Multi-Sensor Based Indoor Vehicle and Pedestrian Navigation

DISSERTATION
zur Erlangung des Grades eines Doktors
der Ingenieurwissenschaften

vorgelegt von
M.Sc. Wennan Chai

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät
der Universität Siegen
Siegen 2019

Stand: December 2019

Betreuer und erster Gutachter

Prof. habil. Dr.-Ing. habil. Otmar Loffeld

Universität Siegen

Zweiter Gutachter

Prof. Dr.-Ing. Hubert Roth

Universität Siegen

Tag der mündlichen Prüfung

13.12.2019

Gedruckt auf alterungsbeständigem holz- und säurefreiem Papier

Acknowledgements

First of all, my thank goes to my mentor Prof. Dr.-Ing. habil. Otmar Loffeld for supervising me. He taught me how to do scientific research works professionally. He encouraged me to pursue my ideas and gave me his patient guidance. Without his help and support, this work would not be finished.

Besides, I would like to thank my second supervisor Prof. Dr.-ing. Hubert Roth. He was supervising my master programme. In my Ph.D. phase, he gave me the chance to attend the Deutscher Akademischer Austauschdienst Dienst (DAAD) joint research project and provided me his helpful guidance.

I also want to extend my gratitude to Dr.-Ing. Holger Nies and Priv.-Doz. Dr.-Ing. habil. Stefan Knedlik. Dr. Nies was our programme manager. He always gave me the best help without hesitation. He provided me not only administrative supports but also many pieces of scientific advice. Dr. Knedlik was my tutor at the beginning of my Ph.D. phase. He guided me to finish my master thesis and start the doctoral research work.

Then I want to thank my college and friend, M.Sc. Cheng Chen. We have been tightly cooperating with each other in indoor navigation and mapping. My research benefits a lot from the cooperation works and frequent discussions with him.

Many thanks go to members of our navigation group and all the colleagues in the Center for Sensor Systems (ZESS) for providing such a wonderful environment for work and study. I also thank the secretaries of ZESS: Silvia Niet-Wunram, Renate Szabo, and Katharina Haut, they were always friendly, patient and helpful.

Special thanks to the master students I supervised, namely M.Sc. Tianyu Zhou, M.Sc. Song Zhang, M.Sc Siqi Huang, and M.Sc. Peng Diao. They did great student project works and finished their master theses in a professional way. They gave my work a huge support with scientific discussions and experimental results.

This work is funded by Multi-Modal Sensor Systems for Environmental Exploration and Safety (MOSES) programme, ZESS, University of Siegen. Thanks for the financial support.

Abstract

Among the positioning techniques in indoor environments, the approach on the basis of exploiting Wi-Fi is attractive, which is expected to yield a cost-effective and easily accessible solution. Most Wi-Fi localization methodologies rely on the received signal strength (RSS) measurements. In this work, different Wi-Fi RSS based positioning algorithms are explored. The performance of each approach is shown with experimental results.

Considering the complementary nature of Wi-Fi positioning and inertial navigation system (INS), the combination of both systems yields a synergetic effect resulting in higher performance. For indoor vehicle navigation, the performance of the INS/Wi-Fi integrated system can be further improved without hardware change. An enhanced integration, which employs adaptive Kalman filtering (AKF) and vehicle constraints, is presented. The experimental results show that the enhanced integrated system provides higher navigation accuracy, compared to using Wi-Fi positioning and conventional INS/Wi-Fi integration.

For personal navigation applications, the pedestrian dead reckoning (PDR) system is employed. With a foot mounted IMU, zero velocity update (ZUPT) and zero angular rate update (ZARU) methodologies can be applied to re-calibrate the IMU, which can reduce the INS drift errors. For personal navigation with the IMU embedded in the portable device, the adapted PDR based on device placement mode classification is presented. Three typical placement modes are discussed. The classification performances with different classifiers are shown with real test results. The adapted PDR is further combined with Wi-Fi positioning. The experimental results show that the integrated system outperforms the standalone navigation systems.

Attitude estimation is a challenging topic for indoor navigation. The camera based visual gyroscope technique can transform information found from images into the camera rotation. Unlike the rate gyroscope in an IMU, the visual-gyro using vanishing points does not suffer from drift errors. In this work, an INS/visual-gyro integration using direction cosine matrix (DCM) based models is presented. Compared to the conventional Euler angle models, the usage of DCM can provide linear system models and avoid singularity problems. The performance of attitude and gyro bias estimation using the integrated system is shown with turntable test and experimental results.

Kurzfassung

Bei den Positionierungstechniken in Innenräumen ist der auf der Nutzung von Wi-Fi basierende Ansatz attraktiv. Durch den Ansatz wird erwartet, eine kostengünstige und leicht zugängliche Lösung einzubringen. Die meisten Wi-Fi-Lokalisierungsmethoden sind auf Messungen der empfangenen Signalstärke (RSS) angewiesen. In dieser Arbeit werden verschiedene, auf Wi-Fi RSS basierende Algorithmen für die Positionierung erforscht. Die Leistung von jedem Ansatz wird mit experimentellen Ergebnissen gezeigt.

In Anbetracht des komplementären Charakters der Wi-Fi-Positionierung und des Trägheitsnavigationssystems (INS) ergibt die Kombination der beiden Systeme einen synergetischen, zu höheren Leistungen führenden Effekt. Für die Fahrzeugsnavigation im Innenraum lässt sich die Leistung des INS/Wi-Fi integrierten Systems ohne Änderung der Hardware verbessern. Eine verbesserte Integration, die die adaptive Kalman-Filterung (AKF) in verbindung mit Fahrzeugeinschränkungen verwendet, wird vorgestellt. Die experimentellen Ergebnisse zeigen, dass das verbesserte integrierte System eine höhere Navigationsgenauigkeit bietet, im Vergleich zur Verwendung der Wi-Fi-Positionierung und der konventionellen INS/Wi-Fi-Integration.

Für persönliche Navigationsanwendungen wird die Fußgänger-Koppelnavigation (PDR) eingesetzt. Mit einer am Fuß montierten IMU sind das Null-Geschwindigkeit-Update (ZUPT) und das Nullwinkelrate-Update (ZARU) Methoden anzuwenden, um die IMU neu zu kalibrieren, dadurch können die Driftfehler vom INS reduziert werden. Für persönliche Navigation mit der in dem tragbaren Gerät eingebetteten IMU wird die angepasste PDR auf Basis der Klassifizierung vom Geräteplatzierungsmodus vorgestellt. Drei typische Platzierungsmodi werden diskutiert. Die Leistungen der Klassifizierung mit verschiedenen Klassifikatoren werden mit echten Testergebnissen gezeigt. Die angepasste PDR wird weiter mit der Wi-Fi-Positionierung kombiniert. Die Versuchsergebnisse zeigen, dass das integrierte System die allein operierenden Navigationssysteme leistungsmäßig übertrifft.

Die Schätzung der Orientierung ist eine Herausforderung für die Indoor-Navigation. Die visuelle, auf Kamera basierende Technik des Gyroskops kann die aus Bilder generierte Information in die Rotation der Kamera verwandeln. Anders als das Gyroskop in einer IMU leidet der visuelle Kreisel mit Fluchtpunkten nicht unter Driftfehlern. In dieser Arbeit wird eine Integration des INS und des visuellen Kreisels, die die auf Richtungskosinusmatrix

(DCM) basierenden Modelle anwendet, vorgestellt. Im Vergleich zu den herkömmlichen Modellen des Eulerwinkels kann die Verwendung von DCM lineare systemmodelle bereitstellen und die Probleme der Singularität vermeiden. Die Leistung der Einstellung und der Schätzung der Kreiselabweichung unter Verwendung des integrierten Systems werden mit experimentellen Testergebnissen gezeigt.

Contents

ACKNOWLEDGEMENTS III

ABSTRACT IV

KURZFASSUNG V

CONTENTS VII

LIST OF FIGURES X

LIST OF TABLES XII

LIST OF ABBREVIATIONS XIII

LIST OF SYMBOLS XVI

CHAPTER 1 INTRODUCTION 1

1.1 SUBJECT OF RESEARCH 1

1.1.1 Indoor localization with Wi-Fi based approaches 1

1.1.2 Indoor vehicle navigation using integration of INS and Wi-Fi positioning 1

1.1.3 Adapted pedestrian navigation using PDR/Wi-Fi integration 1

1.1.4 Indoor attitude estimation using INS/Visual-Gyroscope integration 2

1.2 STRUCTURE OF THE DISSERTATION 2

CHAPTER 2 WI-FI BASED LOCALIZATION TECHNIQUES 5

2.1 BACKGROUND AND CONCEPT OF WI-FI BASED LOCALIZATION 5

2.2 WI-FI LOCALIZATION USING RADIO PROPAGATION MODEL 7

2.2.1 Radio propagation models 7

2.2.2 Localization using propagation model with WAF 10

2.3 WI-FI LOCALIZATION USING RSS FINGERPRINTING 10

2.3.1 Methods for database building 10

2.3.2 Methods for location determination 11

2.4 FIELD EXPERIMENT AND NUMERICAL RESULTS 16

2.4.1 Hardware and test-bed 16

2.4.2 Results and analysis 17

2.5 SUMMARY 19

CHAPTER 3 INDOOR VEHICLE NAVIGATION USING ENHANCED INS/WI-FI INTEGRATION 20

3.1 SYSTEM MODELLING FOR INS/WI-FI INTEGRATION 20

3.1.1 INS process model using Euler angles *20*

3.1.2 Observation models with Wi-Fi positioning *22*

3.1.3 Unscented Kalman filtering........ *23*

3.2 ENHANCEMENTS USING ADAPTIVE KALMAN FILTERING AND VEHICLE CONSTRAINTS 25

3.2.1 Adaptive Kalman filtering (AKF)........ *25*

3.2.2 Vehicle constraints........ *26*

3.2.3 Enhanced INS/Wi-Fi integration *27*

3.3 FIELD EXPERIMENT AND RESULTS 28

3.3.1 Hardware and test-bed........ *28*

3.3.2 Results and comparison *30*

3.4 SUMMARY 34

CHAPTER 4 ADAPTED INDOOR PEDESTRIAN NAVIGATION USING PDR/WI-FI INTEGRATION........ 35

4.1 IMU BASED FOOT-MOUNTED PEDESTRIAN DEAD RECKONING 35

4.1.1 Gait cycle and step detection........ *36*

4.1.2 Stride length estimation........ *37*

4.1.3 Heading determination *39*

4.1.4 Zero velocity update and zero angular rate update........ *39*

4.2 ADAPTED PEDESTRIAN DEAD RECKONING WITH PORTABLE DEVICES 44

4.2.1 Device placement mode definition and features for mode classification *44*

4.2.2 Classification algorithms........ *47*

4.2.3 Placement mode classification results *50*

4.2.4 Adapted PDR based on classified placement mode *52*

4.2.5 Integration of adapted PDR and Wi-Fi based positioning........ *53*

4.3 FIELD EXPERIMENT AND RESULTS 54

4.4 SUMMARY 56

CHAPTER 5 INDOOR ATTITUDE ESTIMATION USING INS/VISUAL-GYROSCOPE INTEGRATION 57

5.1 PROJECTIVE GEOMETRY AND VANISHING POINT DETECTION 58

5.1.1 Projective geometry........ *59*

5.1.2 Vanishing point detection *61*

5.2 DCM BASED INS/VISUAL-GYRO INTEGRATION 72

5.2.1 Attitude estimation with detected vanishing points........ *72*

5.2.2 System modelling........ *73*

5.3 FIELD EXPERIMENTS 77

5.3.1 Turntable test........ *78*

5.3.2 Pedestrian experiment ... *80*

5.4 SUMMARY ... 83

CHAPTER 6 SUMMARY AND CONCLUSIONS ... 84

6.1 SUMMARY ... 84

6.2 CONCLUSIONS ... 85

APPENDIX A ARTIFICIAL NEURAL NETWORKS ... 87

APPENDIX B SUPPORT VECTOR MACHINE ... 91

APPENDIX C UNSCENTED KALMAN FILTERING ... 98

BIBLIOGRAPHY ... 102

List of Figures

Figure 2.1: Bandwidth and range of Wi-Fi 6

Figure 2.2: Example of RSS measurement space for different locations (three APs) 13

Figure 2.3: Experiment test-bed 16

Figure 2.4: Experimental hardware: (a) Linksys router; (b) D-Link router; (b) TP-Link Wi-Fi adapter 17

Figure 2.5: Positioning error cumulative distribution functions (CDFs) 17

Figure 2.6: Positioning error CDFs 18

Figure 3.1: Diagram of INS/Wi-Fi integration with tightly/loosely coupled structure 23

Figure 3.2: Diagram of enhanced INS/Wi-Fi integration 28

Figure 3.3: Experimental hardware 28

Figure 3.4: Experimental scenario 29

Figure 3.5: RSS measurements during the robot movement 30

Figure 3.6: Positioning errors (1st group) using: (a) Wi-Fi empirical fingerprinting positioning; (b) integration of INS and empirical fingerprinting; (c) enhanced INS/Wi-Fi integration 31

Figure 3.7: Positioning errors (2nd group) using: (a) propagation model based fingerprinting positioning; (b) integration of INS and model based fingerprinting; (c) enhanced INS/Wi-Fi integration 32

Figure 3.8: Positioning error CDFs (1st group) 33

Figure 3.9: Positioning error CDFs (2nd group) 33

Figure 4.1: Pedestrian dead reckoning course 36

Figure 4.2: Gait cycle with foot-mounted IMU 36

Figure 4.3: Acceleration measurement along x-axis of foot-mounted IMU 37

Figure 4.4: Linear regression of stride frequency vs. stride length 38

Figure 4.5: Average error using estimated stride length (31 strides) 38

Figure 4.6: Estimated IMU bias errors 43

Figure 4.7: Trajectory estimation results 44

Figure 4.8: Placement cases: (a) body mounted mode (pocket mode); (b) dangling mode; (c) reading mode 45

Figure 4.9: IMU measurements (training and test data from the same user) in different placement modes 45

Figure 4.10: IMU measurements (training and test data from different users) in different placement modes 46
Figure 4.11: Comparison between smoothed data and original data 47
Figure 4.12: Classification accuracy (same user case) 50
Figure 4.13: Classification accuracy (different user case) 51
Figure 4.14: Integration of adapted PDR and Wi-Fi position 53
Figure 4.15: Pedestrian trajectory in different placement modes 54
Figure 4.16: Positioning errors comparison 55
Figure 4.17: Error CDFs 55
Figure 5.1: Vanishing points in 2D images 58
Figure 5.2: Projection of a point in 3D-space onto the camera image plane 59
Figure 5.3: Projection of a line in 3D-space onto the camera image plane 60
Figure 5.4: The results of Canny edge detection 64
Figure 5.5: Illustration of the basic idea of Hough transform for lines 66
Figure 5.6: The results of line detection in the indoor environment 67
Figure 5.7: The result of the best fit line based on point coordinates [84] 69
Figure 5.8: Localization of central vanishing point 71
Figure 5.9: Experiment hardware 77
Figure 5.10: Experiment with a turntable 78
Figure 5.11: Attitude estimation errors 79
Figure 5.12: Gyro bias estimation error using INS/visual-gyro integration 79
Figure 5.13: Pedestrian experiment 80
Figure 5.14: Combination with pedestrian dead reckoning system 80
Figure 5.15: Attitude estimation result (roll) 81
Figure 5.16: Attitude estimation result (pitch) 81
Figure 5.17: Attitude estimation result (yaw) 82
Figure 5.18: Trajectory estimation result 82
Figure A.1: Schematic drawing of k^{th} neuron [58] 87
Figure A.2: Theory and architecture of PNN [59] 89
Figure B.1: Linear classifier: optimal hyperplane in linearly separable case 92
Figure B.2: Case that the samples are not linearly separable 95

List of Tables

Table 2.1: Error means and standard deviations 18

Table 2.2: Error means and standard deviations 18

Table 3.1: Specification and parameters of Xsens MTi IMU [40] 29

Table 3.2: Position error comparison (1st group) 33

Table 3.3: Position error comparison (2nd group) 34

Table 4.1: Confusion matrix of SVM (same user case) 51

Table 4.2: Confusion matrix of SVM (different user case) 51

Table 4.3: Sensor data and detection method for different placement modes 52

Table 4.4: Means and standard deviations of positioning errors 55

Table 5.1: Kinect parameters 78

Table 5.2: Error parameters 79

List of Abbreviations

Acronym	Definition
AKF	Adaptive Kalman filtering
AMISE	Asymptotic Mean Integrated Square Error
ANN	Artificial Neural Networks
AOA	Angle of Arrival
AP	Access Points
AVC	Angular Velocity Constraint
BSD	Berkeley Software Distribution
BSS	Basic Service Set
BVC	Body Velocity Constraint
CDF	Cumulative Distribution Function
CI	Confidence Interval
DAG	Directed Acyclic Graph
ERM	Empirical Risk Minimization
FPS	Frames Per Second
GA	Genetic Algorithm
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HC	Height Constraint
HT	Hough Transform
IBSS	Independent Basic Service Set
IMU	Inertial Measurement Unit
INS	Inertial Navigation System

LOS	Line of Sight
KDE	Kernel Density Estimate
KNN	K-Nearest Neighbor
MEMS	Microelectromechanical Systems
NED	North East Down
NLOS	Non Line of Sight
PCM	Pulse Code Modulation
PDF	Probability Density Function
PDR	Pedestrian Dead Reckoning
PNN	Probabilistic Neural Networks
PSO	Particle Swarm Optimization
PVA	Position, Velocity, and Attitude
RFID	Radio Frequency Identification
RGB	Red Green Blue
RSS	Received Signal Strength
RANSAC	Random Sample Consensus
SHT	Standard Hough Transform
SNR	Signal to Noise Ratio
SRM	Structural Risk Minimization
SVC	Support Vector Classification
SVM	Support Vector Machine
TOA	Time of Arrival
T-R	Transmitter- Receiver
UGA	Ultra Graphics Array
UKF	Unscented Kalman Filtering

UWB	Ultra Wideband
VC	Vapnik Chervonenkis
WAF	Wall Attenuation Factor
WECA	Wireless Ethernet Compatibility Alliance
WLAN	Wireless Local Area Network
WPS	Wi-Fi Positioning System
ZARU	Zero Angular Rate Update
ZUPT	Zero Velocity Update

List of Symbols

Conventions regarding the notation writing

a) Scalars are denoted in italic letters.

b) Vector and matrices are denoted in bold letters.

Symbol	Definition
$\cdot\cdot$	Estimated value of
$\cdot\cdot$	Measured or calculated value of
$\overline{\cdot\cdot}$	Mean value of
Δ	Increment (difference) value of
$\times$	Cross product of
$E[\cdot]$	Expectation of
$\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle$	Inner product of vector $\boldsymbol{\alpha}$ and vector $\boldsymbol{\beta}$
N_{W}	Number of obstructions (walls) between the transmitter and the receiver
C_{W}	Maximum number of obstructions (walls) between the transmitter and the receiver
$S_{\mathrm{OB},l}$	RSS from AP l at the position of the object
$S_{\mathrm{ref},l}$	RSS from AP l at the reference point
S_{DB}	RSS recorded in the fingerprinting database
ρ_l	Distance between the object and the AP l
$\rho_{\mathrm{ref},l}$	Distance between the reference points and the AP l
d_{SS}	Euclidean distance in signal space

$W(\cdot)$	Weight function
$\mathbf{F}_{\mathrm{DB},i}$	RSS matrix of the fingerprinting database
$\mathbf{r}$	Position of the object in navigation frame
$\mathbf{r}_{\mathrm{AP}(l)}$	Position of the AP l
$\mathbf{r}_{\mathbf{DB},i}$	Position of the ith survey point in the database
$d(\boldsymbol{\alpha},\boldsymbol{\beta})$	Distance between vector $\boldsymbol{\alpha}$ and vector $\boldsymbol{\beta}$
$\mathbf{v}$	Velocity vector in navigation frame
$\boldsymbol{\psi}$	Attitude vector in navigation frame
$\tilde{\mathbf{f}}_{\mathrm{b}}$	Acceleration measurement vector from accelerometer
$\tilde{\boldsymbol{\omega}}_{\mathrm{b}}$	Angular rate measurement vector from gyroscope
$\mathbf{v}_{\mathrm{b}}$	Velocity vector in body frame
$\mathbf{f}_{\mathrm{b}}^{\mathrm{bias}}$	Accelerometer bias vector
$\boldsymbol{\omega}_{\mathrm{b}}^{\mathrm{bias}}$	Gyro bias vector
$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$	Frame rotation matrix from the body frame to navigation frame
$\Phi_{\mathrm{b}}^{\mathrm{n}}$	Rotation rate matrix between body frame and navigation frame
θ	Roll angle
ϕ	Pitch angle
φ	Yaw angle
ω_x	Angular rate (roll angle)
ω_y	Angular rate (pitch angle)
ω_z	Angular rate (yaw angle)
$\mathbf{x}$	System state vector

$\mathbf{y}$	Measurement vector
$\mathbf{f}(\cdot)$	State propagation equation
$\mathbf{h}(\cdot)$	Measurement equation
$K(\cdot)$	Kernel function
$f(\cdot)$	Density distribution function
$\mathbf{w}$	System process noise
$\boldsymbol{\eta}$	System measurement noise
$\mathbf{Q}$	Process error covariance
$\mathbf{R}$	Measurement error covariance
$\mathbf{P}$	State covariance
L_{strd}	Stride length
f_{strd}	Stride frequency
$R(\cdot)$	True risk
$R_{\mathrm{emp}}(\cdot)$	Empirical risk
$\varPhi(\cdot)$	Confidence interval
γ	Minimal functional margin
γ_{g}	Minimal geometric margin
$g(\cdot)$	Discriminant function in binary classifier
ξ	Slack variable
f_{strd}	Stride frequency
$\mathbf{C}_{\mathrm{cam}}$	Camera transformation matrix
$\mathbf{r}_{\mathrm{2D}}$	Position vector in 2D image frame

$\boldsymbol{\pi}_\infty$	Plane at infinity
$\mathbf{G}_{\mathrm{V}}$	Gradient in the vertical direction
$\mathbf{G}_{\mathrm{H}}$	Gradient in the horizontal direction
$\mathbf{V}_{\mathrm{vp}}$	Vanishing point location matrix
$\mathbf{K}_{\mathrm{cam}}$	Camera calibration matrix
$f_{\mathrm{cam},\,x}$	Focal length on x-axis
$f_{\mathrm{cam},\,y}$	Focal length on y-axis
$(u_{\mathrm{cam}}, v_{\mathrm{cam}})$	Principal point

Chapter 1
Introduction

1.1 Subject of research

1.1.1 Indoor localization with Wi-Fi based approaches

Indoor navigation is a challenging topic for low-cost vehicle and pedestrian applications nowadays. Many positioning techniques have been developed. Among these techniques, the approach on the basis of exploiting 802.11 WLAN (Wi-Fi) is attractive, which is expected to yield a cost-effective and easily accessible solution. Regarding this topic, some researchers focus on the usage of time of arrival (TOA) and angle of arrival (AOA). However, in the context of indoor WLANs, the approaches using the information of received signal strength (RSS) are mostly employed. In this work, we explore the Wi-Fi RSS positioning techniques, which include the methods based on radio propagation models and the methods using RSS fingerprinting.

1.1.2 Indoor vehicle navigation using integration of INS and Wi-Fi positioning

Microelectromechanical systems (MEMS) based inertial measurement units (IMUs) are commonly used in low cost dead reckoning navigation applications. The inertial navigation system (INS) provides the motion information of the object with a high update rate and it can achieve a high precision in short time duration. However, the INS suffers from local anomalies and error drifts over time. In contrast, the Wi-Fi positioning approaches provide a relatively low accuracy and update rate but its localization error does not propagate with time. Considering the complementary nature of INS and Wi-Fi positioning, the combination of both systems is expected to yield a synergetic effect resulting in higher navigation performance. In this work, the integration of INS and Wi-Fi positioning for indoor vehicle navigation is explored. To further improve the integrated system, the enhancements making use of vehicle constraints and adaptive Kalman filtering algorithm are presented.

1.1.3 Adapted pedestrian navigation using PDR/Wi-Fi integration

IMU based pedestrian dead reckoning (PDR) is widely used for personal navigation. It consists of three parts: step detection, stride length estimation, and user’s heading determination. Like other inertial systems, it suffers from a propagating drift error because of IMU biases. In this work, the PDR algorithm with a foot-mounted IMU is studied. The zero speed of the IMU can be detected during the stance phase of the gait cycle. In this case, zero velocity update (ZUPT) and zero angular rate update (ZARU) can be used to re-estimate the IMU biases and hence reduce the drift error of the dead reckoning system. To design an adaptive PDR algorithm for portal devices which can be arbitrarily placed on the user's body, the step detection method needs to be changed according to different sensor placement modes. Three typical placement modes are considered and classified based on measurement outputs of accelerometers and gyroscopes. Then the adapted PDR is further combined with Wi-Fi based positioning to improve the navigation performance from the standalone systems.

1.1.4 Indoor attitude estimation using INS/Visual-Gyroscope integration

Attitude estimation is a challenging topic for indoor navigation. The camera based visual gyroscope technique can transform information found from images into the camera rotation. Many researchers focus on systems with a priori formed database containing images of recognizable features in the surroundings attached with position and attitude information. However, the database based procedure is restricted to predefined and hence known areas. In contrast, the methods directly calculating the motion of the camera from consecutive images yield a universal solution for real applications. As one of these methods, the vanishing point based visual gyroscope (visual-gyro) technique is employed in this work. Unlike the rate gyroscope in an IMU, the visual-gyro does not suffer from drift errors. But the visual-gyro's availability highly depends on the indoor environment and its performance for fast rotation is limited by the low update rate. In order to overcome the drawbacks of the standalone systems, an INS/visual-gyro integration using direction cosine matrix (DCM) models is presented. Compared to the conventional Euler angle models, the usage of DCM can provide linear system models and avoid singularity problems.

1.2 Structure of the dissertation

In Chapter 2, Wi-Fi positioning techniques are explored. The background and concepts of WLAN localization are overviewed. Wi-Fi signal propagation models and localization methods using the propagation model are introduced. Wi-Fi RSS fingerprinting methods using different database building approaches and location determination algorithms are presented. One field experiment is carried out and the performances of the introduced Wi-Fi positioning approaches are compared and analyzed with experimental results.

In Chapter 3, indoor vehicle navigation using integration of INS and Wi-Fi positioning is described. The system modelling is made for the integration. The system process model is derived based on strap-down INS mechanization equations and the system observation model is provided by Wi-Fi based positioning. The integration structure can be either tightly-coupled or loosely-coupled depending on the employed Wi-Fi positioning method. Because of the nonlinearities of the system models, the unscented Kalman filter is employed for the integration. An enhanced INS/Wi-Fi integration aided with vehicle constraints and adaptive Kalman filtering algorithm is presented. One field experiment is performed, and the results show the advantages of the INS/Wi-Fi integrated system and the enhanced integration with respect to the standalone Wi-Fi positioning approaches.

In Chapter 4, indoor pedestrian navigation using integration of PDR and Wi-Fi is explored. PDR with a foot-mounted IMU is described. Three parts of PDR, namely step detection, stride length estimation and heading determination, are introduced. To reduce the PDR drift error caused by IMU biases, ZUPT and ZARU algorithms are employed to re-estimate the IMU biases and the improvements are shown with experimental results. The adapted PDR designed for portable devices is presented. Device placement mode definition and features for mode classification are introduced. The corresponding classification results are provided with real test data. The adapted PDR is further combined with Wi-Fi positioning. A field experiment is carried out to show the performance of PDR/Wi-Fi integration compared to the standalone navigation systems.

In Chapter 5, indoor attitude estimation using INS/visual-gyro integration is presented. To illustrate the theoretical background of vanishing point based visual-gyro, the projective geometry is introduced. Three steps of vanishing point detection from an image, namely edge detection, line detection and vanishing point localization, are described. Attitude estimation with detected vanishing points is provided. To overcome the limitations of INS and visual-gyro, the integration of both systems is presented. DCM based system modelling is made and Kalman filtering algorithm is utilized for sensor fusion. One turn-table test and one pedestrian experiment are carried out. To show the performance of the integrated system, the numerical results are given and analyzed.

Last but not the least, in Appendix A and B, two pattern recognition algorithms, artificial neural network (ANN) and support vector machine (SVM), are described respectively.

They are utilized in this work for device placement mode classification. The detailed introduction and derivation of the classifiers are presented. In Appendix C, the unscented Kalman filter (UKF), which is employed for the nonlinear system estimation, is introduced.

Chapter 2
Wi-Fi Based Localization Techniques

For outdoor positioning and navigation, solutions based on global navigation satellite system (GNSS) are satisfactory in most applications. But such technology is not utilizable for most indoor applications. Therefore, other positioning techniques have been developed for indoor environments lately, e.g., the methods based on wireless local area network (WLAN), Bluetooth, radio frequency identification (RFID), ultra-wideband (UWB), infrared and ultrasound, etc. Among these techniques, the approach on the basis of exploiting 802.11 WLAN (Wi-Fi) is attractive, which is expected to yield a cost-effective and easily accessible solution. Most localization methodologies based on Wi-Fi rely on signal to noise ratio (SNR) or received signal strength (RSS). Most of them, comprising the widely referred RADAR method, employ fingerprinting methods. In [1], two approaches have been proposed to build the fingerprinting database, namely, the empirical method and the wall attenuation factor (WAF) model based method. The first method always yields relatively better performance but requires significant implementation effort [2]. On the other hand, the positioning can also be achieved by directly employing field propagation models with least squares estimation method, although the estimation accuracy shows large deviations [3].

In this chapter, the background and concept of Wi-Fi localization are briefly described in Section 2.1. In Section 2.2, some widely used Wi-Fi signal propagation models are introduced and localization method using the WAF propagation model is presented. In Section 2.3, Wi-Fi RSS fingerprinting localization methods using different database building approaches and location determination algorithms are explored. Last but not least, in Section 2.4, one indoor experiment is carried out and the performances of the introduced Wi-Fi positioning approaches are compared with the numerical results.

2.1 Background and concept of Wi-Fi based localization

A WLAN represents a reliable solution to connect two or more wireless devices by using some wireless distribution methods and it can also provide a connection through access points to a wider internet. WLAN gives users the mobility to move around within a local coverage area and still be connected to the network. The most modern WLANs are based on the IEEE 802.11 standards (Wi-Fi).

The term Wi-Fi suggests wireless fidelity. The Wi-Fi Alliance was established in 1990 by the wireless Ethernet compatibility alliance (WECA) [4]. It consists of hundreds of companies which have the task to unite the standards of the products from different manufacturers based on IEEE-802.11 standards [5]. Therefore, Wi-Fi is a wireless standard for connecting electronic devices. And it can enable devices to connect to the Internet when the devices are within the range of a wireless network [6].

Basically, there are two parts of the wireless station: access points and clients. Access points (APs), namely routers, are basic devices that allow wireless devices to connect to a wired network using Wi-Fi. Wireless clients are the wireless network’s interface. There are many kinds of clients like mobile devices such as laptops, IP phones, personal digital assistants and other smartphones, or fixed devices such as desktops and workstations which are equipped with a wireless network interface. A single access point has a range of about 120 meters in indoor environments. For outdoor applications, it has an even wider range and therefore multiple overlapping APs can cover large areas. The following figure shows the relationship between the bandwidth and the range of a wireless net [7].

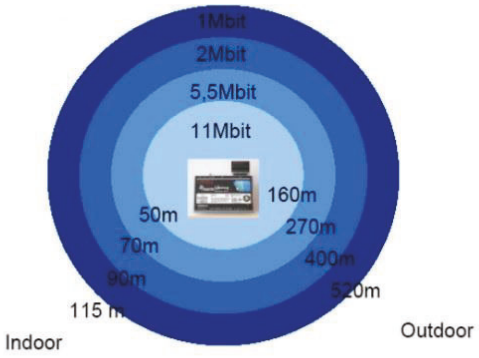


Figure 2.1: Bandwidth and range of Wi-Fi

The basic service set (BSS) is a set of all the stations that can communicate with each other. Normally there are two types of BSS: independent BSS and infrastructure BSS. Every BSS has an identification (ID), the so-called the BSSID. It is also the MAC address of the access point. An independent BSS (IBSS) is an ad-hoc network and it contains no access points, which means that it cannot connect to any other basic service sets. An infrastructure BSS can communicate with other stations that are not in the same basic service sets by communicating through access points.

WLAN positioning, also called Wi-Fi based localization in this work, refers to the use of radio wave signal to determine the location of a mobile device in a reference coordinate system. Regarding this topic, some researchers have explored the usage of time of arrival (TOA) [8] and angle of arrival (AOA) [9]. However, in the context of indoor WLANs, received signal strength (RSS) based approaches are generally employed for WLAN positioning. This is due to the fact that RSS measurements can be obtained relatively effortlessly and inexpensively without the need for additional hardware [10][11]. Moreover, RSS based positioning is noninvasive, as all sensing tasks can be carried out on the mobile client, eliminating the necessity for central processing.

2.2 Wi-Fi localization using radio propagation model

There are many existing radio propagation models. Most of them tend to focus on a particular characteristic such as signal fading or inter-floor loss. The models, which yield a relationship between the signal decaying power and the propagation distance, can be employed for position estimation in a Wi-Fi environment.

2.2.1 Radio propagation models

Rayleigh fading model

Rayleigh fading is a statistical model for the effect of a propagation environment on a radio signal, such as that used by wireless devices. According to a Rayleigh distribution, Rayleigh fading models assume that the magnitude of a signal that has passed through such a transmission medium will vary randomly, or fade. It describes small-scale rapid amplitude fluctuation in the absence of a strong received component [12].

The Rayleigh distribution is widely used to describe multipath fading because of its elegant theoretical explanation and the occasional empirical justification [13]. The Rayleigh model is mostly employed in heavily built-up urban environments. However, there is an assumption made in deriving this distribution is that all signals reaching the receiver have equal strength. In general, this is unrealistic for the applications inside a building. [1]

Rician fading model

Rician fading is a stochastic model for radio propagation when there is no line of sight (NLOS) signal. The signal arrives at the receiver by several different paths (multipath), and at least one of the paths is changing. Rician fading occurs when one of the paths is much stronger than the others; that is to say, there is the strongest path which has much less attenuation than other paths [14]. By Rician fading, the amplitude gain is characterized by a Rician distribution. The Rayleigh distribution is a special case of the Rician distribution when the strong path is eliminated, the amplitude distribution becomes Rayleigh.

Free space path loss

The free space model provides a measurement of path loss as a function of transmitter-receiver (T-R) separation when the transmitter and receiver are within LOS range in a free space environment. This model is not directly applicable to indoor signal propagation, it is a theoretical model, and it is the foundation for all other models. It can be used to compute the path loss at a close-in reference distance as required by the models which are discussed in the following section. The model is given by Equation (2.1) [12]:

$$P_{\mathrm{L}}(d) = -10\log\left[\frac{G_{\mathrm{t}}G_{\mathrm{r}}\lambda^2}{(4\pi)^2 d^2}\right] \tag{2.1}$$

where G_{t} and G_{r} are the radio gains of the transmitting and receiving antennas respectively, λ is the wavelength in meters, and d is the T-R separation in meters. $P_{\mathrm{L}}(d)$ is the power loss from the transmitter.

Log-distance path loss

The log-distance path loss model assumes that path loss varies exponentially with distance and the path loss in dB is given as the following equation:

$$P_{\mathrm{L}}(d) = P_{\mathrm{L}}(d_0) - 10 \cdot a \cdot \log(\frac{d}{d_0}) \tag{2.2}$$

where a is the path loss factor; d is the T-R separation in meters; d_0 is the close-in reference distance in meters. The path loss factor a depends upon the environment. In free space, a is equal to 2. In practice, the value of a depends upon the empirical data. The log-distance path loss model is mainly an outdoor model and neglects the effect due to the surrounding objects [11].

Log-Normal Shadowing

The log-distance path loss model has a disadvantage that it ignores the shadowing effects that can be caused by varying degrees of clutter between the transmitter and receiver. But the log-normal shadowing can make a compensation for the disadvantage.

The log-normal shadowing model predicts path loss as a function of T-R separation by the equation:

$$P_{\mathrm{L}}(d) = P_{\mathrm{L}}(d_0) - 10 \cdot a \cdot \log(\frac{d}{d_0}) + w_{\sigma} \tag{2.3}$$

where w_{σ} is a zero-mean Gaussian random variable. In comparison with Equation (2.3), the random variable w_{σ} compensates for random shadowing effects. And the values of n and σ depend upon the empirical data.

In order to achieve a more accurate and realistic indoor propagation model, the surrounding environment must be considered. The path between receiver and transmitter is usually blocked by walls, ceilings and other obstacles [4]. In this case, the measured signal strength is less than that predicted by the log-distance path loss model. Shadowing and wall attenuation factor (WAF) are the main factors affecting the propagation in an indoor environment.

