ALGORITHMS FOR PERVASIVE INDOOR TRACKING SYSTEMS

HAITAO BAO

(*B. Eng., HUST*)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

NUS GRADUATE SCHOOL FOR INTEGRATIVE

SCIENCES AND ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2014

Declaration

I hereby declare that the thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Haitao Bao

09 Jan 2015

To my parents

Acknowledgements

I would like to express my special thanks of gratitude to my main supervisor Prof. Lawrence Wong, who offered me the opportunity to do the research with him. He has been quite patient in guiding me into this new area, and enriched me with his knowledge, experience and insights. This thesis would not have been possible without his help.

I also want to thank my co-supervisor, Dr. Teng-Tiow Tay, with whom a lot of research problems were discussed. He has inspired me from a different perspective and directly influenced me to study the cooperative localization issue.

Thank all my dear friends and lab mates, for more than 5 years precious memorable time in Singapore. Special thanks to Dr Xiaoli Meng, who shared a lot of research experience in IMU sensor based motion capture.

Thank NGS for such a generous scholarship and great opportunity to study here. I have become a better person here.

In the end, I feel extremely grateful to my families, who has always been supporting and trusting me. I give my special thanks to my wife's support. She has been my girlfriend, my fiancée, and my wife during the writing of this thesis.

Table of Contents

Summary

An extensive amount of research has been conducted on indoor localization, a topic with numerous applications in the healthcare, retail and entertainment industries. In this thesis, we have made a contribution to the design of step-counting dead reckoning (DR) localization systems and the methodologies that can be applied towards a pervasive localization solution.

To accomplish our goals, we proposed the methods which improve the performance of previous step-counting algorithms. This involves three primary improvements: (1) an adaptive step direction estimation method, which improves the step direction estimation from the Principle Component Analysis (PCA) based algorithm; (2) a map matching (MM) method, which rectifies the error in sensor's orientation, step direction and location estimations by the known directions of the corridors; and (3) a specially designed improved particle filter (PF), which performs better than the standard PF applied in previous work in the literature. The algorithms were evaluated through extensive experiments.

We then investigated the algorithms to fuse the results from two sensors for a more robust solution. We focused on the orientation fusion, because the orientation estimation error is the primary source of the DR location error, and

there is no previous work in the literature. The experiments illustrate that the fused orientation estimation achieves more robust results than each individual solution. When we feed the orientation estimate into the DR, we notice an accuracy improvement on the location estimation.

Since personal localization hardware may not be available to all common users, we investigated the cooperative localization scheme using the existing hardware. In a wireless sensor network, the sensors are capable for pair-wise ranging measurements, or pair-wise angle measurements. The cooperative localization methods utilize such relative geo-location information, to construct the network's geo-location topology. The methods are implemented in a centralized or distributed manner. In this thesis, a cluster based scheme is proposed and evaluated. Within the cluster based scheme, three algorithms are implemented and compared: the extended Kalman filter (EKF), semi-definite programming (SDP) and multi-dimensional scaling (MDS). It is found that as the cluster size grows, the cost in terms of network overhead increases. The cluster based EKF was found to have the best performance among the cluster based algorithms, which is close to the centralized EKF.

For 2-dimensional (2D) localization, at least three anchors with known locations and three ranging distances are required to solve the location. The tracked one node's movements, returned by the motion sensing techniques, were found to relax such requirements. The DR technique is applied, so that the displacements of the node's movements can be estimated. Fusing the node's displacements estimations with the ranging distances estimations using the PF, the location can be solved even if there are only one or two anchor

nodes within the network. The simulation results illustrate that, with the combination of the DR algorithm, further improvement on location availability (number of nodes that can be localized) and accuracy can be achieved. The performances of the cluster based cooperative localization algorithm are also enhanced when the DR results are consolidated.

List of Tables

List of Figures

List of Symbols

Symbol	**Meaning**
ψ, θ, ϕ	Euler angles
$\mathbf{b} = [b_x\ b_y\ b_z]^T$	acceleration in sensor's coordinates system
r	East-North-Up (E-N-U) global coordinates system
$\mathbf{r}_g$	the Earth's gravity in East-North-Up (E-N-U) global coordinates system
$\mathbf{b}_g$	measured Earth's gravity in sensor's coordinates system
$\mathbf{r}_m$	the magnetic field in East-North-Up (E-N-U) global coordinates system
$\mathbf{r}_m$	measured magnetic field in sensor's coordinates system
Rot	3 by 3 rotation matrix
q_0	the scalar component of quaternion orientation
e	the vector component of quaternion orientation
q	quaternion orientation
$\boldsymbol{\omega}_k$	3-dimensional angular rate at time k
$\hat{\mathbf{w}}_k$	3-dimensional rotated angle at time k
$\mathbb{R}^3$, $\mathbb{R}^4$	represent three and four dimension, respectively
$\gamma(.)$	a linear mapping from $\mathbb{R}^3$ to $\mathbb{R}^4$

σ_1 and σ_2	standard deviations of the gyroscope's noise represented as Gaussian
ρ_{k+1}	variance of the magnetometer's noise represented as Gaussian
$\hat{\mathbf{\Phi}}$	transition matrix of quaternion
$\Xi(.)$	another transition matrix of quaternion
$\mathbf{P}_{k/k}$	computed a posterior quaternion estimation error covariance matrix
A(.)	attitude matrix
C(.)	matrix for cross-product computation
$\mathbf{V_b}$	vector of norm of different *b* samples
mean(.)	mean function
var(.)	variance function
threshold_1	threshold on average value for step detection
threshold_2	threshold on variance for step detection
$\mathbf{d}_n^p$	direction vector returned by PCA at time *n*
$\mathbf{d}_0^h$	initial direction vector
$\mathbf{A}_0$	sensor's initial orientation matrix
$\mathbf{b}_d$	step direction in sensor's coordinates system
m,n	step ids
$\mathbf{d}_n^h$	step-*n*'s direction estimation based on sensor's orientation analysis
$\mathbf{d}_n$	step-*n*'s direction estimation
threshold_3	threshold to detect a turn
threshold_4	threshold to detect if the sensor moves during a turn
$\mathbf{d}_n^p$	step-*n*'s direction returned by PCA
$\mathbf{x}_n$	Location after step *n*

s_n	step-n's length
κ	constant used for step length estimation, determined by training
$\mathbf{a}_n$	a string of acceleration in global coordinates system in step n
$\mathbf{a}_{n,v\text{-max}}$, $\mathbf{a}_{n,v\text{-min}}$	maximum and minimum values of $\mathbf{a}_n$ in 'Up' direction.
$\mathbf{z}_{1:n}$	observation from step 1 to step n
i, j, k	indices
$\mathbf{x}_n^i$	location of particle i in step n
N_s	number of particles
ω_n^i	weight of particle i in step n
$\delta(.)$	Dirac-Delta function
$p(.)$	the posterior probability density function
$g(.)$	importance density function
$\mathbf{n}_n^i$	noise in step n for particle i
threshold_5	threshold to determine consecutive same direction steps
threshold_6	threshold to determine if steps are within corridor
threshold_7	threshold to determine if step direction is align with corridor direction
L_n	distance from step n to corridor
$\mathbf{d}^c$	direction vector of corridor
$dot(.)$	dot product
cos(.), sin(.)	trigonometric function
$\mathbf{q}_\triangle$	rotation correction component of quaternion
$d_{angle,n}$	direction estimation at time n represented by angle
$b_{d,n}^i$	angle to compensate the direction estimation

	drift for particle i in step n
A, B	sensor ID
$\mathbf{q}_{A,0}$, $\mathbf{q}_{B,0}$	Sensor A and Sensor B's actual orientation at time 0
$\mathbf{q}_{shift}$	required rotation from sensor A to sensor B
$\hat{\mathbf{q}}_{A,t}$, $\hat{\mathbf{q}}_{B,t}$	Sensor A and Sensor B's measured orientation at time t
$\hat{\mathbf{q}}_{A,t}^{B}$	Sensor A's orientation estimation returned by B's data
N(0,variance)	Gaussian variable with zero mean and the variance
$\mathbf{q}_{A,t}^{'}$ $\mathbf{q}_{A,t}^{''}$	candidate fused orientation from Sensor A and Sensor B
$\alpha^{'}$, $\beta^{'}$, α, β	angles
$\mathbf{q}_{diff}$	quaternion difference from $\hat{\mathbf{q}}_{A,t}$ to $\hat{\mathbf{q}}_{A,t}^{B}$
$\chi(.)$	probability density function of standard normal distribution
$\tilde{\mathbf{q}}_{A,t}$, $\tilde{\mathbf{q}}_{B,t}$	sensor's orientation after fusion
$\mathbf{l}_0^j$	marker j's location at time 0
$\overline{\mathbf{q}}_{A,t}$, $\overline{\mathbf{q}}_{B,t}$	fused rotation from time 0 to time t for Sensor A and Sensor B
$\mathbf{q}_{delta}$	required rotation from estimation $\overline{\mathbf{q}}_{A,t}$ to ground truth
T_{range}	signal transmission range, normalised to 1
$\hat{d}_{ij}$, d_{ij}	measured and actual distance between node i and j
γ, w, v	Gaussian noise
r_{ij}	link robustness metric between node i and j
$\mathbf{u}$	node speed
$h(.)$	observation function of Kalman filter
$\mathbf{Q}$, $\mathbf{R}$	error covariance matrix
$\mathbf{H}$	Jacobian matrices of the partial derivatives of $h(.)$ with respect to $\mathbf{X}$

P	a posteriori error covariance matrix
K	optimal Kalman gain
$\alpha_{ij}^{+}, \alpha_{ij}^{-}, \alpha_{kj}^{+}, \alpha_{kj}^{-}, \beta_{ij}^{-}, \beta_{kj}^{-}$	optimization objectives
I	identical matrix
X	matrix of unknown locations
Y	equals $\mathbf{X}^{\mathrm{T}}\mathbf{X}$
Z	$\mathbf{Z} := \begin{pmatrix} \mathbf{I} & \mathbf{X} \\ \mathbf{X}^{T} & \mathbf{Y} \end{pmatrix} \geq 0$
$\mathbf{e}_{ij}$	vector with 1 at the *i*th position, -1 at the *j*th position and zeros everywhere else
$\mathbf{e}_{j}$	vector of all zero except -1 at the *j*th position
$\mathbf{s}_{n}$	measured displacement during time *n*
$\phi(.)$, $\varphi'(.)$	Gaussian distribution

Chapter 1

Introduction

1.1. Background

Location awareness have enabled the development of a wide variety of new exciting location based services (LBS), such as vehicle/pedestrian navigation and tracking, location based routing [1] in sensor networks, fleet management. Compared with broad applications supported by outdoor positioning techniques, like the Global Positioning System (GPS), it is extremely challenging to provide similar ubiquitous and affordable services in indoor environments. In indoor environments, the satellites signals are attenuated or completely obstructed, thus the localization result from GPS becomes unreliable or even unavailable. Unlike GPS, which provides almost full coverage for the whole of the Earth's surface, indoor localization provides a solution with a quite smaller scale.

A cellular tower can provide the signalling coverage within an extremely large area. Thus, a cellular tower based indoor localization technique [2] can

support the localization demands for a certain area using much fewer towers, but sacrifices accuracy. To provide a more accurate solution, additional indoor infrastructure deployment is required. Because of the complexity of the indoor environment, the applied techniques can be quite different.

Accurate localization techniques commonly result in higher costs for the end users. Hence, it is worth providing a location solution for the users who are not equipped with special localization hardware, even with lower accuracy. Cooperative localization is a technique which utilises the locations of certain anchor users and the relative pair-wise ranging measurements between users, to provide location solutions to common users. In this thesis, we improve the localization accuracy for the anchor users and provide localization solutions to a larger number of common users.

1.2. Overview of Existing Indoor Localization Techniques

1.2.1. Infrastructure Based Techniques

Indoor localization usually requires additional hardware deployment [3]. Various localization techniques have been developed based on IEEE 802.11 (Wi-Fi), ultrasound, Bluetooth, and so on [4][5][6]. At the current point in time, Wi-Fi is the most widely adopted wireless communication technology in the indoor and urban environments to provide wireless data access, which minimizes some extra deployment cost in the implementation of a practical indoor location tracking system. At the same time, a Wi-Fi access point provides a much bigger coverage area than that of an ultrasound or a Bluetooth beacon.

Given an indoor environment, a much smaller number of Wi-Fi access points are required to be installed, as compared with ultrasound or Bluetooth beacons, to provide the localization service. In addition, satisfactory localization accuracy can be achieved (normally 5-10 meters) for many usage scenarios [4]. Because of the above mentioned significant advantages, Wi-Fi infrastructure based indoor localization techniques have been the most widely adopted techniques in the literature. Fig. 1.1 provides a good illustration on the current positioning systems including the typical accuracy and coverage area. The horizontal axis represents the accuracy, and the vertical axis represents the coverage range.

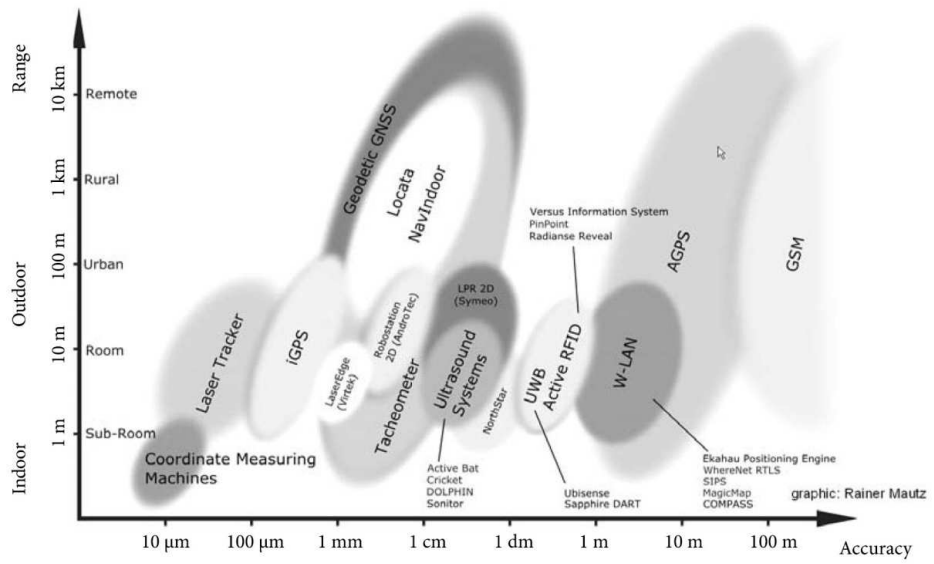


Fig. 1.1: Current positioning systems according to their accuracy and coverage area [7]

Based on the Wi-Fi signalling, two types of algorithms, namely non-training-based algorithms and training-based algorithms are proposed. Non-training-based algorithms adopt geometric trilateration methods, which usually rely on distance estimation, like in [8][9]. A small subset of algorithms also

make use of angle estimation [10][11]. Training-based algorithms rely on off-line ground truth collection. Localization results are obtained by the training process using the ground truths.

In non-training-based algorithms, there are primarily two ways to estimate the distance from the device to the Wi-Fi access point, namely using the time of arrival (ToA) and the received signal strength (RSS). The estimated distances are then utilized in trilateration methods. In ToA methods, the distance is computed by the signal propagation speed multiplied by the propagation time. In RSS methods, the distance is calculated by substituting the received signal strength into a ratio propagation model. Angle estimation requires special antennas which are implemented with multiple-input multiple-output (MIMO) techniques. The special hardware requirement is one of the reasons why angle estimation based algorithms are not as well adopted as the other algorithms.

Because of the huge errors in distance and angle estimation, non-training-based algorithms may not return localization results with satisfied accuracy. Therefore, the training-based algorithms are proposed, which are based on the assumption that the received signals are different in various locations.

Fingerprinting algorithms [8][13] are widely used training-based algorithms for indoor localization. The first step of the algorithms is site survey, which is to collect the RSSs at different locations. The signals then undergo off-line processing to obtain a reference database. In localization phase, once a new signal is received by the device, the signal is compared with those in the

database. The location with the best match signal is returned as the device location.

Given the same deployment, the training-based algorithms usually return more accurate results, as compared with the non-training-based algorithms. However, the training-based algorithms are vulnerable to environment changes, such as breaking down original walls or constructing new walls, which would change the radio propagation pattern. In addition, the site survey process is quite labour intensive.

To reduce the site survey effort, other techniques like DR are fused to help in the training process, as in [14][15][16]. Instead of standing at each possible location to collect the signals, the users can walk for a certain distance with known start and endpoint. The locations in between the start and endpoint can be captured by DR techniques. Some researchers also applied indoor map filtering to further reduce the human effort and enhance the accuracy.

If higher localization accuracy is demanded, researchers have been found to prefer facilities like ultrasound and Bluetooth, as in [5][6], relying on their higher accuracy in distance estimation. A drawback of such techniques is that they have smaller coverage, thus scaling up will incur high deployment costs. Besides, they are not as ubiquitous as Wi-Fi access points in indoor environments.

1.2.2. Dead-Reckoning Approach

To decrease the cost of infrastructure deployment, DR tracking algorithms based on inertial sensors have been proposed, as in [14][18]. In a typical DR system, an Inertial Measurement Unit (IMU) sensor includes accelerometer,

gyroscope, and sometimes, magnetometer. The DR methods derive the current location by adding the estimated displacement to the previously estimated location. The direction of the displacement is primarily determined by the measurements from the gyroscopes; the displacement length is related to the acceleration values.

Additional indoor infrastructure deployment is still required to provide the initial location estimate for the DR methods. It can also perform as the calibration reference, as the major drawback of the DR method is that the tracking error accumulates over time. But the DR method reduces the demand on the density of the deployment.

The angular rate measurements from the gyroscope are applied in estimating the sensor's orientation, so that the measured accelerations in sensor's coordinates system can then be converted to the actual moving coordinates. After that, there are different strategies in calculating the displacement in the movement. A straightforward way to estimate the displacement is to double integrate the accelerations. But double integration easily amplifies a small error to an unacceptable size.

To decrease the accumulating error from the double integration, a solution called zero velocity update (ZUPT) [19][20][21][22], which places the sensor on the foot has been proposed for pedestrian tracking systems. The ZUPT algorithm calibrates the velocity of the sensor based on the fact that the speed of a pedestrian's sole decreases to zero when it steps on the ground during the pedestrian's walking. If the sensor is affixed to the sole, the sensor's speed would also be zero. If the estimated speed is not zero, the difference is the

accumulated error of speed. This algorithm effectively reduces the error of a pedestrian DR system. However, a sensor on the sole requires extra hardware; it also causes inconvenience to the user, which restricts its use in only some special cases, like in healthcare.

Instead of placing the sensor on foot, some authors have proposed algorithms for the scenario of placing the sensor on waist. The emergence of smart phones equipped with IMU sensors impels us to study the DR algorithm in scenarios when a smart phone is used by a pedestrian. We decide to explore the scenario when the sensor is put in trouser pocket. As a study in [23] revealed, 60% of male owners carry their smart phone in the trouser pocket. Since the premise of on-foot sensor for the ZUPT algorithm does not hold in this scenario, an alternative step-counting algorithm to estimate the displacement of each step is applied.

Step-counting is a well-known algorithm to estimate the displacement for in-pocket tracking. The step-counting algorithm does not calculate based on a single acceleration measurement, but looks at the pattern of a string of accelerations. It consists of step detection and then the estimation of its displacement. The location is only updated when a full step is detected by the readings of the acceleration, by adding the displacement of the step. A step can be detected by a pair of peak and valley of the accelerations. Different formulas and algorithms have been proposed to estimate the length and direction of a step.

Both the step length estimation and the step direction estimation are dependent on the sensor's orientation estimation. A straightforward way to

compute the orientation is by the integration of the angular rate measured by the gyroscope. However, the error accumulates if there is no additional information to be fused to calibrate the error.

There is another way to compute the orientation, by using the accelerometer and magnetometer in an IMU. The acceleration and magnetic fields are two physical quantities with different directions. Assume we know their actual values in the global coordinates system (ground truth), and the measured values which are in the sensor's coordinates system. By constructing equations with the rotation matrix, the sensor's rotation can be solved. The details are explained in Chapter 3. The Earth's gravity is one commonly used source of the ground truth. Another source is the Earth's magnetic fields. The advantage of such orientation estimation method is that its accuracy does not affected by time. It is fused with the gyroscope based method, to provide a more reliable orientation estimation results.

To accomplish the orientation estimation, we let the movement (global) coordinates system be east-north-up (E-N-U), which initially coincides with the sensor's x-y-z coordinates system. The rotation of the sensor can be decomposed as rotations about its axes at the sequence of its z-y-x axis by angle ψ-θ-ϕ, respectively. Suppose the sensor is stationary after an arbitrary rotation, the direction of the Earth's gravity is used to resolve θ and ϕ. The quasi-uniqueness of the Earth's gravity over a large area provides a robust solution for θ and ϕ. Based on θ and ϕ, the horizontal and vertical components of the acceleration can be accurately decomposed. From Eq. 3-38 we know that the step length depends only on the vertical component of the acceleration.

Thus, accurate step length can be estimated. Given the θ and ϕ, ψ can be resolved from the known magnetic fields, which directly affects the step direction estimation. If the actual magnetic fields at certain places are different from the one we assume, biased results would be returned. Therefore, the accuracy of step direction estimation depends on the stability of the magnetic fields.

However, studies in [24][25] have indicated that there are considerable random disturbances of the magnetic fields in an indoor environment. Thus, the produced estimation of ψ is unreliable, and hence, the estimation of the step direction is affected. Accurate step direction estimation is an extremely challenging component, which does not always give satisfactory results.

1.2.3. Cooperative Localization

Although various localization systems such as the GPS are used, it is not economically viable to equip every node with a physical localization device, nor does the GPS operate in indoor environments. This motivates research on location estimation using relative location information, such as distance and angle measurements between nodes.

Considering a widely deployed sensor network, the number of anchor nodes, who are able to localize themselves, is typically small. Thus the normal nodes, that need to be localized, may be several hops away from the anchor nodes. None of the normal nodes would have enough information from the anchor nodes to localize itself, because of the limitation of the signal strength. However, by co-sharing the information with nearby nodes (e.g. the pair-wise ranging distances), a node is able to construct the network topology of all

nodes' locations by gathering all pieces of such information. In such a way, the nodes work cooperatively to contribute information for the others' computation. [26] did some research on analysing if a certain network topology is globally rigid that can be uniquely solved. Although different algorithms can be applied, it is worth noting that nodes in a wireless ad hoc or sensor network typically have limited power, transmission range and computational capacity.

In this thesis, we study the localization algorithm based on the locations of reference anchor nodes and the pair-wise ranging measurements between neighbour nodes. A group of work in the literature study the cases when there is no anchor node. In such cases, the solved locations are the relative locations constructing the network topology, which can be rotated as a whole in an arbitrary manner. It would be sufficient to meet some application requirements like location based routing in wireless networks. If absolute and unique solutions are desired, at least three anchor nodes are required in a two dimensional (2D) network. Some algorithms, namely the range free algorithms, do not require the pair-wise ranging measurements. The information being used is that if arbitrary two sensors are within the signal transmission range of each other. A range free algorithm usually results in less accurate solutions than the algorithm using range measurements [27].

Direct or indirect radio links to at least three anchor nodes are required for localization in a 2D network. Fig. 1.2 illustrates an example of the networking scenario, where the five-point star and the small dots are the anchor nodes and the nodes that need to be localized, respectively. The dotted lines represent the

direct linkages between the nodes. The nodes with direct linkages can perform pair-wise ranging measurements.

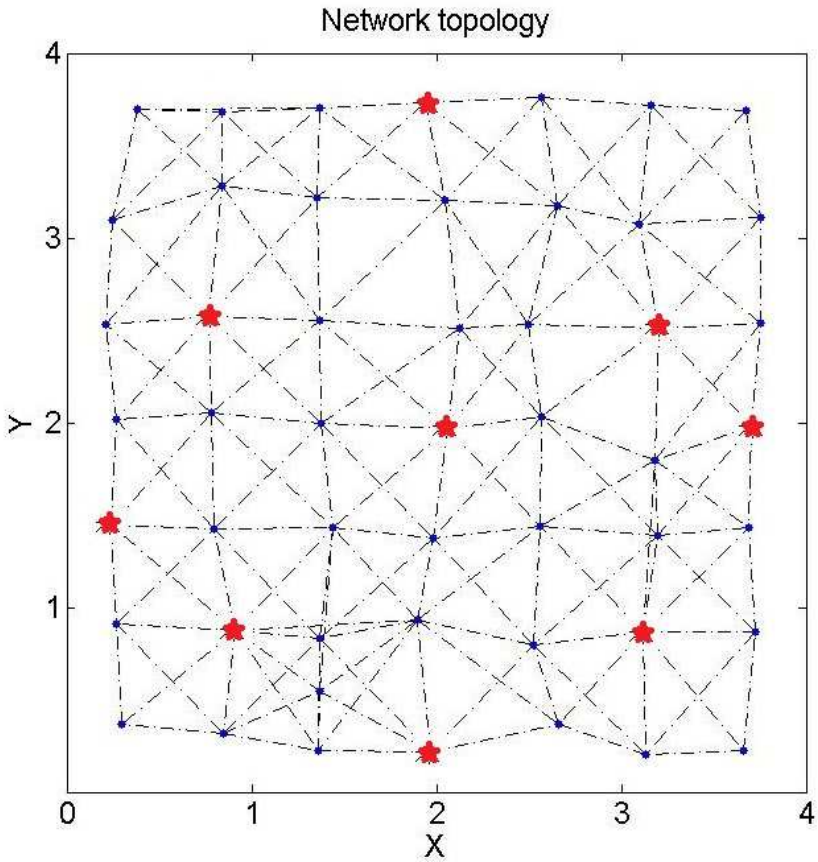


Fig. 1.2: An example of an evenly deployed 50–node wireless network in a 4 by 4 map with a normalised transmission range (1)

1.3. Research Focus and Contributions

1.3.1. Step-Counting with Map Fusion

a. Improved Step Direction Estimation

Step direction estimation is one of the key procedures for step-counting based DR tracking using inertial sensors. It is also quite challenging, especially when the captured motion data is tainted by the user's activity. The

Principal Component Analysis (PCA) is a standard tool in data analysis to reduce a complex dataset to a lower dimension [28].

The PCA based algorithm has provided robust step direction estimation results, regardless of the sensor's relative rotation compared with the human body. However, the PCA based algorithm only returns the principal axis, resolving the 180° ambiguity is another challenge. Meanwhile, the PCA based algorithm does not respond fast enough when people make turns.

In this thesis, the drawback of PCA is compensated with the sensor's orientation analysis, which returns the walking direction by analysing the change in the sensor's orientation. In our adaptive method combining PCA and sensor's orientation analysis, the sensor's orientation analysis algorithm is executed when a direction change is detected by the PCA algorithm. Because of the low computational complexity and restricted usage of orientation analysis, the adaptive method introduces little overhead, when compared to the original PCA method.

b. Map Matching Based Map Fusion

Step length estimation always returns satisfactory results, especially when training is involved to obtain the best parameters for each individual. The direction estimation in an indoor environment is the component that introduces the greatest challenge. If magnetometers are used to estimate direction, indoor environments possess significant magnetic interferences that would significantly adversely affect the accuracy of the direction estimation. On the other hand, drift in gyroscopes also pose a problem in accurate direction estimation.

The localization error of indoor DR primarily comes from the step direction estimation error, which results from the accumulating error of the orientation estimation. In this thesis, we propose a step-counting algorithm incorporating an indoor MM algorithm. It works based on the common pedestrian habit that people walking along a corridor tend to walk in a quasi-straight line along it. Then the knowledge of corridor's direction is used to calibrate the pedestrian's step direction estimation, as well as the sensor's orientation estimation. Better results are returned than the original PF technique.

c. Improved Particle Filtering for Map Fusion

A PF [29] provides good uncertainty estimates by generating enough particles. Hence, it is amenable applying PF to fuse the DR results with other information, like map constraints. The previous PF based DR techniques generated particles using an equal location error model in 2D, which did not take into account previously mentioned different error patterns for step length and direction estimation. Therefore, the modelled location update function of the PF is significantly different from the actual movement, which results in a wrong uncertainty boundary and thus large location error.

In this thesis, an improved PF, which better models the uncertainties in the step length and direction estimation, is proposed. The particles with the wrong location estimation are more likely to be the ones with the wrong direction estimations, which would be eliminated by the map constraints. Therefore, the left over particles would have better direction estimation, as well as location estimation. The improved PF returns more robust results.

1.3.2. Dual Sensor Localization

A single sensor may not provide quite robust results, because of the uncompensated bias during the sensing. Therefore, in this thesis, we applied the Maximum Likelihood Estimation (MLE) method to fuse the estimates from the two IMU sensors. The primary scope of this fusion is to improve the orientation estimation, as we find that the orientation estimation error is the major source of the location error, and that there has not been much work done in this area yet. The proposed solution achieves higher accuracy than either single sensor when two assumptions are met: 1) the measurement error distributions of the two sensors are similar; 2) The measurement errors are based on the Gaussian distribution. If the two assumptions are not met in certain circumstances, suboptimal results would be obtained that the achieved accuracy is in between the ones from Sensor *A* and Sensor *B*.

To illustrate the effect of the orientation estimation on localization, we feed the fused orientation into the DR algorithm. Except for the original solution without fusion, three fusion methods are compared: Option 1 - fusion only on orientation estimation, a DR is then applied separately in Sensor *A* and Sensor *B* for localization; Option 2 - no fusion on orientation, but fusion is applied on the two DR locations as in [92]; Option 3 - fusion on orientation first, and then fusion applied on two DR locations.

Option 1 achieves higher accuracy, as compared with the original non-fusion solution. After location fusion, Options 2 and 3 achieve almost the same average results, which is in between the accuracy of the directly computed accuracy from Sensor *A* and Sensor *B*. But the fusion solutions are

preferred, because of the robustness, which is independent of a single sensor's unpredicted drift error.

1.3.3. Cooperative Localization

a. Cluster Based Cooperative Localization

A cluster based method divides the entire network into small clusters before the location is computed in each of the cluster heads. A cluster based scheme provides a good balance of the advantages and disadvantages of the centralized and distributed schemes.

However, the previous work did not pay much attention to the clustering algorithm and the cost of the clustering. In this thesis, a 2-phase cluster based localization algorithm consisting of the clustering phase (Phase 1) and the localization phase (Phase 2) is introduced and evaluated. In Phase 1, the clusters are updated and the cluster heads receive the relative location information from their member nodes. The locations are computed by the cluster heads in Phase 2 by three algorithms, EKF, SDP and MDS. The localization results are then broadcasted in the clusters. The interferences of message broadcasting and bandwidth constraints are all simulated in this work, and the cost of the clustering algorithm are quantitatively measured. The performances of the three algorithms, namely EKF, SDP and MDS, are compared, as well as their cluster based methods.

b. Inertial Motion Sensing Improves the Localization Potential

The previous methods on cooperative localization only consider the scenario of a fully connected network (more than 3 neighbouring connections) with

many anchor nodes (more than 2) [30]. We investigate that the known movement pattern returned by the motion sensing technique relaxes the 3-anchor and 3-connection requirements for 2D localization. In this thesis, we study the case when there are only one or two anchor nodes in the network; and there is one moving node which is equipped with an IMU sensor to track the movement. A PF is applied to localize the moving node. Once the node moves close to the anchor nodes, it can get localized.

The localized moving node would be a moving anchor node to help localize the other nodes. The DR location is also fed into the original cluster based method which illustrates great improvement: there are more nodes that can be localized (increased from 0 to 48 in the 2-anchor case); and the accuracy is also enhanced.

1.4. Organization of the Thesis

The thesis is organized in the following manner. In Chapter 2, the related work regarding indoor localization is summarized. The related work is categorized by infrastructure based localization, DR localization, and wireless network cooperative localization. Chapter 3 describes our work in the single sensor step-counting localization, which provides more robust step direction estimation and better fusion of indoor map constraints by MM and an improved PF, with experimental evaluation. Chapter 4 details the results of our work and the experimental evaluation in fusing the orientation estimation from two IMU sensors. The fused orientation is fed into the original DR algorithm for a performance comparison. Chapter 5 describes the work in cooperative localization, which includes a cluster based localization scheme

and further enhancements by DR. Simulations are used to evaluate this work. Chapter 6 draws conclusions and recommends directions for future research.

Chapter 2

Literature Review

In this chapter, we classify the existing indoor localization methods in the literature into three main categories, according to their working mechanisms. The three categories are infrastructure based approaches, the dead-reckoning approaches, and the cooperative localization approaches. For each category, the research areas and the progresses are presented.

2.1. Infrastructure Based Approaches

2.1.1. The Geometric Methods

a) Received Signal Strength Indicator (RSSI) Based Methods

RSSI is the power being received by an antenna. There are different ways to represent RSSI. It can be represented as the value of RSS itself, the power attenuation (path loss) experienced during radio propagation, or others. The power attenuation mainly results from the propagation, the fading, and channel fluctuations. A known fact is that the power is attenuated as a function of the transmission distance. If the distances to at least three reference points are

available, the receiver's location can be calculated by trilateration [31]. An illustration of trilateration method is given in Fig. 2.1. The three access points (APs) are with known locations. If the distances from the UE to the APs, namely, $d1$, $d2$, and $d3$, are known, the UE's location can be determined by the intersection of the circles. Fig. 2.1 is for 2D localization scenario. For higher dimension scenarios, similar methods can be applied.

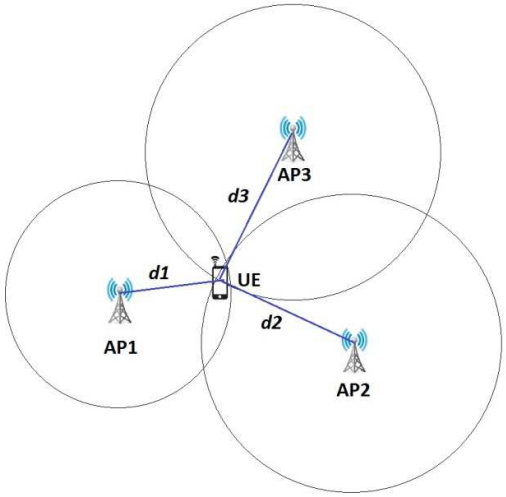


Fig. 2.1: Trilateration localization with three APs

RSSI has been the easiest and cheapest techniques for wireless localization because RSSI information is available with nearly no additional cost. All radio receivers would capture the RSS information by nature. Only the storing and processing of the information would incur the additional cost. However, the fading may result from the multipath propagation or shadowing [32]. These factors make the prediction of RSSI in radio propagation quite difficult, and different path loss models are proposed in the literature. Hence, the distance

estimation using RSSI is not reliable. The complexity of indoor environments makes the problem even more difficult to solve.

b) Time of Arrival (ToA) and Time Difference of Arrival (TDoA) Based Methods

ToA based methods also use trilateration for the location computation. But a different technique is applied for distance estimation. Based on the fact that electromagnetic waves propagate through the space at the constant speed of light ($3 \cdot 10^8$ *m/s*), the distance between the signal transmitter and the receiver can be calculated by multiplying the speed of light by the propagation time. Assume it is one-way ranging; the propagation time equals the signal receiving time minus the time the signal was being sent. It is obvious that for an accurate propagation time calculation, the clocks at the transmitter and the receiver need to be synchronized.

Synchronizing the clocks at the transmitter and the receiver may not be easy. An two-way ranging method is proposed to overcomes the need for synchronization [33]. The round-trip time (RTT) technique can support the two-way ranging method. In IEEE 802.11 standard, suppose in time t_0 a link layer data frame is sent from the transmitter to the AP. Once the AP receives the data frame, the AP would reply a corresponding link layer acknowledgement ACK. Suppose the transmitter receives the data frame in t_1. RTT is estimated by measuring the elapsed time which equals t_1-t_0. If not considering the delay in between the two data frames, the one-way propagation time equals $(t_1-t_0)/2$. The used times are the transmitter's local time. There is no need for clock synchronization.

Because of the high propagation speed of light, a small propagation time error would cause huge error in distance estimation (e.g. a propagation time error of 1 microsecond corresponds to a distance estimation error of 300m). For most indoor applications, the propagation time error must not exceed a few nanoseconds. There are four main sources of time error, namely timing errors, additive noise, multipath effect, and non-line-of-sight (NLoS) effect. Timing errors may caused by the synchronization error, clock error and drift, and the error in estimating the delay in between two data frames in two-way ranging. The addictive noise in the signal would affect the time that a complete signal is considered as received [34]. In multipath propagation, the same signal arrives at the receiver via different propagation paths. The receiver computes the signal arrival time by finding the peak of the cross-correlation between the received signal and the transmitted waveform. The line-of-sight (LoS) path is the desired path for trilateration localization, but it may not be the path with the highest peak. The computed signal arrival time could be larger than the actual time. The NLoS effect is similar to the multipath effect, in which the signal propagation does not follow the shortest path [35]. In indoor environments, the NLoS effect arises because the LoS path could be blocked by walls or other obstacles.

