Pedestrian Navigation System Using Shoe-mounted INS

By
Yan Li

A thesis submitted for the degree of
Master of Engineering (Research)

Faculty of Engineering and Information Technology
University of Technology, Sydney (UTS)

July 2014

Certificate of Original Authorship

I, Yan LI, certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature: _____

Date: _____

Acknowledgment

I would like to express my thanks to my supervisor Dr. Jianguo Jack Wang for his advice and support. Without his support, I cannot come to UTS to pursue my master degree. During two years' study with him, I've learnt a lot, both for research and daily life. His rigorous academic attitude and positive view of life really affect me a lot.

Thanks to the people who have offered me great help and advice. Firstly Prof. Dikai Liu who is the director of the CAS center and he is really kind to help all the students in our lab. I'm really appreciating his help when I apply to come to UTS and his financial support for my research career. Dr. Xiaoing Kong is my co-supervisor and she advises me a lot regarding mathematics and basic concepts of my research topic. My thanks also go to Prof. Hong for her kindness and care. She is not only a supervisor, but also an elder who earns our respect.

My colleague and friend Shifeng Jason Wang and Lei Shi, they always offered me great comfort anytime I felt depressed and can always give me helpful advice for better development. Mr Xiang Luo and Xiang Thomas Ren, thanks for your help during the data collection experiments. Thanks Mr Ankur Sinha for your help to revise my articles and your contribution for constructing the NAO robot navigation data collection system. And thanks all the CAS colleagues, Yuhan Huang and Kanzhi Wu, thanks for your accompany for badminton which is the happy time for one week's entertainment.

Special thanks to my husband Adrian for his love and care. I cannot image if I can finish this master degree without your understanding and support. Thanks to my parents for their support in my life. Hope you are proud of me.

Contents

List of Abbreviations

GPS	Global Positioning System
INS	Inertial Navigation System
IMU	Inertial Measurement Unit
ZUPT	Zero Velocity Update
EKF	Extended Kalman Filter
CUPT	Constant Velocity Update
RFID	Radio Frequency Identification
WLAN	Wireless Local Area Network
UWB	Ultra Wide Band
RSS	Received Signal Strength
SLAM	Simultaneous Localization And Mapping
SINS	Strapdown Inertial Navigation System
MEMS	Micro Electro Mechanical System
ARW	Angular Random Walk
RW	Random Walk
ECEF	Earth-Centred Earth-Fixed
PDR	Pedestrian Dead Reckoning
MV	Moving Variance
MAG	Acceleration Magnitude
ARE	Angular Rate Energy
SHOE	Stance Hypothesis Optimal Estimation
ZVD	Zero Velocity Detectors
RTS	Rauch-Tung-Streibel
VO	Visual Odometry
UKF	Unscented Kalman Filter
PF	Particle Filter

List of Figures

List of Tables

ABSTRACT

Pedestrian navigation using Global Positioning System (GPS) is still a considerable challenge in indoor environments where GPS signals are blocked. Inertial Navigation System (INS) is a self-contained system which can offer a navigation solution in most environments without the need for any additional infrastructures.

A type of pedestrian navigation system with shoe-mounted Inertial Measurement Units (IMUs) has shown promising results. During walking, the foot is briefly stationary at zero velocity on the ground, named as the stance phase. The technique zero velocity update (ZUPT) is implemented to constrain the sensors' error which uses the stance phase in each step to provide corrections periodically.

In this research, a model with 24 error states is applied to correct IMU errors with an Extended Kalman Filter (EKF). The EKF estimated velocity errors are reset to zero in each stance phases, and successively to correct the IMU measurements. These repeated corrections could effectively control the error growth in navigation solution and minimize the drift.

This thesis introduces three main contributions I have achieved for pedestrian navigation system with shoe-mounted IMU. Firstly, I have developed a new approach to detect the stance phase of different gait styles, including walking, running and stair climbing. Secondly, I have proposed a new concept called constant velocity update (CUPT) which is an extension of ZUPT to correct IMU errors on a moving platform with constant velocity, such as elevators or escalators. This new concept has broadened the practical application of pedestrian navigation based on shoe-mounted IMUs in a

modern building environment. Lastly, as ZUPT applied at each step will lead to sharp corrections and discontinuities in the estimated trajectory, I developed a closed-loop step-wise smoothing algorithm to eliminate sharp corrections and smooth the trajectory. A software package in MATLAB has been developed and tested on different subjects. Good pedestrian navigation solutions have been achieved with the proposed method, which are published in journal and conference papers.

KEYWORDS: Pedestrian navigation, IMU, Step Detection, Kalman Filter, ZUPT, CUPT, RTS smoothing.

Chapter I

Introduction

1.1 Background

Pedestrian navigation systems are useful for emergency services, finding and guiding blind persons, security personal and for a wide range of augmented reality applications [1]. In recent years, the GPS has become one of the primary methods for outdoor pedestrian navigation. However, a tracking method based solely on GPS is expected to malfunction in indoor environments where the GPS signal will be blocked.

To obtain seamless navigation in an environment with degraded GPS signals is still a challenging issue. There have been different approaches for indoor positioning and navigation in the form of non-GPS systems, for example, Radio Frequency IDentification (RFID), Wireless Local Area Network (WLAN/WIFI) and Ultra Wide Band (UWB). RFID can use absolute position information embedded in it to aid navigation [2]. WLAN or WIFI provides absolute position information by using Received Signal Strength (RSS) [3]). UWB provides position and location estimation by estimating the distances to all other transmitters. All of these, however, do require some form of infrastructure.

An alternative approach would be to use inertial sensors, commonly configured as an IMU, which has the advantage of being small, low power, inexpensive and not relying on any external infrastructures or landmarks. Inertial sensor based systems are self-contained, environment-independent and can provide continuous navigation information with a high data rate. However, when operated in a stand-alone mode, INS

can experience large position errors due to time-dependent growth of errors. As a result, its use for positioning applications is relatively limited, unless frequent measurement updates from external sensors are available to correct the low-cost IMU error. GPS and INS integration has got much attention in navigation applications. Godha and Lchapelle [4] proposed a system that combined shoe-mounted IMU and GPS to bound drift errors in outdoor scenarios. On the other hand in indoor environments with a significant number of constraints the error can be substantially limited as shown in [5]. Therefore, a priori information about a building map can help to significantly reduce the error growth for the pedestrian indoor dead reckoning.

In order to improve the reliability of navigation systems, a more efficient way is to use redundant sensors or measurements in a navigation system instead of simply to upgrade the sensor grade. Vision sensors (e.g., such as camera, hyper-spectral sensors and laser range finders) are widely used for mapping and environments detection. Hisashi et al. proposed an image sequence matching technique for the recognition of locations and previously visited places [6]. Huang provided a tightly coupled integration by using the residual between the predicted and actual feature location as the filter's input, which was expressed in the image coordinate frame [7]. Hide et al. investigated the use of computer vision derived velocity measurements to frequently correct the drift of a low-cost IMU. A walking user carrying the mobile device in front of him and with the camera pointing toward the ground is considered; his method exploits the extracted images to evaluate the translation between frames and consequently to bound the error accumulation due to the use of the inertial sensors [8]. This approach is limited by the estimation of the orientation of the camera, the lighting conditions and the uniqueness of features in the frames. The Simultaneous Localization And Mapping (SLAM) algorithm augments the landmark locations to a map and estimates the vehicle position

with successive observations. However SLAM with vision sensors can achieve real-time implementation only when a landmark database is kept small. In addition, the errors of navigation using SLAM may grow as the platform moves across a long distance. Besides, unlike the inertial sensor, these sensors are very sensitive to the environment and are unreliable in unfavorable operation conditions.

Recently, the concept of attaching the INS to the pedestrian's shoe makes low cost INS for pedestrian navigation feasible. This results in the dominant advantage that the foot has to be briefly stationary while it is on the ground. Whether a pedestrian is walking, running or climbing stairs, there is a stance phase in each step which can provide a zero velocity measurement for error correction of a shoe-mounted IMU. This approach is called ZUPT.

Velocity errors during this period will be returned to a Kalman filter. The filter propagates and estimates the errors during the stance phase, which are fed back to the INS for correction of the internal navigation states. The ZUPT process could not only correct the user's velocity, but also help restrict the position and attitude errors and estimate the sensor bias errors. Therefore, these repeated corrections to the INS measurements could control the error growth and minimize the position drift. Consequently it is critical to correctly identify the stance phase in each step and then to apply ZUPT for IMU error correction.

1.2 Research Motivation

For a pedestrian navigation system based on a low-cost IMU, the technique known as ZUPT is implemented to control error growth. In order to apply ZUPT, correct stance phase identification is important. Researchers have developed different

methods for step detection, however, most of the step detection algorithms only function for walking, but fail to detect stance phases in high-speed movement (running or jumping). Compared with walking, the duration of the stance phase in running is shorter and the velocity in the stance phase is less close to zero velocity. Simply enlarging the threshold will introduce false stance phase detection. It is a challenge to correctly detect stance phases in all the moving styles.

Moreover, none of the shoe-mounted MEMS inertial sensors based pedestrian navigation systems, using strap-down inertial navigation or pedestrian dead-reckoning methods, can perform well in situations with constant velocity, such as in an elevator or on an escalator. This is due to the fact that they treat stance phases always at zero velocity even if a sensor is not stationary but moving with a constant velocity.

As the errors of estimated position and velocity grow with time rapidly for shoe-mounted MEMS inertial sensors, ZUPT applied at each step leads to sharp corrections of velocity and position; the estimated trajectories have discontinuities and periodic drift, which produces inaccurate navigation and tracking.

Consequently, a pedestrian navigation system that is capable of navigating autonomously in all circumstances is in high demand.

1.3 Contributions

The scope of this thesis is to build a framework for a reliable pedestrian navigation system by addressing the problems of the subject with the sensor in a high speed movement (running) and on a moving platform (lift or escalator).

The research undertaken, which is presented in this thesis, demonstrates three significant contributions:

- We have proposed a robust stance phase detection algorithm to handle both walking and running. The new algorithm consists of a footstep detector to indicate a new step and a swing phase detector to inform the end of the step; a stance phase detector is applied to detect whether the foot is stationary.
- As the extension of ZUPT, a new concept called CUPT has been proposed to apply velocity correction when a pedestrian is on an escalator or in an elevator. CUPT can broaden the practical use of a shoe-mounted pedestrian navigation system in a building environment.
- A closed loop step-wise smoothing algorithm is implemented to eliminate the sharp corrections over the steps.

1.4 Thesis outline

This thesis is arranged into six chapters which are now outlined.

Following this chapter, Chapter 2 describes the theoretical knowledge used throughout the thesis. The fundamental principles for the Strapdown Inertial Navigation System (SINS) will be explained. Then, a brief overview of the fundamentals of the Kalman Filter will be given.

In Chapter 3, a literature search of the ZUPT technique will be reviewed. This chapter will explain some of the current stance phase approaches to the matter of indoor pedestrian navigation. A detailed description of ZUPT aided INS will be explained first,

followed by a consideration of the improved stance phase detector which can handle the running movement. This chapter finishes with experimental results both for the walking and running cases.

Chapter 4 will discuss the problems of indoor navigation when a subject is on a moving platform, such as an elevator or escalator. A specific emphasis will be given to an introduction of the CUPT concept, particularly for constant velocity detection. The performance and limitations of CUPT approaches to aid a low-cost pedestrian navigation system will be explained.

Chapter 5 details what closed-loop step-wise smoothing is and how a closed-loop step-wise smoothing algorithm may be used to solve the discontinuity problem.

Finally, Chapter 6 concludes the thesis by summarizing the major findings. Based on the results achieved during the research, conclusions will be drawn and the continuity of the research will be suggested by recommending further research.

1.5 Publications

Following is the list of publication resulting from the work presented in this thesis:

- **Y Li**, Wang, J.J, "A Pedestrian Navigation System Based on Low Cost IMU", Journal of Navigation.
- **Y Li**, Wang, J.J, Kong, X, ''Zero velocity update with stepwise smoothing for inertial pedestrian navigation'', International Global Navigation Satellite

Systems Society IGNSS Symposium 2013, Gold Coast, Australia,16-18 July, 2013.

- **Y Li,** Wang, J.J, S Xiao, Xiang Luo, "Dead Reckoning Navigation with Constant Velocity Update (CUPT)", 12th International Conference on Control, Automation, Robotics and Vision (ICARCV), Guangzhou, China, 5-7 Dec, 2012.
- **Y Li,** Wang, J.J, "A robust pedestrian navigation algorithm with low cost IMU", IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN), Sydney, Australia, 5-7 Nov, 2012.
- **Y Li,** Xiang Luo, Ren, X.T. and Wang, J.J., "A Robust Humanoid Robot Navigation Algorithm with ZUPT", Proceedings of 2012 IEEE International Conference on Mechatronics and Automation (ICMA), Chengdu, China, August 5 - 8, 2012. Pages: 505-510.
- Xiang Luo, **Y Li,** Ren, X.T. and Wang, J.J, "Automatic Road Surface Profiling System Based on Sensors Fusion", 12th International Conference on Control, Automation, Robotics and Vision (ICARCV), Guangzhou, China, 5-7 Dec, 2012.
- S Xiao, **Y Li,** Xiaofeng Wu, Xueliang Bai, "Design of Imaging Guidance System with Double Optical Wedges", IEEE Conference on Industrial Electronics and Applications (ICIEA), Singapore, 18 -20 July, 2012.
- Xiang Luo, X. Ren, **Y Li,** and Wang, J.J, "Mobile Surveying System for Road Assets Monitoring and Management", IEEE Conference on Industrial Electronics and Applications (ICIEA), Singapore, 18-20 July, 2012.

1.6 Summary

This chapter has provided the relevant background to support the research into low-cost inertial pedestrian navigation systems. It has been pointed out that the current pedestrian navigation systems suffer limitations which become the motivation of this research. A set of contributions were then presented to demonstrate how this research

addresses the issues mentioned accordingly. A brief thesis outline is provided, describing each chapter and finally, the contributions to knowledge made by the research are summarized.

Chapter II

STRAPDOWN INERTIAL NAVIGATION SYSTEM AND KALMAN FILTER

This chapter will present the basic principles of inertial navigation, in particular the Strapdown Inertial Navigation System with its low cost MEMS IMU technology and their associated errors, was covered. The coordinate systems are defined in which the platform is required to navigate and the inertial quantities are measured. Furthermore, the chapter will provide the inertial navigation equations needed for pedestrian navigation application. The inertial navigation equations are then linearized to develop the INS error model. These equations are used to provide the mathematical foundation for error propagation in inertial navigation. It is then followed by an introduction to the Kalman Filter. In my tracking system, an error-state Kalman filter is investigated. This error-state Kalman filter estimates the deviation from the true state rather than the state itself. The principles are the same as those used in a standard Kalman filter. This is then followed by a brief chapter summary at the end.

