An Efficient and Accurate Indoor Positioning System

Zheng Wu

University of Windsor

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Part of the Electrical and Computer Engineering Commons

Recommended Citation

https://scholar.uwindsor.ca/etd/5623

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.
An Efficient and Accurate Indoor Positioning System

by

Zheng Wu

A Thesis
Submitted to the Faculty of Graduate Studies through Electrical and Computer Engineering in Partial Fulfilment of the Requirements for the Degree of Master of Applied Science at the University of Windsor

Windsor, Ontario, Canada

2015

© 2015, Zheng Wu
An Efficient and Accurate Indoor Positioning System

by

Zheng Wu

APPROVED BY:

___________________________

Dr. D. Wu
School of Computer Science

___________________________

Dr. M. Khalid,
Department of Electrical and Computer Engineering

___________________________

Dr. R. Muscedere, Co-advisor
Department of Electrical and Computer Engineering

___________________________

Dr. R. Rashidzadeh, Advisor
Department of Electrical and Computer Engineering

August 6, 2015
DECLARATION OF
CO-AUTHORSHIP/PREVIOUS PUBLICATION

I DECLARATION OF CO-AUTHORSHIP

I hereby declare that this thesis incorporates the outcome of a research conducted under the supervision of my supervisor, Dr. Rashid Rashidzadeh and co-supervisor Dr. Roberto Muscedere.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers in this thesis, and have obtained written permission from each of the co-author(s) to include copyrighted material(s).

I certify that, this thesis and the research to which it refers, is the product of my research work.

II DECLARATION OF PREVIOUS PUBLICATION

This thesis includes two original papers that have been submitted for publication in peer reviewed journals, as follows:

<table>
<thead>
<tr>
<th>Thesis Chapter/Section</th>
<th>Publication title/full citation</th>
<th>Publication status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 2</td>
<td>A Fast and Resource Efficient Method for Indoor Positioning Using Received Signal Strength</td>
<td>Submitted</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>Improved Particle Filter Based on WLAN RSSI Fingerprinting and Inertial Sensors for Indoor Localization</td>
<td>Submitted</td>
</tr>
</tbody>
</table>
I certify that the above material describes the work completed during my study as a graduate student at the University of Windsor.

I declare that, to the best of my knowledge, this thesis does not infringe upon anyone’s copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in this thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis.

I declare that this is a true copy of my thesis, including any revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.
ABSTRACT

In this thesis, an indoor localization method using on-line independent support vector machine (OISVM) classification method and under-sampling techniques is proposed. The proposed positioning method is based on the received signal strength indicator (RSSI) of Wi-Fi signals. A new under-sampling algorithm is developed to address the imbalanced data problem associated with the OISVM, and a kernel function parameter selection algorithm is introduced for the training process. The time complexity of both the training process and the prediction process are decreased. Comparative experimental results indicate that the training speed and the prediction speed are improved by at least 10 times and 5 times, respectively. Furthermore, through on-line learning, the estimation error is decreased by 0.8m. Such an improvement makes the proposed method an ideal indoor positioning solution for portable devices where the processing power and the memory capacity are limited.

A new Particle Filter (PF) scheme for indoor localization using Wi-Fi received signal strength indicator (RSSI) and inertial sensor measurements has also been presented. RSSI is affected significantly by multipath fading, building structure and obstacles in indoor environments. The information provided by inertial sensors combined with the proposed particle filter are used to develop a positioning algorithm supporting a smooth and stable localization experience. To differentiate similar fingerprints, a single-hidden layer feedforward networks (SLFNs) is used to model the multiple probabilistic estimations and to improve the performance of the PF. A new initialization algorithm using Random Sample Consensus (RANSAC) has also been presented to reduce the convergence time. Experimental measurements were carried out to determine the performance of the proposed algorithm. The results indicate that the positioning error falls to less than 1.2 (m).
DEDICATION

To my loving parents:
Father: Zhijian Wu
Mother: Xiaqiong Wu

To my loving girlfriend:
Yan Wang
ACKNOWLEDGEMENTS

I would like to express my gratitude to those who have taught, supported, encouraged me and those who have cooperated with me for finishing this thesis.

A great gratitude to my best supervisor, Dr. Rashid Rashidzadeh, who showed me the path in this field, supported me greatly and gave wonderful lectures. He is a professor who was involved and provided valuable comments with his broad knowledge.

I am also very grateful to the co-supervisor Dr. Roberto Muscedere from the department of electrical and computer engineering, and the committee member Dr. Mohammed Khalid from the department of electrical and computer engineering and Dr. Dan Wu from school of computer science, for spending their time and efforts in reviewing my thesis and providing valuable suggestions.

I also truly appreciate my colleagues, Bingxin Liu, Ali Attaran, Ali Dianat, Zheng Gong and all of those who assisted me on this thesis.

Finally, millions thanks to my lovely families, my parents and my girlfriend, who have supported me financially and mentally for so many years.

Zheng Wu
# TABLE OF CONTENTS

DECLARATION OF CO-AUTHORSHIP/ PREVIOUS PUBLICATION iii  
ABSTRACT v  
DEDICATION vi  
ACKNOWLEDGEMENTS vii  
LIST OF TABLES xi  
LIST OF FIGURES xii  
LIST OF ALGORITHM xiii  
LIST OF ACRONYMS xiv  

1 Introduction 1  
1.1 Wireless Communication Based Approach 1  
1.2 Inertial Measurement Unit and WLAN Fusion Based Approach 3  
1.3 References 4  

2 A Fast and Resource Efficient Method for Indoor Positioning Using Received Signal Strength 6  
2.1 Related works 9  
2.2 Review of RSSI Based Localization System and On-line Independent Support Vector Machines 12  
2.2.1 WLAN Indoor Localization Problem Setting 12  

viii
4 Conclusion and Future Work

4.1 Conclusion .......................................................... 61
4.2 Future Work ......................................................... 62

Appendix A Selected Code ............................................ 64

Appendix B Co-Authorship Proof .................................. 78

VITA AUCTORIS ......................................................... 79
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Specification of the Benchmarks</td>
<td>25</td>
</tr>
<tr>
<td>2.2</td>
<td>Simulation Results for Benchmarks</td>
<td>25</td>
</tr>
<tr>
<td>2.3</td>
<td>Specification of Measured Indoor Positioning Data Sets</td>
<td>27</td>
</tr>
<tr>
<td>2.4</td>
<td>Simulation Results for Indoor Positioning Data Sets</td>
<td>29</td>
</tr>
<tr>
<td>3.1</td>
<td>Simulation results of RANSAC initialization</td>
<td>53</td>
</tr>
<tr>
<td>3.2</td>
<td>Error distance of different methods</td>
<td>55</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

2.1 Structure of proposed method. ........................................ 16
2.2 Example of the Kernelized Cluster Sifting where Class 4 and 5 are deleted because the inter-cluster distance from positive class is too large. ........................................ 21
2.3 The layout of the measurement area. ................................. 24
2.4 Training time of DNA, Satimage, Letter and IP1 1 of Indoor Positioning data for Grid Search, GA, PSO and Proposed Method. ................................. 27
2.5 Positioning accuracy of the proposed solution versus probabilistic and KNN methods. ........................................ 29
2.6 Error distance under different number of training samples during on-line phase learning of proposed method. ................................. 30
2.7 Accuracy of proposed method with and without utilizing Tomek-link deleting. ........................................ 31
2.8 Testing time of OISVM, LIBSVM, KNN and Proposed Method. ................................. 32
3.1 Scenario of similar fingerprints. Blue region and green region in each graph show similar fingerprints due to the deployment of APs and the building structure. ........................................ 41
3.2 A new particle filter scheme by improved initialization phase and improved weighting process ................................. 44
3.3 Architecture of the SLFNs Interpolation .................................... 51
3.4 Fitted model of RANSAC initialization .................................... 52
3.5 A new particle filter scheme by improved initialization phase and improved weighting process ................................. 54
3.6 Trajectory of fingerprinting method .................................... 56
3.7 Trajectory of PDR .................................... 56
3.8 Trajectories of proposed method .................................... 57
LIST OF ALGORITHMS

3.1 RANSAC-based initialization algorithm . . . . . . . . . . . . . . . . . 45
LIST OF ACRONYMS

ANNs  Artificial Neural Networks
AoA   Angle of Arrival
APs   Access Points
CDF   Cumulative Distribution Function
CNN   Condensed Nearest Neighbor
D-US  Distance-based Under-sampling
ENN   Edited Nearest Neighbor
GAs   Genetic Algorithms
GPS   Global Positioning System
IMU   Inertial Measurement Unit
KCS   Kernelized Cluster Sifting
KF    Kalman Filter
KKT   Karush-Kuhn-Tucker condition
KNN   K-Nearest Neighbor
LBS   Location-Based Services
LS-SVMs  Least Square Support Vector Machines
OISVMs On-Line Independent Support Vector Machines
OSS   One-Side Selection
PDR   Pedestrian Dead Reckoning
PF    Particle Filter

PSO   Particle Swarm Optimization

RANSAC Random Sample Consensus

RANSAC Random Sample Consensus

RFID  Radio Frequency Identification

RPs   Reference Points

RSSI  Received Signal Strength Indicator

SIR   Sampling Importance Resampling

SLFNs Single-hidden Layer Feedforward Networks

SVMs  Support Vector Machines

SVs   Support Vectors

TDoA  Time Difference of Arrival

TLD   Tomek-link Deleting

ToA   Time of Arrival

ToA   Time of Arrival

ToF   Time of Flight

UWB   Ultra-Wide Band

WLAN  Wireless Local Area Network
1 Introduction

Indoor real time positioning and tracking systems have been gaining increasing interest due to the significant progress of mobile devices, portable devices and the necessity of a solution for indoor-location-based services. Many researchers have dedicated their study into development of indoor localization from the aspects of sensor fusion, wireless communication, pattern recognition and robotics [1]. However, there is no dominant solution due to the complexity of indoor environments.

Indoor environments suffer from multiple sources of noises. Normally, there is hardly a line-of-sight transmission between indoor transmitters and receivers [2], which results in fading effect. Moreover, there are a variety of obstacles including walls, columns, book shelves, people and so on. These obstacles affect the strength of wireless signals and cause multi-path effect [3]. Moreover, every building has a unique structure so that it is extremely hard to come up with an accurate signal propagation model [4].

1.1 Wireless Communication Based Approach

Researchers have taken different approaches attempting to overcome these issues. The most popular approach utilizes wireless technologies such as Ultra-Wide Band (UWB), Blue-tooth, Radio Frequency Identification (RFID) and Wireless Local Area Network (WLAN) for indoor positioning. Among these wireless technologies, WLAN is supported by current cell phone. Meanwhile, WLAN, which is widely deployed, enables WLAN-based indoor localization a cost effective and infrastructure free solution.

Time of Arrival (ToA), Angle of Arrival (AoA) and Received Signal Strength Indicator (RSSI) are commonly observed for indoor location estimation. Generally, ToA and AoA require extra hardware. Many existing approaches are based on RSSI of Wi-Fi signal. These approaches consist of ranging-based techniques and ranging-free
techniques. For ranging-based techniques, RSSI is employed to estimate the distance between transmitters and receivers whereas the accuracy of this approach is highly influenced by interferences [5]. Ranging-free techniques are reported to have a more stable performance [2]. The main approach for ranging-free category is based on 'fingerprinting' method.

In fingerprinting method, samples (fingerprints) from different reference points (RPs) are first collected and saved in a radio map database during a site survey process. Relying on the uniqueness of fingerprints from different RPs, pattern recognition algorithms are used to determine the location of items or pedestrians. One major challenges of fingerprinting-based positioning is to ease the workload of site survey and to increase the accuracy.

In this work, an On-Line Independent Support Vector Machines (OISVMs) based approach is proposed to deal with the challenges by its on-line learning ability and its superb classification capability. Among all the pattern recognition methods, SVMs is proven to have an outstanding classification performance [6]. OISVM expanded the classic SVM with extra features. First of all, the on-line learning ability enables the classifier to be trained simultaneously and incrementally with the site survey process. Moreover, the model size of OISVM is smaller as compared with classic SVMs. OISVM checks the independency of all the samples to filter out those dependent data. In OISVM, a parameter is introduced to control the trade-off between accuracy and model size. This flexibility empowers the algorithm to be implemented from mobile scenarios to server platforms.

A fast parameter selection algorithm and an under-sampling scheme are proposed to further optimize OISVM for indoor positioning scenario. With these optimization, the algorithm requires less training time, less prediction time and less memory consumption.
1.2 Inertial Measurement Unit and WLAN Fusion Based Approach

Another popular category for indoor positioning mainly utilizes the Inertial Measurement Unit (IMU). Most of the smart phones and wearable devices such as smart watches and smart wristbands contain IMU module. Pedestrian Dead Reckoning (PDR) method is commonly employed [7]. Essentially, this method is to estimate the travel distance and angle of pedestrians. A step sensor is implemented using accelerometer which is used to detect the displacement of pedestrians. Meanwhile, the gyroscope and/or compass are exploited to detect the orientation. This method is cost efficient as it utilizes available sensors from smart phones and portable devices. For pedestrian localization, smooth location estimations are available. However, as the system cannot calibrate itself, it suffers from the cumulative error problem.

