [6]. This model assumes that the communication range is dependent on the angle. DOI is defined as maximum variation of communication range per unit degree angle variation. Let $R$ be an ideal communication range. The communication range at angle $n$ is defined as

$$r_n = \begin{cases} R & ; n = 0 \\ r_{n-1} + rand * DOI * R & ; 0 < n < 360 \end{cases} \tag{5.3}$$

where $rand$ is a uniformly distributed random number from -1 to 1.

In (5.3), the starting angle is set at 0 degree, but we can randomly select any angle to be the starting angle. For any non-integer angle $n$, the communication range is the linear interpolation of the values from two adjacent angles.



(a) Mean error at reference node density 5 %.

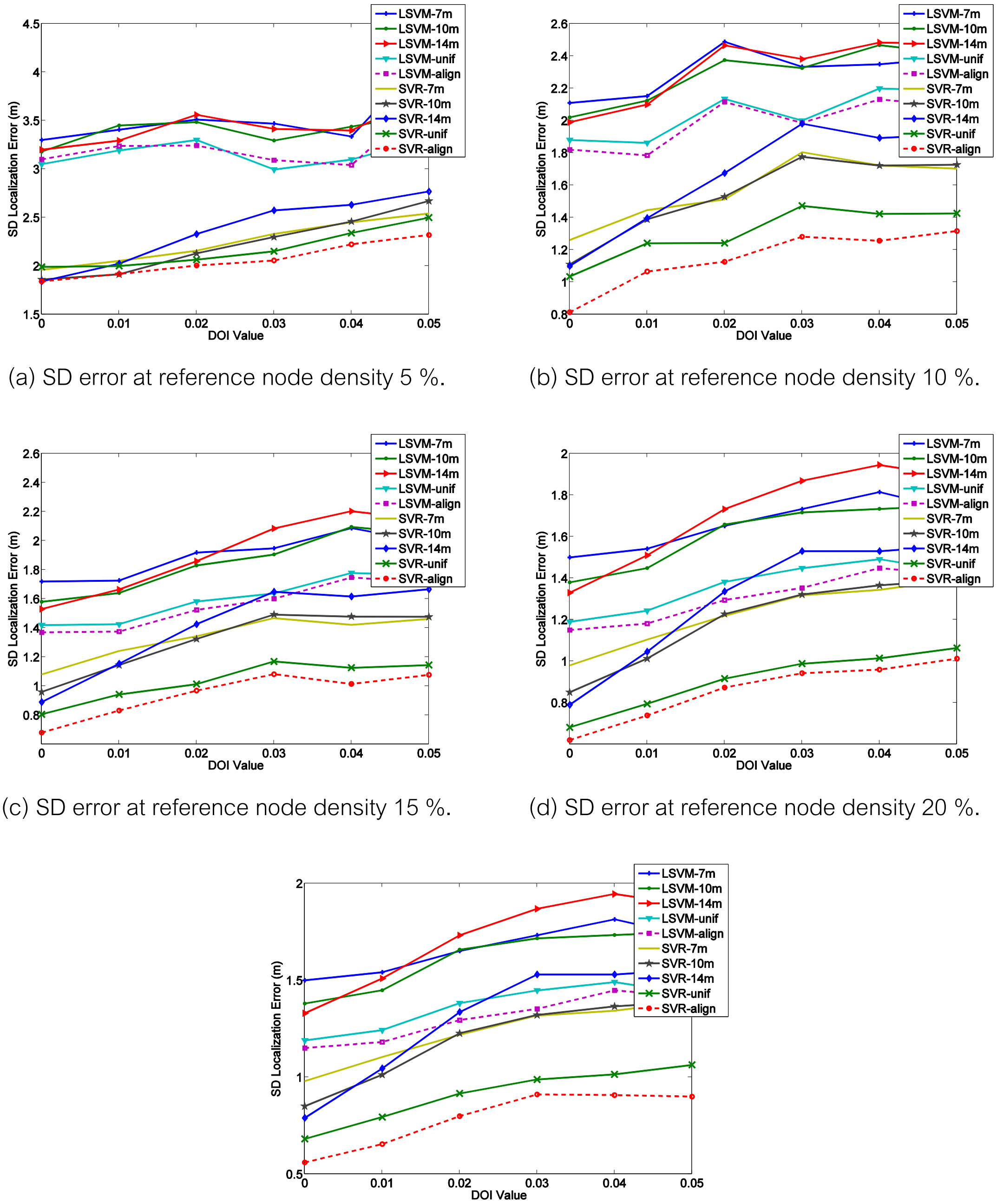
(b) Mean error at reference node density 10 %.