2.1 Introduction to INS

Inertial navigation system is a self-contained navigation unit in which measurements provided by accelerometers and gyroscopes are widely used for tracking applications, such as aircraft, submarines, spacecraft and missiles guiding. An INS consists of an IMU together with a navigation processor. An IMU typically contains three orthogonal accelerometers and three orthogonal gyroscopes (gyros). The

Rate-Gyroscope measures angular velocities and the orthogonal accelerometer measures the linear acceleration in three orthogonal directions. By processing signals from these devices, it is possible to track the position and orientation of a device.

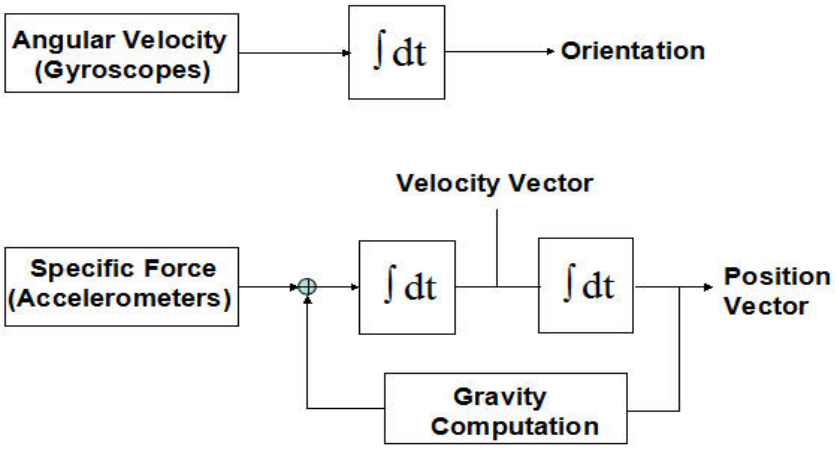


Figure 2.1 Fundamental Inertial Navigation System concept (adopted from [9])

An INS tracks its orientation, velocity and position in the reference frame. Figure 2.1 gives an overview of the strapdown inertial navigation algorithm that is used. The angular velocity measurements are integrated to track the orientation of the IMU relative to the reference frame. Specific force measurements made by the IMU are projected into this frame using the tracked orientation. Acceleration due to gravity is then added to obtain the acceleration of the IMU in the reference frame. This acceleration is then integrated once to track the velocity of the device and again to track its position [10].

Recent advances in the construction of Micro Electro Mechanical System (MEMS) have made it possible to manufacture small and light inertial navigation systems. Because of its attractive specification such as inexpensive, compact size, lightweight and low-power consumption, these sensors are becoming more popular in consumer-grade navigation systems. MEMS based IMUs are available in portable

devices. These sensors can be used to obtain information of relative change in a user's position. However, inherent drifts of these sensors limit their long term application [11]. These drifts results are caused by the errors in MEMS based IMUs.

2.2 INS Sensor Errors

The fundamental observables from INS sensors are specific forces sensed by accelerometer and angular velocity measured by gyroscopes. Either of the observables can be described by the following simplified expression [12]:

$$\mathrm{G}_I = I + b_I + S \cdot I + \epsilon(I) \tag{2.1}$$

Where

G_I: the inertial sensor outputs (specific force and angular velocity)

I: the true inertial sensor measurements

b_I: the sensor bias

S: the diagonal matrix of scale factors

$\epsilon(I)$: the random noise

The MEMS IMU dominant error sources that have appeared in this thesis are bias, scale factor error and random noise. Both accelerometers and gyroscopes have error due to slowly growing constant bias. Bias can be defined as the offset of the output signal from the true value. In other words, the bias is output from the gyroscope and accelerometer when the device is not undergoing any motion. The biases are estimated by measuring the long term average of the output device which is not undergoing any motion [13].

Scale factor is the ratio of the sensor input and sensor output. A scale factor error is the error in this ratio after unit conversion, which means a zero scale factor error produces a unity ratio. It can be caused by, for example, the imperfection in the pick-off sensor inside the IMU assembly [14]. It is often expressed in units of parts-per-million (ppm).

The MEMS IMU outputs are perturbed by various sources of noise, such as thermal noise and electrical noise [13]. Gyro noise is integrated to produce Angular Random Walk (ARW), and accelerometer noise is integrated to produce Random Walk (RW) on its velocity solution. Usually manufacturers specify noise in terms of ARW with units in degrees per hour ($°/\sqrt{hr}$).

The main types of error that perturb measurements from MEMS accelerometers are the same for MEMS gyroscopes. The primary difference is that errors in acceleration measurements are integrated twice to track the position of an IMU, whereas angular velocity measurements are only integrated once to track the estimated orientation. As a result bias errors cause errors in position that grow proportional to t^2, white noise causes a second order random walk in position and so on [15]. More detailed different error sources information can be found in [13].

2.3 Coordinate Frames

2.3.1 Inertial frame (i-frame)

The inertial frame (I) is a reference frame in which Newton's laws of motion are assumed to apply. Hence, the frame is non-rotating and non-accelerating. The definition of the inertial frame is:

Origin: The earth's centre of mass

X-axis: Toward the mean vernal equinox

Y-axis: Complete a right-handed orthogonal frame

Z-axis: Toward the earth's rotation axis

2.3.2 Earth-centred Earth-fixed (ECEF) frame (e-frame)

The ECEF frame is not inertial because it is fixed to the earth. Hence, it rotates relative to the inertial frame with the earth rotation rate. Its definition is:

Origin: The earth's centre of mass

X-axis: Toward the Greenwich meridian in the equatorial plane

Y-axis: Complete a right-handed orthogonal frame

Z-axis: Toward the earth's rotation axis

The e-frame rotates about the z-axis at a rate known as Earth rate. This rotation rate vector with respect to i-frame resolved to the e-frame is given as [16]:

$$w_{ie}^{e} = (0\ 0\ w_e)^T \tag{2.2}$$

Where w_e is the magnitude of Earth rate ($7.292115\times 10^{-5} rad/s$).

The position vector in the e-frame can be expressed in terms of the geodetic latitude (φ), longitude (λ) and height (h) as follows:

$$r_e = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} (R_N + h)cos\varphi cos\lambda \\ (R_N + h)cos\varphi sin\lambda \\ (R_N(1 - e^2) + h)sin\varphi \end{pmatrix} \quad (2.3)$$

Where e is the eccentricity of the reference ellipsoid and R_N is the merdian radius of the curvature.

2.3.3 Navigation frame (n-frame)

NED frame is the right-handed system that is used as a navigation frame and its definition is:

Origin: The centre of INS

X-axis: Toward ellipsoidal true north (North)

Y-axis: Toward ellipsoidal east (East)

Z-axis: Downwards direction along the ellipsoidal normal (Down)

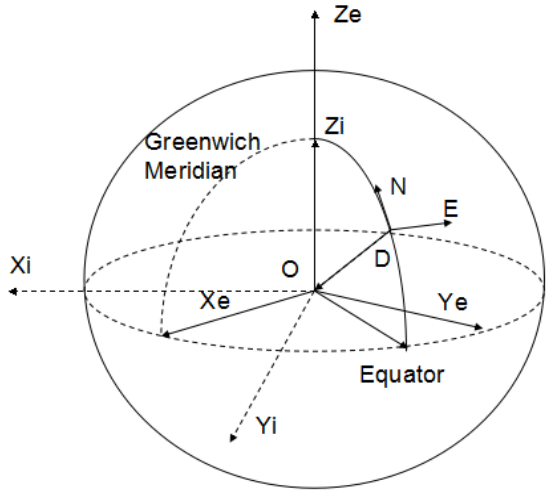


Figure 2.2 The inertial frame, earth fixed frame and navigation frame

2.3.4 Body frame (b-frame)

The body frame (b-frame) is a coordinate frame that remains fixed with respect to the IMU. For angular motion, x-axis, y-axis and z-axis are often known as roll, pitch and yaw-axis respectively.

Origin: The centre of INS

X-axis: Toward direction of travel

Y-axis: Complete a right-handed orthogonal frame

Z-axis: Downwards direction along the gravity

2.3.5 Rotation of coordinate frames

In a navigation system, it is necessary to perform vector transformation from one coordinate to another. The Cosine Matrix (DCM) from the n-frame to the e-frame is expressed in terms of geodetic latitude and longitude as:

$$R_n^e = \begin{bmatrix} -sin\varphi cos\lambda & -sin\lambda & -cos\varphi cos\lambda \\ -sin\varphi sin\lambda & cos\lambda & -cos\varphi sin\lambda \\ cos\varphi & 0 & -sin\varphi \end{bmatrix} \tag{2.4}$$

The Earth rotation rate can be described in the n-frame using

$$w_{ie}^n = R_e^n w_{ie}^e = [w_e cos\varphi \quad 0 \quad -w_e sin\varphi]^T \tag{2.5}$$

The measurements sensed by the accelerometers and gyroscopes are in body

frame while the output of position need to be recognized in navigation frame. To do this, a series of Euler Angles rotations: roll (ϕ), pitch (θ) and yaw (ψ) angles, are rotated in order. DCM from n-frame to b-frame can be written as [16]:

$$R_n^b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{pmatrix} \begin{pmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{pmatrix} \begin{pmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (2.6)$$

Therefore, because of its orthogonality, the DCM from the b-frame to n-frame can be found via its transpose matrix,

$$R_b^n = (R_n^b)^T = \begin{pmatrix} c\theta c\psi & -c\phi s\psi + s\phi s\theta c\psi & s\phi s\psi + c\phi s\theta c\psi \\ c\theta s\psi & c\phi c\psi + s\phi s\theta s\psi & -s\phi c\psi + c\phi s\theta s\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{pmatrix} \quad (2.7)$$

Where sin and cos are denoted as s and c respectively.

The Euler angles can then be extracted from the DCM using the following equations:

$$\phi = tan^{-1}(\frac{R_{32}}{R_{33}})$$

$$\theta = sin^{-1}(R_{31}) \quad (2.8)$$

$$\psi = tan^{-1}(\frac{R_{21}}{R_{11}})$$

Where R_{mn} refers to row (m) and column (n) of elements in Eq 2.7.

The coordinate frame transformation can also use a quaternion. The quaternion, q is a vector that has four components:

$$\mathrm{q} = (q_0\ q_1\ q_2\ q_3)^T \tag{2.9}$$

Where q_0 represents the magnitude of the rotation, and the other three components represent the components of a vector directed along the Euler axes between the two frames. If Euler Angles are used, the transformation from b-frame to n-frame can be computed as [17]:

$$q_b^n = \begin{pmatrix} c\frac{\phi}{2}c\frac{\theta}{2}c\frac{\psi}{2} + s\frac{\phi}{2}s\frac{\theta}{2}s\frac{\psi}{2} \\ s\frac{\phi}{2}c\frac{\theta}{2}c\frac{\psi}{2} - c\frac{\phi}{2}s\frac{\theta}{2}s\frac{\psi}{2} \\ c\frac{\phi}{2}s\frac{\theta}{2}c\frac{\psi}{2} + s\frac{\phi}{2}c\frac{\theta}{2}s\frac{\psi}{2} \\ c\frac{\phi}{2}c\frac{\theta}{2}s\frac{\psi}{2} - s\frac{\phi}{2}s\frac{\theta}{2}c\frac{\psi}{2} \end{pmatrix} \tag{2.10}$$

2.4 Strapdown Inertial Navigation Mechanization

Strapdown inertial navigation is defined when an IMU is ‘strapped’ to the body of a system or onto a device where the IMU is installed. In strapdown systems the inertial sensors are mounted rigidly onto the device, and therefore output quantities are measured in the body frame rather than the global frame. The process of integrating the gyro measurements to generate attitude, and combining the attitude with double integrated accelerometer measurements is known as the INS mechanisation, shown in figure 2.3. Nowadays strapdown systems are widely used for most navigation and positioning applications.

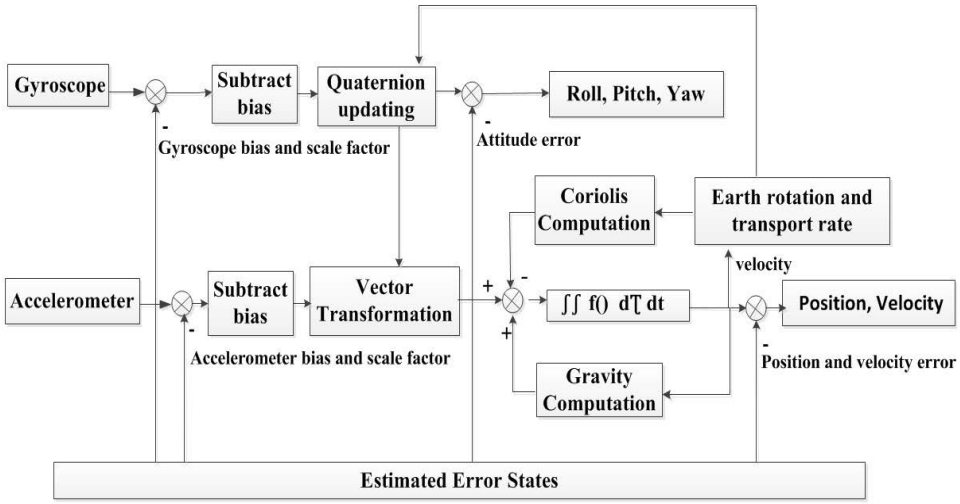


Figure 2.3 The basic blocks of strapdown inertial navigation system mechanization

Based on the measured data of gyroscope and accelerometers, we can get the position, velocity and attitude information of a device. Attitude can be determined by integration of the rotation rate of gyros. The outputs of the gyroscope w_{ib} are integrated to compute the attitude of the device and the direction cosine matrix C_b^n which is required to transform the measured accelerations f^b into navigation frame f^n. Velocity and position can be determined by integration of acceleration with compensation of gravity and coriolis. Within the navigation frame, the rotation of the Earth w_{ie} is taken into account as well as and the rotation of the navigation frame (transport rate) w_{en}^n. The navigation computer integrates the corrected accelerometer signals to compute the corresponding velocity and position data of the device.