Robotic community has solutions for calibrating issue [8]. The robots take advantage of laser range finder to acquire position information, then perform sensor fusion algorithms such as Kalman Filter, Particle Filter and their variants to fuse the information from IMU sensors and laser observation. Similarly, many reported research works [9] [10] utilize the information from cell phone’s IMU and WLAN observations. However, the performance of WLAN localization is not as accurate as laser localization. Especially the fingerprinting methods produce inaccurate estimations occasionally due to similar fingerprints issue. These inaccurate estimations reduce the final accuracy after fusing with noisy IMU data.

In this work, a new particle filter (PF) scheme is proposed to tackle the inaccurate WLAN estimation problem. Random Sample Consensus (RANSAC) [11] algorithm is used for the initialization. As a result, it is not required to have extra hardware nor a slow global initialization phase. It picks the inliers from fingerprinting estimations by a Gaussian PDR model. To overcome the wrong estimations during the on-line phase, a modified version of traditional PF is proposed, in which multiple fingerprint-
ing probabilities is interpolated by SLFNs [12] interpolation. The experiments show improvements on both initialization phase and on-line estimation phase.

The chapters are organized as follows. The improved OISVM based fingerprinting algorithm is discussed in the next chapter. Chapter 3 covers the modified particle filter scheme that propose a RANSAC-based initialization and SLFNs-based weighting. The last chapter concludes this thesis.

1.3 References


2 A Fast and Resource Efficient Method for Indoor Positioning Using Received Signal Strength

With the development of wireless communication, the concept of ubiquitous computing is getting popular. Mobile devices such as smart phones and tablets are now universal. The demand for ambient intelligence which enables the system to be aware of the presence of users is growing [1–3]. The key issue in ambient intelligent is to know whereabouts by virtue of received wireless signals. There are many ongoing studies of indoor subject positioning or localization. An accurate, reliable and real-time localization system is able to determine the position of any portable device, which can be used for navigation, monitoring or tracking and other location-based services (LBS).

Mature outdoor localization systems like global positioning system (GPS) has been developed successfully [4,5]. Indoor environments are more complex as there is hardly a line-of-sight transmission between transmitters and receivers.

A variety of wireless technologies, such as Ultra-Wide Band (UWB), Bluetooth, Radio Frequency Identification (RFID) and Wireless Local Area Network (WLAN) have been used for indoor positioning. WLAN that operates in the 2.4GHz frequency band has become common in public environments, offices, hospitals, academic areas and industry regions in recent years [6]. Existing WLAN infrastructure allows researchers to consider WLAN-based indoor localization as a cost effective and viable solution. Time of Arrival (ToA) [7] and Angle of Arrival (AoA) [8] methods have been used for indoor positioning. The Received Signal Strength Indicator (RSSI) has also been utilized for indoor location estimation without the requirement of extra hardware for implementation [9].

Typically, there are two categories of algorithms for RSSI positioning [10]. The first category can be defined as geometric-related techniques, which utilizes the em-
pirical path loss models for distance estimation. Then triangulation or modeling algorithm is performed for position estimation. The main drawback is that the distance estimation is inaccurate due to the complex propagation phenomenon in indoor environment.

The other category is defined as position-related techniques, which is also called ‘fingerprinting’. In these methods, samples (fingerprints) from different reference points (RPs) are first collected as priori knowledge (radio map database) in a site survey process. Relying on the uniqueness of the fingerprints in different RPs, all kinds of pattern recognition algorithms can be leveraged to implement the application with a reasonable accuracy. One of the major challenges of fingerprinting based positioning is to shrink the workload of site survey. One of the promising methods is to build the radio map with a little work through the crowdsourcing, such as Zee [11]. Crowdsourcing enables the service provider to have a faster deployment speed, less maintenance efforts and higher resistance to environmental dynamics. In this technique, instead of labeling the training data coordinates manually, pedestrian dead reckoning and map matching algorithms are used to assign the coordinates without any human efforts. However, as the training data comes sequentially and randomly, service provider has to train the model regularly from scratch. In this paper, On-line Independent Support Vector Machine (OISVM) is introduced to solve this problem through its on-line learning ability which ensures simultaneous training phase and data collection phase. Furthermore, on-line learning enables the classifier to be trained incrementally.

Among all the pattern recognition methods, SVM has shown an outstanding performance in theory and in practice [12]. OISVM expanded the classic SVM with extra features. Other than on-line learning ability, there are two main features best suitable for the mobile scenario. First, its model size is smaller compared with classic SVMs. OISVM checks the independency of all the samples to filter redundant data. Second,
a parameter is introduced to control the trade-off between accuracy and model size. Benefits of smaller model size are not only less memory space occupation, but more importantly, a faster and power efficient prediction phase, which is crucial when we design a robust commercial mobile application. Another advantage is that OISVM is easy to be extended to one-vs-all multi-classification. When using one-vs-all methodology, the kernel matrix of OISVM is the same for all the machines. This is crucial for indoor positioning problem where there are always hundreds of classes.

Based on the OISVM, optimizations in each step are performed to make it more suitable for indoor positioning. In the training phase, normally, dozens of gamma values are explored in the grid search to get an optimal accuracy. In this work, a new gamma selection method is introduced to reduce the time cost. Values of gamma are determined by the distance calculation of inter-clusters before the SVM training phase with little computational complexity. Therefore, the training time of the proposed method could be much faster than the traditional methods. Meanwhile, different gamma values instead of single one are chosen for different models to ensure a higher accuracy.

Using the inter-cluster distance calculated above, we introduce a new under-sampling algorithm which is consist of kernelized cluster sifting (KCS), distance-based under-sampling (D-US) and Tomek-link Deleting (TLD). The intention of the under-sampling is to reduce the complexity of SVM. Because SVM shows too much complexity both in training and testing phases compared with classic algorithms such as KNN or Bayes classifier. In addition, one-vs-all multi-classification strategy introduces the imbalanced data problem that degrades the accuracy.

In mobile environment, the key point is to decrease the prediction time and complexity while improving the positioning accuracy. With the proposed under-sampling algorithm, results show that the prediction time is more than ten times shorter than existing SVMs by adding no more than 0.1m error distance. It is 75% shorter in
prediction phase with 0.3m less error distance compared to classic algorithms. With on-line learning enabled, results show up to 1m error distance reduction.

Shorter training phase is also helpful. In most cases, the training phases are performed on the server, it saves the deployment resources for the service providers. In some cases, the clients are not willing to disclose their location to the server. Or sometimes positioning of a small area is preferred, like inside the house of a client. A shorter training phase on the mobile devices enables a client to create their own navigation system. With the proposed \( \gamma \) selection method, the training phase of OISVM is decreased by more than 10 folds.

The rest of the paper is organized as follows. Background and related works are covered in section II. Section III presents a mathematical representation of the base algorithm used in this paper. Detail formulations of proposed method are covered in section IV. Simulation and experimental measurement results are demonstrated in section V, followed by conclusions.

2.1 Related works

Fingerprinting techniques can be categorized into two groups [6]: (a) probabilistic techniques and (b) deterministic techniques. Probabilistic techniques are reported to be more robust to noise and disturbance, but it is difficult to construct an explicit RSS distribution and they also suffer from a relatively high computational complexity in general. Probabilistic techniques calculate and store off-line RSSI distributions and utilize probabilistic techniques, for instance Bayes inference to estimate the location of user through on-line RSS observations [9]. Deterministic techniques construct classification or regression models using radio map database. Deterministic modeling techniques for instance k-nearest neighbor (KNN) [13], artificial neural networks (ANNs) and support vector machines (SVMs) have been used for indoor localization [14]. Compared to other deterministic modeling techniques, SVMs can model
linear and nonlinear relations with better generalization performance. They are more robust to ‘noise’, easier to implement and require less number of parameters to specify. Standard SVMs and least square SVMs (LS-SVM) have been applied to indoor localization [15, 16]. As a popular machine learning technique, SVMs can be used in its regression version or classification version. Wu et al. [17] proposed a solution to open area location estimation problems through SVM regression methods and showed promising performances. For complicated indoor environments, SVM classification has the potential to outperform other typical methods [18]. The work of [19] proposed SVM classification to solve the indoor positioning problem. Yet, as classic SVM classification was performed, they had to rebuild the model whenever the environment changed.

As SVM based methods are time consuming and require lots of memory when support vectors(SVs) become large, Orabona et al. proposed a novel version of SVM to shrink requirements for time and space, named on-line independent support vector machine (OISVM) [20]. In this method, the computing complexity can be controlled by a factor called parameter-tolerance and the performance is improved by on-line learning. In general, indoor localization can be considered as a multi-class classification problem, but SVM classifiers deal with two classes.

There are two typical methods to use SVMs for multi-class classification: one-versus-one and one-versus-all. In these methodology, the predicted label, $y_{pred}$, is determined based on the voting scheme. The final value is identified as the $y_{pred}$ of the RP with the maximum votes.

In one-versus-one classifier, the SVM model is built for each pair of classes. Define $r$ as the number of RPs, so each pair of classes forms a pair of classes in SVM. As a consequence, $r(r - 1)/2$ SVM classifiers are constructed and used. Usually, there are dozens or even hundreds of RPs in an indoor positioning scenario. The number of classifier models is in a quadratic relationship with the number of RPs. Therefore, too
many classifiers should be computed and stored. Additionally, it results in a higher prediction time complexity which is linear with the number of SVs and classifier models.

In one-versus-all methodology, each SVM classifier is trained between one class and all the other classes. However, this leads to imbalanced training data that degrades the positioning accuracy, increases the time of training and raises the number of SVs.

The imbalanced data problem from classifier has raised the attention in many fields [21]. Methods in three major directions show more potential [22]. The first one put efforts in data resampling. One can under-sample the majority class or over-sample the minority class. The second one compensates the result during classification. The third one gives penalty to the errors after classification. Considering the characteristics of SVMs based localization on mobile devices, the focus of this work is to utilize the under-sampling method.

The simplest under-sampling method is the random under-sampling [23]. It balances the data set by removing the examples randomly. Besides simplicity, another merit is that it does not take the advantage of training data information. However, it may discard some data potentially important for the classification process. Many intelligent under-sampling methods were proposed to remove samples with heuristic information. For example, Tomek defined a rule to find the noise or borderline samples, called Tomek links [24]. Accuracy of classifier can be improved by removing those noise samples. Laurikkala [25] applied the Wilsons Edited Nearest Neighbor Rule (ENN). Samples are deleted if their labels are different from more than half of their neighbors. Hart [26] proposed condensed nearest neighbor (CNN) rule to find a consistent subset of examples. Those samples that are far away from decision borders are wiped out. One-side selection (OSS) method [27] applied Tomek-links deleting followed by CNN. Gustavo et al. proposed CNN plus Tomek-link deleting which is similar to OSS [28]. All aforementioned under-sampling methods are used for uni-
versal classifiers in two-class circumstances, but for localization based on OISVM, further optimization steps should be considered. Firstly, it is reasonable that undersampling rules should be generalized to nonlinear feature space. Secondly, characteristics of class distribution can be used to speed up the algorithm. The proposed under-sampling method in this work combines these optimization steps.

2.2 Review of RSSI Based Localization System and On-line Independent Support Vector Machines

2.2.1 WLAN Indoor Localization Problem Setting

Given a set of RPs with label, denoted as \([y_1, y_2...y_r]\), then define the position of each RP with the coordinates \(p(y_i) = [\hat{x}_i, \hat{y}_i]\). Normally there are two stages in the fingerprinting solution. In the first stage, multiple RSSI samples are collected in the off-line phase for the radio map database construction, denoted as \(R = [S_1, S_2...S_r]\), where \(S_i\) records \(m\) samples at location \(i\). Each sample is a vector of RSSI, denoted as \(x_i = \{SSI_{i,1}, SSI_{i,2}...SSI_{i,l}\} \in \mathbb{R}^l\). Each sample is labeled by its RP. So \(S_i\) is denoted as

\[
S_i = \begin{bmatrix}
SSI_{1,1} & \cdots & SSI_{1,l} & y_i \\
SSI_{2,1} & \cdots & SSI_{2,l} & y_i \\
\vdots & \vdots & \ddots & \vdots \\
SSI_{m,1} & \cdots & SSI_{m,l} & y_i 
\end{bmatrix}.
\]

The database can be considered as a prior of the characteristics of indoor WLAN signals strength. In the second stage, the system reads a new set of signal strength \(x_t = \{SSI_{t,1}, SSI_{t,2}...SSI_{t,l}\}\), the key problem of indoor localization and tracking is to find the label \(y_t\) and corresponding coordinates \(p(y_t)\) of this vector. In this way, the indoor localization problem can be considered as a typical multi-class classification
2.2.2 Review of On-line Independent Support Vector Machines

A brief review is given for comprehensive understanding of the theory of the following adopted OISVM. The classic problem settings of SVM are given as follows

$$\arg \min_{W,b} \frac{1}{2} \|W\|^2 + C \sum_{i=1}^{n} \xi_i^p,$$

subject to $y_i (W(\phi(x_i) + b)) \geq 1 - \xi_i, i = 1, ..., n,$

(1)

where $W(\phi(x) + b)$, and $b \in \mathbb{R}$ are the decision boundary, $y_i$ is the label of $x_i$, $C \in \mathbb{R}^+$ is the error penalty coefficient, and $\xi_i$ is the approximation of the number of misclassified samples. Normally, $p$ is set to be 1 or 2. Essentially, this is to solve an optimization problem with linear boundary. Lagrangian form and Karush-Kuhn-Tucker (KKT) conditions [12] are commonly chosen for solving the constrained optimization problem. In non-linear boundary cases, a kernel function $K(x,z) = \phi(x) \cdot \phi(z)$ is introduced to map the samples into a high-dimension space, where the problem becomes linear separable. Finally, the decision boundary is reformulated as

$$f(x) = \sum_{i=1}^{n} a_i y_i K(x,z),$$

(2)

where $a_i$ is Lagrangian coefficient. Solving this problem by Lagaragian form and KKT condition leads to two demerits. One is that SVM complexity solution grows linearly with the number of training samples and thus large memory capacity and long training time are required when facing big dataset. The other is that typical SVM classifier adopts batch learning, which means training has to perform from scratch when new training data is collected.