(c) Mean error at reference node density 15 %.

(d) Mean error at reference node density 20 %.

(e) Mean error at reference node density 25 %.

Fig. 5.4 Mean of localization error under the effect of DOI model at various reference node density.



(a) SD error at reference node density 5 %.

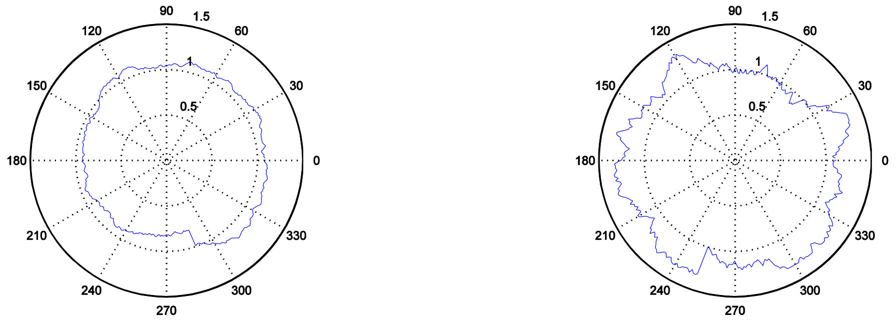
(b) SD error at reference node density 10 %.

(c) SD error at reference node density 15 %.

(d) SD error at reference node density 20 %.

(e) SD error at reference node density 25 %.

Fig. 5.5 Standard deviation of localization error under the effect of DOI model at various reference node density.



(a) DOI = 0.02

(b) DOI = 0.05

Fig. 5.6 Effect of DOI on communication range pattern.

The polar plots in Fig. 5.6 illustrate communication range pattern at two different DOI level; 0.02 and 0.05. As DOI value grows higher, the communication range pattern becomes more irregular. DOI communication model with DOI = 0 is actually equivalent to unit disk communication model. Since each node has different communication range, the two connected nodes can link by either asymmetric link or symmetric link. Asymmetric link is that only one node can reach the other node. Symmetric link is that both nodes can reach each other. In this simulation, we use only symmetric link as valid link in routing process to generate the hop count information.

We run the simulation in uniform network with varying DOI value of the communication model (DOI = 0, 0.01, 0.02, 0.03, 0.04, and 0.05). Mean and standard deviation of localization error are shown in Fig. 5.4 and Fig. 5.5, respectively.

The localization error increases when DOI value becomes higher. For the case of individual communication range, the growing rate of localization error depends on the communication range. Short communication range, as 7 m, has slowest growing rate of localization error. Localization error rises sharply in case of 14 m communication range. Refer to (5.3), level of noise on the communication range pattern depends on the ideal communication range. Noise level is greater on longer communication range. Localization error of unif and align method increases gracefully with respect to the increase in DOI value. Although, unif and align method utilize all three communication ranges, the error rises as slow as 7m case. This result shows that both unif and align method can cope well with this problem. Overall, align method still performs slightly better than unif method, which is consistence with result in previous section.

High DOI value can greatly disturb the hop count data. As the communication range is noisy, hop count data is also noisy. Basically in the field of machine learning, this noisy hop count data is the same problem as noise on the training data point and testing data point.

# CHAPTER VI

# CONCLUSION AND RECOMMENDATIONS

## 6.1 General Conclusion

To achieve improved accuracy of localization in wireless sensor networks, we proposed a framework to utilize multiple transmission power. The localization algorithms in this thesis are based on SVMs. We consider both SVC and SVR. We use SVMs to learn mapping function from hop count data to the node location.