The continuous forms of dynamic equations for strapdown INS navigation parameters are derived. The detailed expression can be found in [9].

$$\dot{r}^n = -\Omega_{en}^n r^n + v^n$$

$$\dot{v}^n = f^n - (\Omega_{en}^n + 2\Omega_{ie}^n)v^n + g^n \tag{2.11}$$

$$\dot{R}_b^n = R_b^n \Omega_{ib}^b - \Omega_{in}^n R_b^n$$

With

$$w_{en}^n = \begin{bmatrix} \dot{\lambda}\cos(\varphi) \\ -\dot{\varphi} \\ -\dot{\lambda}\sin(\varphi) \end{bmatrix}$$

$$w_{ie}^n = \begin{bmatrix} w_{ie}\cos(\varphi) \\ 0 \\ -w_{ie}\sin(\varphi) \end{bmatrix} \tag{2.12}$$

$$w_{in}^n = w_{ie}^n + w_{en}^n$$

Where

r^n: the position vector in n-frame

v^n: the velocity vector in n-frame

g^n: the gravity vector in n-frame

R_b^n: the rotation matrix from the b-frame to the n-frame

φ, λ: the latitude and longitude

f^n: the specific force vector in the n-frame that is transformed from the specific force vector f^b sensed in accelerometers

Ω_{ib}^b: the skew-symmetric form of angular velocity vector w_{ib}^b sensed in gyroscopes

Ω_{en}^n: the skew-symmetric form of the n-frame rotation rate vector w_{en}^n

Ω_{ie}^n: the skew-symmetric form of the n-frame rotation rate vector w_{ie}^n

Ω_{in}^n: the skew-symmetric form of the n-frame rotation rate vector w_{in}^n

2.5 INS error model

The error-state is the difference between the true state and the solution calculated by the INS. The Psi-angle error model is adopted in the thesis to propagate error states that were being estimated [18]:

$$\begin{aligned} \delta\dot{r} &= -w_{en} \times \delta r + \delta v \\ \delta\dot{v} &= -(w_{ie} + w_{in}) \times \delta v - \delta\varphi \times f + \delta g + \nabla \\ \delta\dot{\varphi}^{n} &= -\Omega_{in}^{n}\delta\varphi + \varepsilon \end{aligned} \tag{2.13}$$

Where δr, δv, δφ are the position, velocity, and attitude error vectors in n-frame respectively, ∇ is the accelerometer error vector, δg is the error in the computed gravity vector, ε is the gyro error vector, f is the specific force vector, w_{ie} is the earth rotation rate vector, w_{en} is the transport rate vector and w_{in} is the angular rate vector from the navigation to the inertial frame.

These error equations represent the system dynamic model, which are used to form the dynamic matrix F in the Kalman Filter. Using the knowledge from the error model, a Kalman filter can be used to estimate the inertial sensor errors over time, and subsequently can be used to correct the navigation parameters. In this research, along with the error model, an error-state Kalman Filter was used as the estimation filter.

The Kalman Filter estimates the INS error states based on the IMU measurements. When a new measurement is received, it is used to update the estimated error-state during the filter's correction step. This estimate is then immediately transferred to compensate the INS states before being reset to zero which is known as closed-loop correction.

2.6 Kalman Filter

This research used a traditional error-state KF to estimate INS errors. The error-state KF that was used falls under an EKF, which is a linearized-type of KF that has an INS error control loop (feedback) control system [17], where it linearizes the system dynamic model and the measurement model. This means that the low-cost IMU errors are assumed to be propagated linearly, and the use of EKF with this assumption is deemed reliable for this research.

2.6.1 Principle of the Kalman Filter

The Kalman Filter is a linear estimation technique that comprises a set of algorithms in a recursive configuration. In order to estimate the states parameters of the system, the KF uses two models known as dynamic model and measurement model. The dynamic model is represented in continuous time as

$$\dot{x}(t) = F(t)x(t) + Gw(t) \tag{2.14}$$

Similarly, in discrete form, it is represented as:

$$x_{k+1} = \Phi x_k + w_k \tag{2.15}$$

Where x is the system state vectors, F is the system dynamic matrix, G relates the disturbing forces to the state vectors, w is the disturbing force vectors, Φ is the state transition matrix that relates the state vector from epoch (k) to epoch (k+1) and w_k is the process noise vector.

The measurement model is represented as:

$$z_k = H_k x_k + v_k \tag{2.16}$$

Where z_k is the measurement vector at time epoch k, H_k is the measurement model matrix, which linearly relates states to the measurements and v_k is the measurement noise vector. Both noise vectors (w_k;v_k) are assumed to be uncorrelated with each other. They are also assumed to be zero mean Gaussian white, normally distributed and mutually independent; the corresponding stochastic models are assumed as:

$$\mathrm{E}[\mathrm{w(t)}] = 0$$

$$\mathrm{E}[v(\mathrm{t})] = 0$$

$$\mathrm{E}[w_i v_j^T] = 0 \tag{2.17}$$

$$\mathrm{E}[w_i w_j^T] = \begin{cases} R_k & i = j \\ 0 & i \neq j \end{cases}$$

$$\mathrm{E}[v_i v_j^T] = \begin{cases} Q_k & i = j \\ 0 & i \neq j \end{cases}$$

Where R is the system plant noise covariance matrix and Q is the measurement noise covariance matrix. E is the expectation operator.

2.6.2 Kalman filter prediction

The Kalman filter is an iterative filtering procedure which can be divided into two steps of prediction and measurement update. The first step of the Kalman filter iteration is to predict the estimate of the state $\hat{x}_k^{(-)}$ based on the previous best estimate of the state from the previous epoch $\hat{x}_{k-1}^{(+)}$. This is achieved through using the state transition matrix Φ,

$$\hat{x}_k^{(-)} = \Phi \hat{x}_{k-1}^{(+)}$$

$$P_k^{(-)} = \Phi_{k,k-1} P_{k-1}^{(+)} {\Phi_{k,k-1}}^T + R_{k-1} \tag{2.18}$$

Where $\hat{x}_{k-1}^{(+)}$ and $P_{k-1}^{(+)}$ are the optimal estimators of the state vector and its covariance matrix at the previous (k-1) epoch respectively, and $\Phi_{k,k-1}$ is the transition matrix.

2.6.3 Kalman filter measurement update

Once measurements are available, the state estimate is then updated where the optimal estimate of the state vector and its corresponding measurement update covariance matrix is

$$\hat{x}_k^{(+)} = \hat{x}_k^{(-)} + K_k(z_k - H_k \hat{x}_k^{(-)})$$

$$P_k^{(+)} = (\mathrm{I} - K_k \mathrm{H}_k) P_k^{(-)} \tag{2.19}$$

With

$$K_k = P_k^{(-)} {\mathrm{H}_k}^T [\mathrm{H}_k P_k^{(-)} {\mathrm{H}_k}^T + Q_k]^{-1} \tag{2.20}$$

Where $\mathrm{K_k}$ is the so-called Kalman gain matrix.

It is common to use the Joseph form for Eq (2.19) because it improves numerical stability and has natural symmetry [19]

$$P_k^{(+)} = (\mathrm{I} - K_k \mathrm{H}_k) P_k^{(-)} (\mathrm{I} - K_k \mathrm{H}_k)^T + K_k R_k K_k^T \tag{2.21}$$

2.6.4 State vector and dynamic model

An exact expression for the system equation of an EKF depends on the states selected and the type of error model used to describe them. The EKF used includes the following 24 error states:

$$\delta x = [\delta x_{Nav}, \delta x_{INS}, \delta x_{Grav}]$$

$$\delta x_{Nav} = \left[\delta r_{N,} \delta r_{E,} \delta r_{D,} \delta v_{N,} \delta v_{E,} \delta v_{D,} \delta \varphi_{H,} \delta \varphi_{P,} \delta \varphi_{R}\right]^{T}$$

$$\delta x_{INS} = \left[\nabla_{b,} \nabla_{f,} \varepsilon_{b,} \varepsilon_{f}\right] \tag{2.22}$$

$$\delta x_{Grav} = \left[\delta g_{N,} \delta g_{E,} \delta g_{D}\right]$$

Where δx_{Nav}, δx_{INS} and δx_{Grav} are the navigation error vector, the IMU sensor measurement error vector and gravity uncertainty, respectively. ∇ is the accelerometer error vector, and ε is the gyro error vector. Subscript b stands for bias and subscript f stands for scale factor.

The dynamic matrix is obtained by a linearization of the equation (2.13), as presented in equation (2.23):

$$\begin{bmatrix} \dot{X}_{r} \\ \dot{X}_{v} \\ \dot{X}_{\varphi} \\ \dot{X}_{acc} \\ \dot{X}_{gyro} \\ \dot{X}_{grav} \end{bmatrix} = \begin{bmatrix} F_{11} & I & 0 & 0 & 0 & 0 \\ F_{21} & F_{22} & F_{23} & F_{24} & 0 & I \\ 0 & 0 & F_{33} & 0 & F_{35} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & F_{66} \end{bmatrix} \begin{bmatrix} X_{r} \\ X_{v} \\ X_{\varphi} \\ X_{acc} \\ X_{gyro} \\ X_{grav} \end{bmatrix} + \begin{bmatrix} W_{r} \\ W_{v} \\ W_{\varphi} \\ W_{acc} \\ W_{gyro} \\ W_{grav} \end{bmatrix} \tag{2.23}$$

Where I and 0 are the third-order identity and zero matrices respectively; $x_r, x_v, x_\varphi, x_{acc}, x_{gyro}$, x_{grav} are the position, velocity and attitude error vector,

accelerometer bias and scale factor, gyroscope bias and scale factor, and gravity uncertainty vector; $\mathrm{w_r}, \mathrm{w_v}, , \mathrm{w_\varphi}, \mathrm{w_{acc}}, \mathrm{w_{gyro}}, \mathrm{w_{grav}}$ are all zero-mean Gaussian white noise vectors.

A detailed expression of the dynamic matrix (F) can be given by:

$$F_{11} = \begin{bmatrix} 0 & {F_x}^{11} & -{F_z}^{11} \\ -{F_x}^{11} & 0 & {F_y}^{11} \\ {F_z}^{11} & -{F_y}^{11} & 0 \end{bmatrix} \tag{2.24}$$

With

$$\begin{gathered} {F_x}^{11} = -\dot{\lambda}\sin(\varphi) \\ {F_y}^{11} = \dot{\lambda}\cos(\varphi) \\ {F_z}^{11} = -\dot{\varphi} \\ F_{21} = diag\left[-\frac{g}{R_e} \quad -\frac{g}{R_e} \quad \frac{2g}{R_e}\right] \\ F_{22} = \begin{bmatrix} 0 & {F_x}^{22} & -{F_z}^{22} \\ -{F_x}^{22} & 0 & {F_y}^{22} \\ {F_z}^{22} & -{F_y}^{22} & 0 \end{bmatrix} \end{gathered} \tag{2.25}$$

With

$$\begin{gathered} {F_x}^{22} = -(2w_{ie} + \dot{\lambda})\sin(\varphi) \\ {F_y}^{22} = (2w_{ie} + \dot{\lambda})\cos(\varphi) \\ {F_z}^{22} = -\dot{\varphi} \end{gathered}$$

$$F_{23} = R_b^n \begin{bmatrix} 0 & -f_z^b & f_y^b \\ f_z^b & 0 & -f_x^b \\ -f_y^b & f_x^b & 0 \end{bmatrix}$$

$$F_{24} = \begin{bmatrix} R_b^n & R_b^n \begin{bmatrix} f_x^b & 0 & 0 \\ 0 & f_y^b & 0 \\ 0 & 0 & f_z^b \end{bmatrix} \end{bmatrix} \tag{2.26}$$

$$F_{33} = \begin{bmatrix} 0 & F_x{}^{33} & -F_z{}^{33} \\ -F_x{}^{33} & 0 & F_y{}^{33} \\ F_z{}^{33} & -F_y{}^{33} & 0 \end{bmatrix}$$

With

$$F_x{}^{33} = -(w_{ie} + \dot{\lambda})\sin(\varphi)$$

$$F_y{}^{33} = (w_{ie} + \dot{\lambda})\cos(\varphi)$$

$$F_z{}^{33} = -\dot{\varphi} \tag{2.27}$$

$$F_{35} = \begin{bmatrix} -R_b^n & -R_b^n \begin{bmatrix} w_x^b & 0 & 0 \\ 0 & w_y^b & 0 \\ 0 & 0 & w_z^b \end{bmatrix} \end{bmatrix}$$

$$F_{66} = diag[-\tau_x \quad -\tau_y \quad -\tau_z]$$

Where R_b^n denotes the direction cosine matrix from the b-frame to the n-frame, φ and λ denote latitude and longitude respectively, g is the gravity constant, $f_x^b, f_y^b\ f_z^b$ are the accelerometer sensed specific force in the b-frame, $w_x^b, w_y^b\ w_z^b$ are the gyroscope sensed rotation rate in the b-frame, R_e represents the radii of parallel curvature, τ_x, τ_y, τ_z is 1/(correlation time) of the Gauss-Markov process for gravity

uncertainty, w_{ie} is the Earth rate vector..

The velocity measurement y_v is given by:

$$\begin{aligned} y_v &= v + v_v \\ &= \hat{v} + \delta v + v_v \end{aligned} \tag{2.28}$$

Where $\hat{v}$ is the estimated velocity by INS and v_v is the measurement noise, modelled as a Gaussian random variable with zero-mean vector.

When y_v is available, the input measurement z_v to the error-state Kalman filter is given by:

$$\begin{aligned} z_v &= y_v - \hat{v} \\ &= \delta v + v_v \\ &= H\delta x + v_v \end{aligned} \tag{2.29}$$

Where $H^{3\times24} = [O^{3\times3} I^{3\times3} O^{3\times18}]$.

2.7 INS error correction

A continuous system equation for the INS system can be constructed as:

$$\dot{\delta x}(t) = F(t)\delta x(t) + Gw(t) \tag{2.30}$$

Where F is the dynamic matrix.

The continuous time equation can be transformed into discrete time form as shown:

$$\delta x_{k+1} = \Phi_k \delta x_k + w_k \tag{2.31}$$

Where Φ is the state transition matrix.

The following numerical approximation is used [20]:

$$\Phi_k \approx I + F\Delta t \tag{2.32}$$

After each correction step, the estimated error-states δx will be fed back to the INS mechanization loop to refine the INS output. Errors in velocity and position are compensated by:

$$\begin{aligned} \hat{v} &= \hat{v}^- + \delta v \\ \hat{r} &= \hat{r}^- + \delta r \end{aligned} \tag{2.33}$$

The rotation defined by the error vector $\delta\varphi = (\delta\varphi_x, \delta\varphi_y, \delta\varphi_z)$ is converted into its equivalent rotation matrix form based on small angles approximation.

$$\delta\Theta = \begin{pmatrix} 1 & \delta\varphi_z & -\delta\varphi_y \\ -\delta\varphi_z & 1 & \delta\varphi_x \\ \delta\varphi_y & -\delta\varphi_x & 1 \end{pmatrix} \tag{2.34}$$

Then the corrected attitude refinement is achieved by

$$\hat{R}_b^n = \delta\Theta\hat{R}_b^{n^-} \tag{2.35}$$

2.8 Summary

This chapter mainly introduces the basic knowledge of IMU and the mechanization of strapdown INS. Firstly; a simple sensor model is laid out that incorporated a bias, scale factor error and random noise. Secondly, this chapter gives a brief description of the coordinate frames which are usually used in inertial

measurement processing, coordinates transformation and navigation equations. System error dynamics is also presented followed by the principle of INS mechanization, which provides the dynamics of error propagation and compensation. An error-state Kalman filter is implemented to estimate the error states relates to INS; these errors are sent back to compensate bias of raw measurement of IMU, and integrate the navigation errors to refine the final navigation outputs after each correction step. This closed-loop correction method can be of benefit in drift elimination, accumulated errors correction and can give accurate navigation accuracy.