OISVM overcomes these disadvantages by introducing the concepts of basis vectors and on-line learning ability. The theory of OISVM is explained as follows [20]:
Building the basis matrix  Many values of $a_i$ in (2) are found to be zero, meaning that they are linearly dependent to other vectors. To achieve a sparser representation of $f(x)$ , the basis matrix is constructed after deleting all the dependent vectors. In order to determine the independency of a vector, define the discriminant as

$$\Delta = \min_c \left\| \sum_{j \in B} c_j \phi(x_j) - \phi(x_{n+1}) \right\|^2 > \eta. \quad (3)$$

$B$ is the index of vectors in the basis matrix. $\Delta > \eta$ denotes that the new vector $x_{n+1}$ is linearly independent from the basis vectors if exist any $c_j \in \mathbb{R}$. It is clear that when $\Delta > 0$, the new vector $x_{n+1}$ is linearly independent to other basis vectors and can be added to $B$. A small positive value $\eta$ is introduced to release the lower bound of $\Delta$ from 0 to $\eta$. Therefore, $x_{n+1}$ is a basis vector when $\Delta > \eta$. Intuitively speaking, $\eta$ controls the trade-off between accuracy and the size of $B$. Substitute with the kernel function $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$ and take derivative with respect to $c$ to calculate the minimum value. Finally, when $c = K_{BB}^{-1}k$, $\Delta$ is rewritten as

$$\Delta = K(x_{n+1}, x_{n+1}) - k^T K_{BB}^{-1}k, \quad (4)$$

where $k_i = K(x_i, x_{n+1})$, $i \in B$ and $K_{BB}$ is the kernel matrix of the basis matrix. In this way, a full rank basis kernel matrix is built and it is easy to get the inverse of the matrix by the matrix inversion lemma.

Training incrementally  A modified Newton method [29] has been applied to solve (1) instead of using Lagragian form and KKT condition. From previous step, the constrained problem is reformulated into an unconstrained minimization problem:

$$\arg \min_\beta \frac{1}{2} \beta^T K_{BB} \beta + \frac{1}{2} C \sum_{i=1}^{n} \max (0, 1 - y_i K_{iB} \beta)^2 \quad (5)$$
where \( K_{iB} \) denotes the \( i \) row in \( K_{BB} \). Finally, the algorithm updates \( B \) with the Newton method.

These two steps not only enable OISVM to inherits those advantages of standard SVM, but more importantly, introduce three main features that can be utilized for indoor positioning:

- The tolerance factor \( \eta \) can be used to control the trade-off between accuracy and model size. In some mobile scenario, such as emergency cases, prediction phase needs to be performed on mobile devices. Thus the size of model becomes crucial. \( \eta \) enables clients to tune the model size.

- As Newton method has been applied to solve (5), on-line learning ability is available. Therefore, less efforts and time are required from service providers to maintain the system. Combined with crowdsourcing, fast deployment and higher flexibility can be achieved. Meanwhile, any changes in environment or from access points (APs) can be sensed and learned synchronously by the system.

- One drawback, that SVM is not suitable for multi-class classification, can also be handled by OISVM. As the labels are not used during the sparsification phase, the kernel matrix remains unchanged for all the one-versus-all multi-class problems.
2.3 The Proposed Localization Algorithm

As depicted in Fig. 2.1, the proposed method includes off-line stage and on-line stage. At the off-line stage, the system is trained by pre-collected RSSI dataset. Initially, it calculates the distance of all classes in the kernel space using the radial basis function...
(RBF) kernel for simplicity. The RBF kernel is expressed as

\[ K(x, y) = e^{-\gamma \|x - y\|^2}; \gamma > 0. \]  

(6)

Kernel function parameter \( \gamma \) is optimized for each classifier. Then a novel under-sampling algorithm is applied to decrease the computational complexity. In the process of under-sampling, kernel distance is utilized both in kernelized cluster sifting and distance-based random under-sampling. In the end, the Tomek-link deleting is performed.

During the on-line stage, on-line prediction and on-line training are performed based on discriminant function (4). All the new data are sent to the prediction phase for location estimation. If the new data is independent, it is sent to the on-line training phase as well. Thus with each independent sample of data, the method updates the training model simultaneously. In this way, the model can be updated simultaneously by on-line learning with negligible training time.

**Remark.** It should be noted that the proposed approach not only makes it possible to be implemented in a real-time manner, but also enhances the localization accuracy due to periodical on-line training.

The detail description of kernel parameter optimization, under-sampling procedure and complexity computation is given in the following subsections.

### 2.3.1 Kernel Parameter \( \gamma \) Selection

As mentioned at the previous section, the Radial basis function can be optimal option as a kernel function [11]. In general, norm 2 is chosen for RBF kernel.

Unlike other kernels, RBF kernel has only one kernel parameter \( \gamma \). Therefore, there are three parameters \((\gamma, C, \eta)\) in OISVMs that are required to be defined when RBF kernel is used. Several methods have been proposed to select parameters for
OISVMs. The grid search algorithm is a straightforward way [30]. In this algorithm, the SVM is trained with all desired sets of parameters to select the optimal that has the highest accuracy. Some intelligent algorithms such as the genetic algorithms (GAs) and particle swarm optimization (PSO) [31] are also applied to search the SVM parameters. Although parallel processing skill in these algorithms can speed up the processing time, the computational complexity and the required memory are still very high. With a large sample size, this may even cause the processing system to crash.

In practice, changing the penalty parameter $C$ affects the weight of error terms. The selection of kernel parameters $\gamma$ has impact on feature spaces. Kernel method increases the distance between two classes so that they are separable in the feature space. The larger the distance between classes, the easier the classes are to separate. Based on this idea, Wu and Wang [32] proposed several indices to measure the distance of inter-clusters in the feature space. Simulations show that the distance between class means $\delta$ is more robust than other indexes. From the data set $x_+$ (positive class) and $x_-$ (negative class), $\delta$ can be calculated as:

$$
\delta_F(x_+, x_-) = d(\bar{\phi}(x_+), \bar{\phi}(x_-))
= d\left(\frac{\sum_{x_+ \in x_+} \phi(x_+)}{l_+}, \frac{\sum_{x_- \in x_-} \phi(x_-)}{l_-}\right)
$$

where $l_+$ and $l_-$ are the sizes of $x_+$ and $x_+$, $\bar{\phi}(x_+)$ and $\bar{\phi}(x_-)$ are the class means in the feature space, and $d(\bar{\phi}(x), \bar{\phi}(y))$ is the inter-cluster distance between $x$ and $y$ in the feature space, which can be calculated by

$$
d(\bar{\phi}(x), \bar{\phi}(y)) = \sqrt{\|\phi(x) - \phi(y)\|^2}
= \sqrt{K(x, x) + K(y, y) - 2K(x, y)}. $$
Then substitute (8) into (7), the equation is rewritten as

\[
\delta_F(x_+, x_-) = d(\bar{\phi}(x_+), \bar{\phi}(x_-))
\]

\[
= \sqrt{\frac{\sum_{x_{+i} \in X_+} K(x_{+i}, x_{+j})}{l_+^2} + \frac{\sum_{x_{-p} \in X_-} K(x_{-p}, x_{-q})}{l_-^2} - 2 \frac{\sum_{x_{+m} \in X_+} K(x_{+m}, x_{-n})}{l_+ l_{-\text{nearest}}} - \frac{\sum_{x_{-p} \in X_{\text{nearest}}} K(x_{-p}, x_{-q})}{l_-^2}}
\]

(9)

The distance \( \delta_F \) should be computed with all desired values of \( \gamma \) in parameter search space, for example, \( \gamma \in [0.1, 0.2, \ldots 10] \). The value that gives the largest \( \delta_F \) is selected as optimal \( \gamma \) of SVM. The optimal value is essentially the one that gives the maximum margin between two classes \( x_+ \) and \( x_- \) in the mapping space. The reported results in [32] show that proper kernel parameters can be chosen by \( \delta_F \).

For the proposed indoor localization application, optimal value of \( \gamma \) was selected for every one-against-all classifier respectively. For the \( i \)-th classifier, \( x_+ \) consists of the samples in the \( i \)-th RP, called positive class. \( x_- \) includes all samples in the remaining RPs, namely negative class. The computation complexity of (9) is \( O((r-1)^2 l_+^2) \), which is a computationally intensive process.

To reduce the required processing time, \( \delta_F \) can be estimated by calculating the inter-cluster distance between \( x_+ \) and the nearest subset of \( x_- \) in the feature space. A modified distance is proposed by:

\[
\delta_{\text{nearest}}(x_+, x_-) = d(\bar{\phi}(x_+), \bar{\phi}(x_-))
\]

\[
= \sqrt{\frac{\sum_{x_{+i} \in X_+} K(x_{+i}, x_{+j})}{l_+^2} + \frac{\sum_{x_{-p} \in X_{\text{nearest}}} K(x_{-p}, x_{-q})}{l_-^2} - 2 \frac{\sum_{x_{+m} \in X_+} K(x_{+m}, x_{-n})}{l_+ l_{-\text{nearest}}} - \frac{\sum_{x_{-p} \in X_{\text{nearest}}} K(x_{-p}, x_{-q})}{l_-^2}}
\]

(10)

where \( x_{-\text{nearest}} \) is the subset (with specific label) of \( x_- \), noted by \( x_{-\text{sub}} \), with minimum inter-cluster distance from \( x_+ \) in the feature space. is the solution of the following
problem

\[
\mathbf{x}_{\text{nearest}} = \arg\min_{\mathbf{x}_{\text{sub}} \in \mathbf{x}_-} d_F(\bar{\phi}(\mathbf{x}_+), \bar{\phi}(\mathbf{x}_{\text{sub}})),
\]

where \( \mathbf{x}_{\text{sub}} \) is a vector in a subset, \( d(\bar{\phi}(\mathbf{x}_+), \bar{\phi}(\mathbf{x}_{\text{sub}})) \) is the class mean distance in the feature space, which can be calculated from (9). The computational complexity of (10) and (11) is \( O((r - 1)l_+^2) \). Note that compared to the computational complexity of (9), \( O((r - 1)^2l_+^2) \), this is a significant improvement when \( r > 100 \) which is the case in most indoor localization scenarios.

The penalty parameter \( C \) in OISVM can be chosen by the cross validation process. The tolerance factor \( \eta \) is chosen based on the trade-off between accuracy and speed of OISVM. In practice, \( \eta \in [0.01, 0.1] \) ensures a strong performance.

### 2.3.2 A New Under-sampling Approach

As stated before, with one-against-all technique, the ratio of positive samples and negative samples is 1 : \((r - 1)\). When \( r \) is large, the positive and negative class is strongly imbalanced. This imbalance has a serious impact on the performance of OISVM classifiers, because the results of classifiers are prone to majority class. Furthermore, the use of all samples in each classifier results in large kernel matrix, which requires extra calculation and memory. The proposed under-sampling method deals with these problems.