TDoA is a hyperbolic method, which measures the difference of distances between AP-UE pairs. In order to get the accurate distance difference, the clocks at the APs need to be synchronized. An illustration of the TDoA based localization method is given in Fig. 2.2. Each TDoA measurement would create a hyperboloid. Given the time difference, the distance difference can be

calculated. In Fig. 2.2, distance difference from the AP1 and the AP2 to the UE returns the hyperboloid d_2-d_1, and the distance difference from the AP2 and the AP3 to the UE returns the hyperboloid d_2-d_3. The intersection of them would give the UE's location. There is normally a performance degradation compared with the ToA based localization techniques, given the same information. A comprehensive study regarding the performance comparison of ToA and TDoA based localization techniques is given in [36].

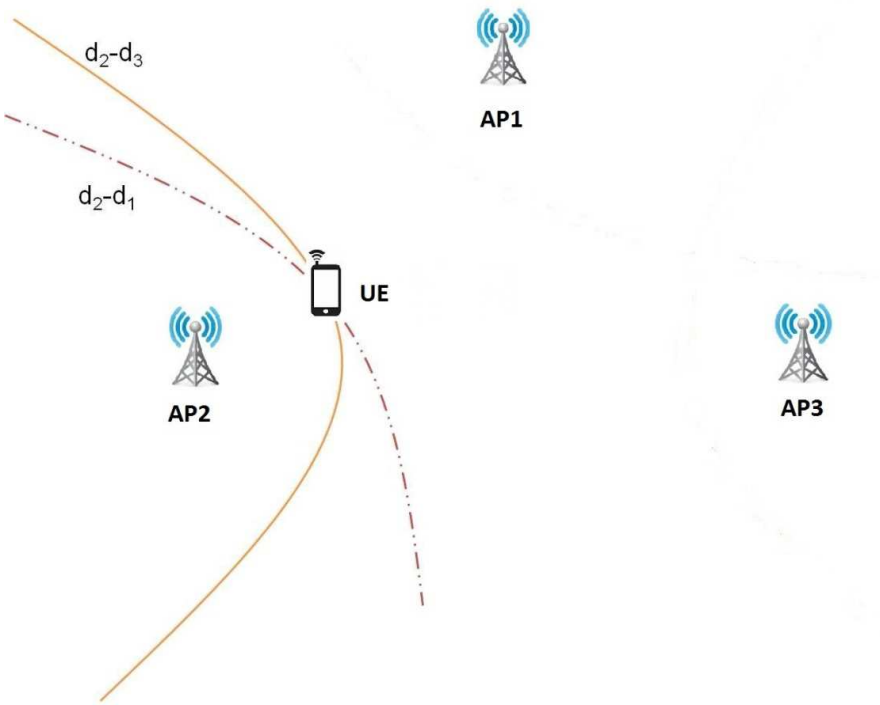


Fig. 2.2: TDoA based localization with three APs

2.1.2. The Fingerprinting Methods

The geometrical approaches described in previous section are based on the ranging measurement techniques. Because of the high complexity of the indoor environments, the ranging measurement techniques may not return the distance estimations with satisfied accuracy, no matter they are using RSSI or ToA. Hence the geometrical approaches perform poorly in indoor environment. The solutions may be even unavailable if the device is not able to receive the signals from three APs at certain places.

The fingerprinting approaches try to make use of the uniqueness of the signals at discrete locations. The approaches perform in two phases, namely the off-line site survey phase and online localization phase. In the off-line site survey phase, the location dependent parameters of the signals are measured at preselected locations, which is then processed and stored in a database. The granularity of the selected locations directly affects the location accuracy. In the online localization phase, the measured signal is compared with those in the database. The location is computed by minimizing some cost function, which tries to quantify the similarities of signals.

RSS has been the most commonly used location dependent parameter in Fingerprinting approaches, including WLAN [37] and cellular networks [38]. The online localization phase can be implemented by deterministic or probabilistic technique. An example of the deterministic technique is the K Nearest Neighbour (KNN) method, as in [37][39]. The returned location estimate is the average (or weighted average as in [39]) of the coordinates of the K locations with the best matching RSS in the database. The RSS

resolution of deterministic technique in database is determined by the granularity of off-line site survey. In a probabilistic technique, models are applied to compute the distribution of the RSS at the locations which are not covered during site survey [40][41][42]. The returned location estimate is the location with the best matching RSS. Compared with the RSS resolution of the deterministic technique, the RSS resolution of the probabilistic technique can be much higher than the one from site survey.

Compared with geometrical approaches, the fingerprinting approaches normally return more accurate location estimate. The main disadvantage of the fingerprinting approaches is the huge expenditure of resources and time, in constructing the location dependent parameters database. Maintaining the database up-to-date is also quite challenging, as the off-line phase need to be repeated to track changes in the environment.

Some attempts have been tried to reduce the site survey effort. Techniques like DR are fused in the site survey process, as in [14][15][16]. DR can track the displacements during walking. Suppose there is a corridor in which the RSS database needs to be built. The standard site survey process would be measuring some locations, and then standing still with the radio receiver at each location to collect the data. The process would be simpler with the DR techniques. Only the two locations at the ends of the corridor are required to be manually marked. The people then walk from one end of the corridor to the other, with the radio receiver and the IMU sensor being carried. The location ground truths are acquired by the DR technique, and the corresponding RSS at each location is obtained from the radio receiver.

2.2. Dead-Reckoning Approaches

2.2.1. Sensor Orientation Estimation

Estimation of the sensor's orientation is the fundamental requirement for IMU sensor based DR [43], which suggests the direction of the movement. According to Euler's rotation theorem, any sequence of rotation of a rigid body about a fixed point is equivalent to a single rotation by a given angle about a fixed axis. Therefore, any rotation in three dimensions can be represented as a combination of three-element vector (axis) and a scalar (angle). Fig. 2.3 illustrates the rotation plane and the rotation axis, and w is the rotated angle.

Quaternion provides a simple way to represent the rotation in four numbers. The orientation estimation based on the gyroscope, has been well studied in the spacecraft industry. [44] is the first comprehensive presentation of the data, theory, and practice in attitude analysis. The explained theory and assumptions in the quaternion update equation in this book, using the angular rate, are the basis for the orientation estimation in the literature. The error would accumulate if we only use the gyroscope as the data source.

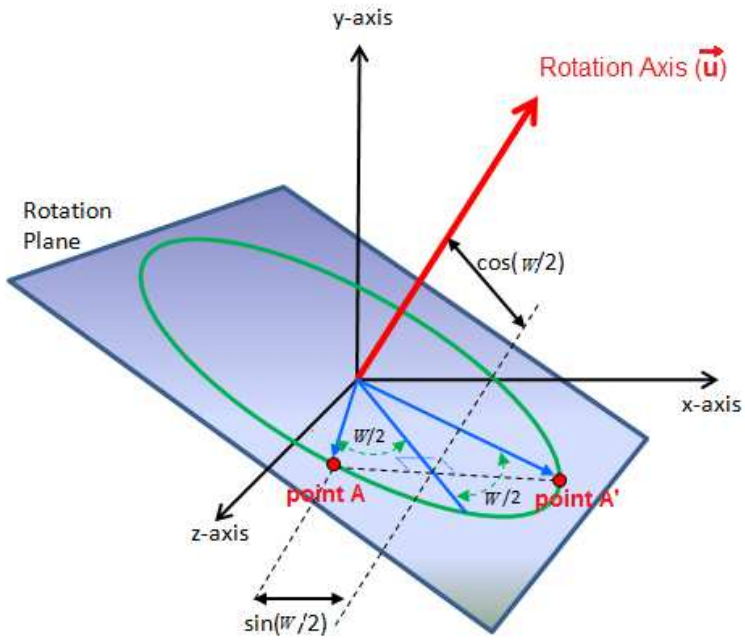


Fig. 2.3 The rotation plane and rotation axis of a rigid body

Nowadays, the inertial sensor normally contains a three-axis accelerometer, a three-axis gyroscope and a three-axis magnetometer. The Earth's magnetic field and gravity can be used to compensate for the accumulating error of orientation from the gyroscope [45][46]. [45] applied an extended Kalman filter to fuse the results from the gyroscope and the accelerometer. In [46], an unscented Kalman filter is applied to fuse the results from all three sources.

In an ideal environment, with identical magnetic fields and only the Earth's gravity, the sensor's orientation could be accurately calibrated. However, because of severe magnetic interference in an indoor environment, accurate orientation estimation is not an easy task.

Several attempts have been made to eliminate the effect of the magnetic disturbance, like the works in [47][48][49]. However, the algorithms work under the assumption that the disturbance comes from a single source and the

evaluation is taken in controlled environments. The sources of indoor magnetic disturbance are much more complex than that. Therefore, in our work, we keep in mind the error in the orientation estimation and try to eliminate the effect.

2.2.2. Step-Counting with Map Fusion

a. Step Direction Estimation

A considerable number of studies have been published on DR pedestrian tracking [14][21][50]. These DR algorithms can be divided into two categories, depending on whether the sensor is placed on the sole of a person's shoe or not. The different sensor placements introduce distinct strategies on calculating the pedestrian's displacement. The displacement of on-foot tracking is obtained by the double integration of acceleration. ZUPT is applied at each step to calibrate the acceleration and velocity, like in [19][20][21][22].

Non-on-foot tracking is calculated by step-counting, which estimates each step's length and direction. The placement of the sensor is normally on the waist or in the trouser pocket, like in [51][52][53][54]. In this thesis, we study in-pocket tracking.

Once the sensor's continuous orientations are determined, there are various methods available to determine the pedestrian's step direction. These methods are categorized by the data being analysed: sensor acceleration in the global coordinates system or sensor's orientation.

PCA is a widely used algorithm to analyse the sensor's acceleration [18][51][52]. It is based on the fact that the variation of accelerations in the

pedestrian's walking direction is the largest compared with the acceleration in other directions. So the accelerations' 1st principle component is parallel to the walking axis. The advantage of the PCA based algorithm is that it has a great tolerance for the sensor's relative movement caused by the user, which provides a robust step axis estimate. However, the PCA algorithm requires an analysis window of accelerations over several steps to obtain reliable principle component, which makes it less sensitive to walking direction changes during turns. Assume the analysis window is 1.5s, which usually covers two to three steps, and a pedestrian makes a turn in step *i*. The accelerations from step *i*-2 to step *i* would be passed to the PCA algorithm. The returned walking axis is the principle component of the last three steps, instead of desired step *i*. Only with another two steps after the turn, the PCA algorithm gives good walking axis estimation. Moreover, the PCA algorithm returns only the walking axis. Consequently, the current method on solving the 180° ambiguity still needs improvement.

Sensor orientation based methods [50][52][53] respond quickly to direction change. But they require the sensor to be relatively steady. If orientation estimation uses magnetometer sensor data, studies in [24][25] have indicated that there are considerable random disturbances of the magnetic field in an indoor environment, which makes the calibration using identical magnetic fields unreliable.

In this thesis, a simplified orientation based method, which is called sensor's orientation analysis, is applied. There are two assumptions for the sensor's orientation based algorithm: (1) that the pedestrian walks facing forward; and

(2) that the sensor is relatively steady to the body segment it is affixed to. From the two assumptions, the change of sensor's orientation only comes from the change in the pedestrian's facing direction, which is also the walking direction, in this case. If the initial walking direction is known, the subsequent walking direction can be determined by the change in the sensor's orientation.

By adaptively applying the PCA and sensor's orientation analysis, the advantage of both types of algorithms can be preserved. Hence, we present an improved PCA based robust and accurate walking direction detection method in this section.

b. Map Fusion Techniques

Considering the in-pocket tracking scenario, the step-counting algorithm [51][52][53][54] is a preferred DR algorithm. The step-counting algorithm is made up of step detection, step length estimation and step direction estimation. A step is detected by the pair of a peak and a valley of the 1-axis accelerations in the global coordinates system, meeting some quantity requirements determined by the experiments. A claimed to be more robust technique is to look at the peaks and valleys of the norm of the three-axis accelerations [55].

Step length is related to the vertical component and the norm of the horizontal component in the global coordinates system. Step direction is determined by the horizontal component of acceleration in the global coordinates system, if we only consider the 2D movement. It is obvious that the fundamental requirement is to compute the sensor's orientation accurately, so that the acceleration in the sensor coordinates system can be accurately resolved to the global coordinates system.

To improve the DR tracking results, map information has been widely used, as the map information is always available. Map filtering is applied to eliminate impossible particles, like walking through a wall or an obstacle, which is quite relevant in an indoor environment with lots of walls and obstacles. References [56][57][58] are examples of applying PF based map filtering in indoor tracking. The uncertainty in location estimation is represented as particles with different locations. While the locations of the particles are updated by the step measurement, particles with an inaccessible path are eliminated and new particles are generated. Improvement has been made in [22] by a so-called backtracking PF (BPF). BPF takes advantage of long-range geometrical constraint information that the estimated path is always backtracked, so that the particles proven to be unsuitable are eliminated in the previous steps. In this way, the localization results of the previous steps are improved.

The existing algorithms only use map information to distinguish accessible and inaccessible areas. More improvement can be achieved by applying map matching (MM). MM has been applied for outdoor environments in road tracking [59][60]. In [61], the tracked path is mapped to a nearby known corridor, if certain requirements are met. Through MM, the sensor's location, orientation, as well as the pedestrian's walking direction can be rectified. The improvement in the sensor's orientation and step direction estimation reduces the error in future tracking.

The drawback of a pure MM algorithm is that it does not make full use of the map constraints, but only the corridors. In this thesis, we apply MM as a

supplementary algorithm to the original PF method, which provides a more robust solution. Another contribution of this thesis is the improvement of the original PF.

In the literature, the error in step-counting localization is simply defined as the additive arbitrary Gaussian noise in location estimation, which does not model the error well. The error should be decomposed as an error in the step length estimation and another in the step direction estimation, which have different attributes. By defining a more accurate system model, our PF performs better than the previous ones given the same map constraint information.

2.3. Cooperative Localization Approaches

2.3.1. Centralized Vs. Distributed Methods

Localization algorithms can be centralized [49] or distributed [50], in the way they computes the locations. In centralized algorithms, the pair-wise ranging measurements and the known locations of anchor nodes are sent to a central node for computation. The computational results are then transmitted from the central node to the normal nodes. In distributed algorithms, each node would compute its location on its own.

A centralized algorithms usually yield more accurate results [51][64][65], compared with distributed algorithms, because more information can be passed to the applied model. They also allow exploiting the correlations in the measurements (e.g., correlation of the shadowing in RSS measurements [66] or sensor data measurements [67]. But the centralized algorithms require a central node with high computational power, which may not be available in

some network. The packages to be delivered to the central node also increase the network overhead. For security reasons, sometimes it is also not desired keeping all the information in one node. The typical centralized algorithms are the maximum likelihood estimator [68][69], SDP [62][70] and MDS [74].

Although a centralized algorithm usually yields more accurate results, a distributed algorithm is favoured, because of previously mentioned shortcomings of the centralized approaches. The distributed algorithms do not require a central node for the computation. This fact makes them more scalable for large networks. The distributed algorithms implement a local optimization solution, because of limited information which is available locally [71][72][73]. Some of them work in an iterative way. At each iteration, the nodes update their estimated locations by computing a local optimization function. The updated estimations are then shared within neighbours, until the convergence criterion is met. Compared with the results of centralized algorithms, which are solved from global optimization, such converged results may not be as good. The convergence process also takes time.

2.3.2. Cluster Based Method

To combine the advantages of both centralized and distributed methods, a cluster based method which divides the entire network into clusters and performs centralized localization within clusters, has been proposed.

SDP and MDS are two algorithms commonly applied for centralized computation. In the literature, the performances of their centralized method and corresponding cluster based method have been compared. In [75][76][77], cluster based MDS (CMDS) algorithms have been proposed to decrease the

error of the centralized MDS method in unevenly distributed networks. In these networks, the required estimations of the multi-hop distances are quite misleading and results in large errors. The cluster based method improves performance by decreasing the existence of multi-hops. The cluster based SDP (CSDP) has been applied in [62][78][79] to achieve comparable results to a centralized SDP.

The use of a decentralized KF (DKF) has been proposed in [80] and studied in [81]. References [81][82] have applied DKF for localization in wireless sensor networks (WSNs). In [81], every node can directly measure its own location with certain error. The relative location information is used to refine the directly measured locations and estimate the velocities of the nodes. We solve a more difficult problem that only a few anchor nodes would be able to perform self-localization. Because of the non-linearity between the states and the measurements, we apply a DEKF to solve our problem. We compare the performances of SDP, MDS, DKF and their corresponding cluster based methods CSDP, CMDS and DEKF.

Although cluster based methods have been applied for localization in wireless sensor networks, there is no study that evaluates the cost of the clustering algorithm for localization and the effect of the cluster size on the performance. The clusters in the literature are either within one hop, like in [75][76][77][83], or formed by centralized pre-partitioning, like in [78]. To study the cost and effect of the cluster based method, we propose a clustering algorithm with a modifiable cluster size (determined by the cluster head's rank, to be explained in Section 5.1.1). All decisions are made individually, without

global knowledge. In [84], cluster heads are chosen by selecting the nodes with the least node ID among its non-clustered neighbours. The size of the formed cluster is the same as ours when the rank of our cluster head is set to 2.

2.3.3. Dead-Reckoning Enhanced Scheme

Previous studies have only considered the scenario of a fully connected network (more than two neighbours) with more than two anchors [30]. But in scenarios, like indoor localization using Wi-Fi access points, the access points are usually quite sparsely deployed. There will be instances when there will not be three anchor nodes fully connected, directly or indirectly. Recent developments in miniature sensor design makes DR techniques [50][85] quite ubiquitous.

It is proven, in this thesis that the tracked movement helps to relax the 3-anchor requirement. When only one or two anchor nodes exist in the network, the DR enabled node acquires and refines its location along the movement. This moving node finally becomes a pervasive moving anchor node which helps to localize the other nodes in the network.

Chapter 3

Single Sensor Step-Counting with Map Fusion

In this chapter, we introduce the methods to improve the performance of previous step-counting algorithms. There are three improvements in total: (1) an adaptive step direction estimation method which improves the step direction estimation with a PCA based algorithm; (2) an MM method, which rectifies the error in sensor's orientation, step direction and location estimation by the direction of corridors; and (3) an improved PF, which better models the noise, and thus, performs better than the previous PF.

The adaptive step direction estimation method incorporates the advantages of both the PCA based algorithm and the sensor's orientation analysis algorithm. It is experimentally proven that the method provides more robust step direction estimation. In most cases, the method revises the error that the PCA based algorithm may make during turns and in solving the 180° ambiguity. In the worst scenarios, when the sensor is relatively moving during

the pedestrian's turns, the proposed method still provides the same performance as the PCA based algorithm, because of the restricted usage of the orientation analysis, which avoids misuse. Moreover, the improvement is achieved with little extra computational cost.

Compared with the existing indoor map fusion work, which applies map filtering using a PF, our MM algorithm calibrates both the location and the sensor's orientation and step direction, which reduces the error in the location update equation. The algorithm has been evaluated experimentally in our laboratory, when the sensor is carried in the pocket while moving.

The experimental results have shown that, given the same map information, MM returns more accurate results, as compared with the original map filtering. In addition, the performance of the MM algorithm is more robust when partial map information is available; in which case, the PF based map filtering may return a drifted path. We also combine MM with the PF, so that a more robust algorithm is proposed.

In the end, the improved PF is proposed to relax the requirement on corridor information, which also improves the step direction estimation. The localization error is decomposed and modelled as errors in step length estimation and step direction estimation with different attributes. The experimental results illustrate that, in a quite dense map constraint environment with corridors, the improvement is not obvious. But when only partial map constraints are applied, the improved PF scheme outperforms the other schemes with less performance dependence on the corridor constraints.

3.1. Improved PCA Based Step Direction Estimation for Dead-Reckoning Localization

The sensor we use in this investigation is the ADIS16405BMLZ [95], containing a three-axis accelerometer, a three-axis gyroscope and a three-axis magnetometer. For better understanding, some specifications of the sensor are presented in Table 3-1.

Table 3-1: ADIS16405BMLZ specifications

	Parameter	Test Conditions	Min	Type	Max	Unit
GYROSCOPES	Dynamic Range		±300	±350		°/sec
	Initial Sensitivity	Dynamic range = ±300°/sec	0.0495	0.05	0.0505	°/sec/LSB
		Dynamic range = ±150°/sec		0.025		°/sec/LSB
		Dynamic range = ±75°/sec		0.0125		°/sec/LSB
	Sensitivity Temperature Coefficient	ADIS16405: −40°C ≤ T_A ≤ +85°C		±40		ppm/°C
	Initial Bias Error	1 σ		±3		°/sec
	In-Run Bias Stability	1 σ, SMPL_PRD = 0x01		0.007		°/sec
ACCELEROMETER	Dynamic Range		±18			*g*
	Initial Sensitivity		3.285	3.33	3.38	mg/LSB
	Sensitivity Temperature Coefficient	ADIS16405: −40°C ≤ T_A ≤ +85°C		±50		ppm/°C
	Initial Bias Error	1 σ		±50		m*g*
	In-Run Bias	1 σ		0.2		m*g*

	Stability					
MAGNETOMETER	Dynamic Range		±2.5	±3.5		gauss
	Initial Sensitivity	25°C	0.49	0.5	0.51	mgauss/LSB
	Sensitivity Temperature Coefficient	25°C，1 σ		600		ppm/°C
	Axis Misalignment	25°C，axis-to-base plate and guide pins		0.5		Degrees
	Initial Bias Error	25°C , 0 gauss stimulus		±4		mgauss

The usage of the sensor is shown in Fig. 3.1. The sensor is connected to a computer through a USB (Universal Serial Bus) interface for data collection. To simulate the usage of a smart phone, the sensor is put in the trouser pocket as illustrated in the figure to collect the data. On occasion, the sensor is rotated arbitrary to mimic the usage interference. The software interface is shown in Fig. 3.2, where the sampled data would be stored in a text file.

Fig. 3.1: ADIS16405BMLZ in-pocket tracking scenario

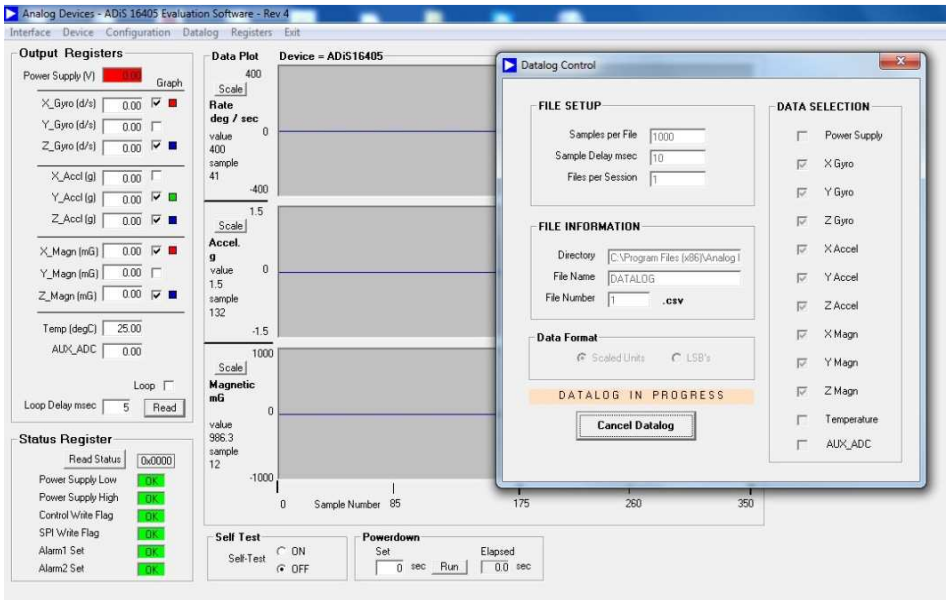


Fig. 3.2: ADIS16405BMLZ evaluation software interface

3.1.1. Step Direction Estimation Process

a. Sensor's Orientation Determination

The concept and computation of quaternion have been given in Appendix. [96] has proposed a simplified quaternion-based orientation estimation algorithm based on the Earth's gravity and magnetic field measurements. A similar method is applied to get the sensor's initial orientation. There are two coordinates systems: the global coordinates system and the sensor coordinates system. The global coordinates system is set as the east-north-up (E-N-U) coordinates system. The orientation from the global coordinates system to the sensor's coordinates system needs to be solved. The known information is the measurements returned by the three-axis inertial sensor, which indicates the physical quantities in the *x*-*y*-*z* sensor coordinates system.

Initially, when the sensor remains static, the accelerometer measures only the Earth gravity, which is represented as $\mathbf{r}_g = [0;0;-1]$ in E-N-U global coordinates system after normalisation. In addition, assume that the magnetic fields at the starting point in the global coordinates system are a known value represented as $\mathbf{r}_m = [m_e; m_n; m_u]$. The measurements in the sensor coordinates system, compared with the global coordinates system, are represented as $\mathbf{b}_g$ and $\mathbf{b}_m$, respectively.

According to Euler's theorem, an arbitrary rotation of a rigid body can be decomposed as consecutive rotations about three axes. In this thesis, a rotation is decomposed as rotating about its axes at the sequence of its *z*-*y*-*x* axis by angle ψ-θ-ϕ, respectively.

Suppose $\mathbf{q}_z$, $\mathbf{q}_y$ and $\mathbf{q}_x$ are the respective quaternion in each axis. From the definition of quaternion in Eq. A-16, $\mathbf{q}_z$, $\mathbf{q}_y$ and $\mathbf{q}_x$ are:

$$\begin{aligned}\mathbf{q}_z &= [\sin(\psi/2)[0;0;1];\cos(\psi/2)]\\ \mathbf{q}_y &= [\sin(\theta/2)\,[0;1;0];\cos(\theta/2)]\\ \mathbf{q}_x &= [\sin(\phi/2)\,[1;0;0];\cos(\phi/2)]\end{aligned} \qquad \textbf{(3-1)}$$

Thus, the initial orientation, represented by quaternion **q,** equals the Hamilton product of the individual quaternion as follows:

$$\mathbf{q}=\mathbf{q}_z \otimes \mathbf{q}_y \otimes \mathbf{q}_x \qquad \textbf{(3-2)}$$

where $\otimes$ represents the Hamilton product for quaternion multiplication [44].

Substitute $\mathbf{q}_z$, $\mathbf{q}_y$, and $\mathbf{q}_x$ into Eq. A-15, we have the attitude matrix at each axis and the final matrix:

$$\begin{aligned}\mathbf{Rot}(z,\psi) &= \mathbf{A}([\sin(\frac{\psi}{2})[0;0;1];\cos(\frac{\psi}{2})]) = \begin{bmatrix}\cos(\psi) & \sin(\psi) & 0\\ -\sin(\psi) & \cos(\psi) & 0\\ 0 & 0 & 1\end{bmatrix}\\ \mathbf{Rot}(y,\theta) &= \mathbf{A}([\sin(\frac{\theta}{2})[0;1;0];\cos(\frac{\theta}{2})]) = \begin{bmatrix}\cos(\theta) & 0 & -\sin(\theta)\\ 0 & 1 & 0\\ \sin(\theta) & 0 & \cos(\theta)\end{bmatrix}\\ \mathbf{Rot}(x,\phi) &= \mathbf{A}([\sin(\frac{\phi}{2})[1;0;0];\cos(\frac{\phi}{2})]) = \begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\phi) & \sin(\phi)\\ 0 & -\sin(\phi) & \cos(\phi)\end{bmatrix}\\ \mathbf{Rot}(z,y,x) &= \mathbf{Rot}(z,\psi)\mathbf{Rot}(y,\theta)\mathbf{Rot}(x,\phi)\end{aligned} \qquad \textbf{(3-3)}$$

In Eq. 3-4 and Eq. 3-5, the right sides in the equations represent the actual Earth's gravity and magnetic fields in sensor's coordinates system after the rotation. The left sides in the equations are the measured values using the sensor. Assuming that there is no error in the measurement, then Eq. 3-4 and Eq. 3-5 should hold.

$$\mathbf{b}_g = \mathbf{Rot}(z, y, x)\mathbf{r}_g \tag{3-4}$$

$$\mathbf{b}_m = \mathbf{Rot}(z, y, x)\mathbf{r}_m \tag{3-5}$$

Substituting $\mathbf{r}_g = [0; 0; -1]$ to Eq. 3-4 we have:

$$\mathbf{b}_g = [\sin(\theta); -\sin(\phi)\cos(\theta); -\cos(\phi)\cos(\theta)] \tag{3-6}$$

Suppose the acceleration measurement $\mathbf{b} = [b_x, b_y, b_z]^T$. In the ideal scenario, $\mathbf{b}$ should equal $\mathbf{b}_g$, then θ and ϕ are solved by:

$$\text{minimize} \|\mathbf{b} - \mathbf{b}_g\|^2 \tag{3-7}$$

There are quite a few existing methods to solve the minimum mean square error problem.

Suppose the magnetic field measurement $\mathbf{m} = [m_x, m_y, m_z]^T$, and substituting θ and ϕ that are solved by Eq. 3-7 into Eq. 3-5, then ψ is solved by:

$$\text{minimize} \|\mathbf{m} - \mathbf{b}_m\|^2 \tag{3-8}$$

where ψ is the only variable.

Given the initial orientation, the continuous orientation estimation and calibration are based on the novel quaternion Kalman filtering algorithm proposed in [97]. Here we illustrate only the results, without the formula derivation steps. Suppose the measurement from the three-axis gyroscope at time k is $\boldsymbol{\omega}_k$, the rotated angle $\hat{\mathbf{w}}_k$ equals

$$\hat{\mathbf{w}}_k = \boldsymbol{\omega}_\mathbf{k} \Delta t \tag{3-9}$$

where Δt is the sampling interval.

Let $\gamma(\mathbf{x})$ denote the linear mappings of $\mathbf{x} = [x_x; x_y; x_z]$ from a three dimensional space $\mathbb{R}^3$ to a four dimensional space $\mathbb{R}^4$ which is defined as follows:

$$\gamma(\mathbf{x}) \triangleq \begin{bmatrix} -\mathbf{C}(\mathbf{x}) & \mathbf{x} \\ -\mathbf{x}^T & 0 \end{bmatrix} \tag{3-10}$$

, which **C**(.) has been defined in Appendix A.2

Let $\Xi(\mathbf{q})$ denote the following transformation:

$$\Xi(\mathbf{q}) = \begin{bmatrix} \mathbf{C}(\mathbf{e}) + q_0 \mathbf{I}_3 \\ -\mathbf{e}^T \end{bmatrix} \tag{3-11}$$

Suppose σ_1 and σ_2 are the standard deviations of the gyroscope's electronic noise and float torque noise, respectively. The sensor's orientation **q** is the state we need to estimate in the Kalman filter. The measurements for the a priori state estimate are the angular rates from gyroscope. The rotation angle $\hat{\mathbf{w}}_k$ is calculated using Eq. 3-9. Eq. 3-12 to Eq. 3-14 are the equations for the a priori state estimate:

$$\hat{\mathbf{\Theta}}_{k/k} = \gamma(\hat{\mathbf{w}}_k) \tag{3-12}$$

$$\hat{\mathbf{\Phi}}_{k/k} = \exp(\frac{1}{2}\hat{\mathbf{\Theta}}_{k/k}) \tag{3-13}$$

$$\hat{\mathbf{q}}_{k+1/k} = \hat{\mathbf{\Phi}}_{k/k}\hat{\mathbf{q}}_{k/k} \tag{3-14}$$

Eq. 3-15 to Eq. 3-20 are applied to get the a priori estimate covariance.

$$\mathbf{\Theta}_k = \gamma(\mathbf{w}_k) \tag{3-15}$$

$$\mathbf{\Phi}_k = \exp(\frac{1}{2}\mathbf{\Theta}_k) \tag{3-16}$$

$$\hat{\mathbf{\Xi}}_{k/k} = \Xi(\hat{\mathbf{q}}_{k/k}) \tag{3-17}$$

$$\hat{\mathbf{M}}_{k/k} = \hat{\mathbf{q}}_{k/k}\hat{\mathbf{q}}_{k/k}^T + \mathbf{P}_{k/k} \tag{3-18}$$

$$\mathbf{P}_k^w = (\sigma_1^2 + \sigma_2^2 \Delta t)\frac{1}{4}[tr(\hat{\mathbf{M}}_{k/k})\mathbf{I}_4 - \hat{\mathbf{M}}_{k/k}] \quad \textbf{(3-19)}$$

$$\mathbf{P}_{k+1/k} = \hat{\mathbf{\Phi}}_k \mathbf{P}_{k/k} \hat{\mathbf{\Phi}}_k^T + \mathbf{P}_k^w \quad \textbf{(3-20)}$$

Suppose the measurement from magnetometer or accelerometer and the respective ground truth in time k+1 are represented using $\mathbf{z}_{k+1}$ and $\mathbf{r}_{k+1}$, respectively. The measurement error is with covariance matrix $\rho_{k+1}\mathbf{I}_3$. Let

$$\mathbf{u}_{k+1} = \frac{1}{2}(\mathbf{z}_{k+1} + \mathbf{r}_{k+1}) \quad \textbf{(3-21)}$$

$$\mathbf{v}_{k+1} = \frac{1}{2}(\mathbf{z}_{k+1} - \mathbf{r}_{k+1}) \quad \textbf{(3-22)}$$

$$\mathbf{H}_{k+1} = \begin{bmatrix} -\mathbf{C}(\mathbf{u}_{k+1}) & \mathbf{v}_{k+1} \\ -\mathbf{v}_{k+1}^T & 0 \end{bmatrix} \frac{1}{2}(\mathbf{z}_{k+1} + \mathbf{r}_{k+1}) \quad \textbf{(3-23)}$$

The measurement equation equals:

$$\mathbf{0} = \mathbf{H}_{k+1}\mathbf{q}_{k+1} - \frac{1}{2}\mathbf{\Xi}_{k+1}\mathbf{v}_{k+1} \quad \textbf{(3-24)}$$

$$\hat{\mathbf{M}}_{k+1/k} = \hat{\mathbf{q}}_{k+1/k}\hat{\mathbf{q}}_{k+1/k}^T + \mathbf{P}_{k+1/k} \quad \textbf{(3-25)}$$

$$\mathbf{B}_{k+1} = \gamma(\mathbf{z}_{k+1}) \quad \textbf{(3-26)}$$

$$\mathbf{P}_{k+1}^v = \frac{1}{4}\rho_{k+1}[tr(\hat{\mathbf{M}}_{k+1/k})\mathbf{I}_4 - \hat{\mathbf{M}}_{k+1/k} - \mathbf{B}_{k+1}\hat{\mathbf{M}}_{k+1/k}\mathbf{B}_{k+1}^T] \quad \textbf{(3-27)}$$

The residual covariance $\mathbf{S}_{k+1/k}$, is calculated by Eq. 3-28:

$$\mathbf{S}_{k+1/k} = \mathbf{H}_{k+1}\mathbf{P}_{k+1/k}\mathbf{H}_{k+1}^T + \mathbf{P}_{k+1}^v \quad \textbf{(3-28)}$$

The optimal Kalman gain $\mathbf{K}_{k+1}$ is calculated by Eq. 3-29:

$$\mathbf{K}_{k+1} = \mathbf{P}_{k+1/k}\mathbf{H}_{k+1}^T\mathbf{S}_{k+1/k}^{-1} \quad \textbf{(3-29)}$$

The a posteriori state estimate $\hat{\mathbf{q}}_{k+1/k+1}$ is calculated by Eq. 3-30:

$$\hat{\mathbf{q}}_{k+1/k+1} = (\mathbf{I}_4 - \mathbf{K}_{k+1}\mathbf{H}_{k+1})\hat{\mathbf{q}}_{k+1/k} \quad \textbf{(3-30)}$$

The a posterior estimate covariance $\mathbf{P}_{k+1/k+1}$ equals:

$$\mathbf{P}_{k+1/k+1} = (\mathbf{I}_4 - \mathbf{K}_{k+1}\mathbf{H}_{k+1})\mathbf{P}_{k+1/k}(\mathbf{I}_4 - \mathbf{K}_{k+1}\mathbf{H}_{k+1})^T + \mathbf{K}_{k+1}\mathbf{P}^v_{k+1}\mathbf{K}^T_{k+1} \quad \textbf{(3-31)}$$

The computed $\hat{\mathbf{q}}_{k+1/k+1}$ may not be a unit vector anymore, Eq. 3-32 is required to normalize it,

$$\mathbf{q}^*_{k+1/k+1} = \frac{\hat{\mathbf{q}}_{k+1/k+1}}{\| \hat{\mathbf{q}}_{k+1/k+1} \|} \quad \textbf{(3-32)}$$

In our work, the state is calibrated by using the acceleration reading when the sensor is considered stationary. Suppose the nth sampling of $\mathbf{b}$ is represented as $\mathbf{b}_n$, $\mathbf{V_b}$ is a vector containing the norm of the sequential values of $\mathbf{b}$ (i.e. $[\|\mathbf{b}_{n-m}\|;\|\mathbf{b}_{n-m+1}\|;...\|\mathbf{b}_n\|;...\|\mathbf{b}_{n+m-1}\|;\|\mathbf{b}_{n+m}\|]$), $mean(\mathbf{V_b})$ is the mean of $\mathbf{V_b}$, and $var(\mathbf{V_b})$ is the variance of $\mathbf{V_b}$. The stationary status is detected by Rule 1.