Chapter III

ZERO VELOCITY UPDATE AIDED PEDESTRIAN NAVIGATION SYSTEM

Recently, the concept of attaching the IMU to the pedestrian's shoe makes low cost INS for pedestrian navigation feasible. Although the low-cost inertial sensors are known to have huge errors increasing with time, utilizing the stance phase during a walking event enables ZUPT to be performed. Stance phase detection is made to introduce ZUPT, which reduces the errors of the navigation system. The EKF propagates and estimates the errors during the stance phase, which are fed back to the INS for correction of the internal navigation states. In this chapter, firstly the related work is reviewed and a detailed description of the foot-mounted ZUPT aided INS mechanisation is presented. The basic concept of ZUPT and the closed loop integration between the IMU measurements, the stance phase detector, the EKF and the INS solution is derived. This is followed by the introduction of improved ZUPT algorithms which are feasible for more sophisticated gait behaviours, such as climbing stairs and running. Finally the experimental results illustrate the utility and practicality of the system.

3.1 Introduction

The pedestrian dead reckoning (PDR) system has become a very popular system and will most likely be the foundation of future commercial indoor navigation systems. It is cheap, hard to jam, easy to operate and gives quite good positioning results. PDR using a foot-mounted IMU is the basis for many indoor localisation techniques. The PDR solution is the process of integrating step lengths and orientation estimations at

each detected step relative to a known starting point, so as to compute the position and orientation of a person.

The IMU mounted on the boot has three dimensional accelerometers, gyroscopes and magnetometers to measure the movements of the users. Double integrating the accelerometer measurements gives user position relative to the starting point if the gravity component can be removed correctly. Since sensor orientation is estimated using semi grade gyros, the orientation estimate will be a little bit off. This results in parts of the gravity component being interpreted as user accelerations, resulting in an ever growing position error [21].

An INS can provide continuous navigation information independently but with accumulated errors, which need to be corrected frequently by additional measurements. This is achieved by using a foot-mounted IMU and applying ZUPT pseudo-measurements when the user's foot is known to be stationary [22]. During the stationary time interval, the estimated velocity should be zero. If it is not, the system takes profit of this knowledge to correct the position and velocity estimations.

The general ZUPT-aided INS process can be divided into three steps. Firstly, estimate the position velocity and attitude information using IMU measurements. Secondly, the stance phase detector detects the stance phases of a walking event. Thirdly, run the ZUPT-aided Kalman filter. Once the stance phase detector detects that a foot is in a steady state, the EKF updates the navigation error states as well as the sensor errors and feeds the error states back into the loop for correction.

3.2ZUPT

As a special character for a pedestrian, when a pedestrian is walking, the stance phase in each step provides zero velocity measurement for error correction of the

shoe-mounted IMU. This approach is called zero velocity update (ZUPT). ZUPT uses the fact that when a person walks, his feet alternate between a stationary phase and a swing phase. This stance interval appears in each step at zero velocity, thus the velocity errors can be reset almost periodically. The ZUPT process could not only correct the user's velocity, but also help restrict the position and attitude errors and estimate the sensor errors. Therefore, these repeated corrections to the INS measurements could control the error growth and minimize the position drift. It is important to identify the stance intervals when the sensors attached to the shoe are stationary and then apply ZUPT to bind the errors with the EKF.

3.2.1 Related work

In order to apply ZUPT measurements in the KF, it is necessary to recognize the periods during which the user's foot is stationary. Correct stance phase detection is essential in a self-contained inertial navigation system that uses ZUPTs, this detection enables ZUPTs to be used correctly in the KF for error estimation.

Various pedestrian navigation systems have been developed. Foxlin recognized the possibility to use the stance phase for performing ZUPT as pseudo measurements in to a Kalman filter [23]. An improved shoe-mounted inertial navigation system was proposed based on the fuzzy logic step detector and an angular updating algorithm in [24]. Cho and Park proposed a pedometer-like approach which counts steps and estimates average length of steps. Step length is estimated from accelerometer readings that are passed through a neural network, and a Kalman Filter was used to reduce the effect of magnetic disturbances [25]. Alternatively, a dead reckoning method of propagating the step length over a certain direction is often used. Huang et al provided a method of fusing both strap-down and dead reckoning methods [26]. Feliz et al. used an

IMU and a GPS and barometer unit in their pedestrian navigation system [27]. A similar zero-velocity detector constructed with high-resolution pressure sensors mounted beneath the heel of a user's shoe is described and used with good results [28]. A detailed description of Kalman-based framework, called INS-EKF-ZUPT (IEZ), to estimate the position and attitude of a person while walking was proposed in [29].

Algorithms for step detection using accelerometers have been presented, which mainly contain three types: peak detection, zero crossing detection and flat zone detection [30]. The walking pattern can be detected with acceleration signal magnitude which crosses zero acceleration twice at each step. The zero crossing can be counted and divided by two for detecting number of steps taken [31]. Another option is to use knowledge of the human walking pattern to detect the stance phase and then apply the ZUPT. Typically, a walking gait can be modelled as a repeating sequence of heel strike, stance, push off and swing. A gait state is modelled as a Markov process and gait states are estimated using the hidden Markov model filter based on force sensors to determine when to apply ZUPT [32]. Similarly, a Markov model is constructed using segmentation of gyroscope outputs in [33]. Slightly different algorithms can be achieved based on both accelerometer and gyroscope outputs. The zero velocity is determined by comparing z-axis accelerometer and y-axis gyroscope outputs with the threshold value in [34]. Alternatively, the zero velocity is determined based on norms of accelerometer and gyroscope along with variance of accelerometer in [29].

Four detection methods to correctly detect stance phase have been investigated extensively by Skog et al. [35]. They are acceleration Moving Variance (MV), acceleration MAGnitude (MAG), Angular Rate Energy (ARE) and Stance Hypothesis Optimal Estimation (SHOE). Ojeda and Borenstein et al. [36] proposed a shoe-based navigation system which consists of a 15-state error model. Their system works well in

both 2D and 3D environments, on stairs and on rugged terrain. With the ARE detector, the horizontal relative error was about 0.49% of the total distance travelled, the vertical error was 1.2%, which might be the highest accuracy in related work.

3.2.2 Stance phase detection

An example of a walking sequence with a shoe mounted IMU is shown in Figure 3.1. The foot is stationary around sample 1.05e4, 1.028e4, 1.04e4,1.054e4 and 1.068e4. During these phases the magnitude of the accelerometer signals is the gravitation constant with some added noise. At the same time the norm of the angular velocity is zero with some additive noise. These are the key characteristics of the test statistic.

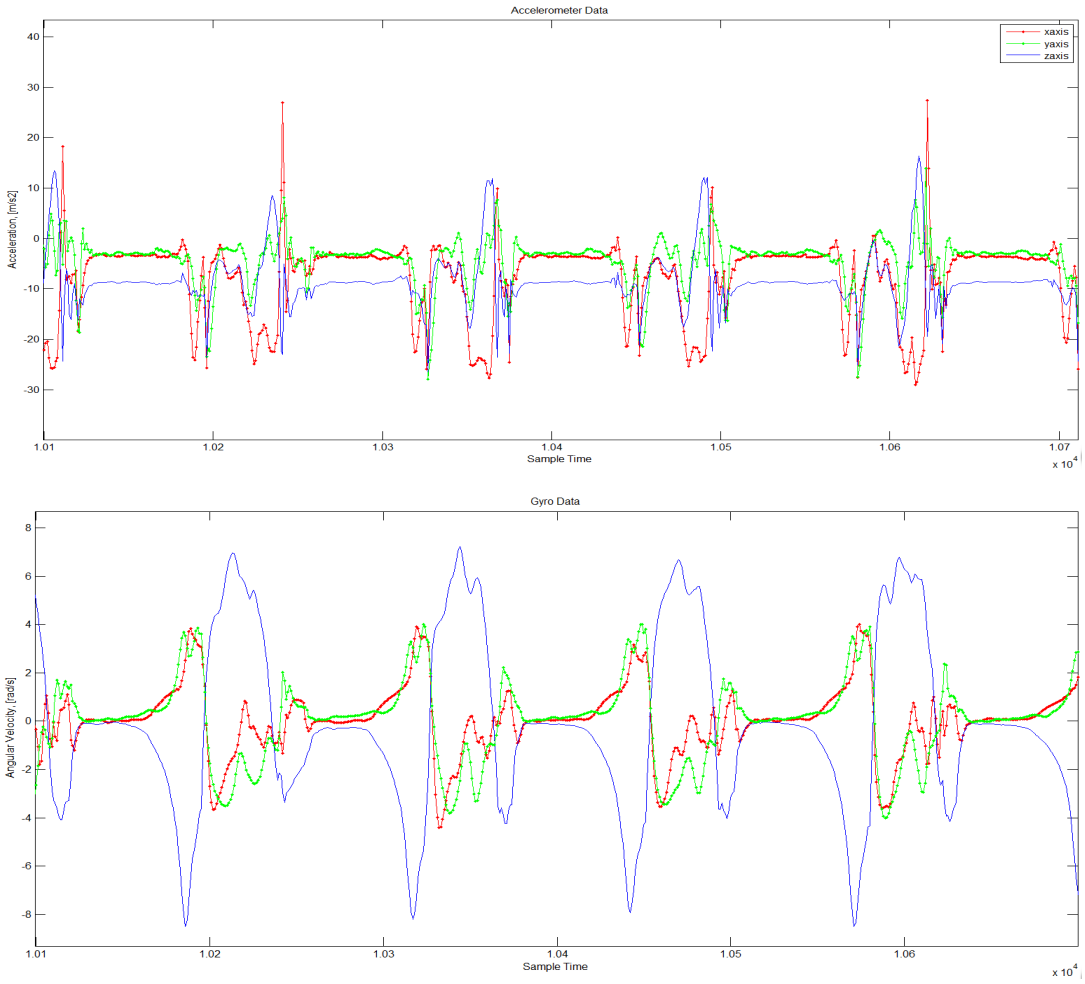


Figure 3.1 Example of raw accelerometer data and gyro data during a walking sequence

The stance phase detector adopted in this thesis consists of the following four logical conditions C1-C4, using both the accelerometer and gyro measurements. These four conditions must be satisfied (= 1) simultaneously to declare a foot as stationary for every sample epoch k.

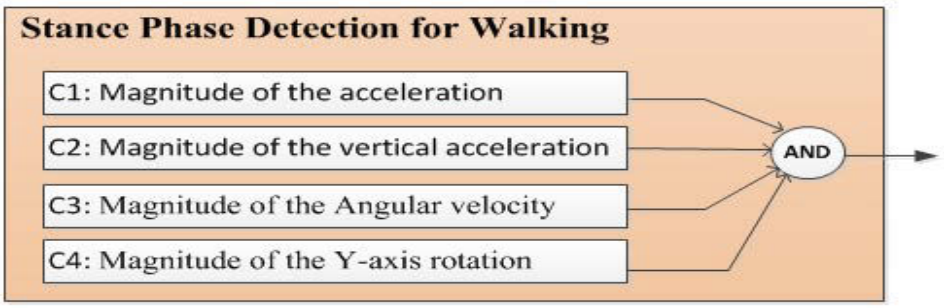


Figure 3.2 Walking stance phase detection

1) The magnitude of the acceleration $|a_k|$, where $a_{xk}\, a_{yk}\, a_{zk}$ are the magnitudes of the acceleration on x, y, z axis respectively:

$$|a_k| = \left[a_{xk}^2 + a_{yk}^2 + a_{zk}^2\right]^{1/2}$$

$$C1 = \begin{cases} 1 & threshold_{amin} < |a_k| < threshold_{amax} \\ 0 & otherwise \end{cases} \tag{3.1}$$

2) The magnitude of the acceleration on z axis $|a_{zk}|$:

$$C2 = \begin{cases} 1 & threshold_{azmin} < |a_{zk}| < threshold_{azmax} \\ 0 & otherwise \end{cases} \tag{3.2}$$

3) The magnitude of the angular velocity $|w_k|$:

$$|w_k| = \left[w_{xk}^2 + w_{yk}^2 + w_{zk}^2\right]^{1/2}$$

$$C3 = \begin{cases} 1 & |w_k| < threshold_{wmax} \\ 0 & otherwise \end{cases} \tag{3.3}$$

4) The magnitude of the angular velocity on y axis $|w_{yk}|$:

$$C4 = \begin{cases} 1 & |w_{yk}| < \text{threshold}_{ywmax} \\ 0 & \text{otherwise} \end{cases} \quad (3.4)$$

Since a foot firmly contacts the ground in a stance phase, the acceleration and rotation of the foot is zero over this duration. In an ideal condition the magnitude of total angular velocity $|w_k|$ in (3.3) and acceleration on the horizontal plane should be zero during a stance phase. However, the actual measurements are not zero due to vibration and the sensors bias, but are still lower than the given thresholds.

By observing the outputs of a shoe-mounted IMU, it is found that the z axis acceleration a_{zk} and the y axis angular velocity w_{yk} are the most significant indicators in pedestrian travelling. Due to the tilt of a shoe's surface, however, the measured $|w_{yk}|$ and $|a_{zk}|$ are not exactly 0 and g respectively. So the thresholds in equation (3.2) and (3.4) are set based on the average value of the z axis accelerometer and the y axis gyro outputs during an initial stationary period, plus a certain level of fluctuation to ensure its robustness. The initial stationary period is at the beginning of sensor data collection when the shoe-mounted IMU is in a stationary condition.

Our system can also roughly estimate stride length. According to [23], when a person walks, their feet alternate between a stationary phase and a moving stride phase, each lasting about 0.5 seconds, which corresponds to 50 samples, (the IMU sampling rate is set to 100 Hz). As we already record all the samples that have been detected to be zero velocity, we need to find out the start sample point S_k for ZUPT in each step by setting a condition that the interval of time should be over 0.5 seconds which could be defined by a mathematical expression $S_k - S_{k-1} > 50$. Since each sample point correspond to a position, I could find the position that the foot falls on the ground for each step, thus obtaining every stride length.

In this thesis, height constraint has been used. Similar to the concept proposed by Abdulrahim et al. [37], I assume that the changes in height are only caused by climbing stairs since the floor is level in most indoor environments. If a change in height over one step is above a threshold, we regard it as reaching a new level or a stair; otherwise the height will be maintained at previous level and this used as a measurement of the Kalman filter. This constraint works well in structured environments with only level floor and stairs, but fails in the case of gently sloping terrains.