**Kernelized Cluster Sifting** At first, the distance should be generalized from original space to the feature space by the kernel trick. It is straightforward to compute the Euclidean distance of two points in the feature space by mapping function. This can be realized through (8). Schlkopf [33] proved that the distance defined in (8) is translation invariant if the kernel function is conditional positive definite. Most common kernels satisfy this condition.
With one-against-all technique, the negative data set $x_-$ consists of all the examples from $r - 1$ RPs. In fact, examples from one location form a cluster in the feature space, although there is overlap in different clusters. Similar to (9), $x_{c,i}$, the distance from the $i$-th cluster of negative data set to the positive data set is estimated by mean cluster distance in the feature space, which is given by

\[
\delta_F(x_+, x_{c,i}) = d(\bar{\phi}(x_+), \bar{\phi}(x_{c,i})) = \left( \frac{\sum_{x_{+i} \in X_+} K(x_{+i}, x_{+j})}{l_+^2} \right)^{1/2} + \left( \frac{\sum_{x_{-p} \in X_{c,i}} K(x_{-p}, x_{-q})}{l_{c,i}^2} \right)^{1/2} - \frac{2}{l_+ l_{c,i}} \sum_{x_{+i} \in X_+ \atop x_{-m} \in X_{c,i}} K(x_{+i}, x_{+j}),
\]

where $\bar{\phi}(x_{c,i})$ is the mean of $x_{c,i}$ in the feature space, and $l_{c,i}$ is the number of examples in cluster. Due to the large number of clusters, positions of some clusters in negative set are distant from that of the positive set in the feature space. The examples in these clusters either are redundant data or distort the hyper plane of the OISVM.
Those distant clusters should be eliminated from the negative data set. Thus, a new parameter is defined as:

$$a_d = \frac{\delta_f(x_+, x_{c,i})}{\delta_f(x_+, x_{\text{nearest}})} > 1,$$

where $x_{\text{nearest}}$ is estimated by (11). $a_d$ is used to determine which cluster is distant. If $\delta_f(x_+, x_{c,i}) > a_d \delta_f(x_+, x_{\text{nearest}})$, the $i$th cluster will be deleted from negative data set. In practice, $a_d \in [1.5, 4]$ works well. Referring to Fig. 2.2, class 1 is the positive class and the rest belongs to the negative set. As the inter-cluster distance of class 4 and class 5 are greater than $a_d \in [1.5, 4]$, they are deleted from the negative set. Class 2 and 3 are kept as the negative class.

**Distance Based Random Under-sampling** When all the distant clusters are removed, remaining clusters form a new negative data set. In many cases, the total negative clusters still overwhelm the positive one if a small $a_d$ is picked or if there are too many samples in near clusters. Random under-sampling is performed in this circumstance. However, completely randomly deleting samples may cause the loss of valuable SVs. The proposed kernel distance based random under-sampling reduces this possibility. Given a sampling coefficient $K$, $l_-$ examples are selected from the new negative data set where $l_- = K \times l_+$. The number of samples to keep in each cluster $P$ is determined by the cluster mean distance $\delta_F$ calculated in the gamma selection phase. Suppose there are clusters left after performing KCS. Distance between each negative cluster with positive cluster is ranked in set $D = \{D_1, D_2, \ldots, D_m\}$. The weight of each cluster is given by $P_i = (1/D_i)/(\sum_{i=1}^{m} 1/D_i)$. The inverse of distance is chosen because the closer the cluster is, the more weight it should have. In some cases $P_i$ is larger than the cluster total counts, then $S_i$ is introduced to shift the overflowed number to the next rank cluster.
\[ P_i = \min \left( \frac{1}{D_i} \sum_{n=1}^{m} \frac{1}{D_n} * l_+ + S_i, l_i \right) \]

where \( S_{i+1} = \max \left( \frac{1}{D_i} \sum_{n=1}^{m} \frac{1}{D_n} * l_+ + S_i - l_i, 0 \right), S_1 = 0 \) \hfill (13)

After getting the final number of each cluster, random under-sampling is performed to delete unwanted samples.

**Tomek-Link Deleting**  After distance based random under-sampling, generalized Tomek-links are eliminated from the selected examples. Generalized Tomek-link in the feature space is defined as follows.

Given that \( \mathbf{E}_i \) and \( \mathbf{E}_j \) belong to positive class and selected examples of negative class respectively, \( d_F(\mathbf{E}_i, \mathbf{E}_j) \) is the distance between \( \mathbf{E}_i \) and \( \mathbf{E}_j \) in the feature space calculated by (7). A pair of \((\mathbf{E}_i, \mathbf{E}_j)\) is called a generalized Tomek-link if an example does not exist, such that \( d_F(\mathbf{E}_i, \mathbf{E}_m) < d_F(\mathbf{E}_i, \mathbf{E}_j) \) or \( d_F(\mathbf{E}_j, \mathbf{E}_l) < d_F(\mathbf{E}_i, \mathbf{E}_j) \). Existence of Tomek-links degrades the classification accuracy. The proposed method detects all the Tomek-Links and deletes them from the samples.

**Prediction Phase Complexity**  As stated above, one of the main motivations of under-sampling for mobile clients is to reduce the prediction complexity. For SVM multi-class classifier with RBF kernel, the prediction complexity is \( O(N_{sv}d) \), where \( N_{sv} \) is the total number of SVs in all classifiers, \( d \) is the input dimension, i.e., number of APs. \( N_{sv} \) is the product of amount of SVs in one classifier \( n_{sv} \) and number of classifiers \( r \). \( n_{sv} \) is highly dependent on the specific application as well as the selection of parameter \( C \). Based on our observation in indoor positioning dataset, \( N_{sv} \) of proposed method is from 1/3 to 1/10 of \( N_{sv} \) of normal one-versus-one methodology. As a lazy training algorithm, KNN suffers from large number of radio map data in
prediction phase. It needs to go through all the training samples and calculate the distance with each of the samples and calculate the distance with each of them and then rank. The complexity of KNN is $O(dl^2)$.

Note that $l$ is the number of total samples which is much larger than the total SVs. Obviously, the time complexity of KNN is higher.

Fig. 2.3: The layout of the measurement area.

2.4 Experimental Evaluation

The experiments were conducted in two different stages. In the first stage, because there is not a well-recognized indoor positioning benchmark in the field, the algorithm was applied to different multi-class benchmarks to demonstrate its generalization ability. Training time, number of models and number of SVs are the major performance measurements. The amount of SVs is crucial because it determines the on-line prediction speed, on-line learning speed and also the memory cost. In the second stage, measured RSS data was applied to this method. The RSS data was collected on the
second floor in the building of the Centre for Engineering Innovation at the University of Windsor. The layout of this area is shown in Fig. 2.3. The test area has dimensions of 23 m by 20 m. The total test bench environment consists of three hallways and two rooms. All the experiments were conducted with the same workstation. Matlab R2012 was the simulation platform. Android cellphone Galaxy 3 and tablet Nexus 9 were chosen for data collection.

Table 2.1: Specification of the Benchmarks

<table>
<thead>
<tr>
<th>Data sets</th>
<th># of Classes</th>
<th># of Features</th>
<th># of Training Data</th>
<th># of Testing Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNA</td>
<td>3</td>
<td>180</td>
<td>1400</td>
<td>1186</td>
</tr>
<tr>
<td>Satimage</td>
<td>6</td>
<td>36</td>
<td>4435</td>
<td>2000</td>
</tr>
<tr>
<td>Letter</td>
<td>26</td>
<td>16</td>
<td>10500</td>
<td>5000</td>
</tr>
</tbody>
</table>

Table 2.2: Simulation Results for Benchmarks

<table>
<thead>
<tr>
<th>Data sets</th>
<th>Algorithms</th>
<th>Accuracy</th>
<th># of SVs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNA</td>
<td>Grid Search+LIBSVM</td>
<td>94.6%</td>
<td>829</td>
</tr>
<tr>
<td></td>
<td>GA+LIBSVM</td>
<td>94.4%</td>
<td>1020</td>
</tr>
<tr>
<td></td>
<td>PSO+LIBSVM</td>
<td>94.3%</td>
<td>743</td>
</tr>
<tr>
<td></td>
<td>KNN</td>
<td>84%</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Bayes</td>
<td>93.9%</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Proposed Method</td>
<td>94%</td>
<td>212</td>
</tr>
<tr>
<td>Satimage</td>
<td>Grid Search+LIBSVM</td>
<td>91.7%</td>
<td>1642</td>
</tr>
<tr>
<td></td>
<td>GA+LIBSVM</td>
<td>91.6%</td>
<td>1620</td>
</tr>
<tr>
<td></td>
<td>PSO+LIBSVM</td>
<td>91.6%</td>
<td>1666</td>
</tr>
<tr>
<td></td>
<td>KNN</td>
<td>88%</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Bayes</td>
<td>79.6%</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Proposed Method</td>
<td>91.7%</td>
<td>575</td>
</tr>
<tr>
<td>Letter</td>
<td>Grid Search+LIBSVM</td>
<td>97%</td>
<td>6014</td>
</tr>
<tr>
<td></td>
<td>GA+LIBSVM</td>
<td>96.9%</td>
<td>6017</td>
</tr>
<tr>
<td></td>
<td>PSO+LIBSVM</td>
<td>96.1%</td>
<td>5738</td>
</tr>
<tr>
<td></td>
<td>KNN</td>
<td>94.5%</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Bayes</td>
<td>62%</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>Proposed Method</td>
<td>96.7%</td>
<td>870</td>
</tr>
</tbody>
</table>
2.4.1 Simulation Results with Benchmarks

The proposed algorithm was applied to three multi-class test benches, namely Satimage, Letter and DNA obtained from UCI machine learning repository. Table 2.1 lists the details of these benchmarks. The benchmarks were chosen due to their similarity with indoor positioning data. Two of the datasets were scaled into [-1, 1] for training. DNA was not scaled because the original data was located in this range. Then the proposed method was compared to the grid search, GA and PSO combined with LIBSVM, KNN and Bayes algorithm. Grid search is a widely used optimization method for SVM whereas GA and PSO are two popular intelligent methods for searching SVM parameters. The deterministic algorithm KNN developed in Radar system, the probabilistic Bayes algorithm developed in Horus system have been implemented for comparison. RBF kernel was used in this experiment. Both one-versus-one and one-versus-all methods for multi-class SVM were implemented as reference.

In the training phase, the main optimization targets were the penalty parameter $C$ and RBF kernel parameter $\gamma$. In the parameter optimization phase, the data was separated into training and validation data by applying 3-fold cross validation since the testing set should not be included for parameter selection.

Table 2.2 summarizes the comparison of all other resources including accuracy and number of SVs among different methods. It shows that the accuracy of the proposed method is comparable with other SVM algorithms and increases by up to 30% compared to KNN and Bayes. Number of SVs show 3 to 10 times reduction over other optimization methods. These results show that proposed method and other SVMs have better classification performance than general methods. Meanwhile, proposed method achieves the results with significant less number of SVs.

The training time of the proposed method includes the time required for distance calculation and the original OISVM training time. The training time of other methods usually consist of parameter selection algorithm and the corresponding SVM training.

26
time. A portion of Fig. 2.4 shows the training time required for Grid Search+ LIBSVM, GA+LIBSVM, PSO+LIBSVM and the proposed method in the benchmark. As KNN and Bayes algorithms are not a major concern in terms of training time, they were not considered for this comparison. The simulation time of proposed technique is at least 14 times faster than other algorithms in DNA dataset, and at least 20 times faster than others in Satimage and Letter datasets. The proposed method is faster in the Satimage and Letter since the distance calculation for all classes is comparable to the training time for a small training data.

Fig. 2.4: Training time of DNA, Satimage, Letter and IP1 1 of Indoor Positioning data for Grid Search, GA, PSO and Proposed Method.

Table 2.3: Specification of Measured Indoor Positioning Data Sets

<table>
<thead>
<tr>
<th>Data sets</th>
<th># of RPs</th>
<th># of Training Data</th>
<th>RP Intervals(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP1</td>
<td>112</td>
<td>3356</td>
<td>1</td>
</tr>
<tr>
<td>IP2</td>
<td>25</td>
<td>2500</td>
<td>1</td>
</tr>
<tr>
<td>IP3</td>
<td>50</td>
<td>1500</td>
<td>0.5</td>
</tr>
</tbody>
</table>
2.4.2 Indoor Positioning Simulation and Implementation

Different datasets were collected in the test bench environment to perform different simulations and verify proposed method. Three datasets were measured. Both the training and testing phase were performed at the cloud server. Popular localization algorithms such as Radar and Horus were adopted for comparison.

For all the positioning data sets, the total number of available APs in the measurement area was 13. The details of indoor positioning data sets are described in Table 2.3.

The training time and accuracy comparison were performed with IP1. As shown in the last column of Fig. 2.4, the training time is at least 16 times faster than other optimization algorithms. The accuracy evaluation was accomplished against other popular systems. Radar [13], Horus [9] and LIBSVM [34] were selected for comparison. In proposed algorithm, the tolerance factor, $\eta$ in all OISVM classifiers is set to be 0.1. Fig. 2.5 presents the accuracy with respect to the error distance. The error cumulative distribution function (CDF) of proposed method with batch learning outperforms the Radar and Horus. With the on-line learning, the accuracy shows substantial improvement over other methods. The result shows about 15% increase in CDF.

Some other performance measurement from the simulation results with IP1 have been summarized in Table 2.4. The number of SVs of proposed method with batch learning shows about 2000 reduction compared to other SVM algorithms. As stated above, less number of SVs results in a faster prediction speed. A reduction of 0.8m can be seen in terms of error distance when on-line learning was implemented.
Fig. 2.5: Positioning accuracy of the proposed solution versus probabilistic and KNN methods.