A receiver in a Wi-Fi network is said to be in the shadow region when there is an obstacle blocking its line-of-sight to the access point [6]. Different materials produce varying amounts of attenuation in the shadow region and the amount of attenuation is also frequency dependent. WAF represents the reflection of the electromagnetic signal on the wall. The WAF model is described by:

$$P_{\mathrm{L}}(d) = P_{\mathrm{L}}(d_0) - 10 \cdot a \cdot \log(\frac{d}{d_0}) + w_{\sigma} - \begin{cases} N_{\mathrm{W}} \cdot F_{\mathrm{WA}} & N_{\mathrm{W}} < C_{\mathrm{W}} \\ C_{\mathrm{W}} \cdot F_{\mathrm{WA}} & N_{\mathrm{W}} \geq C_{\mathrm{W}} \end{cases} \tag{2.4}$$

where N_{W} is the number of obstructions between the transmitter and the receiver; C_{W} is the maximum number of obstructions (walls); F_{WA} is the wall attenuation factor. In general, the values of a and F_{WA} depend on the building layout and construction material. The value of $P(d_0)$ can be derived empirically or obtained from the wireless network hardware specifications. The WAF model is widely used for indoor applications [15].

2.2.2 Localization using propagation model with WAF

The propagation model with WAF is employed for Wi-Fi based localization in this work. This model can be reformulated as:

$$S_{\mathrm{ob},l} = S_{\mathrm{ref},l} - 10 \cdot a_l \cdot \log(\frac{\rho_l}{\rho_{\mathrm{ref},l}}) - N_{\mathrm{w},l} \cdot F_{\mathrm{WA}} + \eta_{s,l} \tag{2.5}$$

where $S_{\mathrm{ob},l}$ denotes RSS from AP l at the position of the object; $S_{\mathrm{ref},l}$ denotes RSS from AP l at the reference point; a is the signal decaying rate; ρ_l represents the distance between the object and the AP l; $\rho_{\mathrm{ref},l}$ is the distance between the reference points and the AP l; N_{w} denotes number of walls in the signal path; F_{WA} is the wall attenuation factor. $\eta_{s,l}$ is the Gaussian noise term. To simplify Equation (2.5), it can be formulated as

$$\begin{gathered} S_{\mathrm{ob},l} = -10 \cdot a_l \cdot \log(\rho_l) + \Omega_l + \eta_{s,l} \\ \text{with} \\ \rho_l = \sqrt{(r_x - r_{\mathrm{AP}(l),x})^2 + (r_y - r_{\mathrm{AP}(l),y})^2 + (r_z - r_{\mathrm{AP}(l),z})^2} \end{gathered} \tag{2.6}$$

From above equation, $\mathbf{r} = \begin{bmatrix} r_x & r_y & r_z \end{bmatrix}^T$ denotes the position vector of the object and $\mathbf{r}_{\mathrm{AP}(l)}$ denotes the position of the AP l. a_l and Ω_l are the parameters of the signal propagation model from AP l. With real test data, they can be pre-estimated using linear regression. After the parameters are estimated, with at least three independent observation models ($l \geq 3$), the position of the object can be obtained using least squares estimation method.

2.3 Wi-Fi localization using RSS fingerprinting

2.3.1 Methods for database building

Fingerprinting methods are widely used for Wi-Fi RSS based localization. They operate in two distinguished phases: database building phase and location determination phase. In the first phase, a database or a radio map is constructed, which contains the signals emitted by the Wi-Fi AP on a grid of fix known positions.

This task can be performed by the collection of direct in-situ measurements, which is also called the empirical method. The received signal strength (RSS) at every survey point, which is recorded in the fingerprinting database, is the mean value of the collected samples.

The second method to build the database is using field propagation models instead of using real measurements, which is called propagation model based method. If the AP position and the survey point position are known, the RSS at the survey point can be calculated using the propagation models introduced in Section 2.2.1. In this work, the simplified WAF model shown with Equation (2.6) is employed for database building. The knowledge of the AP position is the prerequisite of this method.

According to the existing research results, the empirical method can provide a more reliable Wi-Fi fingerprinting database [1]. However, this method yields a labor-intensive survey phase and requires more implementation effort.

2.3.2 Methods for location determination

In the location determination phase, the localization can be done by comparison of the RSS measurement at the object position and the RSSs composing the database. Two localization methods are introduced in this work, which include nearest neighbor method and Kernel based method.

Nearest neighbor method

The nearest neighbor in signal space is a widely studied location determination method for Wi-Fi fingerprinting approaches [11]. The idea is to compute the Euclidean distance d_{SS} (in signal space) between the observed set of RSS measurements of the object S_{OB} and the RSSs at a fixed set of locations recorded in database S_{DB}, which is expressed as:

$$d_{\mathrm{SS},i} = \sqrt{\sum_{l=1}^{L} (S_{\mathrm{OB},\,l} - S_{\mathrm{DB},\,l}^{(i)})^2}, \qquad i \in I(j) \tag{2.7}$$

where L is the total number of available APs and $I(j)$ is the location index set of the database. Then the location i^*, i.e., $d_{\mathrm{RSS},i^*} = \min\left\{ d_{\mathrm{RSS},i} \,\middle|\, i \in I(j) \right\}$, is picked for positioning the object.

Kernel-based positioning method

Most RSS positioning efforts have been geared toward addressing the biggest challenge in fingerprinting: that of obtaining an estimate based on a new RSS observation. This essentially involves the calculation of a distance between the new RSS observation and the training record at each survey point. In the simplest case, the Euclidean distance is used to find the distance between the observation and the center of the training RSS vectors at each survey point. The location estimate is either the survey point whose fingerprint has the smallest distance to the observation (nearest neighbor algorithm) or the average of *k* closest survey points (KNN). Despite its simplicity, the Euclidean distance may fail to deliver adequate performance in cases where the distribution of RSS training vectors included in the fingerprints are nonconvex [16].

The kernel-based positioning method produces a position estimate without abandoning any training information. That is, a function $\hat{\mathbf{r}} = g\left(\mathbf{r}_{\mathbf{DB},1}, \mathbf{r}_{\mathbf{DB},2}, \ldots, \mathbf{r}_{\mathbf{DB},M}\right)$ is sought to estimate the object position $\hat{\mathbf{r}}$ with respected to all survey points (position vector: $\mathbf{r}_{\mathbf{DB}}$) recorded in the fingerprinting database. If $g(\cdot)$ is restricted to a kind of linear functions, the problem is reduced to determine a set of weights $W(\cdot)$ such that

$$\hat{\mathbf{r}} = \sum_{i=1}^{N} W\left(\mathbf{S}_{\mathrm{OB}}, \mathbf{F}_{\mathrm{DB},\,i}\right) \cdot \mathbf{r}_{\mathbf{DB},i} \tag{2.8}$$

where *N* is the total number of the survey points, $\mathbf{S}_{\mathrm{OB}}$ is the observation RSS vector from the object receiver and the fingerprinting database matrix $\mathbf{F}_{\mathrm{DB},i}$ with raw data is an *L*×*N* matrix defined as

$$\mathbf{F}_{\mathrm{DB},i} \triangleq \begin{bmatrix} S_i^{(1)}(1) & \cdots & S_i^{(1)}(N) \\ \vdots & & \vdots \\ S_i^{(L)}(1) & \cdots & S_i^{(L)}(N) \end{bmatrix} \tag{2.9}$$

where *L* is the number of APs, *N* implies the number of samples that are collected at each survey point, $S_i^{(l)}(n)$ indicates the RSS measurement of the *n*th sample collected from the AP *l* at survey point *i* in the fingerprinting database building phase.

The weight functions are required to be decreasing functions so that survey points whose training records closely match the observation are assigned with high weights. In particular, they should satisfy the following properties. [10]

1. $\sum_{i=1}^{M} W\left(\mathbf{S}_{\mathrm{OB}}, \mathbf{F}_{\mathrm{DB},i}\right) = 1$ so that the estimated position belongs to the convex hull defined by the set of survey positions. This can be achieved by including a normalization term in Equation (2.7).

2. $W\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right)$ is a monotonically decreasing function in both arguments with respect to a distance measure $d\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right)$

$$d\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right)\geq d\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},j}\right)\Rightarrow W\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right)\leq W\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},j}\right) \tag{2.10}$$

The average normalized inner product of training vectors and observation vector is chosen as the weight function.

$$\begin{aligned} W\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right) &= \frac{1}{N}\sum_{n=1}^{N}\frac{\left\langle \mathbf{S}_{\mathrm{OB}},\mathbf{S}_{i}\left(n\right)\right\rangle}{\left\|\mathbf{S}_{\mathrm{OB}}\right\|\cdot\left\|\mathbf{S}_{i}\left(n\right)\right\|} \\ d\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right)^{2} &= \frac{1}{N}\sum_{n=1}^{N}\left\|\frac{\mathbf{S}_{\mathrm{OB}}}{\left\|\mathbf{S}_{\mathrm{OB}}\right\|}-\frac{\mathbf{S}_{i}\left(n\right)}{\left\|\mathbf{S}_{i}\left(n\right)\right\|}\right\|^{2} \end{aligned} \tag{2.11}$$

In this case, as shown in Equation (2.11) the weight function is actually the average of the cosines between the observation vector and training vectors. The maximum distance occurs when these two vectors are orthogonal. However, according to [10], it is proved experimentally unpractical to use this measure since the angular measure between two vectors in L-dimensional space is relatively small. Furthermore, owing to the complexity of the RSS patterns, it is also required to improve the efficiency of the weight function (e.g., use nonlinear weight function instead of linear weight function). Figure 2.2 displays the complex distribution of the RSS patterns for six given survey points with real test data. The RSSs are shown in dBs in Figure 2.2.

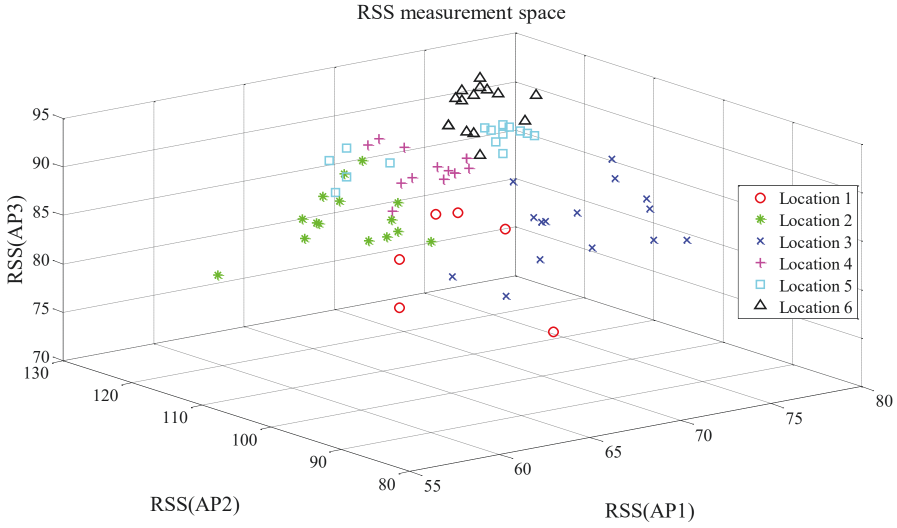


Figure 2.2: Example of RSS measurement space for different locations (three APs)

Hence, the RSS vectors are nonlinearly mapped to a higher (possibly infinite) dimensional space $\mathscr{F}$, where the computation of the weights takes place. The mapping is given by $\phi(\cdot):\mathbf{S}\in\mathbb{R}^L \mapsto \phi(\mathbf{S})\in\mathscr{F}$.

It seems to be intractable to calculate the weights in a possibly infinite dimensional space due to the computational complexity. However, the kernel trick can be used to calculate the inner product in high dimensional feature space without the need for explicit evaluation of the mapping function $\phi(\mathbf{S})$. That is, the kernel trick allows the replacement of inner products in high dimensional feature space by a kernel evaluation on the low dimensional input vectors [16]. So the kernelized weight function now becomes

$$\begin{aligned} W\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right) &= \frac{1}{N}\sum_{n=1}^{N}\frac{\left\langle \phi\left(\mathbf{S}_{\mathrm{OB}}\right),\phi\left(\mathbf{S}_i(n)\right)\right\rangle}{\left\|\phi\left(\mathbf{S}_{\mathrm{OB}}\right)\right\|\cdot\left\|\phi\left(\mathbf{S}_i(n)\right)\right\|} \\ &= \frac{1}{N}\sum_{n=1}^{N}\frac{K\left(\mathbf{S}_{\mathrm{OB}},\mathbf{S}_i(n)\right)}{\sqrt{K\left(\mathbf{S}_{\mathrm{OB}},\mathbf{S}_{\mathrm{OB}}\right)K\left(\mathbf{S}_i(n),\mathbf{S}_i(n)\right)}} \end{aligned} \tag{2.12}$$

A number of widely used kernel functions have been already developed (such as linear kernel, polynomial kernel, exponential kernel and Gaussian kernel). Among them, the Gaussian kernel has been widely studied and applied to plenty of pattern classification problems. Using Gaussian kernel guarantees the weight function outputs values in the interval of [0, 1]. Moreover, the mapped features $\phi(\mathbf{S}_i(1))$, $\phi(\mathbf{S}_i(2))$,…, $\phi(\mathbf{S}_i(N))$ are linearly independent in the high dimensional space. The Gaussian kernel is defined in Equation (2.13) [16].

$$K\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right)=\exp\left(-\frac{\left\|\boldsymbol{\alpha}-\boldsymbol{\beta}\right\|^2}{2\sigma^2}\right) \tag{2.13}$$

Then the weight function becomes

$$W\left(\mathbf{S}_{\mathrm{OB}},\mathbf{F}_{\mathrm{DB},i}\right)=\frac{1}{N}\sum_{n=1}^{N}\exp\left(\frac{-\left\|\mathbf{S}_{\mathrm{OB}}-\mathbf{S}_i(n)\right\|^2}{2\sigma^2}\right) \tag{2.14}$$

where parameter σ determines the width of the kernel.

An approach for determination of the kernel width parameter capitalizes on the knowledge available in Parzen-window density estimation. Specifically, given the sample vector $\mathbf{S}_i(n)$, n=1,..., N, from a sequence of independent and identically distributed random variables, the kernel density estimate (KDE) of the unknown density $\hat{f}(\mathbf{S}_{\text{OB}} \mid p_i)$ is

$$\hat{f}(\mathbf{S}_{\text{OB}} \mid p_i) = \frac{\sigma^{-L}}{N} \sum_{n=1}^{N} K(\mathbf{S}_{\text{OB}}, \mathbf{S}_i(n)) = \sigma^{-L} W(\mathbf{S}_{\text{OB}}, \mathbf{F}_{\text{DB},i}) \tag{2.15}$$

In general, the parameter σ is determined to minimize the asymptotic mean integrated square error (AMISE) between the estimated and true densities [16][17]. It can be shown that, for the multivariate KDE, the optimal bandwidth is on the order of $O\left(n^{\frac{-1}{4+L}}\right)$, corresponding to a minimum AMISE that is on the order of $O\left(n^{\frac{-4}{4+L}}\right)$. In particular, the following formula is recommended in [18] as a quick estimate of the parameter for a Gaussian kernel.

$$\begin{aligned} \sigma^* &= \left(\frac{4}{2L+1}\right)^{\frac{1}{L+4}} \hat{\sigma} n^{-\frac{1}{L+4}} \\ \hat{\sigma}^2 &= \frac{1}{L}\sum_{l=1}^{L} \sigma_{r_i^l}^2 \end{aligned} \tag{2.16}$$

where $\hat{\sigma}^2$ is the average of marginal variances.

And the scaled Gaussian weights is finally chosen for the location estimation, which is shown as

$$W(\mathbf{S}_{\text{OB}}, \mathbf{F}_{\text{DB},i}) = \frac{1}{N\left(\sqrt{2\pi\sigma}\right)^{L}} \sum_{n=1}^{N} \exp\left(-\frac{\left\|\mathbf{S}_{\text{OB}} - \mathbf{S}_i(n)\right\|^2}{2\sigma^2}\right) \tag{2.17}$$

In this case, the position estimate becomes

$$\hat{\mathbf{r}} = \frac{\sum_{i=1}^{M} W(\mathbf{S}_{\text{OB}}, \mathbf{F}_{\text{DB},i}) \cdot \mathbf{r}_{\mathbf{DB},i}}{\sum_{i=1}^{I} W(\mathbf{S}_{\text{OB}}, \mathbf{F}_{\text{DB},i})} = \frac{\sum_{i=1}^{M} \mathbf{r}_{\mathbf{DB},i} \sum_{n=1}^{N} \sigma_i^{-L} K(\mathbf{S}_{\text{OB}}, \mathbf{S}_i(n))}{\sum_{i=1}^{M}\sum_{n=1}^{N} \sigma_i^{-L} K(\mathbf{S}_{\text{OB}}, \mathbf{S}_i(n))} \tag{2.18}$$

where $K(\cdot)$ is the Gaussian kernel function.

2.4 Field experiment and numerical results

2.4.1 Hardware and test-bed

To test the stationary localization performance of each aforementioned approach, a field experiment has been carried out, which is in the corridor on the second floor of Hoelderlinstr.-F building at the University of Siegen. The test-bed scenario is shown in Figure 2.3. There are 11 available access points and 5 of them are with known positions, which are marked in Figure 2.3. The dimension of the whole floor is about 42 m by 37 m and the test field (the corridor) is about 25 m by 22 m. The fingerprinting database is built of RSS measurements at 182 survey points with a separation distance of 1 meter. 20 pre-surveyed points are chosen for propagation model parameter estimation using a least-square algorithm.

The experimental hardware shown in Figure 2.4 includes:

1) Wi-Fi Access points:

4 pieces of Linksys, TL-WN722NC, 2.4GHz, wireless-G broadband router

1 piece of D-LINK, Air plus Xtreme G, 802.11g, wireless access point

2) Wireless receiver:

1 piece of TP-LINK TL-WN722N 150Mbps Wi-Fi adapter

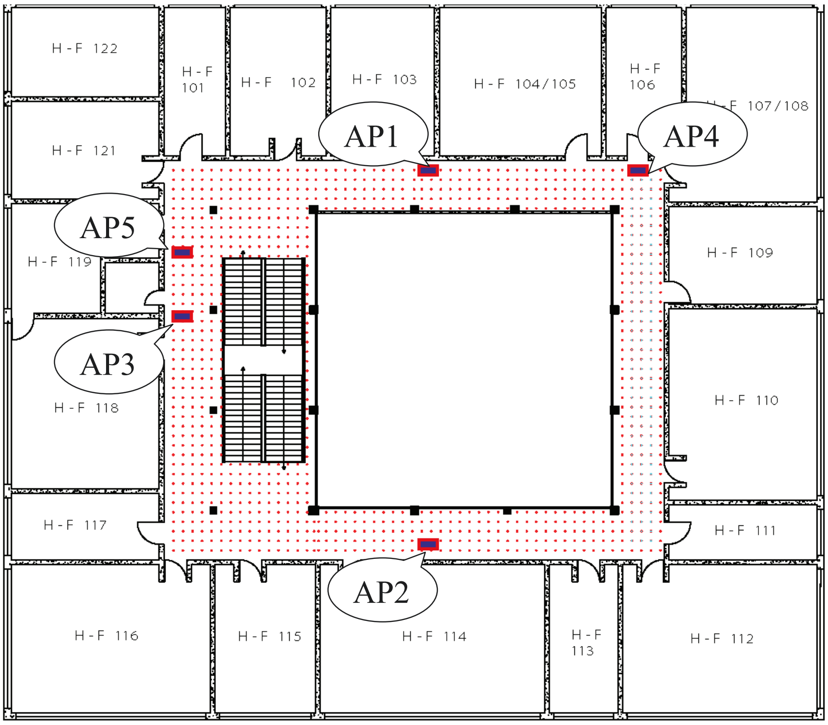
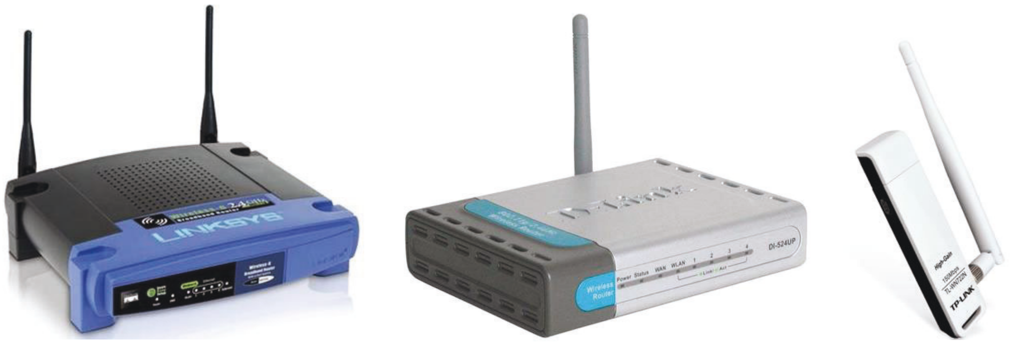


Figure 2.3: Experiment test-bed



(a) (b) (c)

Figure 2.4: Experimental hardware: (a) Linksys router; (b) D-Link router; (c) TP-Link Wi-Fi adapter

2.4.2 Results and analysis

100 random chosen blind points are localized with three Wi-Fi based approaches respectively. The first method is empirical fingerprinting (database building with the empirical method); the second one is model based fingerprinting (database building with WAF propagation model); the third one is positioning directly using WAF propagation model for position estimation. Both fingerprinting approaches employ the nearest neighbor method for location determination.

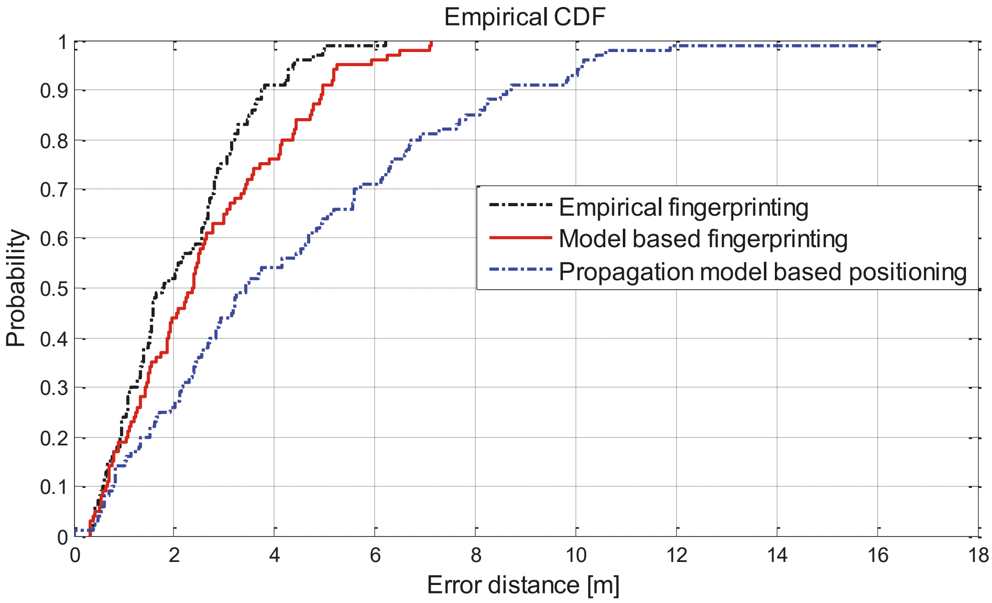


Figure 2.5: Positioning error cumulative distribution functions (CDFs)

Table 2.1: Error means and standard deviations

	Empirical FP	Model based FP	Propagation model
Mean error [m]	2.48	2.93	4.33
Standard deviation [m]	1.81	2.11	3.18

The positioning results are described with error distances shown in Figure 2.5 and Table 2.1. It can be found that empirical fingerprinting provides better positioning performance (mean error: 2.48 m) than the one based on WAF model (mean error: 2.93 m). That is, compared to the model based fingerprinting, the empirical method can not only build a more reliable database with real RSS collection but also take advantage of the RSS information from the APs with unknown positions. The model based fingerprinting yields better performance than the method directly using the propagation model (mean error: 4.33 m). That is, the fingerprinting approaches can take advantage of map constrained information.

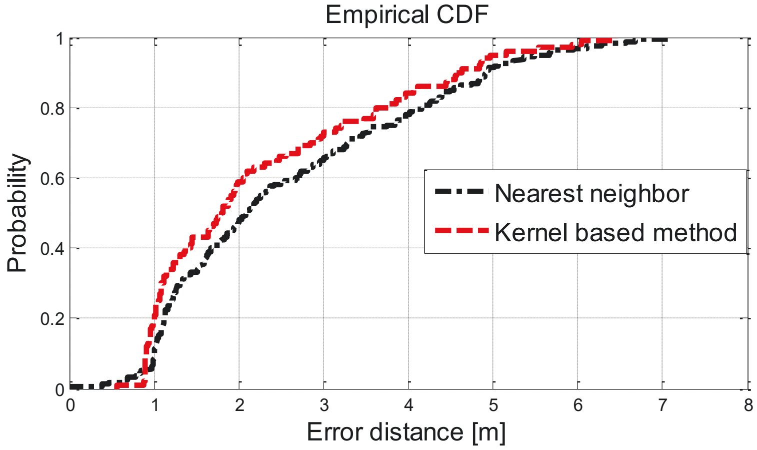


Figure 2.6: Positioning error CDFs

Table 2.2: Error means and standard deviations

	Kernel based method	Nearest neighbor
Mean [m]	2.20	2.48
Standard deviation [m]	1.44	1.81

To show the influence on fingerprinting positioning performance with different location determination methods, nearest neighbor and Kernel based method are tested and compared. The database is built with the empirical method. The results are shown in

Figure 2.6 and Table 2.2. It can be found that the Kernel based fingerprinting can provide a better positioning result.

2.5 Summary

In this chapter, the background and concept of Wi-Fi based positioning are discussed. The Wi-Fi localization approaches are explored, which include the one directly using radio propagation model and the ones employing RSS fingerprinting. The fingerprinting approach consists of two steps: database building phase and localization phase. The database can be built with empirical method or model based method, and the localization phase can be done with nearest neighbor method or kernel based method. The field experiment is performed, and the results show that 1) the approach directly using the propagation model yields higher positioning errors than the fingerprinting approaches; 2) the empirical fingerprinting provides a better positioning performance than the model based fingerprinting; 3) kernel based localization method yields lower positioning errors comparing to the nearest neighbor method.

Chapter 3
Indoor Vehicle Navigation Using Enhanced INS/Wi-Fi Integration

For indoor continuous navigation applications, especially those related to mobile object tracking, other approaches for Wi-Fi based continuous localization can be employed, such as Viterbi-like algorithms and Baum-Welch algorithms [19][20]. However, they are based on the "most likely" trajectory or the path model which limits their robustness in practical applications. The micro-electromechanical systems (MEMS) based IMU is widely used in navigation applications. The IMU can provide the motion information of the object with a relatively accurate output within a short time and a high update rate. Nevertheless, for a low-cost IMU, its navigation solutions drift quickly over time due to the accumulation of sensor errors (i.e., sensor bias, noise, scale factors, etc) [21]. Therefore, in outdoor environments, the IMU is often integrated with the global positioning system (GPS) receiver. The integration of INS and GPS has been proven to be a reliable solution for continuous outdoor navigation [22]-[28]. Considering the complementary nature of INS and Wi-Fi positioning, the combination of both systems is expected to yield a synergetic effect resulting in higher navigation performance.

In this chapter, system models of INS/Wi-Fi integration are derived in Section 3.1. The state propagation model is provided by INS mechanization and the observation model is from Wi-Fi positioning approaches. The unscented Kalman filter is employed for system integration. To further improve the performance of the integration, the enhancements using vehicle constraints and adaptive Kalman filtering are presented in Section 3.2. In Section 3.3, one field experiment is made and the results show the advantages of the INS/Wi-Fi integrated system and the enhanced integration with respect to the standalone Wi-Fi positioning approaches.

3.1 System modelling for INS/Wi-Fi integration

3.1.1 INS process model using Euler angles

There are four reference frames related to indoor navigation, which are the inertial frame, earth frame, navigation frame and body frame. The inertial frame is a reference frame in which Newton’s law of motion applies. All the inertial sensors make measurements relative to an inertial frame [24]. We can take any point as its origin in the inertial coordinate system, and consider three mutually perpendicular directions as its axes. The earth frame has its origin fixed on the center of the earth. The navigation frame is attached to a fixed point on the surface of the earth at some convenient point for local measurements. The body frame is rigidly attached to the vehicle of interest, usually at a fixed point such as the gravity center of the vehicle, which is also the origin of the body coordinate system [29]. In this work, two frames are mainly considered, which are the local navigation frame and the body frame. The vector in the body frame is denoted as $\boldsymbol{\upsilon}_{\mathrm{b}}$ while the local navigation frame is the default frame in this work, which means $\boldsymbol{\upsilon} = \boldsymbol{\upsilon}_{\mathrm{n}}$.

Regarding the system modelling for our integrated system, the system state vector $\mathbf{X}$ is composed of position $\mathbf{r}$, velocity $\mathbf{v}$ and attitude $\boldsymbol{\psi}$ in the navigation frame. The strap-down INS mechanization model is defined as the system process model [30]. For low-cost MEMS based IMUs, the effects from the earth rotation cannot be observed, so the Coriolis and centrifugal terms are not considered in the INS process model. In this model, the gravity is assumed as a constant and the transport rate is no longer considered for simplicity [31]. The simplified mechanization model in discrete time can be expressed in navigation frame as:

$$\begin{aligned}
\mathbf{r}_k &= \mathbf{r}_{k-1} + \mathbf{v}_{k-1} \cdot \Delta t + \mathbf{w}_{\mathbf{r},k-1} \\
\mathbf{v}_k &= \mathbf{v}_{k-1} + \left[\mathbf{C}^{\mathrm{n}}_{\mathrm{b},k-1} \cdot (\tilde{\mathbf{f}}_{\mathrm{b},k-1} - \hat{\mathbf{f}}^{\mathrm{bias}}_{\mathrm{b}}) + \mathbf{g} \right] \cdot \Delta t + \mathbf{w}_{\mathbf{v},k-1} \\
\boldsymbol{\psi}_k &= \boldsymbol{\psi}_{k-1} + \boldsymbol{\Phi}^{\mathrm{n}}_{\mathrm{b},k-1} \cdot (\tilde{\boldsymbol{\omega}}_{\mathrm{b},k-1} - \hat{\boldsymbol{\omega}}^{\mathrm{bias}}_{\mathrm{b}}) \cdot \Delta t + \mathbf{w}_{\boldsymbol{\psi},k-1}
\end{aligned} \tag{3.1}$$

where Δt is the sampling time; $\tilde{\mathbf{f}}_{\mathrm{b}}$ is the acceleration measurement vector from IMU (accelerometers); $\tilde{\boldsymbol{\omega}}_{\mathrm{b}}$ represents the measurement vector of angular rate from IMU (gyroscopes); $\hat{\mathbf{f}}^{\mathrm{bias}}_{\mathrm{b}}$ and $\hat{\boldsymbol{\omega}}^{\mathrm{bias}}_{\mathrm{b}}$ are the pre-estimated accelerometer and gyroscope bias error terms; $\mathbf{w}$ terms represent the corresponding white noise of the model; $\mathbf{C}^{\mathrm{n}}_{\mathrm{b}}$ is the frame rotation matrix from the body frame to navigation frame. It is assumed that the object has an attitude which can be obtained by three successive rotation angles ϕ , θ and ψ around the x, y and z axis.

$$\mathbf{C}^{\mathrm{n}}_{\mathrm{b}} = \begin{bmatrix} \mathrm{c}\varphi\mathrm{c}\phi & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{s}\theta - \mathrm{s}\varphi\mathrm{c}\theta & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{c}\theta + \mathrm{s}\varphi\mathrm{s}\theta \\ \mathrm{s}\varphi\mathrm{c}\phi & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{s}\theta + \mathrm{c}\varphi\mathrm{c}\theta & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{c}\theta - \mathrm{c}\varphi\mathrm{s}\theta \\ -\mathrm{s}\phi & \mathrm{c}\phi\mathrm{s}\theta & \mathrm{c}\phi\mathrm{c}\theta \end{bmatrix} \tag{3.2}$$

$\Phi_{\mathrm{b}}^{\mathrm{n}}$ is the rotation rate matrix between body frame and navigation frame, and it can be expressed as:

$$\mathbf{\Phi}_{\mathrm{b},k}^{\mathrm{n}} = \begin{bmatrix} 1 & \mathrm{s}\theta\mathrm{t}\phi & \mathrm{c}\theta\mathrm{t}\phi \\ 0 & \mathrm{c}\theta & -\mathrm{s}\theta \\ 0 & \mathrm{s}\theta/\mathrm{c}\theta & \mathrm{c}\theta/\mathrm{c}\phi \end{bmatrix} \tag{3.3}$$

In the Equation (3.2) and (3.3), $\mathrm{c}X = \cos X$, $\mathrm{s}X = \sin X$, $\mathrm{t}X = \tan X$ and θ, ϕ, φ represent the roll, pitch and yaw respectively. Equation (3.1) is employed as the system propagation model in the integration system.

3.1.2 Observation models with Wi-Fi positioning

The simplified propagation model is derived in Section 2.2.1 as Equation (2.6). It can be rewritten as the measurement equation with respect to the position of the object:

$$S_{\mathrm{OB},l} = -10 \cdot a_l \cdot \log(\sqrt{(r_x - r_{\mathrm{AP}(l),x})^2 + (r_y - r_{\mathrm{AP}(l),y})^2 + (r_z - r_{\mathrm{AP}(l),z})^2}) + \Omega_l + \eta_{s,l} \tag{3.4}$$

where $S_{\mathrm{OB},l}$ is the received signal strength measurements from AP l; $\mathbf{r}$ and $\mathbf{r}_{\mathrm{AP}(n)}$ represent the position vectors of the object and AP l respectively; $\eta_{s,l}$ is additional Gaussian noise term; a_n and Ω_l are the Wi-Fi signal related parameters which can be pre-estimated. It can be found that Equation (3.4) can be directly used as the observation model for the INS/Wi-Fi integration, which yields a tightly coupled integration structure.

If a Wi-Fi fingerprinting approach is employed for the integration instead of direct usage of the propagation model, the RSS measurements are pre-processed by a fingerprinting algorithm and the estimated position vector is the input of measurement update. In this case, the integration yields a loosely coupled structure. The tightly coupled and loosely coupled integration structures are illustrated in Figure 3.1.

Due to the nonlinearities of the system models, the unscented Kalman filter (UKF) is used for the tightly/loosely coupled integration systems. The discrete-time UKF is described in Section 3.1.3.

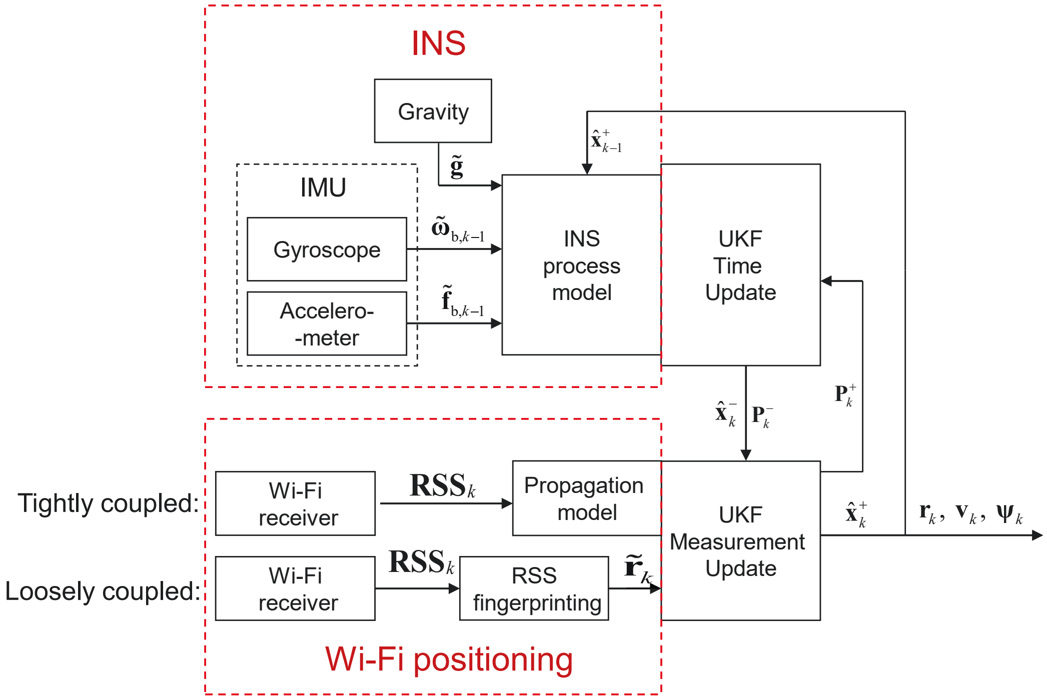


Figure 3.1: Diagram of INS/Wi-Fi integration with tightly/loosely coupled structure

3.1.3 Unscented Kalman filtering

The derivations of time update and measurement update equations of the UKF are shown as follows.