Rule 1: If $\|mean(\mathbf{V_b})-\|\mathbf{r_g}\|\| < threshold\_1$ and $\mathrm{var}(\mathbf{V_b}) < threshold\_2$, the sensor is stationary.

The first inequality ensures that the average acceleration is close to the Earth's gravity. But an average value is not sufficient enough to determine the quasi-stationary state. Therefore, the second inequality would further ensure each acceleration measurement would be close to the Earth's gravity. The sensor we use is quite accurate in acceleration estimation with an in-run bias of $0.0002\|\mathbf{r_g}\|$ (Table 3-1). An example of an acceleration measurement during a 10-step walk is also given in Fig. 3.3.

Based on the known sensor accuracy, strict criteria is chosen in our implementation, in which *threshold_1* is set to $0.008||\mathbf{r_g}||$ and *threshold_2* is set to $0.0001||\mathbf{r_g}||^2$. As shown in Fig. 3.3, the acceleration fluctuates during walking and its norm equals to the Earth gravity for some instances. If a small *m* is chosen, the pedestrian status could be wrongly detected as static by Rule 1 during walking. But if a large *m* is chosen that it is more difficult to meet the condition, the quaternion orientation may only be calibrated by the detected stationary state after a long period of time. So there is a trade-off between the accurate stationary detection and the higher chance of getting calibrated. In our algorithm, *m* is set to 20, when the sampling rate is 100 Hz, which would be able to capture the scenario when a pedestrian is hesitating or slowly making turns.

b. Step Detection

[55] has argued that by analysing the pattern of $||\mathbf{b}||$, rather than acceleration on a single axis, a more robust step detection algorithm can be proposed. A 3Hz low pass filter is applied on $||\mathbf{b}||$. An example is illustrated in Fig. 3.3, when 10 steps are taken. It is interesting to observe that the shapes for the left foot and the right foot steps are different. The right foot step ends with the acceleration close to the Earth's gravity. But the left foot does not. The reason for this is that the sensor is placed in the person's right pocket. When the right foot steps on the ground, the sensor is in a quasi-stationary status.

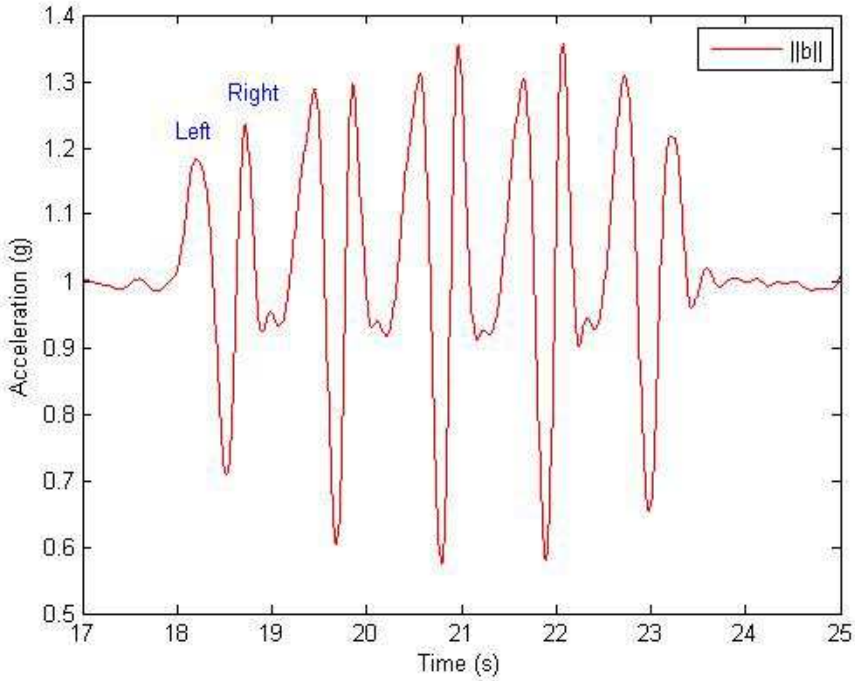


Fig. 3.3: An example of acceleration ||b|| for 10 steps

3.1.2. Adaptive Step Direction Estimation

a. Principle Component Analysis (PCA) Based Direction Estimation

A PCA based step axis estimation has been applied in [21][51]. In this thesis, 2D horizontal accelerations of window size 1.5 seconds are processed by the PCA. The returned first component is the step axis, not the direction. By analysing the pattern of the vertical and forward accelerations, [18] also indicates that the positive peak of forward acceleration falls in the time when the vertical acceleration has an increasing slope. This is proven in Fig. 3.4, when the real data is plotted. This property solves the 180° ambiguity by determining the direction from the step axis. The direction of step *n*, returned by the PCA, is represented as $\mathbf{d}_n^p$.

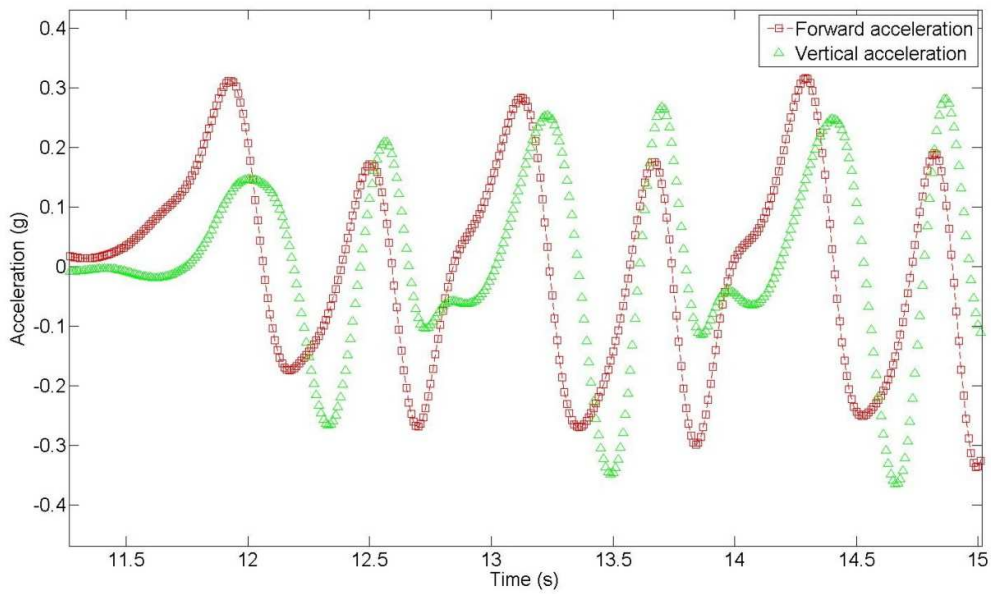


Fig. 3.4: Typical patterns in vertical and forward acceleration during walking

b. Sensor's Orientation Based Direction Estimation

The following assumptions apply:

1) The pedestrian walks facing forward; and

2) The sensor does not move relatively to the body segment it is affixed to.

The direction estimation problem has been simplified to determine the pedestrian's facing (step direction), given the initial facing and the pedestrian's angular velocity (sensor's angular velocity) when a step ends. By comparing the sensor's orientations at the step endpoints, the difference indicates the direction change. Here, we only consider walking in the horizontal plane. Suppose the initial direction is the 2-D unit vector $\mathbf{d}_0^h$ and the sensor's attitude matrix is represented as **A**. The vector $\mathbf{b}_d$ in the sensor coordinates system pointing in the step direction is calculated as:

$$\mathbf{b}_d = \mathbf{A}_0[\mathbf{d}_0^h; 0] \tag{3-33}$$

From the 2nd assumption, $\mathbf{b}_d$ in the sensor coordinates system always points toward the step direction. Suppose step-n's direction and sensor's orientation matrix are $\mathbf{d}_n$ and $\mathbf{A}_n$, respectively. Then the following equation holds

$$\mathbf{b}_d = \mathbf{A}_n \mathbf{d}_n \tag{3-34}$$

Substituting Eq. 3-33 into Eq. 3-34 we obtain:

$$\mathbf{d}_n = (\mathbf{A}_n)^{-1} \mathbf{A}_0[\mathbf{d}_0^h; 0] \tag{3-35}$$

In the horizontal plane, the direction from the orientation analysis $\mathbf{d}_n^h$ is

$$\mathbf{d}_n^h = [\mathbf{d}_n(1); \mathbf{d}_n(2)] / \sqrt{(\mathbf{d}_n(1))^2 + (\mathbf{d}_n(2))^2} \tag{3-36}$$

The first assumption is always met for the normal cases considered in pedestrian tracking. But as part of the personal mobile device, the sensor moves from time to time. Therefore, the sensor's orientation analysis algorithm cannot be applied in these scenarios. The computational complexity for the orientation analysis is almost negligible, as it is only executed after each step.

c. Adaptive Direction Estimation

The sensor's orientation analysis algorithm responds quickly to the direction change, but is sensitive to the sensor's relative movement. In our adaptive direction estimation method, it works as a supplement to the PCA based algorithm during turns. The process is shown in Fig. 3.5. A turn detected by the PCA may be caused by the 180° ambiguity or real direction change. In both cases, verification of the result by orientation analysis is necessary. The

key component of the process is to differentiate if there is a relative movement of the sensor during a turn. Only when the sensor keeps relatively static, is its direction estimation result reliable.

The detailed pseudo-code for the process is presented in Fig. 3.6. The dot product of the continuous step is calculated. If the dot product is larger than *threshold_3*, the pedestrian is detected as walking in about the same direction. A large *threshold_3* would make the algorithm sensitive to a direction change, but easily trigger false turn detections. The value is chosen by the experiment. It is observed that even though a pedestrian walks towards a line, the estimated direction may differ up to 15°. The difference comes from the pedestrian walking pattern, as well as the error in the direction estimation. Such a difference can be easily observed in Fig. 3.7 a). In our implementation, *threshold_3* is set as $cos(15^{o})$, as such, a false turn detection would only consume a little bit more computation power. Same threshold is set for *threshold_4*, as the 15° difference may be caused by the error in direction estimation, not by the sensor's movement during the turn.

When one of the previous dot products is smaller than *threshold_3*, and the current dot product is larger than *threshold_3*, the pedestrian is deemed to have made a turn. As turns normally only last for two or three steps, checking for changes in four consecutive steps will be adequate to detect a change in direction. The PCA algorithm, with a window size of 1.5s, is adequate to reflect the direction change.

The PCA algorithm returns accurate direction estimates during the non-turning phase. So $\mathbf{d}_{m-1}^{p}$ and $\mathbf{d}_{n}^{p}$ in Phase 6 are the true directions. For the step

count between them, the smaller the better, as there would be a higher chance that the sensor does not relatively move. The determination of whether the sensor relatively moves is based on the true directions before (m-1) and after (n) a turn, given by the PCA. Given $\mathbf{d}_{m-1}^{p}$ as a sensor's orientation based algorithm's initial direction in step m-1, the algorithm should return almost the same direction estimate for step n ($\mathbf{d}_{n}^{h} \approx \mathbf{d}_{n}^{p}$) if the sensor does not move. Although conversely, the statement is not absolutely true, it is good enough to be used as an empirical condition.

Moreover, another condition that is applied in our implementation is that the estimated step directions by orientation analysis during a turn should have the same trend as the one from $\mathbf{d}_{m-1}^{p}$ to $\mathbf{d}_{n}^{p}$ (left or right). This further reduces the probability of an erroneous orientation analysis. Even in the erroneous case, the estimated direction meeting the conditions is close to the true direction. In addition, the restricted usage of the orientation analysis keeps the introduced overhead low.

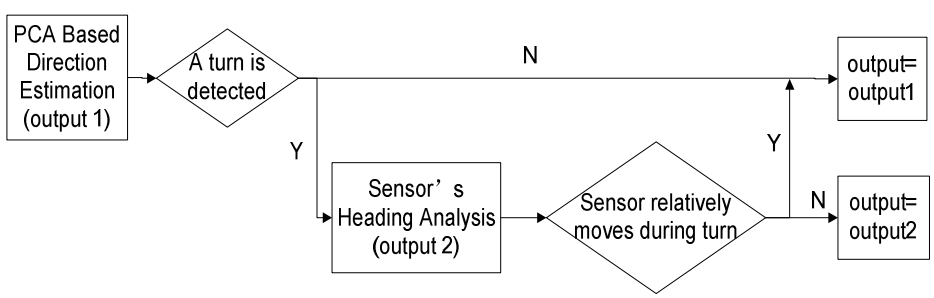


Fig. 3.5: Process of adaptive direction estimation

```
Adaptive direction estimation (n):
Suppose estimated direction of step n is represented as d_n,
then d_n = d_n^p :
compute dot_n = <d_{n-1}^p, d_n^p>, (n > 1)
if ∃ dot_{n-j} < threshold_3, j ∈ [1,4]
  then a turn is detected,
  if dot_n > threshold_3
  find largest m meets dot_m > threshold_3, m ∈ [n-4, n-1]
    then d_{m-1}^p and d_n^p are accurate directions before and after the turn separately
    let d_{m-1}^h = d_{m-1}^p, compute d_n^h
    if < d_n^p, d_n^h > > threshold_4
      then the sensor does not move relative to the pedestrian during the turn
      for k=m to n-1, compute d_k^h, d_k = d_k^h, dot_k = 1
```

Fig. 3.6: Pseudo-code for adaptive direction estimation

Therefore, if the sensor is detected to be relatively static during a turn, the sensor's orientation based algorithm returns a more reliable direction estimate for these steps. This rectifies not only the PCA's error in detecting the step axis during the turns, but also the error in solving the 180° ambiguity in certain cases. The sensor's orientation analysis in the adaptive method is only executed when a turn is detected. Therefore, the incurred extra cost is quite limited which makes the method as efficient as the PCA based method. Once the previous step directions are rectified by the sensor's orientation analysis, the location estimation of the previous steps also needs to be recalculated by the step-counting algorithm.

3.1.3. Experimental Studies

To view the effect of the step direction estimation algorithm, it is applied in a step-counting localization algorithm. The estimated location after step n is calculated by Eq. 3-37.

$$\mathbf{x}_n = \mathbf{x}_{n-1} + s_n \mathbf{d}_n \tag{3-37}$$

where s_n is step-n's length. In order to resolve s_n and d_n , the premise is that the sensor's orientation is accurately determined and a step is correctly detected.

The estimation of the step length, given the accelerations, have been well studied in [20][54] . We use the estimation formula in [54] illustrated in Eq. 3-38:

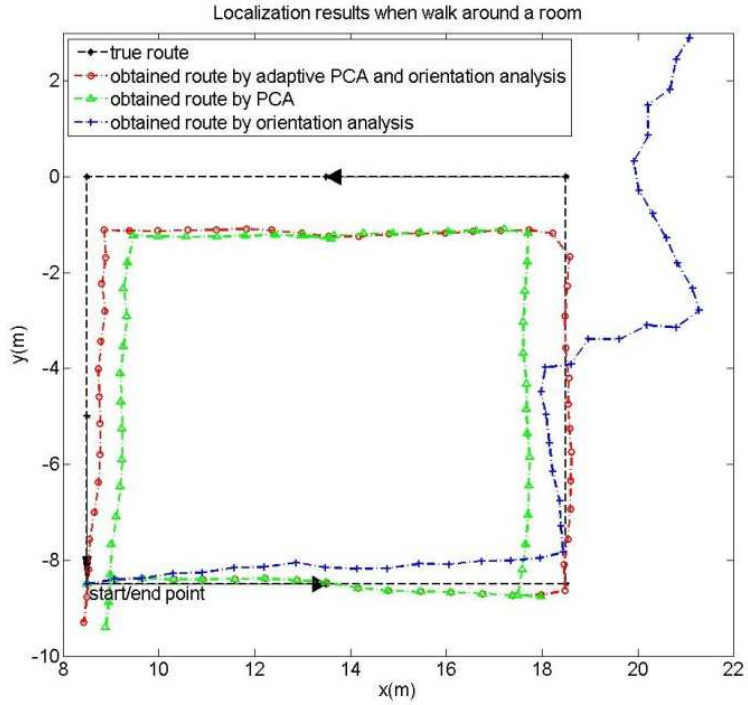
$$s_n = \kappa \sqrt[4]{\mathbf{a}_{n,v-\max} - \mathbf{a}_{n,v-\min}} \tag{3-38}$$

where: $\mathbf{a}_n$ is a string of a projected acceleration in the global coordinates system in step n and $\mathbf{a}_{n,v\text{-max}}$ and $\mathbf{a}_{n,v\text{-min}}$ are the maximum and minimum values of its vertical components respectively. κ is a constant for each pedestrian. Some work have used the pedestrian's height, which is a good indicator, to estimate κ. But it does not take into account individual walk pattern. If higher accuracy is required, κ is able to determined by a simple training. The pedestrian can be asked to walk for a known distance. The summed up steps' length should equal to the known distance. While κ is the only variable in the equation, it can be solved easily.

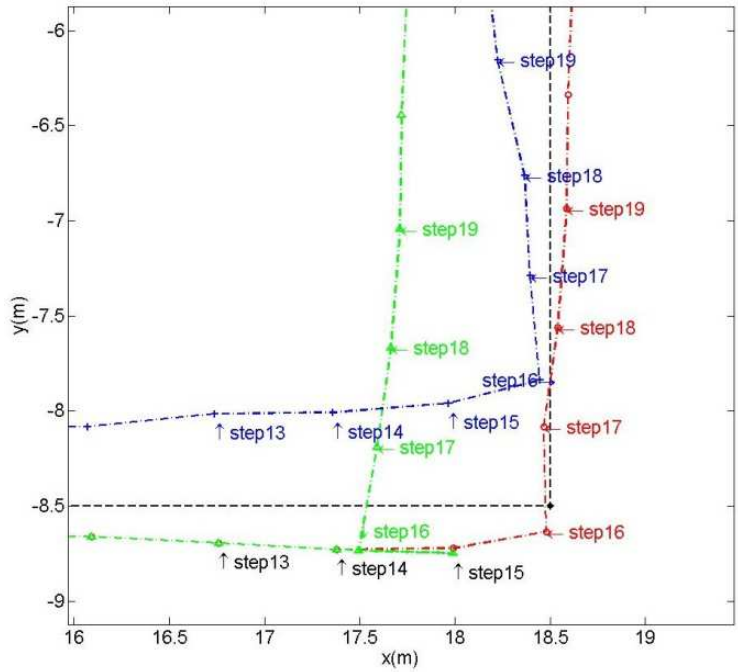
Data for 16 trials of the same route have been collected. An exemplary localization result of one trial is given in Fig. 3.7. The x-y axes represent the 2-

D location. The actual route we have chosen for the experiments and the estimated routes are represented using different lines in Fig. 3.7 (a). A pedestrian is asked to walk along the arrow's direction when the sensor is carried in the pocket. The path starts from the lower left corner and finally returns to the starting point. Because of the sensor's relative movement, the pure sensor's orientation analysis algorithm shows poor results with significant drift from the true positions. It is illustrated in the figure that, in most cases, the adaptive method gets the same direction estimate compared to the PCA algorithm. But when an error is returned by the PCA algorithm, as in lower right corner in Fig. 3.7 (a), the adaptive method successfully removes the error. An amplified region is shown in Fig. 3.7 (b). There is an 180°-error in step-16's direction estimation for the PCA algorithm.

In the 10 trials, the sensor remains relatively steady when the pedestrian turns. During the non-turning phase, the sensor arbitrarily chooses to relatively move or not. To study the extreme scenario when the sensor is always moving (during both the turning and non-turning phases) and to understand how this interferes with the orientation analysis method, the other 6 trials of data are collected under this situation. The results illustrated in Tables 3-2 and 3-3; the data includes the average error and the variance of the errors. The estimation error is the angle in the radians (rad) between the estimated step direction and the actual direction.



a)



b)

Fig. 3.7: Example of localization results applying different direction estimation algorithms

Table 3-2: Performance when sensor is relatively static during turns

Trial #	**1**	**2**	**3**	**4**	**5**
PCA Error (rad) /Variance	0.056/ 0.0037	0.168/ 0.3034	0.119/ 0.0363	0.170/ 0.1601	0.117/ 0.1611
adaptive method Error (rad) / Variance	0.058/ 0.0036	0.073/ 0.0082	0.098/ 0.0150	0.119/ 0.0258	0.057/ 0.0024
Trial #	**6**	**7**	**8**	**9**	**10**
PCA Error (rad) / Variance	0.090/ 0.1566	0.182/ 0.2685	0.096/ 0.1481	0.110/ 0.1521	0.182/ 0.2884
adaptive method Error (rad) / Variance	0.055/ 0.0048	0.078/ 0.0141	0.058/ 0.0044	0.067/ 0.0058	0.080/ 0.0115

Table 3-3: Performance when sensor is relatively moving during turns

Trial #	**11**	**12**	**13**	**14**	**15**	**16**
PCA Error (rad) / Variance	0.134/ 0.0960	0.084/ 0.0056	0.072/ 0.0065	0.094/ 0.0179	0.078/ 0.0062	0.068/ 0.0022
adaptive method Error (rad) / Variance	0.089/ 0.0084	0.084/ 0.0056	0.073/ 0.0065	0.094/ 0.0179	0.089/ 0.0124	0.068/ 0.0022

It is shown in Table 3-2 that the adaptive method does improve the performance compared with the PCA based algorithm. The improvement arises from verifying the estimate during the detected turns using the sensor's orientation analysis. When the assumption for the sensor's orientation analysis does not hold, the adaptive method should not be applied, which then returns the results of the PCA based algorithm. This is validated by the data in Table 3-3. In the worst case, when sensor is relatively moving during the turns, the adaptive method achieves almost the same performance as the PCA based algorithm. Even when the orientation analysis is wrongly applied in Trial 15, the result is still as reliable as the one from the PCA based algorithm.

3.2. An Indoor Dead-Reckoning Algorithm with Map Matching

A step-counting method derives the current location by Eq. 3-37. If s_n and $\mathbf{d}_n$ can be accurately resolved from the sensor's measurement, it is intuitive to conclude that a reliable solution is proposed. But because of the error in the estimation, it is necessary to fuse the external information to achieve a better performance.

3.2.1. Particle Filtering and Map Matching

It is useful to have indoor map information to improve the location results. Previous works have applied indoor map information in map filtering techniques, which eliminates location estimates in impossible areas. However, the accumulating error of the step direction will still eventually lead to large uncertainty in the location estimate. MM will relieve this problem.

a. Particle Filtering

A PF is commonly implemented to apply the map filtering technique. PF implements a recursive Bayesian filter using the Sequential Monte-Carlo method. It is particularly useful in solving non-linear and non-Gaussian problems. A set of random samples with weights, called particles, are used to represent the posterior probability density of the state, which is given by:

$$p(\mathbf{x}_n \mid \mathbf{z}_{1:n}) \approx \sum_{i=1}^{N_s} \omega_n^i \delta(\mathbf{x}_n - \mathbf{x}_n^i) \qquad \textbf{(3-39)}$$

where $\mathbf{x}_n$ and $\mathbf{z}_{1:n}$ are the states and measurements, respectively, $\{\mathbf{x}_n^i, i=1,\ldots, N_s\}$ is a supporting particle at time n with the associated weight $\{\omega_n^i, i = 1,\ldots, N_s\}$. N_s is set to 100 in our algorithm.

The weight update equation is given by Eq. 3-40, where $g(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i, \mathbf{z}_n)$ is the importance density:

$$\omega_n^i \propto \omega_{n-1}^i \frac{p(\mathbf{z}_n \mid \mathbf{x}_n^i) p(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i)}{g(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i, \mathbf{z}_n)} \quad \textbf{(3-40)}$$

There are some research conducted in choosing the optimal or suboptimal importance density as in [86][88]. For the purpose of localization, the importance density is always chosen as the same as the prior function:

$$g(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i, \mathbf{z}_n) = p(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i) \quad \textbf{(3-41)}$$

So the weight update equation can be estimated by:

$$\omega_n^i \propto \omega_{n-1}^i p(\mathbf{z}_n \mid \mathbf{x}_n^i) \quad \textbf{(3-42)}$$

In our step-counting algorithm, the system update and measurement update are executed after each step. During the system update phase,

$$\mathbf{x}_n^i = \mathbf{x}_{n-1}^i + s_n \mathbf{d}_n \quad \textbf{(3-43)}$$

where: $\mathbf{x}_n^i$ and $\mathbf{x}_{n-1}^i$ are the 2D states after n steps and n-1 steps, respectively, and s_n and $\mathbf{d}_n$ are step-n's length and direction, respectively.

A rule for pedestrian walking is that a pedestrian cannot walk through walls or obstacles. Particles with this motion are seen as invalid, so the weight is updated according to the following equation:

$$\omega_n^i = \begin{cases} 0, & \text{crosing the wall or obstacle} \\ \omega_{n-1}^i, & \text{otherwise} \end{cases} \quad \textbf{(3-44)}$$

The weights are then normalized such that $\sum_i \omega_n^i = 1$.

Overall, the above equations constitute the implementation of a sequential importance sampling (SIS) PF for step counting localization. A common problem with the SIS PF is the degeneracy phenomenon. It can be expected from Eq. 3-44 that after a few iterations, an increasing number of particles would have zero weight. In the end, all but a few particles will have zero weight. The degeneracy phenomenon is unavoidable by using SIS PF.

A solution is proposed to reduce the effects of degeneracy by re-sampling whenever a significant degeneracy is observed. The re-sampling step would generate a new set of particles by sampling the original particles based on their weights. The particles with small weights would be eliminated. But this method would introduce another problem. The particles that have high weights are always sampled many times, and the particles with small weights are eliminated. This leads to a loss of location diversity as there would be many repeated particles. This problem is known as sample impoverishment.

In this chapter, we applied a method named sampling importance re-sampling (SIR) , which is proposed in [89]. It is a Monte Carlo method that can be applied to recursive Bayesian filtering problems. Compared with previous PFs, the samples of SIR are generated by adding in a process noise sample:

$$\mathbf{x}_n^i = \mathbf{x}_{n-1}^i + s_n \mathbf{d}_n + \mathbf{n}_n^i \quad \textbf{(3-45)}$$

where $\mathbf{n}_n^i$ is a random variable, which a Gaussian distribution is assumed.

Re-sampling is applied at every iteration, so we always have $\omega_n^i = 1/N_s$.

b. Map Matching

The step direction estimation error results in large localization errors. To reduce the direction estimation error, the estimate is calibrated when the user is walking in a corridor. Our proposed MM method works as follows.

A corridor has two possible directions. Suppose one direction of the corridor is $\mathbf{d}^c$, if the following conditions are met, the user is seen as walking along the corridor.

Condition 1: Suppose six consecutive steps are with directions $\{\mathbf{d}_{n-i}, i=0,1,\ldots5\}$, which are represented as angles $\{d_{angle,n-i}, i=0,1,\ldots5\}$. $var\{d_{angle,n-i}, i=0,1,\ldots5\}<threshold\_5$.

Condition 2: Suppose the corridor is represented as a line segment, c; the distances from the locations of the six steps to c are $\{L_{n-i}, i=0,1,\ldots5\}$, respectively, $\sum_{i=0}^{5} L_{n-i} / 6 < threshold\_6$

Condition 3: The average direction $\mathbf{d}_{avg} = \sum_{i=0}^{5} \mathbf{d}_{n-i} / \| \sum_{i=0}^{5} \mathbf{d}_{n-i} \|$ should meet $\|dot(\mathbf{d}_{avg}, \mathbf{d}^c)\| < threshold\_7$, where $dot(.)$ means the dot product.

Condition 1 ensures that the pedestrian walks straight, which is the typical pattern when people walk along a corridor. In our implementation, *threshold_5* is set as $(8^o)^2$ obtained by training when people walk in corridor. Condition 2 ensures that the track of the pedestrian is close to the corridor.

threshold_6 is set as 2 meters. If a higher *threshold_6* is used, there is higher chance that the people may actually walk within an area close to the corridor, but yet considered as in the corridor. Condition 3 is based on the fact that the error in the direction estimation is within a limited angle. So if the difference between the estimated direction and the corridor's direction is too different, the pedestrian may not be walking in the corridor. *threshold_7* is set as 30° in our experiments. The three conditions ensure that the person is walking along a corridor with high probability. Even in the worst case scenario, that the pedestrian is actually not, the resulting error is limited by the conditions. After that, the direction with 180° ambiguity is chosen as:

$$\mathbf{d}^c = \begin{cases} \mathbf{d}^c, & dot(\mathbf{d}_{avg}, \mathbf{d}^c) > 0 \\ -\mathbf{d}^c, & \text{otherwise} \end{cases} \tag{3-46}$$

The large error in the direction estimation occurs when the pedestrian is making a turn. Thus the estimates from the last turning step to the current step need to be calibrated. Maximally, the locations of previous 20 steps would be calibrated, to avoid over calibration. Also, the projections of the locations to the corridor c should be within the corridor ends. Suppose the locations from step n- N_t to step n meet the two requirements, and the projection of location $\mathbf{x}_{n-N_t}$ to corridor c is $\mathbf{x}_{c,n-N_t}$, then let $\mathbf{x}_{n-N_t} = \mathbf{x}_{c,n-N_t}$.

In a 2D space, a counter clockwise rotation of a vector through angle ψ along the x-axis is represented using the rotation matrix $\begin{pmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{pmatrix}$, and the rectified directions $\{\mathbf{d}'_{n-j}$, i=0,1,…N_t -1$\}$are solved using:

$$\begin{cases} \begin{pmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{pmatrix} \mathbf{d}_{avg} = \mathbf{d}^{c} \\ \begin{pmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{pmatrix} \mathbf{d}_{n-i} = \mathbf{d}'_{n-i}, i=0,1,\ldots N_t-1 \end{cases} \quad \textbf{(3-47)}$$

The locations are then calibrated by Eq. 3-48, using the rectified direction estimate:

$$\mathbf{x}_{n-i} = \mathbf{x}_{n-i-1} + s_{n-i}\mathbf{d}'_{n-i}, \; i=0,1,\ldots N_t\text{-}1 \quad \textbf{(3-48)}$$

The sensor's orientation estimate **q** is also calibrated. The counter clockwise rotation with angle ψ in 2D space is rotating along the z-axis for ψ in 3D space:

$$\begin{cases} \mathbf{q}_{\triangle} = \cos(\frac{\psi}{2})[0,0,0,1] + \sin(\frac{\psi}{2})[0,0,1,0] \\ \mathbf{q} = \mathbf{q}_{\triangle} \otimes \mathbf{q} \end{cases} \quad \textbf{(3-49)}$$

where $\otimes$ stands for quaternion multiplication.

Unlike PF being executed at every step, the MM algorithm is only triggered when a pedestrian is considered to be walking along a certain corridor. Our simulation illustrates that to return the tracking results for the same route, the MM based on the step-counting algorithm takes 1/6 CPU time, as compared with the one using PF (100-particles).

3.2.2. Experimental Evaluation

In order to perform the experimental study, our laboratory was chosen as the indoor tracking testbed. The layout of the laboratory is presented in Fig. 3.8. The size of the testbed is approximately 22m by 18m. The coordinates system and the coordinates for some locations are also indicated. The map contains obstacles, walls and accessible areas. A corridor is an accessible area with a small width. In our experiment we define four corridors in this testbed as

illustrated in Fig. 3.8. The corresponding walls constituting the corridors are also indicated.

For evaluation purpose, [90] collected 1 trial of data during a 10-min walk. [91] collected 3 trials of data with around 10 mins for each trial. [92] collected 10 trials of data with each trial is 90 meters. [93] has tested the algorithm in shopping mall and residential building. Each test used one trial of data. In [94], method was tested in ten different locations. In our experiments, two routes are chosen for the experiment (85 meters each). For each route, 10 trials of data are collected when a pedestrian walks along the route with an IMU sensor. Whenever a certain point in the walking path is being passed, the timestamp of the passing is saved. There are 31 pieces of ground truth with timestamps for Routes 1 and 2. The sensor we use is ADIS16405BMLZ [95]. The routes are shown in Fig. 3.9, which include both the primary corridor and the path towards the seating area. The obstacles and walls are simplified as rectangles and lines. Although Route 1 and Route 2 cover the same indoor environment, the difference in walking directions results in a different calibration timing and effect by PF or MM, which would return different localization results. An example with more detailed explanation is provided in Fig. 3.10.

It is worth evaluating the performance given incomplete map information. In order to make the following statement simple, Table 3-4 illustrates the corresponding expressions when different parts of the map information are provided.

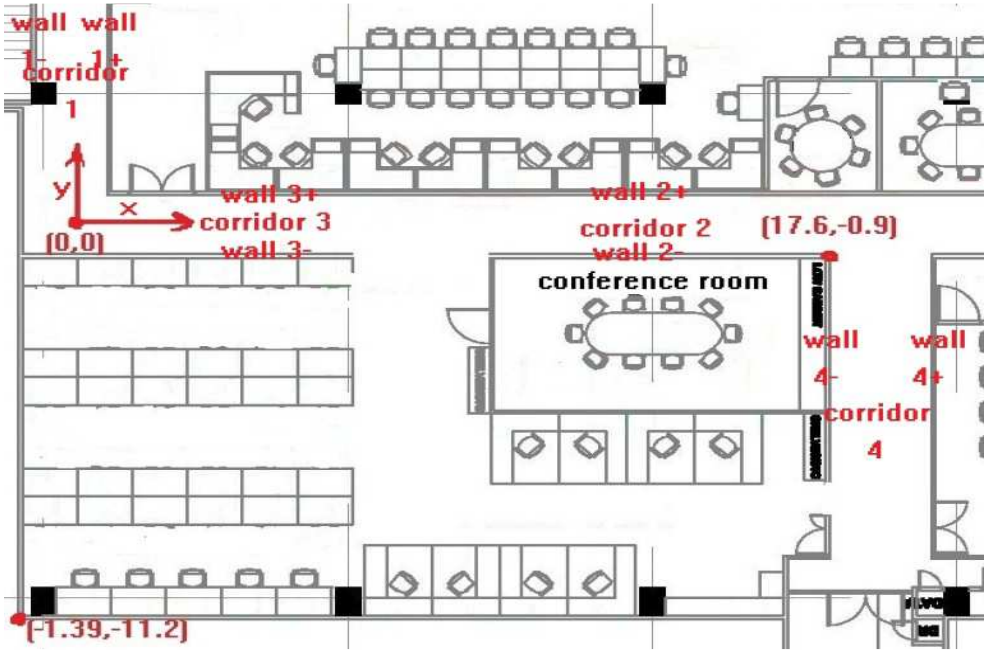
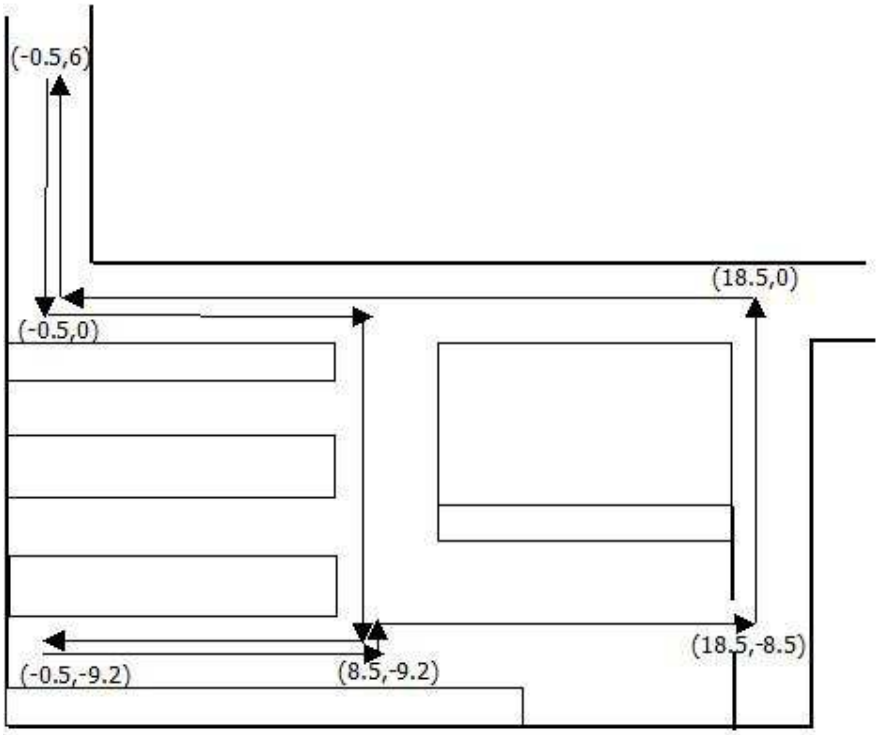
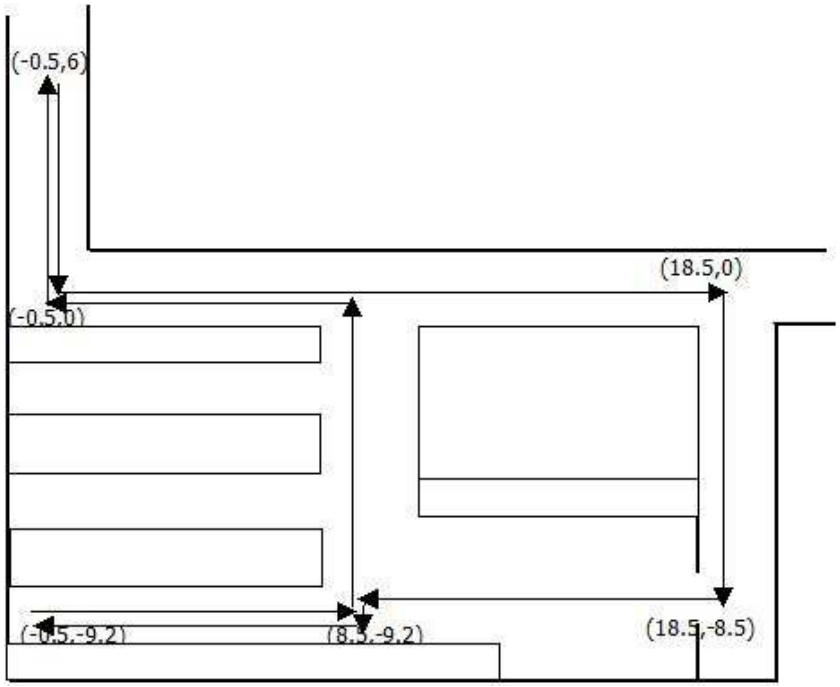


Fig. 3.8: Experimental testbed with walls and obstacles indicated



a) Route 1



b) Route 2

Fig. 3.9: Two routes being used in the evaluation

Table 3-4: Expressions When Given Different Parts of a Map

Map information being used	**Expression used in the following content**	
	PF	***MM***
All map information	PF-all	/
Walls 1- to 4- and walls 1+ to 4+ /Corridor 1 2 3 4	PF-4corridors (PF-4s)	MM-4corridors (MM-4s)
Wall 1- and wall 1+ /Corridor 1	PF-1 (PF-1)	MM -1 (MM-1)
Wall 2- and wall 2+ /Corridor 2	PF-2 (PF-2)	MM -2 (MM-2)

Fig. 3.10 shows an example of the tracked results when different algorithms are applied for Routes 1 and 2. It is illustrated that the fused PF and MM algorithm does improve the performance of the direct step-counting algorithm. If we compare the paths obtained by PF and MM, the one from MM coincides better with the ground truth, especially in the area without the map

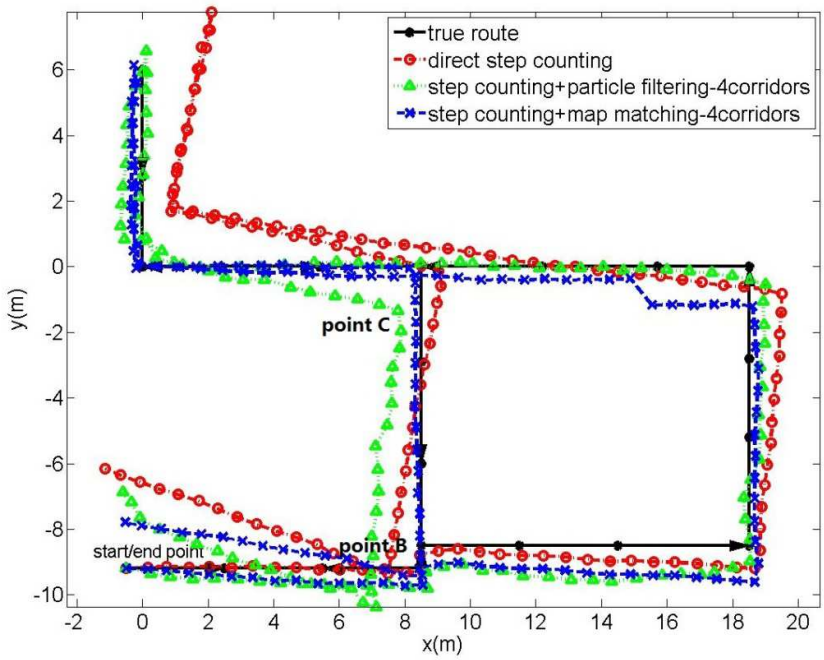
information. That is because the PF only rectifies the current location, the error in the sensor's orientation estimation and the step direction estimation remain in future calculations. The MM algorithm, in contrast, calibrates the location as well as the sensor's orientation and step direction. Hence, the error in the location update equation (Eq. 3-37) is reduced.