Table 2.4: Simulation Results for Indoor Positioning Data Sets

<table>
<thead>
<tr>
<th>Data sets</th>
<th>Algorithms</th>
<th>Mean Error Distance (m)</th>
<th># of SVs</th>
<th>Model Size(MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP1</td>
<td>Proposed Method with On-line learning</td>
<td>1.2</td>
<td>1455</td>
<td>10</td>
</tr>
<tr>
<td>IP1</td>
<td>Proposed Method with Batch learning</td>
<td>2.1</td>
<td>1203</td>
<td>8</td>
</tr>
<tr>
<td>IP1</td>
<td>Original OISVM</td>
<td>2.0</td>
<td>3025</td>
<td>53</td>
</tr>
<tr>
<td>IP1</td>
<td>LIBSVM</td>
<td>2.0</td>
<td>3177</td>
<td>56</td>
</tr>
</tbody>
</table>

It is clear from Fig. 2.5 and Table 2.4 that the proposed method with on-line learning leads to substantial improvements. The improvements depend on the testing environment. In this experiment, a reduction of 0.8m mean error distance can be seen.

In terms of SVM model size, the key impact factor is the number of SVs. In
practice, the kernel matrix can be saved as a triangular matrix with dimension of the number of SVs since it is symmetrical. Note that in original LIBSVM, the kernel matrix is not saved directly in the model. For comparison simplicity, it is extracted the same way as original OISVM. It is shown in Table 2.4 that the model sizes of other SVM algorithms are at least 5 times larger than the proposed method with batch learning and on-line learning.

Fig. 2.6: Error distance under different number of training samples during on-line phase learning of proposed method.

Fig. 2.6 gives the relation of error distance and number of training samples. The error distance drops from 2.2m to 1.2m when the on-line learning ability is turned on within a few days. For this experiment, dataset IP2 were collected sequentially. Among the training samples, 750 samples were collected at the first day. Each day another 250 samples were measured and trained to update the model. With the addition of new samples each time, the simulation only needs to train the new samples, not the whole dataset. The model was updated every day by on-line learning. As mentioned in section 4, this ability can be a complement of many other positioning systems that use the crowdsourcing for training.
IP3 was selected to evaluate the effect of Tomek-link deleting. According to the experiment observations, more overlapping models usually imply more Tomek-links. Data collected in every 0.5m has a higher existence rate of Tomek-links. Elimination of more of these samples results in an increase in accuracy. A visible improvement can be seen from Fig. 2.7. Around 10% increase in accuracy was observed at 3m of error distance.
Fig. 2.8 shows the comparison of testing time among original OISVM, LIBSVM, KNN and proposed method. The proposed method requires less testing time than the other methods. This feature makes it suitable for mobile applications.

2.5 Conclusion

A fast OISVM classification technique empowered by a novel under-sampling method is developed in this work to provide an efficient indoor positioning solution. To reduce the computational complexity compared to traditional SVMs, borderline samples were removed and the kernel parameter was optimized. The model size is reduced significantly by proposed under-sampling algorithm, which in turn lowers the required computational resources. This allows the implementation of the proposed indoor positioning solution on mobile devices where the resources are limited.

Multiple experiments have been performed. Experimental measurement results indicate that the proposed solution with on-line learning ability reduces the error distance by 0.8m while lowering the prediction time by more than 5 fold as compared to existing methods. It also reduces the time of training phase and testing phase by
10 to 50 times as compared to traditional SVMs.

2.6 References


techniques and systems,” Systems, Man, and Cybernetics, Part C: Applications

based motion recognition for smartphone indoor wireless positioning,” Sensors,


support vector classification,” in Networking, sensing and control, 2004 IEEE

dent support vector machines,” Pattern Recognition, vol. 43, no. 4, pp. 1402–
1412, 2010.

[21] H. He and E. A. Garcia, “Learning from imbalanced data,” Knowledge and Data

in class imbalance problems,” Pattern Recognition, vol. 36, no. 3, pp. 849–851,
2003.


3 Improved Particle Filter Based on WLAN RSSI Fingerprinting and Inertial Sensors for Indoor Localization

The demand for accurate positioning for indoor location based services (LBS) is growing rapidly. The GPS technology cannot be easily used for indoor positioning as the direct line of sight obstructed which reduces the positioning accuracy [1] [2]. To overcome the challenges of indoor positioning, many techniques have been developed in the past few years. The reported indoor positioning methods fall into two main approaches. In the first approach, wireless signals such as Bluetooth, Zigbee, RFID, Ultra-Wide Band (UWB), WLAN (IEEE 802.11) [3] are leveraged for indoor positioning. In the second approach inertial sensors are utilized [4].

Two methods are developed for positioning algorithms utilizing wireless signal techniques. In the first method, the distance is estimated from Time of Flight (ToF), Time of Arrival (ToA)/Time Difference of Arrival (TDoA), Angle of Arrival (AoA) or Received signal Strength Indicator (RSSI) [3]. RSSI for ranging is highly dependent on the environment structure and has limited accuracy [5]. Time measurement based method supports high positioning accuracy but it commonly requires extra infrastructure to accurately measure the time difference. This requirement increases the cost of implementation. An infra-structure free solution which utilizes available wireless local area network (WLAN) have also been reported [6]. In this method, it is assumed that each reference point has a unique RF signal strength vector, which is also called fingerprint. The fingerprints of reference points in a building are collected ahead of time and stored in a database. Then, pattern recognition algorithms are used to match the on-line vector with pre-collected fingerprints. This approach is infrastructure-free but labor-intensive [7]. Moreover, this system normally cannot provide a smooth location estimation because it suffers from problems such as similar...
fingerprints [8], missing value and noisy RSSI value.

In the sensor based indoor positioning method, the key task is to estimate the travel distance and angle of humans or objects [4]. As the inertial measurement unit (IMU) which comprise accelerometer, gyroscope, compass and barometer is widely integrated in portable devices, many researchers have dedicated their work to develop new IMU based solutions for indoor positioning. The common algorithm of IMU assisted positioning is Pedestrian Dead Reckoning (PDR). In this method, a step sensor which is implemented by accelerometer, is used to detect the displacement of a user. Meanwhile, the gyroscope and/or compass are used to detect the orientation [9]. The cost of this method is very low and a smooth location estimation is achieved. However, as the system cannot calibrate itself, it suffers from cumulative error problem.

To tackle this problem, algorithms that combine Wi-Fi fingerprinting and IMU assisted positioning have been explored by researchers. Inherited from robotics community, standard Kalman Filter (KF), Particle Filter (PF) and their variants are introduced to fuse the sensor information [10]. In general, KF is applicable to linear and Gaussian models. For complex noisy environments, PF is widely chosen for its superiority in handling nonlinear system and non-Gaussian noise [11].

Although, the results after combining fingerprinting and IMU are smooth and self-calibrated, the accuracy is still restricted since fingerprinting algorithm requires the assumption of unique fingerprints [8]. In practice, due to the multi-path effect and the arrangement of the location of access points (APs), two distant reference points may share very similar fingerprints. As a result, the pattern recognition algorithms cannot guarantee the correct estimation. An inaccurate position estimation can deteriorate the overall performance. Meanwhile, the time required to initialize PF is also an important factor. Global initialization has a slow convergence speed. Deploying extra hardware at entrances increases the total cost.

With the proposed method, it is not required to have extra hardware like RFID to
initialize the PF. It selects the inliers from fingerprinting estimations by a Gaussian model established by PDR data. To overcome the problem of wrong estimations after initialization, the weighting portion of the conventional PF is improved in the proposed method to model multiple fingerprinting probabilities by single-hidden layer feedforward networks (SLFNs) interpolation [12].

There are two major contributions in this paper. First, RANSAC-based approach [13] is performed to get rid of the inaccurate estimations from Wi-Fi fingerprinting during initialization phase. Experimental results show the proposed method reduces required initialization iterations by 8.1 and reduces 1.5 (m) error distance. Second, the probabilities of different reference points from Wi-Fi fingerprinting algorithm are considered to minimize the error introduced by similar fingerprints problem. SLFNs is performed to interpolate the probability of multiple results and then the PF weighting based on the interpolated model is started. Proposed method show 1 (m) error distance reduction compared to convention method in the experiments.

The rest of the paper is organized as follows. A description of the related works and motivations is provided in the next section. Section III introduces the preliminaries of this paper. Section IV presents the RANSAC-based initialization approach and a novel particle filter weighting scheme by SLFNs interpolation. Simulation and experimental results are demonstrated in section V. Section VI concludes the paper.

3.1 Background

3.1.1 Related Work

For infrastructure-free indoor localization, fingerprinting-based method is very popular and well-studied. In fingerprinting-based methods, deterministic approaches and probabilistic approaches are two major approaches that utilize the pre-collected RSSI fingerprints for location estimation. Deterministic approaches mainly apply the concept of classification or regression from pattern recognition. Bahl et al. proposed
RADAR system [6] which was based on the K-Nearest-Neighbor (KNN) and reported acceptable accuracy. Support Vector Machines [14] were also implemented due to its superb classification and regression ability for non-linear problems. The main limitation of fingerprinting methods is the sensitivity to the RSSI variation caused by multi-path effect and large-scale fading effect [15]. Probabilistic approaches improve the stability against RSSI variation by modeling the RSSI distribution of certain APs. Youssef et al. proposed Horus system that implemented such approach that reported a higher accuracy and stability. However, very accurate RSSI distribution for each AP is not practical to achieve and a biased distribution can degrade the accuracy.

There are a few works dedicated to the fusion of fingerprinting methods and PDR algorithm. Leppäkoski et al. suggested Extended Kalman Filter and PF for the fusion and got improved performance, yet the system requires a known initial point [10]. An upgraded PF with a fallback filter for particles initialization in which the filter requires a heavy global state space search is proposed in [16]. The method developed in [17] takes processed Wi-Fi RSSI for azimuth estimation and PF initializes uniformly in particle space. None of these works specifies particular mechanism to handle the occasional poor observation from fingerprinting.

![Fig. 3.1: Scenario of similar fingerprints. Blue region and green region in each graph show similar fingerprints due to the deployment of APs and the building structure.](image)
3.2 Motivation

The motivation of this paper is three-fold. Firstly, preferred method should make sure that the PF does not miss the optimal result. Traditional fingerprinting methods only deliver one estimation in terms of RSSI fingerprints. Because of the noisy indoor environment, missing value and the similarity among the fingerprints, it might be distant from the correct estimation so that particles follow the wrong estimation. As shown in Fig.3.1, there is no means that fingerprinting methods can differentiate the blue region and green region. When the system produces the wrong estimation, the accuracy of predicted trajectory from PF degrades. Second motivation is how to weight the particles based on the solution from fingerprinting methods. The weights for particles shall be smooth and continuous so that the movement of particles can be smooth in accordance with human activity. Thirdly, the method shall offer accurate initial guess for PF without any aid from extra-hardware. The accurate initial guess enables the system to acquire a faster convergence rate and a more accurate location estimation.

3.3 Preliminaries

3.3.1 Indoor Localization Problem Setting

Particle filter based indoor localization problem is to find the joint posterior

\[ p(x_{1:t}|z_{1:t}, u_{1:t-1}, m) \]

about the trajectory \( x_{1:t} \) of user in the indoor environment. In this problem, the observations \( z_{1:t} = z_1, ..., z_t \) and the motion odometry measurements \( u_{1:t} = u_1, ..., u_{t-1} \) are obtained by different sensors. Map \( m \) is usually known to the system. The inertial measurements are usually obtained by the IMU module. PDR is a common methodology implemented for human navigation. There are multiple methods to get the observations including laser range finder, infrared, Wi-Fi fingerprinting etc. In this paper, we mainly focus on the solution by Wi-Fi fingerprinting.
### 3.3.2 Review of Particle Filter with Fingerprinting Algorithm

**Sampling Importance Resampling Particle Filter**  In particle filter, each particle is a pose hypothesis of the current state. Proposed by Gordon *et al.* [18], Sampling importance resampling (SIR) particle filter is widely used because it keeps the diversity of particles. A SIR filter processes the sensor observation and motion odometry readings iteratively when the data is available. In every iteration, it updates and resamples all the particles which represents the posterior of the trajectory. Each iteration can be demonstrated by the following steps.

- **Sampling:** Given the previous trajectory $x_{1:t-1}^{(i)}$, the particles of next time slot are obtained by drawing the samples from the proposal distribution $\pi(x_t | x_{1:t-1}^{(i)}, z_{1:t}, m)$.

- **Importance weighting:** All the particles are weighted by $w_t^{(i)}$, calculated from

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{p(z_t | x_t^{(i)}) p(x_t^{(i)} | x_{t-1}^{(i)}, u_{1:t-1})}{\pi(x_t^{(i)} | x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t-1})}.$$  \hspace{1cm} (14)

Then the weights are normalized. Weighting has a major effect on the final performance. Incorporated with Wi-Fi fingerprinting methods, normally the likelihood of observation $p(z_t | x_t^{(i)})$ is a Gaussian distribution of the observed location. Unlike the observation made by sensors from robotics, this observation is not so accurate and stable. Special steps shall be done to handle the unreliable observation.