A discrete system model with additional zero mean Gaussian white noises is formulated as:

$$\begin{aligned}\mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1}\\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k) + \boldsymbol{\eta}_k\\ \mathbf{w}_k &\sim N\left(\mathbf{0},\mathbf{Q}_k\right)\\ \boldsymbol{\eta}_k &\sim N\left(\mathbf{0},\mathbf{R}_k\right)\end{aligned} \tag{3.5}$$

Among many UKF algorithms, the following algorithm with $2n$ equally weighted sigma points is employed in this work [32]:

1) Initialize the state and covariance

$$\begin{aligned}\hat{\mathbf{x}}_0^+ &= E(\mathbf{x}_0)\\ \mathbf{P}_0^+ &= E\left[\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+\right)\left(\mathbf{x}_0 - \hat{\mathbf{x}}_0^+\right)^T\right]\end{aligned} \tag{3.6}$$

2) Choose the sigma points (SPs) for time update (n denotes the dimension of the state vector)

$$\begin{aligned}
&\hat{\mathbf{x}}_{k-1}^{(i)} = \hat{\mathbf{x}}_{k-1}^{+} + \tilde{\mathbf{x}}^{(i)} \qquad i = 1,2,...,2n \\
&\tilde{\mathbf{x}}^{(i)} = \left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T \qquad i = 1,2,...,n \\
&\tilde{\mathbf{x}}^{(n+i)} = -\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T \qquad i = 1,2,...,n
\end{aligned} \tag{3.7}$$

3) Time update to obtain the priori state estimate and covariance

$$\begin{aligned}
&\hat{\mathbf{x}}_{k}^{(i)} = \mathbf{f}(\hat{\mathbf{x}}_{k-1}^{(i)}) \\
&\hat{\mathbf{x}}_{k}^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\hat{\mathbf{x}}_{k}^{(i)} \\
&\mathbf{P}_{k}^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\left(\hat{\mathbf{x}}_{k}^{(i)} - \hat{\mathbf{x}}_{k}^{-}\right)\left(\hat{\mathbf{x}}_{k}^{(i)} - \hat{\mathbf{x}}_{k}^{-}\right)^T + \mathbf{Q}_{k-1}
\end{aligned} \tag{3.8}$$

4) Choose SPs for measurement update

$$\begin{aligned}
&\hat{\mathbf{x}}_{k}^{(i)} = \hat{\mathbf{x}}_{k}^{-} + \tilde{\mathbf{x}}^{(i)} \qquad i = 1,2,...,2n \\
&\tilde{\mathbf{x}}^{(i)} = \left(\sqrt{n\mathbf{P}_{k}^{-}}\right)_i^T \qquad i = 1,2,...,n \\
&\tilde{\mathbf{x}}^{(n+i)} = -\left(\sqrt{n\mathbf{P}_{k}^{-}}\right)_i^T \qquad i = 1,2,...,n
\end{aligned} \tag{3.9}$$

5) Predict measurements

$$\begin{aligned}
&\hat{\mathbf{y}}_{k}^{(i)} = \mathbf{h}(\hat{\mathbf{x}}_{k}^{(i)}) \\
&\hat{\mathbf{y}}_{k} = \frac{1}{2n}\sum_{i=1}^{2n}\hat{\mathbf{y}}_{k}^{(i)}
\end{aligned} \tag{3.10}$$

6) Estimate the covariance and cross covariance

$$\begin{aligned}
&\mathbf{P}_{k}^{\mathbf{y}} = \frac{1}{2n}\sum_{i=1}^{2n}\left[\left(\hat{\mathbf{y}}_{k}^{(i)} - \hat{\mathbf{y}}_{k}\right)\left(\hat{\mathbf{y}}_{k}^{(i)} - \hat{\mathbf{y}}_{k}\right)^T\right] + \mathbf{R}_{k} \\
&\mathbf{P}_{k}^{\mathbf{xy}} = \frac{1}{2n}\sum_{i=1}^{2n}\left[\left(\hat{\mathbf{x}}_{k}^{(i)} - \hat{\mathbf{x}}_{k}^{-}\right)\left(\hat{\mathbf{y}}_{k}^{(i)} - \hat{\mathbf{y}}_{k}\right)^T\right]
\end{aligned} \tag{3.11}$$

7) Measurement update to obtain the posteriori state estimate and covariance

$$\begin{aligned}
&\mathbf{K}_{k} = \mathbf{P}_{k}^{\mathbf{xy}}\left(\mathbf{P}_{k}^{\mathbf{y}}\right)^{-1} \\
&\hat{\mathbf{x}}_{k}^{+} = \hat{\mathbf{x}}_{k}^{-} + \mathbf{K}_{k}\left(\mathbf{y}_{k} - \hat{\mathbf{y}}_{k}\right) \\
&\mathbf{P}_{k}^{+} = \mathbf{P}_{k}^{-} - \mathbf{K}_{k}\mathbf{P}_{k}^{\mathbf{y}}\mathbf{K}_{k}^{T}
\end{aligned} \tag{3.12}$$

The UKF uses a deterministic sampling approach to capture the mean and covariance estimate with a minimal set of sample points, and its estimation accuracy can reach up to the third order of the Taylor series expansion for any nonlinearity [33]. While the extended Kalman filter (EKF) based on linearizing the nonlinear system with Jacobian matrices, which yields a first order approximation [34][35]. The detailed description and derivation of the UKF are introduced in Appendix C.

3.2 Enhancements using adaptive Kalman filtering and vehicle constraints

3.2.1 Adaptive Kalman filtering (AKF)

When the prior statistics of the system process and measurement noises are known, the KF methods can provide reliable system estimates. In most practical applications, these statistics are not possible to be known analytically, because they are strongly related to the type of application at hand, in our case, the dynamic of the complicate Wi-Fi signal environments. In such cases, using the prior given statistics may degrade the system performance. In order to improve the estimation accuracy, the prompt reflection of the changes in the noise statistics from the external influences is required. The AKF is designed for adapting the measurement and the process noise statistics. Innovation-based AKF and residual based AKF are the existing approaches for estimating time-varying noise covariance matrices [36]. However, innovation based AKF may cause numerical problems in calculation procedures and hence, make the filter diverge [37]. In this work, residual-based AKF is used.

For the INS/Wi-Fi integrated system discussed in this work, the IMU measurements are used in the system dynamic model. The error statistics can be found in the IMU product specifications, or most of them can be detected through self-alignment and calibration processes. Therefore, the parameters in the process noise covariance matrix '**Q**' can be determined approximately. However, the Wi-Fi positioning measurements used in the system observation model are related to many external influences such as multipath, wall attenuation and Wi-Fi signal fluctuations. Hence, in this work, with the assumption that the statistical information about the '**Q**' is known approximately; the measurement noise covariance '**R**' is to be estimated.

In residual based AKF, the residual sequence is computed as the discrepancy between the incoming measurements and the ones calculated from the observation model:

$$\mathbf{z}_k = \mathbf{y}_k - \hat{\mathbf{y}}_k^+ \tag{3.13}$$

where $\hat{\mathbf{y}}_k^+ = \mathbf{h}_k(\hat{\mathbf{x}}_k^+)$. An estimate of the residual sequence covariance matrix is obtained by averaging the previous residual sequence over a window length *N*, which is shown as:

$$\hat{\mathbf{C}}_{\boldsymbol{\eta},k} = \frac{1}{N} \sum_{j=k-N+1}^{k} \mathbf{z}_j \mathbf{z}_j^T \tag{3.14}$$

The estimated measurement error covariance matrix $\hat{\mathbf{R}}_k$ for the UKF discussed in Section 2 can be estimated adaptively as:

$$\hat{\mathbf{R}}_k = \hat{\mathbf{C}}_{\boldsymbol{\eta},k} + \frac{1}{2n} \sum_{i=1}^{2n} \left[\left(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k \right) \left(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k \right)^T \right] \tag{3.15}$$

3.2.2 Vehicle constraints

Vehicle constraints take advantage of the knowledge of the vehicle's dynamics and the physical conditions based on practical application experiences, which can be used to reduce the system drift error caused by IMU biases [38]. In this work, three constraints are explored, which are body velocity constraint (BVC), constant height constraint (HC) and body angular velocity constraint (AVC). The constraints are used as pseudo-measurements to be added to the observation model of the integrated system.

Body Velocity Constraint

The body velocity constraint is based on the assumption that vehicles travel on the ground without significant side sliding. Therefore, velocities in the body frame along the y and z axes can be assumed to be almost zero ($v_{\mathrm{b},y} \cong 0$ and $v_{\mathrm{b,z}} \cong 0$) [39].

The body velocity constraint can be employed as a pseudo-measurement added to the observation model for our system and the corresponding measurement equation is derived as:

$$\mathbf{y}_{\mathbf{vc},k} = \begin{bmatrix} \tilde{v}_{\mathrm{b},y} \\ \tilde{v}_{\mathrm{b,z}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \mathbf{h}_{\mathbf{vc}}(\mathbf{x}_k) + \boldsymbol{\eta}_{\mathbf{vc},k} \tag{3.16}$$

With

$$\mathbf{h}_{\mathbf{vc}}(\mathbf{x})=\begin{bmatrix} \mathrm{s}\varphi\mathrm{c}\phi & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{s}\theta+\mathrm{c}\varphi\mathrm{c}\theta & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{c}\theta-\mathrm{c}\varphi\mathrm{s}\theta \\ -\mathrm{s}\phi & \mathrm{c}\phi\mathrm{s}\theta & \mathrm{c}\phi\mathrm{c}\theta \end{bmatrix}\cdot\mathbf{v} \tag{3.17}$$

where **v** denotes the velocity of the object and $\boldsymbol{\eta}_{\mathbf{vc},k}$ is the measurement noise vector.

Height Constraint

The height constraint utilizes the fact that elevation remains almost constant for short time durations ($r_{z,k}=r_{\mathrm{hc}}$ and $v_{z,k}=0$). The pseudo-measurement equation can be derived as

$$\mathbf{y}_{\mathrm{hc},k}=\begin{bmatrix} r_{\mathrm{hc}} \\ 0 \end{bmatrix}=\mathbf{h}_{\mathrm{hc}}(\mathbf{x}_k)+\boldsymbol{\eta}_{\mathrm{hc},k} \tag{3.18}$$

With

$$\mathbf{h}_{\mathrm{hc}}(\mathbf{x})=\begin{bmatrix} r_z \\ v_z \end{bmatrix} \tag{3.19}$$

Angular Velocity Constraint

Based on the scenarios in which vehicles travel on the ground, the angular velocity constraint assumes that among the three angular velocities, only the change in the heading angle is not zero [39]. The corresponding pseudo-measurement equation is:

$$\mathbf{y}_{\mathrm{avc},k}=\begin{bmatrix} 0 \\ 0 \end{bmatrix}=\mathbf{h}_{\mathrm{avc}}(\mathbf{x}_k)+\boldsymbol{\eta}_{\mathrm{avc},k} \tag{3.20}$$

With

$$\mathbf{h}_{\mathrm{hc}}(\mathbf{x}_k)=\begin{bmatrix} \varphi_k-\hat{\varphi}_{k-1} \\ \phi_k-\hat{\phi}_{k-1} \end{bmatrix} \tag{3.21}$$

where $\hat{\varphi}_{k-1}$ and $\hat{\phi}_{k-1}$ are estimated roll and pitch from the previous step.

3.2.3 Enhanced INS/Wi-Fi integration

After employing AKF and vehicle constraints in the INS/Wi-Fi integrated system discussed in Section 3.1, an enhanced system can be obtained, which is shown in Figure 3.2. The measurement noise covariance is re-estimated with AKF algorithm. The vehicle constraints provide the extra observation models, described with Equation (3.16), (3.18) and (3.20), for the system measurement update.

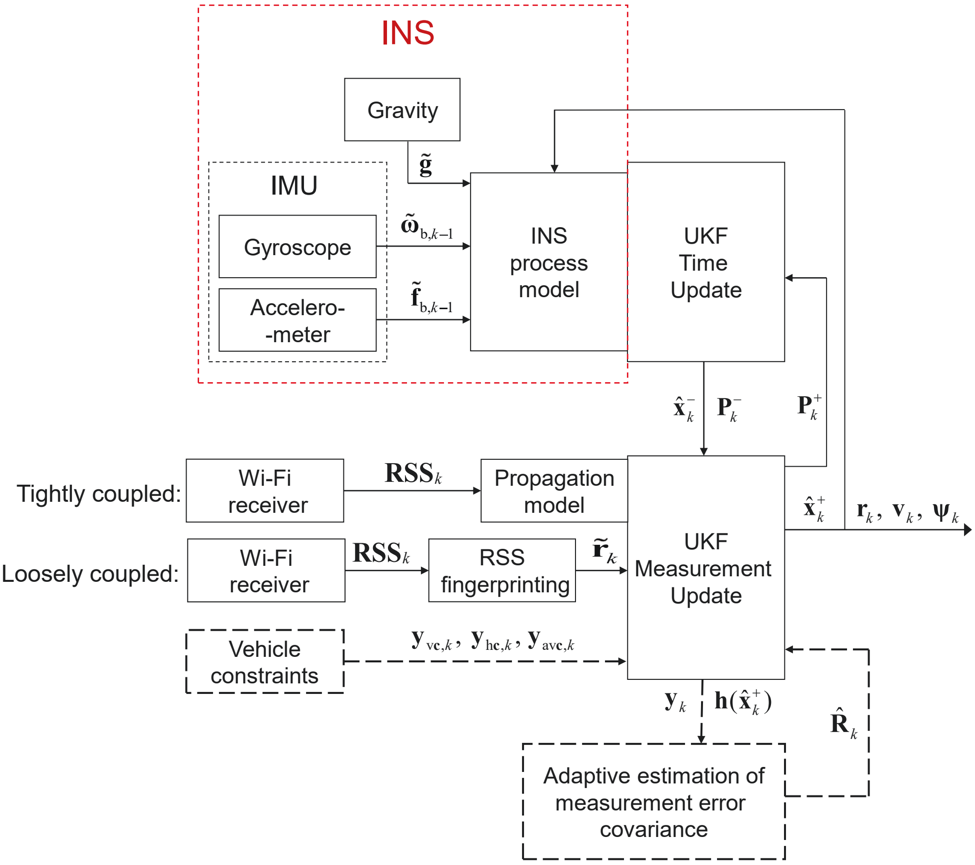


Figure 3.2: Diagram of enhanced INS/Wi-Fi integration

3.3 Field experiment and results

3.3.1 Hardware and test-bed

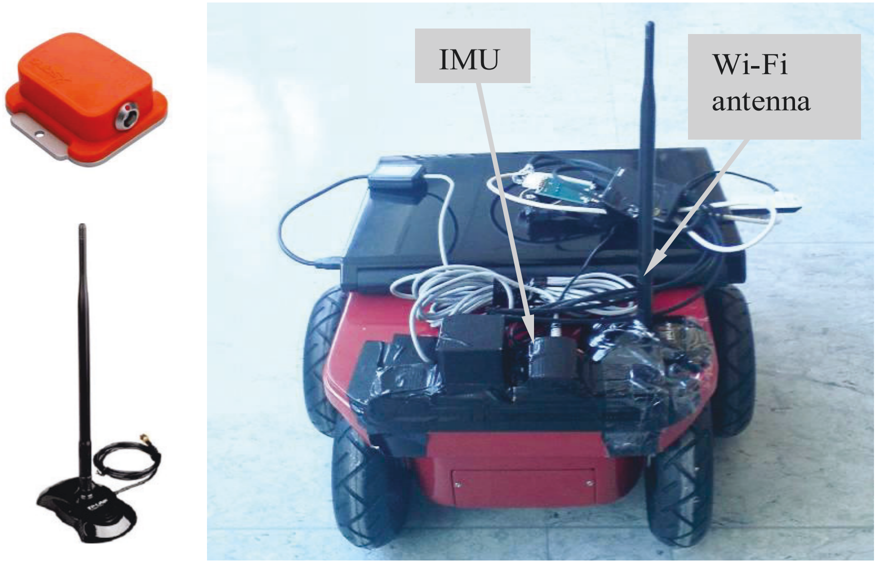


Figure 3.3: Experimental hardware

Table 3.1: Specification and parameters of Xsens MTi IMU [40]

		rate of turn	acceleration	magnetic field	temperature
Unit		[deg/s]	[m/s^2]	[mGauss]	[°C]
Dimensions		3 axes	3 axes	3 axes	-
Full Scale	[units]	+/- 300*	+/- 50	+/- 750	-55...+125
Linearity	[% of FS]	0.1	0.2	0.2	<1
Bias stability	[units 1σ][11]	1	0.02	0.1	0.5[12]
Scale factor stability	[% 1σ][11]	-	0.03	0.5	-
Noise density	[units /√Hz]	0.05[13]	0.002	0.5 (1σ)[14]	-
Alignment error(15)	[deg]	0.1	0.1	0.1	-
Bandwidth	[Hz]	40	30	10	-
A/D resolution	[bits]	16	16	16	12

A mobile robot, which is shown in Figure 3.3 has been employed as the object that needs to be tracked. The Wi-Fi measurements are collected from a low-cost Wi-Fi antenna (TP-Link TL-ANT2408C) and a MEMS-based IMU (Xsens MTi) is used in the test. The update rate of the IMU measurement is 50Hz while the update rate of the Wi-Fi RSS is 1Hz. The specification and parameters of the IMU are shown in Table 3.1.

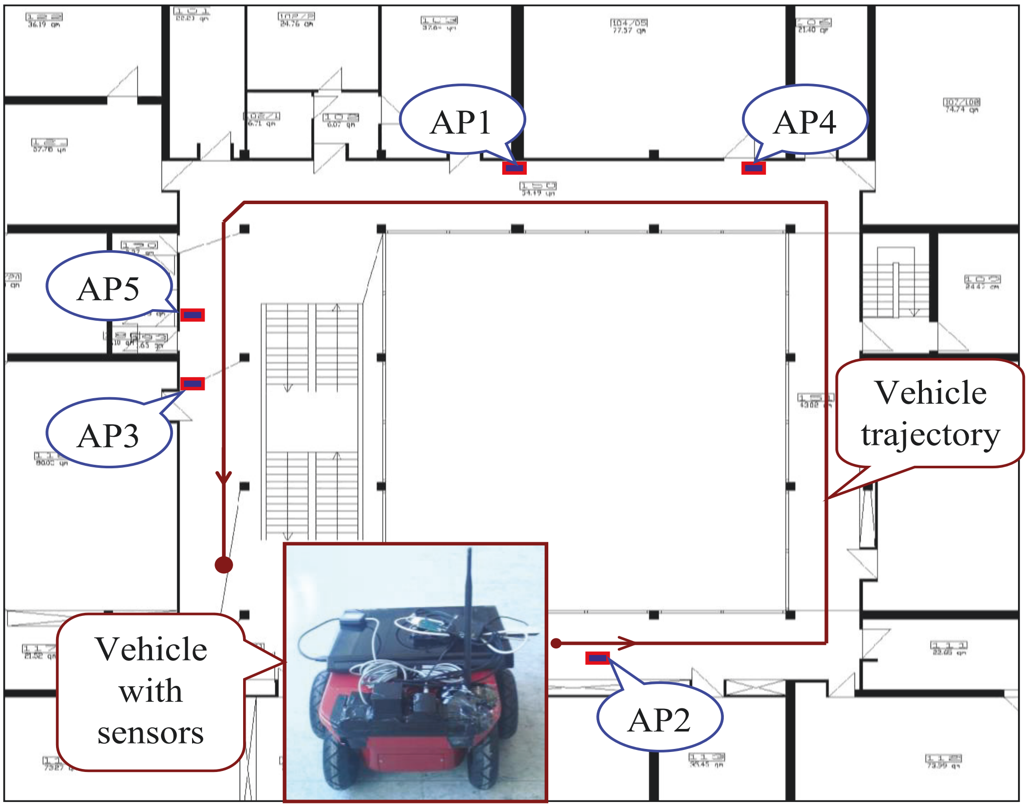


Figure 3.4: Experimental scenario

The field experiment is carried out in the same test-bed introduced in Section 2.4.1. There are 5 available APs with known positions (shown in Figure 3.4) and 6 APs with unknown positions. To implement the empirical fingerprinting method, about 182 database points have been collected with the RSS measurements from 11 APs in database building phase. The distance between two adjacent points in the corridor is 1 meter. For propagation model based fingerprinting, 20 pre-surveyed points have been used to estimate parameters of the propagation model.

The initial IMU bias errors are detected by averaging IMU measurements when the platform is stationary. The robot movement lasts 221 s. The reference trajectory (shown in Fig. 2) has been measured manually. The RSS measurements received from the APs with known positions are shown in Figure 3.5. When the robot is approaching a certain AP, the received signal strength increases, and, vice versa, when the vehicle is leaving, the signal power decreases.

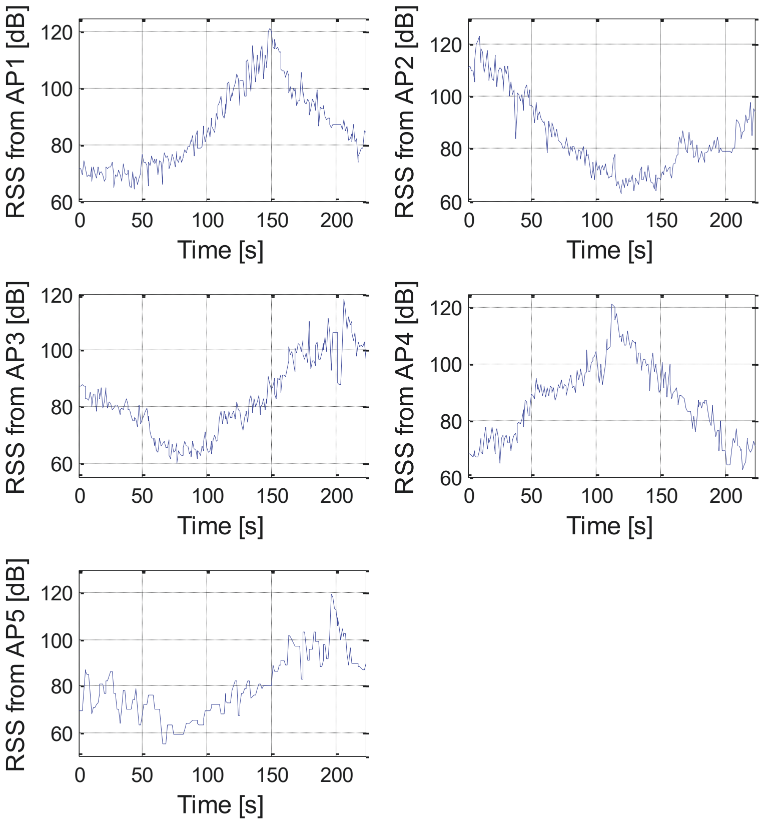


Figure 3.5: RSS measurements during the robot movement

3.3.2 Results and comparison

Figure 3.6 shows the first group of the position estimation results separately using the three discussed approaches: (1) Wi-Fi empirical fingerprinting, (2) integration of INS and empirical fingerprinting using UKF and (3) enhanced INS/Wi-Fi integration aided with AKF and vehicle constraints. The second group of estimation results using propagation-model-based fingerprinting and the corresponding integrated systems are shown in Figure 3.7. It should be noted that the positioning errors mentioned in this work denote the distances between the estimated positions and the corresponding reference positions, which are magnitudes of the positioning error vectors.

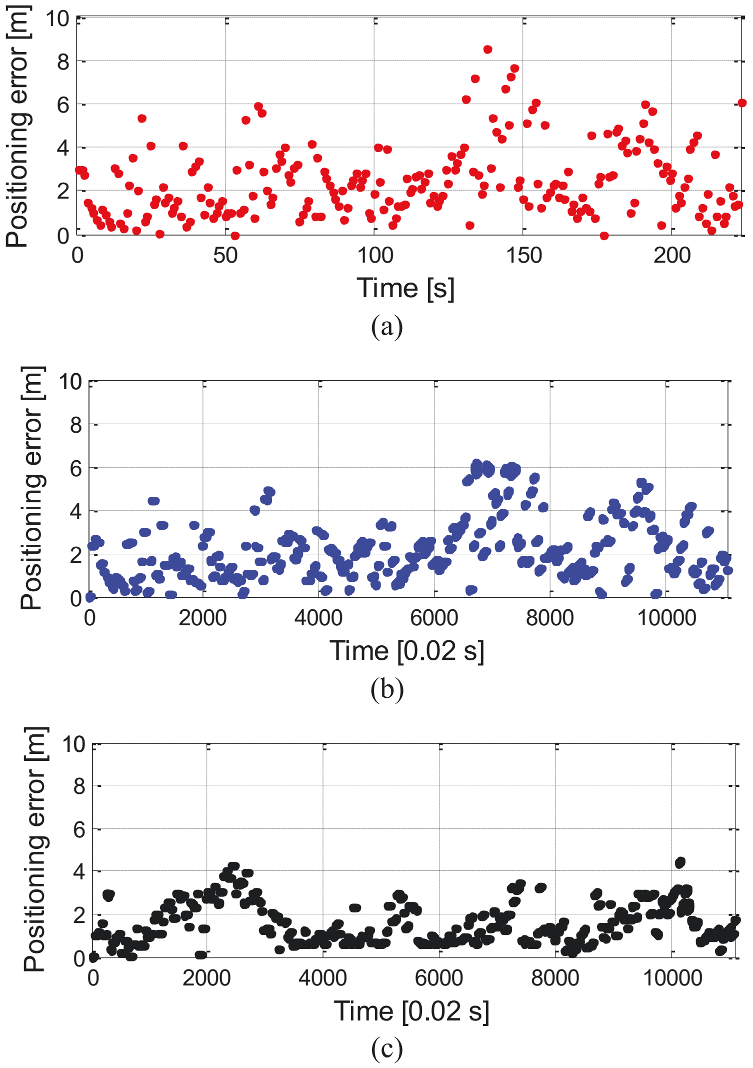
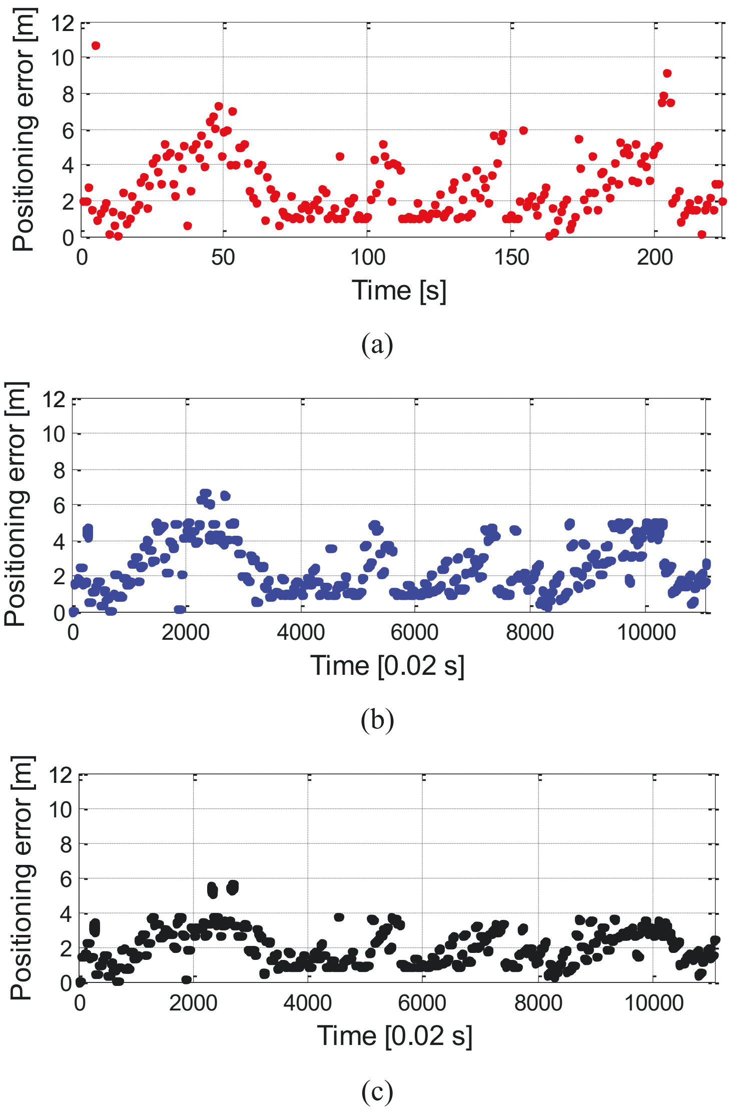


Figure 3.6: Positioning errors (1st group) using: (a) Wi-Fi empirical fingerprinting positioning; (b) integration of INS and empirical fingerprinting; (c) enhanced INS/Wi-Fi integration



(a)

(b)

(c)

Figure 3.7: Positioning errors (2nd group) using: (a) propagation model based fingerprinting positioning; (b) integration of INS and model based fingerprinting; (c) enhanced INS/Wi-Fi integration

For a better comparison, the error cumulative distribution functions (CDFs) are plotted in Figure 3.8 and Figure 3.9; the error properties are shown in Table 3.2 and Table 3.3. It can be found that the enhanced integration yields the best navigation performance in both groups and the integrated system provides lower positioning error than standalone Wi-Fi positioning.

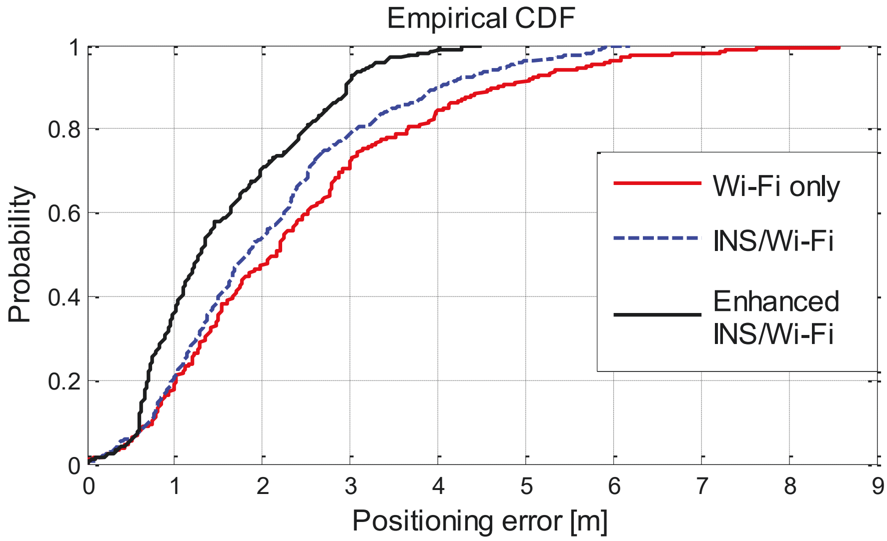


Figure 3.8: Positioning error CDFs (1st group)

Table 3.2: Position error comparison (1st group)

Method / Error [m]	Wi-Fi only	INS/Wi-Fi	Enhanced INS/Wi-Fi
Maximum	8.57	6.18	4.49
Median	2.16	1.83	1.29
Mean	2.42	2.11	1.56
Standard deviation	1.62	1.32	0.94

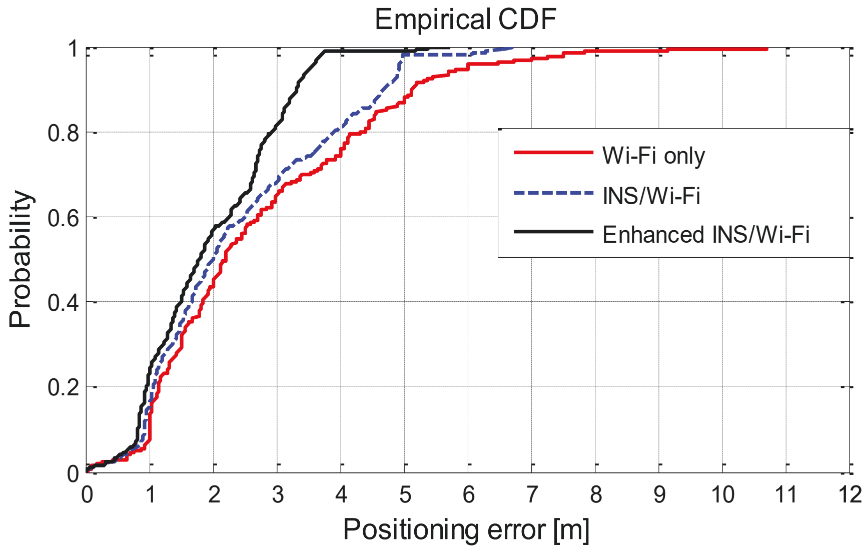


Figure 3.9: Positioning error CDFs (2nd group)

Table 3.3: Position error comparison (2nd group)

Method / Error [m]	Wi-Fi only	INS/Wi-Fi	Enhanced INS/Wi-Fi
Maximum	10.70	6.71	5.78
Median	2.19	1.99	1.80
Mean	2.74	2.41	1.96
Standard deviation	1.73	1.45	1.01

3.4 Summary

In this chapter, the indoor vehicle navigation with integration of INS and Wi-Fi based positioning is discussed. The system time update model is provided by strap-down INS mechanization, and the measurement update model is from Wi-Fi positioning. Due to the nonlinearities of the system models, the UKF is employed for the integration. To further improve the INS/Wi-Fi integration, the enhancements using the vehicle constraints and AKF are proposed. The results of the experiment show that the INS/Wi-Fi integration provides a better tracking performance compared to the standalone Wi-Fi positioning, and the integration with the enhancements can further improve the tracking performance.

Chapter 4
Adapted Indoor Pedestrian Navigation Using PDR/Wi-Fi Integration

IMU based pedestrian dead reckoning (PDR) is widely studied in the personal navigation field [41][42]. Like other dead reckoning systems that use inertial sensors, it can achieve a high precision for short time but suffers from a propagating drift error because of IMU biases. Many researchers focus on foot-mounted systems [43][44]. Using a foot-mounted IMU, zero speed situations can be detected during the user's walking. In this case, zero velocity update (ZUPT) and zero angular rate update (ZARU) algorithms can be employed to reduce the heading estimation drift [45]. To design a PDR algorithm for portal devices which can be arbitrarily placed on the user's body, the step detection method needs to be chosen adaptively for different placement modes [46]. In this case, the placement mode classification is necessary for the PDR [47]. To further improve the indoor localization performance of the PDR, the dead reckoning system is integrated with Wi-Fi based positioning. In [48], a Wi-Fi assisted pedestrian dead reckoning navigation system is proposed and the experimental results are encouraging.

In this chapter, IMU based foot-mounted PDR is explored in Section 4.1. The methodologies of step detection, stride length and heading estimation are introduced. The application of ZUPT and ZARU are also described and the improvements are shown with pedestrian test results. In Section 4.2, an adapted PDR with portable devices is presented. The placement mode definition and classification, as well as the classification results are given. The integration of the adapted PDR and Wi-Fi positioning using a Kalman filter is described. In Section 4.3, a field experiment is carried out to show the performance of PDR/Wi-Fi integration compared to the standalone navigation systems.

4.1 IMU based foot-mounted pedestrian dead reckoning

The pedestrian dead reckoning (PDR) includes the solution of the navigation equations by estimating the user's position, velocity, and attitude (PVA) in a reference navigation frame from sensor measurements in the body frame of the user. Pedestrian dead reckoning methods explore the kinematics of human gait with the travelled distance and heading information [49]. Essentially, the pedestrian dead reckoning is the determination of a new position from the knowledge of a previous position utilizing current distance and heading information [43].