The transmission power of wireless transceiver can be easily adjusted, this consequently changes the communication range. Different communication ranges result in different set of hop count data. By using multiple transmission power, we can generate several hop count data. These hop count data can be combined at various levels of integration; early integration, intermediate integration, and late integration. In this thesis, we consider unif method for early integration and align method for intermediate integration.

Mutual information between hop count data and location can be used as criteria for selection of the best combination set of communication ranges. Optimal set of communication ranges has the highest mutual information which should correspond to the lowest localization error.

The simulation study shows the performance of our proposed localization method. It can work well even in low reference node density or in network with coverage holes. SVR method performs better than SVM in all cases. Regarding utilizing multiple communication ranges, both unif method and align method can greatly improved the localization accuracy when compared with single communication range. They are also more robust to the problem of noise on communication range and reference node locations.

## 6.2 Recommendations

The framework for learning from multiple data representation can be used more extensively than showing in this thesis. Although we only consider using hop count data from multiple transmission power as different data representation, we can also use other data sources. For example, we can treat hop count, RSSI, and TDoA as different data sources to combine them together and achieve better accuracy localization. For intermediate integration, we can consider using other multiple kernel learning methods in place of align method. We can also substitute a variety of machine learning method choices in place of SVMs, especially the kernel-based machine learning algorithm. The localization accuracy also can be further improved by running a post-localized refinement algorithm.

# REFERENCES

[1] Sayed, A.H. Tarighat, A. Khajehnouri, N., **Network-based wireless location: challenges faced in developing techniques for accurate wireless location information, IEEE Signal Processing Magazine**, 22, 4 (2005) : 24-40.

[2] Rui Huang and Gergely V. Zaruba, **Incorporating Data from Multiple Sensors for Localizing Nodes in Mobile Ad Hoc Networks, IEEE Transactions on Mobile Computing**, 6, 9 (September 2007) : 1090-1104.

[3] Ali Shareef and Yifeng Zhu, **Localization Using Extended Kalman Filters in Wireless Sensor Networks, Kalman Filter Recent Advances and Apllications**, InTech, (2009).

[4] Yiu-Tong Chan, Herman Yau Chin Hang, and Pak-chung Ching, **Exact and Approximate Maximum Likelihood Localization Algorithms, IEEE Transactions on Vehicular Technology**, 55, 1 (January 2006) : 10-16.

[5] D. Niculescu and B. Nath, **Ad Hoc Positioning System (APS), Proc. GLOBECOM'01**, 5 (November 2001) : 2926-2931.

[6] T. He, C. Huang, B.M. Blum, J.A. Stankovic, and T. Abdelzaher, **Range-Free Localization Schemes for Large Scale Sensor Networks, Proc. ACM MobiCom'03**, (September 2003) : 81-95.

[7] Guoqiang Mao, Baris Fidan, and Brian D.O. Anderson, **Wireless sensor network localization techniques, Computer Network**, 51 (January 2007) : 2529-2553.

[8] Yanying Gu, Anthony Lo, and Ignas Niemegeers, **A Survey of Indoor Positioning Systems for Wireless Personal Networks, IEEE Communications Surveys & Tutorials**, 11, 1 (2009) : 13-32.

[9] Bernhard Scholkopf and Alexander J. Smola, **Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond**, The MIT Press, 2002.

[10] Duc A. Tran and Thinh Nguyen, **Localization in Wireless Sensor Networks Based on Support Vector Machines, IEEE Transactions on Parallel and Distributed Systems**, 19, 7 (July 2008) : 981-994.

[11] Mauro Brunato and Roberto Battiti, **Statistical learning theory for location fingerprinting in wireless LANs, Computer Networks**, 47, 6 (April 2005) : 825-845.

[12] Xuanlong Nguyen, Michael I. Jordan, and Bruno Sinopoli, **A Kernel-Based Learning Approach to Ad Hoc Sensor Network Localization, ACM Transactions on Sensor Networks**, 1, 1 (August 2005) : 134–152.

[13] Hao Wu, Jiming Chen, Chengqun Wang, and Youxian Sun, A Kernel-based Localization Approach in Wireless Sensor Networks, Proc. IEEE Second International Conference on Future Generation Communication and Networking, 1 (December 2008) : 31-34.