3.2.3 ZUPT aided INS

In order to get high accuracy navigation solution, ZUPT is applied with an EKF to reduce navigation errors. The algorithm relies on detecting the foot stance phase in order to build a zero velocity based pseudo-measurement which is delivered to the Kalman filter. This allows EKF to correct the velocity error after each step, breaking the cubic-in-time error growth of the position and replacing it with an error accumulation that is linear in the number of steps. It is obvious that the stance phase appears in each step, thus the velocity errors can be reset almost periodically and will not be accumulated to the next step.

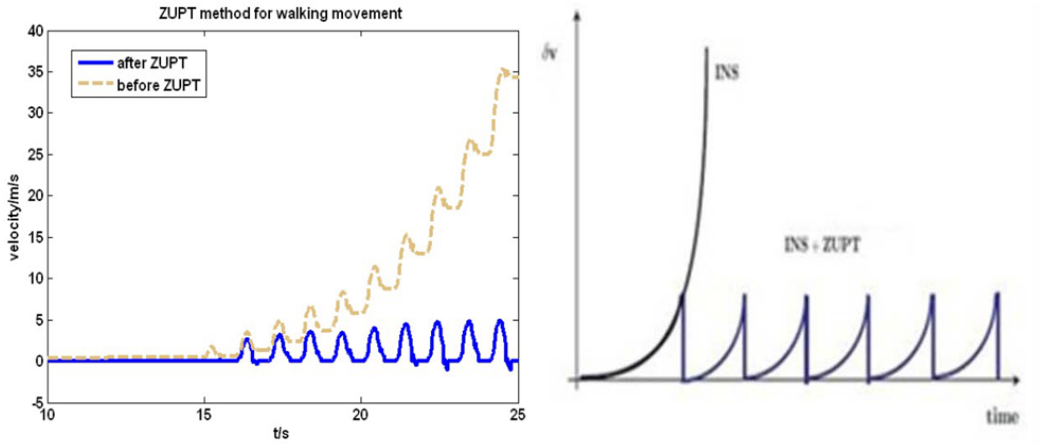


Figure 3.3 Performance before and after applying ZUPT (adopted from [38])

The framework of the proposed navigation system is illustrated in Figure 3.4. It has three main blocks: 1) INS Mechanization to compute the position, velocity and attitude; 2) Stance Phase Detection block to determine stance phases and apply ZUPT; 3) EKF block to estimate the error states and bound navigation errors.

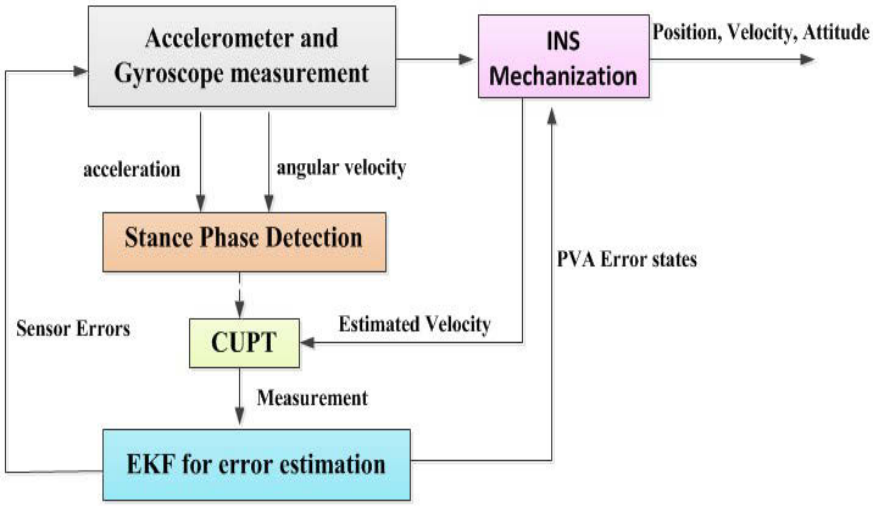


Figure 3.4 The main blocks in the framework used for pedestrian inertial navigation

It is important to mention that the navigation error vectors are reset to zero after the INS uses them to refine the current position, velocity and attitude. This is because those errors are already compensated and incorporated into the INS estimations. The only items that are maintained over time in the EKF are the sensor errors. Detailed explanation can be found in [23] and [29].

3.3Running aided ZUPT

Walking and running are the two most common forms of human gait. However, one of the most noticeable differences is the existence of a flight phase in running, rather than the double support phase that occurs in walking.

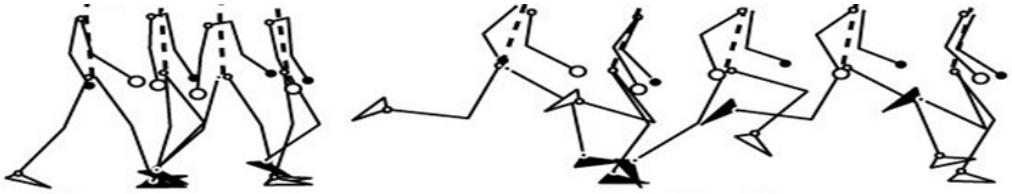


Figure 3.5 Walking VS running

Immediately, it is apparent that the gait cycle of a runner is shorter than that of a walker and that the profile of the gait cycle is significantly different. In the walking gait, the push-off and swing phases are mirrored by the heel strike and stance phases. While one leg is in the push-off phase, the other is in heel strike phase, and while one is in swing phase, the other is in stance phase. This symmetry in the walking gait shows the rhythmical transfer of weight between the two limbs. Conversely, during the running gait this symmetry does not exist. Instead, both legs can be in the swing phase at the same time, and heel strike occurs while the other leg is still in the air [39]

ZUPT is an effective way for inertial pedestrian navigation in a GPS-denied environment. The stance phase in each step provides zero velocity measurement into Kalman filter for inertial navigation error correction. Various zero velocity detectors (ZVDs) have been introduced for stance phase detection, some even use extra sensors. Most of the existing stance phase detectors only function for walking, but fail to detect stance phases in high-speed movement (running or jumping) with the same algorithm-setting. Compared with walking, the duration of the stance phase in running is shorter and the velocity in the stance phase is less qualified as zero velocity. Barely enlarging the threshold will introduce false stance phase detection. Even if the correct threshold value for detecting high-speed movement can be determined, how to switch the threshold values between activities with different velocities is still a problem. The ZUPT algorithm must be informed to change the threshold values when the operated movement starts to change.

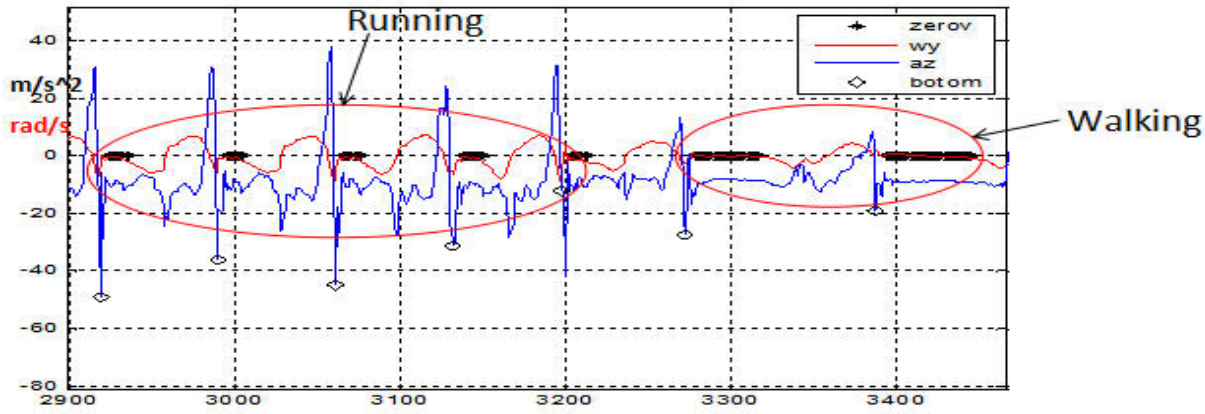


Figure 3.6 Duration of the stance phase in walking and running

Figure 3.6 shows one section of sensor outputs for running alternate walking case. The black dots stand for the sample points that have been detected to be zero velocity, the red and blue lines represent the y-axis measurement of the gyroscope and the z-axis is the measurement of the accelerometer. The symbol "◇"denotes a new step.

In this section I propose a novel ZUPT algorithm which can achieve robust pedestrian navigation in a wide range of locomotion. It is not limited to walking with normal walking speed and stair climbing, but also including running by combining foot fall recognition. Our stance phase detection consists of one footstep detector and two ZVDs. The footstep detector is used to mark when each new step arrives, and the first ZVD (ZVD1) can successfully detect zero velocity while walking by setting condition based on both the measurements of accelerometer and gyroscope. The second ZVD (ZVD2) is introduced for running with a relative larger threshold on gyroscope measurement only. How to switch the zero velocity detectors between activities with different velocities is still a problem as the ZUPT algorithm must be informed to change the threshold values when the operated movement starts to change. In our system, it is not necessary to distinguish walking or running. After a new step has been detected, ZVD2 is applied within the first half of the step; simultaneously ZVD1 is applied to

detect zero velocity. Once the ZVD1 conditions for zero velocity are satisfied, switch on the ZVD1 and switch off the ZVD2."

3.3.1 Footstep Detector

In pedestrian navigation with a shoe-mounted IMU, especially for running, it is important to know when a pedestrian takes a step. Footstep detector can not only mark the beginning of a gait cycle, but can also help to segment the pedestrian's gait cycle into four phases: stance, heel-off, swing, heel strike. A footstep detector mainly detects the heel strike after a foot completes the swing phase in the air, reaches the highest point and touches down on the ground. Once the heel has hit the ground, the deceleration of the forefoot is very dramatic and characterized by large changes in the acceleration profile. Thus, we simplify the characteristic as this: in the heel strike phase, the z-axis acceleration experiences a monotonic decreasing which is unique and obvious to detect so we denote it as a new step.

3.3.2 Stance Phase Detector for Running

The novel stance phase detection algorithm for running includes a footstep detector which indicates every new step, and two zero velocity detectors. ZVD1 was introduced in section 3.2.2. The ZVD1 uses four conditions based on the y-axis measurement of the gyroscope and the z-axis measurement of the accelerometer for the stance phase of the walking case when the foot is in contact with the ground and not moving, the outputs are flat for a period which can be easily detected once the four conditions are satisfied. However, for the running case, according to figure 3.6, z-axis acceleration is no longer flat enough to be a significant indicator. Besides, the short range of time of the stance phase also increases the difficulty of detecting zero velocity.

ZVD2 is designed for running based on the motion information of foot dynamics

where only the gyroscope measurement is used. ZVD2 uses only y-axis angular velocity which indicates the rotation of the ankle. Obviously, ZVD2 is less strict than ZVD1 and can easily result in false detection. Thus we apply these two ZVDs simultaneously in the half step length of a new step, once the more strict four conditions of ZVD1 are satisfied, ZVD2 is switched off and ZVD1 is applied. By strategically combining the three detectors, robust stance phase detection for both walking and running has been achieved. The zero velocity is determined once "norm (gyro) < E" is satisfied, where E is the threshold of energy. The process of the stance phase detector is shown in Figure 3.7.

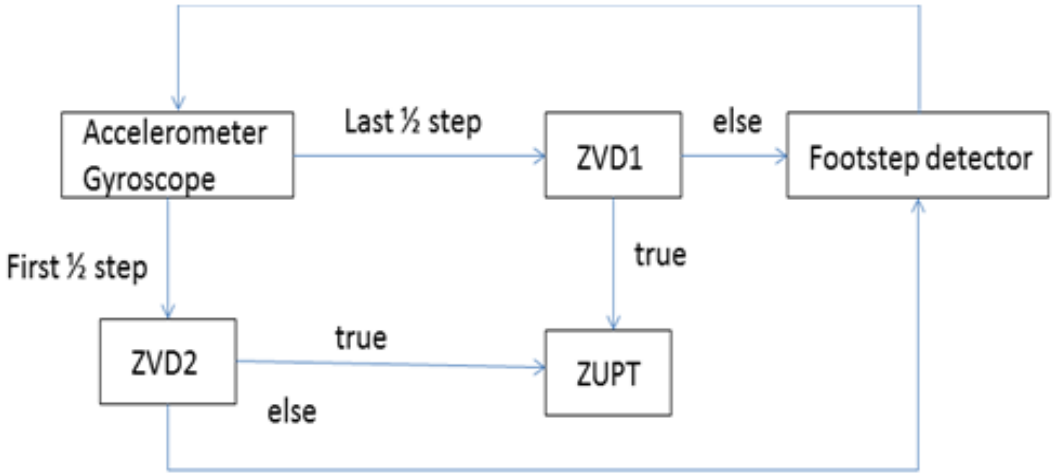


Figure 3.7 The process of the stance phase detector

Once the footstep detector decides that a new step arrives, in the first half step, we apply both ZVD1 and ZVD2 while the last half step using ZVD1 only. In our case, we do not need to classify the motion as walking or running. Since the threshold for running is larger than that for walking (Er>Ew), in the first half step, as long as the condition for walking is satisfied, we apply Ew afterwards, unless Ew is not satisfied during the whole half step in which case we only apply Er. For the last half step, ZVD1 is used to prevent some zero-crossings in the swing phase mistakenly detected or the case that the foot keeps still for a few seconds.

3.3.3 Swing Phase Detector

Another method is to combine the footstep detector and the proposed swing phase detector. A swing phase detector is designed to identify swing phases and prevent false stance phase detection in swing phases when enlarging the thresholds of the stance phase detector for the running case. The swing phase detector introduced here is based on the energy of rotation threshold TS. y_k^g is the angular velocity at the time instant k, W is the window size. The swing phase detector is combined with the footstep detector to separate a swing phase from other phases. As swing phases are identified, false stance phase detection in swing phases can be avoided.

$$T(k) = \frac{1}{W}\sum_k^{k+W-1}\left\|y_k^g\right\|^2 < T_s \tag{3.5}$$

Figure 3.8 shows the energy of rotation T in a couple of gait cycles. It should be noted that the footstep detector can also use this T to detect heel strikes. The energy of rotation goes through a big jump when the foot falls down to the ground, as Figure 3.8 indicated.