If we compare the tracked results from Route 1 and Route 2, a performance difference can be observed in the PF algorithm, when the pedestrian is walking back towards the start/endpoint in the last segment of the route. The tracked results in Route 2 are misled to a neighbouring corner, compared with the actual endpoint, which are separated by the seating area in the map. That is because when the pedestrian walks from Point A(18.5,-8.5) towards Point B(8.5,-8.5) in Route 2, even a 10° error in the direction estimation would result 1.74 ($10*sin(10^o)$) meter error in the location estimation in the y-axis. The error in the x-axis is quite small, which is 0.15 ($10-10*cos(10^o)$) meters. Fig. 3.11 illustrates the scenario with more detail, while a) is for Route 1 and b) is for Route 2.

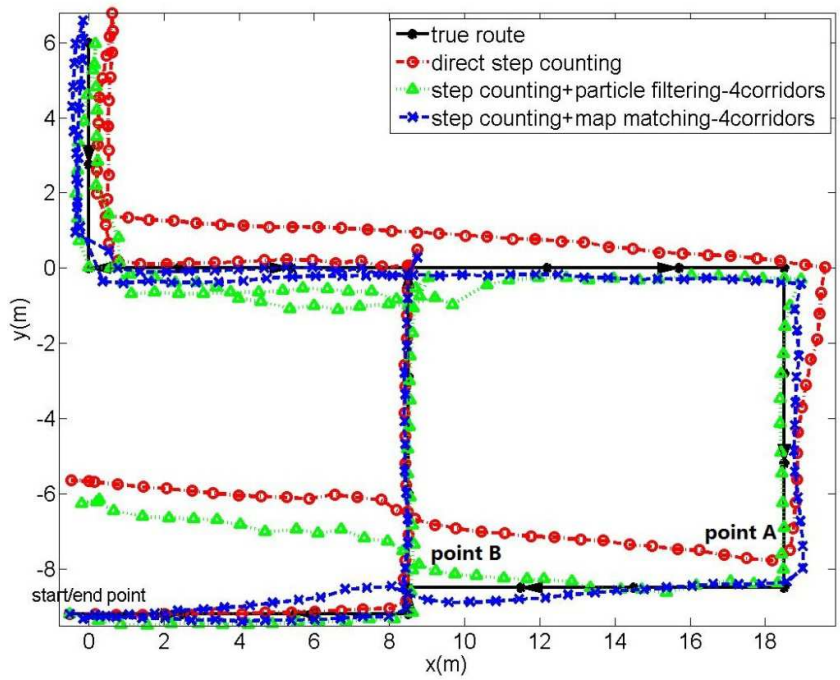
The lines and rectangles in the figure represent the walls and obstacles we applied for the PF. We explain the Route 2 scenario first (Fig. 3.11 b). The 100 small circles represent the particles with the centre at Point B, when there is no error in the location estimation. The '+' dots represent the particles when there is a 10° error in the direction estimation towards the right side of the walking direction. While the pedestrian continues the walk towards the $-x$ direction along the x-axis, the '+' dots (remaining particles) would move towards obstacle 1. Once the particles become close to Obstacle 1, it is

predictable that a number of particles would be filtered. From the figure we notice that there are quite a few particles facing area1, which is the actual path. Instead, the majority of the particles would be kept in Area 2. We can see this effect from Fig. 3.10 b). To make the situation even worse, the right side of the walking direction is more spacious than the left side. So the particles at the left side are more easily eliminated by map constraints. More particles at the right side increase the probability of misleading the tracked path to area2. This also explains why applying PF sometimes may have even worse results.

Route 1 is different because the direction estimation error would only result in location bias at the x-axis, which is illustrated in Fig. 3.11 a), where the small circles represent the particles without error and the small triangles represent the particles with such error. The pedestrian then walks towards the -x direction of the x-axis. The biased particles would result in a location estimation with bigger value in x coordinate. But the majority of the particles would still be close to the actual path.

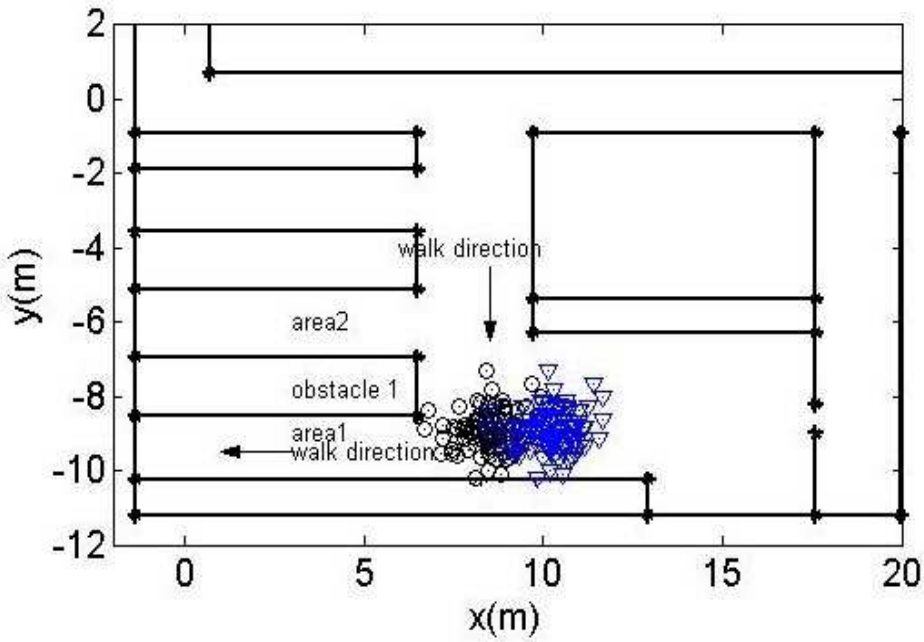


a) Route 1

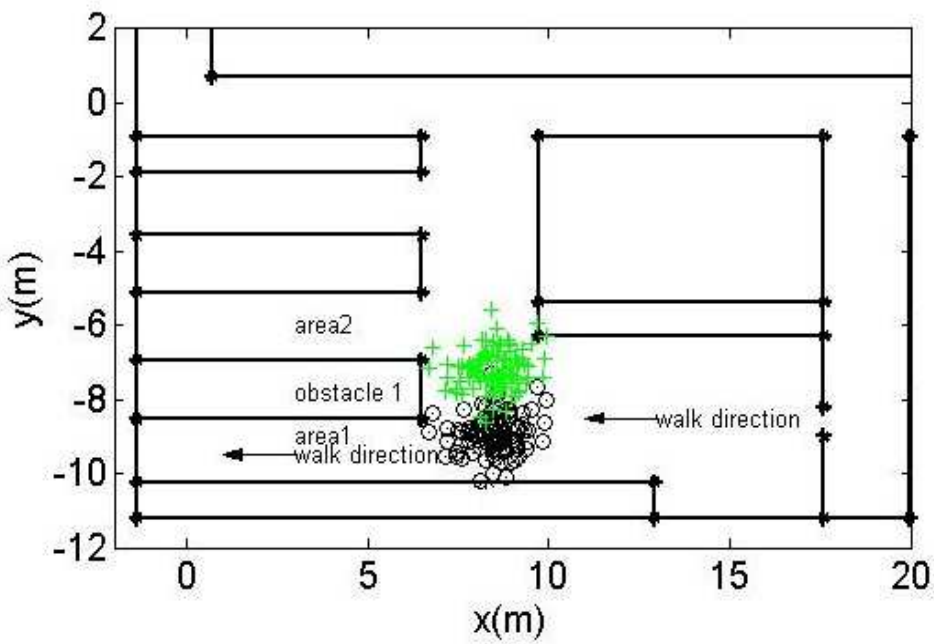


b) Route 2

Fig. 3.10: An example of the tracked results for Route 1 and 2 given different algorithms



a) Route 1



b) Route 2

Fig. 3.11: Particles at Point *B* when there is a 10° error in walking direction estimation for Route 1 and Route 2

Table 3-5: Average Error for Each Trial (Route 1)

	Average error (m)									
	1	2	3	4	5	6	7	8	9	10
Step counting	1.43	1.20	1.34	1.45	1.89	0.94	0.99	1.21	1.64	0.96
PF-1	1.89	1.76	2.14	1.45	1.16	0.87	0.93	1.31	1.06	1.42
MM-1	0.89	1.13	1.03	1.67	2.00	1.04	0.89	0.86	1.52	0.84
PF-2	1.90	1.69	1.97	1.71	1.87	1.12	1.44	1.87	2.10	1.64
MM-2	1.25	1.02	1.02	1.26	1.33	0.75	0.98	1.03	1.22	1.05
PF-4s	1.64	1.04	1.13	1.05	0.98	1.60	0.95	1.32	1.38	0.94
MM-4s	0.61	0.58	0.66	0.48	0.49	0.79	0.80	0.69	0.97	0.82
PF-all	1.04	0.82	0.77	0.86	0.60	1.46	0.73	0.89	1.09	0.68

Table 3-6: Average Error for Each Trial (Route 2)

	Average error (m)									
	1	2	3	4	5	6	7	8	9	10
Step counting	1.86	1.01	2.18	1.02	1.40	1.14	1.32	1.20	1.32	1.30
PF-1	2.50	2.46	2.54	1.50	1.89	1.82	1.45	1.96	1.65	1.90
MM-1	1.19	1.54	1.44	1.03	1.40	1.35	1.03	1.84	0.75	1.00
PF-2	2.06	0.99	2.38	1.76	1.44	1.31	2.14	1.93	1.54	1.84
MM-2	1.99	0.81	1.48	1.13	1.24	1.21	1.46	0.98	0.84	1.46
PF-4s	0.90	0.75	0.86	1.04	1.09	1.73	1.06	0.90	1.53	1.18
MM-4s	0.81	0.60	0.86	1.03	1.11	1.40	0.75	0.64	0.76	1.00
PF-all	0.94	0.84	0.94	0.93	0.94	1.57	0.81	0.91	1.01	1.00

Let the localization error be the distance between the ground truth and the estimated location. The average tracking error for the 10 trials for each route is shown in Tables 3-5 and 3-6, respectively. Even for the same route, the pure step-counting method for each trial returns a different accuracy, which results in a huge performance difference for the upper layer algorithms. When the

complete map information is available, both the PF and the MM improves the performance of the direct step-counting algorithm. Given the same map information, the MM returns more accurate results than the PF in most cases.

Therefore, it is worth noting that fusing map information does not always return the better results, especially for PF. The tracking may be misled to a drifted path by the error in the existing estimation, especially when there is incomplete map information. The results in the tables illustrate that PF-1 and PF-2 tend to return drifted results more often. The MM is more robust with incomplete map information, as compared with the PF.

3.3. Map Matching Enabled Particle Filter and Improved Particle filtering

3.3.1. Map Matching Enabled Particle Filter Methods

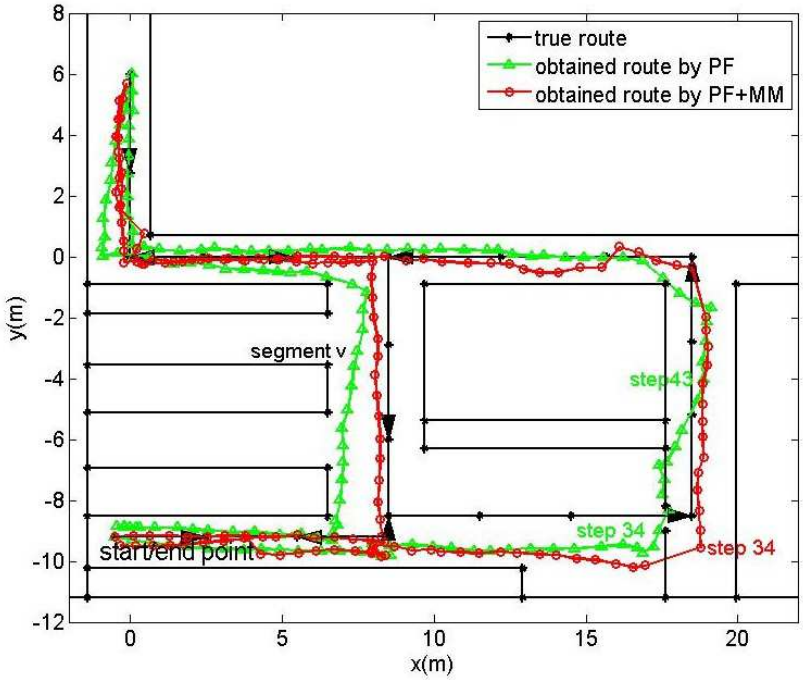
Although the MM algorithm provides reliable results with less CPU cost, the required narrow corridors may not always exist in an indoor environment. In addition, it does not make full use of the map information. Moreover, the administrator needs to manually state the coordinates and directions of selected corridors. Therefore, it is better to perform the MM as a supplementary algorithm to other techniques, rather than work as a primary technique.

It is quite natural to apply the MM on the PF based step-counting algorithm, which is called MM enabled PF in this thesis. Suppose the location of a turning step is $\mathbf{x}_{n-N_t}$ and its projection to corridor (represented as c) is $\mathbf{x}_{c,n-N_t}$, then the particles of step n-N_t ($\mathbf{x}^i_{n-N_t}$) are regenerated with the mean $\mathbf{x}_{c,n-N_t}$.

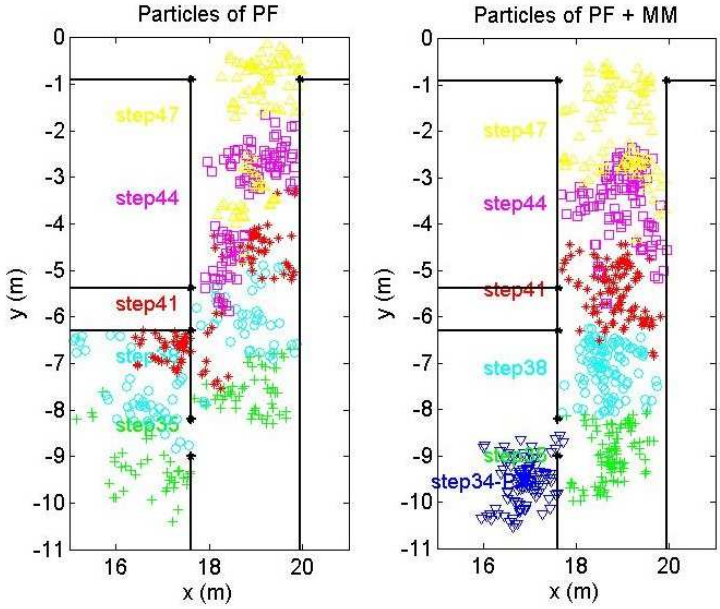
The particles for the subsequent steps are updated by Eq. 3-50 using the rectified direction estimation. The weight update rule remains the same.

$$\mathbf{x}_{n-j}^{i} = \mathbf{x}_{n-j-1}^{i} + s_{n-j}\mathbf{d}_{n-j}^{'} + \mathbf{n}_{n-j}^{i} \quad j=0,1,\ldots N_t\text{-}1 \qquad \textbf{(3-50)}$$

A performance comparison between the original PF and the MM enabled PF is shown in Fig. 3.12. In segment *v* shown in Fig. 3.12(a), there is a clear drift for the PF algorithm because of the error in the step direction estimation. A more detailed explanation of the process is shown in Fig. 3.12(b), where the particles at nearby steps are differentiated using different shapes. The particles for PF are sparsely distributed so that there are always particles in the actual location. Although erroneous particles are filtered during walking, there is a trend that the particles drift to the right-hand side. For the PF+MM algorithm, the MM is triggered in step 43 and the particles are recalculated from step 34. The particles at step 34 for the PF+MM algorithm before the execution of the MM algorithm are also represented (step34-P)). Once the MM algorithm is triggered, all the particles of step34-P are regenerated in the corridor. The particles at the subsequent steps are recalculated using the rectified direction estimation as shown in Fig. 3.12(b).



a) Tracked route



b) Particles for certain steps

Fig. 3.12: Performance comparison of PF and PF + MM

3.3.2. Improved Particle Filter

In this section, we propose an improved PF that models the localization error better than the original one and, thus returns more accurate results. From the previous discussion, it is known that the step direction estimation always suffers from the errors in the sensor's orientation estimation. Because of the uniqueness of the Earth's gravity, it becomes a reliable direction reference. From the previous discussion, accurate vertical and horizontal components of acceleration are decomposed using Eq. 3-7. With reliable vertical component of acceleration, accurate step length estimation is achieved using Eq. 3-38. In the previous section, the direction has been represented by a unit vector. In this section, we use the angle $d_{angle,n}$ to represent it. So the position update equation for PF can be written as

$$\begin{aligned} angle &= d_{angle,n} + b_{d,n}^{i} + n_{d,n}^{i} \\ \mathbf{x}_{n}^{i} &= \mathbf{x}_{n-1}^{i} + (s_n + n_{s,n}^{i})(\cos(angle); \sin(angle)) \end{aligned} \quad \textbf{(3-51)}$$

where $n_{s,n}^{i}$ and $n_{d,n}^{i}$ represent the noise in length and direction estimation, respectively. $b_{d,n}^{i}$ is to compensate for the direction angle estimation drift for particle *i* in step *n*, which will be explained later.

To further explain the algorithm, the performance of the pure step-counting algorithm shown in Fig. 3-10 need to be examined. It shows that when a pedestrian walks along a straight line, the pure step-counting algorithm provides quite a consistent direction estimation which can be seen as a line. Additionally, there is clear direction estimation error during the turn.

Because of the robustness of the length estimation and direction estimation during line-mode walk, $n^i_{s,n}$ and $n^i_{d,n}$ are set as Gaussian variables with small variances. Suppose initially the direction estimation is accurate, so that $b^i_{d,n}$ = 0. $b^i_{d,n}$ is updated as follows

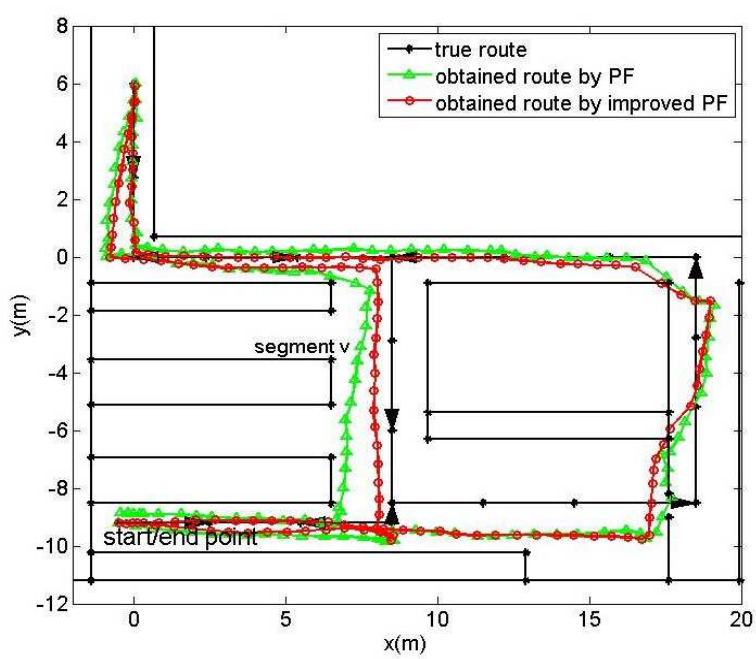
$$b^i_{d,n} = \begin{cases} b^i_{d,n-1} + n^i_{d,n-1} + N(0, var), & \text{a turn is detected in step } n \\ b^i_{d,n-1} + n^i_{d,n-1}, & \text{otherwise} \end{cases} \quad \textbf{(3-52)}$$

where *N(0,variance)* is a Gaussian variable with larger variance. The particle weights are updated also using Eq. 3-44, and then go through the re-sample process.

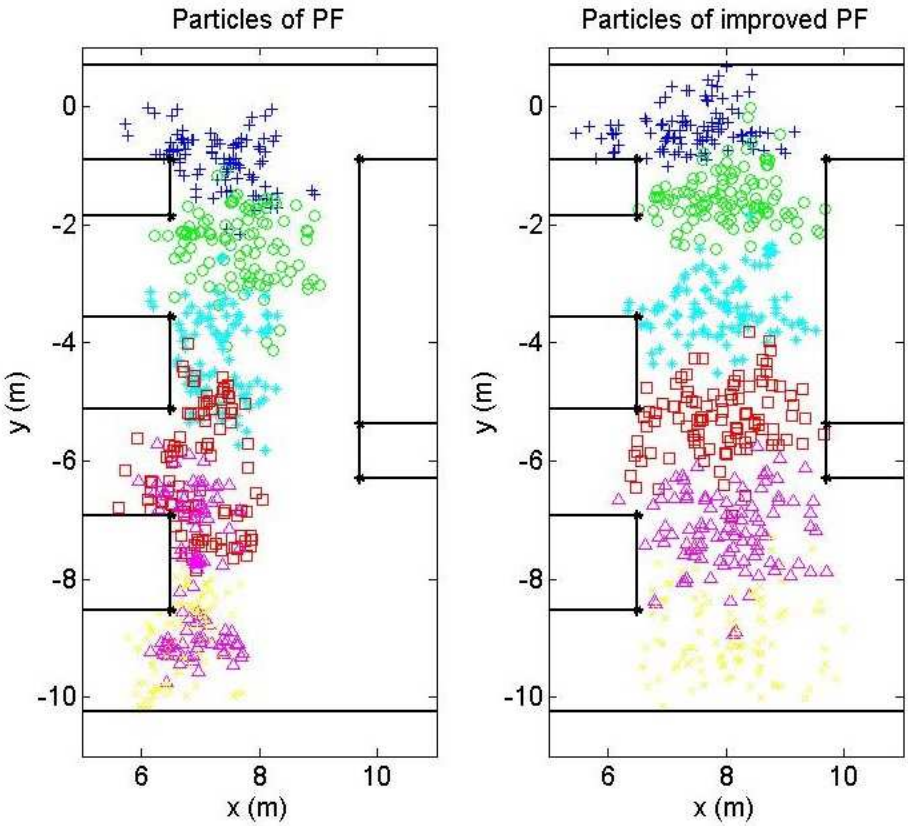
For the original PF, the estimated direction always equals $d_{angle,n}$. Although impossible paths can be eliminated by the accessibility constraints, the error in direction estimation accumulates. In the end, the model in Eq. 3-45 becomes unsuitable for describing the walking process. The advantage of the improved PF is that it incorporates the uncertainty in direction estimation, which is represented as $b^i_{d,n}$ and $n^i_{d,n}$. The choice of its value is further refined by the pattern that large uncertainty occurs during turns. In this way this model describes the pedestrian walk better than the previous one. Impossible paths as well as erroneous directions can be eliminated in this algorithm.

A performance comparison between the previous original PF and our improved PF is shown in Fig. 3.13. There is not much difference in performance when the pedestrian is walking along the corridors. However, in segment *v* the original PF shows a clear drifted path as shown in Fig. 3.13(a). This drift comes from the error in direction estimation. Our improved PF, on

the other hand, fits the true route quite well as the drift is compensated in the model. The particles with erroneous drift estimates are eliminated as they pass the walls or obstacles. Detailed observation in segment v can be found in Fig. 3.13(b) where the particles are drawn. The particles are drawn every three steps so that the particles from different steps would be easy to distinguish. The particles at nearby steps are differentiated by different shapes. A clear drift trend can be found for the particles from the original PF algorithm compared with our improved one. It does not learn from previous filtering results. The improved PF, on the other hand, always eliminates the erroneous $b^{i}_{d,n}$ estimation in the direction estimation and retains the correct ones. Compared with the MM algorithm, the improved PF rectifies the direction estimation without specifically defining the corridors. Its CPU cost is the same as the original PF, which is less than the MM enabled PF algorithm.



a) Tracked route



b) Particles for certain steps

Fig. 3.13: Performance comparison of PF and improved PF

3.3.3. Evaluation

The data is collected in the same manner as described in the previous section. But the map constraints are redefined as in Fig. 3.14. There are three corridors available in total in this map for the MM algorithm. Later the map will be represented as simple lines and rectangles.

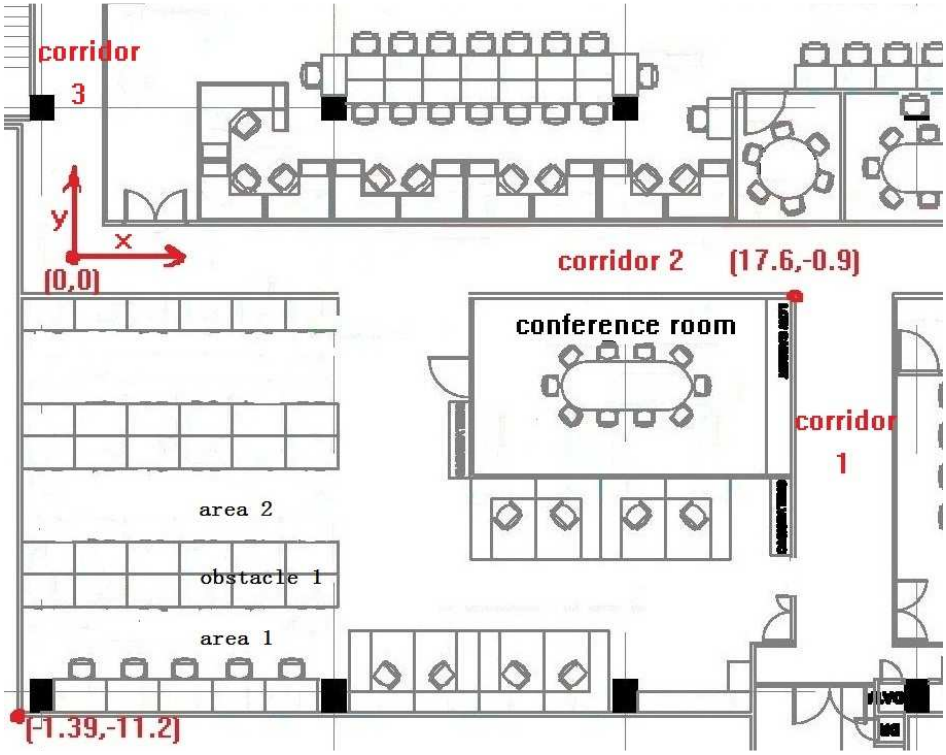


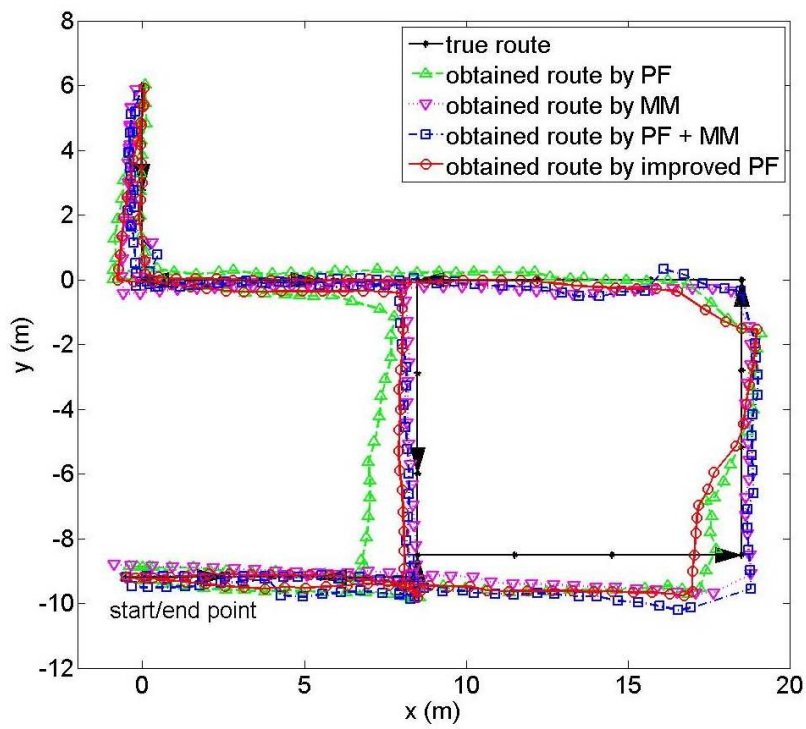
Fig. 3.14: Experimental testbed with walls and obstacles

a. Performance on Full Map Information

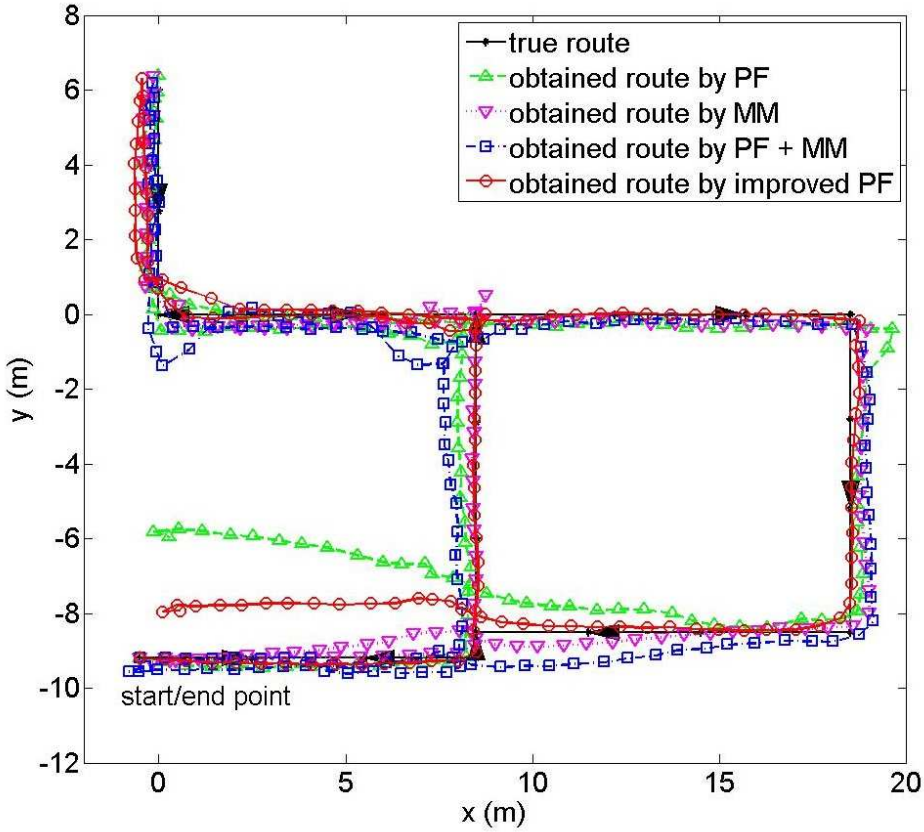
Fig. 3.15 shows an example of the tracking results when different algorithms are applied for routes 1and 2. In both routes, the start and end points are kept the same, the difference is that the direction in Route 1 is counter clockwise while the one in Route 2 is clockwise. The indoor map constraints are kept the same as in the previous example. It is shown that the fused PF and MM algorithms do improve the performance of the direct step-counting algorithm. If we compare the paths obtained by the original PF and the MM schemes, it can be found that the one from the MM coincides better with the ground truth, especially in the area right after the MM gets executed. That is because the PF only rectifies the current location, the error in the

sensor's orientation estimation and step direction estimation remains in future calculations. The MM algorithm, in contrast, calibrates the location as well as the sensor's orientation and step direction. The error in the location update equation is reduced. Our improved PF overcomes the drawback by modelling the error in direction estimation.

It is worth noting that the improved PF also encounters the similar misdirecting effect as what we explained in Fig. 3.11 for Route 2. But because the error in walking direction estimation is reduced during the filtering, the tracked path would be less misled towards area 2 in the end. That is why we can see the path ends in between the one from original PF and the actual path.



a) Route 1



b) Route 2

Fig. 3.15: An example of the tracked results for Routes 1 and 2 given different algorithms

Table 3-7: Average (Avg) Error for Each Trial (Route 1)

	Average error (m)										
	1	2	3	4	5	6	7	8	9	10	Avg
Step counting	1.31	1.16	1.22	1.63	2.02	0.90	0.97	1.71	0.86	2.03	1.38
PF	1.04	0.82	0.77	0.86	0.60	1.46	0.73	1.09	0.68	0.72	0.88
MM	0.49	0.59	0.62	0.44	0.41	0.65	0.70	0.83	0.69	0.60	0.60
PF+MM	0.68	0.67	0.61	0.68	0.68	0.75	0.60	0.86	0.88	0.77	0.72
Improved PF	0.62	0.41	0.56	0.47	0.58	0.56	0.39	0.75	0.51	0.68	0.55

Table 3-8: Average Error for Each Trial (Route 2)

	Average error (m)										
	1	2	3	4	5	6	7	8	9	10	Avg
Step counting	1.71	1.10	0.95	1.47	1.15	1.11	1.36	1.32	1.25	1.56	1.30
PF	0.94	0.84	0.93	0.94	1.57	0.81	0.91	1.01	1.00	1.06	1.00
MM	0.75	0.58	0.86	1.09	1.52	0.65	0.61	0.73	1.01	0.85	0.87
PF+MM	0.48	0.72	0.78	1.14	1.27	0.50	0.58	0.72	0.86	0.75	0.78
Improved PF	0.97	0.68	0.74	1.15	1.37	0.75	0.84	0.73	0.97	1.12	0.93

Let the localization error be the distance between ground truth and estimated location. The average tracking errors for the 10 trials for each route are shown in Table 3-7 and Table 3-8, respectively. The last column shows the average error for the 10 trials. MM returns more accurate results than PF in the 17 trials of the total 20 trials. In the rest of the 3 trials, the performances are quite close (0.68/0.69, 0.94/1.09, and 1.00/1.01). Therefore, the MM outperforms the PF in the given map environment. A similar conclusion can be made when we compare the accuracy of MM enabled PF with PF, and the accuracy of our improved PF with PF. PF provides the least accuracy among the three, which is only better than pure step-counting without map constraints. When we compare the average error from MM, MM enhanced PF and improved PF, we see quite close location accuracy.