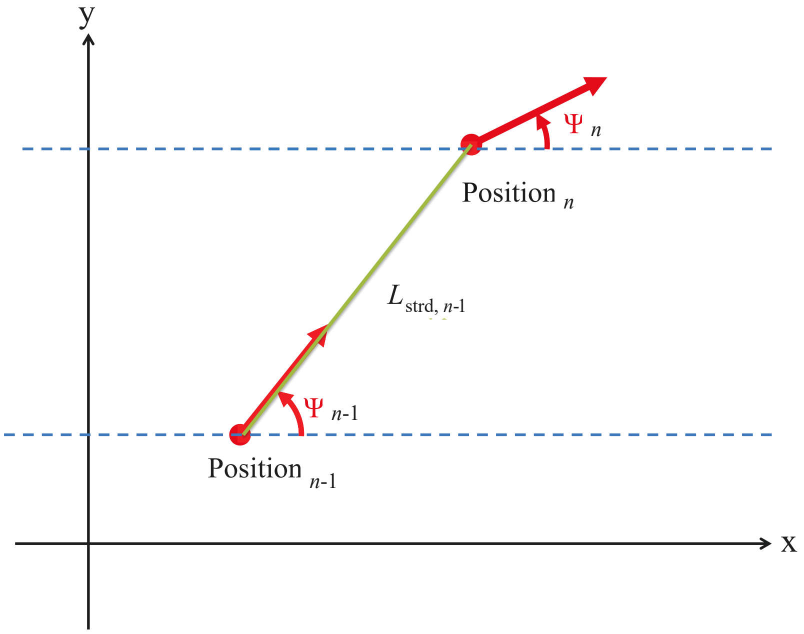


Figure 4.1: Pedestrian dead reckoning course

PDR consists of three important components: stride detection; stride length (L_{strd}) estimation of the distance traveled by the user since previous step $n-1$ and the user's heading (ψ) during the step $n-1$ to n. The coordinates ($r_{x,n}, r_{y,n}$) of a new position with respect to a position of the previous stride ($r_{x,n-1}, r_{y,n-1}$) can be computed as follows:

$$\begin{aligned} r_{x,n} &= r_{x,n-1} + L_{\text{strd},n-1}\cos\psi_{n-1} \\ r_{y,n} &= r_{y,n-1} + L_{\text{strd},n-1}\sin\psi_{n-1} \end{aligned} \tag{4.1}$$

4.1.1 Gait cycle and step detection

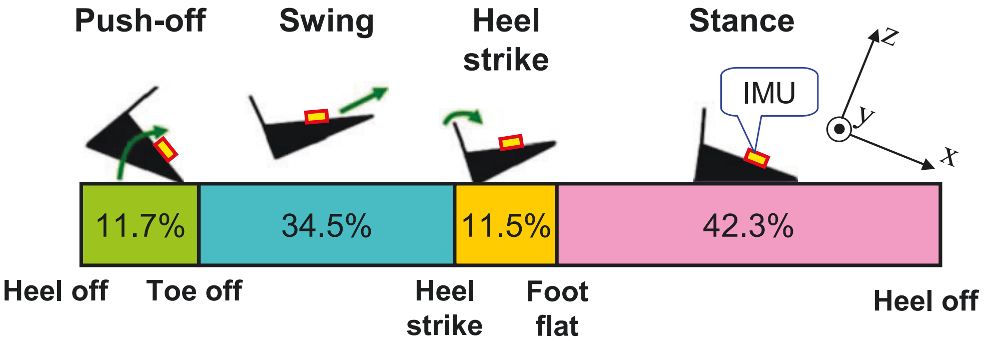


Figure 4.2: Gait cycle with foot-mounted IMU

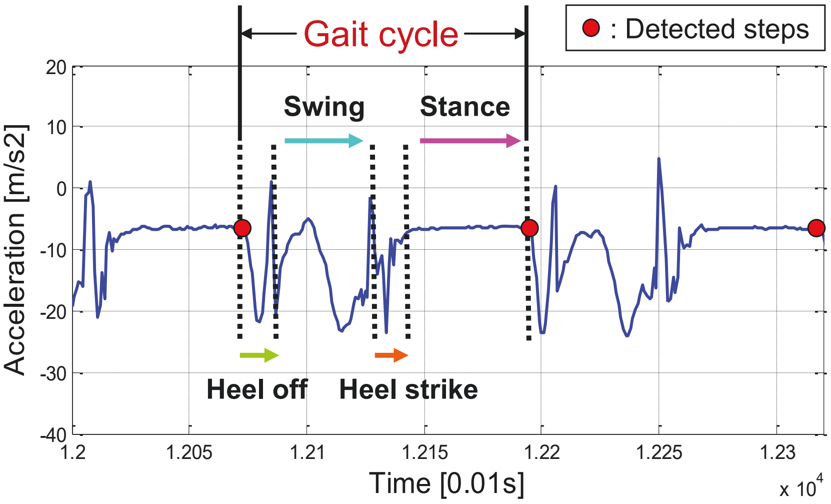


Figure 4.3: Acceleration measurement along x-axis of foot-mounted IMU

The step detection is based on pedestrian gait cycle [50]. A gait cycle includes four sequential phases, namely push-off, swing, heel strike and stance (shown in Figure 4.2) [46]. The percentage of each phase in a gait cycle is user dependent. As shown in Figure 4.2, the IMU is mounted on user's foot. The steps and gait phases can be detected from the IMU measurement using a sliding window and predefined thresholds. Figure 4.3 shows the detected steps and gait cycle phases based on the accelerometer output. The IMU used in this test is Xsens MTi IMU, which is employed in Section 3.3.1.

The stride length of pedestrian walking can be estimated from the detected frequency. A linear relationship is built to relate the step frequency to the stride length. The mathematical relationship is computed by curve fitting of various test subjects.

4.1.2 Stride length estimation

The stride length of pedestrian walking can be estimated from the detected stride frequency introduced in Section 4.1.1. In this work, a linear relationship is built to relate the frequency f to the stride length L_{strd} as:

$$L_{\text{strd},k} = a + b \cdot f_k \qquad \text{with} \quad f_k = \frac{1}{t_k - t_{k-1}} \tag{4.2}$$

where the k^{th} stride is detected at time t_k from the previous stride at t_{k-1}; a and b are the parameters of the linear mathematical relationship, which is computed by linear regression of various test subjects. In this work, four users take part in this walking test and each of them walks 3 times. As shown in Figure 4.4, the linear regression is done using the average values of stride length and stride frequency from different users in each test, so that the parameters can be determined.

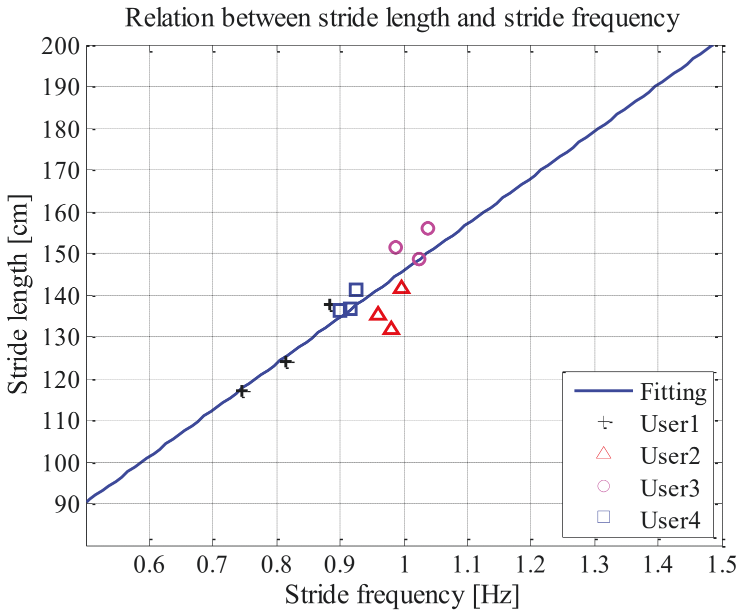


Figure 4.4: Linear regression of stride frequency vs. stride length

One test has been carried out and Figure 4.5 shows the error of stride length estimation, the values of error and accumulative error. The dots marked on the result curves represent the detected steps. It can be found that the accumulative error is about 1.1 m within 31 strides (about 45 m walking length) which is acceptable for pedestrian navigation.

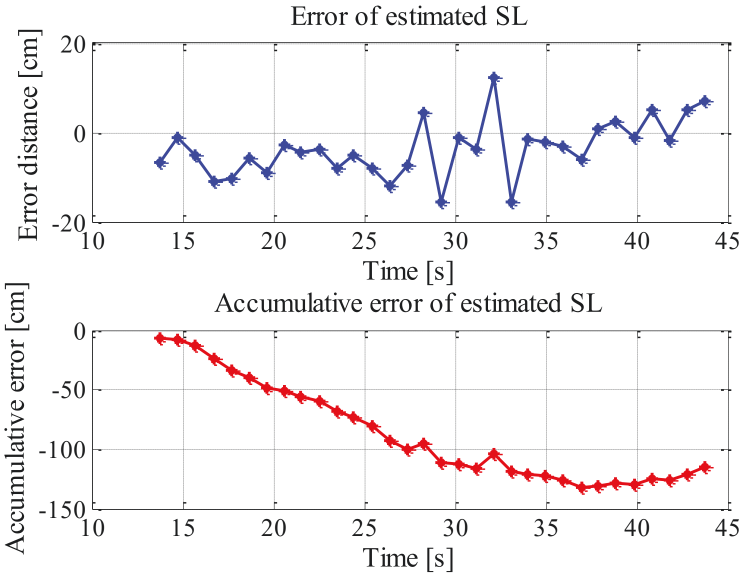


Figure 4.5: Average error using estimated stride length (31 strides)

4.1.3 Heading determination

The magnetometer is affected by magnetic disturbances, which limits the use of magnetometer in indoor navigation [51]. The ideal case is that the device is firmly tied on the human body and the orientation of the device is aligned with the body. The local frame is defined as follows: the positive directions of x-axis and y-axis are represented in Figure 4.3, the positive direction of z-axis indicates the inverse direction of gravity. In case of a local frame, the update of the heading information can be easily estimated as the integral of the gyro output in z-axis ω_z during the period between two adjacent steps, plus heading of the previous step φ_{i-1}, as shown in Equation (4.3).

$$\varphi_i = \int_{t_{i-1}}^{t_i} \omega_z(t)\,dt + \varphi_{i-1} \tag{4.3}$$

However, due to the variable placement of the device, the measured signals in the body frame are misaligned with the real attitude in the local frame. Thus, a rotation matrix is needed for the conversion of the sensor signals. Via the rotation matrix, the sensor outputs with an arbitrary orientation can be transformed into the local navigation frame. The discrete propagation equation of the attitude can be rewritten as:

$$\boldsymbol{\psi}_k = \boldsymbol{\psi}_{k-1} + \boldsymbol{\Phi}_{\mathrm{b},k-1}^{\mathrm{n}} \cdot (\tilde{\boldsymbol{\omega}}_{\mathrm{b},k-1} - \boldsymbol{\omega}_{\mathrm{b},k-1}^{\mathrm{bias}}) \cdot \Delta t + \mathbf{w}_{\boldsymbol{\psi},k-1}$$

with

$$\boldsymbol{\Phi}_{\mathrm{b},k}^{\mathrm{n}} = \begin{bmatrix} 1 & \mathrm{s}\theta\mathrm{t}\phi & \mathrm{c}\theta\mathrm{t}\phi \\ 0 & \mathrm{c}\theta & -\mathrm{s}\theta \\ 0 & \mathrm{s}\theta/\mathrm{c}\theta & \mathrm{c}\theta/\mathrm{c}\phi \end{bmatrix} \tag{4.4}$$

where $\mathrm{c}X = \cos X$, $\mathrm{s}X = \sin X$, $\mathrm{t}X = \tan X$; Δt is the sampling time; $\boldsymbol{\psi} = [\theta \quad \phi \quad \varphi]^T$ denotes the attitude (Euler angles) vector in navigation frame; θ, ϕ, φ represent the roll, pitch and yaw; $\boldsymbol{\Phi}_{\mathrm{b}}^{\mathrm{n}}$ is the rotation rate matrix between body frame and navigation frame; $\tilde{\boldsymbol{\omega}}_{\mathrm{b}}$ represents the measurement vector of angular rate from the gyroscopes; $\boldsymbol{\omega}_{\mathrm{b}}^{\mathrm{bias}}$ denotes the gyroscope bias error term; $\mathbf{w}_{\boldsymbol{\psi}}$ is the white noise of the propagation model.

4.1.4 Zero velocity update and zero angular rate update

For a foot-mount IMU, in the gait cycle (shown in Figure 4.2), the stance phase, where the velocity and angular rate of the IMU are approximately zero, can be detected from IMU output using predefined thresholds. In this case, zero velocity update (ZUPT) and zero angular rate update (ZARU) can be applied to estimate the IMU bias errors.

System modelling

For system modelling to estimate the IMU biases, the system state vector $\mathbf{x}$ is composed of velocity $\mathbf{v}$, attitude $\boldsymbol{\psi}$, accelerometer bias vector $\mathbf{f}_{\mathrm{b}}^{\mathrm{bias}}$ and gyro bias vector $\boldsymbol{\omega}_{\mathrm{b}}^{\mathrm{bias}}$. The simplified INS mechanization model is defined as the system process model, which is introduced in Section 3.1.1. The discrete process model is shown as:

$$\begin{aligned}
\mathbf{v}_k &= \mathbf{v}_{k-1} + \left[\mathbf{C}_{\mathrm{b},k-1}^{\mathrm{n}} \cdot (\tilde{\mathbf{f}}_{\mathrm{b},k-1} - \mathbf{f}_{\mathrm{b},k-1}^{\mathrm{bias}}) + \mathbf{g}\right] \cdot \Delta t + \mathbf{w}_{\mathbf{v},k-1} \\
\boldsymbol{\psi}_k &= \boldsymbol{\psi}_{k-1} + \boldsymbol{\Phi}_{\mathrm{b},k-1}^{\mathrm{n}} \cdot (\tilde{\boldsymbol{\omega}}_{\mathrm{b},k-1} - \boldsymbol{\omega}_{\mathrm{b},k-1}^{\mathrm{bias}}) \cdot \Delta t + \mathbf{w}_{\boldsymbol{\psi},k-1} \\
\mathbf{f}_{\mathrm{b},k}^{\mathrm{bias}} &= \mathbf{f}_{\mathrm{b},k-1}^{\mathrm{bias}} + \mathbf{w}_{\mathbf{f},\mathrm{bias},k-1} \\
\boldsymbol{\omega}_{\mathrm{b},k}^{\mathrm{bias}} &= \boldsymbol{\omega}_{\mathrm{b},k-1}^{\mathrm{bias}} + \mathbf{w}_{\boldsymbol{\omega},\mathrm{bias},k-1}
\end{aligned} \tag{4.5}$$

The bias terms $\mathbf{f}_{\mathrm{b}}^{\mathrm{bias}}$ and $\boldsymbol{\omega}_{\mathrm{b}}^{\mathrm{bias}}$ are modelled as random walk; $\mathbf{w}$ terms denote the process noises; $\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$ is the frame rotation matrix from the body frame to navigation frame.

$$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}} = \begin{bmatrix} \mathrm{c}\varphi\mathrm{c}\phi & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{s}\theta - \mathrm{s}\varphi\mathrm{c}\theta & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{c}\theta + \mathrm{s}\varphi\mathrm{s}\theta \\ \mathrm{s}\varphi\mathrm{c}\phi & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{s}\theta + \mathrm{c}\varphi\mathrm{c}\theta & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{c}\theta - \mathrm{c}\varphi\mathrm{s}\theta \\ -\mathrm{s}\phi & \mathrm{c}\phi\mathrm{s}\theta & \mathrm{c}\phi\mathrm{c}\theta \end{bmatrix} \tag{4.6}$$

$\Phi_{\mathrm{b}}^{\mathrm{n}}$ is the rotation rate matrix between body frame and navigation frame, and it can be expressed as:

$$\boldsymbol{\Phi}_{\mathrm{b},k}^{\mathrm{n}} = \begin{bmatrix} 1 & \mathrm{s}\theta\mathrm{t}\phi & \mathrm{c}\theta\mathrm{t}\phi \\ 0 & \mathrm{c}\theta & -\mathrm{s}\theta \\ 0 & \mathrm{s}\theta/\mathrm{c}\theta & \mathrm{c}\theta/\mathrm{c}\phi \end{bmatrix} \tag{4.7}$$

Making use of the fact that velocity and angular rate are approximately zero during the stance phase ($\mathbf{v} \cong \mathbf{0}$ and $\boldsymbol{\omega} \cong \mathbf{0}$). The system observation model can be derived as

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \boldsymbol{\eta}_k = \left[\mathbf{0}_{6\times 1}\right] \tag{4.8}$$

with

$$\mathbf{h}(\mathbf{x}_k) = \begin{bmatrix} \mathbf{v}_k \\ \dfrac{\boldsymbol{\psi}_k - \hat{\boldsymbol{\psi}}_{k-1}}{\Delta t} \end{bmatrix} \tag{4.9}$$

Because the process model is nonlinear, the extended Kalman filter (EKF) is employed here to estimate the system states.

Extended Kalman filtering

The Extended Kalman filter (EKF), is the most widely used nonlinear state estimation technique that has been applied in the past few decades. The principal feature of the EKF is linearization of the system equation and the measurement equation around the previous estimate or the current prediction. The linearized system is then represented by the Jacobian transformations of the nonlinear process and measurement functions. The normal KF formulas are then applied to the linearized system.

Suppose the system model is:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k, \boldsymbol{\eta}_k) \\ \mathbf{w}_k &\sim N\left(\mathbf{0}, \mathbf{Q}_k\right) \\ \boldsymbol{\eta}_k &\sim N\left(\mathbf{0}, \mathbf{R}_k\right) \end{aligned} \tag{4.10}$$

A Taylor series expansion of the state equation can be performed around $\mathbf{x}_{k-1} = \hat{\mathbf{x}}_{k-1}^{+}$ and $\mathbf{w}_{k-1} = \mathbf{0}$ to obtain the following:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\hat{\mathbf{x}}_{k-1}^{+}, \mathbf{u}_{k-1}, \mathbf{0}) + \left.\frac{\partial \mathbf{f}(\cdot)}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1}^{+}} \left(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^{+}\right) + \left.\frac{\partial \mathbf{f}(\cdot)}{\partial \mathbf{w}}\right|_{\hat{\mathbf{w}}_{k-1}^{+}} \mathbf{w}_{k-1} \\ &= \mathbf{f}(\hat{\mathbf{x}}_{k-1}^{+}, \mathbf{u}_{k-1}, \mathbf{0}) + \mathbf{F}_{k-1}\left(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}^{+}\right) + \mathbf{L}_{k-1}\mathbf{w}_{k-1} \\ &= \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \left[\mathbf{f}(\hat{\mathbf{x}}_{k-1}^{+}, \mathbf{u}_{k-1}, \mathbf{0}) - \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1}^{+}\right] + \mathbf{L}_{k-1}\mathbf{w}_{k-1} \\ &= \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \tilde{\mathbf{u}}_{k-1} + \tilde{\mathbf{w}}_{k-1} \end{aligned} \tag{4.11}$$

$\mathbf{F}_{k-1}$ and $\mathbf{L}_{k-1}$ are defined by the above equation. The known input signal $\tilde{\mathbf{u}}_{k-1}$ and the noise vector $\tilde{\mathbf{w}}_{k-1}$ are defined as follows:

$$\begin{aligned} \tilde{\mathbf{u}}_k &= \mathbf{f}(\hat{\mathbf{x}}_k^{+}, \mathbf{u}_k, \mathbf{0}) - \mathbf{F}_k\hat{\mathbf{x}}_k^{+} \\ \tilde{\mathbf{w}}_{k-1} &\sim N\left(\mathbf{0}, \mathbf{L}_k\mathbf{Q}_k\mathbf{L}_k^T\right) \end{aligned} \tag{4.12}$$

The measurement equation can be linearized around $\mathbf{x}_k = \hat{\mathbf{x}}_k^{-}$ and $\boldsymbol{\eta}_{k-1} = \mathbf{0}$ to obtain:

$$\begin{aligned}
\mathbf{y}_k &= \mathbf{h}(\hat{\mathbf{x}}_k^-,\mathbf{0}) + \left.\frac{\partial \mathbf{h}(\cdot)}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_k^-} \left(\mathbf{x}_k - \hat{\mathbf{x}}_k^-\right) + \left.\frac{\partial \mathbf{h}(\cdot)}{\partial \boldsymbol{\eta}}\right|_{\hat{\boldsymbol{\eta}}_k^-} \boldsymbol{\eta}_k \\
&= \mathbf{h}(\hat{\mathbf{x}}_k^-,\mathbf{0}) + \mathbf{H}_k\left(\mathbf{x}_k - \hat{\mathbf{x}}_k^-\right) + \mathbf{M}_k\boldsymbol{\eta}_k \\
&= \mathbf{H}_k\mathbf{x}_k + \left[\mathbf{h}(\hat{\mathbf{x}}_k^-,\mathbf{0}) - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right] + \mathbf{M}_k\boldsymbol{\eta}_k \\
&= \mathbf{H}_k\mathbf{x}_k + \mathbf{z}_k + \tilde{\boldsymbol{\eta}}_k
\end{aligned} \tag{4.13}$$

$\mathbf{H}_k$ and $\mathbf{M}_k$ are defined by the above equation. The vector $\mathbf{z}_k$ and the noise vector $\tilde{\mathbf{e}}_k$ are defined as

$$\begin{aligned}
\mathbf{z}_k &= \mathbf{h}(\hat{\mathbf{x}}_k^-,\mathbf{0}) - \mathbf{H}_k\hat{\mathbf{x}}_k^- \\
\tilde{\boldsymbol{\eta}}_k &\sim N\left(\mathbf{0},\mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T\right)
\end{aligned} \tag{4.14}$$

Then we obtain a linear process model in Equation (4.12) and a linear measurement model in Equation (4.14), which means that the state can be estimated with the standard KF equations as follows:

1. Time update equation

$$\begin{aligned}
\mathbf{P}_k^{\text{-}} &= \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{L}_{k-1}\mathbf{Q}_{k-1}^+\mathbf{L}_{k-1}^T \\
\hat{\mathbf{x}}_k^- &= \mathbf{f}(\hat{\mathbf{x}}_{k-1}^+,\mathbf{u}_{k-1,}\mathbf{0})
\end{aligned} \tag{4.15}$$

2. Measurement update equation

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{P}_k^-\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T\right)^{-1} \\
\mathbf{z}_k &= \mathbf{h}(\hat{\mathbf{x}}_k^-,\mathbf{0}) - \mathbf{H}_k\hat{\mathbf{x}}_k^- \\
\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^- - \mathbf{z}_k\right) \\
&= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left[\mathbf{y}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-,\mathbf{0})\right] \\
\mathbf{P}_k^+ &= \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-
\end{aligned} \tag{4.16}$$

Test results

To show the performance of ZUPT and ZARU, one field test is carried out using a MEMS based IMU (Xsens MTi) mounted on the user's foot. Figure 4.5 shows the estimated IMU bias errors during one detected stance phase which lasts about 0.29 s.

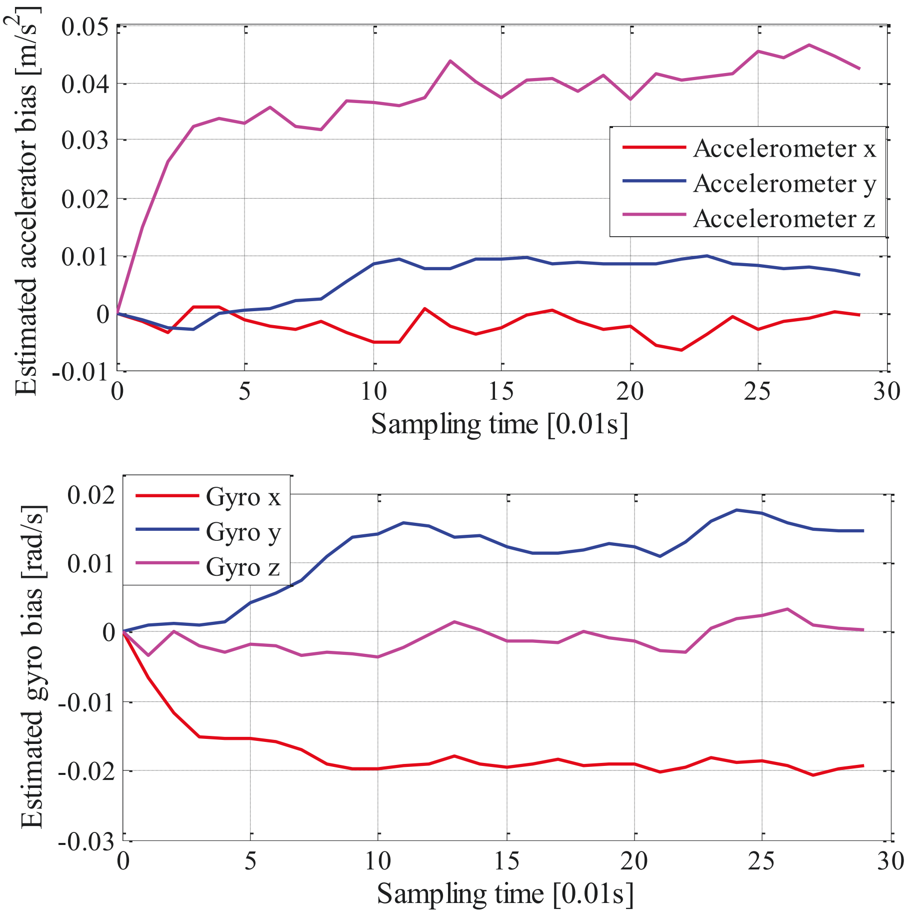


Figure 4.6: Estimated IMU bias errors

For every gait cycle, the biases are re-estimated during the stance phase using ZUPT and ZARU. The estimated accelerometer bias errors are not used in the pedestrian dead reckoning calculation. The estimated gyro biases are used in attitude determination via the following equation:

$$\boldsymbol{\psi}_k = \boldsymbol{\psi}_{k-1} + \boldsymbol{\Phi}^{\mathrm{n}}_{\mathrm{b},k-1} \cdot (\tilde{\boldsymbol{\omega}}_{\mathrm{b},k-1} - \boldsymbol{\omega}^{\mathrm{bias}}_{\mathrm{b},k-1}) \cdot \Delta t \qquad (4.17)$$

Combined with the stride detection introduced in Section 4.1.1 and the stride length estimation introduced in Section 4.1.2, the user's trajectory can be estimated using Equation (4.1).

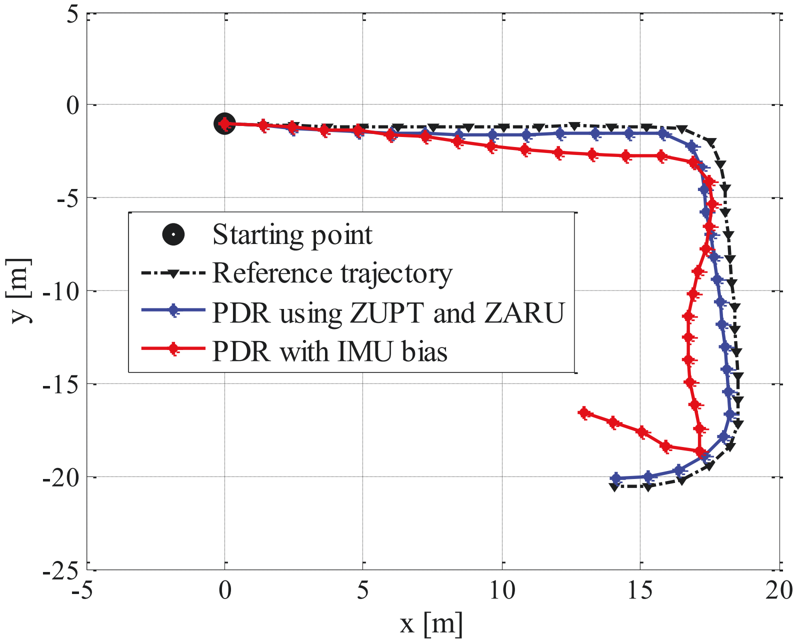


Figure 4.7: Trajectory estimation results

Figure 4.7 shows the trajectory estimation results. Through estimating and subtracting the gyro bias errors, the PDR using ZUPT and ZARU can provide a much better heading estimation than the PDR using the IMU raw data with the bias errors.

4.2 Adapted pedestrian dead reckoning with portable devices

4.2.1 Device placement mode definition and features for mode classification

Unlike the foot-mounted systems, personal portable devices (such as smartphones and tablet PCs) can be either mounted on the user's body or carried by the user. The sensor unit, especially the IMU embedded in the device, may respond to additional dynamics due to the movement of the user's body [52][53]. In that case, the navigation output of the IMU depends on its placement, its orientation relative to the subject, and the subject's posture and activity.

One solution is to identify the device placement information using intelligent context awareness so that the system can adapt proper pedestrian dead reckoning algorithm. From the user's experience of commonly used portable digital device, three typical placement modes are defined as: (1) device mounted on body, such as placed in the pocket: body mounted mode (pocket mode); (2) device dangling in hand during walking: dangling mode; (3) device held in front of the body when the user is reading or messaging: reading mode. Figure 4.8 illustrates the application cases for every placement mode. The mode definition in this work is not exhaustively all-inclusive for practical applications, but representative to study the impact of typical sensors placement from a navigation perspective.

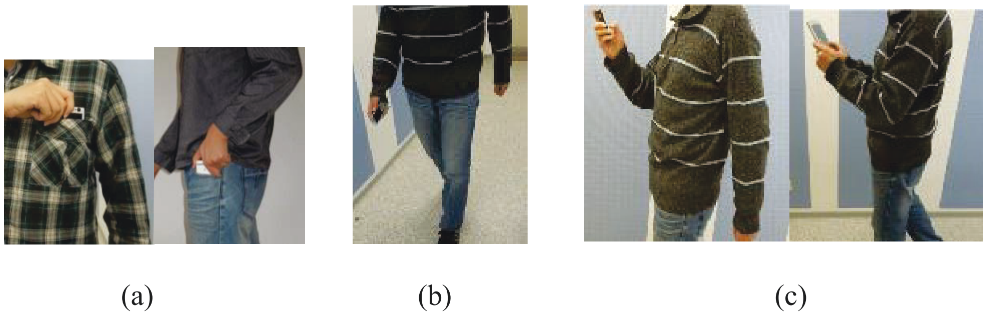

(a) (b) (c)

Figure 4.8: Placement cases: (a) body mounted mode (pocket mode); (b) dangling mode; (c) reading mode

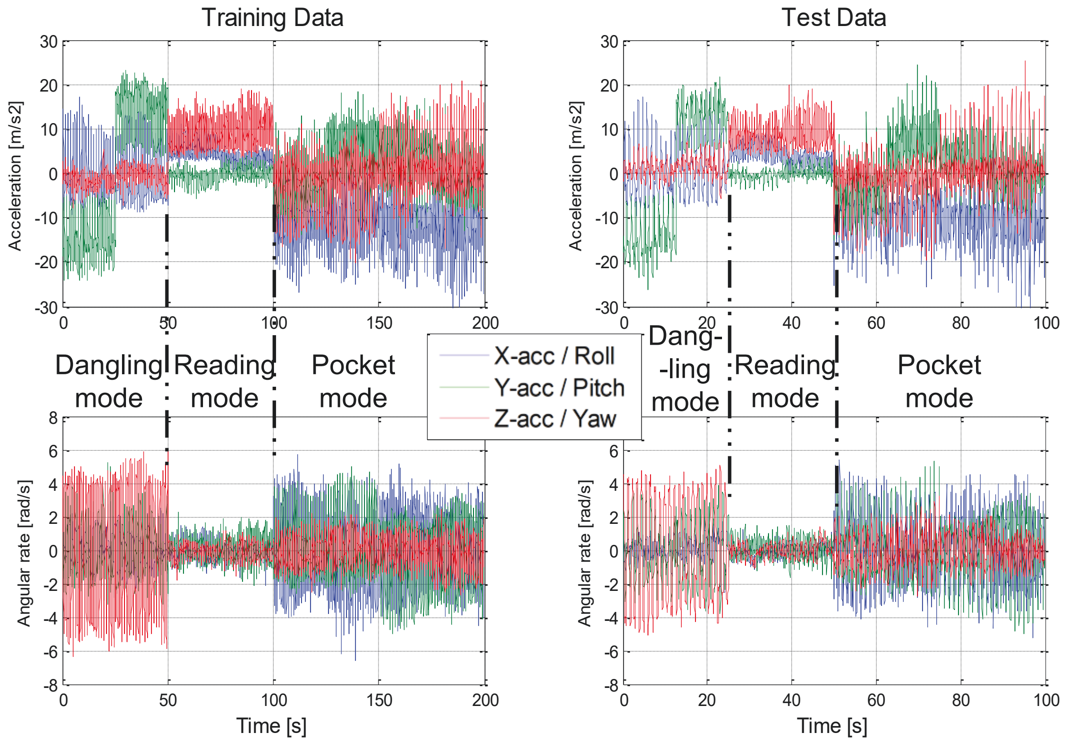


Figure 4.9: IMU measurements (training and test data from the same user) in different placement modes

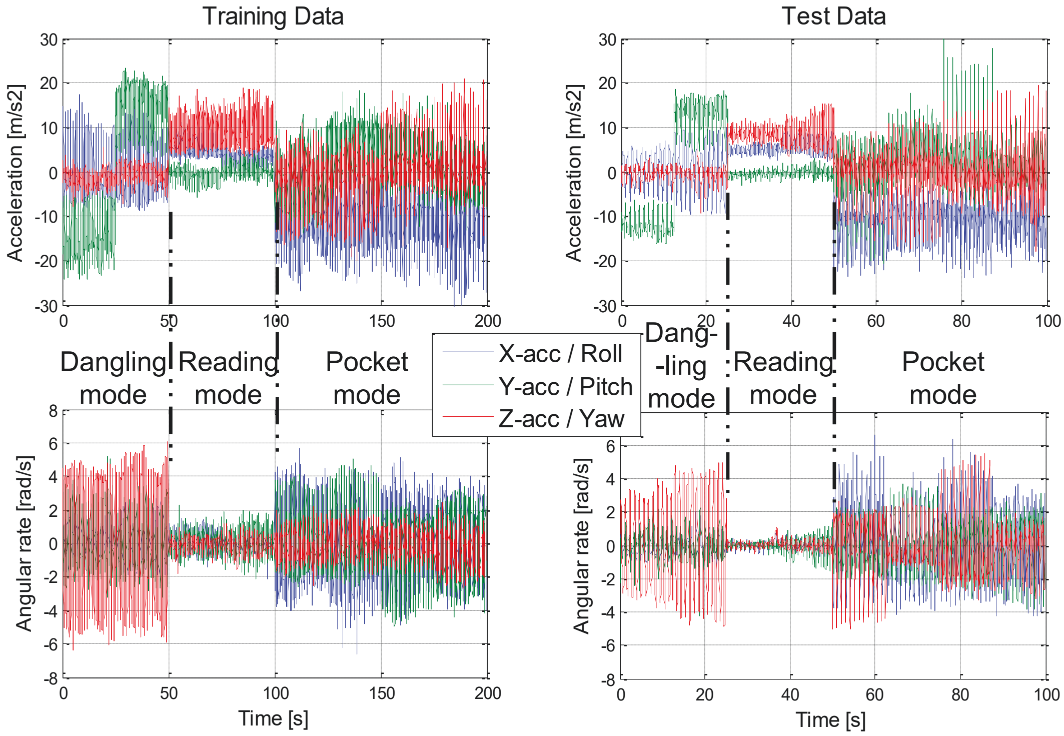


Figure 4.10: IMU measurements (training and test data from different users) in different placement modes

Mode classification, which is also called as pattern recognition, is based on the output data of the IMU. The measurements from gyroscopes and accelerometers in 3-axis are chosen as the features for the classification. Figure 4.9 and Figure 4.10 show the training and test data collected in same user case and in different user case respectively. The same training data is illustrated in both figures. The IMU employed here is Xsens MTi, which is used in previous sections. The output rate is 100 Hz. It can be found from the training data that the phase of the dangling mode is from 0 s to 50 s, the reading mode phase is from 50 s to 100s and the pocket mode phase is from 100 s to 200 s.

In this work, the feature measurements are pre-processed before being used as inputs for the pattern classification. The purpose of the data preprocessing is to clean selected data for better quality. In general, data cleaning means to filter, aggregate, and fill in missing values. By filtering data, the selected data are examined for outliers and redundancies. Outliers differ greatly from the majority of data, or data that are clearly out of range of the selected data groups. Redundant data are the same information recorded in several different ways. By aggregating data, data dimensions are reduced to obtain aggregated information. By smoothing data, missing values of the selected data are found and new or reasonable values are then added [54]. Figure 4.11 shows the performance of the smoothing with moving-average algorithm (time-based window size: 0.2 s).

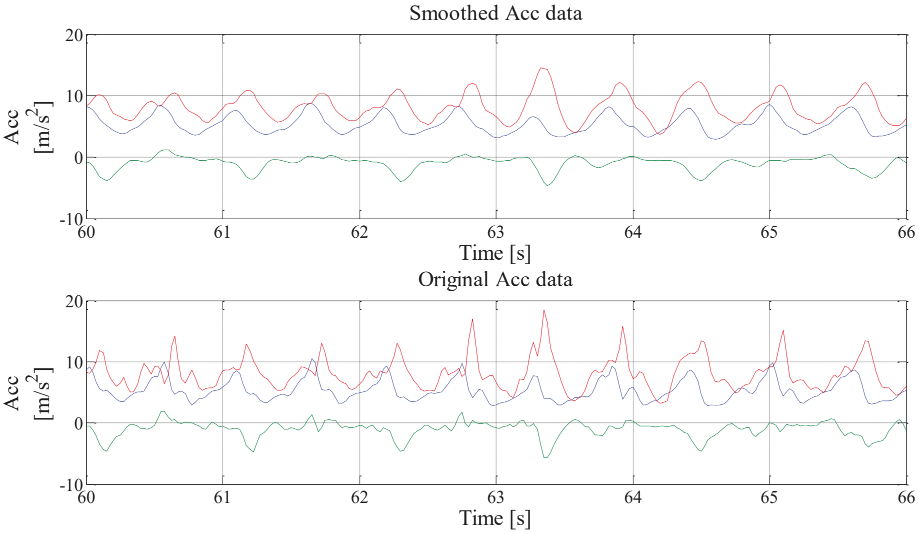


Figure 4.11: Comparison between smoothed data and original data

4.2.2 Classification algorithms

With the defined modes and features, the classification can be carried out with a classifier. In most research papers, machine learning algorithms are used as classifiers for pattern recognition. They can be generally divided into two types: supervised learning and unsupervised learning. The distinction between these two types is reflected in whether the training samples have given labels or not. If not, the algorithm needs to discover the hiding rules among the data patterns by itself. In this case, it is not called a classification algorithm any more, but a clustering algorithm.

In this work, three supervised learning algorithms are discussed: k-nearest neighbor (KNN), artificial neural network (ANN) and support vector machine (SVM).

K-Nearest Neighbor

K-nearest-neighbor classification is one of the most fundamental and simple classification methods and should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution of the data [55]. KNN classification is developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine [56].

KNN is a non-parametric learning algorithm, which means that it does not make any assumptions on the underlying data distribution (e.g., Gaussian mixtures, linearly separable). It is also a lazy algorithm. It does not use training data points for generalization. There is no explicit training phase or it is very minimal. The training phase is fast. The training data is needed during the testing phase. There is a nonexistent or minimal training phase but a costly testing phase (cost in terms of time and memory) [57]. In the worst case, all data points might take part in decision, more time is needed. As all training data needs to be stored, the required memory is relatively large.

It is assumed by KNN that data is in a feature space. Data points are in a metric space. Data can be scalar data or multidimensional vector data. Training data consists of vectors and class label associated with each vector. It will be either positive or negative classes. KNN works equally well with an arbitrary number of classes.