- **Resampling:** Resample $i$ particles from existing particle set with proportional to their weights. It is used to avoid the particle degeneracy. It is necessary to keep an adequate number of particles to approximate the actual prior distribution.

The role of PF in this system is to combine the PDR estimation with fingerprinting estimation. Whereas it remains open about how to decide the likelihood of observation from fingerprinting estimation.
Wi-Fi RSSI Fingerprinting Methods  Wi-Fi RSSI fingerprinting methods mainly deliver the observation \( z_{1:t} = z_1, ..., z_t \) to the system. The output can be a grid or a set of coordinates, which corresponds to classification or regression. As for classification, collect \( K \) samples with labels from \( n \) reference points (RP) as the training data. The labels are denoted as \([y_1, y_2, ..., y_n]\). Suppose there are \( l \) Wi-Fi APs around the environment, then define each sample as \( x_k = \{SSI_{k,1}, SSI_{k,2}, ..., SSI_{k,l}\} \in \mathbb{R}^l \). All the samples are used as RSSI fingerprints to construct the fingerprints database. The positioning problem is to find the label \( y_r \), given the RSSI fingerprint \( x_r \). The interpretation of regression is similar to the classification except that the regression requires the numeric solution, which are the user’s coordinates on the map.

3.4 Particle Filter with RANSAC-based Initialization and Improved Weighting Scheme

![Diagram](image-url)

Fig. 3.2: A new particle filter scheme by improved initialization phase and improved weighting process
As depicted in Fig. 3.2, the inputs of the system include Wi-Fi RSSI scans and IMU readings. Two major improvements are applied to traditional particle filter. Firstly, a RANSAC-based initialization phase is introduced to the system. This method requires several scans from Wi-Fi module and a trajectory model made by PDR algorithm. It filters out all the outliers by the trajectory and keeps the inliers for initialization, which increase the convergence rate of the PF as well as the accuracy. After a normal sampling phase, an improved importance weighting method is introduced. This phase initially collects multiple fingerprinting estimations with their probability. Then perform model fitting algorithm to construct a Gaussian mixture model. Each particle obtains a weight from the constructed model. Finally, resample the particles based on the weights and calculate the average location from the new particles.

3.4.1 RANSAC-based Initialization

Algorithm 3.1 RANSAC-based initialization algorithm

| INPUT: | Data of fingerprinting estimations: $\Psi$ |
| INPUT: | Model from PDR: $M_{PDR}$ |
| INPUT: | Maximum iterations: $N$ |
| INPUT: | Minimum data points to fit the model: $min$ |
| INPUT: | Model tolerance factor: $\epsilon$ |
| INPUT: | Error tolerance factor: $E_{total}$ |
| OUTPUT: | Set of inliers $S$ and fitted model $M_{PDR}(\hat{x}, \hat{y})$ |

1: while iterations $< N$ do
2: Randomly select $S^{(u)}$ from $\Psi$, $u = min$
3: Fit $d_i$ to model $M_{PDR}(\hat{x}, \hat{y})$
4: for $d \in \Psi - S^{(u)}$ do
5: Calculate $Error(d; M_{PDR}(\hat{x}, \hat{y}))$
6: if $Error(d; M_{PDR}(\hat{x}, \hat{y})) < \epsilon$ then
7: $u++$
8: end if
9: end for
10: if $u > E_{total}$ then
11: return $M_{PDR}(\hat{x}, \hat{y}), S^{(u)}$
12: end if
13: end while
Using RSSI fingerprinting to initialize the particle filter, there is no requirement for extra hardware. A faster convergence speed is available compared to global initialization. Normally, multiple scans are required for obtaining stable estimations. Whereas because of the noisy indoor environment and the similarity among fingerprints, outliers are always observed during the initialization phase. There is a chance that the PF initializes in a completely wrong area and therefore produces wrong results.

To tackle this issue, we introduced Random Sample Consensus (RANSAC) with PDR trajectory. As an iterative algorithm, RANSAC is widely used in computer vision and regression problems [19]. It harnesses regression techniques to generate models. It retains the inliers that can be fit into the model while filter out the outliers. Unlike pure regression techniques, it has the ability to deal with contaminated dataset. It comprises two phases: model generation and model evaluation. It randomly picks up a subset of data for multiple times then model each of them. The models are generated based on the prior of the application. Then the model is evaluated and finally keeps the one with the most inliers. RANSAC considers the evaluation problem as an optimization problem formulated as

$$\hat{M}_{\text{PDR}}(\hat{x}, \hat{y}) = \arg\min_{M_{\text{PDR}}(\hat{x}, \hat{y})} \sum_{d \in \Psi} \text{Loss}(\text{Error}(d; M_{\text{PDR}}(\hat{x}, \hat{y}))),$$

(15)

where $\Psi$ is the data of fingerprinting estimations, $M_{\text{PDR}}(\hat{x}, \hat{y})$ is the generated model with parameter $(\hat{x}, \hat{y})$, Loss and Error are the loss function and error function respectively. In RANSAC, the Loss is defined as:

$$\text{Loss}(\text{Error}) = \begin{cases} 0 & |\text{Error}| < \epsilon \\ 1 & \text{otherwise} \end{cases}. \quad (16)$$

46
Pseudo-code for RANSAC-based initialization is detailed in Algorithm 3.1. The input of this algorithm is a set of fingerprinting estimations $\Psi$. The model is constructed by PDR trajectory $M_{PDR}$ and a few experienced parameters. Step 2 and 3 are the model generation phase. The algorithm randomly selects $\min$ data points to the set of inliers $S^{(u)}$ from dataset $\Psi$. With the given model, it runs model fitting algorithm for data points $S^{(u)}$. Note that PDR trajectory is a perfect model to be used since the error of IMU is quite small in a short time interval. The parameter of the model is $\hat{x}, \hat{y}$ which denotes the initial point of user. Then the model evaluation phase starts. The error distance of all the data points from set $\Psi - S^{(u)}$ are calculated and compared with model tolerance factor $\epsilon$. Those with error distance smaller than $\epsilon$ are added to $S$. Finally, if $u$ is greater than $k$, algorithm returns. Otherwise repeat all the steps until reach the maximum iterations $N$.

The implementation of RANSAC is based on two assumptions: outliers from the samples are minority and a model is available to fit the inliers. Both of these assumptions are satisfied in indoor positioning scenario. Firstly, for fingerprinting algorithms, the reported average error distance is in meter range, generally about 1-5m [20]. According to the error cumulative distribution function (CDF) of classic fingerprinting algorithms [6] [21], it is observed that the error distance of over 90% of the estimations are less than 4-5m, which means that majority of the estimations are inliers. These reported results meet the experimental results of this paper. Secondly, a natural model is given by the trajectory from PDR algorithm. The variables are $(\hat{x}, \hat{y})$ that determines the initial point of the trajectory.

To construct the PDR model, we first perform PDR for $c$ iterations. For each iteration, the relative position of user can be computed by

$$ (x_{i+1}, y_{i+1}) = (x_i + L_i \sin(\phi_i), y_i + L_i \cos(\phi_i)) $$

(17)
where $x_i$ and $y_i$ are the coordinates, $L_i$ and $\phi_i$ are the stride length and heading after $i$th step. Gaussian least squares fitting are leveraged to fit the trajectory to a Gaussian model $G(x)$. Note that $G(x)$ has certain domain $x \in [x_{\min}, x_{\max}]$. $G(x)$ is described by

$$G(x) = \sum_{i=1}^{\hat{n}} a_i e^{-(x-b_i)^2}, x \in [x_{\min}, x_{\max}]$$ (18)

where $\hat{n}$ is the number of terms, $a_i, b_i, c_i$ are the coefficients of Gaussian model and $x_{\min}, x_{\max}$ are the minimum and maximum values of the PDR trajectory. To shift the function to arbitrary position on the map, two coefficients of $(x', y')$ are introduced to estimate the initial point. As a result, (18) is transformed to

$$M_{PDR}(x', y'; x, y) = \sum_{i=1}^{\hat{n}} a_i e^{-(x-x'i-b_i)^2} + y',$$ (19)

where $x \in [x_{\min}, x_{\max}]$. In order to find $(\hat{x}, \hat{y})$, the algorithm requires to estimate the initial point by Wi-Fi fingerprinting during initialization period which minimize the estimation errors. Applying the selected min number of fingerprinting estimation $S$ from $\Psi$, $(\hat{x}, \hat{y})$ becomes

$$\arg\min_{(x,y)} \sum_{d \in S^{(o)}} Distance(d; M_{PDR}(x, y))^2,$$ (20)

where $Distance$ is the distance between the instance and the model. As $M_{PDR}(x, y)$ is a curved line segment, the distance from point $A(x_A, y_A)$ to a curve is calculated by

$$D(x_A, y_A) = \sqrt{(x - x_A)^2 + (y - y_A)^2},$$ (21)

Let deviation of (21) equals to 0, then get the closest point $o$ on the curve $(x_o, y_o)$.
Finally calculate Distance by

\[
Distance(d; M_{PDR}(x, y)) =
\begin{cases}
\sqrt{(x_o - x_d)^2 + (y_o - y_d)^2} & x_o \in [x_{min}, x_{max}], \\
C & \text{otherwise}
\end{cases}
\]  

(22)

where \( C \) is the minimum value of the distances from \( d \) to the two endpoints.

### 3.4.2 Modeling Fingerprinting Estimation by SLFNs interpolation

Our approach requires fingerprinting methods to output multiple estimations with their possibilities, denoted as \( P = \{P(y_1|x_k), P(y_2|x_k), ..., P(y_n|x_k)\} \). Different pattern recognition algorithms perform different methodologies to achieve this purpose. The most straightforward methodology is the Bayesian scheme. Normally, this methodology is to find the label \( y \) from \( [y_1, y_2, ..., y_n] \) that maximize the probability \( P(y|x_k) \).

Using Bayesian theory, it is equivalent to:

\[
\arg \max_y P(y|x_k) = \arg \max_y P(x_k|y),
\]  

(23)

where \( P(x_k|y) \) is calculated by \( \prod_{i=1}^{n} P(x_k|y_i) \). In traditional approach, the estimation is given by \( y_i \) with the highest probability. In our approach, we leverage the whole set of \( P(x_k|y_i) \) to keep the entire information of fingerprinting method such that the particles are able to track the real location. In order to calculate \( P(x_k|y_i) \), one can use probabilistic methods [21] or deterministic methods such as Support Vector Machines (SVMs) [22]. The result comparison of these methods are given in the experimental section.

For probabilistic methods, the Gaussian distribution can be used to approximate the distribution of RSSI of one AP at a certain location. \( P(x_k|y_i) \) is calculated by
\[ \prod_{m=1}^{l} f_m(SSI_m; \mu_m, \sigma_m), \]

where \( f_m \) is Gaussian distribution of AP \( m \) with mean \( \mu_m \) and variance \( \sigma_m^2 \). Mean and variance values are determined by training data. Finally, probability \( P(x_k|y_i) \) is normalized.

For SVMs, a pairwise coupling method [23] from LIBSVM [24] is widely used for probability estimation. As a two-class classifier, SVMs requires pairwise coupling to extend the two-class probability scheme \( a_{ij} = P(y = i|y = i \ or \ j, x_k) \) to multi-class probability \( P(x_k|y_i) \). Essentially, it is achieved by solving the optimization problem given by

\[
\min_{P} \sum_{i=1}^{n} \sum_{j:j \neq i} (a_{ji}P(x_k|y_i) - a_{ij}P(x_k|y_j))^2,
\]

subject to \( \sum_{i=1}^{n} P(x_k|y_i) = 1, P_i \geq 0, \forall i \quad \text{(24)} \)

where \( a_{ij} + a_{ji} = 1, \forall i \neq j \). \( P(x_k|y_i) \) is obtained by solving (24).

\( P(x_k|y) \) are the probabilities for discrete grid maps. Whereas the particles require a continuous weighting model. A interpolation method that fit the discrete probabilities to a continuous model is required. We select SLFNs based interpolation to obtain the model for two reasons. First reason is that arbitrary target function is required since \( P(x_k|y) \) does not follow any certain distribution. Second, the error should be extremely small. Based on the literature [25], the SLFNs are able to approximate any target distribution with arbitrary small error. It has been proved in [26] that SLFNs can interpolate samples with negligible error.

The mathematical expression of SLFNs with \( N \) hidden nodes and activation function \( f(x) \) on this interpolation problem is given as

\[
P(x_k|y_i) = \sum_{i=1}^{N} c_i f(w_i \cdot x(k) + \theta_i),
\]

where \( w_i \in \mathbb{R}^2 \) and \( c_i \in \mathbb{R} \) are the input weight vector and output weight vector that
connect the $i$th hidden node with the input and output. $\theta_i \in \mathbb{R}$ is the threshold of the $i$th hidden node. Note that there are two input nodes for inputs $(\hat{x}, \hat{y})$ and one output node for the probability interpolation $P(x_k|y_i)$. The architecture of such SFLNs is shown in Fig.3.3.