It is interesting to find that the MM enabled PF improves the performance in Route 2, but not in Route 1. That is because there are many corridors in our map for MM to take effect, and the paths are along the corridors. MM alone is sufficient to obtain good results. When the distribution of actual location is represented by particles, the particles should be distributed sparsely enough to

avoid execution failure when all particles get removed by the map constraints. In our lab environment, an empirical size of the particle distribution would be a quasi-circle with radius of around 1.5 meters. The computed centre from the distribution may not be as close to the actual location as MM, especially in corridors.

The improved PF performs poorer than MM and PF+MM in Route 2 because of the misdirecting effect as what we explained in Fig. 3.11. MM is triggered only when the pedestrian is in the corridor, which wastes the other map constraints information. MM enabled PF is quite a natural method to keep the advantages of both algorithms. Our improved PF takes advantage of PF and improves direction estimation at the same time. In the next section when less corridor information is given, we will see the improvements of such algorithms.

b. Performance on Incomplete Map Information

The previous evaluations have been carried out in an ideal corridor environment like our lab. It is necessary to test the performance in another scenario. Here, the new scenario is virtually created by giving incomplete map information to the algorithms. From the algorithms' perspective, it makes no difference compared to collecting the data in a physical new environment and using relative map constraints. Fig. 3.16 shows the incomplete map we use to evaluate the performance for the same routes in Section a). Map 1 keeps the map constraints in top left corner, with one wall of a horizontal corridor. Map 2 keeps the map constraints in the bottom right corner, with the conference room.

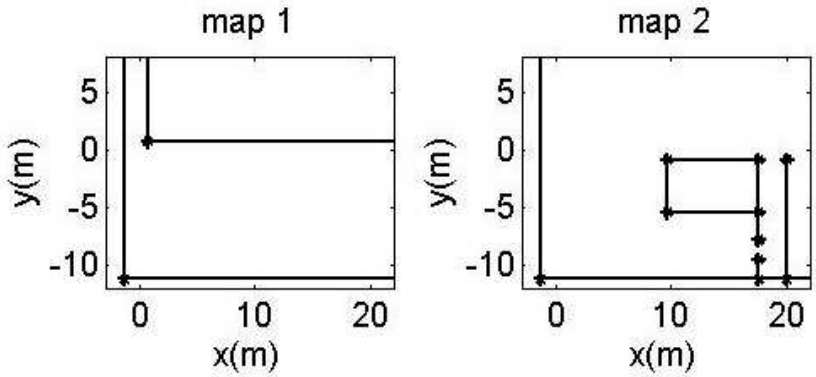


Fig. 3.16: The used incomplete map

Table 3-9: Average Error for Each Trial (Map 1, Route 1)

	Average error (m)									
	1	2	3	4	5	6	7	8	9	10
Step counting	1.31	1.16	1.22	1.63	2.02	0.90	0.97	1.71	0.86	2.03
PF	1.78	1.28	1.54	1.46	1.09	1.22	1.00	1.30	1.68	1.21
MM	0.96	1.22	1.07	1.90	2.17	1.08	0.92	1.64	0.90	1.80
PF+MM	1.65	1.22	1.42	1.20	0.86	1.38	1.38	1.24	1.28	1.24
Improved PF	1.25	1.02	1.27	1.07	0.99	1.07	1.12	0.99	1.00	1.14

Table 3-10: Average Error for Each Trial (Map 1, Route 2)

	Average error (m)									
	1	2	3	4	5	6	7	8	9	10
Step counting	1.71	1.10	0.95	1.47	1.15	1.11	1.36	1.32	1.25	1.56
PF	1.90	1.06	1.54	1.41	1.56	1.70	1.24	1.78	1.61	1.75
MM	1.03	1.65	1.05	1.47	1.36	0.86	2.01	0.75	0.99	0.99
PF+MM	2.00	0.81	1.62	1.47	1.63	1.74	0.83	1.61	1.83	1.79
Improved PF	1.62	0.77	1.21	1.07	1.28	1.52	0.86	1.06	1.27	1.37

Table 3-11: Average Error for Each Trial (Map 2, Route 1)

	Average error (m)									
	1	2	3	4	5	6	7	8	9	10
Step counting	1.31	1.16	1.22	1.63	2.02	0.90	0.97	1.71	0.86	2.03
PF	1.75	1.48	1.64	2.08	1.74	1.19	1.43	1.93	1.55	1.90
MM	2.32	0.92	1.80	0.55	1.09	0.84	0.97	1.10	1.91	0.95
PF+MM	1.80	1.01	1.36	0.89	0.87	1.05	1.17	1.13	1.62	1.05
Improved PF	1.57	0.70	0.92	1.09	1.64	1.07	1.24	1.52	1.15	1.38

Table 3-12: Average Error for Each Trial (Map 2, Route 2)

	Average error (m)									
	1	2	3	4	5	6	7	8	9	10
Step counting	1.71	1.10	0.95	1.47	1.15	1.11	1.36	1.32	1.25	1.56
PF	1.38	1.29	0.88	1.45	1.48	1.21	1.03	1.37	1.21	1.26
MM	1.41	0.77	1.03	1.29	1.31	0.97	1.01	1.43	1.14	1.84
PF+MM	1.41	0.65	1.04	0.99	1.54	1.44	0.87	1.35	1.10	1.67
Improved PF	1.58	1.26	0.91	1.49	1.43	1.19	1.21	0.89	1.14	1.31

Table 3-13: Average Error for Each Map and The Overall

Average Over	Error for Applied Methods (m)				
	Step counting	PF	MM	PF+MM	Improved PF
Map 1	1.34	1.46	1.29	1.41	1.15
Map 2	1.34	1.46	1.23	1.20	1.23
Overall	1.34	1.46	1.26	1.31	1.19

The average accuracy for each of the trials with respect to the route and map are presented in Table 3-9 to Table 3-12. Table 3-13 shows the average error for map 1 and map 2, and the overall average. We first notice that PF no longer provides better results than pure step-counting algorithm. It is because

in the settings with few map constraints, the particles are widely spread out without calibration. The tracking may have been misled to a drifted path if the pedestrian encounters an obstacle in that case.

There is slight enhancement for MM and MM enhanced PF, compared with step-counting. Although from the perspective of average value, MM enhanced PF performs worse than MM (1.31m to 1.26m), we find that MM enhanced PF actually provides more robust results. MM returns large errors in trial 5 in table 3-9 (2.17m), trial 7 in table 3-10 (2.01m), and trial 1 in table 3-11 (2.32m), while MM enhanced PF dramatically reduces the error (0.86, 0.83 and 1.80, respectively). The proposed improved PF achieves the most accurate result among all the algorithms with the same computational complexity as the original PF. Considering there is no additional requirement on the manually provided corridor information; improved PF is the most robust solution in various map scenarios.

Chapter 4

Dual Sensor Fusion

In this chapter, we introduce a dual sensor fusion method. The primary contribution of this chapter has been the fusing of the orientation estimates from the two sensors, which has not been discussed in the literature. The orientation has been represented by the quaternion in this thesis. We consider the orientation fusion problem as the fusion of two 4-dimensional unit vectors; the angle of these two vectors can be determined by the quaternion computation. We test the performance by firmly sticking Sensor *A* and Sensor *B*, close to each other, and capture the real orientation by an optical motion capturing system called the Osprey Digital Real Time System (ODRTS). The experimental results show that the fused solution achieves better orientation accuracy and performance robustness. We further apply the orientation fusion method into the DR tracking application. It also illustrates higher localization accuracy, as compared with individual sensor DR. When we compare it with

the existing dual fusion method on location, the proposed method achieves the same accuracy.

4.1. Motivations

In Chapter 3, we have described our methods in applying the DR for indoor localization purposes, which are based on the measurements from a single sensor. We observed that quite accurate step length estimation can be obtained; the error in step direction estimation introduces large localization error. Hence, higher accuracy may be achieved by fusion of the results from the two sensors. The authors in [92] have done some work in dual sensor fusion on location while a pedestrian holds two devices at the same time.

The hand-held devices equipped with accelerometers, a gyroscope and magnetometers are quite commonly available in the market. Recent developments in miniature sensor design have made the IMU sensor smaller, cheaper, and more accurate, with less power consumption. If a strong use case is given, it can be foreseen that the hand-held device manufacturers are able to incorporate two 9-axis IMU sensors in a single device. Therefore, we propose the localization scenario when two sensors are firmly attached close to each other, which is to simulate the two sensors in one device scenario.

4.2. Problem Definition

The authors in [92] proposed an a posteriori method to fuse the DR location results from the two IMU sensors, while the sensors maintain a constant distance. In the scenario we propose, the same method can be applied while

the constant distance is set as zero. However, the major contribution of this chapter is the fusion method of the orientation estimation from the two sensors.

In Section 3.2.1, we introduced the algorithm for a single sensor's orientation estimation. Suppose there are two Sensor *A* and Sensor *B*, being attached firmly close to each other. At time 0, Sensor *A* and Sensor *B* are with quaternion rotations $\mathbf{q}_{A,0}$ and $\mathbf{q}_{B,0}$, respectively. The orientation shift $\mathbf{q}_{shift}$ is given by

$$\mathbf{q}_{shift} = \mathbf{q}_{A,0}^{-1} \otimes \mathbf{q}_{B,0} \tag{4-1}$$

where $\mathbf{q}^{-1}$ represents the quaternion inverse in this thesis.

At any time *t*, the same relationship is maintained because Sensor *A* and Sensor *B* are firmly attached close to each other:

$$\mathbf{q}_{shift} = \mathbf{q}_{A,t}^{-1} \otimes \mathbf{q}_{B,t} \tag{4-2}$$

Suppose the measured orientations for Sensor *A* and Sensor *B* at time *t* are $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{B,t}$, respectively. Suppose

$$\hat{\mathbf{q}}_{A,t}^{B} = \hat{\mathbf{q}}_{B,t} \otimes \mathbf{q}_{shift}^{-1} \tag{4-3}$$

If both orientation measurements of the two sensors are accurate, we should have $\hat{\mathbf{q}}_{A,t}$ equals $\hat{\mathbf{q}}_{A,t}^{B}$ because the sensors are firmly attached close to each other.

To fuse the two measurements, we need to solve the following problem:

$$\text{maximize } f(\mathbf{q}_{A,t}, \mathbf{q}_{B,t} \mid \hat{\mathbf{q}}_{A,t}, \hat{\mathbf{q}}_{B,t}) \tag{4-4}$$

, which then transforms to Eq.4-5 according to Bayes' Theorem:

$$\text{maximize } \frac{f(\hat{\mathbf{q}}_{A,t}, \hat{\mathbf{q}}_{B,t} \mid \mathbf{q}_{A,t}, \mathbf{q}_{B,t}) f(\mathbf{q}_{A,t}, \mathbf{q}_{B,t})}{f(\hat{\mathbf{q}}_{A,t}, \hat{\mathbf{q}}_{B,t})} \tag{4-5}$$

Because the two sensors measure the orientation independently, thus:

$$f(\hat{\mathbf{q}}_{A,t}, \hat{\mathbf{q}}_{B,t} \mid \mathbf{q}_{A,t}, \mathbf{q}_{B,t}) = f(\hat{\mathbf{q}}_{A,t} \mid \mathbf{q}_{A,t}) f(\hat{\mathbf{q}}_{B,t} \mid \mathbf{q}_{B,t}) \quad \textbf{(4-6)}$$

$f(\mathbf{q}_{A,t}, \mathbf{q}_{B,t})$ and $f(\hat{\mathbf{q}}_{A,t}, \hat{\mathbf{q}}_{B,t})$ are not affected by the values of $\mathbf{q}_{A,0}$ and $\mathbf{q}_{B,0}$, thus Eq. 4-5 becomes:

$$\text{maximize } f(\hat{\mathbf{q}}_{A,t} \mid \mathbf{q}_{A,t}) f(\hat{\mathbf{q}}_{B,t} \mid \mathbf{q}_{B,t}) \quad \textbf{(4-7)}$$

and:

$$f(\hat{\mathbf{q}}_{B,t} \mid \mathbf{q}_{B,t}) = f(\hat{\mathbf{q}}_{B,t} \mid \mathbf{q}_{A,t}\mathbf{q}_{shift}) = f(\hat{\mathbf{q}}_{B,t}\mathbf{q}_{shift}^{-1} \mid \mathbf{q}_{A,t}) = f(\hat{\mathbf{q}}_{A,t}^{B} \mid \mathbf{q}_{A,t}) \quad \textbf{(4-8)}$$

Thus the problem is further transformed into Eq. 4-9:

$$\text{maximize } f(\hat{\mathbf{q}}_{A,t} \mid \mathbf{q}_{A,t}) f(\hat{\mathbf{q}}_{A,t}^{B} \mid \mathbf{q}_{A,t}) \quad \textbf{(4-9)}$$

4.3. Maximum A Posteriori Fusion

Take $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$ as unit vectors in a 4-dimensional space. The error is the angle between the vector and $\mathbf{q}_{A,t}$. The noise in $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$ are modelled as a Gaussian distribution with $N(0, sigma\_1^2)$ and $N(0, sigma\_2^2)$, respectively.

Theorem 1: The objective function 4-9 only becomes maximized when the $\mathbf{q}_{A,t}$ is within the hyperplane determined by $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$.

Theorem 1 Proof:

In Fig. 4.1 a), there are two observations: $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$. Suppose the solution is $\mathbf{q}_{A,t}^{'}$, which is out of the hyperplane *hyper_m* determined by $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$. The angles between the solution $\mathbf{q}_{A,t}^{'}$ to $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$ are $\alpha^{'}$ and $\beta^{'}$, respectively, then

$$f(\hat{\mathbf{q}}_{A,t} \mid \mathbf{q}_{A,t}) f(\hat{\mathbf{q}}^{B}_{A,t} \mid \mathbf{q}_{A,t}) = \frac{1}{sigma\_1}\chi(\frac{\alpha^{'}}{sigma\_1})\frac{1}{sigma\_2}\chi(\frac{\beta^{'}}{sigma\_2}) \quad \textbf{(4-10)}$$

where $\chi(x)$ equals:

$$\chi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \quad \textbf{(4-11)}$$

Through $\mathbf{q}^{'}_{A,t}$, a hyperplane perpendicular *hyper_m* can be identified, and the unit component of the intersection is marked as $\mathbf{q}^{''}_{A,t}$. The angles between $\mathbf{q}^{''}_{A,t}$ to $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}^{B}_{A,t}$ are α and β, respectively. From geometry, $\alpha < \alpha^{'}$ and $\beta < \beta^{'}$, as such we have:

$$\chi(\frac{\alpha^{'}}{sigma\_1})\chi(\frac{\beta^{'}}{sigma\_2}) < \chi(\frac{\alpha}{sigma\_1})\chi(\frac{\beta}{sigma\_2}) \quad \textbf{(4-12)}$$

, which means $\mathbf{q}^{'}_{A,t}$ is not the solution. Therefore, Theorem 1 is proven.

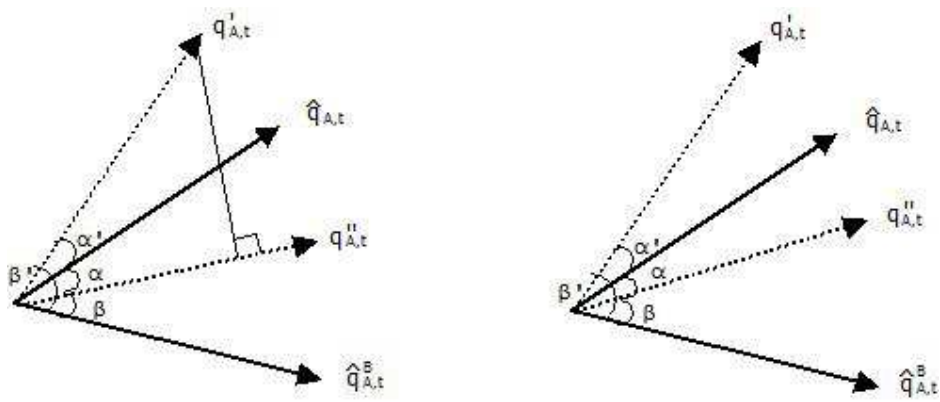
Theorem 2: The objective function 4-9 only becomes maximized when the $q_{A,t}$ is in between $\hat{q}_{A,t}$ and $\hat{q}^{B}_{A,t}$.

Theorem 2 Proof:

In Fig. 4.1 b), the vectors are in the same plane *hyper_m.* $\mathbf{q}^{'}_{A,t}$ is supposed to be the solution which is not in between $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}^{B}_{A,t}$, and is closer to $\hat{\mathbf{q}}_{A,t}$. The angles between the solution $\mathbf{q}^{'}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}^{B}_{A,t}$ are $\alpha^{'}$ and $\beta^{'}$, respectively. $\mathbf{q}^{''}_{A,t}$ can be drawn with the angle $\alpha = \alpha^{'}$, but opposite $\hat{\mathbf{q}}_{A,t}$. The angle between $\mathbf{q}^{''}_{A,t}$ and $\hat{\mathbf{q}}^{B}_{A,t}$ is β, which is less than $\beta^{'}$:

$$\chi(\frac{\alpha^{'}}{sigma\_1})\chi(\frac{\beta^{'}}{sigma\_2}) < \chi(\frac{\alpha^{'}}{sigma\_1})\chi(\frac{\beta}{sigma\_2}) \quad \textbf{(4-13)}$$

As such, $\mathbf{q}'_{A,t}$ is not the solution. When $\mathbf{q}'_{A,t}$ is closer to $\hat{\mathbf{q}}^{B}_{A,t}$, the same result holds. Thus, we find that the maximum likelihood solution is in between $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}^{B}_{A,t}$.



a) Scenario 1 **b)** Scenario 2

Fig. 4.1: Different scenarios the maximum likelihood solution may achieve

Maximum A Posteriori:

Theorem 1 and Theorem 2 help us to narrow down the boundary of the maximum likelihood solution. Maximum a posterior estimation is then applied in the following paragraphs to get the result.

The quaternion difference from $\hat{\mathbf{q}}_{A,t}$ to $\hat{\mathbf{q}}^{B}_{A,t}$ is given as Eq. 4-14:

$$\mathbf{q}_{diff} = \hat{\mathbf{q}}^{-1}_{A,t} \otimes \hat{\mathbf{q}}^{B}_{A,t} = [\sin(\theta/2)[k_x; k_y; k_z]; \cos(\theta/2)] \qquad \textbf{(4-14)}$$

Suppose $\tilde{\mathbf{q}}_{A,t}$ is the maximum likelihood solution, which has angle differences α and β to $\hat{\mathbf{q}}_{A,t}$ and $\hat{\mathbf{q}}_{A,t}^{B}$, respectively, then $\tilde{\mathbf{q}}_{A,t}$ is given by Eq. 4-15:

$$\theta = \alpha + \beta$$
$$\tilde{\mathbf{q}}_{A,t} = \hat{\mathbf{q}}_{A,t} \otimes [\sin(\alpha/2)[k_x; k_y; k_z]; \cos(\alpha/2)] \quad \textbf{(4-15)}$$

, where $[k_x ; k_y ; k_z]$ is a unit vector representing the axis of rotation. Eq. 4-10 becomes

$$F(\alpha) = f(\hat{\mathbf{q}}_{A,t} \mid \mathbf{q}_{A,t}) f(\hat{\mathbf{q}}_{A,t}^{B} \mid \mathbf{q}_{A,t}) = \frac{1}{sigma\_1}\chi(\frac{\alpha}{sigma\_1})\frac{1}{sigma\_2}\chi(\frac{\theta-\alpha}{sigma\_2}) \quad \textbf{(4-16)}$$

When the derivative of F, $F'(\alpha)=0$, $F(\alpha)$ obtains the maximum value. Thus

$$F'(\alpha) = F(\alpha)(\frac{-\alpha}{sigma\_1} + \frac{\theta-\alpha}{sigma\_2}) = 0 \quad \textbf{(4-17)}$$

The two sensors are identical, so let *sigma_1= sigma_2*. This yields:

$$\alpha = \frac{\theta}{2} \quad \textbf{(4-18)}$$

The solution with highest probability $\tilde{\mathbf{q}}_{A,t}$ is calculated by Eq. 4-19:

$$\tilde{\mathbf{q}}_{A,t} = \hat{\mathbf{q}}_{A,t} \otimes [\sin(\theta/4)[k_x; k_y; k_z]; \cos(\theta/4)] \quad \textbf{(4-19)}$$

The maximum likelihood solution for Sensor *B* is calculated by:

$$\tilde{\mathbf{q}}_{B,t} = \tilde{\mathbf{q}}_{A,t} \otimes \mathbf{q}_{shift} \quad \textbf{(4-20)}$$

In the previous steps, we calculated $\tilde{\mathbf{q}}_{A,t}$ before $\tilde{\mathbf{q}}_{B,t}$. If we choose to calculate $\tilde{\mathbf{q}}_{B,t}$ first, the same result would be obtained.

It is worth noting that the presented calculation of the maximum likelihood solution is based on two assumptions: 1) the measurement error distributions

of the two sensors are similar; 2) The measurement errors are based on the Gaussian distribution. If the two assumptions are not met in certain circumstances, suboptimal results would be obtained. In the following section, we would notice such cases.

4.4. Experimental Evaluation on the Orientation Estimation

In order to evaluate the performance of the proposed method, we firmly stuck two inertial sensors, as the one used in Chapter 3, together using adhesive tape. Two sensors are relatively steady, as shown in Fig. 4.2. The sensors are connected to a computer through a USB port; the captured sensor data are presented in text format.

Fig. 4.2: Attached Sensor *A* and Sensor *B*

4.4.1. Experimental Testbed Setup and Ground Truth Calculations

The primary difficulty for the evaluation is to get the ground truth of the orientation. In this work, we have been using the motion tracking software, Cortex from Motion Analysis [98], which utilises an eight camera Osprey Digital Real Time System (ODRTS) for 1 millimetre level accuracy location tracking, as illustrated in Fig. 4.3. The ODRTS only tracks the location of the markers. In order to track the actual orientation, five markers are stuck to a quadrilateral cardboard, as well as to the attached dual-sensor. When the cardboard is rotated, the dual-sensor should have the same rotation. The rotation of the cardboard can be determined based on the tracked location of the 5 markers, with fixed topology, during movement.

Suppose the 5 markers are with locations $\mathbf{l}_0^1$, $\mathbf{l}_0^2$, $\mathbf{l}_0^3$, $\mathbf{l}_0^4$, $\mathbf{l}_0^5$ at time 0, respectively. In time *t,* the tracked locations are represented as $\mathbf{l}_t^1, \mathbf{l}_t^2, \mathbf{l}_t^3, \mathbf{l}_t^4, \mathbf{l}_t^5$. The quaternion rotation $\mathbf{q}_t$ is the solution of Eq. 4-21:

$$\min \sum_{i,j} [(\mathbf{l}_t^j - \mathbf{l}_t^i) - \mathbf{A}(\mathbf{q}_t)(\mathbf{l}_0^j - \mathbf{l}_0^i)], i \neq j \;\&\; i, j \in [1,5] \qquad \textbf{(4-21)}$$

, where **A** is the attitude matrix defined in Appendix A.2.

Fig. 4.3: Osprey Digital Real Time System for location tracking

From the known sensed magnetic field and the Earth's gravity, the initial orientation of Sensor A and Sensor B are computed as $\mathbf{q}_{A,0}$ and $\mathbf{q}_{B,0}$. At time t, the orientation of Sensor A after dual sensor fusion is $\tilde{\mathbf{q}}_{A,t}$, then the rotation until time t is:

$$(1;1;0)^{\mathrm{T}} Z(1;1;0) = 2 \qquad \textbf{(4-22)}$$

$$(\alpha_k;e_j)^T Z(\alpha_k;e_j) - \alpha_{kj}^{+} + \alpha_{kj}^{-} = \hat{d}_{kj}^2, \forall (k,j) \in N_e \qquad \textbf{(4-23)}$$

As the dual sensor is attached firmly on the cardboard, $\overline{\mathbf{q}}_{A,t}$ should equal $\mathbf{q}_t$ if the results are accurate.

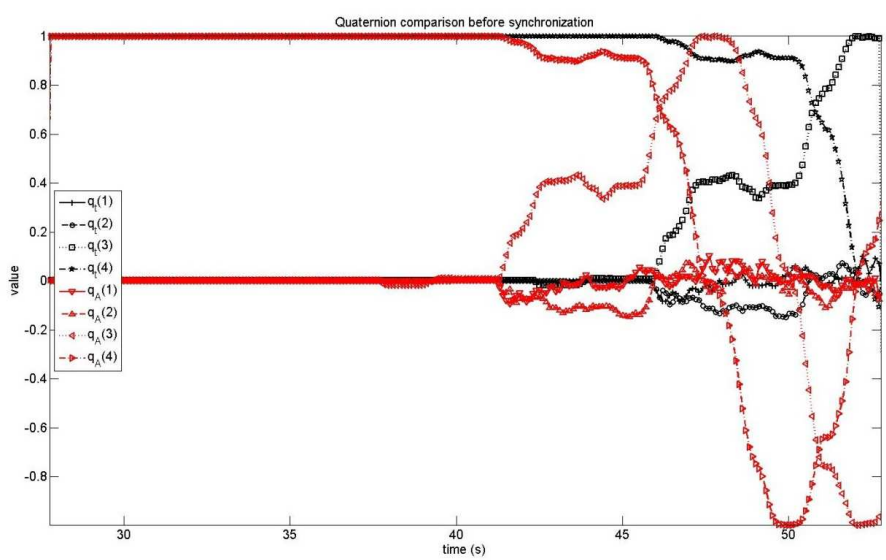
4.4.2. System Synchronization

Both the dual sensor and ODRTS are sampled at the 100Hz rate. Because of the time difference in initializing the sampling process of these two systems, the samples are asynchronous. To solve this problem, an impulse is given to

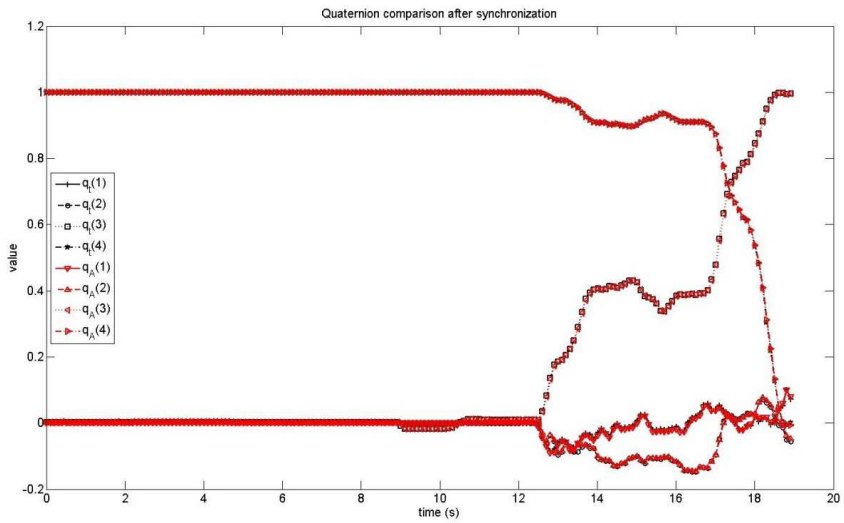
the cardboard at the beginning of the data collection process. Because of the high accuracy of the gyroscope, the solved $\overline{\mathbf{q}}_{A,t}$ should be close to $\mathbf{q}_t$ within a short period of rotation. The computed quaternion orientation before synchronization is shown in Fig. 4.4 a). The four values in vector $\mathbf{q}_t$ and the four values in vector $\overline{\mathbf{q}}_{A,t}$ are plotted. By shifting the $\mathbf{q}_t$ values, the values in $\mathbf{q}_t$ and $\overline{\mathbf{q}}_{A,t}$ may coincide.

The best time shift t_{shift} for synchronization is obtained when the minimum mean squared error is obtained in Eq. 4-23. The results after synchronization are illustrated in Fig. 4.4 b), where the signal is truncated to start right before the rotation. In the following sections, the times are presented in a synchronized manner.

$$\min \sum_{t=40}^{t=50} \| \overline{\mathbf{q}}_{A,t-t_{shift}} - \mathbf{q}_t \|^2 \ , t_{shift} \in [0,5] \tag{4-24}$$



a) Before synchronization



b) After synchronization

Fig. 4.4: Computed quaternion orientation before and after synchronization

4.4.3. Experimental Results

We collected a total of 8 trials of data, with each trial lasting for 250 seconds at a 100Hz sampling rate. We made sure all possible rotations were tracked. An example is shown in Fig. 4.5, where each element in $\mathbf{q}_t$ has a value range of [-1 1]. The rotation required from $\overline{\mathbf{q}}_{A,t}$ to $\mathbf{q}_t$ is:

$$\mathbf{q}_{delta} = \overline{\mathbf{q}}_{A,t}^{-1} \otimes \mathbf{q}_t \qquad \textbf{(4-25)}$$

The error size is not related to the rotation axis, but is related to the rotation angle value. In this thesis, we use the angle value $err = 2 * \arccos(|\mathbf{q}_{delta}(4)|)$ to represent the error.

We compare the error before and after the fusion. An example of the results is presented in Fig. 4.6. The errors of Sensor *A* and Sensor *B* and the results after fusion (Sensor *A*) are illustrated. The angle is measured in radians. Fig. 4.6 b) shows more zoomed in results.

The effect of the fusion algorithm can be categorized into two cases. In the first case, the fused error is in between the error of pure Sensor *A* and Sensor *B*. This can be explained when the estimation error of Sensor *A* and Sensor *B* are on the same side, as compared with the actual orientation. In the second case, the fused error is less than the error from both Sensor *A* and Sensor *B*. This can be explained when the error of Sensor *A* and Sensor *B* are on opposite sides, as compared to the actual orientation, and the fused result is closer to the actual orientation.

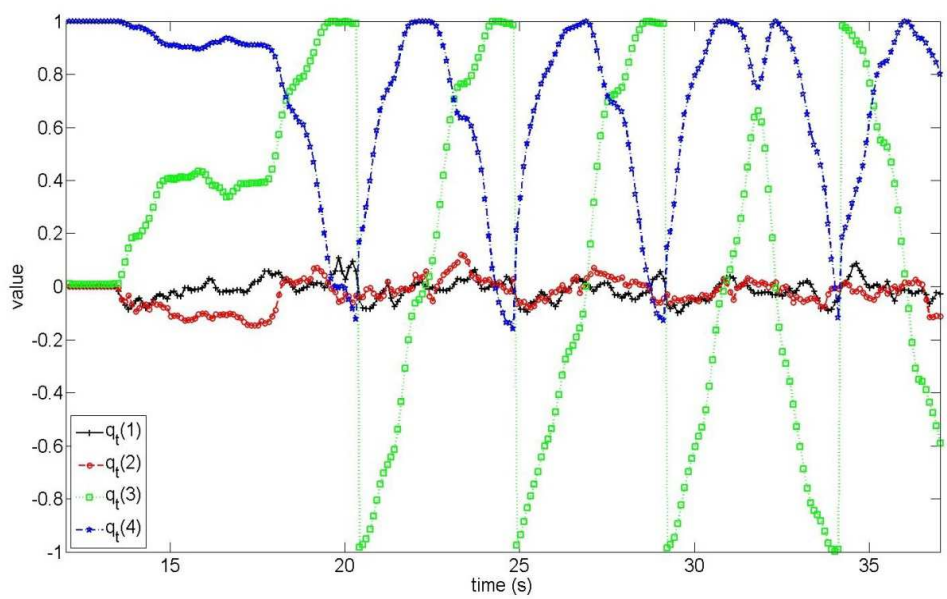
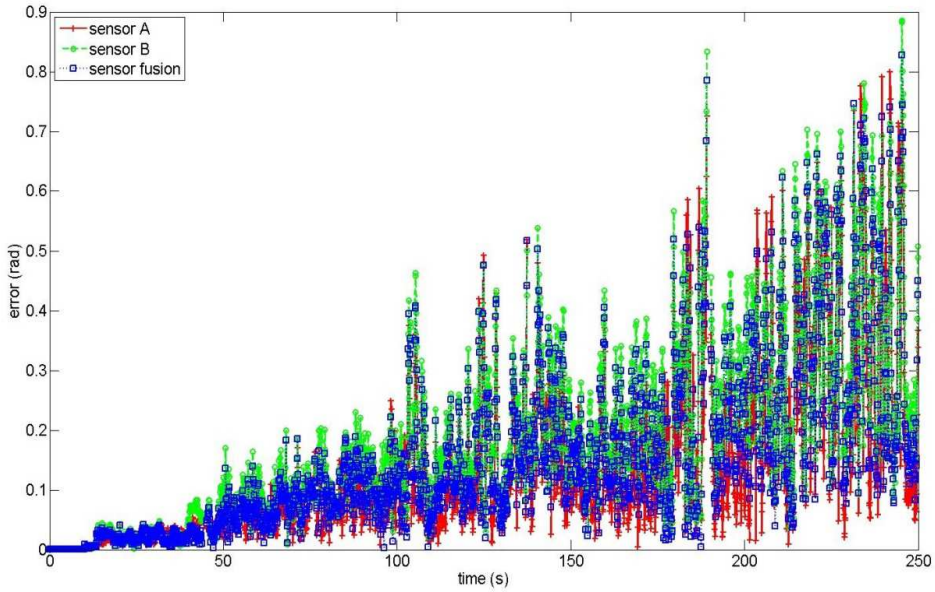
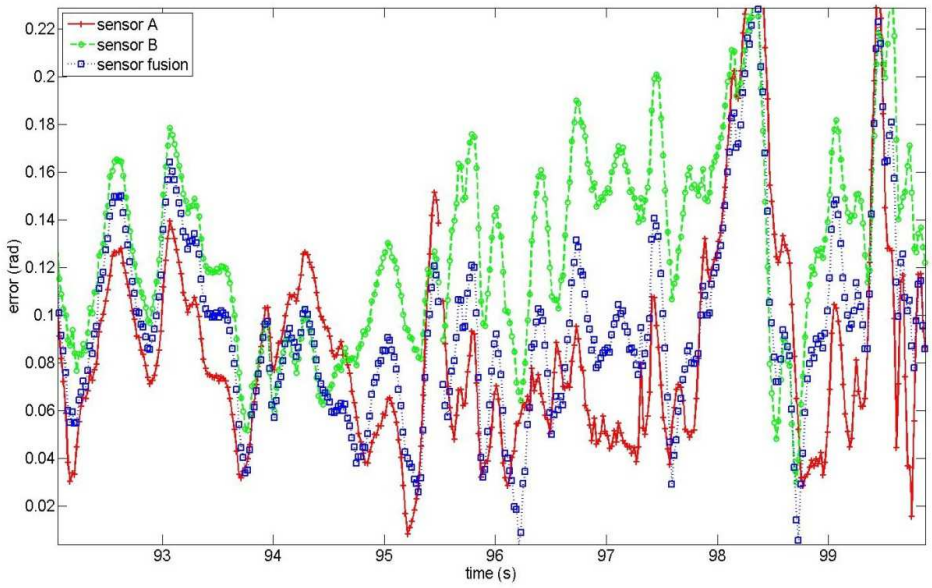


Fig. 4.5: The real temporal rotation of Trial 1



a) Temporal error comparison in 250s range



b) Temporal error comparison after being zoomed in

Fig. 4.6: Example of orientation accuracy

The means and variances of the error for the 8 trials are illustrated in the Fig. 4.7. The unit of the mean is rad. The fused results of Sensor *A* and Sensor *B* have the same values. A first observation is that, for the same sensor, various results would be obtained for different experimental trials. This is determined

by the difference of sensor error. The bias error of a gyroscope is due to a number of components like calibration errors and bias drift. For a gyroscope, its bias error tends to vary, both with temperature and over time. Thus, each time the sensor is switched on to collect the data, a slightly different result is expected.