[14] Ruohong Huan, Qingzhang Chen, Kenji Mao, and Yun Pan, A Three-dimensional Localization Algorithm for Wireless Sensor Network Nodes Based on SVM, Proc. IEEE International Conference on Green Circuits and Systems (ICGCS), (June 2010) : 651-654.

[15] Liqun Shan, Jinkuan Wang, Zhigang Liu, and Ruiyan Du, SVM-based Target Tracking in Sensor Networks Combined with Sensor Scheduling, Proc. IEEE International Conference On Computer Design And Applications (ICCDA), 2 (June 2010) : 622-625.

[16] Bin Yang, Jianhong Yang, Jinwu Xu, and Debin Yang, Area Localization Algorithm for Mobile Nodes in Wireless Sensor Networks Based on Support Vector Machines, Mobile Ad-hoc and Sensor Networks Lecture Notes in Computer Science, 4864 (2007) : 561-571.

[17] Keunho Yun and Daijin Kim, Robust location tracking using a dual layer particle filter, Pervasive and Mobile Computing, 3 (2007) : 209-232.

[18] Woojin Kim, Jaemann Park, and H. Jin Kim, Target Localization using Ensemble Support Vector Regression in Wireless Sensor Networks, Proc. IEEE Wireless Communications and Networking Conference (WCNC), (April 2010).

[19] Woojin Kim, Jaemann Park, and H. Jin Kim, Support Vector Learning Approaches for Object Localization in Acoustic Wireless Sensor Networks, Proc. IEEE Intelligent Systems (IS) 5th International Conference, (July 2010) : 485-489.

[20] Alistair Shilton, Bharat Sundaram, and Marimuthu Palaniswami, Ad-Hoc Wireless Sensor Network Localization using Support Vector Regression, ICT Mobile Summit'08, (June 2008).

[21] Wei Wang, Tinglei Huang, Hui Liu, and Fei Pang, Localization algorithm based on SVM-Data Fusion in wireless sensor networks, Proc. IEEE Third International Conference on Genetic and Evolutionary Computing (WGEC'09), (October 2009) : 447-450.

[22] M. Palaniswami, Bharat Sundaram, Jayavardhana Rama G. L., and Alistair Shilton, Target Localization Using Machine Learning, Informatics in Control, Automation and Robotics II, (2007) : 27-33.

[23] V. Vapnik, Statistical Learning Theory, Wiley, 1998.

[24] Paul Pavlidis, Jason Weston, Jinsong Cai, and William Noble Grundy, Learning gene functional classifications from multiple data types, Journal of Computation Biology, 9, 2 (2002) : 401-411.

[25] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh, Two-Stage Learning Kernel Algorithms, Proc. of the 27th International Conference on Machine Learning, (2010).

[26] A. Goldsmith, **Wireless Communications**, Cambridge University Press, 2005.

[27] Texas Instruments, **CC2530 Datasheet**, http://www.ti.com.

[28] Yiming Yang and Jan O. Pedersen, **A Comparative Study on Feature Selection in Text Categorization, Proc. Fourteenth International Conference on Machine Learning (ICML'97)**, (1997).

[29] Roberto Battiti, **Using Mutual Information for Selecting Features in Supervised Neural Net Learning, IEEE Transactions on Neural Networks**, 5, 4 (July 1994) : 537-550.

[30] Chih-Chung Chang and Chih-Jen Lin, **LIBSVM : a library for support vector machines**, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

[31] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin, **A Practical Guide to Support Vector Classification**, http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

# BIOGRAPHY

Prakit Jaroenkittichai was born on 13 May 1987, in Bangkok, Thailand. He received his Bachelor’s degree in Electrical Engineering from Chulalongkorn University in 2009. In the same year, he has been granted a scholarship from Telecommunications Research and Industrial Development Institute (TRIDI), National Telecommunications Commission Fund, to pursue his Master’s degree in Electrical Engineering at Chulalongkorn University. He is a member of Embedded System and IC Design Research Laboratory (ESID), Chulalongkorn University. His current research interests include wireless sensor networks, embedded system applications, and machine learning.