The reason why this process can deal with similar fingerprints problem is because it evaluates all the RFs and delivers all the corresponding probabilities to the weighting process of PF. When there are similar fingerprints that are distant from each other, particles are able to differentiate them after the weighting and resampling process.

![Architecture of the SLFNs Interpolation](image)

**Fig. 3.3: Architecture of the SLFNs Interpolation**

### 3.5 Experimental Results

Simulations and experiments were conducted to verify the efficiency of the PF scheme at the second floor of Centre for Engineering Innovation of University of Windsor. PDR data and fingerprinting data were collected. The experiment was conducted at a time where people were inside the building and the RSSI samples were affected by environmental factors such as moving objects and people. Therefore, the RSSI fingerprinting based approach suffers from multi-path effect and moving objects significantly. This is similar to a practical scenario case. The PDR algorithm is imple-
mented on cell phone using the data collected by IMU. We applied these data to test the initialization phase and the estimation accuracy of the proposed PF scheme. For fingerprinting algorithm, a total of 84 reference points were selected with one meter interval in a 30 (m) by 35 (m) area. 30 fingerprints were collected at each reference point. As comparison, SVM and probabilistic algorithm are selected.

![Fitted model of RANSAC initialization](image)

**Fig. 3.4: Fitted model of RANSAC initialization**

### 3.5.1 Experiment on Initialization Phase

To test the proposed initialization approach, we collected 10 datasets under initial condition at 10 different places with different tracks. Each initial dataset includes 11 steps as well as 11 Wi-Fi scans. We first examine the Gaussian model fitting on each dataset. Fig.3.4 shows examples of the fitted model from PDR data. Number of terms is determined by RMSE. We set the threshold of RMSE to be 0.1 and select the lowest number of terms when satisfy the threshold. Then we apply the model to filter out the outliers of Wi-Fi fingerprinting estimations. As denoted in Fig.3.4, inliers are distinguished by the proposed RANSAC approach. Finally, the estimated start point
is marked on the graph. PF is initialized on the current location of the acquired model.

The simulation results of this experiment are listed in Table 3.1. RMSE of PDR model fitting is used to test the precision of the model compared to PDR trajectory. It is shown that the average RMSE is 0.21 (m). Number of outliers demonstrates the effectiveness of the RANSAC approach. In this experiment, proposed method filtered out 4.2 outliers in average. Initialization error distance and maximum error distance illustrate the initialization accuracy. As a comparative algorithm, KNN based fingerprinting initialization is selected. Proposed method reduced the average error distance by 1.6 (m) and reduced maximum error distance by 2.6 (m).

For convergence speed, proposed method only requires these 11 iterations to perform model fitting and RANSAC algorithm. As a comparison, the global initialization requires 19.1 iterations in average to converge.
3.5.2 Experiments for Testing Proposed PF scheme

In this experiment, all the algorithms use the same fingerprints database and the same PDR data. Fig.3.6 shows the pedestrian trajectory estimated by original fingerprinting method. Consecutive estimations are connected by red lines. It can be seen that the results suffer from inconsistent observations and estimations. In some cases, the continuous estimations are distant from each other due to the missing value, noisy data or the similarity of the fingerprints. Fig.3.7 demonstrates the trajectory from original PDR. In this figure, each red dot represents one step. This approach performs well during first three hallways and then accumulates large errors. Fig.3.8 shows the trajectory of proposed method, which combines the information from fingerprinting and PDR. It is shown that the algorithm fixes the noisy fingerprinting data and PDR data and improves the final performance.

Proposed PF scheme leverages SLFNs for probability distribution model construction (Fig.3.5). This surface is employed to perform particles weighting phase. Each particle is able to acquire a probability based on its location. Those particles with
higher probability are more likely to be saved after resampling phase.

Table 3.2: Error distance of different methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Average error distance</th>
<th>Maximum error distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM [14]</td>
<td>3.2 (m)</td>
<td>9.3 (m)</td>
</tr>
<tr>
<td>Probabilistic algorithm [21]</td>
<td>3.4 (m)</td>
<td>11.2 (m)</td>
</tr>
<tr>
<td>PDR with probabilistic algorithm</td>
<td>2.2 (m)</td>
<td>4.1 (m)</td>
</tr>
<tr>
<td>Proposed method with SVM</td>
<td>1.2 (m)</td>
<td>2.9 (m)</td>
</tr>
<tr>
<td>Proposed method with probabilistic algorithm</td>
<td>1.3 (m)</td>
<td>3 (m)</td>
</tr>
</tbody>
</table>

As Table 3.2 shows, the average error distances of proposed method with SVM and probabilistic algorithm are 1.2 (m) and 1.3 (m) respectively. This value is about 1 (m) lower than the value of PDR with SVM. It also shows great improvements on the maximum error distance. As given in Fig.3.6, at some points the fingerprinting algorithm provides distant estimations, which results in a high maximum error distance. This phenomenon can not be seen from the trajectory of proposed method.

### 3.6 Conclusion

In this paper, a new particle filter with a hardware-free initialization phase is presented to improve the accuracy of indoor location positioning using received signal strength. The hardware-free initialization is implemented by RANSAC algorithm. This algorithm filters out outliers from the fingerprinting estimations by a constructed PDR model. Inliers are remained to acquire the initial point and the current location. The PF is initializing based on the current location. This initialization phase achieves 1.1 (m) average error distance in the experimental demonstration. For enhancing the fusion of fingerprinting and PDR, we proposed a SFLNs based model fitting algorithm. The algorithm takes advantage of the probabilities of all the reference points from fingerprinting method. The algorithm fits a SFLNs model to the probabili-
Fig. 3.6: Trajectory of fingerprinting method

Fig. 3.7: Trajectory of PDR
ties and constructs a probability surface over the interested area. The particles are weighted by this continuous surface to reduce the error. This approach makes sure that the particles would not suffer from the similar fingerprints issue. The experimental results show about 1.2 (m) average error distance in compare to 2.2 (m) in comparative methods.

3.7 References


4 Conclusion and Future Work

4.1 Conclusion

A fast and efficient OISVM scheme integrated with a new parameter selection phase and a novel under-sampling method is proposed in this thesis. To reduce the computational complexity compared to traditional SVMs, borderline samples were removed and the kernel parameter was optimized. Training complexity and testing complexity were reduced with an improvement on accuracy. Multiple experiments have been performed. Simulation results and experimental results indicate that the proposed solution with on-line learning ability reduces the error distance by 0.8m. Meanwhile, the prediction time is lowered by more than 5 fold as compared to existing methods. The time consumption of training phase and testing phase is reduced by 10 to 50 times as compared to traditional SVMs.

To enhance the performance of fingerprinting algorithms, the thesis discussed a novel solution to fuse the IMU data with fingerprinting estimations. A new particle filter scheme with a hardware-free initialization phase and improved weighting phase to fuse the fingerprinting estimations and the PDR data is proposed. The hardware-free initialization takes advantage of RANSAC algorithm, which filters out outliers from the fingerprinting estimations by a constructed Gaussian PDR model. Inliers are remained to acquire the initial point and the current location. With the estimated start location, the PF is initialized faster than traditional global initialization. The weighting phase of proposed PF scheme make use of SFLNs interpolation, which normalizes the probability of fingerprinting estimations to a probabilistic surface. The particles are weighted by this probabilistic surface. Experimental results show proposed initialization algorithm reduces the error distance by 1.6 (m). The performance of the proposed particle filter scheme shows 1 (m) accuracy improvement and also a stabilized estimation performance. The original contribution of this
work are as follows:

- On-line independent support vector machine has been introduced in this work for indoor positioning. This method enables the system to have a higher classification accuracy and on-line learning ability.

- A $\gamma$ selection algorithm has been used in this work for reducing the training time of support vector machine. Inter-cluster distance required by this method is also utilized in the following under-sampling algorithms.

- An under-sampling scheme including Kernelized Cluster Sifting (KCS), Distance-based Under-Sampling (D-US) and Tomek-link Deleting (TLD) has been introduced. These under-sampling techniques dealt with unbalanced data problem and also reduced the prediction time and model size.

- A RANSAC based fingerprinting initialization algorithm has been proposed. With this method, the initialization accuracy is improved so that particle filter has a faster convergence speed.

- This work presents a new particle filter with a Single-hidden Layer Feedforward Networks (SLFNs) interpolation based weighting process, which handles the unique fingerprints assumption required by fingerprinting algorithms.

### 4.2 Future Work

Proposed method improves the fingerprinting method and particle filter accuracy, training time, prediction time, model size and robustness. However, to further reduce the fingerprints collection time, an accurate crowdsourcing technique is required. Also, data from inertial measurement unit requires to be processed to reduce the noise. Specifically, an accurate PDR algorithm can be used both for accurate crowdsourcing
and for accurate particle filter scheme. PDR algorithm can be improved from different aspects such as an accurate step sensor or a precise heading estimation algorithm.

Reducing the complexity of the particle filter is also an important requirement. A lightweight particle filter scheme enables the system to be deployed on portable devices easily.
Appendix A  Selected Code

%---------------------------------OISVM-based positioning system---------------------------------

function [ClassInterDist]=MyClassInterDist(GAMMA,X,Y)

%Compute the average inner distance of class data X
%Input: GAMMA: RBF kernel parameter
%Input: X: positive class
%Input: Y: negative class
%Output: ClassInterDist: inter cluster distance

mX=size(X,1);
mY=size(Y,1);
TempKernelX=zeros(mX,1);
TempKernelXY=zeros(mY,1);
for i=1:mX
    TempKernelX(i)=sum(exp(-1*GAMMA*sum(bsxfun(@minus,X,X(i,:)).^2,2)));
    TempKernelXY(i)=sum(exp(-1*GAMMA*sum(bsxfun(@minus,Y,X(i,:)).^2,2)));
end
KernelTotalX=sum(TempKernelX);
TempKernelY=zeros(mY,1);
for i=1:mY
    TempKernelY(i)=sum(exp(-1*GAMMA*sum(bsxfun(@minus,Y,Y(i,:)).^2,2)));
end
KernelTotalY=sum(TempKernelY);
KernelTotalXY=sum(TempKernelXY);

64
ClassInterDis = \sqrt{\frac{\text{KernelTotalX}}{(mX^2)} + \frac{\text{KernelTotalY}}{(mY^2)} - (2 \times \frac{\text{KernelTotalXY}}{mX \times mY})};

\begin{verbatim}
function [OptiGAMMA, IndexOptGAMMA] = MyGammaOptimization(TrainingData, labels, range)

% Gamma selection algorithm
% Input: TrainingData: training RSSI matrix
% Input: labels: labels of training value
% Input: range: GAMMA searching range; Default range
% range = -7:1:7
% Output: OptiGAMMA: optimal GAMMA value

PosLabels = unique(labels);
Subdata = [];
for i = 1:length(PosLabels)
    Index = find(labels == PosLabels(i));
    Subdata{i} = [Index TrainingData(Index, :) ones(size(Index, 1), 1) * PosLabels(i)];
end

GAMMA = 2. ^ range;
for K = 1:length(GAMMA)
    DistClass = [];
    for i = 1:length(Subdata)
        for j = i:length(Subdata)
            DistClass(i, j) = MyClassInterDist(GAMMA(K), Subdata{i}( :, 2 : (end-1) ), Subdata{j}( :, 2 : (end-1) ));
        end
    end
end
\end{verbatim}
CombinedClass=\texttt{tril}(\texttt{DistClass'})+\texttt{triu}(\texttt{DistClass},-1); \%
make lower triangular matrix into a diagonal matrix

\texttt{DistMatrix\{K\}}=\texttt{CombinedClass}; \%
distance matrix under parameter \texttt{GAMMA(K)}; In each cell of \texttt{DistMatrix} the component \((i,j)\) is distances from class \(i\) to class \(j\).

end;

\%
Compute the modified estimated delta4F for each
position(class) with different \texttt{GAMMA}
\%
delta4F is the indicator of the distance between
positive samples and negative samples(check the paper)

for \(K=1:length(\texttt{GAMMA})\)
for \(i=1:length(\texttt{Subdata})\)
\[\texttt{[AA, Index]}=\texttt{sort}(\texttt{DistMatrix\{K\}(i,:))}\];
\texttt{ClassInterDis(K,i)}=\texttt{DistMatrix\{K\}(i,Index(2))};
end
end

for \(i=1:size(\texttt{ClassInterDis},2)\)
\[\texttt{[ValueInterDis IndexMax]}=\texttt{max}(\texttt{ClassInterDis(:,i))}\];
\texttt{OptiGAMMA(i)=GAMMA(IndexMax)}; \%
got optimal \texttt{GAMMA} for
the maximum mean distance between the positive
class and its neatest negative class in feature space.(decrease the calculation complexity)
\texttt{IndexOptGAMMA(i)=IndexMax}; \%
and estimated delta4F
end
function IDX=MyFeatureNeighbor(X,Y,GAMMA)

%find the nearest neighbor of each Y row in X, IDX is the index of the neighbor in X, and vectors is corresponding vectors for Tomek Link

%Input: X: all instances in positive(negative) class
%Input: Y: instance in negative(positive) class
%Input: GAMMA: RBF kernel parameter
%Output: IDX index of nearest neighbor