The effect of the fusion algorithm on the mean error can be categorized into two cases. In Trials 2, 4, 5, 6 and 8, the fused mean error is less than the error from either Sensor *A* and Sensor *B*. In Trials 1, 3, and 7, it is shown that the fused algorithm delivers results with the accuracy in between those of Sensor *A* and Sensor *B*'s individual orientation estimation algorithms. The fused solution is not necessarily more accurate than just using Sensor A and B alone. Similar result are achieved in [92]. It can be explained by the temporal results of Trial 1 illustrated in Fig. 4.6. Although Sensor *A* and Sensor *B* are the same types of devices and are supposed to have identical error distributions, it is illustrated in Fig. 4.6 b) that Sensor *A* has smaller errors most of the time for this trial of experiment. Most of the time, the temporal fusion error is larger than the errors from Sensor *A* and less than the one from Sensor *B*. The final result, which is the aggregated average, is in between the accuracies of Sensor *A* and Sensor *B*.

Although the proposed scheme does not always achieve the highest accuracy, the proposed scheme is still useful, even in the worst cases. One may argue why not use the one that returns a better accuracy and omit the fusion process. But in a real situation, it is difficult, or even impossible, to determine which one is providing the better results. In the case of Sensor *A*

and Sensor *B*, either sensor would potentially have higher bias error during data collection. In Trials 1 and 3, Sensor *A* provides better results than Sensor *B*, but in Trial 8, Sensor *A* yields worse results. The fusion algorithm provides a more robust and accurate solution. Besides, even in the cases the proposed scheme does not return the highest accuracy, the accuracy is closer to the one with a smaller error. In Trials 1, 3, and 7, the average errors for Sensor *A* are 0.13, 0.13 and 0.12; the ones for Sensor *B* are 0.18, 0.17 and 0.18. The respective errors from fused method are 0.15, 0.14 and 0.13. They are closer to the better results from Sensor *A*. This is because the average accuracy is only obtained when the orientation estimation error of Sensor *A* and Sensor *B* are on the same side, as compared to the actual orientation during the entire tracking time. Once the two errors can be partially eliminated, the fused error, in that instance, is less than the average accuracy of Sensor *A* and Sensor *B*.

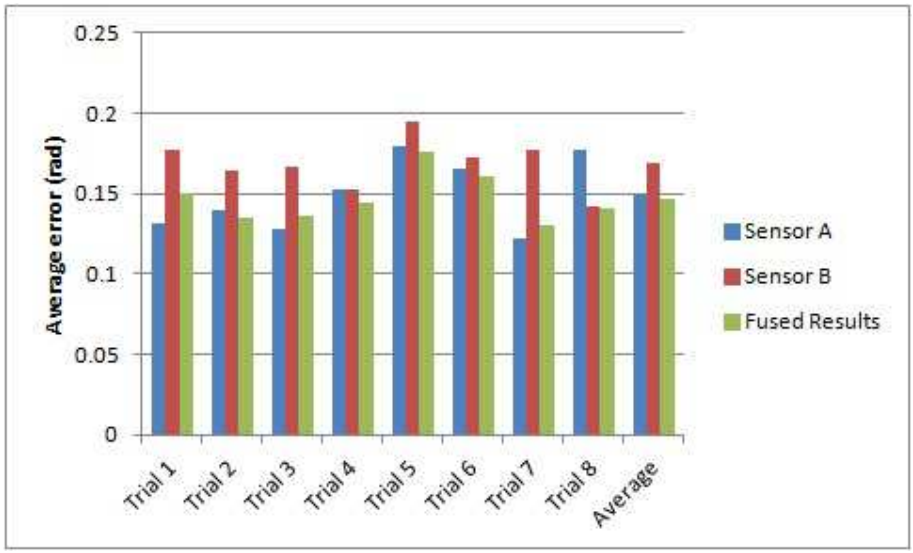


Fig. 4.7: Orientation estimation performance comparison before and after fusion

4.5. Dual Sensor Dead-Reckoning

In the previous sections of this chapter, we discussed the dual sensor fusion method on orientation estimation. Here we test the effect of orientation fusion on the performance of DR. The plan is that we carry the attached dual sensor system to collect data when we walk in a known ground truth environment. Based on the collected two sets of data, different localization results can be obtained given the various fusion methodologies. There are three fusion options for localization: (1) only fusion on the orientation estimation, then each sensor shall proceed with DR; (2) only fusion on the final DR location estimation as in [92], without fusion on the orientation estimation; (3) fusion on the orientation estimation first, then fusion on the DR location estimation using (2). Option 2 is based on the fact that if the two sensors are attached close to each other, their distance always keep the same. It is equivalent to the case when pedestrian carries two sensors. The authors in [92] had studied this case. MLE can be applied to fuse the localization results from the two sensors. No matter which fusion method we use, we should be aware of the previous knowledge that the fusion methods do not guarantee accuracy improvements, but help to provide more robust results.

4.5.1. Experimental Testbed

There were 5 trials of data collected when a pedestrian was asked to walk around a rectangle conference room (10m by 8.5m) for 4 rounds. The walking distance for each trial was 148 m. When a certain point in the walking path was passed, the passing timestamp was saved. There were 28 pieces of ground truths with timestamps for the route. An example of the route is illustrated in

Fig. 4.8 where the arrow line and the line connected by the small triangles represent the actual route and the estimated route, respectively. The route starts from the bottom left corner, continues in a counter clockwise direction, and finally ends at the starting point after 4 rounds. There is a clear drifted trend on the direction estimation.

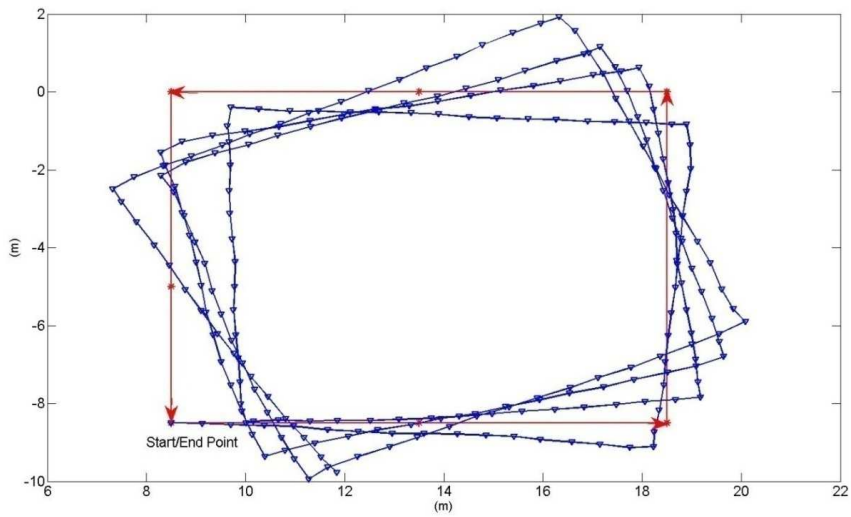


Fig. 4.8: 1 trial of results: the pedestrian walked around a rectangle conference room for 4 rounds

4.5.2. Algorithm Briefing

As discussed previously, there are three fusion options, which will be compared in the experiments. In Option 1, we only fuse the orientation estimation of the two sensors. The returned orientation is the input for each of the sensors to do a step-counting DR algorithm, which has been illustrated in Chapter 3. In Option 2, we fuse the DR location of the two sensors, as in [92]. In Option 3, there are two levels of fusion. We fuse the orientation estimation

for each of the sensors to do a DR; the two DR locations are fused in the manner presented in Option 2.

4.5.3. Experimental Results

Fig. 4.9 presents the average localization errors for each trial using the different methods, as well as the average of the 5 trials. In Option 1, only the sensor's orientation is fused, so the estimated locations are different because of the difference in the other measurements. In Options 2 and 3, the locations are fused, so Sensor *A* and Sensor *B* share the same localization results.

In Option 1 when only the orientation is fused, we find an average accuracy improvement for both sensors, which shows the advantage of the orientation fusion on location tracking. When the location is fused in Options 2 and 3, the accuracy is somewhat in between the accuracies of Sensor *A* and Sensor *B*. The results from Options 2 and 3 illustrate no significant difference, which suggests that fusing orientation before location fusion is not necessary. Because the orientation estimation is one input for the DR location estimation, fusing the location already fuses the orientation output, to a certain extent.

So the primary advantage of this orientation fusion method is still to provide more robust orientation estimation for the rotation tracking use cases. For the DR localization application, the orientation fusion still improves the single sensor performance with higher accuracy. But compared to Option 2, with direct location fusion, the improvement is limited.

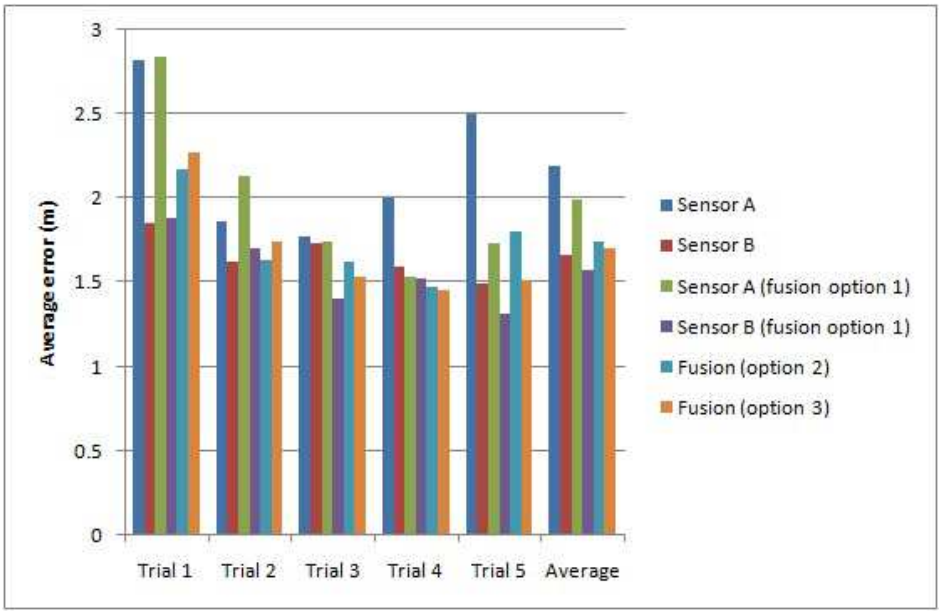


Fig. 4.9: Localization accuracy comparison before and after fusion

Chapter 5

Cooperative Localization

It is not costly effective to equip all users with self-localization hardware like GPS. This motivates research on localization estimation in a wirelessly connected network using relative location information, such as pair-wise ranging measurements between nodes. That's because the radio transmission hardware are quite ubiquitous in many devices, which can be used for the distance estimation. With the relative location information, different algorithms can be applied to compute the location which can be categorized as centralized algorithms and distributed algorithms. In centralized algorithms, there is a central node computing the locations for all nodes, while in distributed algorithms, the node typically computes the location for itself. Centralized algorithms usually yield more accurate results, but suffer from single node failure. To combine the advantages of both methods, the preferred methodology in this chapter is a cluster based method.

In this chapter, we propose a cluster based localization scheme so that different cluster based algorithms can be easily implemented. To carry out a comparative evaluation on the performance in a fully connected network with many anchor nodes, three algorithms are implemented, which are based on the use of extended Kalman filter (EKF), semi-definite programming (SDP) and multi-dimensional scaling (MDS). Their cluster based variants are the decentralized EKF (DEKF), cluster based SDP (CSDP) and cluster based MDS (CMDS), respectively. The algorithms are evaluated in both static and low mobility environments. Simulation results show that DEKF performs as well as EKF in both static and low mobility environments, and they outperform CSDP and CMDS. DEKF requires fewer anchor nodes, smaller clusters, while achieving more accurate location estimation.

Then we investigate that the known movement pattern returned by the motion sensing technique relaxes the 3-anchor and 3-connection requirements for 2D localization. The DR technique is applied using PF when the displacement can be estimated and there are only one or two anchor nodes. Simulation results show that when exact displacement is given, accurate location results are returned in all 3 defined cases. When there is error in the direction estimation, two anchor nodes are required and it is more robust to use more particles. The localized mobile node would become a moving anchor which helps to localize the rest of the nodes in the network. The DR location is fed into the original cluster based method which shows significant improvement. There are more nodes that can be localized (increased from 0 to

48 in 2-anchor case), and the accuracy is also enhanced (reached 0.05 for CMDS).

5.1. Preliminaries

This section describes the key definitions and assumptions used in our algorithm.

5.1.1. Definition of Terms

- Node *ID*: Each node has a unique *ID* to identify itself. Cluster *ID* is the cluster head's node *ID*.
- Anchor node: Node with known location
- Transmission range: It defines the furthest distance for successful message transmission (abbreviate as T_{range}, normalized to 1). Nodes within the transmission range are one hop away.
- Head Rank: It is an integer identifying how far a node can be away from a cluster head. Suppose the rank of the cluster head is set to n, then the nodes within n-1 hops may join it.
- Localization error: The distance between a node's actual location and its estimated location

5.1.2. Assumptions

First we assume that all nodes in the network have the same T_{range} (which is normalised to 1). The distances are multiples of T_{range}. Nodes are assumed to be capable of message transmission and distance measurement within T_{range}. The measurement error is modelled as Gaussian noise as in [65][79][82][101]. The distance measurement is given as in [79]

$$\hat{d}_{ij} = d_{ij} * (1 + \gamma) \quad \textbf{(5-1)}$$

where $\hat{d}_{ij}$ is the measured distance, $\hat{d}_{ij}$ is the actual distance, $\gamma \sim N(0, noisefactor^2)$ is a random Gaussian variable.

In addition, a limited subset of nodes in the network is assumed to be anchor nodes.

5.2. Dense Network with Many Anchor Nodes

In this section, we discuss the localization problem in a fully connected wireless network with many anchor nodes. The full connection (more than 2 neighbours) ensures there are enough distance constraints for each of the sensor, and the anchor nodes (more than 2) provide enough ground truth of global coordinates system.

5.2.1. Localization Algorithm

The localization algorithm is performed in two phases: the clustering phase and the localization phase. The two phases together make up one iteration.

a. Phase 1: Clustering

In the clustering phase, clusters are formed by distributed decisions of the nodes. Our clustering algorithm comprises three components, namely, cluster head initialization, cluster maintenance and route maintenance.

- **Cluster Head Initialization**

It begins with each node entering the head election phase being given a predefined probability, and each node elects itself to be the cluster head based on this probability. After this phase, if the node becomes the new cluster head,

it broadcasts a cluster update message which includes the cluster *ID*, as well as its rank in the cluster. Suppose a node receives a cluster update message from a rank *n* cluster head, it will get rank *n*-1, and if the rank it gets is larger than 1, it rebroadcasts this message. The rank in each cluster update message is decreased by one after every rebroadcast, and the broadcast ends at the rank-1 nodes, as shown in Fig. 5.1.

- **Cluster Maintenance**

During the cluster maintenance phase, the members of the clusters in the network are updated. Even though a node may receive messages from different cluster heads, each node can only join one cluster. To determine the cluster to join, each node maintains the received cluster sets which include the cluster *ID*s and the obtained ranks in the clusters. Upon receiving a cluster update message, the cluster sets are updated. At the end of the broadcasting phase, the node joins the cluster where it can get the highest rank. If there is more than one cluster with the same highest rank, it joins the one with the highest cluster *ID*.

To ensure that clusters have a regular shape and size, a cluster head should not receive cluster update messages from other heads. Upon receiving a cluster update message, the cluster head compares its own *ID* with the cluster *ID* in the message. We impose a rule that the cluster with a higher cluster *ID* is kept. So if a cluster head has a smaller ID, it joins another cluster and broadcasts a head deletion message, the result is shown in Fig. 5.2.

If a node does not receive any cluster update message at the end of the phase, it becomes a cluster head and broadcasts a cluster update message. This ensures that all nodes join clusters at the end of the phase.

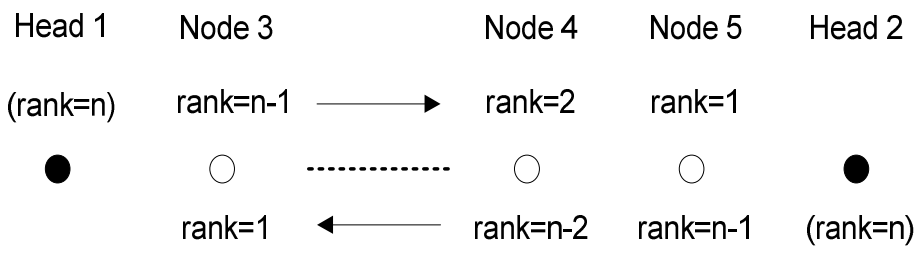


Fig. 5.1: Explanation of the rank in cluster

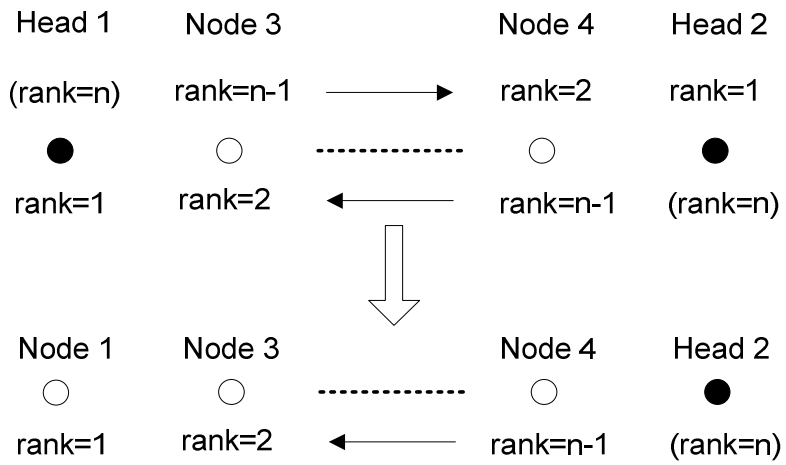


Fig. 5.2: Lower *ID* head joins the cluster with larger *ID* if they are in range

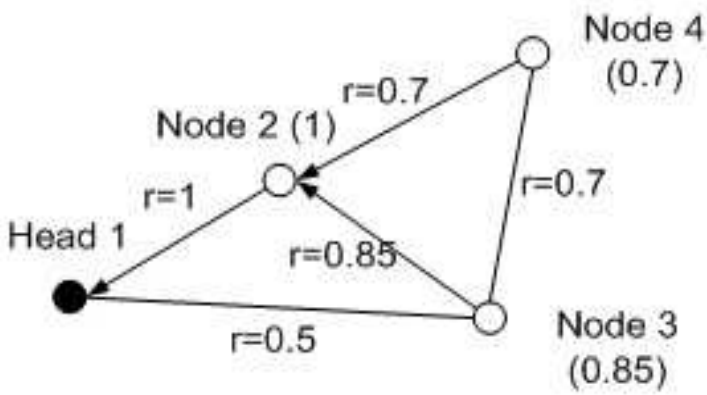


Fig. 5.3: Maintained routes from member nodes to cluster head

- **Route Maintenance**

The route maintenance method maintains the next hops from the member nodes to their respective cluster heads. At every iteration in the cluster maintenance, a cluster head broadcasts messages to its member nodes. The logs of the broadcast process provide the routes from the cluster head to all the member nodes. Conversely, we use the routes to update data packets from the member nodes to the cluster head.

The route robustness metric between neighbour nodes, r_{ij}, is obtained as

$$r_{ij}(\hat{d}_{ij}) = \begin{cases} 1, & 0 \le \hat{d}_{ij} < \dfrac{T_{range}}{2} \\ \dfrac{-2}{T_{range}}\hat{d}_{ij} + 2, & \dfrac{T_{range}}{2} \le \hat{d}_{ij} < T_{range} \end{cases} \qquad \textbf{(5-2)}$$

, which takes a value between [0,1], and where $\hat{d}_{ij}$ is the measured distance between node i and j, and T_{range} is the transmission range. When two nodes are connected by two or more hops, the route robustness is obtained by multiplying the robustness of each hop. The node chooses the route with the

highest robustness value as shown in Fig. 5.3. Although node 3 has a direct link to node 1, it actually prefers to relay through node 2 with the largest robustness value of 0.85. One reason for this choice is that as the sender and receiver gets closer, the received signal strength gets larger, which results in larger SNR, or with shorter distance, the node can transmit signal with less power, which conserves energy.

An example of the clustering algorithm is shown in Fig. 5.4 (200 nodes in an (8 by 8) map, head rank=3). The small dots in the figure represent the nodes to be localized in the network, and the larger dots are the elected cluster heads.

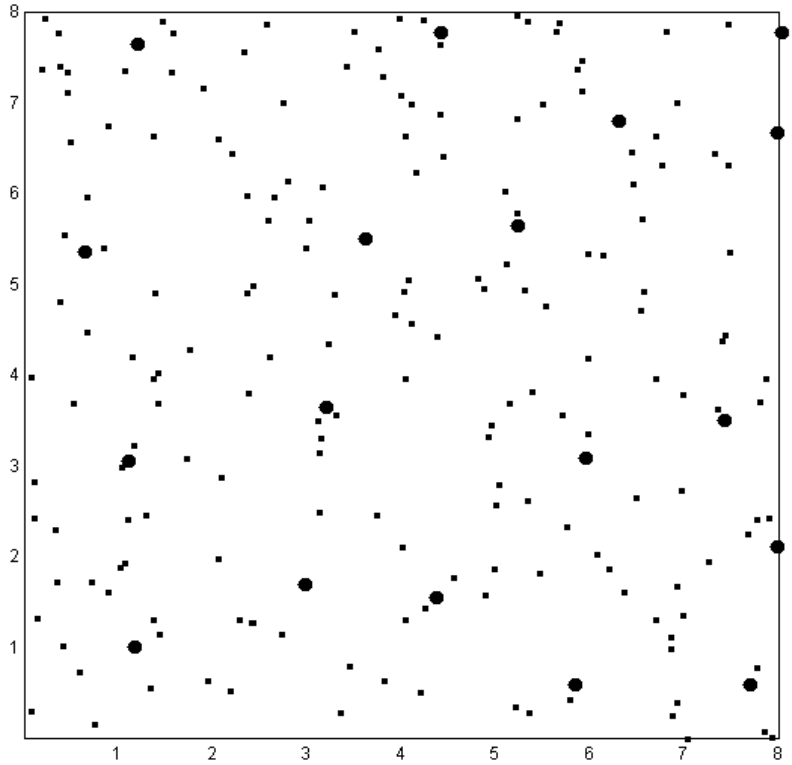


Fig. 5.4: Example of clustering algorithm (head rank=3)

b. Phase 2: Localization

The localization phase is to carry out location estimation. After the localization process is introduced, we illustrate the implementation of the localization algorithms.

- **Localization Process**

The process starts with each member node updating its relative location message to its cluster head. The message includes the estimated locations of itself and its neighbour nodes (for anchor nodes they have actual absolute locations), as well as the estimated distances between them. The contents of the relative location message are shown in Fig. 5.5.

After the cluster heads receive the messages, they compute the locations for their member nodes. The computed results are broadcasted within the cluster in the form shown in Fig. 5.6.

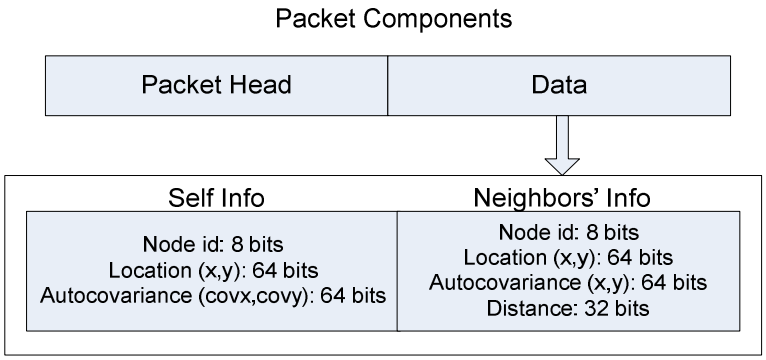


Fig. 5.5: Components of relative location message

Fig. 5.6: Components of head localization results

- **KF Implementation**

The Kalman Filter (KF) [99] provides an efficient computational recursive means to estimate the state of a process which minimizes the mean of the squared error, and has long been used in target tracking. The KF works with initial estimates of the positions of nodes. DV-distance [100] is an appropriate algorithm in this phase because of its simplicity and yet provides enough accuracy for the KF. The implementation of DV-distance is not described here.

The implementation of the KF has been described in [101]. The major difference from [101] is that they only considered the centralized EKF.

Suppose

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{u}_k + \mathbf{w}_k \tag{5-3}$$

$$\mathbf{z}_k = h(\mathbf{z}_k) + \mathbf{v}_k \tag{5-4}$$

represent the controlled process and the measurement respectively, where $\mathbf{x}$ = $[a_1, b_1, a_2, b_2, a_3, b_3, \ldots]^T$ is the location of nodes (the subscript represents the node id, a and b represent the 2-d coordinates), $\mathbf{u}$ is the speed of the nodes, and $\mathbf{z}$ is the measured distances between neighbour nodes. $h(\mathbf{x}_k)$ = $[d_{12}, d_{13}, d_{21}, d_{23}, d_{31}, d_{32}, \ldots]^T$ (suppose node 1, 2 and 3 are neighbours). Let $\mathbf{w}_k$ and $\mathbf{v}_k$ represent the matrices of the process and measurement noise which we assume to have zero mean Gaussian distributions, and $\mathbf{Q}$ and $\mathbf{R}$ are their covariance matrices, respectively.

Let $\mathbf{H}$ be the Jacobian matrices of the partial derivatives of h with respect to $\mathbf{x}$. The expression of $\mathbf{H}$ can be found in [101]. The node speed $\mathbf{u}$ cannot be detected, so it is deleted from the state transition equation. The state equation

is compensated by larger diagonal entries in **Q** (larger uncertainty). Suppose **P** is the a posteriori error covariance matrix, and **K** is the optimal Kalman gain. Then the operation of the EKF can be represented by following equations.

$$\hat{\mathbf{x}}_{\bar{k}} = \hat{\mathbf{x}}_{k-1} \tag{5-5}$$

$$\mathbf{P}_{\bar{k}} = \mathbf{P}_{k-1} + \mathbf{Q} \tag{5-6}$$

$$\mathbf{K}_k = \mathbf{P}_{\bar{k}} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{\bar{k}} \mathbf{H}_k^T + \mathbf{R})^{-1} \tag{5-7}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{\bar{k}} + \mathbf{K}_k (\mathbf{Z}_k - \mathrm{h}(\hat{\mathbf{x}}_{\bar{k}})) \tag{5-8}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{\bar{k}} \tag{5-9}$$

In the decentralized version of KF, the system model and observation model are partitioned. The state transition equation can be easily distributed to each node itself. The major difficulty is in the way the a posteriori error covariance matrix P is handled. In a dynamic network, since the member nodes in a cluster keep changing, the cluster head does not have a consistent evolution of error covariance. In our algorithm, each node only memorizes its own error autocovariance and contributes it to the cluster head. The cross-covariance terms are set to 0. The formed P is a diagonal matrix.

- **SDP Implementation**

The SDP algorithm we use is described in [79], which provides an SDP relaxation based method for localization. Suppose there are m anchor nodes with coordinates $\mathbf{a}_k \in R^2$, $k = 1,\ldots,m$, and n unknown nodes with coordinates $\mathbf{x}_j \in R^2$, $j = 1,\ldots,n$. Suppose the transmission range is T_{range}. There are two sets N_e and N_l being defined as

if $\| \mathbf{x}_i - \mathbf{x}_j \|^2 = \hat{d}_{ij}^2 \le T_{\text{range}}$, $\| \mathbf{a}_k - \mathbf{x}_j \|^2 = \hat{d}_{ij}^2 \le T_{\text{range}}$, then (i,j), $(k,j) \in N_e$

else if $\| \mathbf{x}_i - \mathbf{x}_j \|^2 > T_{\text{range}}$, $\| \mathbf{a}_k - \mathbf{x}_j \|^2 > T_{\text{range}}$, then (i,j), $(k,j) \in N_l$.

The localization problem is written as a standard SDP problem:

$$\begin{aligned}
\min & \sum_{i,j\in N_e, i<j} (a_{ij}^{+} + a_{ij}^{-}) + \sum_{k,j\in N_e} (a_{kj}^{+} + a_{kj}^{-}) \\
& + \sum_{i,j\in N_l, i<j} \beta_{ij}^{-} + \sum_{k,j\in N_l} \beta_{kj}^{-} \qquad \textbf{(5-10)} \\
s.t. & \left(1;0;0\right)^{\mathrm{T}} \mathbf{Z}\left(1;0;0\right) = 1 \\
& \left(0;1;0\right)^{\mathrm{T}} \mathbf{Z}\left(0;1;0\right) = 1 \\
& \left(1;1;0\right)^{\mathrm{T}} \mathbf{Z}\left(1;1;0\right) = 2 \\
& \left(\mathbf{a}_k;\mathbf{e}_j\right)^{\mathrm{T}} \mathbf{Z}\left(\mathbf{a}_k;\mathbf{e}_j\right) - a_{kj}^{+} + a_{kj}^{-} = \hat{d}_{kj}^2, \forall (k,j) \in N_e \qquad \textbf{(5-11)} \\
& \left(\mathbf{0};\mathbf{e}_{ij}\right)^{\mathrm{T}} \mathbf{Z}\left(\mathbf{0};\mathbf{e}_{ij}\right) + \beta_{ij}^{-} > \bar{r}^2, \forall (i,j) \in N_l, i<j \\
& \left(\mathbf{a}_k;\mathbf{e}_j\right)^{\mathrm{T}} \mathbf{Z}\left(\mathbf{a}_k;\mathbf{e}_j\right) + \beta_{kj}^{-} > \bar{r}^2, \forall (k,j) \in N_l \\
& \alpha_{ij}^{+}, \alpha_{ij}^{-}, \alpha_{kj}^{+}, \alpha_{kj}^{-}, \beta_{ij}^{-}, \beta_{kj}^{-} \geq 0 \\
& \mathbf{Z} \geq 0
\end{aligned}$$

where $\mathbf{Z} := \begin{pmatrix} \mathbf{I} & \mathbf{X} \\ \mathbf{X}^T & \mathbf{Y} \end{pmatrix} \geq 0$, $\mathbf{Y}=\mathbf{X}^{\mathbf{T}}\mathbf{X}$, and $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2\ \ldots\ \mathbf{x}_n]$ is the $2 \times n$ matrix that needs to be solved. $\mathbf{e}_{ij}$ is the vector with 1 at the ith position, -1 at the jth position and zeros everywhere else; and $\mathbf{e}_j$ is the vector of all zero except -1 at the jth position. We use SDPT3 [102] as the solver to find the globally optimum SDP solution. Compared with the Kalman filter, the SDP does not need the initial estimates of the locations to start the algorithm. But SDP needs at least three anchor nodes to solve the problem.

- **MDS Implementation**

The CMDS algorithm applied in this paper is different from that described in [75]. The nodes' distances beyond one hop are not estimated. The computational costly merging phase in [75] which merges local maps to the

global coordinates system is not applied. So the cluster head needs the locations of at least three anchor nodes to localize the member nodes, same as the SDP.

5.2.2. Evaluation

We simulate the performance of our algorithm using MATLAB. The simulator is downloaded from [103]. IEEE 802.11 (CSMA/CA, Virtual carrier sensing, and RTS-CTS-DATA-ACK mechanisms are implemented) is the underlying communication protocol. In our algorithm, each iteration includes the clustering and localization phases. The period of one iteration is set according to the type of environment. We compare the performance of three cluster based methods (CMDS, CSDP and DEKF) and the centralized methods (MDS, SDP and EKF). In the centralized methods, the head is assumed to have all relative location information.

In the simulations, we study various factors affecting the performance, including the number of anchor nodes, the value of the *noisefactor*, the size of the cluster (head rank), the bandwidth and the maximum velocity. In our simulation settings, T_{range} equals 1, the size of the network map and localization error is represented as a multiple of T_{range}. The algorithm is evaluated in a 50-node network. The nodes are assumed to be uniformly placed initially. Unless otherwise specified, the number of anchor nodes equals 9, the network is in a 4 by 4 map, the *noisefactor* is set to 0.05, and the bandwidth is set at 1Mbps. The errors are averaged over the number of localized nodes.

a. Static Environment

Here we evaluate the algorithms in a static environment. The period of one iteration is 1 second. The algorithms are run for 50 seconds and the performance metrics are the average values for the last 25 seconds.

- **Effect of Number of Anchor Nodes on Performance**

First we evaluate the effect of the number of anchor nodes on the location accuracy in a 4 x 4 map. More than 2 anchor nodes are required to determine a 2D topology. We increase the number of anchor nodes from 5 to 9. The results are given in Table 5-1. The values shown in the Table 5-1 are the average localization errors expressed as a multiple of T_{range}. It shows that DEKF and EKF achieve almost the same reliable location estimation, and their performances are less eroded by decreasing the number of anchor nodes. MDS, SDP and their cluster based methods show comparatively worse results.

We then study the effect of the number of anchor nodes m with respect to the number of localized nodes. There are a total of (50-m) nodes that need to be localized in the network. Table 5-2 shows the average number of nodes that can be localized for different number of anchor nodes for the 3 algorithms. CMDS and CSDP both work only when there are more than 2 anchor nodes within a cluster, while DEKF does not have this requirement. So it can be inferred that given the same number of anchor nodes, DEKF will localize more nodes than CMDS and CSDP. This is validated in Table 5-2. CMDS and CSDP localize the same number of nodes. It is shown that only when there are sufficient anchor nodes (9), the CMDS localizes nearly the same number of

nodes as DEKF (41 versus 40.8). It can be concluded that DEKF achieves better performance than CMDS and CSDP with fewer anchor nodes.

Table 5-1: Relationship of Anchor Nodes on Accuracy

No. of Anchors	***5***	***6***	***7***	***8***	***9***
DEKF-rank 3	0.029	0.024	0.024	0.024	0.023
EKF	0.023	0.023	0.023	0.022	0.022
CMDS- rank 3	0.234	0.068	0.060	0.096	0.084
MDS	0.122	0.042	0.042	0.041	0.042
CSDP-rank 3	0.268	0.266	0.250	0.179	0.249
SDP	0.353	0.194	0.182	0.158	0.120

Table 5-2: Relationship of Anchor Nodes on Number of Localized Nodes

No. of Anchors	**5**	**6**	**7**	**8**	**9**
DEKF-rank 3	44.7	43.8	42.8	41.8	40.8
CMDS-rank 3 / CSDP-rank 3	32	38.5	37.5	36.5	41

- **Effect of Noise Factor on Performance**

The distance estimation precision is influenced by the *noisefactor* and it has a significant effect on localization accuracy. The simulation results are shown in Table 5-3 and the values corresponding to the algorithms are the average localization errors expressed as a multiple of T_{range}. It is shown that as the *noisefactor* increases, all the algorithms obtain worse results. Both EKF and DEKF are seen to be more resilient to different values of *noisefactor* than MDS and SDP. The reason is that the computation of MDS and SDP utilize only the information from the current iteration, which easily suffers from instances of large distance estimation errors. Their cluster based algorithms

are even more vulnerable because of less available information for each computation.

Table 5-3: Influence of *Noisefactor* on Accuracy

Noisefactor	**0.02**	**0.04**	**0.05**	**0.06**	**0.08**
DEKF-rank 3	0.010	0.019	0.023	0.028	0.041
EKF	0.009	0.017	0.022	0.028	0.043
CMDS-rank 3	0.065	0.070	0.084	0.191	0.246
MDS	0.020	0.034	0.042	0.049	0.072
CSDP-rank 3	0.120	0.140	0.149	0.177	0.209
SDP	0.045	0.089	0.120	0.154	0.225

- **Effect of Rank of Cluster Head on Performance**

The size of the cluster is determined by the rank of the cluster head. As the size increases, there are more hops required for message transmission from the member nodes to the cluster head. The results in Table 5-4 show that the overhead increases as head rank gets larger. But both CMDS and CSDP benefit from larger clusters with more nodes, which is easier to meet the requirement of more than 2 anchor nodes in each cluster for localization. This is clearly seen in Table 5-4 when the head rank equals 2, the cluster is too small resulting in only 18.8 nodes being localized.