The given number "*k*" decides how many neighbors (where the neighbor is defined based on the similarity) influence the classification. This is usually an odd number if the number of classes is 2. If *k*=1, then the algorithm is simply called the nearest neighbor algorithm. The KNN algorithm is sensitive to the local structure of the data. The best choice of *k* depends upon the data; generally, larger values of *k* reduce the effect of noise on the classification but make boundaries between classes less distinct.

The most popular way to evaluate a similarity measure is the use of distance measures. And the widely used distance measure is the Euclidean distance, defined as

$$d(\mathbf{z}_u,\mathbf{z}_w)=\sqrt{\sum_{j=1}^{N_d}(\mathbf{z}_{u,j}-\mathbf{z}_{w,j})^2} \tag{4.18}$$

where $\mathbf{z}_u$ and $\mathbf{z}_w$ are two instances in feature space.

The purpose of learning is to know how to discern the samples whose labels are not known. Now a new sample is collected and used for the judge. The KNN classifier calculated the distances between the new one and each of the known samples, which is the worst case. Through kd-tree algorithm, the data space can be partitioned and then the corresponding data-indices are constructed. In this way, the exhaustive search can be avoided. Thus actually within KNN algorithm, there is no real training cause, because each time a new sample is collected a computation based on almost all of the training data would be processed.

Artificial Neural Networks

As a mathematical model derived from biological neural networks, an ANN consists of simple and highly interconnected processing elements, the neurons. Each of them receives one or more inputs and sums them to produce an output, which works in a similar way to biological neurons [58]. In most cases a neural network is an adaptive system that adjusts its structure continually during the training phase, thus it is very efficient to use ANN model complex relationships between inputs and outputs or to find patterns in data.

There are various sorts of ANN algorithms such as back-propagation neural network and Hopfield network. The one employed in this work is the probabilistic neural network (PNN). PNN is a kind of feed-forward neural networks. It is predominantly a classifier that maps input patterns to a quantity of classifications [60]. Yet it can be also forced into a more general function approximator. A PNN is an implementation of a statistical algorithm called kernel discriminant analysis in which the operations are organized into a multilayered network with four layers: (a) input layer; (b) pattern layer; (c) summation layer; (d) output layer.

When an input is presented, the pattern layer produces a vector whose elements indicate how close the input is to the vectors of the training set. The summation layer calculates the average of the output of the probability density function (PDF) for all samples in each single population [61]. Finally, the competitive transfer function produces a "1" corresponding to the largest element of the input vector, and a "0" elsewhere. Thus, the network classifies the input vector into a specific *i* class because that class has the maximum probability of being correct.

The detailed description of the ANN algorithm is given in Appendix A.

Support Vector Machine

SVM is a binary classifier that derived from statistical learning theory and kernel based methods. An SVM model is a representation of the examples as points mapped in space so that the examples of the separate categories are divided by a clear gap that is as wide as possible. In the center of the gap lies the boundary of different categories: hyperplane [63]. New examples are then mapped into that same space and predicted to belong to a category based on which side of the hyperplane they fall on [62].

The SVM method finds the optimum separating hyperplane so that the samples with different labels are located on each of the planes. The distance of the closest samples to the hyperplane in each side is maximized. These samples are called support vectors. The optimal hyperplane is located where the margin between two classes of interest is maximized so that the error is minimized.

For a multiclass classification, multiple binary classifiers are required. Each classifier is trained to discriminate one class from the remaining classes. During the testing or application phase, data are classified by computing the margin from the linear separating hyperplane. Data are assigned to the class labels of the SVM classifiers that produce the maximal output. In this work, the SVM with the Gaussian kernel is employed.

In Appendix B, the detailed introduction and derivation of the SVM classifier are presented.

4.2.3 Placement mode classification results

In this section, classification results obtained respectively using KNN, PNN and SVM are shown and compared. The example of training data and test data sets collected from the same user are shown in Figure 4.9 and the one from different users are shown in Figure 4.10.

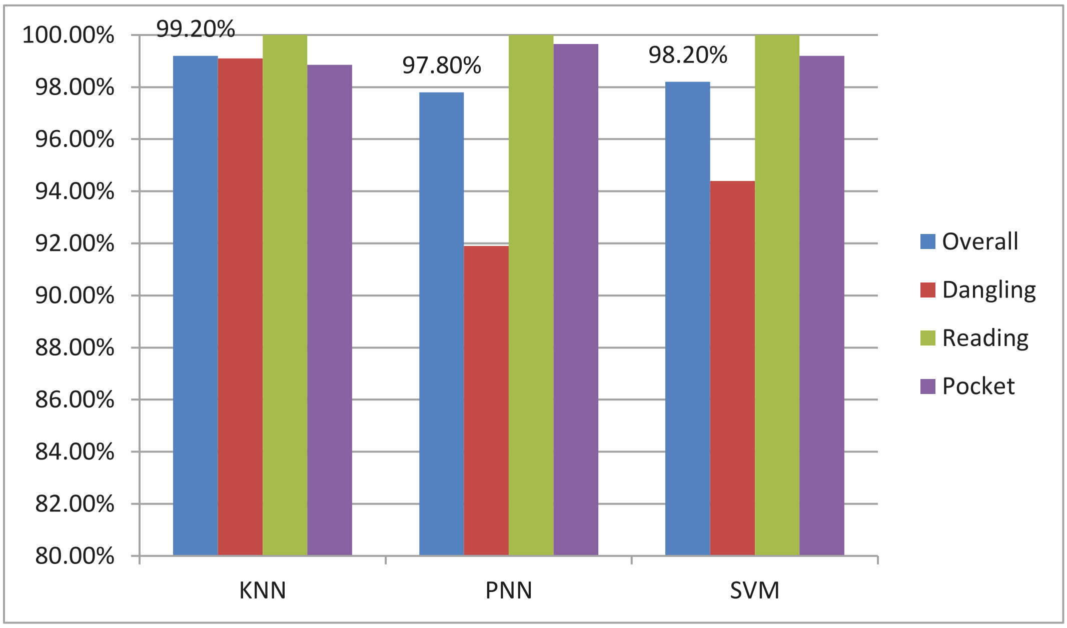


Figure 4.12: Classification accuracy (same user case)

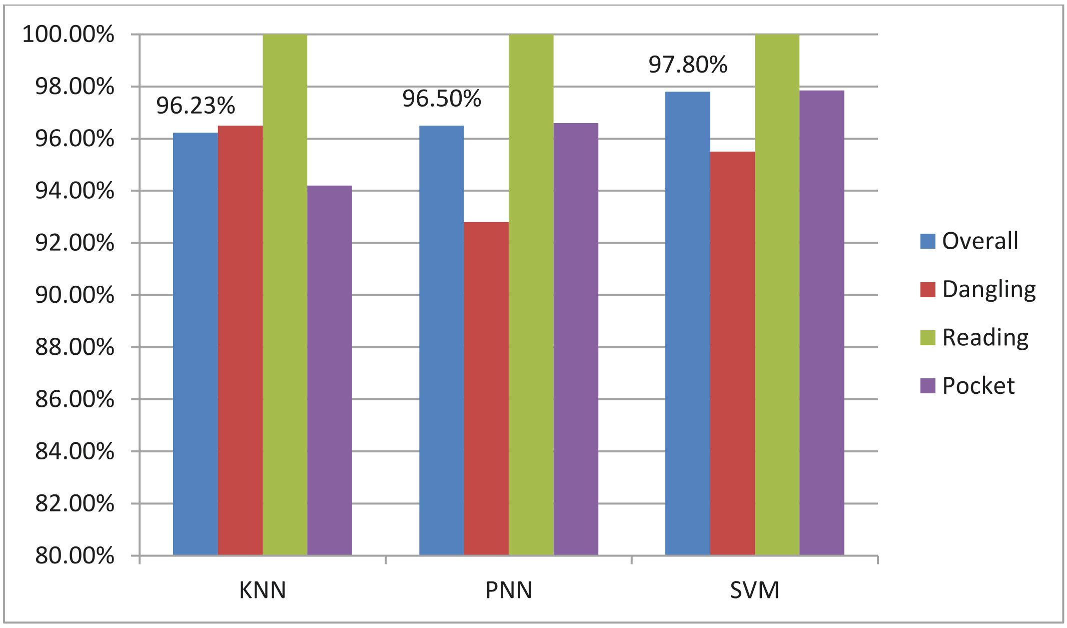


Figure 4.13: Classification accuracy (different user case)

Figure 4.12 and Figure 4.13 show the classification results using the three classifiers in same user case and different user case respectively. It can be found that KNN fits the patterns best (99.2%) in same user case. But if it is tested by a different user, KNN becomes the one who produces the worst result (96.23%), shown in Figure 4.13. Reviewing the performance of SVM, though the result is not such good as KNN's in the first case, SVM yields the best result in the second case. Considering the different user case is more important in practical applications and SVM also provides the slightly better overall performance in both cases, SVM is chosen as the classifier for user's placement mode recognition.

Table 4.1: Confusion matrix of SVM (same user case)

	Dangling	Reading	Pocket
Dangling	94.4%	0%	5.6%
Reading	0%	100%	0%
Pocket	0.8%	0%	99.2%

Table 4.2: Confusion matrix of SVM (different user case)

	Dangling	Reading	Pocket
Dangling	95.5%	0%	4.5%
Reading	0%	100%	0%
Pocket	2.15%	0%	97.85%

The misclassification of SVM can be better observed from the confusion matrix shown in Table 4.1 and Table 4.2. The overall performance degrades a little bit from same user case to different user case (overall accuracy is dropped from 98.2% to 97.8%), in which the demotion of performance for pocket mode is dominant (1.35% lower). More especially: reading mode is 100% correctly classified for both situations while dangling mode and pocket mode are misclassified occasionally as each other. Nevertheless, the three modes can be distinguished with a high accuracy (all above 90%) whether it is tested by the same user or not.

4.2.4 Adapted PDR based on classified placement mode

Same as food-mounted systems introduced in Section 4.1, the adapted PDR with the portable device consists of stride detection, stride length estimation and heading determination.

For stride detection, the employed sensor data and detection method depend on the classified placement mode, which is shown in Table 4.3. Based on the detected stride frequency, the stride length estimation in different placement modes uses the same algorithm afore-introduced in Section 4.1.2.

Table 4.3: Sensor data and detection method for different placement modes

	Dangling mode	Reading mode	Pocket mode
Sensor data	Gyroscope z-axis	Accelerometer z-axis	Gyroscope y-axis
Detection method	Zero crossing detection	Peak detection	Peak detection

The heading determination method applied for the adapted PDR is the one described in Section 4.1.3. The IMU needs to be re-aligned with the user's body when the placement mode changes. Unlike the foot mounted system, ZUPT and ZARU are not available with the portable device when the user is walking, which means that the gyro bias errors cannot be estimated and compensated during the pedestrian movement. To obtain a heading determination without high drift errors, the gyro bias vector is pre-estimated by averaging the gyro triad measurements when the IMU is stationary.

4.2.5 Integration of adapted PDR and Wi-Fi based positioning

Considering the complementary nature of dead reckoning and Wi-Fi positioning, the combination of both systems is expected to yield a synergetic effect resulting in higher performance. The integration structure is shown in Figure 4.14.

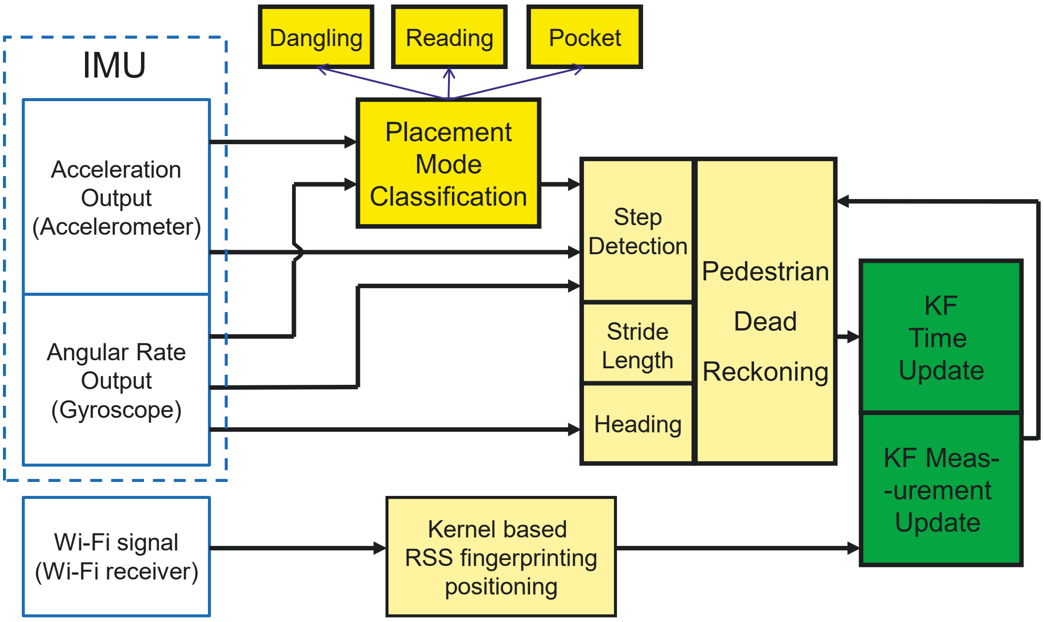

Dangling
Reading
Pocket
IMU
Acceleration Output (Accelerometer)
Angular Rate Output (Gyroscope)
Placement Mode Classification
Step Detection
Stride Length
Heading
Pedestrian Dead Reckoning
KF Time Update
KF Meas- -urement Update
Wi-Fi signal (Wi-Fi receiver)
Kernel based RSS fingerprinting positioning

Figure 4.14: Integration of adapted PDR and Wi-Fi position

After placement mode classification, the PDR provides system propagation mode for state prediction. The system state is the user's position vector $\begin{bmatrix} r_x & r_y \end{bmatrix}^T$. The propagation mode is shown as following equation:

$$\begin{bmatrix} r_{x,n} \\ r_{y,n} \end{bmatrix} = \begin{bmatrix} r_{x,n-1} \\ r_{y,n-1} \end{bmatrix} + \begin{bmatrix} \hat{L}_{\mathrm{strd},n-1} \cos \hat{\psi}_{n-1} \\ \hat{L}_{\mathrm{strd},n-1} \sin \hat{\psi}_{n-1} \end{bmatrix} + \mathbf{w}_{\mathbf{r},n-1} \tag{4.19}$$

where $\hat{L}_{\mathrm{strd}}$ and $\hat{\psi}$ denote the estimated stride length and user's heading respectively; $\mathbf{w}_{\mathbf{r}}$ is the system process noise term.

The system measurement update input is provided by Wi-Fi positioning. The positioning algorithm employed here is Kernel based fingerprinting, which is presented in Section 2.3.2. The Kalman filter is applied for the integration, which is described with the following procedures [32].

A discrete system model with additional Gaussian noises is formulated as:

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \\
\mathbf{y}_k &= \mathbf{H}_k\mathbf{x}_k + \boldsymbol{\eta}_k \\
\mathbf{w}_k &\sim N\left(\mathbf{0},\mathbf{Q}_k\right) \\
\boldsymbol{\eta}_k &\sim N\left(\mathbf{0},\mathbf{R}_k\right)
\end{aligned} \tag{4.20}$$

The time update equations for the state **x** and the covariance of the estimation error **P** are:

$$\begin{aligned}
\hat{\mathbf{x}}_k^- &= \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1}^+ \\
\mathbf{P}_k^- &= \mathbf{F}_{k-1}\mathbf{P}_{k-1}^+\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}
\end{aligned} \tag{4.21}$$

The measurement update process can be described by the following equations:

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{P}_k^-\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1} \\
\hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_k^-\right) \\
\mathbf{P}_k^+ &= \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-\left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)^T + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^T \\
&= \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_k^-
\end{aligned} \tag{4.22}$$

where **K** is the Kalman filter gain matrix.

4.3 Field experiment and results

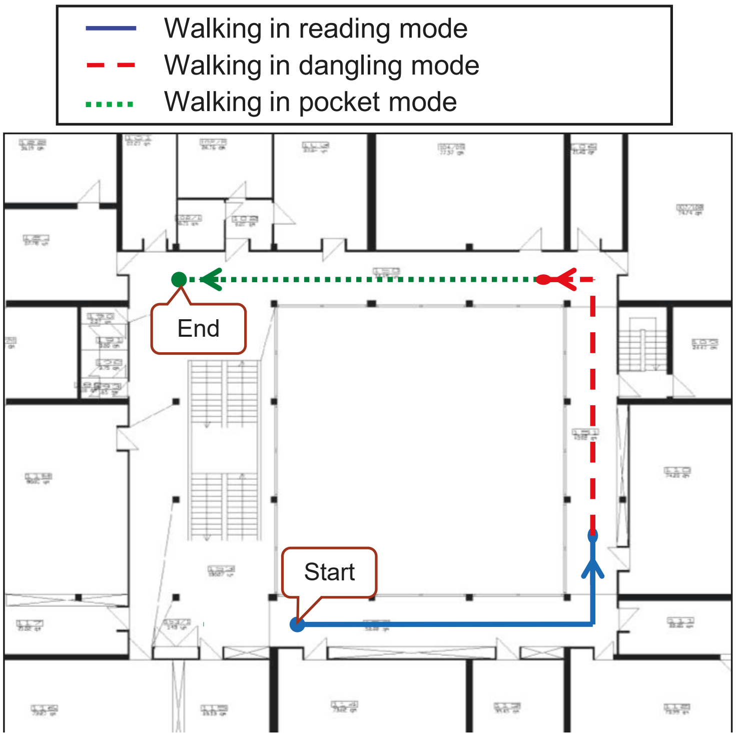


Figure 4.15: Pedestrian trajectory in different placement modes

To show the performance of the adapted PDR and PDR/Wi-Fi integrated system, one field experiment has been carried out at the same test site as the one introduced in Section 2.4.1 and 3.3.1. Sensors used in the experiment include the TP-LINK TL-WN722N Wi-Fi receiver and Xsens MTi MEMS IMU. The fingerprinting database is built of 182 fingerprinting points with a separation distance of 1 meter. As shown in Figure 4.15, the user with the sensors is walking along the trajectory firstly in reading mode then in dangling mode and finally in pocket mode. The pedestrian trajectory consists of 42 stride cycles.

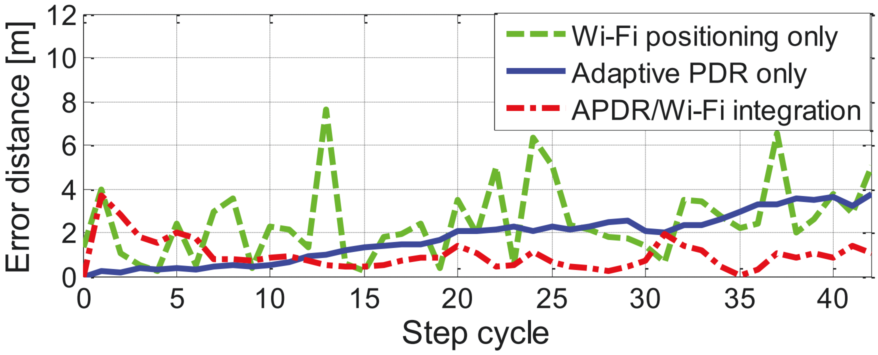


Figure 4.16: Positioning errors comparison

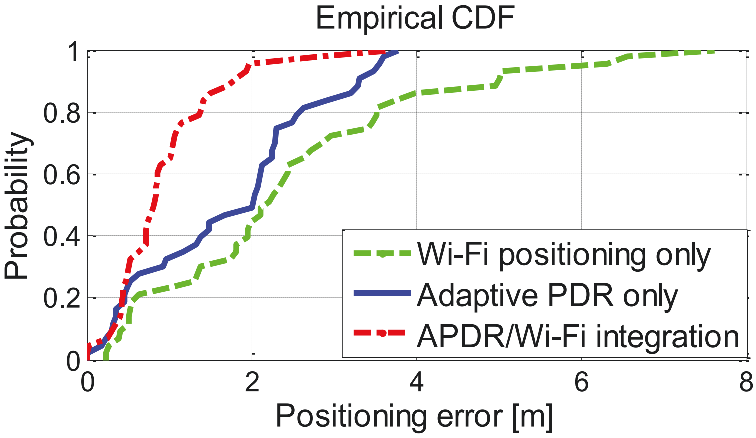


Figure 4.17: Error CDFs

Table 4.4: Means and standard deviations of positioning errors

Method / Error [m]	Wi-Fi	APDR	APDR/Wi-Fi
Mean	2.49	1.75	1.01
Standard deviation	1.78	1.11	0.75

The experimental results are shown in Figure 4.16 as position errors using three tracking algorithms. It is noticeable that the positioning result using the standalone adapted PDR is fine at the beginning, but the deviations are accumulated which leads to a drift error at the end (although the gyro outputs are pre-calibrated by subtracting the pre-estimated gyro biases). In contrast, the result using the APDR/Wi-Fi integration shows relatively stable performance. To show a better numerical comparison, the error cumulative distribution functions (CDFs) and the error properties are presented in Figure 4.17 and Table 4.4 respectively. It can be found that the integrated algorithm can provide an improved tracking performance from the standalone systems.

4.4 Summary

In this chapter, the IMU based PDR is discussed. In the stance phase of the gait cycle, zero velocity update and zero angular rate update with EKF are applied to estimate the IMU biases, and hence reduce the drift error of the dead reckoning system.

To design a PDR algorithm for portal devices which can be arbitrarily placed on the user’s body, the step detection method needs to be chosen adaptively according to different sensor placement modes. Typical placement modes were defined and classified based on measurement outputs of accelerometers and gyroscopes. Three classifiers, namely nearest neighbor, artificial neural networks and support vector machine have been discussed, and the corresponding classification results have been compared and analyzed. The adapted PDR is further combined with Wi-Fi based positioning with a Kalman filter. The result of the experiment shows that the PDR/Wi-Fi integration outperforms the standalone systems

Chapter 5
Indoor Attitude Estimation Using INS/Visual-Gyroscope Integration

Precise attitude estimation is a challenging topic for indoor navigation. As mentioned in the previous chapter, the attitude determination using standalone IMU suffers from drift error propagation because of gyro biases. The magnetometer is affected by magnetic disturbances, which limit the use of magnetometer for attitude determination in indoor applications. In this case, the camera-based direction assistance was proposed to improve the heading estimation.

Many researchers focus on the systems with a priori formed database containing images of recognizable features in the surroundings attached with position and attitude information. However, the database based procedure is restricted to predefined and hence known areas [65][66]. Moreover, the limited storage space of low-cost devices can be a challenge to the navigation system with a large image database. In contrast, the methods directly calculating the motion of the camera from consecutive images yield a more suitable solution for this work. As one of these methods, the visual gyroscope (visual-gyro) technique has been introduced in [67] and [68]. In [69] and [70], the visual-gyro is integrated with the inertial navigation system (INS) in a multi-sensor fused navigation system using Euler angle based models. However, there are still challenges to the existing INS/visual-gyro integrations: 1) how to deal with singularity problems when using the conventional Euler angle based model; 2) how to improve estimation performance and reduce the computational effort when using the nonlinear models based on Euler angle or quaternion.

In response to the challenges, in this work, an INS/visual-gyro integration using direction cosine matrix (DCM) based models is presented. The INS/visual-gyro integration using the DCM attitude representation avoids the singularity problem that occurs with Euler angle representations. Furthermore, a linear (or pseudo-linear if augmenting the gyro bias) process model for the DCM update with respect to its parameters is formulated. Moreover, a linear observation model is derived for the visual-gyro triad with respect to the DCM elements.

In this chapter, visual gyroscope using vanishing points is introduced in Section 5.1. The vanishing point finding approach and the camera attitude determination method are given. Section 5.2 presents DCM based INS/visual-gyro integration. The derivations of system process and observation models are provided. In Section 5.3, field experiments are described and the numerical results are presented.

5.1 Projective geometry and vanishing point detection

The visual gyroscope technique makes use of computer vision algorithms to transform information found from images into the camera rotation. Unlike rate gyroscope in a MEMS based IMU, the visual gyroscope does not suffer from drift errors. Therefore, in this work, it is used to calibrate the IMU gyro output by estimating the gyro bias through INS/visual-gyro integration. The visual gyroscope employed here is based on tracking vanishing points in consecutive images.

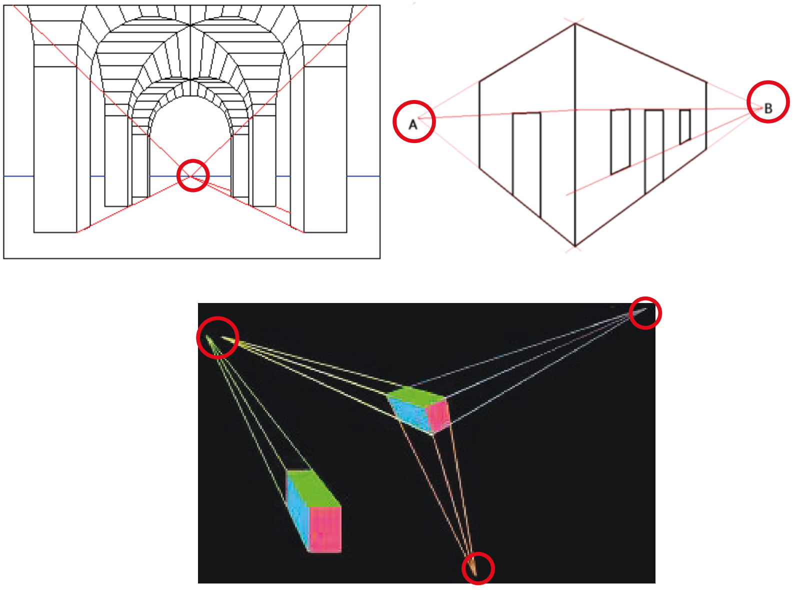


Figure 5.1: Vanishing points in 2D images

The vanishing points are the points in an image where the lines parallel in the real world seem to intersect, which are shown with red cycles in **Figure 5.1**. The detected motion of the vanishing points can be converted into camera rotation. There are normally three useful vanishing points in an image which are central, vertical and horizontal vanishing points. They are intersects of the parallel lines along 3 orthogonal axes. This algorithm is suitable for indoor environments since the human-made constructions are mainly composed of orthogonal structures and the parallel lines, as well as the vanishing points, can be detected.

5.1.1 Projective geometry

The vanishing point can be described mathematically in different ways, on the one hand, with 2D image entities, on the other hand, corresponding to 3D space directions. For this reason, the key issue is how to transform the coordinates of a point expressed in the 3D camera frame into the 2D image frame.

The image plane information is mapped into the sphere by means of the projective transformation given by the calibration matrix of the camera. The image plane information is to be calibrated and normalized, i.e. the information is mapped into a sphere located at the optical center of the image, with a radius of one, and the z-axis of the sphere is aligned with the optical axis. The following concepts of projective geometry are based on [73], as this provides important tools for the extraction of attitude information from camera images. Projective geometry is an elementary non-metrical form of geometry, meaning that it is not based on a concept of distance. In two dimensions it begins with the study of configurations of points and lines. In this work, projective geometry describes the properties of projective transformations which take place e.g. in cameras. The projective action of a camera on points in 3D-space can be expressed as

$$\mathbf{r}_{2\mathrm{D}} = \mathbf{C}_{\mathrm{cam}} \cdot \mathbf{r}_{3\mathrm{D}} \tag{5.1}$$

where $\mathbf{C}_{\mathrm{cam}}$ is camera transformation matrix, $\mathbf{r}_{2\mathrm{D}}$ and $\mathbf{r}_{3\mathrm{D}}$ denote respectively the homogenous coordinates of points in real 3D space and on the 2D image plane [74], as shown in Figure 5.2.

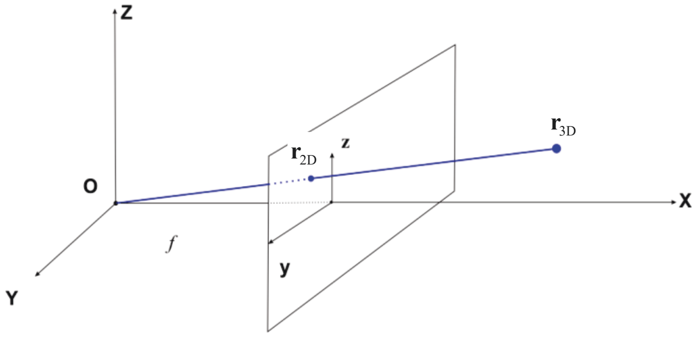


Figure 5.2: Projection of a point in 3D-space onto the camera image plane

A line in 3D-space is represented by

$$\mathbf{r}_{3D}(\lambda) = \mathbf{A} + \lambda \mathbf{D} \tag{5.2}$$

$\mathbf{A}$ is a finite point on the line and $\mathbf{D} = \begin{bmatrix} \mathbf{d}^T & 0 \end{bmatrix}^T$ denotes the line direction in homogeneous coordinates. This line is visualized in Figure 5.3 in blue, and the direction vector is shown in red. The camera projects the 3D-line onto a line on the 2D image plane, with $\mathbf{r}_{2D}(\lambda) = \mathbf{C}_{\text{cam}} \cdot \mathbf{r}_{3D}(\lambda)$, which is shown in green color. The intersection of $\mathbf{r}_{3D}(\lambda)$ with the plane at infinity $\boldsymbol{\pi}_\infty = [\mathbf{0}_{4\times1}]$ is the infinite point $\mathbf{r}_{3D,\infty} = \begin{bmatrix} \mathbf{d}^T & 0 \end{bmatrix}^T$, which is equal to the direction $\mathbf{D}$ of the line and is independent from the finite point $\mathbf{A}$. The projection of $\mathbf{r}_{3D,\infty}$ onto the image plane is the vanishing point $\mathbf{r}_{2D,vp}$, and $\mathbf{r}_{2D,vp} = \mathbf{C}_{\text{cam}} \cdot \mathbf{r}_{3D,\infty}$. The projection of all 3D-lines parallel to $\mathbf{r}_{3D}(\lambda)$ ends in the same VP on the image plane [74].

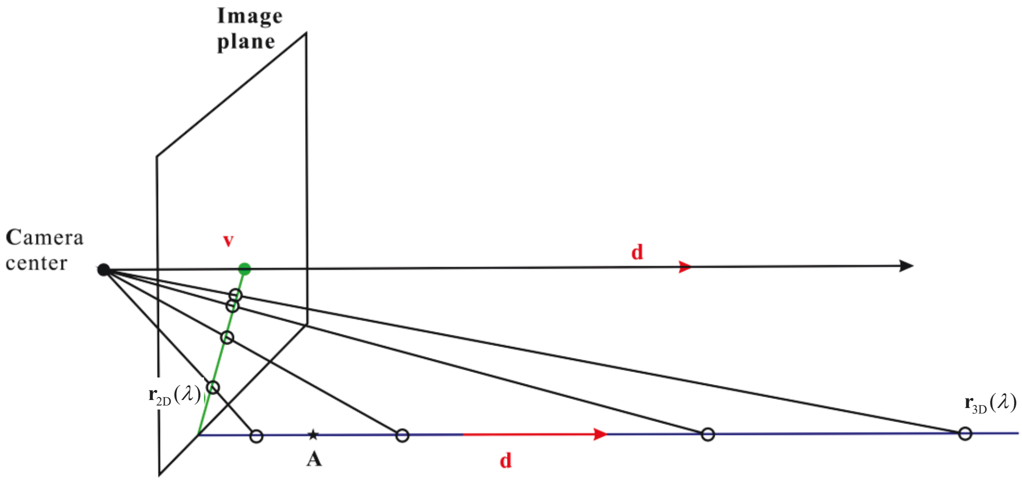


Figure 5.3: Projection of a line in 3D-space onto the camera image plane

The benefit of extracting and tracking VPs for indoor navigation applications is that they contain information about attitude changes, which is used for navigation aiding. The relative attitude between two camera poses in 3D-space is expressed using three unknown angles. The location of vanishing points in the image plane conveys a great deal of information describing the position of the camera relative to the scene so that the attitude change can be calculated. Details will be given in Section 5.2.1.

5.1.2 Vanishing point detection

Vanishing point detection in this work consists of three main steps, namely edge detection, line detection and vanishing point localization. First of all, an edge detection algorithm can significantly reduce the amount of data to be processed and may therefore filter out information that may be regarded as less relevant, while preserving the important structural properties of an image. Secondly, making the use of the line detection algorithm, straight lines are separated from other edges. Lastly, the random sample consensus (RANSAC) is applied to localize the vanishing point.

Edge detection

One common preprocessing step in analyzing digital images is to find the edges, i.e., points where the magnitude of the gradient is high in one direction as compared with the rest of the image. Pixels where this gradient magnitude is above a particular threshold are identified as edges or edge pixels. The following basic concept and theories of edge detection are based on [75], [76] and [77].

There are many methods for edge detection, but most of them can be classified into two categories, search-based methods and zero-crossing based methods [76]. The search-based methods detect edges by first computing a measure of edge strength, usually a first-order derivative expression such as the gradient magnitude, and then searching for local directional maxima of the gradient magnitude using a computed estimate of the local orientation of the edge, usually the gradient direction. The zero-crossing based methods search for zero crossings in a second-order derivative expression computed from the image in order to find edges, usually the zero-crossings of the Laplacian or the zero-crossings of a non-linear differential expression. As a pre-processing step to edge detection, a smoothing stage, typically Gaussian smoothing, is applied.

The edge detection methods that have been published mainly differ in the types of smoothing filters that are applied and the way the measures of edge strength are computed. As many edge detection methods rely on the computation of image gradients, they also differ in the types of filters used for computing gradient estimates in the x- and y-directions [76].

Rather than actually computing a derivative of the image, the gradient is usually approximated by evaluating the convolution of the image with a small kernel or mask. Common convolution kernels used for this task include those proposed by Roberts [78], Prewitt [79] and Sobel [80]. Convolution of the digital image with the first mask of each pair produces the gradient in the vertical direction $\mathbf{G}_{\mathrm{V}}$ and convolution with the second mask produces the gradient in the horizontal direction $\mathbf{G}_{\mathrm{H}}$.

Convolution kernels such as these are commonly used to approximate the derivative of a digital image. The Sobel kernels provide a smoothing effect which is helpful in suppressing noise. As Gonzalez and Woods [81] observed, 2×2 masks are simple computationally, but the symmetry about a center point offered by odd-dimensioned masks is more useful for determining edge directions. Larger masks provide more accurate approximations of the derivative, since they incorporate more information into each calculation. However, larger masks also require more computations.

The true gradient magnitude is determined by evaluating the Euclidean norm of the vertical and horizontal gradients for each pixel, but the less computationally expensive method of simply adding their absolute values as shown in Equation (5.3) is commonly used when constrained by data processing capacity [81].

$$\left\|\mathbf{G}(i,j)\right\| \approx \left|\mathbf{G}_{\mathrm{V}}(i,j)\right| + \left|\mathbf{G}_{\mathrm{H}}(i,j)\right| \tag{5.3}$$

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It has been developed by John F. Canny in 1986 [77]. The main steps of Canny edge algorithm are noise reduction, finding the intensity gradient of the image and tracing edges through the image and hysteresis thresholding. Noise reduction is not only the first stage of the Canny edge algorithm, but also a very important step. Because the Canny edge algorithm is susceptible to the noise from raw unprocessed image data, it uses a filter based on a Gaussian (bell curve). The raw image is convolved with the Gaussian filter. The result is a slightly blurred version of the original which is not affected by a single noisy pixel to any significant degree.

The second stage is finding the intensity gradient of the image. An edge in an image may point in a variety of directions, so the Canny algorithm uses four masks to detect horizontal, vertical and diagonal edges in the blurred image. The edge detection operator returns a value for the first derivative in the horizontal direction ($\mathbf{G}_{\mathrm{H}}$) and the vertical direction ($\mathbf{G}_{\mathrm{V}}$). From the following equation, the edge gradient and direction can be determined.