[mx, nx]=size(X);
[my, ny]=size(Y);
if (nx˜=ny)
    error(’The column size of X and Y need to be the same’);
end

Dist=zeros(my,mx);
for i=1:my
    Dist(i,:)=sqrt(2-2.*(exp(-1*GAMMA*sum(bsxfun(@minus,X,Y(i,:)).^2,2))));
    [M,INDEX]=sort(Dist(i,:));
    vector(i,:)=X(INDEX(2,:));IDX(i)=INDEX(2);Distance(i)=M(2);
end
end

function trainingDataTLD=MyTomekLinkDeleting(
trainingData, trainingDataPositive, GAMMA)

% Delete Tomek Link for each one versus all classifier
% Input: trainingData: the RSSI training data matrix
% Input: trainingDataPositive: positive class
% Input: GAMMA: RBF kernel parameter
% Output: trainingDataTLD: trainingdata after deleted Tomek Links

IDX1 = MyFeatureNeighbor(trainingData', trainingDataPositive, GAMMA);
trainingDataT = trainingData';
TomekLinkNegativeClass = [];
for i = 1:length(IDX1)
    if (y_tr(IDX1(i)) == -1)
        IDX2 = MyFeatureNeighbor(trainingDataT, trainingDataT(IDX1(i), :), GAMMA);
        if (vector2 == trainingDataPositive(i, :))
            TomekLinkNegativeClass = [TomekLinkNegativeClass; trainingDataT(IDX1(i), :)IDX1(i)];
        end
    end
end

trainingDataTemp = [];
iNumber = 1;
if isempty(TomekLinkNegativeClass)
    trainingDataTemp = trainingData;
else
    for i = 1:size(trainingData, 2)
if ~isempty(find(TomekLinkNegativeClass(:, end)))
trainingDataTemp(:, iNumber) = trainingData(:, i);
iNumber = iNumber + 1;
end
end
end
end
trainingDataTLD = trainingDataTemp;
end

%------------------------------------------------------------------------
function negativeClassKCS = MyKCS(ALPHA, DistMatrix)
%Kernelized cluster sifting

%Input: ALPHA: parameter for determining the distant
classes; Default ALPHA=3
%Input: DistMatrix:
distance matrix
%Output: negativeClassKCS: negative class after
performing KCS

[sortedData, Index] = sort(DistMatrix(j, :)); %find the nearest class
Xtemp = [];
NearestDistance = DistMatrix(j, Index(2));
for i = 2:length(Index)
    if (DistMatrix(j, Index(i)) <= (ALPHA * NearestDistance))
        Xtemp = [Xtemp; Subdata{Index(i)}(:, 2:end-1)];
    end
end
end
negativeClassKCS = Xtemp';
end
function  modelOISVM=MyOISVMTraining(GAMMA,C,
    trainingData,label)

% train one class of OISVM model
% Input: GAMMA: kernel parameter of RBF kernel
% Input: C: penalty coefficient
% Input: trainingData: all the trainingData
% Input: label: all the labels of trainingData
% Output: modelOISVM: trained OISVM model
hp.type = 'rbf'; % Gaussian kernel: exp(-gamma |x_i-x_j|^2)
% hp.gamma = OptiGama(j); % parameter of Gaussian kernel
hp.gamma = GAMMA;
% Initialize an empty model for training
model_bak = model_init(@compute_kernel, hp);
model_bak.eta = 0.1; % parameter 'eta' of the OISVM,
    range [0,1], best for [0.01,0.1];
fprintf( 'Training..%n',j);
modelOISVM = k_oisvm_train(trainingData, label,
    model_bak); % training OISVM
fprintf( 'Done!
');
fprintf( 'Number_of_support_vectors_last_solution:%d
', numel(modelOISVM.beta));
end

%---------------------------------Particle Filter---------------------------------

classdef pedestrian

70
%%Particle Filter Class

%%Particle Filter properties and methods

%%properties:

\[ x = 0; \]
\[ y = 0; \]
\[ orientation = 0; \]
\[ forwardNoise = 0; \]
\[ turnNoise = 0; \]
\[ senseNoise = 0; \]

end

%%methods:

function \( \text{obj} = \text{pedestrian}(init_x, init_y, init\_orientation) \)

if (nargin > 0)

\[ \text{obj}.x = \text{init}_x; \]
\[ \text{obj}.y = \text{init}_y; \]
\[ \text{obj}.orientation = \text{init}\_orientation; \]
\[ \text{obj}.forwardNoise = 0; \]
\[ \text{obj}.turnNoise = 0; \]
\[ \text{obj}.senseNoise = 0; \]

end
end

function \( \text{obj} = \text{Set}(\text{obj}, \text{newX}, \text{newY}, \text{newOrientation}) \)

\[ \text{obj}.x = \text{newX}; \]
\[ \text{obj}.y = \text{newY}; \]
obj.orientation = newOrientation;
end

function obj = SetNoise(obj, newFnoise, newTnoise, newSnoise)
    obj.forwardNoise = newFnoise;
    obj.turnNoise = newTnoise;
    obj.senseNoise = newSnoise;
end

function obj = Move(obj, turn, forward)
    if (forward < 0)
        display('Error, forward cannot be less than 0');
    end
    orientationLocal = obj.orientation + (turn) + normrnd(0, obj.turnNoise);
    orientationLocal = mod(orientationLocal, 2*pi);
    dist = (forward) + normrnd(0, obj.forwardNoise) * forward;
    deltaX = sin(orientationLocal) * dist;
    deltaY = cos(orientationLocal) * dist;
    X = obj.x + deltaX;
    Y = obj.y + deltaY;
    obj = obj.Set(X, Y, orientationLocal);
    obj = obj.SetNoise(obj.forwardNoise, obj.turnNoise, obj.senseNoise);
end
function finalParticles = particleFilter (currentParticles, turnAngle, distance, net)
%Main function for running particle filter
%Input: currentParticles: particle object vector
%Input: turnAngleVector: gyroscope reading
%Input: distanceVector: pedometer reading
%Input: net: neural network object, created by
  probabilities interpolation
%Output: finalParticles: resampled particles
movedParticles = currentParticles;
finalParticles = currentParticles;
nParticles = length(currentParticles);
weightVector = zeros(1,nParticles);
for i = 1:nParticles
  movedParticles(i) = currentParticles(i).Move(
    turnAngle, distance);
  weightVector(i) = sim(net, [movedParticles(i).x
    movedParticles(i).y]);
end
totalWeight = sum(weightVector);
weightVector = weightVector/totalWeight;
maxWeight = max(weightVector);
%Resampling
beta = 0;
index = randi(nParticles - 1);
for i = 1:nParticles
    beta = beta + rand(1) * 2 * maxWeight;
    while (beta > weightVector(index))
        beta = beta - weightVector(index);
        index = index + 1;
        if (index == nParticles)
            index = 1;
        end
    end
end
finalParticles(i) = movedParticles(index);
end

function net = slfnInterpolation(hiddenLayerSize, coordinatesVector, probabilityVector)
% function for SLFNs based interpolation
% Input: hiddenLayerSize: size of hidden layer, default: 15;
% Input: coordinatesVector: coordinates of reference points on the map, dimension [N*2];
% Input: probabilityVector: probabilities of every reference point
% Output: net: the fitted probabilistic model
net = fitnet(hiddenLayerSize);
net.divideParam.trainRatio = 100/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;
[net, tr] = train(net, coordinatesVector',

%−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
function [inliers,model] = RANSAC(estimationVector,
pdrModel,maxIteration,minPoints,
modelToleranceFactor,errorToleranceFactor)

% RANSAC-based initialization algorithm
% Input: estimationVector: fingerprinting estimations
% Input: pdrModel: gaussian pdr model, estimated by
% pdrModel = fit(initSteps_X,initSteps_Y,'gauss3');
% gauss3 can be changed to
% any gauss model
% Input: maxIteration: maximum iteration of current algorithm
% Input: minPoints: minimum initial points
% Input: modelToleranceFactor: threshold for judging the inliers
% Input: errorToleranceFactor: threshold for judging the model
% Output: inliers: all the inliers
% Output: model: the initialization model
for i=1:maxIteration
    rnd = randperm(length(estimationVector));
    rndEstimationVector = [];
    nInliers=0;
    inliers = [];
    for j=1:minPoints
        rndEstimationVector=[rndEstimationVector;
    end
end
estimationVector(rnd(j),:)];
end

syms x;
minDistance=0;
minIndex=1;
ftemp=pdrModel;
for lc=1:length(rndEstimationVector)
    ftemp.b1=ftemp.b1+rndEstimationVector(lc,1);
    ftemp.b2=ftemp.b2+rndEstimationVector(lc,1);
    ftemp.b3=ftemp.b3+rndEstimationVector(lc,1);
    ytemp = ftemp.a1*exp(-((x−ftemp.b1)/ftemp.c1)^2) +
            ftemp.a2*exp(-((x−ftemp.b2)/ftemp.c2)^2) + ftemp.a3*exp(-((x−ftemp.b3)/ftemp.c3)^2)+
            rndEstimationVector(lc,2);
    distanceVector=distanceCalc(ytemp, rndEstimationVector);
    sumDistance=sum(distanceVector);
    if (lc==1)
      minDistance=sumDistance;
    elseif (sumDistance<minDistance)
      minDistance=sumDistance;
      minIndex=lc;
    end
end

ftemp.b1=ftemp.b1+rndEstimationVector(minIndex,1);
ftemp.b2=ftemp.b2+rndEstimationVector(minIndex,1);
ftemp.b3=ftemp.b3+rndEstimationVector(minIndex,1);
\[ y_{\text{temp}} = f_{\text{temp}.a1} \exp(-((x-f_{\text{temp}.b1})/f_{\text{temp}.c1})^2) + f_{\text{temp}.a2} \exp(-((x-f_{\text{temp}.b2})/f_{\text{temp}.c2})^2) + f_{\text{temp}.a3} \exp(-((x-f_{\text{temp}.b3})/f_{\text{temp}.c3})^2) + \text{rndEstimationVector}(l_\text{c},2); \]

\textbf{for} \ l_\text{c}=1: \textbf{length}(\text{estimationVector})
\text{distance}=\text{distanceCalc}(y_{\text{temp}},\text{estimationVector}(l_\text{c},:));
\textbf{if} \ \text{distance}<\text{modelToleranceFactor}
\text{nInliers}=\text{nInliers}+1;
\text{inliers}=[\text{inliers},\text{estimationVector}(l_\text{c},:)];
\textbf{end}
\textbf{end}

\textbf{if} \ \text{nInliers}>\text{errorToleranceFactor}
\text{model}=f_{\text{temp}};
\textbf{break};
\textbf{end}
\textbf{end}
Appendix B  Co-Authorship Proof

Zheng Wu

From: Rashid Rashidzadeh <rashidza@uwindsor.ca>
Sent: August 12, 2015 11:12 AM
To: Neal Wu
Subject: RE: Co-authorship Declaration

Hi Neal,

As a coauthor, I give my permission to include the following two papers in your thesis.

1. Title: A Fast and Resource Efficient Method for Indoor Positioning Using Received Signal Strength
2. Title: Improved Particle Filter Based on WLAN RSSI Fingerprinting and Inertial Sensors for Indoor Localization

Thanks,
Rashid
--------------------------------------
Rashid Rashidzadeh, Ph.D, P.Eng.
Manager, Research Centre for Integrated Microsystems & Adjunct Professor
Department of Electrical and Computer Engineering
Room 2211, CEI Building
University of Windsor
401 Sunset Avenue
Windsor, ON, Canada N9B 3P4
Tel: 519-253-3000 ext. 3931
Email: rashidza@uwindsor.ca
Website: www.vlsi.uwindsor.ca/~rashidz

From: Zheng Wu [mailto:wu111s@uwindsor.ca]
Sent: Wednesday, August 12, 2015 10:51 AM
To: Rashid Rashidzadeh <rashidza@uwindsor.ca>
Subject: Co-authorship Declaration

Hi, Dr. Rashidzadeh

I need to use the paper “A Fast and Resource Efficient Method for Indoor Positioning Using Received Signal Strength” in chapter 2.2 and paper “Improved Particle Filter Based on WLAN RSSI Fingerprinting and Inertial Sensors for Indoor Localization” in chapter 3 of my thesis “An Efficient and Accurate Indoor Positioning System”. Please give me the permission. Thanks.

Regards,
----------------------------------
Zheng Wu
MASc Candidate, Research Centre for Integrated Microsystems
Department of Electrical and Computer Engineering
University of Windsor
401 Sunset Avenue
Windsor, ON, Canada N9B3P4
Email: wu111s@uwindsor.ca
VITA AUCTORIS

NAME: Zheng Wu

PLACE OF BIRTH: Changsha, Hunan, China

YEAR OF BIRTH: 1989

EDUCATION: Central South University, Hunan, China
Bachelor of Engineering, Electronic Information Engineering 2012

University of Windsor, Windsor ON, Canada
Master of Applied Science, Electrical and Computer Engineering 2015