Table 5-4: Data Amount with Different Rank

Head rank	***2***	***3***	***4***
Average Overhead (kbps)	33.5	104.7	154.0
No. of Localized Nodes: DEKF	40.3	40.8	40.0
No. of Localized Nodes: CMDS/ CSDP	18.8	41	41

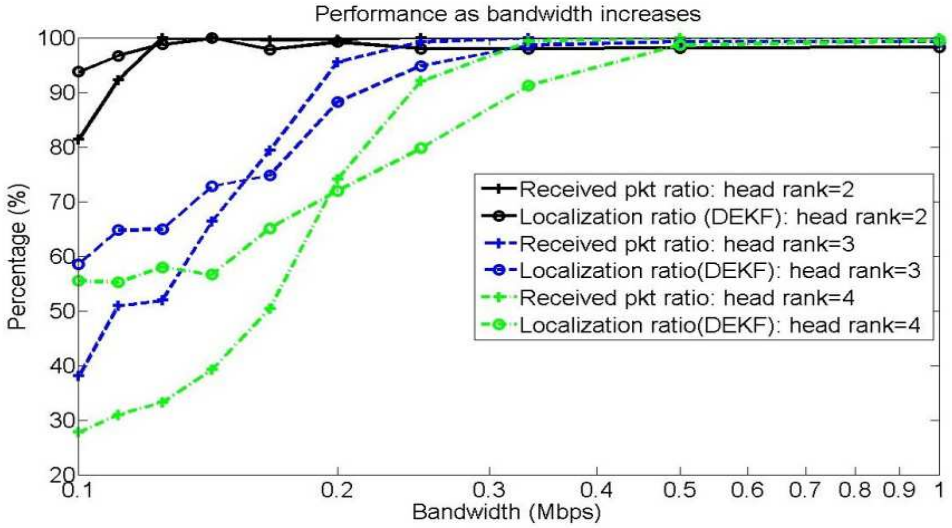


Fig. 5.7: Localization performance as bandwidth increases

- **Bandwidth Requirement**

The previous simulation results have shown the average overhead under different cluster sizes. In this section, we evaluate the bandwidth requirement for message transmission and localization. When the bandwidth is small, there is a higher chance of packets being lost, which results in less localized nodes. Unlike CSDP and CMDS, nodes will be localized by DEKF unless there is packet loss. So we study the relationship between packet loss and the number of localized nodes using DEKF. Fig. 5.7 shows the influence of bandwidth on successful packets receiving ratio (number of messages received divided by the total number of messages transmitted) and the localization ratio of nodes (number of localized nodes divided by the total number of non-anchor nodes). It is seen that the bandwidth requirement increases as the cluster head rank gets larger to achieve 100% packets receiving ratio and 100% localization ratio. From Fig. 5.7, we see that the minimum bandwidth requirements for

rank-2 clusters, rank-3 clusters and rank-4 clusters are about 1/8 Mbps, 1/3Mbps, and 1/2 Mbps, respectively.

b. Mobile Environment

The mobility model used in our investigations is the random waypoint model, in which the node moves toward a randomly selected destination with a random speed. Upon reaching the destination, the node will rest for a while and choose the next destination and random speed. The random speed has a uniform distribution from 0 to a pre-determined maximum speed. We investigate the effect of the increasing speed on performance.

The settings for the simulations are: bandwidth is set to 1 Mbps, *noisefactor* equals 0.05, and the number of anchor nodes equals 9. The nodes are deployed in a 4 by 4 map. The iteration interval is set to 0.5 second. The results are shown in Table 5-5.

We see from Table 5-5 that there is a clear trend that as the maximum speed increases, the accuracy of DEKF and EKF declines. This is because the algorithm is only based on distance information, and no acceleration or velocity of the nodes is measured and utilized. So the state transition equation applied gets increasingly unreliable as the maximum speed increases. One possible solution is to decrease the period of iterations, which limits the state transition changes in each iteration. However, shorter interval requires larger bandwidth. MDS and SDP are not affected by increasing of maximum speed as they do not have state transition equations. CMDS and CSDP are more affected by irregular shape of clusters. DEKF and EKF still gets the most reliable results.

Table 5-5: Localization accuracy as maximum speed increases

Maximum Speed (/s)	0.1	0.2	0.3	0.4	0.5
DEKF-rank 3	0.041	0.034	0.044	0.112	0.116
EKF	0.039	0.032	0.042	0.108	0.114
CMDS-rank 3	0.108	0.194	0.28	0.291	0.446
MDS	0.086	0.096	0.072	0.057	0.092
CSDP-rank 3	0.163	0.193	0.288	0.268	0.387
SDP	0.146	0.215	0.186	0.164	0.147

5.3. Networks with Fewer Anchor Nodes

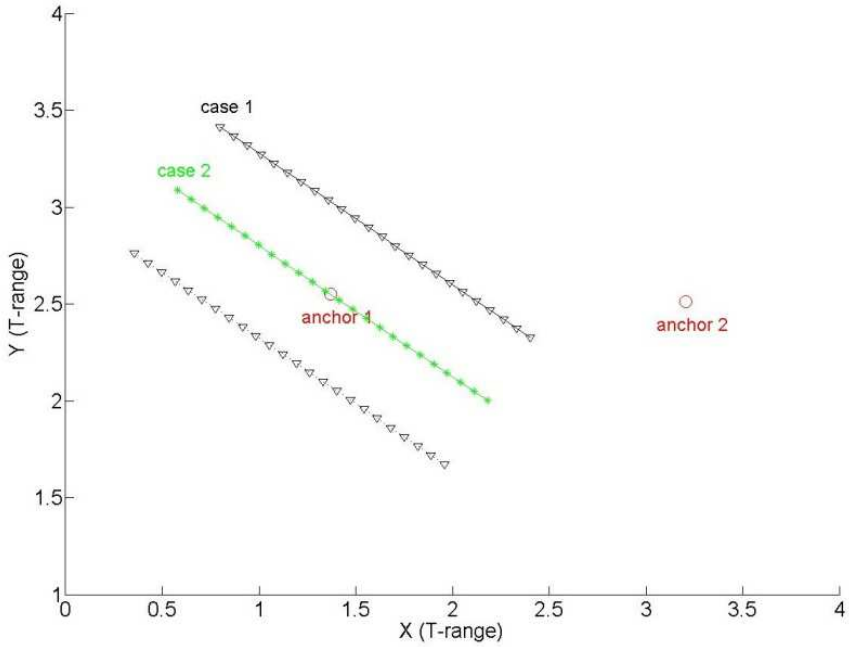
In the previous section, we have discussed that more than 2 anchor nodes are required in a fully connected wireless network for 2D localization purpose. However, that is based on the assumption that only the anchor nodes locations and the relative distances are known for location estimation. In this section, we investigate that mobility increases the localization potential which demands fewer anchor nodes and less connection to neighbours.

DR techniques [50][85] have been widely applied for localization purpose. There are two typical patterns of DR algorithms. One is that they return no absolute location, but the displacement compared with the starting point. The other is that within a short distance, the estimated displacement is quite accurate. Here we consider a scenario where the number of anchor nodes is reduced to one or two, and one moving node in the network is equipped with the DR technique through which its displacement is known. If the location of this node can be computed, this node becomes an anchor node and the localization of the other nodes becomes a solvable problem as demonstrated in the previous section. This is quite useful because in many scenarios like

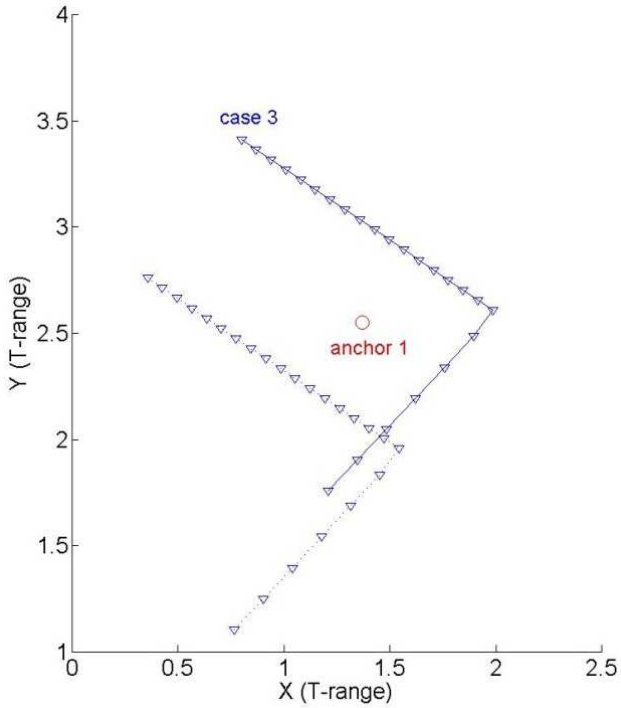
indoor localization using Wi-Fi access points, the deployment of the access points are always quite sparse and there may not be the sufficient minimum of three access points fully connected directly or indirectly. At the same time hand-held devices like smart phones are capable of deploying the DR algorithm. Solving this problem greatly improves the localization potential.

5.3.1. Case with Accurate Dead-Reckoning Estimation

In this section we discuss the case when accurate displacement is returned by the DR algorithm. Fig. 5.8 shows an example when a node passes by an anchor node. The subsequent locations are represented as dots. In case 1 shown in Fig. 5.8(a) when the node moves as a straight line, the estimated trajectory can be any fix-length parallel line to the real path because of the known displacement. The distance measurements to anchor 1 further refine the trajectory to two possible cases: The real one (case 1) and the one that is symmetrical to anchor node 1 (dotted line). In order to distinguish the real path in this case, the sensor needs the distance measurements from anchor 2. Unless the sensor moves in the same direction as the line linking anchor 1 and anchor 2, the ambiguity problem can be solved by the new measurements. Another choice is given in case 2 where the node moves directly toward anchor 1, in which case the symmetric trajectory is itself.



a) Case 1 & 2



b) Case 3

Fig. 5.8: Cases when a moving node passes by anchor nodes

Another choice is shown in Fig. 5.8(b) in which the sensor makes a turn when it is still in the transmission range of anchor 1. The difference in distance estimation after the turn will remove the wrong trajectory (dotted line). To sum up, the minimal anchor requirement for case 1 is two, while the requirement for case 2 and 3 is one.

a. Particle Filter

A straightforward method for the computation is by using the PF. The PF implements a recursive Bayesian filter using the Sequential Monte-Carlo method. It is particularly useful in solving non-linear and non-Gaussian problems. A set of random samples with weights, called particles, are used to represent the posterior probability density of the state, which is given by:

$$p(\mathbf{x}_n \mid \mathbf{z}_{1:n}) \approx \sum_{i=1}^{N_s} \omega_n^i \delta(\mathbf{x}_n - \mathbf{x}_n^i) \tag{5-12}$$

, where x_n and $z_{1:n}$ are the state and measurement, respectively, $\{\mathbf{x}_n^i, i=1,\ldots, N_s\}$ is a set of supporting particles at time n with the associated weights $\{\omega_n^i, i = 1,\ldots, N_s\}$. N_s are set to 100 by default.

The weight update equation is given by Eq. 5-13, where $g(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i, \mathbf{z}_n)$ is the importance density.

$$\omega_n^i \propto \omega_{n-1}^i \frac{p(\mathbf{z}_n \mid \mathbf{x}_n^i) p(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i)}{g(\mathbf{x}_n^i \mid \mathbf{x}_{n-1}^i, \mathbf{z}_n)} \tag{5-13}$$

In this specific case, when the node first gets into the transmission range of an anchor node, the initialization of the particles is executed. The particles are

generated around the circle with the radius of the measured distance with an equal weight ω_0.

The filter is executed after each second. During the prediction phase,

$$\mathbf{x}_n^i = \mathbf{x}_{n-1}^i + \mathbf{s}_n + \mathbf{n}_n^i \qquad \textbf{(5-14)}$$

where $\mathbf{x}_n^i$ and $\mathbf{x}_{n-1}^i$ are the 2D states at time n and time n-1, respectively, $\mathbf{s}_n$ is the measured displacement during time n. $\mathbf{n}_n^i$ is a random variable with Gaussian distribution.

The weights are updated by the distance measurements to the anchor nodes. For each of the measurements, there are two cases. One is that the sensor is within the transmission range of the anchor node j (with location $\mathbf{a}_j \in R^2$). Suppose the distance measurement is z_n^j and the distance from particle i to the anchor node is $\| x_n^i - \mathbf{a}_j \|$. In this case the probability is

$$p(z_n^j \mid \mathbf{x}_n^i) = \varphi(\| \mathbf{x}_n^i - \mathbf{a}_j \|) \qquad \textbf{(5-15)}$$

where $\varphi(x)$ is the Gaussian distribution of $N(z_n^j, noisefactor^2)$.

Another case happens when there is no distance measurement to anchor node j, which means the distance is larger than the transmission range (equals 1). The probability is

$$p(z_n^j \mid \mathbf{x}_n^i) = \begin{cases} \varphi'(\| \mathbf{x}_n^i - \mathbf{a}_j \|), & \| \mathbf{x}_n^i - \mathbf{a}_j \| \le 1 \\ 1, & \| \mathbf{x}_n^i - \mathbf{a}_j \| > 1 \end{cases} \qquad \textbf{(5-16)}$$

, where $\varphi'(x)$ is the Gaussian distribution of $N(1, noisefactor^2)$.

The weights are updated by the following equation:

$$\omega_n^i = \omega_{n-1}^i * \prod_{j=1}^{num\ of\ anchors} p(z_n^j \mid \mathbf{x}_n^i),\ i=1, 2, \ldots N_s \qquad \textbf{(5-17)}$$

The weights are then normalised by Eq. 5-18, and the estimated location is given by the weighted sum.

$$\omega_n^i = \omega_n^i / \sum_{i=1}^{N_s} \omega_n^i\ ,\ i=1, 2, \ldots N_s \qquad \textbf{(5-18)}$$

The degeneracy of the weights is unavoidable. Here we involve the idea of re-sampling which eliminates particles that have small normalised weights. A measure of degeneracy is introduced in [104] where the effective sample size N_{eff} is estimated by

$$N_{eff} \approx 1/\sum_{i=1}^{N_s} (\omega_n^i)^2\ ,\ i=1, 2, \ldots N_s \qquad \textbf{(5-19)}$$

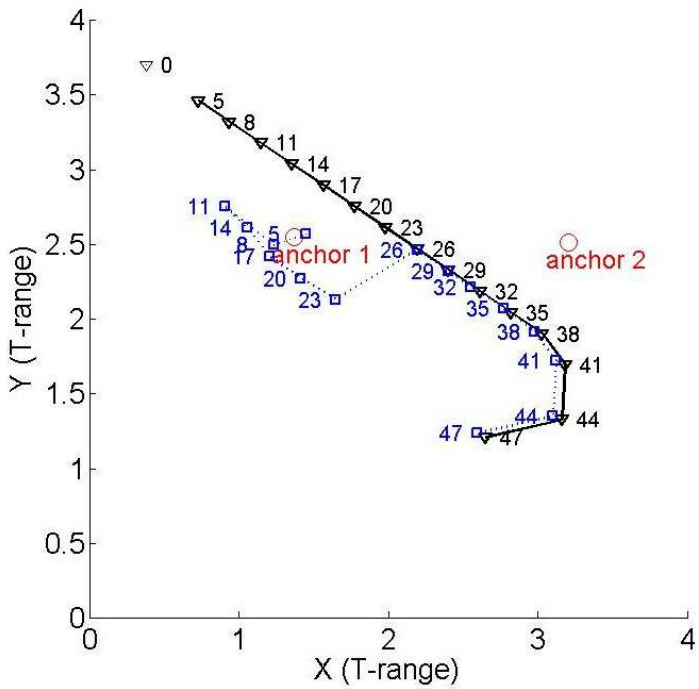
When N_{eff} is less than some threshold, the particles are re-sampled according to their weights. At the very beginning $\omega_n^i = 1/N_s$, then $1/\sum_{i=1}^{N_s} (\omega_n^i)^2 = N_s$. We set the threshold as $N_s/2$.

b. Evaluation

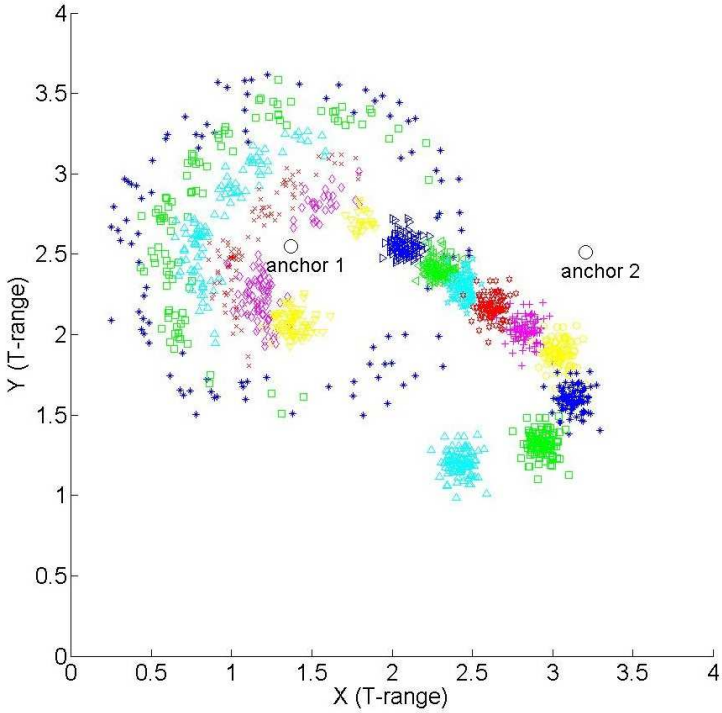
Figs 5.9-5.11 show the results for each of the cases. The (a) figures show the estimated trajectory compared to the real one. The triangles show the real locations at certain times. The squares show the estimated locations. The (b) figures show the respective particles at certain times and the particles at nearby steps are differentiated using different shapes. In Fig. 5.9 when the time is smaller than 5, the sensor does not receive any signal from the anchor node, so there is no location estimation. At time 5, the particles are generated around anchor 1 which results in the estimated location near anchor 1. As the

sensor moves, the particles are divided as two parts symmetric to anchor 1, which coincides with the description in Fig. 5.8(a). The symmetric problem is finally solved when the sensor moves into anchor 2's transmission range.

In Fig. 5.10 the particles shrink as the sensor moves towards anchor 1. In Fig. 5.11 there are two bunches of particles before the turn. The erroneous bunch of particles is removed after the turn.

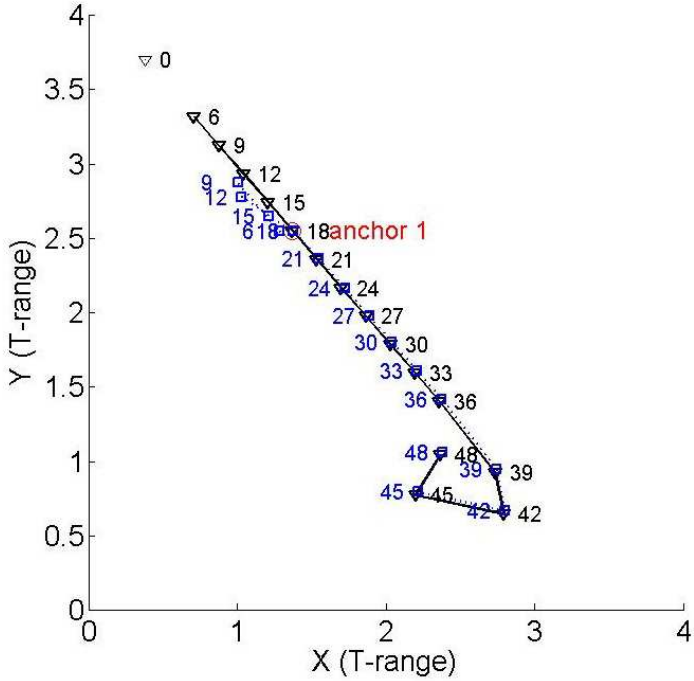


a) Trajectory

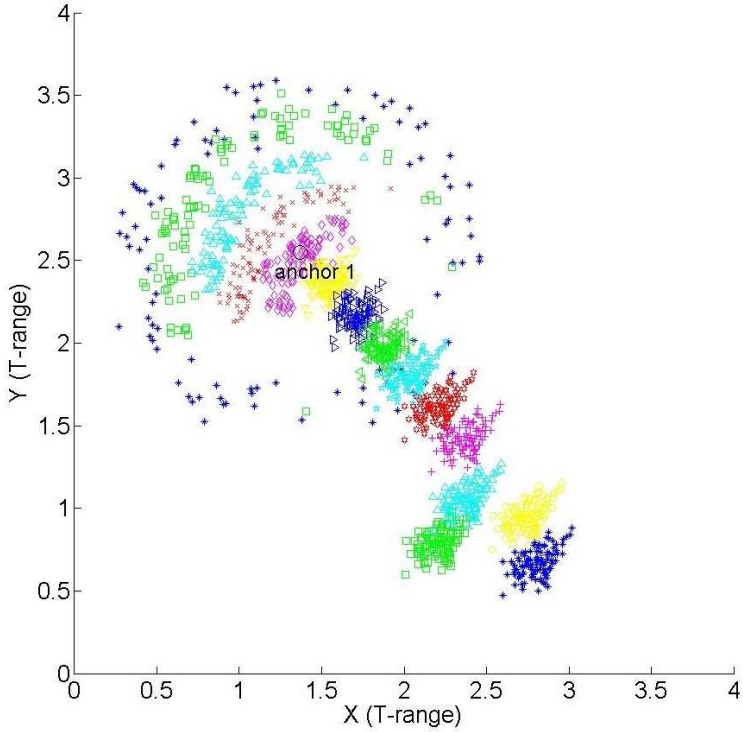


b) Particles

Fig. 5.9: Localization results for case 1

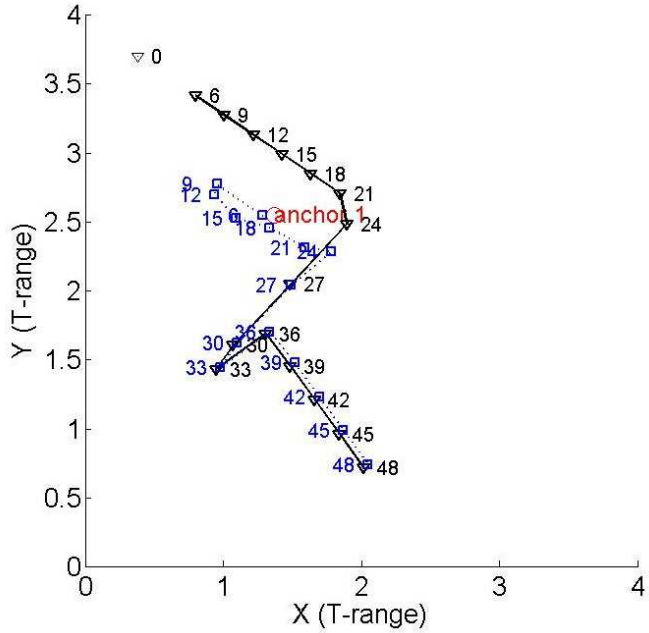


a) Trajectory

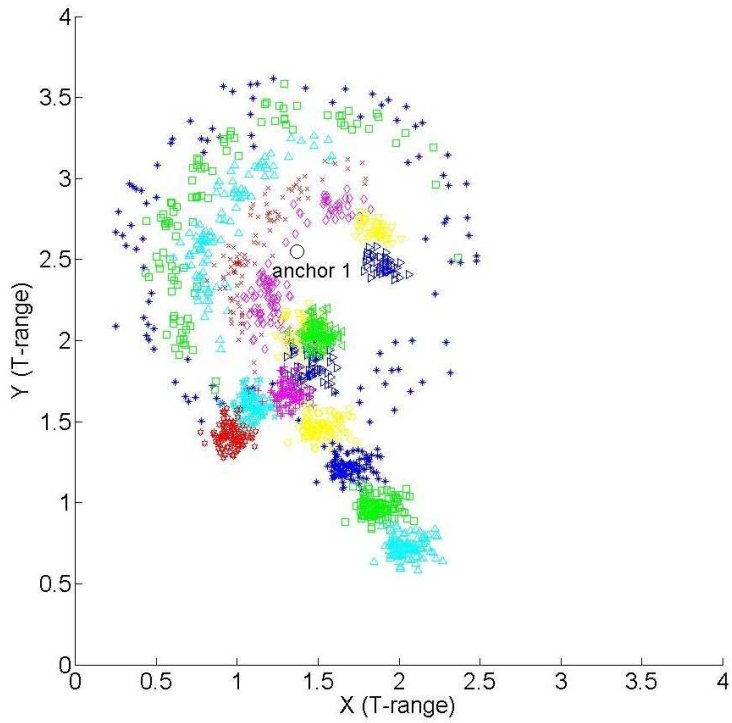


b) Particles

Fig. 5.10: Localization results for case 2



a) Trajectory



b) Particles

Fig. 5.11: Localization results for case 3

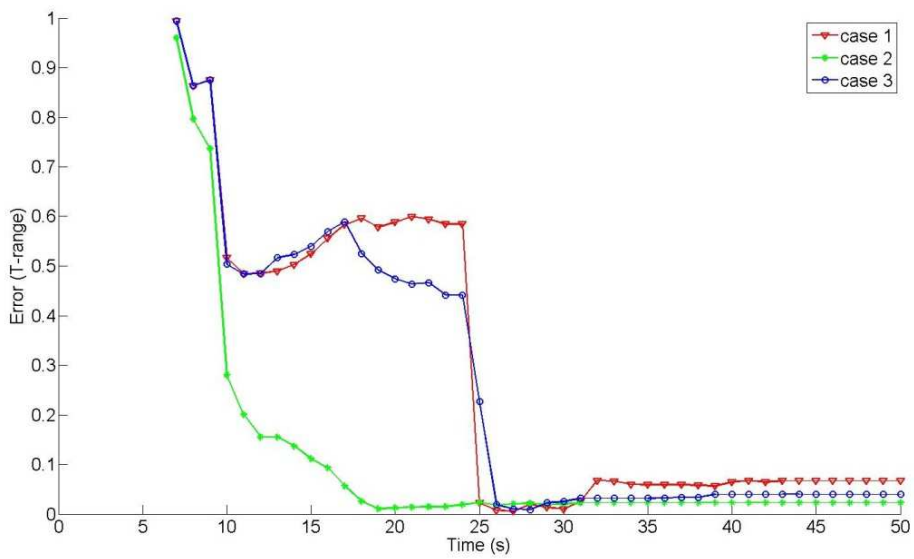


Fig. 5.12: Error in different time

Fig. 5.12 shows the convergence of the errors for the different cases. When the sensor first moves into the transmission range of an anchor node, the error drops tremendously. Except for case 2 in which the error directly decreases to

a stabilized value, in case 1 and case 3 the error drops for a second time when new information becomes available, irrespective of the distance estimation to another anchor in case 1 or turning in case 3.

5.3.2. Case with Inaccurate Dead-Reckoning Direction Estimate

When the dead-reckoning result is inaccurate, the problem becomes more challenging. It is discussed in [61] that the DR error mainly comes from the drift in direction estimation. The drift remains almost the same over a short distance. So the condition becomes that where the sensor's displacement length is known, and the direction is with some drift angle. The purpose of the algorithm is to estimate the sensor's location as well as the drift angle value.

In this scenario one anchor node cannot remove the effect of the drift angle which is shown in Fig. 5.13. The solid line represents the real trajectory. Because of the 20-degree error in direction estimation, the estimated trajectory would be the dotted line. Two anchor nodes are required in this scenario.

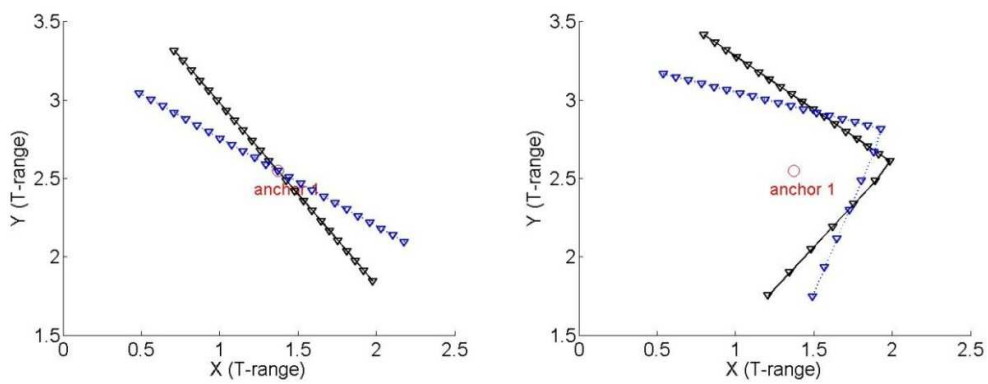


Fig. 5.13: Illustration when there is error in direction estimation

There is difference in the design of the PF. When the node first gets into the transmission range of an anchor node, the particles are generated in the same

way. Besides the weight ω_0, each particle i is also associated with a random angle b^i which tries to compensate for the drift angle.

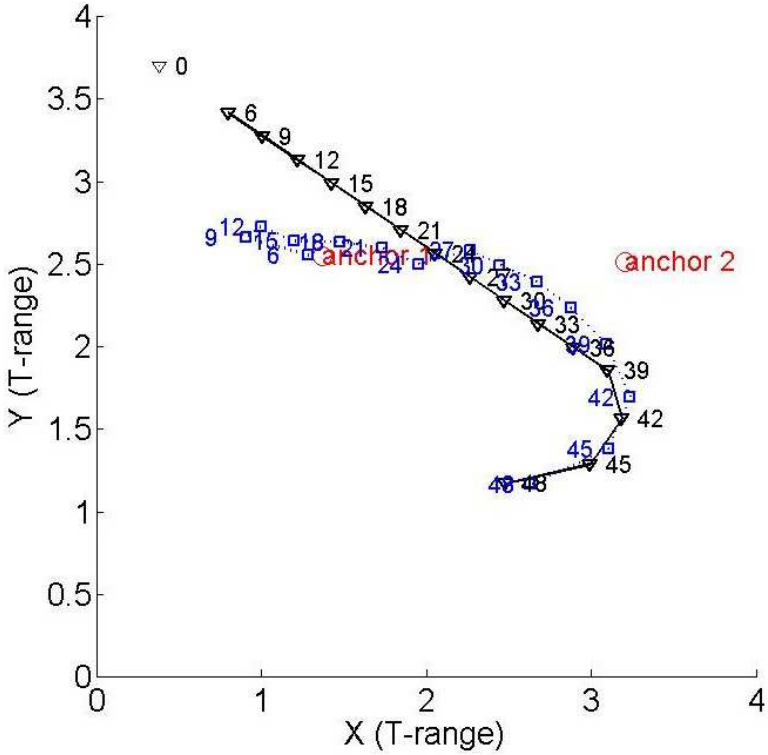
The prediction equation is given by

$$\begin{aligned} angle &= d_{angle,n} + b^i_{d,n} + n^i_{d,n} \\ \mathbf{x}^i_n &= \mathbf{x}^i_{n-1} + (s_n + n^i_{s,n})[\cos(angle); \sin(angle)] \end{aligned} \quad i = 1,\ 2,\ \ldots N_s \qquad \textbf{(5-20)}$$

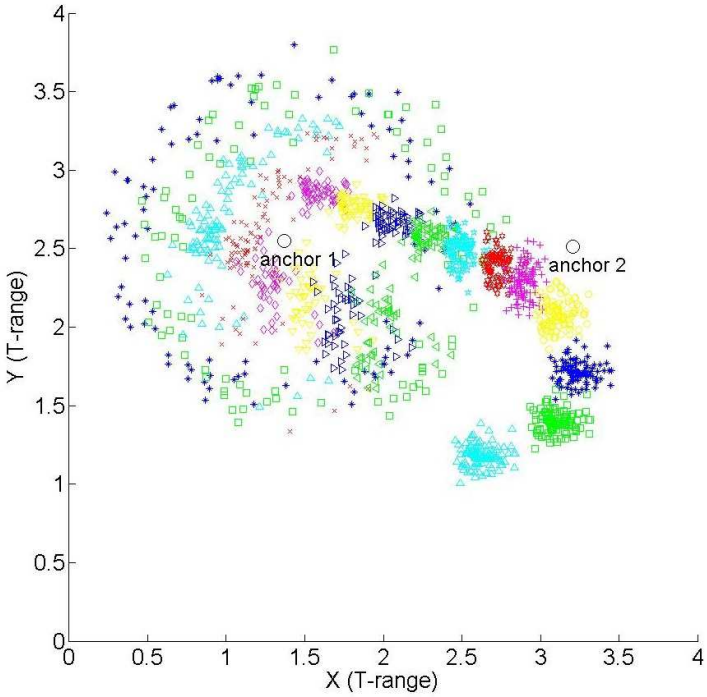
where s_n and $d_{angle,n}$ are the displacement length and angle estimation, respectively. $n^i_{s,n}$ and $n^i_{d,n}$ represent the noise in length and direction estimation, respectively. b^i is the drift compensation angle for particle i.

The importance sampling and re-sampling algorithm remains the same. Hopefully the particles with the right locations as well as the right direction compensations have higher weights. An example of the results is shown in Fig. 5.14 when 100 particles are generated.

In this scenario, the particles represent the possible locations and drift compensation angle at the same time. The particles with the right locations and drift compensation angle at the same time are preserved after filtering. When only 100 particles are generated, there is high possibility that there is no such particle being generated. The erroneous particles will not return accurate results. The solution is to generate more particles so that all the possibilities can be represented. We compare the performances when 100 particles and 1000 particles are generated. The results are shown in Table 5-6 when the sensor has different direction drift angles.



a) Trajectory



b) Particles

Fig. 5.14: Localization results for case 1 with inaccurate direction estimate

Table 5-6: Performance when different number of particles are used

Direction drift (°)	**-20**	**-15**	**-10**	**-5**	**5**	**10**	**15**	**20**
N_s=100 (error)	0.03	0.03	0.09	0.07	0.10	0.16	0.13	0.73
N_s=100 (compensation angle °)	16.0	15.9	11.8	7.10	1.58	1.72	1.42	19.1
N_s=1000 (error)	0.05	0.02	0.04	0.03	0.05	0.03	0.05	0.06
N_s=1000 (compensation angle °)	21.7	15.0	10.1	6.5	-2.2	-8.9	-12.2	-16.9

The compensation angle in the table is the weighted average of b^i. When this value plus the direction drift equals zero, the direction drift is totally compensated and accurate direction estimation is obtained. It can be observed that when the direction drift is larger than zero, the returned compensation angle when N_s=100 is quite erroneous, which results in large localization error. When N_s=1000, it returns more accurate and robust results in different direction drift angles. So in the case of the existing error in the direction estimation, it is much safer to use the PF with a large particle number.

5.4. Enhanced Localization in Sensor Network by Dead-Reckoning

In order to demonstrate the advantage of using DR in a sparse or a network with fewer anchors, we use almost the same settings as applied in Section 5.2.2 a): the 50-node network is in a 4 by 4 map (T_{range}=1), the *noisefactor* is set to 0.05, and the bandwidth is set to 1Mbps. The period of one iteration is 1 second. The only difference is that one non-anchor node (Node 10) is assigned with the capability to estimate the accurate displacement, and it is randomly moving within the 4 by 4 map. We test the performance when the number of

anchor nodes is decreased from 5 to 2. The algorithms are run for 100 seconds. The location error is represented as a multiple of T_{range}.

5.4.1. Distributed Localization

From the discussion of last section, we know that Node 10 will obtain its location as it approaches the anchor nodes. Once the location of Node 10 is calculated, it becomes a moving anchor node which helps to localize the neighbour nodes by multi-lateration. The localized neighbour nodes then become the new anchors to localize their neighbours. In the end, each node keeps updating its location using the neighbours' location. In this way, the location solution spreads to the whole network.

Fig. 5.15 shows how Node 10 can help to localize the nodes when there are only two anchor nodes existing in the network, which is impossible without the DR technique. The horizontal axis represents the time, and the left and right vertical axes represent the average location error and the number of localized nodes respectively. From time 0 to 8, no node is localized as Node 10 has not localized itself. The line in the bottom represents the number of localized nodes. At time 9, Node 10 localizes itself with accuracy of 0.078. The accuracy fluctuates as more measurements are available. But overall, the accuracy has improved and reaches 0.006 at time 16. From time 17, Node 10 starts to localize other nodes. It shows that as Node 10 moves around, more nodes are localized. As we put high criteria on the relative location of anchor nodes to ensure a certain accuracy level, the number does not increase dramatically. But it is predictable that given enough time, Node 10 can localize all the nodes in the network with good accuracy. Another choice is

that after Node 10 has localized a certain amount of nodes, the results can be fed into the cluster based method to compute the location for the rest of the nodes. This will be covered in the next section.

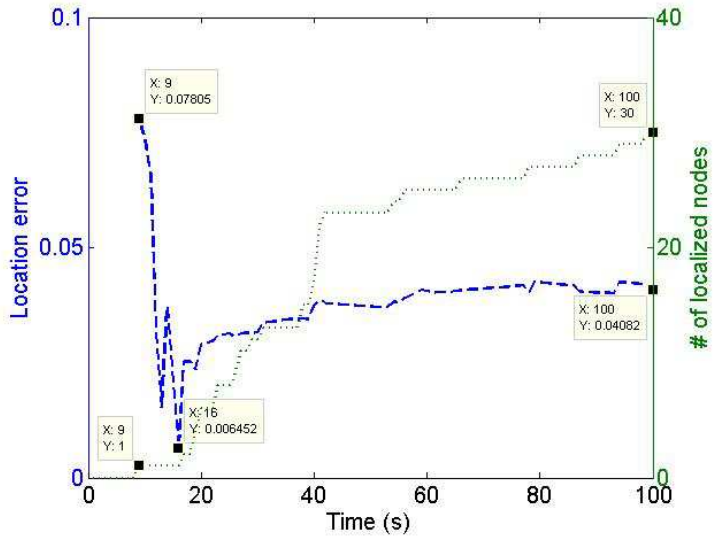


Fig. 5.15: Iterations of localization results

5.4.2. Enhanced Cluster Based Method

In Section 5.2 when we discuss the cluster based method, the main drawback of CMDS and CSDP has been the high requirement on the number of anchor nodes. By combining the DR algorithm, the small number of anchor nodes is no longer a constraint, as the nodes localized in Section 5.4.1 can be the pseudo anchors. At each iteration, some of them are chosen as the anchors to calibrate the location for the rest. The result is shown in Table 5-7. For each algorithm, we show the localization error and the number of localized nodes at given number of anchors.