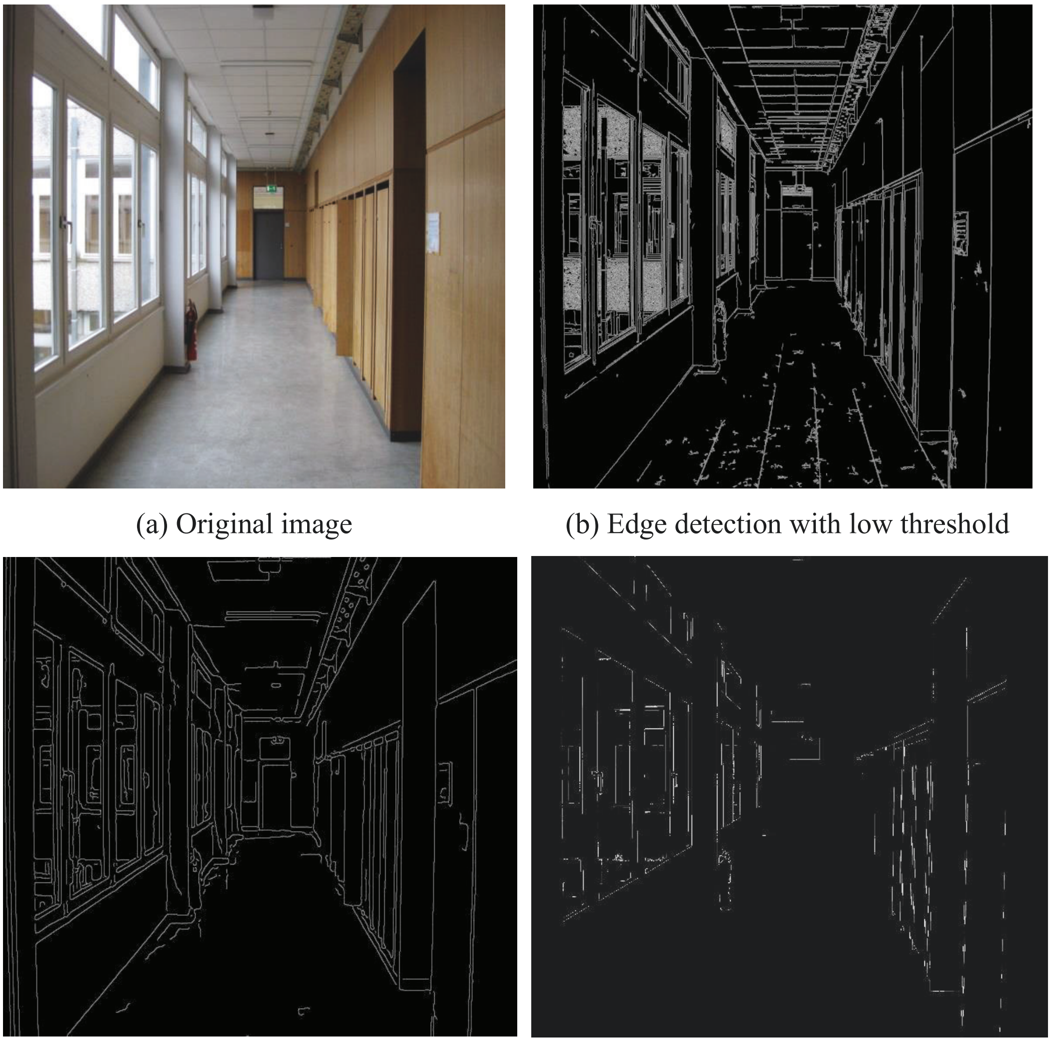
$$\begin{aligned} \mathbf{G}(i,j) &= \sqrt{\mathbf{G}_{\mathrm{V}}(i,j)^2 + \mathbf{G}_{\mathrm{H}}(i,j)^2} \\ \mathbf{\Theta}(i,j) &= \arctan\left(\frac{\mathbf{G}_{\mathrm{V}}(i,j)}{\mathbf{G}_{\mathrm{H}}(i,j)}\right) \end{aligned} \tag{5.4}$$

The edge direction angle is rounded to one of four angles representing vertical, horizontal and the two diagonals (0, 45, 90 and 135 degrees).

The last stage is tracing edges through the image and hysteresis thresholding. Large intensity gradients are more likely to correspond to edges than small intensity gradients. Making the assumption that important edges should be along continuous curves in the image allows us to follow a faint section of a given line and to discard a few noisy pixels that do not constitute a line but have produced large gradients. Therefore a high threshold should be applied at the beginning. It can fairly mark out the genuine edges. Starting from these, using the directional information derived earlier, edges can be traced in the image. The lower threshold is applied to trace faint sections of edges until it finds the starting point.

Canny presented that strong and weak edges are determined by establishing both high and low gradient thresholds. Strong edges occur where the magnitude of the image gradient is above the upper threshold. Weak edges occur where the gradient is between the upper and lower thresholds. Only weak edges which are adjacent to strong edges are declared as edge pixels in the final edge image. Once this process is complete a binary image will be obtained, where each pixel is marked as either an edge pixel or a non-edge pixel. From complementary output from the edge tracing step, the binary edge map obtained in this way can also be treated as a set of edge curves, which after further processing can be represented as polygons in the image domain [77]. No matter which method is used, the final result of edge detection is a binary image, where ones represent pixels that are declared as edges and zeros represent all other pixels. Figure 5.4 shows the result of performing the Canny edge detection operation on an image of a hallway in the Hölderlin-Building-F of the University of Siegen. Canny edge detection can be used with the built-in function edge in Matlab. This function contains a number of adjustable parameters, which can affect the computation time and effectiveness of the algorithm.

The usage of two thresholds with hysteresis allows more flexibility than in a single-threshold approach, but general problems of thresholding approaches still apply. A threshold set too high can miss important information, as shown in Figure 5.4 (d). A number of important structure lines on the ceiling have failed to be detected. On the other hand, a threshold set too low will falsely identify irrelevant information (such as noise) as important, as shown in Figure 5.4 (b), which requires more computational effort and processing time.



(a) Original image (b) Edge detection with low threshold

(c) Edge detection with large Gaussian filter (d) Edge detection with high threshold

Figure 5.4: The results of Canny edge detection

The size of the Gaussian filter: the smoothing filter used in the first stage directly affects the results of the Canny algorithm. Smaller filters cause less blurring, and allow detection of smaller and sharper lines. A larger filter causes more blurring, smearing out the value of a given pixel over a larger area of the image, which is more helpful for detecting larger and smoother edges, as shown in Figure 5.4 (c).

The Canny algorithm is adaptable to various environments. Its parameters allow it to be tailored to the recognition of edges of differing characteristics depending on the particular requirements of a given implementation. But it is difficult to give a generic threshold that works well for all images and conditions, the parameters need to be adjusted according to the actual situation. [77]

Line detection

After detecting the edges, the next step is to extract straight lines. The problem of identifying straight lines in digital images has been investigated by many researchers and lots of methods have been developed. However, most methods are at least loosely based on either the Hough transform or Burns line extractor. According to the result of [75], though the processing times of Hough transform is not always faster than the other method, it is still chosen as the line detection method for this work because both the Matlab and OpenCV software packages contain functions for implementing it within their respective libraries. The following brief introduction and theoretical basis of Hough transform are based on [82] and [83].

The Hough transform (HT), which is presented by Paul Hough in 1962, is a technique used to find shapes in a binary digital image. The classical Hough transform is concerned with the identification of lines in the image, but later the Hough transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses. The Hough transform as it is universally used today is invented by Richard Duda and Peter Hart in 1972. It is also called generalized Hough transform.[83]

In order to use the Hough transform to find lines in digital images, an appropriate parameterization of a line must be selected. Often, lines in the Cartesian space are represented by a slope a and y_i-intercept b, as shown in Equation (2.6). [75]

$$y_i = a \cdot x_i + b \tag{5.5}$$

These parameters a and b can be used to represent a straight line in the Cartesian space as a single point (a, b) in the parameter-space.

And if the variables and parameters are reversed, Equation (2.6) can be rewritten as follows:

$$b = -x_i \cdot a + y_i \tag{5.6}$$

Likewise, a straight line in (a,b) space can also represent a single point in (x,y) space. The relationship between *x-y* space and parameter-space is shown in Figure 5.5.

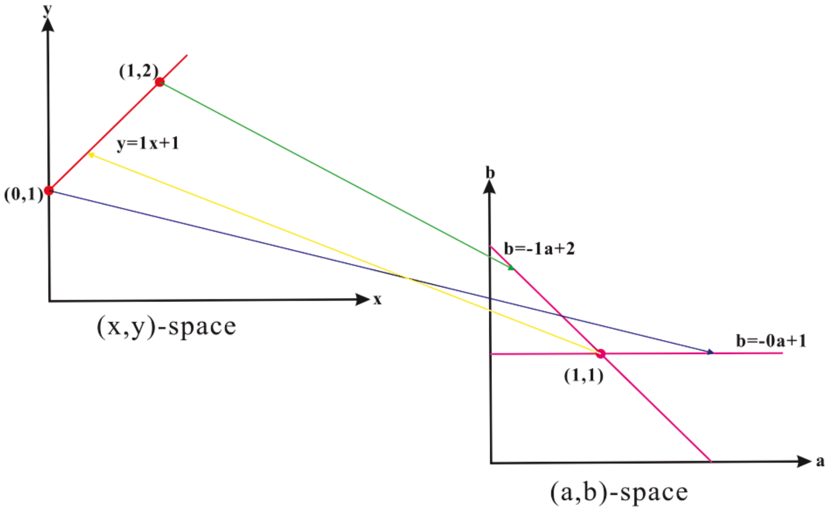


Figure 5.5: Illustration of the basic idea of Hough transform for lines

However, this parameterization presents difficulties when the Hough transform is applied. Because the slope parameter is unbounded, as manifested by the infinite slope of vertical lines. Thus, for computational reasons, Duda and Hart have proposed the use of a different pair of parameters, denoted ρ and θ, for the lines in the Hough transform [83]. These two values, taken in conjunction, define a polar coordinate. According to [83], the equation of a line corresponding to this geometry is

$$\rho = x_i \cdot \cos\theta + y_i \cdot \sin\theta \tag{5.7}$$

The parameter ρ represents the distance between the line and the origin, while θ is the angle of the vector from the origin to this closest point. It is therefore possible to associate with each line of the image a pair (ρ,θ) which is unique if $\theta \in [0,\pi)$, $\rho \in \mathbb{R}$ or if $\theta \in [0,2\pi)$, $\rho \geq 0$. For the image plane θ can vary between ±90 degree and ρ can vary between $\pm d$, where d is the diagonal of the image frame in pixels. In this case, a straight line can then be transformed into a single point in the parameter space (ρ,θ); this is also called the Hough space.

Similarly, for an arbitrary point on the image plane with coordinates, e.g., (x_0, y_0) the lines that go through it are the pairs (ρ,θ) with

$$\rho(\theta) = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta \tag{5.8}$$

Where ρ denotes the distance between the line and the origin; ρ is determined by θ.

This corresponds to a sinusoidal curve in the (ρ, θ) plane, which is unique to that point. If the curves corresponding to two points are superimposed, the location (in the Hough space) where they cross corresponds to a line (in the original image space) that passes through both points. More generally, a set of points that form a straight line will produce sinusoids which cross at the parameters for that line. Thus, the problem of detecting collinear points can be converted to the problem of finding concurrent curves.

In this work, the Hough function implements the standard Hough transform (SHT). The SHT is a parameter space matrix whose rows and columns correspond to ρ and θ values respectively. The elements in the SHT represent accumulator cells. Initially, the value in each cell is zero. Then, for every non-background point in the image, ρ is calculated for every θ. ρ is rounded off to the nearest allowed row in SHT. That accumulator cell is incremented. At the end of this procedure, a value of Q in SHT (ρ, θ) means that Q points lie on the line specified θ and ρ in the *x-y* plane. The peak values in the SHT represent potential lines in the input image.

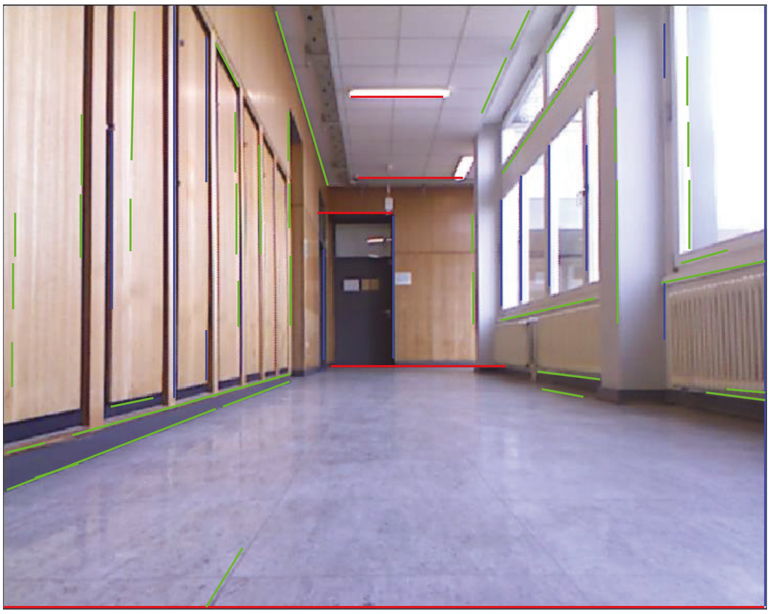


Figure 5.6: The results of line detection in the indoor environment

The performance of the line detection is shown in Figure 5.6. The blue lines and the red lines are respectively the totally vertical and totally horizontal structure lines in the indoor environment. In contrast, the green lines represent the general lines in the original image that is neither vertical nor horizontal, and one part of them is classified to go along the forward direction, which plays an important role in detecting the central vanishing point.

Vanishing point localization

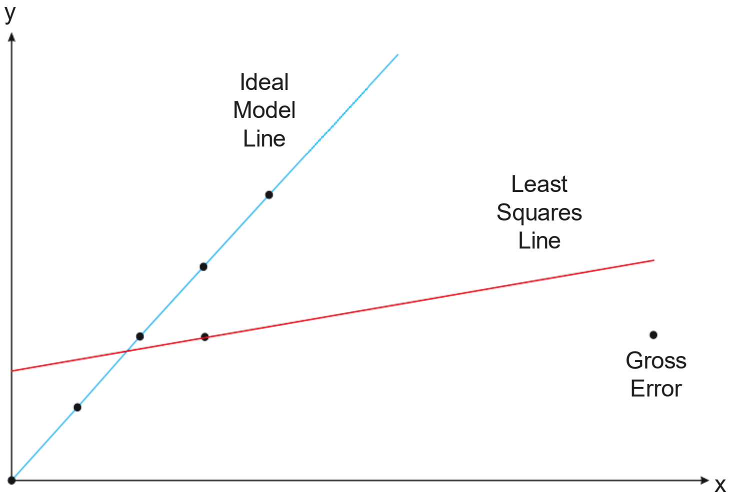
The projection loses valuable information like the depth of the scene. According to introduce of Section 5.1.1, though straight lines stay straight in projections, the parallel ones don't stay parallel but seem to intersect at a point. In other words, parallel lines in the image must be retrieved for calculation of their intersection point, the vanishing point. This is done by first looking for the edges of objects in the image and then identifying the straight lines among them [69].

However, when real-world images are processed, due to image noise the extracted line segments do not perfectly intersect a single VP. For this reason, a robust estimation method is necessary which extracts VP in the presence of noise. As the mostly used VP detecting algorithm in recent years, the random sample consensus (RANSAC) method is applied in this work.

The algorithm has been introduced by Fischler and Bolles at SRI International in 1981, as an iterative method to estimate parameters of a mathematical model from a set of observed data which contains outliers [84]. It is a non-deterministic (stochastic) algorithm in the sense that it produces a reasonable result only with a certain probability, with this probability increasing as more iteration are allowed [84]. The RANSAC procedure is opposite to that of conventional smoothing techniques: rather than using as much of the data as possible to obtain an initial solution and then attempting to eliminate the invalid data points, RANSAC uses as small an initial data set as feasible and enlarges this set with consistent data when possible.

According to [85] and [86], the RANSAC method has a basic assumption that the data consists of "inliers", i.e., data whose distribution can be explained by some set of model parameters and "outliers" which are data that do not fit the model. For instance, the outliers can come from extreme values of the noise or from erroneous measurements or incorrect hypotheses about the interpretation of data. RANSAC also assumes that, given a (usually small) set of inliers, there exists a procedure which can estimate the parameters of a model that optimally explains or fits this data. In order to better illustrate this method a simple example is cited that find the best fit line in two dimensions according to a set of observations.

Assuming that this set contains both inliers, i.e., points which approximately can be fitted to a line, and outliers, points which cannot be fitted to this line, classical techniques for parameter estimation, such as least squares, generally produces a line with a bad fit to the inliers for line fitting. The reason is that it is optimally fitted to all points, including the outliers. But RANSAC can produce a model which is only computed from the inliers. The probability of choosing only inliers in the selection of data is sufficiently high. There is no guarantee for this situation, however, and there are a number of algorithm parameters which must be carefully chosen to keep the level of probability reasonably high. The result of this simple example based on [84] is shown in the following figure.



Point	(x,y)
1	(0,0)
2	(1,1)
3	(2,2)
4	(3,2)
5	(3,3)
6	(4,4)
7	(10,2)

Figure 5.7: The result of the best fit line based on point coordinates [84]

According to [84] and [75], the RANSAC paradigm is stated as follows.

The procedure for establishing a model from a set S of datum points that is known to contain a proportion ε of outliers is outlined below.

1. Randomly select a minimum subset s from S and instantiate the model with s.

2. Determine the set of points S_i that is within a threshold t of the model established by s . The set S_i is the consensus set of S.

3. If the size of S_i is greater than a threshold T , re-estimate the model based on all the points in S_i and terminate.

4. If the size of S_i is less than T and fewer than N trials have been performed, select a new random minimum subset s and repeat steps 2 through 4.

5. After N trials, re-estimate the model with the largest consensus set S_i and terminate.

The RANSAC paradigm contains three unspecified parameters: specifically the threshold t is the error tolerance used to determine whether or not a point is compatible with a model; the threshold T which determines how many data should fit the model before terminating; and the maximum number N of random minimum subsets to examine before terminating the process.

The threshold value t used for declaring data as either inliers or outliers is often determined empirically. By assuming measurement errors are zero-mean with a known standard deviation, t can be computed from a χ^2 distribution. The size threshold T for determining what is an adequately large consensus set is determined from the expected number of outliers as in the following equation.

$$T = (1-\varepsilon)\cdot S \tag{5.9}$$

Lastly, the minimum number of iterations N required to be assured with probability p that at least one minimum subset s is free from outliers is determined by

$$N = \frac{\log(1-p)}{\log\left(1-(1-\varepsilon)^s\right)} \tag{5.10}$$

where ε is the proportion of outliers expected to be found in S. Naturally, p is preferred to be very nearly equal to 1, with 0.99 frequently used in practice.

In many practical applications, ε may not be known. Under these circumstances, the threshold T of inliers needed to end the loop cannot be determined. Instead, only the minimum number of iterations is used for deciding when to terminate the process. In order to calculate the minimum number of iterations to perform, a worst case value can be used initially, and both ε and N can be recomputed in subsequent iterations. If one random minimum subset s produces a proportion of outliers smaller than the current value of ε, N is recomputed from Equation (5.10) using the new, smaller ε. In the case where N is found to be smaller than the number of iterations that have already been performed, the algorithm terminates and the largest consensus set is used to estimate the model [75].

The method described herein calculates the VP locations using a voting scheme. An intersection of two randomly selected line segments is selected as a VP candidate, at the same time an assisted measure is introduced. This process is repeated several times and the largest set of inliers is selected. Based on this so-called consensus set, the optimal central VP is selected as the intersection point with the most votes inside the image. Figure 5.8 shows the result of central VP localization, which is marked with the yellow cross.

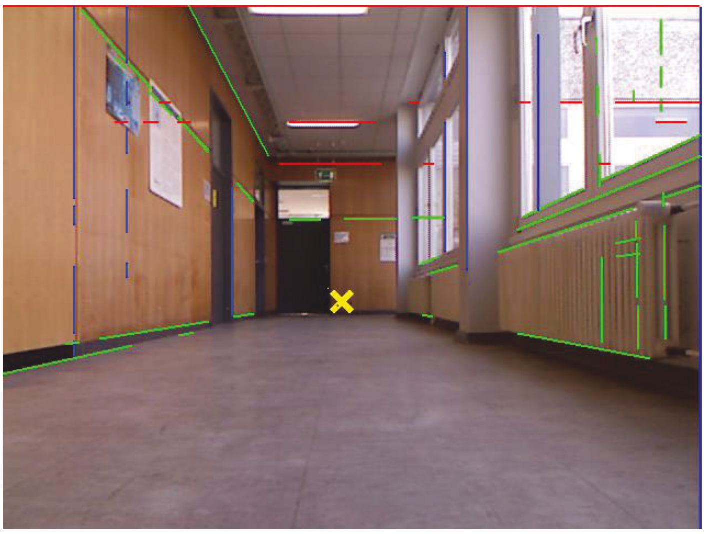


Figure 5.8: Localization of central vanishing point

To determine the vertical and horizontal VPs, an additional decision condition with an assisted measure is applied for VP detection. The assisted measure is the orthogonal distance between this VP candidate and the selected auxiliary line. The auxiliary line in this work consists of two straight lines, one is the horizontal line through the central point of the input image, and the other is the vertical line through the central point of the input image. Thus the assisted measure d_x and d_y represent respectively the distances between the VP candidate and the horizontal auxiliary line, as well as the vertical auxiliary line. When the camera moves through the hallway of the building with the small roll and pitch angels, according to the perspective geometry except for the location of the central VP lie inside the image, the vertical and horizontal VPs are localized very far away outside the image. If the selected intersection is the vertical VP or horizontal, the assisted measure d_x or d_y should be over the pre-set threshold. Therefore, the determination of vertical or horizontal VPs is that the point obtains the most parts of the votes and meets the requirement of the assisted measure.

5.2 DCM based INS/visual-gyro integration

5.2.1 Attitude estimation with detected vanishing points

After the vanishing points have been successfully found, the remaining task is to extract information of the attitude changes from the locations of the vanishing points. The rotation matrix of the camera may be resolved by using VPs and the calibration matrix of the camera [69], which is shown with Equation (5.11)

$$\mathbf{V}_{\mathrm{vp}} = \mathbf{K}_{\mathrm{cam}} \cdot \mathbf{C}_{\mathrm{R}} \tag{5.11}$$

where $\mathbf{V}_{\mathrm{vp}}$ is the vanishing point location matrix, $\mathbf{K}_{\mathrm{cam}}$ is the calibration matrix and $\mathbf{C}_{\mathrm{R}}$ is the camera rotation matrix. $\mathbf{V}_{\mathrm{vp}}$ includes the locations of the three vanishing points (horizontal, vertical and central): $\mathbf{V}_{\mathrm{vp}} = \begin{bmatrix} \mathbf{V}_{\mathrm{vp},1} & \mathbf{V}_{\mathrm{vp},2} & \mathbf{V}_{\mathrm{vp},3} \end{bmatrix}$. Each vanishing point is represented respectively in the image coordinate frame using homogeneous pixel coordinates $\mathbf{V}_{\mathrm{vp},i} = \begin{bmatrix} r_{vp,i,x} & r_{vp,i,y} & l_{\mathrm{cam},i} \end{bmatrix}^T$, where $r_{vp,i,x}$ and $r_{vp,i,y}$ denote the coordinates of the vanishing point; l_{cam} represents a camera constant that can be pre-estimated. The calibration matrix $\mathbf{K}_{\mathrm{cam}}$ is expressed by the following equation:

$$\mathbf{K}_{\mathrm{cam}} = \begin{bmatrix} f_{\mathrm{cam},\,x} & 0 & u_{\mathrm{cam}} \\ 0 & f_{\mathrm{cam},\,y} & v_{\mathrm{cam}} \\ 0 & 0 & 1 \end{bmatrix} \tag{5.12}$$

which contains information about the intrinsic camera parameters: focal length $\left(f_{\mathrm{cam},\,x}, f_{\mathrm{cam},\,y}\right)$ and principal point $\left(u_{\mathrm{cam}}, v_{\mathrm{cam}}\right)$. They can be approximated using the values in the image file headers, yet their exact values are obtained by calibrating the camera. The matrix $\mathbf{C}_{\mathrm{R}}$ for the three-dimensional rotation of the camera is shown in the following equation.

$$\mathbf{C}_{\mathrm{R}} = \begin{bmatrix} \mathrm{c}\varphi\mathrm{c}\phi & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{s}\theta - \mathrm{s}\varphi\mathrm{c}\theta & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{c}\theta + \mathrm{s}\varphi\mathrm{s}\theta \\ \mathrm{s}\varphi\mathrm{c}\phi & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{s}\theta + \mathrm{c}\varphi\mathrm{c}\theta & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{c}\theta - \mathrm{c}\varphi\mathrm{s}\theta \\ -\mathrm{s}\phi & \mathrm{c}\phi\mathrm{s}\theta & \mathrm{c}\phi\mathrm{c}\theta \end{bmatrix} \tag{5.13}$$

where $\mathrm{c}\psi = \cos\psi$, $\mathrm{s}\psi = \sin\psi$ and θ, ϕ, φ represent the roll, pitch and yaw respectively.

However, there are limitations of the visual-gyro application. Though the visual-gyro does not suffer from drift errors, its availability highly depends on the indoor environment. For instance, the visual-gyro does not work when the vanishing point finding is not available, which happens when no vanishing point can be found or too many disturbing lines are detected in the image. Moreover, since the high computational requirement, the frequency of the visual-gyro output is relatively low and the performance for fast rotation estimation is limited.

Making use of the complementary nature of INS and visual gyroscope, the INS/visual-gyro integration is expected to yield a synergetic effect, which can overcome the limitations of both systems.

5.2.2 System modelling

In this work, DCM based system models are used for INS/visual-gyro integration to estimate the attitude and gyro-bias.

The advantage of using a DCM based model can provide a linear process model for the DCM update with respect to its parameters. However, augmenting the gyro bias vector makes the process model non-linear because it introduces state multiplication which can be handled using a pseudo-linear process model as shown in [88]. Another important benefit of using the DCM attitude representation is avoiding the singularity problem that occurs with Euler angle representation when the pitch angle reaches plus/minus 90 degrees.

The elements of the DCM are trigonometric functions of the Euler angles defined as

$$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}} = \begin{bmatrix} \mathrm{c}\varphi\mathrm{c}\phi & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{s}\theta - \mathrm{s}\varphi\mathrm{c}\theta & \mathrm{c}\varphi\mathrm{s}\phi\mathrm{c}\theta + \mathrm{s}\varphi\mathrm{s}\theta \\ \mathrm{s}\varphi\mathrm{c}\phi & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{s}\theta + \mathrm{c}\varphi\mathrm{c}\theta & \mathrm{s}\varphi\mathrm{s}\phi\mathrm{c}\theta - \mathrm{c}\varphi\mathrm{s}\theta \\ -\mathrm{s}\phi & \mathrm{c}\phi\mathrm{s}\theta & \mathrm{c}\phi\mathrm{c}\theta \end{bmatrix} \tag{5.14}$$

where $\mathbf{C}_{\mathrm{b}}^{\mathrm{n}}$ is the DCM rotation matrix that transforms the inertial measurements from the body frame to the local navigation frame. It can be found from Equation (5.27) that the DCM matrix is equal to the camera rotation matrix: $\mathbf{C}_{\mathrm{b}}^{\mathrm{n}} = \mathbf{C}_{\mathrm{R}}$. The DCM can be written in terms of the nine elements as:

$$\mathbf{C}_{\mathrm{b}}^{\mathrm{n}} = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \tag{5.15}$$

The element in the *i*th row and the *j*th column represents the cosine of the angle between the *i*-axis of the reference frame and the *j*-axis of the body frame [89].

For the integration system, the state vector to be estimated is composed of the nine elements of the DCM and gyro triad bias vector, which is described as the following form:

$$\begin{gathered} \mathbf{x} = \begin{bmatrix} \mathbf{c}_1^T & \mathbf{c}_2^T & \mathbf{c}_3^T & \mathbf{b}_{gyro}^T \end{bmatrix}^T \\ \text{where} \\ \mathbf{c}_1 = \begin{bmatrix} c_{11} & c_{12} & c_{13} \end{bmatrix}^T, \mathbf{c}_2 = \begin{bmatrix} c_{21} & c_{22} & c_{23} \end{bmatrix}^T, \mathbf{c}_3 = \begin{bmatrix} c_{31} & c_{32} & c_{33} \end{bmatrix}^T \\ \mathbf{b}_{gyro}^T = \begin{bmatrix} b_{gyro,x} & b_{gyro,y} & b_{gyro,z} \end{bmatrix}^T \end{gathered} \tag{5.16}$$

The system process update can be divided into two parts, which are update of the DCM elements and update of the gyro bias. The rotation angle vector which is the turn angle of the body frame with respect to a fixed inertial frame is used to update the attitude DCM. The attitude update with respect to a local navigation frame is composed of two updates: one due to the body frame rotation with respect to the inertial frame and the other due to the rotation of the navigation frame with respect to the fixed inertial frame [90]. For low cost gyros, the craft rate can be ignored and the update will be mainly due to the turn rate of the body frame with respect to the inertial frame. The discrete-time solution is given as

$$\mathbf{C}_{\mathrm{b},k}^{\mathrm{n}} = \mathbf{C}_{\mathrm{b},k-1}^{\mathrm{n}} \mathbf{A}_{k-1} \tag{5.17}$$

The matrix **A** is computed as [91]

$$\mathbf{A}_{k-1} = e^{\int_{t_{k-1}}^{t_k} \boldsymbol{\omega}_{\mathrm{ib}}^{\mathrm{b}} \times dt} \tag{5.18}$$

where $\boldsymbol{\omega}_{\mathrm{ib}}^{\mathrm{b}}$ is rotation rate of the body frame. Assuming that the angular velocity vector has little change during the update interval, we can approximate the angle rotation vector as

$$\boldsymbol{\sigma} = \int_{t_{k-1}}^{t_k} \boldsymbol{\omega}_{\mathrm{ib}}^{\mathrm{b}} dt, \ \boldsymbol{\sigma} \approx \boldsymbol{\omega}_{\mathrm{ib}}^{\mathrm{b}} \Delta t \tag{5.19}$$

The vector $\boldsymbol{\sigma}$ consists of σ_x σ_y and σ_z with the magnitude $|\boldsymbol{\sigma}|=\sqrt{\sigma_x^2+\sigma_y^2+\sigma_z^2}$. And

$$\boldsymbol{\sigma}\times=\begin{bmatrix} 0 & -\sigma_z & \sigma_y \\ \sigma_z & 0 & -\sigma_x \\ -\sigma_y & \sigma_x & 0 \end{bmatrix} \tag{5.20}$$

Equation (5.18) can be expanded as

$$\mathbf{A}_{k-1}=\exp[\boldsymbol{\sigma}\times]=\mathbf{I}+[\boldsymbol{\sigma}\times]+\frac{[\boldsymbol{\sigma}\times]^2}{2!}+\frac{[\boldsymbol{\sigma}\times]^3}{3!}+\cdots \tag{5.21}$$

Based on Equation (5.24), it can be shown that:

$$\begin{aligned} [\boldsymbol{\sigma}\times]^2 &= \begin{bmatrix} -(\sigma_y^2+\sigma_z^2) & \sigma_x\sigma_y & \sigma_x\sigma_z \\ \sigma_x\sigma_y & -(\sigma_x^2+\sigma_z^2) & \sigma_y\sigma_z \\ \sigma_x\sigma_z & \sigma_y\sigma_z & -(\sigma_x^2+\sigma_y^2) \end{bmatrix} \\ [\boldsymbol{\sigma}\times]^3 &= -(\sigma_x^2+\sigma_y^2+\sigma_z^2)[\boldsymbol{\sigma}\times] \\ [\boldsymbol{\sigma}\times]^4 &= -(\sigma_x^2+\sigma_y^2+\sigma_z^2)[\boldsymbol{\sigma}\times]^2 \\ &\vdots \end{aligned} \tag{5.22}$$

From Equation (5.21) and Equation (5.22), we can get:

$$\mathbf{A}_{k-1}=\mathbf{I}+\left[1-\frac{|\boldsymbol{\sigma}|^2}{3!}+\frac{|\boldsymbol{\sigma}|^4}{5!}-\cdots\right][\boldsymbol{\sigma}\times]+\left[\frac{1}{2!}-\frac{|\boldsymbol{\sigma}|^2}{4!}+\frac{|\boldsymbol{\sigma}|^4}{6!}-\cdots\right][\boldsymbol{\sigma}\times]^2 \tag{5.23}$$

which can be written as

$$\mathbf{A}_{k-1}=\mathbf{I}+\frac{\sin|\boldsymbol{\sigma}|}{|\boldsymbol{\sigma}|}[\boldsymbol{\sigma}\times]+\frac{1\text{-}\cos|\boldsymbol{\sigma}|}{|\boldsymbol{\sigma}|^2}[\boldsymbol{\sigma}\times]^2 \tag{5.24}$$

Equation (5.24) yields an exact representation of the attitude matrix. In this work, in order to reduce the computational effort, the matrix $\mathbf{A}_{k-1}$ is reformulated with a second order approximation. Based on Equation (5.21) and Equation (5.22), we may write:

$$\mathbf{A}_{k-1}\approx\begin{bmatrix} 1\text{-}\frac{(\sigma_y^2+\sigma_z^2)}{2} & \frac{\sigma_x\sigma_y}{2}-\sigma_z & \frac{\sigma_x\sigma_z}{2}+\sigma_y \\ \frac{\sigma_x\sigma_y}{2}+\sigma_z & 1\text{-}\frac{(\sigma_x^2+\sigma_z^2)}{2} & \frac{\sigma_y\sigma_z}{2}-\sigma_x \\ \frac{\sigma_x\sigma_z}{2}-\sigma_y & \frac{\sigma_y\sigma_z}{2}+\sigma_x & 1\text{-}\frac{(\sigma_x^2+\sigma_y^2)}{2} \end{bmatrix} \tag{5.25}$$

Considering the gyro bias $\mathbf{b}_{gyro}$, the error model with gyro measurement $\tilde{\boldsymbol{\omega}}_{\mathrm{ib}}^{\mathrm{b}}$ is given as:

$$\boldsymbol{\omega}_{\mathrm{ib}}^{\mathrm{b}} \approx \tilde{\boldsymbol{\omega}}_{\mathrm{ib}}^{\mathrm{b}} - \mathbf{b}_{gyro} \tag{5.26}$$

By ignoring the quadratic terms of the gyro errors of bias and noise, the matrix $\mathbf{A}$ can be approximated and written in terms of the exact matrix $\tilde{\mathbf{A}}$ as

$$\mathbf{A}_{k-1} \approx \tilde{\mathbf{A}}_{k-1} - \left[\mathbf{b}_{gyro}\Delta t\right]\times \tag{5.27}$$

The DCM update can be formulated as

$$\begin{aligned}
\mathbf{x}_{1:3,k} &= \tilde{\mathbf{A}}_{k-1}^{T}\mathbf{x}_{1:3,k-1} - \mathbf{x}_{1:3,k-1} \times \mathbf{x}_{10:12,k-1}\Delta t \\
\mathbf{x}_{4:6,k} &= \tilde{\mathbf{A}}_{k-1}^{T}\mathbf{x}_{4:6,k-1} - \mathbf{x}_{4:6,k-1} \times \mathbf{x}_{10:12,k-1}\Delta t \\
\mathbf{x}_{7:9,k} &= \tilde{\mathbf{A}}_{k-1}^{T}\mathbf{x}_{7:9,k-1} - \mathbf{x}_{7:9,k-1} \times \mathbf{x}_{10:12,k-1}\Delta t
\end{aligned} \tag{5.28}$$

The gyro bias vector can be modelled as a random walk plus constant. The system process model is given as

$$\begin{aligned}
\mathbf{x}_k &= \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{w}_{k-1} \\
\mathbf{F}_{k-1} &= \begin{bmatrix} \tilde{\mathbf{A}}_{k-1}^{T} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \text{-}\Delta t\ \mathbf{x}_{1:3}\times \\ \mathbf{0}_{3\times3} & \tilde{\mathbf{A}}_{k-1}^{T} & \mathbf{0}_{3\times3} & \text{-}\Delta t\ \mathbf{x}_{4:6}\times \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \tilde{\mathbf{A}}_{k-1}^{T} & \text{-}\Delta t\ \mathbf{x}_{7:9}\times \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} \end{bmatrix}
\end{aligned} \tag{5.29}$$

which yields a pseudo-linear form [92].

With system alignment between INS and visual-gyro, the DCM elements are linear related to the camera rotation matrix $\mathbf{C}_{\mathrm{R}}$. The linear system observation model can be derived from the visual-gyro model shown in Equation (5.11). The system measurement vector includes the positions of the detected vanishing points in the image frame: $\mathbf{y} = \begin{bmatrix} r_{vp,1,x} & r_{vp,1,y} & r_{vp,2,x} & r_{vp,2,y} & r_{vp,3,x} & r_{vp,3,y} \end{bmatrix}^{T}$.

The system observation model is given by Equation (5.30). $\boldsymbol{\eta}$ denotes the system measurement noise. The camera parameters included in matrix $\mathbf{H}$ are introduced in the previous section.

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x}_k + \boldsymbol{\eta}_k$$

$$\mathbf{H}_k = \begin{bmatrix} f_{\text{cam},x} & 0 & 0 & 0 & 0 & 0 & u_{\text{cam}} & 0 & 0 & \mathbf{0}_{1\times3} \\ 0 & 0 & 0 & f_{\text{cam},y} & 0 & 0 & v_{\text{cam}} & 0 & 0 & \mathbf{0}_{1\times3} \\ 0 & f_{\text{cam},x} & 0 & 0 & 0 & 0 & 0 & u_{\text{cam}} & 0 & \mathbf{0}_{1\times3} \\ 0 & 0 & 0 & 0 & f_{\text{cam},y} & 0 & 0 & v_{\text{cam}} & 0 & \mathbf{0}_{1\times3} \\ 0 & 0 & f_{\text{cam},x} & 0 & 0 & 0 & 0 & 0 & u_{\text{cam}} & \mathbf{0}_{1\times3} \\ 0 & 0 & 0 & 0 & 0 & f_{\text{cam},y} & 0 & 0 & v_{\text{cam}} & \mathbf{0}_{1\times3} \end{bmatrix} \quad (5.30)$$

Then Kalman filtering algorithm is employed in this work for estimating the state vector. If the gyro-bias is augmented in the system state, the process model is pseudo-linear and the extended Kalman filter (EKF) is applied for system integration. The EKF algorithm has been introduced in Section 4.1.4.