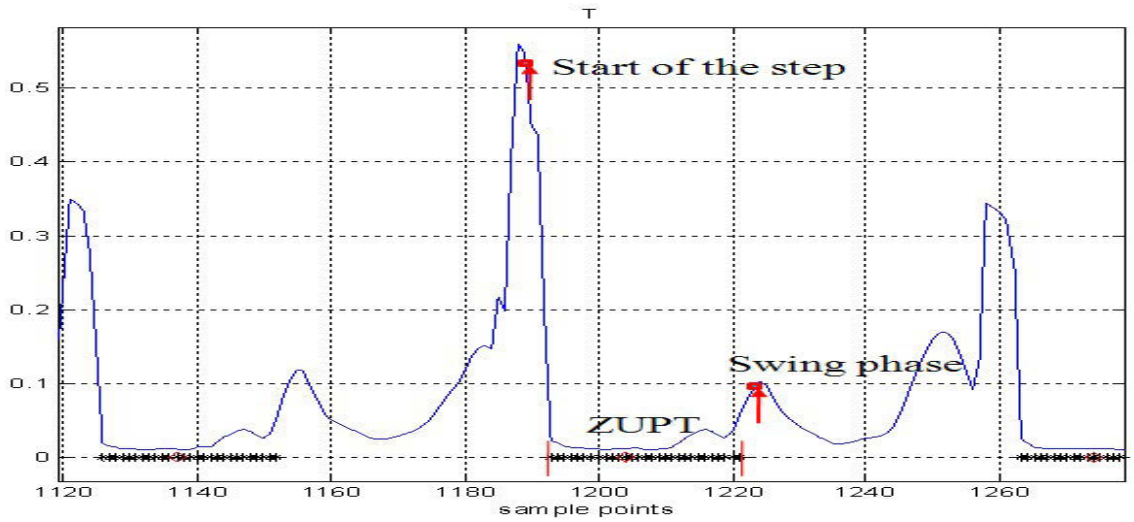


Figure 3.8 The energy of rotation T

3.4Experimental Results

The proposed pedestrian navigation system is tested for walking, running and stair climbing.

3.4.1 Hardware description

We implemented our pedestrian navigation system with the Navchip IMU from Intersense. The small size of Navchip makes it easy to attach to a boot and has no effect on walking. Table 3.1 lists the specification of Navchip (encapsulated within casing).

Table 3.1 Specification of Navchip IMU

Gyroscope Performance		Accelerometer Performance	
Range (deg/sec)	±480	Range(m/s^2)	±8
Noise Density ($^o/s/\sqrt{Hz}$)	0.004	Noise Density ($ug/\sqrt{Hz}$)	70
Bias Stability ($^o/hr, 1\sigma$)	12	Bias Stability (mg, 1σ)	0.1

The IMU should be mounted on a boot front surface, resulting in excessive vibration. During initial test, we discovered the phenomenon of excessive shock and acceleration measurement overflow. We took countermeasures such as placing shock absorbing padding and protective casing. Tests afterwards show reduced shock and improved measurement. Figure 3.9 shows the accelerometer measurement before (upper figure) and after the shock countermeasure. The boot mounted with the IMU was worn by a pedestrian for testing.

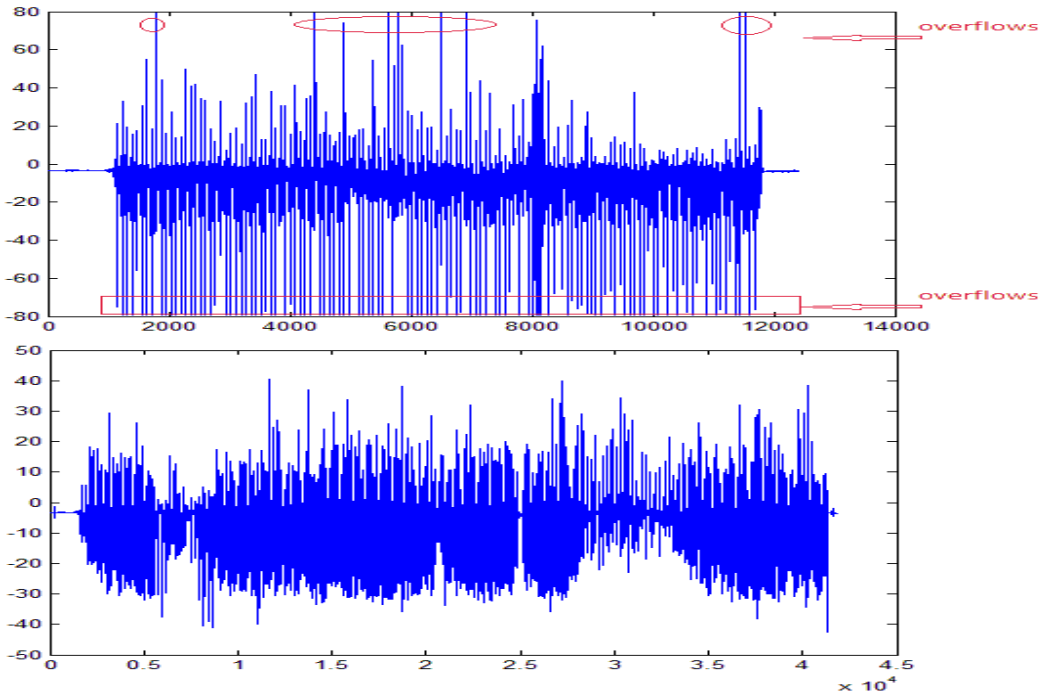


Figure 3.9 Comparison of acceleration before and after shock reduction

Figure 3.10 Shoe mounted with the IMU

3.4.2 Walking applying ZUPT results

In this section we present results in different scenarios:

- 2D closed-loop experiments
- Climbing stairs

a) 2D closed-loop experiments

In this test, a rectangular trajectory of about 25 meters in length, 15 meters in

width, resulting in a total path length of D = 80 m. The subject walked at the normal walking speed of 1m/sec for 3 loops. In all cases before walking, the subject stood still at the start point for about 4 seconds and stopped at the position where he started after each loop for 2 seconds. Since the start and stop point are exactly the same location in reality, the distance between the dark and blue dot in the below figures represent the error on the horizontal plane.

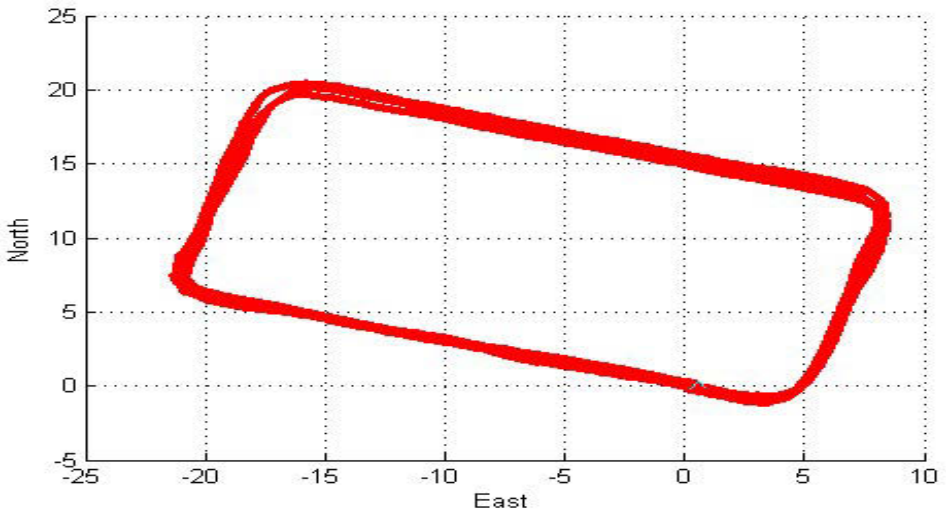


Figure 3.11 2-D closed loop experiments for trajectory 1

Another test was in our working office within a square-shaped path. The testing subject randomly walked through different working area.

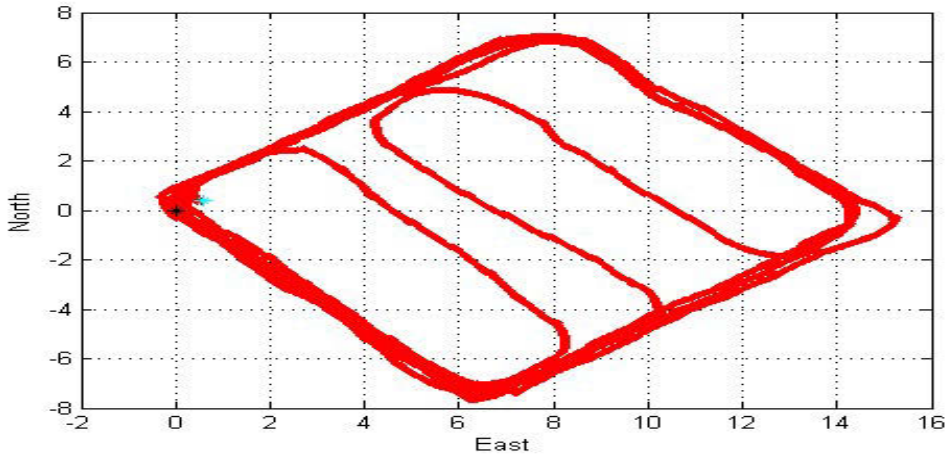


Figure 3.12 2-D closed loop experiments for trajectory 2

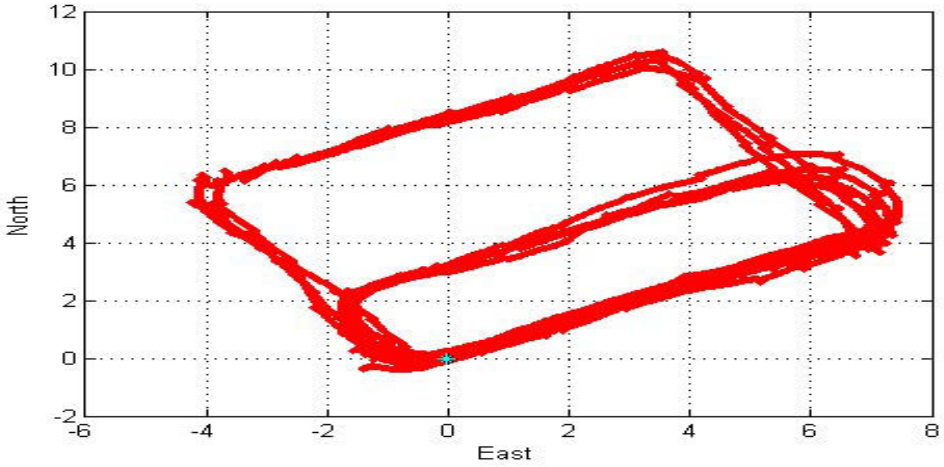


Figure 3.13 2-D closed loop experiments for trajectory 3

b) Climbing stairs

Here we present three types of walking in 3D environment.

The first test was simply going upstairs and downstairs.

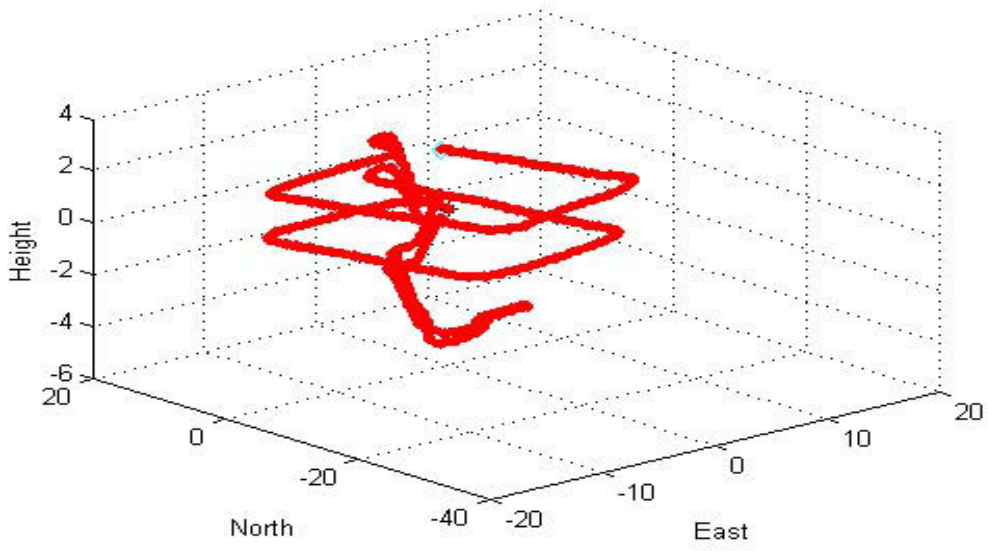


Figure 3.14 3-D closed loop experiments for trajectory 1

In the second type 3D environment walking, the subject started the walking at level 5, walked up a two-layer staircase to level 6. Then walked around the square corridor, went down to level 5 again through another staircase, followed by a walking along the corridor before the subject stopped at the exact original location.

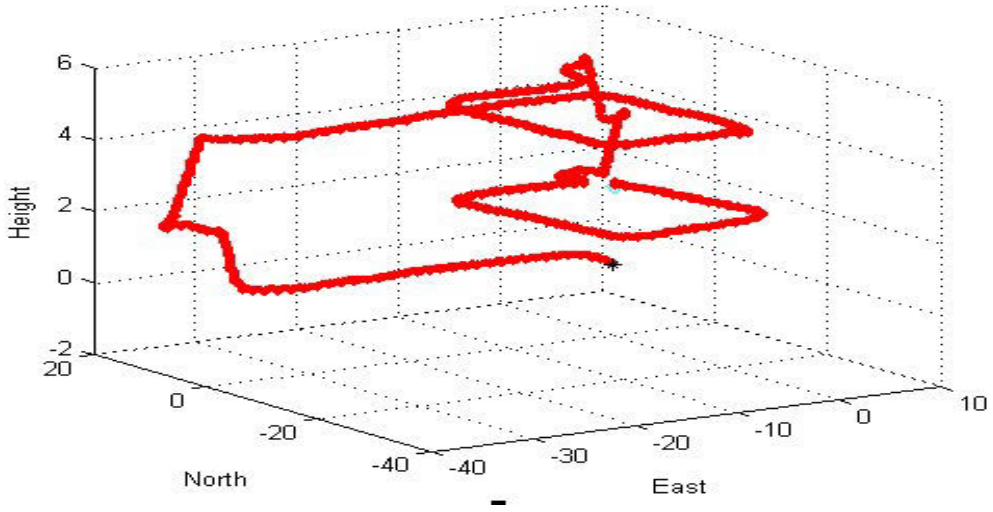


Figure 3.15 3-D closed loop experiments for trajectory 2

The last experiment followed a more complex and longer trajectory. The subject went up through a two-layer staircase, turned around and went down through UTS fire door to the bottom of the building, then climbed the stairs back to the place where he started followed by walking along the corridor.