By combining the DR with the cluster based methods, more nodes can be localized during the same time period, which reaches the total amount of non-

anchor nodes. The number of anchor nodes does not show any effect on the performance because of the moving Node 10. The performance of CMDS improves the most, which achieves high accuracy with only 0.05 T_{range} error. For the DEKF and CSDP, although the number of localized nodes increases compared with the original methods without DR, the accuracy does not improve.

Table 5-7: Relationship of anchor nodes on accuracy

No. of Anchors	***DR***		***DEKF-rank 3***		***CMDS-rank 3***		***CSDP-rank 3***	
	Error	*# of Nodes*	Error	*# of Nodes*	Error	*# of Nodes*	Error	*# of Nodes*
2	0.04	30	0.17	48	0.05	48	0.32	48
3	0.05	30	0.17	47	0.11	47	0.34	47
4	0.05	31	0.16	46	0.11	46	0.33	46
5	0.06	33	0.08	45	0.05	45	0.29	45

Chapter 6

Conclusions and Future Work

In this thesis, we developed a variety of techniques appropriate for a pervasive indoor localization system. We investigated the IMU sensor based DR algorithm, which provides quite a robust location solution, without the requirement on costly infrastructure deployment. We improved the performance based on our MM and the improved PF scheme. We also explored the algorithm to fuse the two sensors to provide a more robust solution, especially for the orientation estimation. To provide a pervasive location solution to devices without such hardware, we studied the scenario of cooperative localization. The cost of clustering was quantified and different algorithms were compared. We demonstrated that various complementary fusion mechanisms to the DR can significantly enhance the overall performance.

6.1. Step-Counting with Map Fusion

We proposed methods to improve the performance of the step-counting algorithm. Three improvements were made: an adaptive step direction estimation method; an MM algorithm, and an improved PF.

The step direction estimation is one of the key procedures for step-counting based DR tracking using inertial sensors. It is also quite challenging, especially when the captured motion data is tainted by the user's activity. The PCA based algorithm has provided robust estimation results, regardless of the sensor's relative rotation compared to the human body. However, the PCA based algorithm only returns the principal axis, resolving the 180° ambiguity is another challenge. The drawback of the PCA is compensated with the sensor's orientation analysis, which returns the walking direction by analysing changes in sensor's orientation.

In our adaptive method, the sensor's orientation analysis algorithm is executed when a direction change is detected by the PCA algorithm. Because of the low computational complexity and the restricted usage of the orientation analysis, the adaptive method is preferred, as it introduces little overhead compared to the original PCA method. The experimental results show that the adaptive algorithm provides more robust and accurate results, when compared to the original PCA algorithm.

To compensate for the accumulating error in a DR tracking system, extra techniques are always fused together to form a hybrid system. Currently, various PF based map filtering algorithms have been proposed. We proposed a MM algorithm that calibrates not only the location, but also the sensor's

orientation and step direction, which reduces the error in the location update equation. We then combined MM with the PF, so that a more robust algorithm is proposed.

To relax the requirements on corridor information, an improved PF is also proposed, which enhances the step direction estimation without the requirement on corridors. The experimental results illustrate that in a quite dense map constraint environment with corridors, the improvement is not obvious. But when only partial map constraints are applied, the MM enabled PF and the improved PF achieve more robust and accurate results. The improved PF scheme outperforms the other schemes with less performance dependence on corridor constraints.

Because of the limitations of the DR scheme requiring an initial location and orientation estimation, we suggest that an area of future research be on an automatic reshaping trajectory based on the map constraints. The DR algorithm returns a quite accurate displacement trajectory in a short period of time. So, by a limited expansion and rotation of the originally returned trajectory, the new trajectory should fit into the map constraints. As the pedestrian moves, the location accuracy and the direction estimation should be improved.

6.2. Dual Sensor Fusion

A dual sensor's orientation fusion method is proposed in this thesis. Most of the time, the fused method has a higher orientation accuracy than that of Sensor *A* and Sensor *B*, while in some cases, an accuracy in between that of Sensor *A* and Sensor *B* are obtained. The in between accuracy is closer to the

individual result with less error. In real use cases because the user will not know which sensor gives better results, the proposed fused method is more reliable.

In order to show the effect of the orientation estimation on location tracking, the fused orientation is fed into the DR algorithm. Compared with the original individual DR, the methods with orientation fusion obtain higher location accuracy. But when we further apply the location fusion algorithm on top of the orientation fusion, it makes no accuracy difference, when compared to the one where only the location fusion is applied. This is because fusing the location indirectly fuses the orientation, as the orientation estimation is one input for the DR location estimation. The primary advantage of this orientation fusion method is in providing more robust orientation estimation for rotation tracking use cases.

The current solution occasionally returned results which may not have the highest accuracy. This may not meet the needs for higher accuracy. The major reason for this is that the two sensors are modelled with identical static error noise, which does not reflect the error correctly. If another data source is available to calibrate the error from individual sensors, we may be able to model the noise better before the fusion process. Incorporating the indoor map could be a way to identify the sensor with higher accuracy at specific times and give it higher weight. In the end, higher accuracy should be achieved.

6.3. Cooperative Localization

In this thesis, we evaluated three algorithms under cluster based environments. Using an efficient distributed clustering algorithm, the network

is divided into clusters to simplify the computational complexity and control the overhead traffic. In our simulations, we compare the performances of EKF, MDS, SDP, and their cluster based methods DEKF, CMDS and CSDP, respectively. We also discuss that the DR technique relaxes the connection requirement, as well as the anchor number requirement for localization. PF is applied for this DR enabled localization.

It is illustrated that as the size of the cluster increases, a higher bandwidth is required for control and signalling. The centralized methods require the highest bandwidth. We also determined that DEKF achieves nearly the same performance as EKF, which means that the DEKF achieves the same performance with less cost. Compared with CMDS and CSDP, the DEKF requires fewer anchor nodes, and a smaller cluster size; yet, it provides more accurate localization results. Therefore, we recommend to using the DEKF for cluster based localization.

The proposed DEKF is suitable for low mobility environments. In a high mobility environment, the algorithm should operate either with smaller iteration intervals (higher bandwidth required), or incorporate a speed estimation mechanism. Such recommended future work on cooperative localization will be useful for enhancing location awareness between smart mobile devices.

The analysis and simulations show that depending on the movement of the node towards the proximity to the anchor nodes, at least 1 anchor is enough for localization when an accurate DR result is available. When there is error in the direction estimation, two anchor nodes will be required. It is also found

that a large particle number (e.g. 1000) is preferred for robust performance; this will increase the computational complexity. The localized mobile node would become a moving anchor, which helps to localize the rest of the nodes in the network.

The simulation results illustrate that more nodes can be localized, compared with the scenario without DR. The output is then fed into the original cluster based method, which improves the performance in terms of the number of localized nodes and the location accuracy when there are fewer anchor nodes. CMDS can achieve impressive accuracy with only a 0.05 T_{range} error.

Simulations were used to evaluate the current research. Further improvement can be achieved if a real test bed can be set up, including the set up of Wi-Fi access points, development of mobile application running the DR algorithm, and the implementation of inter-device communication.

Bibliography

[1] Y. B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks," *Wireless Networks*, vol. 6, issue 4, pp. 307-321, Jul. 2000.

[2] V. Otsason, A. Varshavsky, A. LaMarca, and E. D. Lara, "Accurate GSM indoor localization," in *Proc. UbiComp*, Tokyo, Japan, pp. 141–158, Sept. 11-14, 2005.

[3] K. Pahlavan, X. Li, and J. P. Makela, "Indoor geolocation: science and technology," IEEE Communications Magazine, vol. 40, pp. 112-118, Feb. 2002.

[4] T. Manodham, L. Loyola, and T. Miki, "A novel wireless positioning system for seamless internet connectivity based on the WLAN infrastructure," *Wireless Personal Communications*, vol. 44, issue 3, pp 295-309, Feb. 2008.

[5] H.H. Lin, C.C. Tsai, and J. C. Hsu, "Ultrasonic localization and pose tracking of an autonomous mobile robot via fuzzy adaptive extended information filtering," *IEEE Trans. Instrumentation and Measurement*, vol. 57, issue 9, Sept. 2008.

[6] K. Wendlandt, M. Berbig, and P. Robertson, "Indoor localization with probability density functions based on Bluetooth," in *Proc. IEEE PIMRC*, Berlin DE, vol.3, pp. 2040-2044, 11-14 Sept. 2005.

[7] Mautz, R, "Overview of current indoor positioning systems," Geod. Cartography 2009, vol. 35, issue 1, pp. 18–22.

[8] S. A. Golden and S. S. Bateman, "Sensor measurements for Wi-Fi location with emphasis on time-of-arrival ranging," IEEE Trans. Mobile Computing, vol. 6, pp. 1185-1198, Oct. 2007.

[9] M. Ciurana, F. Barcelo-Arroyo, and F. Izquierdo, "A ranging method with IEEE 802.11 data frames for indoor localization," in Proc. IEEE WCNC, pp. 2092-2096, Mar. 2007.

[10] C.Wong, R. Klukas, and G. Messier, "Using WLAN infrastructure for angle-of-arrival indoor user location," In Proc. of the IEEE VTC Conf., pages 1–5, Sept. 2008.

[11] J. Xiong and K. Jamieson, "ArrayTrack: a fine-grained indoor location system," In Proc. of 10th NSDI, Lombard, IL, 2-5 Apr 2013

[12] M. Brunato and R. Battiti, "Statistical learning theory for location fingerprinting in wireless LANs," Computer Networks, vol. 47, pp. 825-845, Apr. 2005.

[13] Teemu Roos, Petri Myllymäki, Henry Tirri, Pauli Misikangas and Juha Sievänen, "A probabilistic approach to WLAN user location estimation," International Journal of Wireless Information Networks, vol. 9, pp. 155-164, Jul. 2002.

[14] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," In Proc. of the ACM Int. Conf. on Mobile Computing and Networking (MobiCom), Istanbul, Turkey, pp. 269-280, 22-26 Aug 2012.

[15] C. Wu, Z. Yang, Y. Liu, and W. Xi, "WILL: Wireless indoor localization without site survey," In Proceedings of IEEE INFOCOM, Orlando, Fl, pp. 64-72, 25-30 Mar 2012.

[16] A. Rai, K. Chintalapudi, V. Padmanabhan, and R. Sen, "Zee: Zero-effort crowdsourcing for indoor localization," In Proc. of ACM MobiCom, Istanbul, Turkey, pp. 293-304, 22-26 Aug 2012.

[17] Y. Jin, W. S. Soh, M. Motani, and W. C. Wong, "A robust indoor pedestrian tracking system with sparse infrastructure support," *IEEE Transaction on Mobile Computing*, vol. 12, issue 7, pp 1392-1403, May. 2012.

[18] M. Kourogi and T. Kurata, "Personal positioning based on walking locomotion analysis with self-contained sensors and a wearable camera," in *Proc. IEEE and ACM Int. Symp. Mixed and Augmented Reality*, Tokyo, Japan, pp. 103-112, 7-10 Oct. 2003.

[19] Q. Ladetto, "On foot navigation: continuous step calibration using both complementary recursive prediction and adaptive Kalman filtering," in *Proc. ION GPS 2000*, Salt Lake City, Utah, pp. 1735-1740, Sept. 19-22 2000.

[20] J. W. Kim, H. J. Jang, D. H. Hwang and C. Park, "A step, stride and heading determination for the pedestrian navigation system," in *Proc. Int. Symp. GNSS/GPS*, Sydney, Australia, vol.3, issue 1-2, pp. 273-279, 6–8 Dec. 2004.

[21] B. Krach and P. Roberston, "Cascaded estimation architecture for integration of foot-mounted inertial sensors," in *Proc. IEEE/ION PLANS*, Monterey, California, USA, pp. 112-119, May 2008.

[22] M. Klepal and S. Beauregard, "A backtracking particle filter for fusing building plans with PDR displacement estimates," in *Proc. WPNC*, Hannover, pp. 207-212, 27 Mar. 2008.

[23] F. Ichikawa, J. Chipchase and R. Grignani, "Where's the phone? a study of mobile phone location in public spaces," in *Proc. 2nd Int. Conf. Mobile Technology, Applications and Systems*, Guangzhou, pp. 1-8, 15-17 Nov. 2005.

[24] M. H. Afzal , V. Renaudin, and G. Lachapelle, "Use of earth's magnetic field for mitigating gyroscope errors regardless of magnetic perturbation," *Sensors*, vol. 11, issue 12: 11390-11414, Nov. 2011.

[25] M. H. Afzal, V. Renaudin, and G. Lachapelle, "Assessment of indoor magnetic field anomalies using multiple magnetometers," *ION GNSS*, Session F1, Portland, Oregon, pp. 525-533, 21-24 Sept. 2010.

[26] Z. Yang, Y. Liu, and X. Li, "Beyond trilateration : on the localizability of wireless ad hoc networks," IEEE/ACM Trans. Netw., vol. 18, no. 6, pp. 1806-1814, Dec.
2010.

[27] L. Doherty, K. S. J. Pister and L. E. Ghaoui, "Convex position estimation in wireless sensor networks," in *Proc. IEEE INFOCOM*, vol. 3, pp. 1655-1663, Apr. 2001.

[28] J. Shlens, “A tutorial on principal component analysis,” Dec 2005, available at http://www.cs.cmu.edu/~elaw/papers/pca.pdf, link is tested in Dec 2014.

[29] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” IEEE Trans. Signal Processing, vol. 50, pp. 174-188, Feb. 2002.

[30] H. Bao, W.C. Wong and T.T. Tay, “Cluster based localization algorithm in wireless networks,” in *Proc. IEEE ICCS*, Singapore, pp. 458-462, 21-23 Nov, 2012.

[31] M. Laaraiedh, L. Yu, S. Avrillon, and B. Uguen, “Comparison of Hybrid Localization Schemes using RSSI, TOA, and TDOA,” in Proc. 17th European Wireless Conference, Apr. 2011.

[32] T. S. Rappaport, “Wireless communications: principles and practice,” Prentice Hall, January 2002.

[33] F. Barceló, J. Paradells, E. Zola, F. Izquierdo, and M. Ciurana, “Performance evaluation of a TOA-based trilateration method to locate terminals in WLAN,” Proceedings of International Symposium on Wireless Pervasive Computing, pp. 1-6, January 2006.

[34] E. Robinson and A. H. Quazi, “Effect of sound-speed profile on differential time-delay estimation,” J. Acoust. Soc. Am., vol. 77, no. 3, pp. 1086-1090, Mar. 1985.

[35] I. Güvenc and C.-C. Chong, “A survey on TOA based wireless localization and NLOS mitigation techniques,” IEEE Commun. Surveys Tuts., vol. 11, no. 3, pp. 107-124, quarter 2009.

[36] A. Urruela, J. Sala, and J. Riba, “Average performance analysis of circular and hyperbolic geolocation,” Vehicular Technology, IEEE Transactions on, vol. 55, no. 1, pp. 52 – 66, Jan. 2006.

[37] P. Bahl and V. Padmanabhan, “RADAR : an in-building RF-based user location and tracking system,” in Proc. IEEE International Conference on Computer Communications, INFOCOM'00, vol. 2, 2000, pp. 775-784.

[38] H. Laitinen, J. Lahteenmaki, and T. Nordstrom, “Database correlation method for GSM location,” in Proc. IEEE Vehicular Technology Conference, VTC Spring 2001, vol. 4, 2001, pp. 2504-2508.

[39] M. Brunato and R. Battiti, “Statistical learning theory for location fingerprinting in wireless LANs,” Computer Networks, vol. 47, pp. 825-845, Apr. 2005.

[40] B. Ferris, D. Hähnel, and D. Fox, “Gaussian processes for signal strength-based location estimation,” in Robotics : Science and Systems, G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, Eds. The MIT Press, 2006.

[41] P. Myllymäki, T. Roos, H. Tirri, P. Misikangas, and J. SievÄanen, “A probabilistic approach to WLAN user location estimation,” International Journal of Wireless Information Networks, vol. 9, pp. 155-164, Jul. 2002.

[42] A. Kushki, K. N. Plataniotis, and A. N. Venetsanopoulos, “Kernel-based positioning in wireless local area networks,” IEEE Trans. Mobile Computing, vol. 6, pp. 689-705, Jun. 2007.

[43] S. Sun, X.L. Meng, L.Y. Ji, J.K. Wu, and W.C. Wong, "Adaptive Sensor Data Fusion in Motion Capture," *Information Fusion (FUSION)*, Edinburgh, pp. 1-8, 26-29 Jul. 2010.

[44] James Richard Wertz, "Spacecraft attitude determination and control," Springer, 1978, Publication Date: December 31, 1980 | ISBN-10: 9027712042 | ISBN-13: 978-9027712042.

[45] Kim, A. and Golnaraghi, M.F, "A quaternion-based orientation estimation algorithm using an inertial measurement unit," Position Location and Navigation Symposium, 2004. PLANS 2004, pp. 268-272, 26-29 April 2004.

[46] Kraft, E., "A Quaternion-based Unscented Kalman Filter for Orientation Tracking," Information Fusion, 2003. Proceedings of the Sixth International Conference of (Volume:1), pp. 47-54, 8-11 July 2003, Cairns, Queensland, Australia.

[47] X. Li, Q. Zhou, S. Lu, and H. Lu, "A new method of double electric compass for localization in automobile navigation," in *Proc. 2006 IEEE Int. Conf. Mechatronics and Automation*, Luoyang, China, pp. 514-519, Jun. 25 - 28, 2006.

[48] D. Roetenberg, H. J. Luinge, C. T. M. Baten, and P. H. Veltink, "Compensation of magnetic disturbances improves inertial and magnetic sensing of human body segment orientation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 13, no. 3, pp. 395-405, Sept. 2005.

[49] D. Roetenberg, C. T. M. Baten, and P. H. Veltink, "Estimating body segment orientation by applying inertial and magnetic sensing near ferromagnetic materials," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 15, no. 3, pp. 469-471, Sept. 2007.

[50] S.W. Lee and K. Mase, "Activity and location recognition using wearable sensor," *IEEE Pervasive Computing*, vol. 1, pp. 24-32, July-Sept. 2002.

[51] K. Kunze, P. Lukowicz, K. Partridge, and B. Begole, "Which way am I facing: inferring horizontal device orientation from an accelerometer signal," in *Proc. Int. Symp. Wearable Computers*, Linz, Austria, pp. 149-150, 4-7 Sept. 2009.

[52] U. Blanke and B. Schiele, "Sensing location in the pocket," in *Adjunct Poster Proc. UbiComp'08*, Seoul, South Korea, pp.1-2, Jan. 2008.

[53] U. Steinhoff and B. Schiele, "Dead reckoning from the pocket - an experimental study," *8th Ann. IEEE Int. Conf. Pervasive Computing and Communications*, Mannheim, Germany, pp 162-170, Apr. 2010.

[54] D. Alvarez, R.C. Gonzalez, A. Lopez, and J.C. Alvarez, "Comparison of step length estimators from wearable accelerometer devices," *in. Proc. IEEE Eng. Med. Biol. Soc.* , 1:5964-7, 2006.

[55] H.J. Jang, J.W. Kim, and D.H. Hwang, "Robust step detection method for pedestrian navigation systems," *Electronics Letters*, vol. 43 issue: 14, Jul. 5 2007.

[56] P. Kemppi, T. Rautiainen, V. Ranki, F. Belloni, and J. Pajunen, "Hybrid positioning system combining angle-based localization, pedestrian dead reckoning and map filtering," *Int. Conf. Indoor Positioning and Indoor Navigation (lPIN)*, ZUrich, Switzerland, pp. 1-7, 15-17 Sept. 2010.

[57] P. Davidson, J. Collin, and J. Takala, "Application of particle filters for indoor positioning using floor plans," *Ubiquitous Positioning Indoor Navigation and*

Location Based Service (UPINLBS), pp.1-4, ISBN: 978-1-4244-7880-4, Dec. 2010.

[58] E. Vildjiounaite, E. J. Malm, J. Kaartinen, and P. Alahuhta, "Location estimation indoors by means of small computing power devices, accelerometers, magnetic sensors, and map knowledge," *In Pervasive Computing*, vol. 2414, pp. 211-224, 2002.

[59] I. Spassov, M. Bierlaire, and B. Merminod, "Map matching for pedestrians via bayesian inference," *European Navigation Conf. – Global Navigation Satellite Systems*, Manchester, UK, 7-10 May 2006.

[60] M. A. Quddusa, W. Y. Ochiengb, and R. B. Nolandb, "Current map-matching algorithms for transport applications: State-of-the art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 15, issue 5, pp. 312–328, Oct. 2007.

[61] H. Bao and W.C. Wong, "An Indoor Dead-Reckoning Algorithm with MM," in *Proc. IEEE IWCMC 2013*, Cagliari, Italy, July 1-5, 2013.

[62] P. Biswas, T. C. Lian, T. C. Wang, and Y. Ye, "Semidefinite programming based algorithms for sensor network localization," ACM Transactions on Sensor Networks, vol. 2, no. 2, pp. 188–220, May 2006.

[63] C. Savarese, J. M. Rabaey and J. Beutel, "Localization in distributed ad-hoc wireless sensor networks," in *Proc. IEEE ICASSP*, Salt Lake City, USA, pp. 2037-2040, May 2001.

[64] S. Severi, G. Abreu, and D. Dardari, "A quantitative comparison of multi-hop localization algorithms," in Proc. IEEE Workshop on Positioning Navigation and Communication, WPNC'010, Mar. 2010, pp. 200-205.

[65] A. Savvides, W. Garber, S. Adlakha, R. Moses, and M. B. Srivastava, "On the error characteristics of multi-hop node localization in ad-hoc sensor networks," in *Proceedings of The 2th International Conference on Information Processing in Sensor Networks*, Palo Alto, USA, pp. 317-332, 2003.

[66] N. Patwari and P. Agrawal, "Effects of correlated shadowing : connectivity, localization, and RF tomography," in Proc. IEEE International Conference on Information Processing in Sensor Networks, IPSN'08, Apr. 2008, pp. 82-93.

[67] N. Patwari and I. Hero, A.O., "Manifold learning algorithms for localization in wireless sensor networks," in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'04, May 2004, pp. 857-860.

[68] R. Moses, D. Krishnamurthy, and R. Patterson, "A self-localization method for wireless sensor networks," Eurasip J. on Applied Signal Processing, Special Issue on Sensor Networks, vol. 2003, no. 4, pp. 148-158, 2003.

[69] N. Patwari, I. Hero, A.O., M. Perkins, N. Correal, and R. O'Dea, "Relative location estimation in wireless sensor networks," IEEE Trans. Signal Process., vol. 51, no. 8, pp. 2137-2148, Aug. 2003.

[70] H. Noureddine, D. Castelain, and R. Pyndiah, "Cooperative network localizability via semidefinite programming," in Proc. IEEE Personal, International Symposium on Indoor and Mobile Radio Communications, PIMRC'011, Sep. 2011.

[71] H. Noureddine, N. Gresset, D. Castelain, and R. Pyndiah, "A new variant of nonparametric belief propagation for self-localization," in Proc. IEEE Int. Conf. on Tele- communications, ICT'010, Apr. 2010, pp. 822-827.

[72] C. Pedersen, T. Pedersen, and B. Fleury, "A variational message passing algorithm for sensor self-localization in wireless networks," in Proc. IEEE International Symposium on Information Theory, Jul. 2011.

[73] V. Savic, H. Wymeersch, F. Penna, and S. Zazo, "Optimized edge appearance probability for cooperative localization based on tree-reweighted nonparametric belief propagation," in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'011, May 2011, pp. 3028-3031.

[74] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from Mere Connectivity," in Proc. ACM International Symposium on Mobile Ad Hoc Networking & Computing, 2003, pp. 201-212.

[75] Y. Shang and W. Ruml, "Improved MDS-based localization," *IEEE INFOCOM*, vol. 4, pp. 2640-2651, March 2004.

[76] Gwo-Jong Yu and Shao-Chun Wang, "A hierarchical MDS-based localization algorithm for wireless sensor networks," *22nd International Conference on Advanced Information Networking and Applications*, ISBN: 978-0-7695-3095-6, March 25 2008.

[77] Jian Shu, Ronglei Zhang, Linlan Liu, Xhenhua Wu, and Zhiping Zhou, "Cluster-based three-dimensional localization algorithm for large scale wireless sensor networks," *Journal of Computers*, vol. 4, no. 7, pp. 585-592, doi: 10.4304/jcp.4.7.585-592, Jul 2009.

[78] P. Biswas and Y. Ye, "A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization," Stanford, CA 94305, Oct 2003.

[79] Pratik Biswas, and Yinyu Ye, "Semidefinite programming for ad hoc wireless sensor network localization," in *Proc. IEEE IPSN*, pp. 46-54, California, USA, April 26–27, 2004.

[80] S. Grime, H.F. Durrant-Whyte: "Data fusion in decentralized sensor networks," *Control Eng. Practice,* vol. 2, issue 5, pp. 849-863, Oct. 1994.

[81] Vesa Hasu and Heikki Koivo, "Decentralized Kalman filter in wireless sensor networks-case studies," *Advances in Computer, Information, and Systems Sciences, and Engineering*, pp. 61-68, doi: 10.1007/1-4020-5261-8_11, 2006.

[82] Leon Evers, Wouter Bach, Dennis Dam, Mischa Jonker, Hans Scholten, Paul Havinga, "An iterative quality-based localization algorithm for ad hoc networks," In *Proc. of Pervasive,* Key: citeulike:359293, 2002.

[83] Zhanyang Zhang, Olga Berger, Shane Sorbello, "A cluster based approach toward sensor localization and K-coverage problems," *Ubiquitous Computing and Communication Journal*, vol. 2, num. 4. pp. 15 - 23, 2007.

[84] C. R. Lin and M. Gerla, "Adaptive clustering for mobile wireless network," *IEEE Journal on Selected Areas in Communications*, vol. 15, issue 7, pp. 1265-1275,Sep 1997.

[85] D. Roetenberg, P. J. Slycke, and P. H. Veltink, "Ambulatory position and orientation tracking fusing magnetic and inertial sensing," *IEEE Trans. Biomed. Eng.,* vol. 54, no. 5, pp. 883-890, May 2007.

[86] A. Doucet, “On sequential Monte Carlo methods for Bayesian filtering,” Dept. Eng., Univ. Cambridge, UK, Tech. Rep., 1998.

[87] J. B. Kuipers, "Quaternions and rotation Sequences: a Primer with Applications to Orbits, Aerospace, and Virtual Reality", Princeton University Press ISBN 978-0-691-10298-6, 1999.

[88] R. van der Merwe, A. Doucet, J. F. G. de Freitas, and E. Wan, “The unscented particle filter,” Adv. Neural Inform. Process. Syst., Dec. 2000.

[89] N. Gordon, D. Salmond, and A. F. M. Smith, “Novel approach to nonlinear and non-Gaussian Bayesian state estimation,” Proc. Inst. Elect. Eng., F, vol. 140, pp. 107–113, 1993.

[90] S. Beauregard, Widyawan, and M. Klepal, “Indoor PDR performance enhancement using minimal map information and particle filters,” In Proc. of the IEEE/ION PLANS 2008, Monterey, USA, May 2008.

[91] P. Robertson, M. Angermann, and B. Krach, “Simultaneous Localization and Mapping for Pedestrians using only Foot-Mounted Inertial Sensors,” In Ubicomp, 2009.

[92] Y. Jin, H.-S. Toh, W.-S. Soh, and W.-C. Wong, “A robust dead-reckoning pedestrian tracking system with low cost sensors, ” in Proc. IEEE PERCOM, pp. 222-230, Mar. 2011.

[93] Yin Chen, Dimitrios Lymberopoulos, Jie Liu, and Bodhi Priyantha, "Indoor Localization Using FM Signals", IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 12, NO. 8, AUGUST 2013.

[94] Kaishun Wu, Jiang Xiao, Youwen Yi, Dihu Chen, Xiaonan Luo, and Lionel M. Ni, "CSI-Based Indoor Localization", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 24, NO. 7, JULY 2013

[95] ADIS16400/16405 data sheet, document available at http://www.analog.com/static/imported-files/data_sheets/ADIS16400_16405.pdf, link is tested in Dec 2014.

[96] X. Yun, E. R. Bachmann, and R. B. McGhee, “A simplified quaternion-based algorithm for orientation estimation from earth gravity and magnetic field measurements,” *IEEE Trans. Instrum. Meas.*, vol. 57, no. 3, pp. 638-650, Mar. 2008.

[97] D. Choukroun, I. Y. B. Itzhack, and Y. Oshman, “Novel quaternion Kalman filter, ” *IEEE Trans. Aerosp. Electron. Syst.*, vol.42, issue 1, pp.174-190, Jan. 2006.

[98] Cortext in Motion Analysis, description available at http://www.motionanalysis.com/html/industrial/cortex.html, link is tested in Dec 2014.

[99] G. Welch and G. Bishop, “An introduction to the Kalman filter,” *Technical Report TR 95-041, Computer Science*, UNC Chapel Hill, 1995.

[100] D. Niculescu and B. Nath, “Ad hoc positioning system (APS),” in *Proc. IEEE GLOBECOM*, San Antonio, TX, vol. 5, pp. 2926-2931, Nov 2001.

[101] A. Savvides, H. Park and M. B. Srivastava, “The bits and flops of The n-hop multilateration primitive for node localization problems,” in *Proc. of the 1st*

ACM International Workshop on Wireless Sensor Networks and Applications, Atlanta, USA, pp. 112-121, September, 2002.

[102] K. C. Toh, M. J. Todd, and R. H. Tutuncu, "SDPT3—a MATLAB software package for semidefinite programming," *Optimization Methods and Software*, vol. 11, pp. 545–581, 1999.

[103] "wireless-matlab-0.5.zip" available at http://wireless-matlab.sourceforge.net/, link is tested in Dec 2014.

[104] A. Kong, J. S. Liu and W. H. Wong, "Sequential imputations and Bayesian missing data problems", *Journal of the American Statistical Association*.vol. 89, pp. 278-288, 1994.

Appendix

A. Quaternion

In mathematics, the quaternions [87] are a number system that extends the complex numbers. A quaternion is defined by Hamilton as the quotient of two directed lines in a three-dimensional space. The basic equations are

$$\boldsymbol{i}\times\boldsymbol{i}=\boldsymbol{j}\times\boldsymbol{j}=\boldsymbol{k}\times\boldsymbol{k}=\boldsymbol{i}\times\boldsymbol{j}\times\boldsymbol{k}=-1 \qquad \textbf{(A-1)}$$

where ***i***, ***j***, and ***k*** denote the standard orthonormal basis for a three dimensional space $\mathbb{R}^3$. The orthonormal basis can be written as triplets of scalars

$$\boldsymbol{i}=[1;0;0] \quad \boldsymbol{j}=[0;1;0] \quad \boldsymbol{k}=[0;0;1] \qquad \textbf{(A-2)}$$

A quaternion is a 4-tuple that can be written as

$$\mathbf{q}=q_1\boldsymbol{i}+q_2\boldsymbol{j}+q_3\boldsymbol{k}+q_0=[q_1;q_2;q_3;q_0] \qquad \textbf{(A-3)}$$

A.1. Quaternion Properties

Hamilton product (represented by $\otimes$) of two quaternions is determined by the products of the basis elements and the distributive law.

$$\begin{aligned}\mathbf{q}_1\otimes\mathbf{q}_2 &= (b_1\boldsymbol{i}+c_1\boldsymbol{j}+d_1\boldsymbol{k}+a_1)\otimes(b_2\boldsymbol{i}+c_2\boldsymbol{j}+d_2\boldsymbol{k}+a_2)\\ &=(a_1b_2+b_1a_2+c_1d_2-d_1c_2)\boldsymbol{i}+(a_1c_2-b_1d_2+c_1a_2+d_1b_2)\boldsymbol{j}\\ &\quad+(a_1d_2+b_1c_2-c_1b_2+d_1a_2)\boldsymbol{k}+a_1a_2-b_1b_2-c_1c_2-d_1d_2\end{aligned} \qquad \textbf{(A-4)}$$

The complex conjugate of the quaternion is denoted as $\mathbf{q}^*$, which equals

$$\mathbf{q} = -\boldsymbol{i}q_1 - \boldsymbol{j}q_2 - \boldsymbol{k}q_3 + q_0 = [-q_1; -q_2; -q_3; q_0] \qquad \textbf{(A-5)}$$

and it follows that

$$(\mathbf{q}_1 \otimes \mathbf{q}_2)^* = \mathbf{q}_2^* \otimes \mathbf{q}_1^* \qquad \textbf{(A-6)}$$

where $\otimes$ represents the Hamilton product for quaternion multiplication [44].

The *norm* of a quaternion **q** is denoted by $N(\mathbf{q})$ where

$$N(\mathbf{q}) = \sqrt{\mathbf{q}^* \otimes \mathbf{q}} \qquad \textbf{(A-7)}$$

A unit quaternion has a *norm* equals to one that is

$$\| \mathbf{q} \| = 1 \quad \text{and} \quad N(\mathbf{q}) = \sqrt{\mathbf{q}^* \otimes \mathbf{q}} = 1 \qquad \textbf{(A-8)}$$

We have $\mathbf{q}^{-1} \otimes \mathbf{q} = \mathbf{q} \otimes \mathbf{q}^{-1} = 1$ by definition of inverse. By multiplying $\mathbf{q}^*$ at both sides we have

$$\mathbf{q}^* \otimes \mathbf{q} \otimes \mathbf{q}^{-1} = N^2(\mathbf{q})\mathbf{q}^{-1} = \mathbf{q}^* \qquad \textbf{(A-9)}$$

, from which we have

$$\mathbf{q}^{-1} = \frac{\mathbf{q}^*}{N^2(\mathbf{q})} = \frac{\mathbf{q}^*}{\| \mathbf{q} \|} \qquad \textbf{(A-10)}$$

If **q** is an unit quaternion, then

$$\mathbf{q}^{-1} = \mathbf{q}^* \qquad \textbf{(A-11)}$$

A.2. Quaternion Rotation

According to Euler's rotation theorem, any arbitrary rotation of a rigid body in 3-dimensional space is equivalent to a single rotation by a given angle about a fixed axis (Euler axis). Fig. 2.3 illustrates an example of rotation from point A to point A'. Point A has rotated about the rotation axis $\vec{\mu}$ by an angle w. This rotation would be represented using quaternion by

$$\mathbf{q} = \cos(\frac{w}{2}) + \vec{\mu}\sin(\frac{w}{2}) \qquad \textbf{(A-12)}$$

where $\vec{\mu}$ is an unit vector (the rotation axis) and w is the rotated angle.

We should take note that the coordinates system could also rotate, after which a rigid body would also have new coordinates. A rotation by the coordinates system is equivalent to its conjugate rotation by the rigid body. Suppose the coordinate system has been rotated by **q**, then the original vector $\vec{\mathbf{v}}$ in the transformed coordinate system would equal $\vec{\mathbf{v}}'$ that

$$\begin{aligned}\vec{\mathbf{v}}' = \mathbf{q}^{-1}\vec{\mathbf{v}}\mathbf{q} &= \left(\cos(\frac{w}{2}) - \vec{\mu}\sin(\frac{w}{2})\right)\vec{\mathbf{v}}\left(\cos(\frac{w}{2}) + \vec{\mu}\sin(\frac{w}{2})\right) \\ &= \vec{\mathbf{v}}_{\perp}\cos(w) - (\vec{\mu}\times\vec{\mathbf{v}}_{\perp})\sin(w) + \vec{\mathbf{v}}_{\parallel}\end{aligned} \qquad \textbf{(A-13)}$$

where $\vec{\mathbf{v}}_{\perp}$ and $\vec{\mathbf{v}}_{\parallel}$ represents the components of $\vec{\mathbf{v}}$ which are perpendicular and parallel to $\vec{\mu}$, respectively.

q can be represented by the vector:

$$\mathbf{q}^T = [\mathbf{e}^T, q_0] = [\sin(w/2)[e_x, e_y, e_z], \cos(w/2)] \qquad \textbf{(A-14)}$$

where $\mathbf{e}^{\mathbf{T}} = [e_x, e_y, e_z]$ represents the rotation axis, and w is the rotation angle.

Eq. A-13 can be represented using orthogonal matrix that $\vec{\mathbf{v}}' = \mathbf{A}(\mathbf{q})\vec{\mathbf{v}}$, and

$$\mathbf{A}(\mathbf{q}) = (q_0^2 - \mathbf{e}^T\mathbf{e})\mathbf{I}_3 + 2\mathbf{e}\mathbf{e}^T - 2q_0\mathbf{C}(\mathbf{e}) \qquad \textbf{(A-15)}$$

where $\mathbf{I}_3$ is a 3 by 3 identity matrix, $\mathbf{C}(\mathbf{e})$ is the matrix for cross-product computation, which is then used to multiply another vector:

$$\mathbf{C}(\mathbf{e}) = \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 0 \end{bmatrix} \qquad \textbf{(A-16)}$$