5.3 Field experiments

To verify the presented INS/visual-gyro integration, indoor field experiments have been performed. The hardware of the integrated system is shown in Figure 5.9.

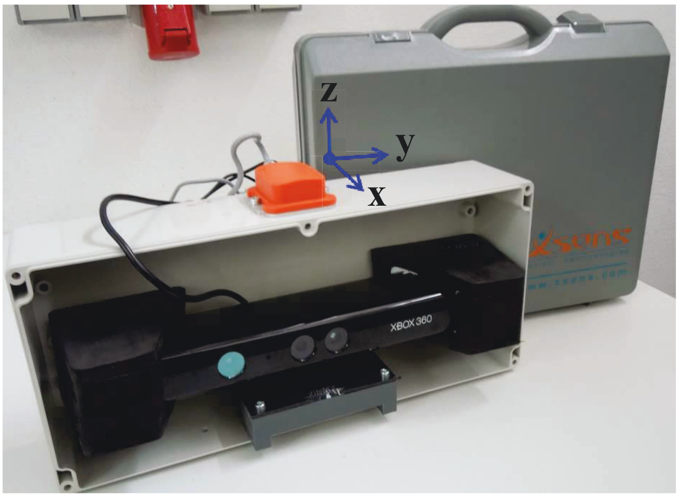


Figure 5.9: Experiment hardware

The inertial sensor is a MEMS IMU (Xsens MTi). The specifications of the IMU based on manufacturer datasheet are given in Table 3.1. The three axes of the body frame coordinate are marked in Figure 5.9. And raw, pitch and yaw represent the rotation angles of the body frame from the local navigation frame around x, y and z axes respectively. The camera employed here is the Kinect (Microsoft) sensor. Kinect is a 2D/3D sensor. Only the 2D camera function is used in this experiment. The main parameters are shown in Table 5.1.

Table 5.1: Kinect parameters

Depth sensor range (m)	From 1.2 to 3.5
Horizontal field of view (deg)	57
Vertical field of view (deg)	43
Frame rate (FPS)	≤ 30
Resolution of depth stream (pixel)	QVGA 320×240
Resolution of color stream (pixel)	VGA 640×480

5.3.1 Turntable test

The first experiment is a 2D test. As shown in Figure 5.10. The sensors are mounted on a single axis turntable (Acutronic 1-axis rate table series AC1120S). The attitude accuracy of the turntable is less than 0.005 degree, and rate resolution achieves to 0.001 deg/s.

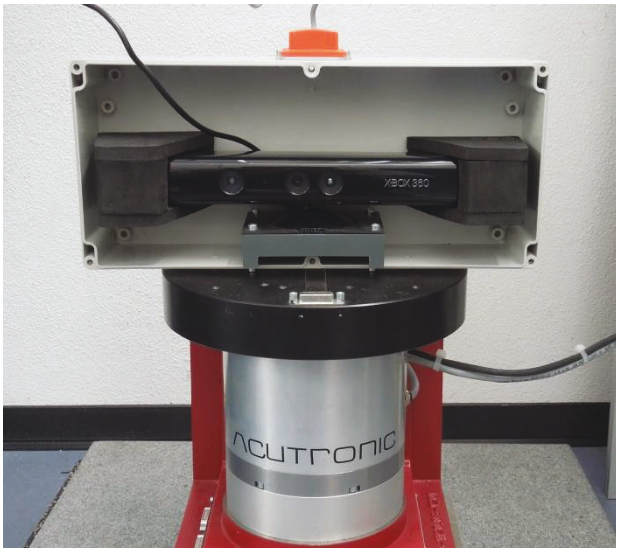


Figure 5.10: Experiment with a turntable

The turntable was set to rotate with a slow rate of 0.5 deg/s lasting 24 s. The frame rate of the gyroscope is 30 FPS and output rate of the IMU is 100 Hz. The estimation results are shown in Figure 5.11 and Table 5.2 and Figure 5.12. Figure 5.11 shows the attitude estimation errors using standalone INS, visual-gyro and DCM based INS/visual-gyro integration respectively. The mean and stand covariance values of the errors are presented in Table 5.2. Figure 5.12 shows the gyro-bias (yaw) estimation error using the integrated system.

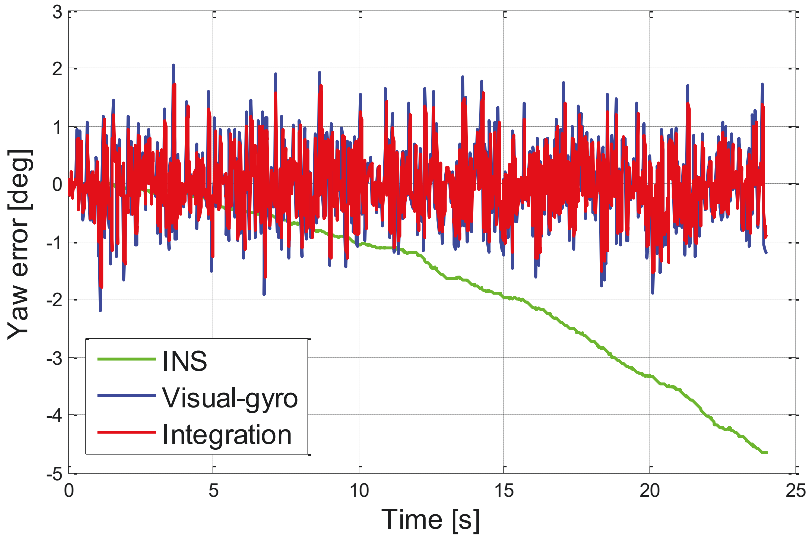


Figure 5.11: Attitude estimation errors

Table 5.2: Error parameters

Method / Error (°)	INS	Visual-gyro	INS/visual-gyro
Mean	1.66	0.55	0.44
Standard deviation	1.38	0.43	0.34

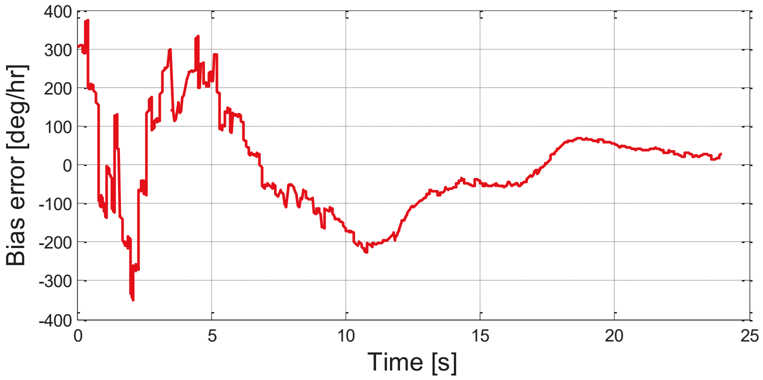


Figure 5.12: Gyro bias estimation error using INS/visual-gyro integration

It can be found that the standalone INS suffers from the drift error while the two other systems do not; the integration can slightly improve the attitude estimation performance from visual-gyro; the gyro bias can be also estimated using the INS/visual-gyro integration.

5.3.2 Pedestrian experiment

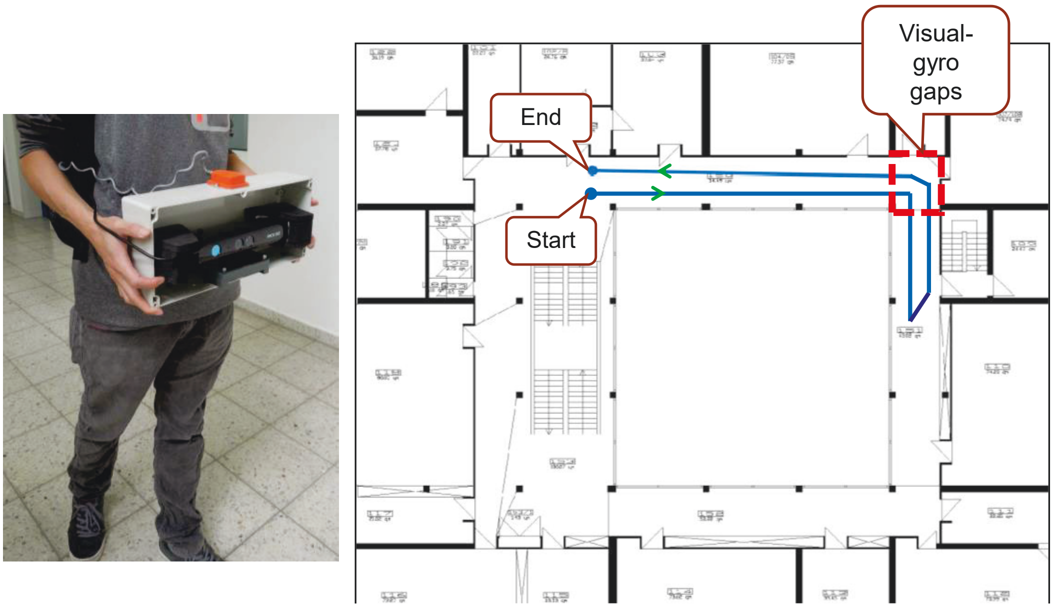


Figure 5.13: Pedestrian experiment

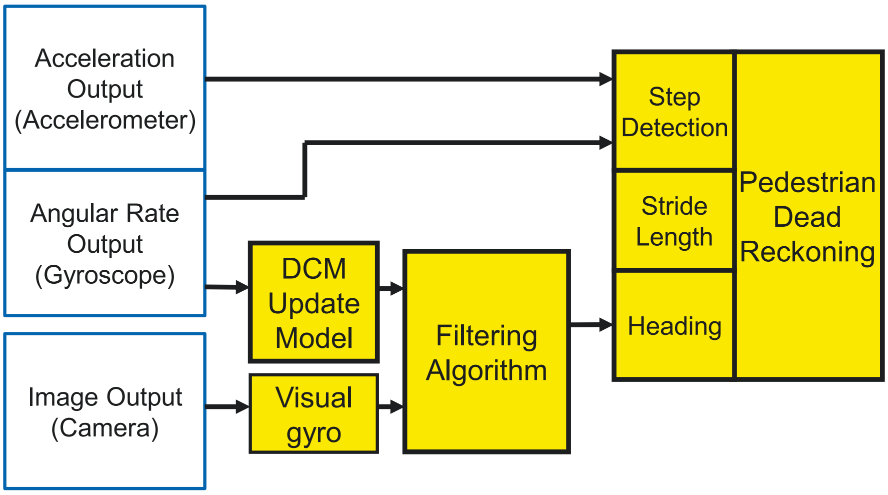


Figure 5.14: Combination with pedestrian dead reckoning system

The second experiment is conducted in the H-F building at the University of Siegen. The test-bed has been described in the previous chapters. The experiment scenario is described in Figure 5.13: one person is holding the system and taking a continuous walk along the trajectory shown with blue curve in the figure. The length of the trajectory is about 35 meters lasting approximately 32 seconds.

To estimate the pedestrian trajectory besides the system attitude, the INS/visual-gyro integrated system is further combined with the pedestrian dead reckoning algorithm presented in Chapter 4. The block diagram is shown in Figure 5.14.

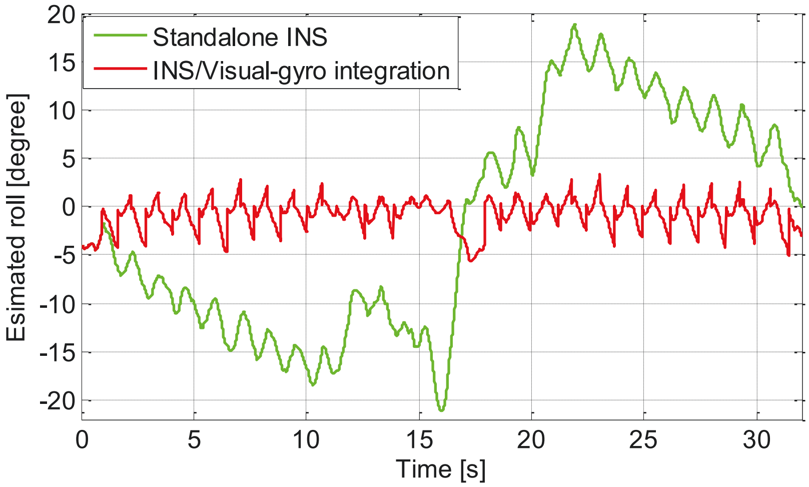


Figure 5.15: Attitude estimation result (roll)

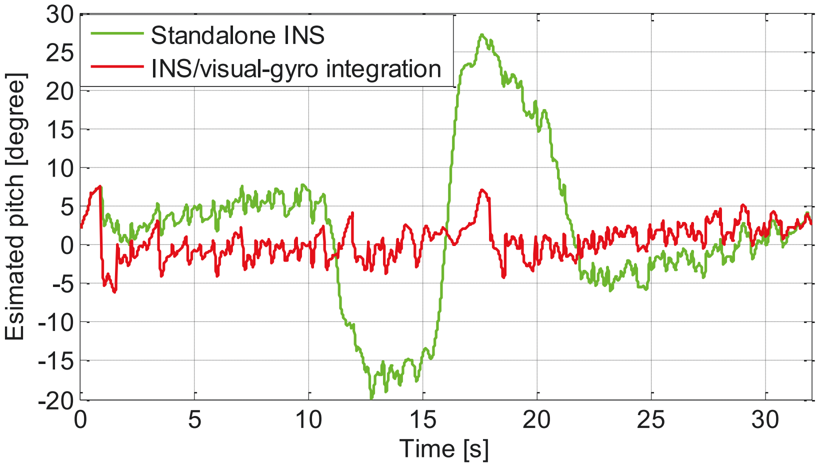


Figure 5.16: Attitude estimation result (pitch)

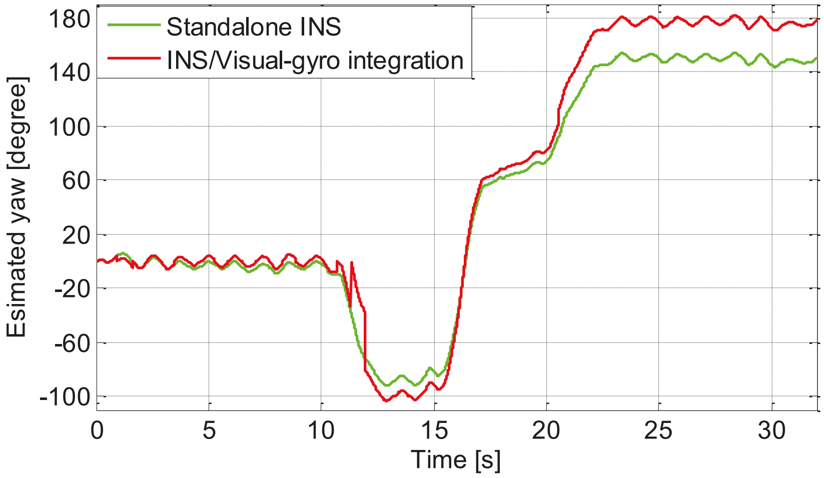


Figure 5.17: Attitude estimation result (yaw)

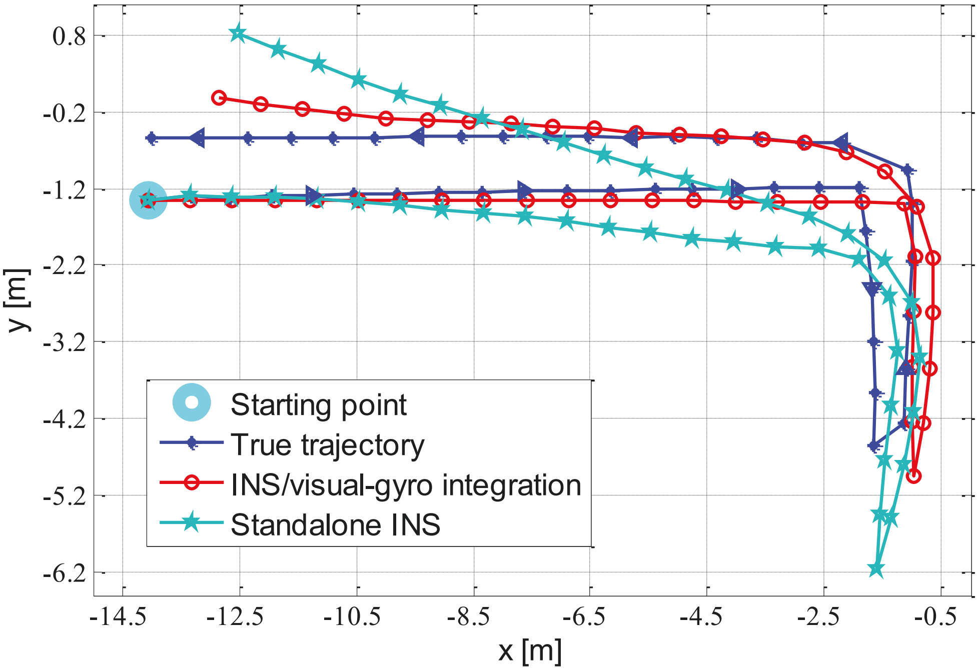


Figure 5.18: Trajectory estimation result

Figure 5.15, Figure 5.16 and Figure 5.17 show the estimation results of the roll, pitch and yaw respectively using standalone INS and the proposed INS/visual-gyro integration. It can be found that there are differences between the attitude estimations using the two methods. INS/visual-gyro integration is likely to provide more stable estimations of roll and pitch because the user is holding the hardware steadily in front of the body without significant tilts during the walking movement.

The test result is shown in Figure 5.18. The figure shows the estimated trajectories based on the attitude determination using the INS and INS/visual-gyro integration. From the figure, it can be found that the heading estimation using INS only suffers from the drift error. The integrated system can provide much better heading (yaw) estimation performance than the standalone INS. There are VP finding failures when the person is approaching the corner. In this case, the visual-gyro gaps are bridged with the INS.

5.4 Summary

In this chapter, indoor attitude estimation with integration of INS and visual gyroscope is discussed. The vanishing point (VP) based visual gyroscope and the VP detection are described. Unlike rate gyroscopes in IMUs, the visual gyroscope does not suffer from drift error propagations. But its availability highly depends on the indoor environment and its performance for fast rotation is limited. To overcome the limitations of INS and visual gyroscope, a DCM based integration of both systems is presented. One turn-table test and one pedestrian experiment have been carried out. Numerical results show that the INS/visual-gyro integration can estimate gyro bias and yields a better attitude estimation performance compared to the standalone systems.

Chapter 6
Summary and Conclusions

6.1 Summary

In this work, solutions for low-cost indoor vehicle and pedestrian navigation have been presented. The absolute localization solution is provided by the Wi-Fi received signal strength (RSS) based positioning and the dead reckoning solution is given by the inertial navigation system. Making use of the complementary nature of both systems, the integration of them is expected to yield a better navigation performance than the standalone systems. This work has explored the Wi-Fi based positioning, IMU based dead reckoning (DR) and DR/Wi-Fi integration. The corresponding enhancements and adapted solutions for vehicle and pedestrian applications have been described. For indoor attitude estimation, the integration of INS and camera based visual gyroscope using DCM modelling has been presented.

The Wi-Fi localization methods discussed in this work include the one directly using radio propagation model and the ones employing RSS fingerprinting. The fingerprinting approach consists of two steps: database building phase and localization phase. The database can be built either by collecting real in-situ measurements or by using the radio propagation model. The former is called empirical method while the latter is called propagation model based method. The fingerprinting localization phase can be done with nearest neighbor method or kernel based method. In this work, these Wi-Fi based localization approaches have been explored. One experiment was performed, and the results showed: 1) the approach directly using the propagation model yields higher positioning errors than the fingerprinting approaches; 2) the empirical fingerprinting provides a better positioning performance than the model based fingerprinting; 3) kernel based localization method yields lower positioning errors comparing to the nearest neighbor method.

The indoor vehicle navigation in this work employs integration of INS and Wi-Fi based positioning. The system process model is provided by strap-down INS mechanization and the observation model is from Wi-Fi positioning. Depending on the different Wi-Fi positioning approaches used in the system, the structure of the integration can be either tightly coupled or loosely coupled. Due to the nonlinearities of the system models, the UKF is employed for the integration. To further improve the INS/Wi-Fi integration without hardware change, the enhancements using vehicle constraints and AKF algorithm have been presented. One field experiment was carried out and the results showed that the INS/Wi-Fi integration provides a better tracking performance compared to the standalone Wi-Fi positioning and the enhanced integration outperforms the one without the enhancements.

IMU based pedestrian dead reckoning (PDR) is used for personal navigation in this work. The PDR algorithm with a foot-mounted IMU has been studied. The zero speed of the IMU can be detected in the stance phase of the gait cycle during the user's walking. In this case, ZUPT and ZARU with EKF are applied to re-estimate the IMU biases and hence reduce the drift error of the dead reckoning system. The improvement was shown with real test results. To design a PDR algorithm for portal devices which can be arbitrarily placed on the user's body, the step detection method needs to be chosen adaptively for different sensor placement modes. Typical placement modes were introduced and classified based on measurement outputs of accelerometers and gyroscopes. Three classifiers, namely nearest neighbor, ANN and SVM, have been discussed and the corresponding classification results have been compared and analyzed. The adapted PDR was further combined with Wi-Fi based positioning with a KF and the experimental results showed the advantage of the PDR/Wi-Fi integration with respect to the standalone systems.

For indoor attitude estimation, the camera based visual gyroscope is employed. Three steps of vanishing point detection from an image, namely edge detection, line detection and vanishing point localization, have been described. Attitude estimation with detected vanishing points has been provided. Unlike rate gyroscopes in IMUs, the visual-gyro does not suffer from drift error propagations. But its availability highly depends on the indoor environment and its performance for fast rotation is limited by the low update rate. To overcome the limitations of INS and visual-gyro, a DCM based integration of both systems has been presented. One turn-table test and one pedestrian experiment were carried out. Numerical results showed that the INS/visual-gyro integration yields a better attitude estimation performance compared to the standalone systems.

6.2 Conclusions

The main contributions of this work are listed as follows:

1. Different Wi-Fi based indoor localization approaches have been explored. For continuous navigation and tracking applications, Wi-Fi based positioning methods

have been integrated with INS with either loosely-coupled structure or tightly-coupled structure. This contribution has been published in [93] and [94].

2. To further improve the INS/Wi-Fi integration without hardware change for indoor vehicle navigation, the enhancements using vehicle constraints and adaptive Kalman filtering algorithm have been presented. The vehicle constraints include body velocity constraint, constant height constraint and body angular velocity constraint. The constraints have been converted to pseudo-measurements for the system observation model. This contribution has been published in [95] and [96].

3. IMU based pedestrian dead reckoning, including step detection, stride length estimation and heading determination has been explored. Zero velocity update and Zero angular rate update have been employed for foot-mounted systems to re-estimate the IMU bias terms. To yield a better navigation performance, the pedestrian dead reckoning has been integrated with Wi-Fi based positioning. This contribution has been published in [97] and [98].

4. Camera based visual gyroscope using vanishing point has been studied for indoor attitude determination. A DCM based INS/visual-gyro integration has been proposed. DCM system modelling was derived, which can reduce the system nonlinearity and avoid singularity problems. This contribution has been published in [99]. Using the iterative closest point (ICP) algorithm and depth visual gyroscope, the depth data from an RGB-D camera can be converted into positioning and attitude updates respectively, which have been further employed as additional system measurements for object tracking. This contribution has been published in [100] and [101].

Appendix A
Artificial Neural Networks

Neural networks, also named artificial neural networks (ANN), are mathematical model derived from biological neural networks. Commonly they consist of simple processing units, the neurons. Each of them receives one or more inputs (representing one or more dendrites) and sums them to produce an output (representing a biological neuron's axon), which works in a similar way to biological neurons [58]. In most cases a neural network is an adaptive system that adjusts its structure continually during the training phase, thus it is very efficient to use ANN model complex relationships between inputs and outputs or to find patterns in data.

Except neurons, one more basic component of a neural network is the weighted connection between the neurons. Here, the strength of a connection (or the connecting weight) from input x_i is referred to ω_{ki} . Data are transferred between neurons via connections with the connecting weight being either excitatory or inhibitory. The general structure of a neuron is displayed in Figure A.1.

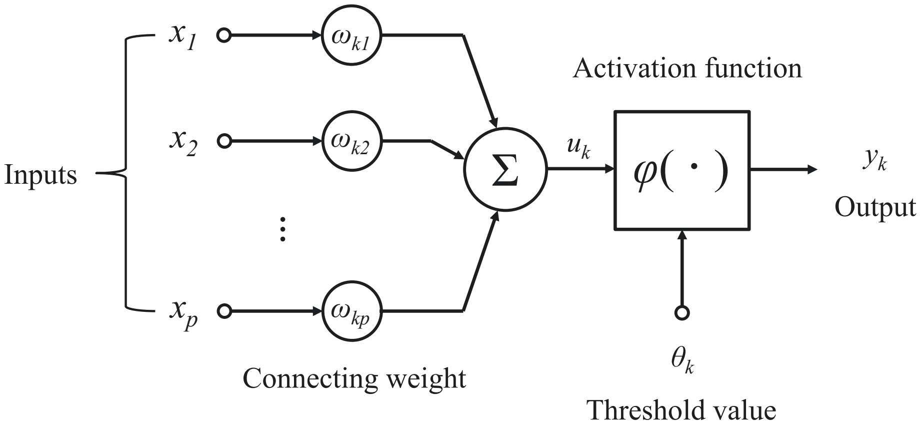


Figure A.1: Schematic drawing of k^{th} neuron [58]

The output y_k of k^{th} neuron is:

$$y_k = \varphi\left(\sum_{i=1}^{p} \omega_{ki} \cdot x_i\right) \tag{A.1}$$

Based on the model of nature, every neuron is, to a certain extent, at all times active, or excited. The reactions of the neurons to the input values depend on this activation state.

The activation state indicates the extent of a neuron's activation and is often referred as α_k, which is y_k in Equation (A.1).

At a certain moment, the activation α_k of a neuron *k* depends on the previous activation state of the neuron and the external input. It is defined by the activation function:

$$\begin{aligned} &\alpha_k(t) = \varphi\left(u_k(t), \alpha_k(t-1), \theta_k\right) \\ &\text{where } u_k(t) = \sum_{i=1}^{p} \omega_{ki} \cdot x_i(t) \end{aligned} \qquad \text{(A.2)}$$

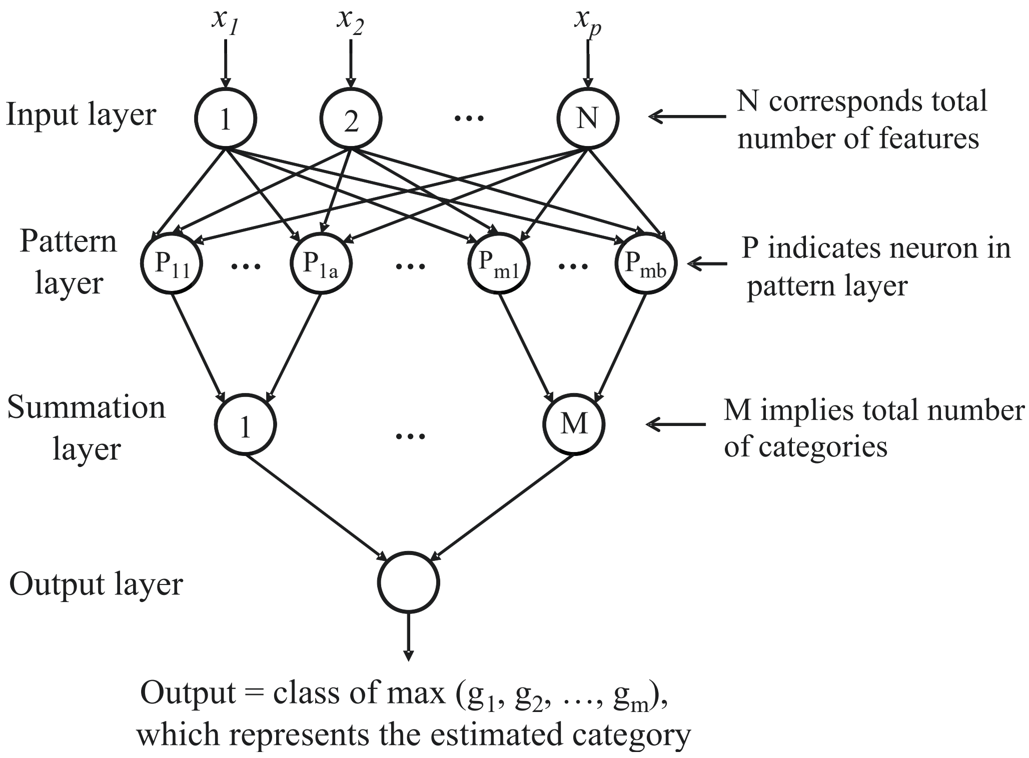
It transforms the network input $u_k(t)$, as well as the previous activation state $\alpha_k(t-1)$ into a new activation state $\alpha_k(t)$, with the threshold value playing an important role, as mentioned before. Near the threshold value, the activation function of a neuron reacts particularly sensitive. From the biological point of view, the threshold value represents the threshold at which a neuron starts firing. Neurons get activated if the network input exceeds their threshold value.

Unlike the other variables within the neural network, the activation function is often defined globally for all neurons or at least for a set of neurons and only the threshold values are different for each neuron. The activation function is also called transfer function. Common activation function includes Heaviside function, sigmoid function and hyperbolic tangent function. In case of Heaviside function, if the input is above a certain threshold (zero in the figure), the function changes from one value to another, but otherwise remains constant. In case of the sigmoid function, the function maps to the range of values of [0, 1] and the hyperbolic tangent function, maps to [-1, 1].

There are various sorts of artificial neural networks like back-propagation neural network and Hopfield network. The only one concerned in this work is the probabilistic neural network (PNN). A PNN is a feed-forward neural network, which was introduced by Donald F. Specht in 1990 [59]. It is predominantly a classifier that maps input patterns to a quantity of classifications [60]. Yet it can be also forced into a more general function approximator. A PNN is an implementation of a statistical algorithm called kernel discriminant analysis in which the operations are organized into a multilayered network with four layers: (a) input layer; (b) pattern layer; (c) summation layer; (d) output layer.

When an input is presented, the pattern layer produces a vector whose elements indicate how close the input is to the vectors of the training set. These elements are multiplied, element by element, by the bias and sent to the radial basis transfer function. An input vector close to a training vector is represented by a number close to one in the output vector. If an input is close to several training vectors of a single class, it is represented by several elements of the output vector that are close to "1".

The summation layer calculates the average of the output of the probability density function (PDF) for all samples in each single population. Finally, the competitive transfer function produces a "1" corresponding to the largest element of the input vector and a "0" elsewhere. Thus, the network classifies the input vector into a specific *i* class because that class has the maximum probability of being correct. [61]



x_1
x_2
x_p
Input layer
1
2
...
N
N corresponds total number of features
Pattern layer
P_{11}
P_{1a}
P_{m1}
P_{mb}
P indicates neuron in pattern layer
Summation layer
1
M
M implies total number of categories
Output layer
Output = class of max (g_1, g_2, …, g_m), which represents the estimated category

Figure A.2: Theory and architecture of PNN [59]

The PDF for a single population $g_i(\mathbf{x})$ has the form (the weighting function is assumed Gaussian function):

$$g_i(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{p}{2}} \sigma^p n_i} \sum_{k=1}^{n_i} e^{-\frac{\|\mathbf{x}-\mathbf{x}_{ik}\|^2}{2\sigma^2}} \tag{A.3}$$

where p implies the dimension of input vector, σ is the smoothing parameter, n_i indicates the amount of the samples in the i^{th} population, $\mathbf{x}_{ik}$ is the k^{th} training sample in the i^{th} population. The classification criteria of competitive transfer function:

$$g_i\left(\mathbf{x}\right) > g_j\left(\mathbf{x}\right), \text{ for all } j \neq i \tag{A.4}$$

PNN yields a good performance with short training time and high precision in classification if the training set is representative and the large memory requirement can be tolerated.

Appendix B
Support vector machine

A support vector machine is one of the famous supervised learning algorithms that analyze data and recognize patterns, used for both classification and regression analysis [62]. The theoretic foundation of SVM is structural risk minimization (SRM).

Assuming that a model is derived from a set of samples (training data) using an arbitrary learning algorithm, a classification result is obtained based on the model tested by that training data set. Further assuming that the result is 96% correct, and then this incorrect partition 4% is called empirical risk. In many previous research works, it is always considered to minimize this value for an optimal solution, but it is later found completely wrong.

In the training phase, the empirical risk can be easily minimized to zero, but the obtained model cannot deal with any new sample that is different to each one of the training samples, because it overfits the training data, or in other words, it is short of generalization ability.

The precondition of successfully using empirical risk minimization (ERM) is: the training data set must be perfect, which includes almost all possible samples. However, the size of samples used for training is never large enough in reality. As a result, the definitions of the confidence interval (CI) and the confidence level are inducted. CI is a parameter that describes the reliability of an estimate. The probability that the result is contained in this interval is called confidence level. And the previously mentioned SRM is composed of ERM and CI. The true risk minimization is defined as

$$R(\omega) \leq R_{\text{emp}}(\omega) + \Phi(n/h) \tag{B.1}$$

where $R(\omega)$ is the true risk, $R_{\text{emp}}(\omega)$ is the empirical risk and $\Phi(n/h)$ is the CI, where n implies the size of training data set and h indicates the Vapnik Chervonenkis (VC) dimension, which describes the capacity of the trained model. The capacity of a model can be simply interpreted as the complexity of the learning machine. The larger h is, the more complicated the model would be, and the lower confident the model is, to make a correct prediction. $\Phi(n/h)$ is a monotone decreasing function, thus it can be checked that the smaller h is, the closer the confidence interval to zero lies. Thus the new strategy becomes to keep the $R_{\mathrm{emp}}(\omega)$ fixed and minimize the confidence interval. And the actual risk becomes closer to the empirical risk.

An SVM model is a representation of the examples as points in space so that the examples of the separate categories are divided by a clear gap that is as wide as possible so that the error is minimized. In the center of the gap lies the boundary of different categories: hyperplane [63]. New examples are then mapped into that same space and predicted to belong to a category based on which side of the hyperplane they fall on [62]. A hyperplane is an n-dimensional plane. In case of 2-dimensional space, it is simply a line (see Figure B.1). The hyperplane can be characterized by a linear function:

$$\mathbf{w}^T\mathbf{x}+b=\mathbf{0} \tag{B.2}$$

where **w** is the weight vector and **x** is the feature vector.

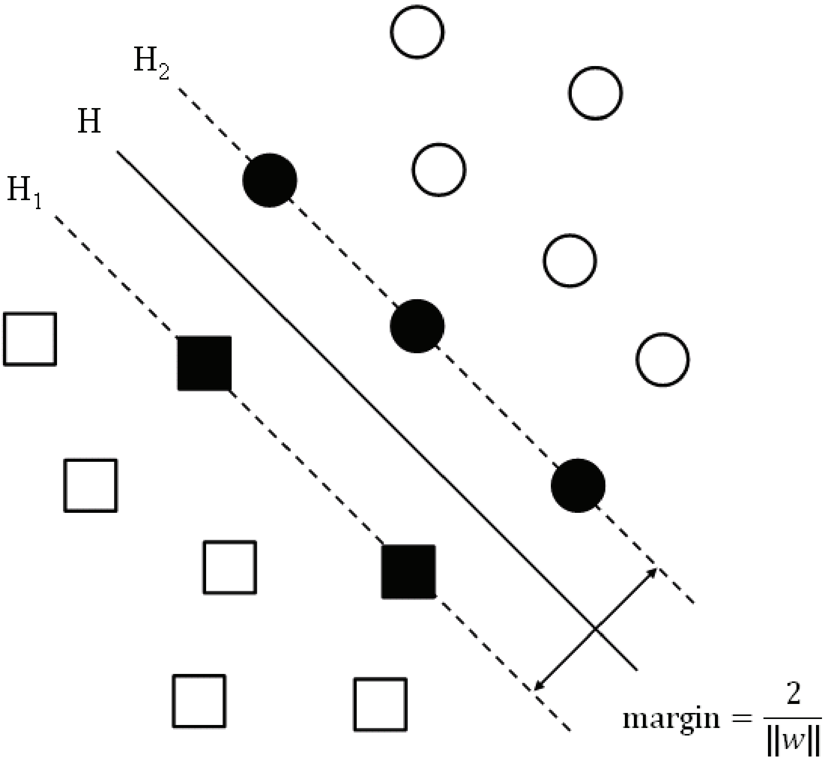


Figure B.1: Linear classifier: optimal hyperplane in linearly separable case

As in the case of KNN or ANN, the position of the hyperplane would be optimized with respect to all samples, which leads to a bad effort in some cases. Because it is possible that the distance between two categories are reduced when the hyperplane is calculated based on all samples, within it some samples are located very far to the hyperplane and minimizing their distances to the hyperplane makes no sense at all. In contrast to other classifiers, SVM considers only the distances between the hyperplane and those samples, which are closest to the hyperplane. Those closest samples are called support vectors. Such kind of hyperplane is called optimal hyperplane (or the maximal margin hyperplane, because it separates the data with maximal margin). An example of interpretation is illustrated in Figure B.1.