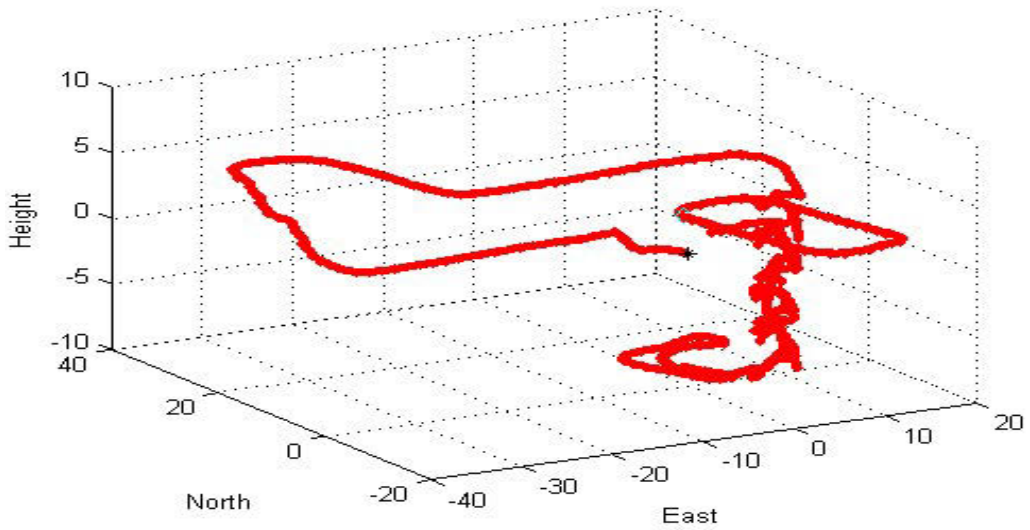


Figure 3.16 3-D closed loop experiments for trajectory 3

Table 3.2 shows the final return position errors for 3D and 2D experiments. For 2D tests, we set a constraint of walking on flat ground. Table 3.3 summarizes the checkpoint error in each loop. During the 2D walking test, the subject stopped at the target point (0, 0) for a while for each loop, in other words, the checkpoint is (0, 0) in

our system. In addition, we also list the errors that do not apply ZUPT during walking.

The absolute return position error was computed as stated in [36]

$$e_{xy} = \sqrt{x_e^2 + y_e^2} \tag{3.6}$$

$$e_z = \sqrt{z_e^2} \tag{3.7}$$

$$E_{xy} = \frac{e_{xy}}{D} \tag{3.8}$$

$$E_z = \frac{e_z}{D} \tag{3.9}$$

Where $x_e\ is$ the return position error in X direction, y_e is the return position error in Y direction, z_e is the return position error in X direction. D is the travel distance. The relative return position error E expresses the error as a percentage of the total travel distance.

Table 3.2 Return position errors

Event	Estimate Distance (m)	Relative (%)	
		X-Y plane	Z direction
3D Walk 1	354.8	0.30%	0.61%
3D Walk 2	240.8	0.14%	0.70%
3D Walk 3	277.4	0.01%	0.71%
3D Walk 4	339.2	0.08%	0.90%
3D Walk 5	408.6	0.75%	0.25%
Average	324.2	**0.26%**	**0.63%**
2DWalk 1	240.2	0.22%	3.6e-5%
2DWalk 2	206.4	0.21%	5.8e-6%
2DWalk 3	185.2	0.05%	1.8e-2%
Average	210.5	**0.16%**	**0.006%**

Table 3.3 Checkpoints of 2D closed loop trajectory

Event	Loop	Distance(m)	Error (ZUPT)	Error (no ZUPT)
2DWalk 1	Loop 1	79.9	0.40%	*
	Loop 2	160.8	0.20%	258.3%
	Loop 3	240.0	0.22%	933.1%
2DWalk 2	Loop 1	41.6	0.50%	*
	Loop 2	82.7	0.33%	154.7%
	Loop3	122.7	0.32%	352.6%
	Loop4	181.3	0.18%	696.2%
	Loop5	206.4	0.21%	953.2%
2DWalk 3	Loop 1	36.8	0.35%	*
	Loop 2	61.2	0.81%	208.3%
	Loop 3	82.8	0.58%	614.4%
	Loop 4	114.4	0.31%	1322.2%
	Loop 5	144.9	0.09%	2049.0%
	Loop 6	185.2	0.05%	2724.2%
Average			**0.33%**	

The symbol "*" in table 3.3 represents that for the first loop, we still apply ZUPT to reduce drift; for the loops afterward which follow the INS mechanization computing position, velocity and attitude without any correction results in quite large errors. Moreover, we can find that although the absolute return position becomes further away from the start point (0, 0) in each loop after applying ZUPT, the return position error degrades or remains at a relative low level with distance; all of the above indicate the effectiveness of ZUPT.

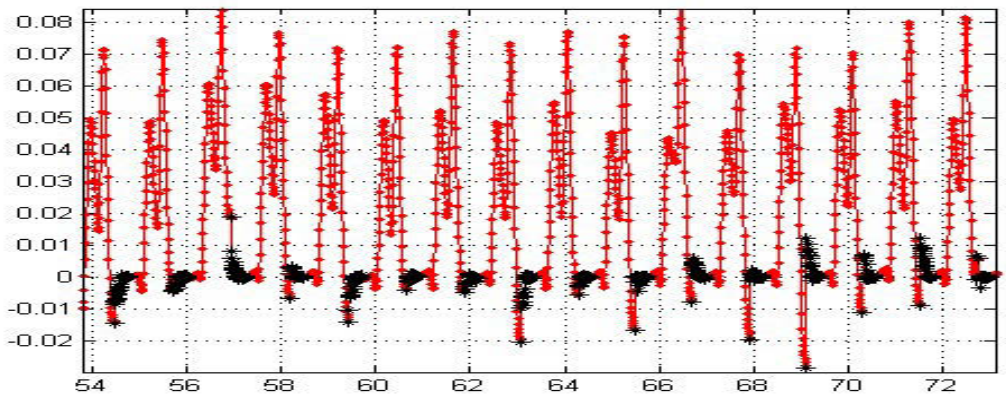


Figure 3.17 Height for 2D path

Figure 3.17 shows one section of the computed height for 2D level ground experiments, which is flat and keeps at a normal walking height. The black dot stands for the sample point which has been detected to be zero velocity, while the red one represents the height of a foot. Comparing 2D walking with 3D walking, errors in the horizontal and vertical plane are smaller than those of 3D walking; these big differences may due to the failure or missing detection of zero velocity when the subject is climbing stairs.

3.4.3 Running applying ZUPT results

In this section, we present running experimental results applying two methods respectively.

a) Applying ZVD1 & ZVD2

In this section we present three trajectories and each trajectory is conducted three times with different human gaits which include walking, running and walk alternated with running.

The first two trajectories are conducted on level ground; the last trajectory includes climbing stairs. Trajectory 1 is an “8” shape closed loop that the walking test is conducted for three loops. During our tests, the data will be blocked due to the sensor characteristics when experiencing consecutive overflow, thus our running and walk alternate run tests only follow the trajectory for one and two loops respectively.

Trajectory 2 follows a squared corridor and all the tests are conducted for three loops by another subject. The walking behaviour includes walking, walk alternate run, running and inverse run.

Trajectory 3 extends the 2D environment to 3D which includes climbing stairs.

This test combines the three gait behaviours together and during the tests, the subject walks for the first three loops, follows by a three loops' running, and ends up the tests with randomly walking alternate running for the last three loops.

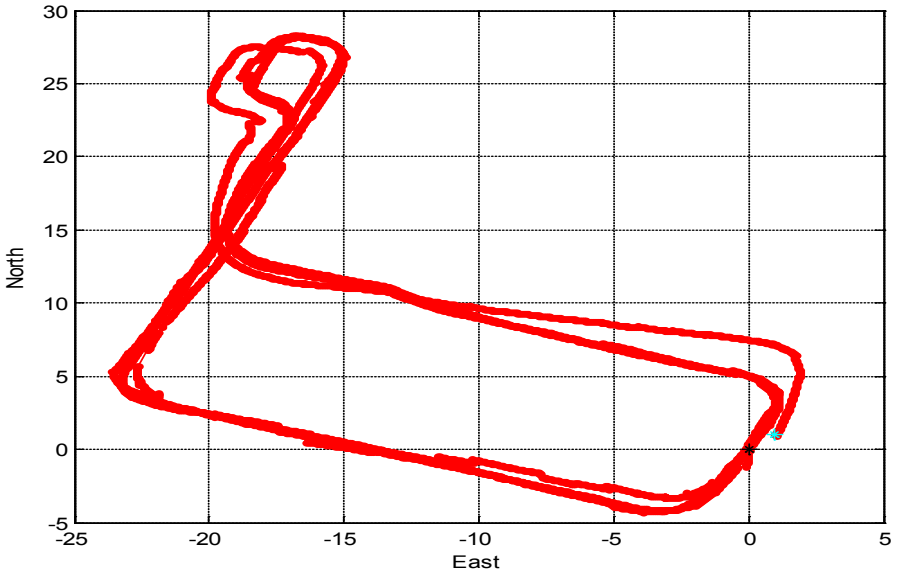


Figure 3.18 Trajectory 1

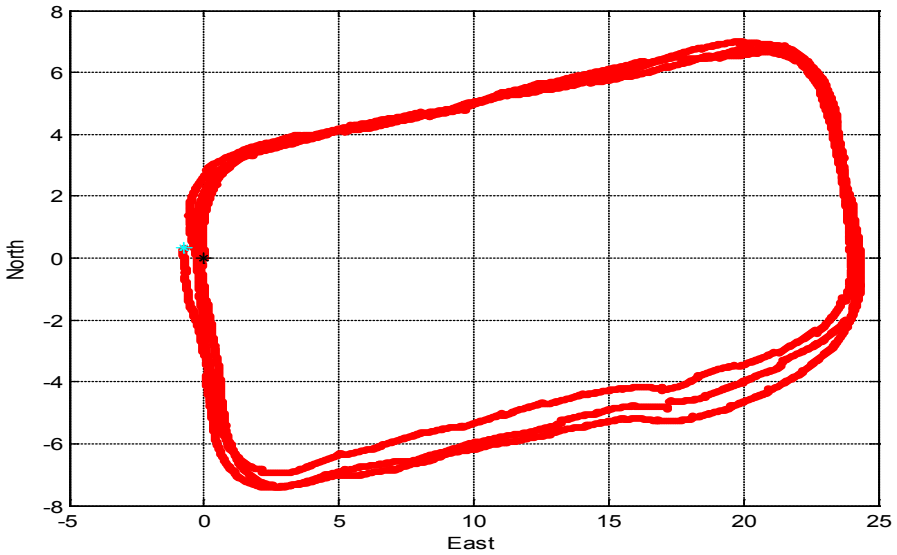


Figure 3.19 Trajectory 2

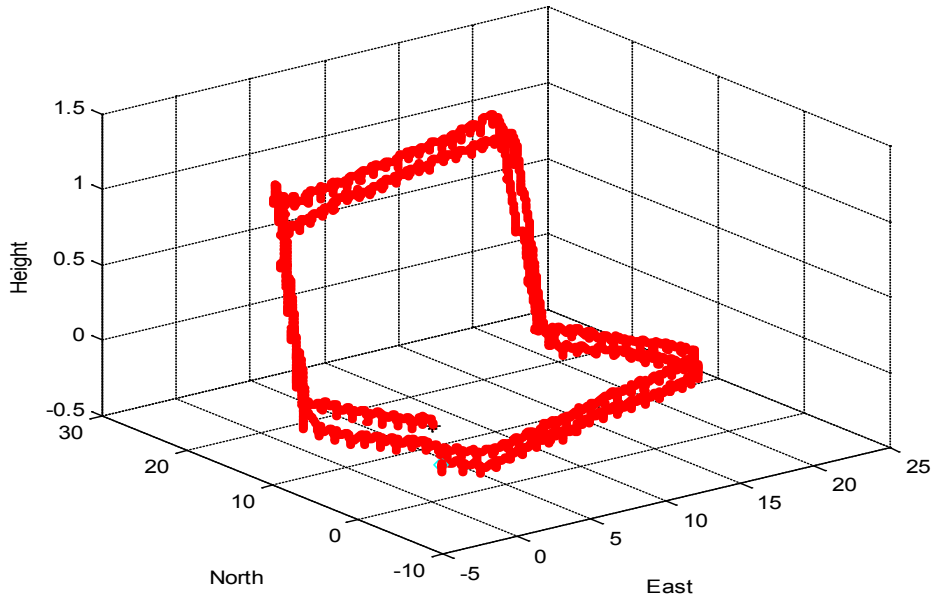


Figure 3.20 Trajectory 3

Table 3.4 summarises the final return position errors for each trajectory and table 3.5 also gives the comparison results of different gait behaviours.

Ideally the error of running should be larger than that of walk alternate running since the higher dynamics experienced during running shows more variability than in the walking gait, which expressed as fewer ZUPT applies. However, from our results as shown in table 3.5, we get a different conclusion. The main reason is that when the subject changes his gait randomly, sometimes it is easy to get the footstep detector confused, however, since ZVD1 is applied to prevent the swing phase, we can still get reasonable results.

Table 3.4 Return position errors

Event		Estimate Distance (m)	Relative (%)	
			X-Y plane	Z direction
Trajectory1	Walk	288.8	0.49%	8.7e-5%
	Walk & run	171.0	1.22%	0.17%
	Run	89.1	0.87%	0.0015%
Trajectory2	Walk	192.2	0.28%	1.9e-5%
	Walk & run	192.8	0.86%	0.037%
	Run	192.2	0.29%	5.5e-4%
	Inverse run	187.4	1.22%	0.001%
Trajectory3	Walk	175.1	0.39%	5.3e-5%
	Walk & run	175.3	1.04%	0.21%
	Run	175.2	0.79%	0.001%
Average		**183.5**	**0.72%**	**0.074%**

Table 3.5 Errors of Gait behaviour

	Relative (%)	
	X-Y plane	X-Y plane
Walk	0.39%	5.3e-5%
Walk & run	1.04%	0.21%
Run	0.79%	0.001%

Here we segment the last test into three segments since each gait is maintained for three loops and always ends back at the start point after each loop. In the X-Y plane, the red line draws the first three walking loops; the green line represents the running loops while the blue one is the trajectory conducted by walking alternate running. The figures also demonstrate the same results as table 3.5 that the running performance is better than that of walk alternate running.

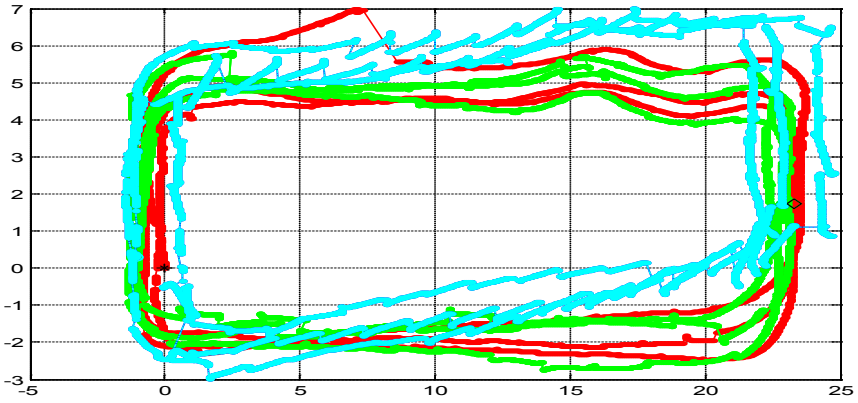


Figure 3.21 Trajectory3 including all gaits

In addition, because of the excessive shock, the bi-directional overflow will affect the accuracy to a large extend; although we have already taken measures to reduce the impact, it is still inevitable that the overflow would affect the INS mechanism computing vertical velocity and position. Figure 3.22 shows the effectiveness of shock reduction for walking behaviour; the figure below indicates that our current protective measure is not capable use with running. In figure 3.22, the z-axis acceleration is severely overflowed that we are considering using other large range sensors to do the tests. The diamond here is the detection of each new step by the footstep detector, while the black dots are the sample points that have been detected to be zero velocity.