The circles and squares represent respectively two categories. H implies the aforementioned optimal hyperplane. H1 and H2 are two parallel hyperplanes which contain the closest samples to H. And those closest samples, denoted by filled circles and squares, are support vectors.

To describe the separating hyperplane the definition of functional margin is given by

$$\gamma^{(i)} = y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right) \tag{B.3}$$

where $\gamma^{(i)}$ implies the functional margin (i.e., the functional distance to the separating hyper-plane $\mathbf{w}^T\mathbf{x}+b=0$) of ith sample, $y^{(i)}$ denotes the label (i.e., "1" or "-1") of ith sample. The separating hyper-plane separates the data using the criteria:

$$y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right) \geq \gamma,\ i = 1,\ldots,m \quad \text{with} \quad \gamma = \min_{i=1,\ldots,m} \gamma^{(i)} \tag{B.4}$$

where γ indicates the minimal functional margin among all training samples.

For a certain sample $x^{(i)}$ the functional margin can vary widely if the vector **w** and parameter *b* increase or decrease synchronously. To seek a unique specified pair of **w** and *b*, the functional margin is to be normalized. It is the so-called geometric margin, defined as:

$$\gamma_{\mathrm{g}}^{(i)} = y^{(i)}\left(\left(\frac{\mathbf{w}}{\|\mathbf{w}\|}\right)^T \mathbf{x}^{(i)} + \frac{b}{\|\mathbf{w}\|}\right) \quad \text{with} \quad \gamma_{\mathrm{g}} = \min_{i=1,\ldots,m} \gamma_{\mathrm{g}}^{(i)} \tag{B.5}$$

The minimal geometric margin γ_{g} is shown in Equation (B.5) as the distance between H and H$_i$.

As mentioned previously, the SVM algorithm concerns only the distance between the separating hyperplane and the support vectors. The larger this distance becomes, the more reliable the predictions given by the model are. Thus the problem becomes:

$$\begin{aligned}&\max_{\gamma,\mathbf{w},b}\ \gamma_g\\&\quad\text{s.t. } y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)}+b\right)\geq\gamma,\ i=1,...,m\end{aligned}\tag{B.6}$$

γ_g is given as $\frac{\gamma}{\|\mathbf{w}\|}$. To simplify the problem the functional margin γ is defined as one. And the object function is then transformed to $\frac{1}{\|\mathbf{w}\|}$. It is equivalent to find the maximum of $\frac{1}{\|\mathbf{w}\|}$ and the minimum of $\|\mathbf{w}\|$. But in order to simplify the calculation the object function $\|\mathbf{w}\|$ is replaced by $\frac{1}{2}\|\mathbf{w}\|^2$, with which the position of the minimum is not moved. So the revised formulation of the problem is:

$$\begin{aligned}&\min_{\gamma,\mathbf{w},b}\ \frac{1}{2}\|\mathbf{w}\|^2\\&\quad\text{s.t. } y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)}+b\right)\geq 1,\ i=1,...,m\end{aligned}\tag{B.7}$$

This is a typical quadratic programming problem. It can be solved using the Lagrangian function. Here the calculation procedure is omitted due to the limitation of space. And **w** can be formulated as

$$\mathbf{w}=\sum_{i=1}^{m}\alpha_i y^{(i)}\mathbf{x}^{(i)}\tag{B.8}$$

where α_i is the so-called Lagrange multiplier.

The discriminant can be reformulated as

$$\begin{aligned}g(\mathbf{x})&=\mathbf{w}^T\mathbf{x}+b\\&=\left(\sum_{i=1}^{m}\alpha_i y^{(i)}\mathbf{x}^{(i)}\right)^T\mathbf{x}+b\\&=\sum_{i=1}^{m}\alpha_i y^{(i)}\left\langle\mathbf{x}^{(i)},\mathbf{x}\right\rangle+b\end{aligned}\tag{B.9}$$

where $\left\langle\mathbf{x}^{(i)},\mathbf{x}\right\rangle$ implies the inner product of $\mathbf{x}^{(i)}$ and $\mathbf{x}$.

An example to show the usefulness of kernel function is given as follows.

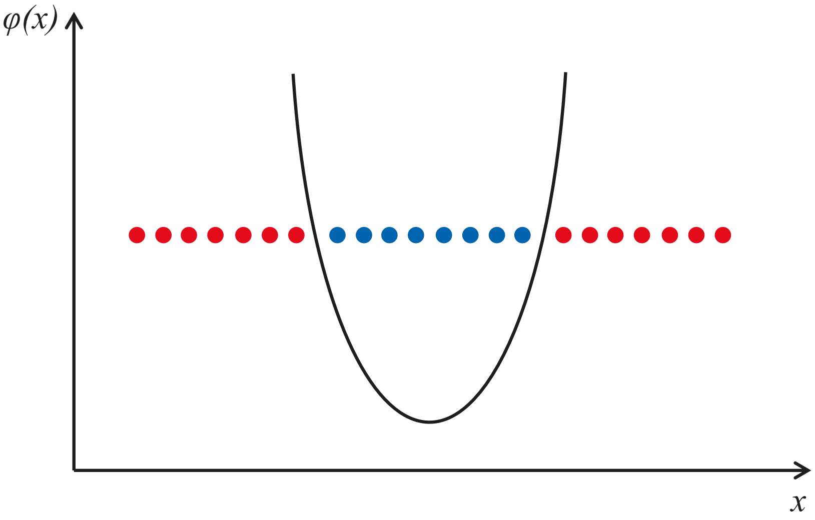


Figure B.2: Case that the samples are not linearly separable

The red dots and the blue dots denote class one and class two respectively. As it shows, in the 2-dimensional case it is impossible to completely divide these two sets of dots into two parts using a single line. But they can be easily separated by a parabola, which is not a linear function, described by:

$$g(x) = c_0 + c_1 x + c_2 x^2 \tag{B.10}$$

The new sample with $g(x) > 0$ will be categorized into class two (denoted by blue dots), and if not, it will be classified as class one (denoted by red dots). Here (2.26) can be rewritten in vectors form as the following expression:

$$g(x) = \begin{bmatrix} c_0 & c_1 & c_2 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} + 0 = \tilde{\mathbf{w}}^T \phi(x) + b \qquad \text{with} \quad \tilde{\mathbf{w}} = \sum_{i=1}^{m} \alpha_i y^{(i)} \phi\left(x^{(i)}\right) \tag{B.11}$$

The process of $x \rightarrow \phi(x)$ is called feature mapping. The mapping function $\phi(x)$, which is $\phi(x) = [1\ x\ x^2]^T$ in this example, maps the features from low dimensional space to high dimensional space. The adapted Equation (B.11) is obviously a linear function regarding $\phi(x)$. Thus via mapping function, those samples, which are formerly not linearly separable in low dimensional feature space, become now linearly separable in high dimensional feature space. Accordingly, the Equation (B.9) can be revised as

$$\begin{aligned} g(\mathbf{x}) &= \tilde{\mathbf{w}}^T \phi(\mathbf{x}) + b \\ &= \left(\sum_{i=1}^{m} \alpha_i y^{(i)} \phi\left(\mathbf{x}^{(i)}\right) \right)^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^{m} \alpha_i y^{(i)} \left\langle \phi\left(\mathbf{x}^{(i)}\right), \phi(\mathbf{x}) \right\rangle + b \end{aligned} \tag{B.12}$$

In some complicated cases, the mapping function $\phi(x)$ is very difficult to determine, and in addition, even it may be computationally intractable to calculate the inner product of two extra high dimensional feature vectors. Fortunately, the kernel trick can be used to calculate the inner product in high dimensional feature space without the need for explicit evaluation of the mapping function $\phi(x)$. That is to say, the kernel trick allows the replacement of inner products in high dimensional feature space by a kernel evaluation on the low dimensional input vectors [16].

And the discriminant is now updated to:

$$\begin{aligned} g(\mathbf{x}) &= \tilde{\mathbf{w}}^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^{m} \alpha_i y^{(i)} K\left(\mathbf{x}^{(i)}, \mathbf{x}\right) + b \end{aligned} \tag{B.13}$$

Gaussian kernel Mercer's Theorem guarantees the correspondence between a kernel function and an inner product in a feature space $\mathscr{F}$, given that the kernel is a positive definite function. The Gaussian kernel is employed for the SVM classifier in this work. The kernel function has been introduced with Equation (2.13) in Section 2.3.2.

Nevertheless, it is possible that the samples are still non-separable after the feature mapping. In this case the optimal hyperplane to perfectly divide these two categories doesn't exist anymore. To solve this problem, V. Vapnik presented an altered maximum margin idea that allows the existence of mislabeled samples, which is the so-called soft margin [64].

The modified optimization problem becomes:

$$\begin{aligned} &\min_{\gamma,\mathbf{w},b} \ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m} \xi_i \\ &\text{s.t.} \ \ y^{(i)}\left(\mathbf{w}^T \mathbf{x}^{(i)} + b\right) \geq 1 - \xi_i,\ i = 1,\dots,m \\ &\qquad \xi_i \geq 0,\ i = 1,\dots,m \end{aligned} \tag{B.14}$$

where ξ is called slack variable, C is the corresponding penalty factor.

The value of the slack variable indicates how far the mislabeled sample strays from the majority. The value of penalty factor determines how acceptable the occurrence of a mislabeled sample is. The larger *C* is, the lower tolerance of error is (i.e., the stronger influence the mislabeled samples have on finding the optimal hyperplane).

To sum up, an SVM is a kernel-based soft-margin-supporting linear classifier, which is dominant in binary classification problems. For an *M* class classification, M binary SVM classifiers are created. Each classifier is trained to discriminate one class from the remaining *M*-1 classes. During the testing or application phase, data are classified by computing the margin from the linear separating hyperplane. Data are assigned to the class labels of the SVM classifiers that produce the maximal output.

Appendix C
Unscented Kalman Filtering

The extended Kalman filter (EKF) has been the standard approach for state estimation of nonlinear system models over the past decades. The principal feature of the EKF is a linearization of the system equation and the measurement equation around the previous estimate or the current prediction. The linearized system is then represented by the Jacobian transformations of the nonlinear process and measurement functions. The normal KF formulae are then applied to the linearized system. However, the procedure produces sub-optimal estimates of the state of the system [32]. In some applications, the first-order term may be insufficiently accurate to approximate the nonlinearities of the system, especially if the system has larger nonlinearities and the higher order terms are non-negligible.

The unscented Kalman filter (UKF) was developed to overcome the limitations of the EKF. Distinguishing itself from the standard KF, the UKF calculates the filtering parameters by using a set of sigma points, which can be directly mapped into the nonlinear functions of the system, instead of linearization via the Jacobian matrices. The parameters derived from the sigma points include the UKF gain matrix, the state prediction and its covariance, the measurement prediction and its covariance, as well as the estimated covariance. The UKF is expected to give a better approximation to a nonlinear system because it is easier to approximate an arbitrary nonlinear function or transformation. In comparison with the EKF, the UKF usually has faster convergence, especially when the initial conditions of the filter states are too far from ‘truth’ [33].

The underlying intuition of the UKF is that, with a fixed number of parameters, it should be easier to approximate a Gaussian distribution than it is to approximate an arbitrary nonlinear function or transformation. The state distribution is again represented by a Gaussian random variable, but is now specified using a minimal set of carefully chosen sample points, called the sigma points (SPs). SPs capture the true mean and covariance of the PDF and, when propagated through the true nonlinear system, capture the transformed mean and covariance accurately up to the third order for any nonlinearity [34].

The derivation of the unscented transformation

1. Beginning with an n-element vector $\mathbf{x}$ with known mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P}$, a known nonlinear transformation $\mathbf{y} = \mathbf{h(x)}$ is given. The mean and covariance of $\mathbf{y}$ is denoted as $\bar{\mathbf{y}}$ and $\mathbf{P}^{\mathrm{y}}$.

2. Suppose **x** is an $n \times 1$ vector, chose $2n$ SP vectors $\mathbf{x}^{(i)}$ as follows:

$$\begin{aligned} \mathbf{x}^{(i)} &= \bar{\mathbf{x}} + \tilde{\mathbf{x}}^{(i)} \qquad i = 1,2,...,2n \\ \tilde{\mathbf{x}}^{(i)} &= \left(\sqrt{n\mathbf{P}}\right)_i^T \qquad i = 1,2,...,n \\ \tilde{\mathbf{x}}^{(n+i)} &= -\left(\sqrt{n\mathbf{P}}\right)_i^T \qquad i = 1,2,...,n \end{aligned} \tag{C.1}$$

where $\sqrt{n\mathbf{P}}$ is the matrix square root of $n\mathbf{P}$ such that $\left(\sqrt{n\mathbf{P}}\right)^T \sqrt{n\mathbf{P}} = n\mathbf{P}$, and $\left(\sqrt{n\mathbf{P}}\right)_i$ is the i th row of $\sqrt{n\mathbf{P}}$.

3. Transform the sigma points as follows:

$$\mathbf{y}^{(i)} = \mathbf{h}(\mathbf{x}^{(i)}) \quad i = 1,2,...,2n \tag{C.2}$$

4. Approximate the mean and covariance of $\mathbf{y}$ as follows:

$$\begin{aligned} \bar{\mathbf{y}} &= \frac{1}{2n}\sum_{i=1}^{2n}\mathbf{y}^{(i)} \\ \mathbf{P}^{\mathrm{y}} &= \frac{1}{2n}\sum_{i=1}^{2n}\left(\mathbf{y}^{(i)} - \bar{\mathbf{y}}\right)\left(\mathbf{y}^{(i)} - \bar{\mathbf{y}}\right)^T \end{aligned} \tag{C.3}$$

The derivation of the UKF

Suppose the system model is

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{w}_{k-1} \\ \mathbf{y}_k &= \mathbf{h}(\mathbf{x}_k) + \boldsymbol{\eta}_k \\ \mathbf{w}_k &\sim N\left(\mathbf{0}, \mathbf{Q}_k\right) \\ \boldsymbol{\eta}_k &\sim N\left(\mathbf{0}, \mathbf{R}_k\right) \end{aligned} \tag{C.4}$$

1. To propagate system state from time step $k-1$ to k, sigma points $\mathbf{x}_{k-1}^{(i)}$ are firstly chosen, with appropriate changes since the current best guess for the mean and covariance of $\mathbf{x}_k$ are $\hat{\mathbf{x}}_{k-1}^{+}$, and $\mathbf{P}_{k-1}^{+}$.

$$\begin{aligned}
&\hat{\mathbf{x}}_{k-1}^{(i)} = \hat{\mathbf{x}}_{k-1}^{+} + \tilde{\mathbf{x}}^{(i)} \qquad i = 1,2,...,2n \\
&\tilde{\mathbf{x}}^{(i)} = \left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T \qquad i = 1,2,...,n \\
&\tilde{\mathbf{x}}^{(n+i)} = -\left(\sqrt{n\mathbf{P}_{k-1}^{+}}\right)_i^T \qquad i = 1,2,...,n
\end{aligned} \tag{C.5}$$

2. Use the known nonlinear system equation $\mathbf{f}(\cdot)$ to transform the sigma points into $\hat{\mathbf{x}}_k^{(i)}$.

$$\hat{\mathbf{x}}_k^{(i)} = \mathbf{f}(\hat{\mathbf{x}}_{k-1}^{(i)}) \tag{C.6}$$

3. Combine the $\hat{\mathbf{x}}_k^{(i)}$ vectors to obtain the priori state estimate at time k .

$$\hat{\mathbf{x}}_k^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\hat{\mathbf{x}}_k^{(i)} \tag{C.7}$$

4. Estimate the priori error covariance as shown in Equation (C.3). However, we should add $\mathbf{Q}_{k-1}$ to the end of the equation to take the process noise into account.

$$\mathbf{P}_k^{-} = \frac{1}{2n}\sum_{i=1}^{2n}\left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k^{-}\right)\left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k^{-}\right)^T + \mathbf{Q}_{k-1} \tag{C.8}$$

5. Now the KF time update equations are done. To derive the measurement update equations, the sigma points $\hat{\mathbf{x}}_k^{(i)}$ are chosen as:

$$\begin{aligned}
&\hat{\mathbf{x}}_k^{(i)} = \hat{\mathbf{x}}_k^{-} + \tilde{\mathbf{x}}^{(i)} \qquad i = 1,2,...,2n \\
&\tilde{\mathbf{x}}^{(i)} = \left(\sqrt{n\mathbf{P}_k^{-}}\right)_i^T \qquad i = 1,2,...,n \\
&\tilde{\mathbf{x}}^{(n+i)} = -\left(\sqrt{n\mathbf{P}_k^{-}}\right)_i^T \qquad i = 1,2,...,n
\end{aligned} \tag{C.9}$$

This step can be omitted if desired. That is, instead of generating new sigma points we can reuse the sigma points that are obtained from the time update. This will save computational effort if we are willing to sacrifice performance [32].

6. Use the known nonlinear system equation $\mathbf{h}(\cdot)$ to transform the sigma points into $\hat{\mathbf{y}}_k^{(i)}$ (predicted measurements) as follows:

$$\hat{\mathbf{y}}_k^{(i)} = \mathbf{h}(\hat{\mathbf{x}}_k^{(i)}) \tag{C.10}$$

7. Combine the $\hat{\mathbf{y}}_k^{(i)}$ to obtain the priori state estimate at time k .

$$\hat{\mathbf{y}}_k = \frac{1}{2n}\sum_{i=1}^{2n}\hat{\mathbf{y}}_k^{(i)} \tag{C.11}$$

8. Estimation of covariance of the predicted measurement can be done. However, we should add $\mathbf{R}_k$ to the end of the equation to take the measurement noise into account.

$$\mathbf{P}_k^{\mathbf{y}} = \frac{1}{2n}\sum_{i=1}^{2n}\left(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k\right)\left(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k\right)^T + \mathbf{R}_k \tag{C.12}$$

9. Estimate the cross covariance between $\hat{\mathbf{x}}_k^-$ and $\hat{\mathbf{y}}_k$ as:

$$\mathbf{P}_k^{\mathbf{xy}} = \frac{1}{2n}\sum_{i=1}^{2n}\left(\hat{\mathbf{x}}_k^{(i)} - \hat{\mathbf{x}}_k^-\right)\left(\hat{\mathbf{y}}_k^{(i)} - \hat{\mathbf{y}}_k\right)^T \tag{C.13}$$

10. The measurement update of the state estimate can be performed using the normal Kalman filter equations as follows:

$$\begin{aligned} \mathbf{K}_k &= \mathbf{P}_k^{\mathbf{xy}}\left(\mathbf{P}_k^{\mathbf{y}}\right)^{-1} \\ \hat{\mathbf{x}}_k^+ &= \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left(\mathbf{y}_k - \hat{\mathbf{y}}_k\right) \\ \mathbf{P}_k^+ &= \mathbf{P}_k^- - \mathbf{K}_k\mathbf{P}_k^{\mathbf{y}}\mathbf{K}_k^T \end{aligned} \tag{C.14}$$

Bibliography

[1] B. Paramvir and V. Padmanabhan, "RADAR: An In-Building RF-based User Location and Tracking System," *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2000

[2] R. Hansen, R. Wind, C. Jensen and B. Thomsen, "Algorithmic Strategies for Adapting to Environmental Changes in 802.11 Location Fingerprinting," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010

[3] B. Paramvir and V. Padmanabhan, "Enhancements to the RADAR User Location and Tracking System," *Technical Report, MSR-TR-200-12, Microsoft Research,* 2000

[4] IEEE 802.11 Working Group, "IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," *IEEE 802.11, ISBN: 978-0-7381-3044-6*, 1999

[5] M. Cypriani, F. Lassabe, P. Canalda and F. Spies, "Wi-Fi-Based Indoor Positioning: Basic Techniques, Hybrid Algorithms and Open Software Platform," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010

[6] A. R. Sandeep, Y. Shreyas, S. Seth, R. Agarwal, and G. Sadashivappa, "Wireless Network Visualization and Indoor Empirical Propagation Model for a Campus Wi-Fi Network," *World Academy of Science*, 2008

[7] A. Goldsmith, "Wireless Communications," *Cambridge Univ. Press, ISBN: 978-0521837163*, 2005

[8] X. Li and K. Pahlavan, "Super-Resolution TOA Estimation with Diversity for Indoor Geolocation," *IEEE Transactions on Wireless Communications, Vol. 3, No. 1, pp. 224-234*, 2004

[9] D. Niculescu and B. Nath, "VOR Base Stations for Indoor 802.11 Positioning," *Proc. MobiCom, pp. 58-69*, 2004

[10] A. Kushki, K. N. Plataniotis and A. N. Venetsanopoulos, "Kernel-Based Positioning in Wireless Local Area Networks," *IEEE Transactions on Mobile Computing, Vol. 6, No. 6, pp. 689-705*, June 2007

[11] A. Luo and G. Lei, "Indoor location detection using WLAN," *School of Information and Communication Technology, Royal Institute of Technology (KTH),* 2010

[12] J. Proakis, "Digital Communications," *3rd ed., McGraw–Hill Book*, 1995

[13] B. Sklar, "Rayleigh Fading Channels in Mobile Digital Communication Systems," *IEEE Communications Magazine Information,* 1997

[14] P. S. Kumar, M. G. Sumithra and M. Sarumathi, "Performance Comparison of Rayleigh and Rician Fading Channels In QAM Modulation Scheme Using Simulink Environment," *International Journal of Computational Engineering Research*, 2013

[15] K. El-Kafrawy, M. Youssef, A. El-Keyi and A. Naguib, "Propagation Modeling for

Accurate Indoor WLAN RSS-based Localization," *Vehicular Technology Conference Fall*, 2010

[16] J. Shawe-Taylor and N. Cristianini, "Kernel Methods for Pattern Analysis," *Cambridge University Press*, 2004

[17] S. Gansemer, S. Pueschel, R. Frackowiak, S. Hakobyan and U. Grossmann, "Improved RSSI-based Euclidean Distance Positioning Algorithm for Large and Dynamic WLAN Environments," *International Journal of Computing,* 2010

[18] B. W. Silverman, "Density Estimation for Statistics and Data Analysis," *Chapman and Hall*, 1986

[19] P. Addesso, L. Bruno and R. Restaino, "Adaptive Localization Techniques in WiFi Environments," *Fifth International Symposium on Wireless Pervasive Computing (ISWPC 2010)*, 2010

[20] M. Cypriani, F. Lassabe, P. Canalda and F. Spies, "Wi-Fi-Based Indoor Positioning: Basic Techniques, Hybrid Algorithms and Open Software Platform," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010

[21] Y. Thong, M. Woolfson, J. Crowe, B. Hayes-Gill, and R. Challis, "Dependence of inertial measurements of distance on accelerometer noise," *Measurement Science and Technology, vol. 13, no. 8, p. 1163*, 2002

[22] M. Grewal, L. Weill and A. Andrews, "Global Positioning Systems, Inertial Navigation and Integration," *John Wiley & Sons*, 2001

[23] E. Kaplan and C. Hegarty, "Understanding GPS: Principles and Applications," *Technical Artech House, Second Edition*, 2006

[24] E.-H. Shin, "Accuracy Improvement of Low Cost INS/GPS for Land Applications," *UCGE Reports Number 20156, The University of Calgary*, 2001

[25] J. Farrel, "Aided Navigation: GPS with High Rate Sensors," *McGraw-Hill New York*, 2008

[26] S. Sukkarieh, "Low Cost, High Integrity, Aided Inertial Navigation Systems for Autonomous Land Vehicles," *Ph.D. Dissertation, University of Sydney*, 2000

[27] S. Panich and N. Afzulpurkar, "Absolute Positioning Instruments for Odometry System Integrated with Gyroscope by Using IKF," *Global Journal of Researches in Engineer*, 2010

[28] M. Ibrabeem, "Gyroscope-Enhanced Dead Reckoning Localization System For an Intelligent Walker," *International Conference on Information, Networking and Automation (ICINA)*, 2010

[29] P. Zhang, J. Gu, E. Milios, P. Huynh, "Navigation with IMU/GPS/Digital Compass with Unscented Kalman Filter," *Proceeding of the IEEE international Conference on Mechatronics & Automation Niagara Falls*, 2005

[30] D. Titterton and J. Weston, "Strapdown Inertial Navigation Technology," 2nd ed., *Institution of Engineering and Technology*, 2004

[31] J. Zhou, S. Knedlik, E. Edwan, and O. Loffeld, "Low-cost INS/GPS with Nonlinear Filtering Methods," *IET Fusion*, 2010

[32] D. Simon, "Optimal State Estimation: Kalman, Hinfinity, and Nonlinear Approaches," *John Wiley and Sons*, 2006

[33] E. A. Wan and R. van der Merwe, "The unscented Kalman filter for nonlinear estimation," *Symposium on Adaptive Systems for Signal Processing, Communication and Control, IEEE Press*, 2006

[34] P. Maybeck, "Stochastic models, estimation, and control," *Academic Press*, 1982

[35] O. Loffeld, "Estimationstheorie I + II," *Oldenbourg Verlag München,* 1990

[36] A. Mohamed, K. Schwarz, Adaptive Kalman filtering for INS/GPS, *Journal of Geodesy, 73: 193-203*, 1999

[37] C. Hide, T. Moore, and M. Smith, "Adaptive Kalman Filtering for Low-cost INS/GPS," *Journal of Navigation, vol. 56, pp. 143-152*, Jan 2003

[38] I. Klein, S. Filin and T. Toledo, "Pseudo-measurements as aiding to INS during GPS outages," *Journal of the Institute of Navigation 57: 25-34*, 2010

[39] I. Klein, S. Filin and T. Toledo, "Vehicle constraints enhancement for supporting INS navigation in urban environments," *Journal of the Institute of Navigation 58: 7-15*, 2011

[40] Xsens, "MTi and MTx User Manual and Technical Documentation," *Xsens Techologies*, 2009

[41] S. Godha and G. Lachapelle, "Integrated GPS/INS system for Pedestrian Navigation in a Signal Degraded Environment," *Proceedings of the 19th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS)*, 2006

[42] S. Cho and C. Park, "MEMS Based Pedestrian Navigation System," *The Journal of Navigation, vol. 59, no. 01, pp. 135–153*, 2006

[43] S. Godha and G. Lachapelle, "Foot mounted inertial system for pedestrian navigation," *Measurement Science and Technology*, 2008

[44] N. Castaneda and S. Lamy-Perbal, "An improved shoe-mounted inertial navigation system," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010

[45] J. Nilsson, I. Skog and P. Haendel, "Tuning and Calibration of Foot-mounted ZUPT Aided INS," *Royal Institute of Technology (KTH) report*, 2010

[46] Z. F. Syed, "Design and Implementation Issues of a Portable Navigation System," *UCGE Report 20288, University of Calgary*, 2009

[47] X. Zhao, S. Saeedi, N. EI-Sheimy, Z. Syed and C. Goodall, "Towards Arbitrary Placement of Multi-sensors Assisted Mobile Navigation System," *International Technical Meeting of the Satellite Division of The Institute of Navigation*, 2010

[48] X. Zhao, C. Goodall, Z. Syed, B. Wright and N. El-Sheimy, "Wi-Fi Assisted Multi-sensor Personal Navigation System for Indoor Environments," *ION 2010*

International Technical Meeting (ITM 2010), 2010

[49] R. Chen, L. Pei, T. Kroger, H. Kuusniemi, J. Liu and W. Chen, "Knowledge-based error detection and correction method of a Multi-sensor Multi-network positioning platform for pedestrian indoor navigation," *IEEE/ION Position Location and Navigation Symposium (PLANS)*, 2010

[50] J. Kim, H. Jang, D. Hwang, and C. Park, "A step, stride and heading determination for the Pedestrian Navigation System," *Journal of Global Positioning Systems, vol. 3, pp. 273–279*, 2010

[51] D. Roetenberg, H. Luinge, and P. Veltink, "Inertial and magnetic sensing of human movement near ferromagnetic materials," *in ISMAR '03: Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2003

[52] W. Zijlstra and A. L. Hof, "Displacement of the pelvis during human walking: experimental data and model predictions," *Gait & Posture, vol. 6, no. 3, pp. 249 – 262*, 1997

[53] D. Gusenbauer, C. Isert, and J. Kroandsche, "Self-contained indoor positioning on off-the-shelf mobile devices," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010

[54] D. L. Olson and D. Delen, "Advanced Data Mining Techniques," *1st Edition, Springer*, 2008

[55] L. E. Peterson, "K-Nearest Neighbor," *scholarpedia.org*, 2009

[56] T. Cover and P. Hart, "Nearest neighbour pattern classification," *IEEE Transactions on Information Theory 13 (1): 21–27*, 1967

[57] N. Mulla and Sheetal Girase, "A New Approach to Requirement Elicitation Based on Stakeholder Recommendation and Collaborative Filtering," *International Journal of Software Engineering & Applications, Vol. 3, No. 3*, 2012

[58] D. Kriesel, "A Brief Introduction to Neural Networks," *http://www.dkriesel.com*, 2007

[59] D. F. Specht, "Probabilistic Neural Networks," *Neural Networks, Vol. 3, pp. 109-118*, 1990

[60] M. Moghavvemi and S. Yang, "ANN Application Techniques for Power System Stability Estimation," *Electric Machines & Power Systems, 28(2), 167-178*, 2000

[61] P. D. Wassermann, "Advanced Methods in Neural Computing," *1st Edition, Van Nostrand Reinhold, pp. 35-55*, 1993

[62] K. Parasuraman and P. Subin, "SVM Based License Plate Recognition System," *IEEE International Conference on Computational Intelligence and Computing Research*, 2010

[63] G. Foody and A. Mathur, "A relative evaluation of multiclass image classification by support vector machines," *IEEE Transactions on Geoscience and Remote Sensing, vol.42, no.6, pp. 1335-1343*, 2004

[64] V. Vapnik, "The Nature of Statistical Learning Theory," *2nd Edition, Springer*, 2000

[65] P. Corke, J. Lobo, and J. Dias, "An introduction to inertial and visual sensing," *The International Journal of Robotics Research,* 2007

[66] M. Sonka , V. Hlavac, R. Boyle, "Image Processing, Analysis and Machine Vision, third edition," *Stamford, Connecticut: Cengage Learning*, 2008

[67] G. Klein and T. Drummond, "A Single-Frame Visual Gyroscope," *Proceedings of British Machine Vision Conference (BMVC)*, 2005

[68] V. Huttunen and R. Piche, "A monocular camera gyroscope," *Gyroscope and Navigation, vol. 3, no 2, pp. 124-131*, 2012

[69] L. Ruotsalainen, H. Kuusniemi, and R. Chen, "Visual-aided two dimensional pedestrian indoor navigation with a smartphone," *Journal of Global Positioning Systems, vol 10, no 1*, 2011

[70] H. Kuusniemi, L. Chen, L. Ruotsalainen, L. Pei, Y. Chen and R. Chen, ,"Multi-sensor Multi-Network Seamless Positioning with Visual Aiding," *Proceedings of ICL-GNSS*, 2011

[71] D. Liebowitz, "Camera calibration and reconstruction of geometry," *PhD thesis, University of Oxford*, 2001

[72] M. Doncel, "Detection and Tracking of Vanishing Points in dynamic Environments," *PhD thesis*, *Universidad Politecnica de Madrid*, 2010

[73] C. Kessler, C. Ascher, N. Frietsch, M. Weinman, and G. Trommer, "Vision-based attitude estimation for indoor navigation using vanishing points and lines," *IEEE/ION Position Location and Navigation Symposium* (*PLANS)*, 2010

[74] R. Harley and A. Zisserman, "Multiple View Geometry in Computer Vision," *2nd ed., Cambridge University Press,* 2003

[75] D. Prahl, B. Captain, USAF, "Coupling Vanishing Point Tracking with Inertial Navigation to Estimate Attitude in a Structured Environment," *DEPARTMENT OF THE AIR FORCE AIR UNIVERSITY*, 2011

[76] D. Ziou and S. Tabbone (1998) "Edge detection techniques: An overview," International Journal of Pattern Recognition and Image Analysis, 8(4):537–559, 1998

[77] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986

[78] L. Roberts, "Machine Perception of Three-Dimensional Solids," *Technical Report TR315, Massachusetts Institute of Technology, Lexington Lincoln Laboratory*, 1965

[79] J. Prewitt, "Object Enhancement and Extraction," *Picture Processing and Psychopictorics*, 1970

[80] I. Sobel, "Camera Models and Machine Perception," *Ph.D. thesis, Stanford University*, 1970

[81] R. Gonzalez and R. Woods, "Digital Image Processing," *Pearson Prentice Hall, 3rd edition*, 2008

[82] P. Hough, "Method and Means for Recognizing Complex Patterns," U.S. Patent 3069654, December 1962

[83] R. Duda, and P. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Commun. ACM, 15(1):11–15*, 1972

[84] M. Fischler and R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the Association for Computing Machinery*, 1981

[85] C. Sunglok, K. Taemin, and Y. Wonpil, "Performance Evaluation of RANSAC Family," *Proceedings of the British Machine Vision Conference (BMVC)*, 2009

[86] O. Chum, "Two-View Geometry Estimation by Random Sample and Consensus," *PhD Thesis*, 2005

[87] D. Forsyth and J. Ponce, "Computer vision: a modern approach," *Prentice Hall*, 2003

[88] D. Choukroun, "Novel Methods for Attitude Determination Using Vector Observations," *The Technion, PhD Dissertation*, 2003

[89] D. Titterton and J. Weston, "Strapdown Inertial Navigation Technology," *2nd ed, The American Institute of Aeronautics and Astronautics and The Institution of Electrical Engineers*, 2004

[90] P. Savage, "Strapdown System Computational Elements," *NATO RTO Educational Notes, SET-064, Neuilly-sur-Seine Cedex: France*, 2004

[91] E. Bortz, "A New Mathematical Formulation for Strapdown Inertial Navigation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-7, pp. 61–66, 1971

[92] E. Edwan, J. Zhou, J. Zhang, and O. Loffeld, "A New Loosely Coupled DCM Based GPS/INS Integration Method," *Navigation: Journal of the Institute of Navigation, Vol 59. No. 2*, 2012

[93] W. Chai, J. Zhou, C. Chen, H. Nies and O. Loffeld, "Continuous Indoor Localization and Navigation Based on low-cost INS/Wi-Fi integration," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2011

[94] C. Chen, W. Chai, A. Nasir and H. Roth, "Low cost IMU based indoor mobile robot navigation with the assist of odometry and Wi-Fi using dynamic constraints," *IEEE/ION Position Location and Navigation Symposium (PLANS)*, 2012

[95] W. Chai, C. Chen, E. Edwan, J. Zhang and O. Loffeld, "Enhanced INS Wi-Fi Integration for Indoor Vehicle Navigation," *Journal of Computer and Communication (JCC)*, 2012

[96] W. Chai, C. Chen, E. Edwan, J. Zhang and O. Loffeld, "INS/Wi-Fi Based Indoor Navigation Using Adaptive Kalman Filtering and Vehicle Constraints," *9th IEEE Workshop on Positioning Navigation and Communication (WPNC)*, 2012

[97] W. Chai, C. Chen, E. Edwan, and O. Loffeld, "2D/3D Indoor Navigation Based on Multi-sensor Assisted Pedestrian Navigation in Wi-Fi Environments," *International Conference and Exhibition Ubiquitous Positioning Indoor Navigation and Location*

Based Service (UPINLBS), 2012

[98] W. Chai, C. Chen, E. Edwan, J. Zhang and O. Loffeld, "Adaptive Indoor Navigation Using Integration of IMU Based Pedestrian Dead Reckoning and RSS Fingerprinting," Inertial Sensors and Systems - Symposium Gyro Technology, 2012

[99] W. Chai, C. Chen, E. Edwan, and O. Loffeld, "DCM Based Attitude Estimation with INS/Visual-Gyroscope Integration for Indoor Navigation", Inertial Sensor and Systems - Symposium Gyro Technology, 2013

[100] C. Chen, W. Chai, and H. Roth, "A RGB and D Vision Aided Multi-Sensor System for Indoor Mobile Robot and Pedestrian Seamless Navigation," IEEE/ION Position Location and Navigation Symposium (PLANS), 2014

[101] C. Chen, W. Chai, and H. Roth, "A Single Frame Depth Visual Gyroscope and Its Integration for Robot Navigation and Mapping in Structured Indoor Environments," Journal of Intelligent & Robotic Systems, Springer, DOI:10.1007/s10846-014-0167-x, 2015

[102] W. Chai, E. Edwan and C. Chen, " Enhanced indoor navigation using fusion of IMU and RGB-D camera," CISIA, Advances in Computer Science Research, 2015

[103] W. Jiang, "Estimation techniques for a typical nonlinear system," Project Work, ZESS, University of Siegen, 2011

[104] T. Zhou, "Estimation Techniques for Orbit Determination from TDOA Radar Measurements," Project Work, ZESS, University of Siegen, 2011

[105] S. Huang, P. Diao, and S. Zhang, "Indoor localization techniques in Wi-Fi environment," Project Work, ZESS, University of Siegen, 2012

[106] T. Zhou, "Development of an indoor navigation system based on INS/Wi-Fi integration," Master Thesis, ZESS, University of Siegen, 2012

[107] S. Zhang, " 2D/3D Indoor Pedestrian Navigation with a Foot-mounted IMU," Master Thesis, ZESS, University of Siegen, 2013

[108] S. Huang, "Development of a pedestrian navigation system for portable devices," Master Thesis, ZESS, University of Siegen, 2013

[109] P. Diao, "Integration of INS and Visual-gyroscope with Nonlinear Kalman Filtering," Master Thesis, ZESS, University of Siegen, 2013