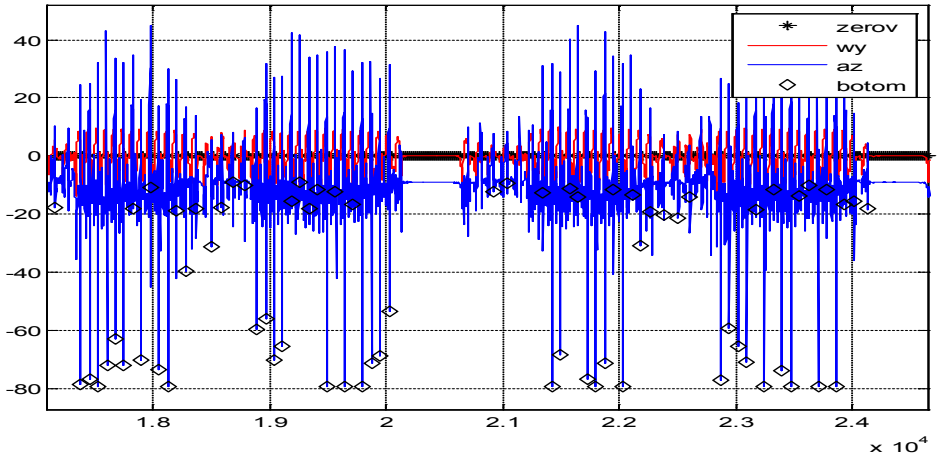


Figure 3.22 Raw accelerometer data

b) Applying Swing phase detector

In this section we present two trajectories and each trajectory is traversed several times with different human gaits which include walking, running and alternating between walking and running. All the trajectories are closed loop.

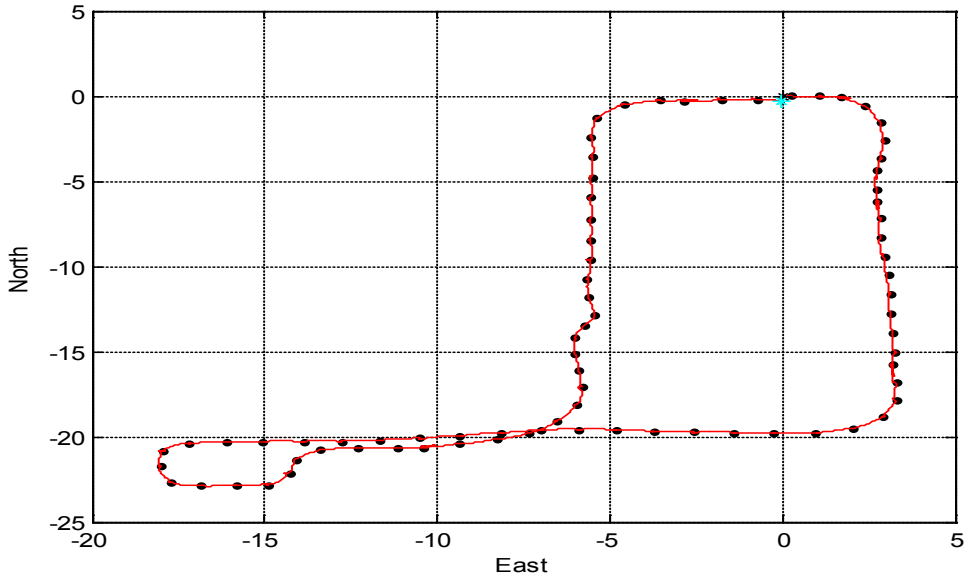


Figure 3.23 Trajectory 1

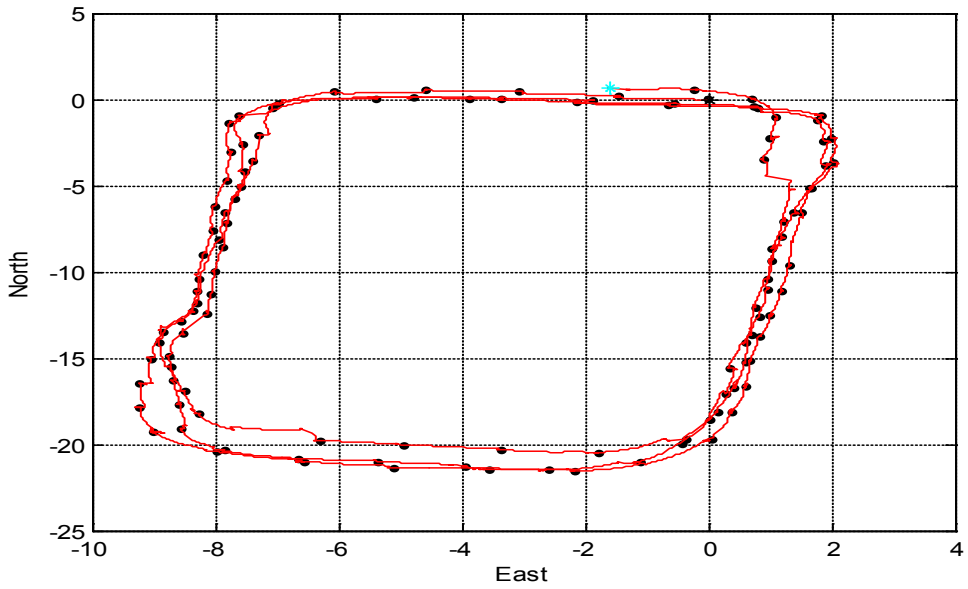


Figure 3.24 Trajectory 2

Table 3.6 summarizes the final return position errors for each trajectory and table 3.7 also gives the comparison results of different gait styles.

Table 3.6 Return Position Errors

Test		Estimated Distance (m)	Relative Position Error (%)	
			X-Y plane	Z direction
Trajectory1	Walk	292.1	0.24%	1.24e-4%
	Walk & run	173.6	0.96%	8.5e-5%
	Run	67.9	0.42%	0.002%
Trajectory2	Walk	195.7	0.20%	4.54e-4%
	Walk & run	173.2	0.95%	8.66e-5%
	Run	168.9	1.01 %	0.0015%
Average		**178.6**	**0.63%**	**7.08e-4%**

Table 3.7 Errors of Different Gait styles

Gait style	Relative Position Error (%)	
	X-Y plane	X-Y plane
Walk & run	0.96%	8.58e-5%
Run	0.72%	0.0018%

From our results as shown in table 3.7, I obtain the same conclusion as table 3.5. Comparing the Return Position Errors between table 3.6 and table 3.4, we can find that the swing phase detector is more efficient at detecting stance phase in the running movement. This is mainly due to the step length that the first method is based on is not reliable especially when the subject is frequently changing the speed and movement between walking and running.

3.4.4 Reference data processed results

In this section, we also give the results by processing the reference dataset provided in [40]. In their paper, the authors also give the processed results using two customary methods which fail when processing the running data. The table below shows the distance-travelled error normalized to the total distance travelled.

Table 3.8 Distance-Travelled Errors Normalized to the Total Distance Travelled

Data Set		Distance Travelled (m)	Positioning Error (%)
Walk	“ Walk 2D – Rectangle other direction”	28.68	0.050%
	“ Walk 2D – 8”	47.46	0.049%
	“ Walk 2D – On table”	24.42	0.43%
	“ Walk 2D – Patrick long”	2.79e2	0.10%
Run	‘Run 2D - Straight’	28.34	0.13%
	“ Run 2D – Circle”	73.94	0.26%
	“ Run 2D – Circle other direction”	40.41	0.20%

3.5Summary

In this chapter, we present the framework of ZUPT aided INS based on the Kalman filter and develop a robust algorithm for better step detection based on accelerometer and gyroscope measurements which can achieve high accuracy in both 2D and 3D environments with good performance. For 3D climbing stairs experiments, the average final return position error on the horizontal plane is around 0.26% of distance travelled, the height drift is about 0.63%, while the final return position error for 2D walking is even smaller, about 0.16% on X-Y plane by average and we can always get back to the original position for each loop with an average error of 0.33%. These position accuracies are better than or comparable to some of the highest position accuracy figures reported in the literature: 0.49% on the horizontal plane and 1.2% in the vertical direction [36], final error 30 cm with an exploration path length of 40 meters [24] and 0.4% accuracy reported in [28]. In addition, we also proposed a robust algorithm for a self-contained PDR system which can handle running cases. The test results show satisfactory performance, thus we can draw a conclusion that the algorithm is efficient in limiting the growth of IMU error.

Chapter IV

CONSTANT VELOCITY UPDATE

Most shoe-mounted MEMS inertial sensors based pedestrian navigation systems can perform well in modern concrete building environments with walking, running or stairs climbing by applying ZUPT to constrain the error drift. However, none of the mentioned ZUPT algorithms can handle the case when a subject is on a moving platform with velocity, such as in an elevator or on an escalator. This is due to the fact that they treat stance phases always at zero velocity even if a sensor is not stationary but moving with a constant velocity. Some pedestrian navigation systems may get good results on these platforms but with assisted sensors, such as a barometer or speedometer. To the best of the author's knowledge, none of the existing navigation systems can give reasonable navigation solutions based on only IMU when a person is on a moving platform.

This chapter will introduce a new algorithm for dead reckoning navigation named Constant Velocity Update (CUPT), which is an extension of the popular ZUPT. With a low-cost IMU attached to a user's shoe, the proposed algorithm can efficiently reduce IMU errors by detecting not only the stance phases during walking, but also the cases at constant velocity, such as in an elevator or on an escalator. The concept, design and test of a CUPT prototype are detailed in this chapter. Firstly, a brief introduction of the CUPT concept will be presented, in which we will explain the difference and connection between ZUPT and CUPT, followed by a detailed description of constant

velocity detection as well as CUPT application both for escalator and elevator. Finally, we will give experimental results applying CUPT for escalator and elevator cases.

4.1CUPT Introduction

As mentioned in [37], ZUPT will fail in two situations: elevator and escalator. This is due to the fact that the stance phases detected on a moving platform are not zero velocity. If a ZUPT algorithm is applied to a moving platform, it will regard the moving platform as having zero velocity and perform ZUPT. As the result, the navigation solution is obviously wrong. Use of other sensors or additional measurements may help to address these issues, but it will be complex and costly. Without additional sensors, we have proposed a CUPT algorithm that uses constant velocity as the additional measurement.

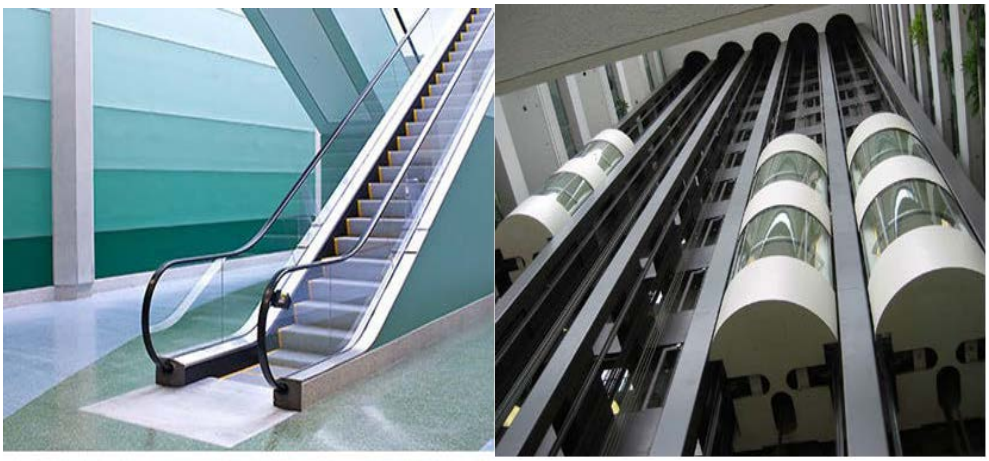


Figure 4.1 Escalator and elevator

ZUPT is just a special case in CUPT that the true velocity in the stance phases is zero. CUPT can detect the stance phases as ZUPT when stepping on the ground, and also a constant velocity when the pedestrian is standing in a moving elevator or on an escalator.

The core concept of the CUPT algorithm is to detect the velocity of a platform instead of assuming it at zero as in ZUPT. Obviously, the stance phase detection on a moving platform is exactly the same as the stance phase detection on the ground without considering a platform velocity. However, CUPT needs not only to identify a stance phase, but also to measure the velocity of the moving platform and decide whether the velocity is constant or not.

In principle, all the constancies that are correctly detected can be applied to EKF for estimated error states correction in inertial navigation, including but not limited to zero velocity. Here constant velocity is tackled; as a consequence the algorithm is called CUPT. The proposed algorithm can detect the stance phases as ZUPT when stepping on the ground, and also the constant velocity when standing in a moving elevator or on a moving escalator.

ZUPT	CUPT
Zero Velocity Update	Constant Velocity Update
Ground	*Moving platform(escalator/ elevator)*

In summary, the motivation of this technique is that when a person is on an escalator or steps into an elevator, it cannot be regarded as zero velocity, since the foot is stationary relative to the moving platform but not the ground or on the ground.

So firstly we need to emphasize that ZUPT is just a special case of CUPT, the true velocity condition is exactly zero.

Secondly, the stance phase detection for ZUPT has in common with CUPT since the foot is statically contacted with the ground in both cases.

The difference is that ZUPT is for a person walking on the ground while CUPT is to handle the case on a moving platform.

In short, in order to apply CUPT, we firstly need to apply the stance phase detector to declare that the foot is static, and then determine that the foot is on a moving platform.

4.2 Constant Velocity Detection

CUPT is applied in the EKF if the detector decides that the foot is stationary as well as the foot is on a moving platform. Here "stationary" means not only that the foot is contacted with ground and not moving, but also keeps static relative to the moving platform. Whatever update technique is applied, the characteristic of the sensor outputs is the same; for this reason, the stance phase for both cases can be detected using the ZVD1 (see section 3.2.2).

After the ZVD1 indicates that the foot is stationary, we need to decide whether the foot is on the ground or on a moving platform with velocity. In summary, the stance phase detection for CUPT is the same with ZUPT since the foot is statically contacted with a platform in both cases. But for CUPT the velocity of the moving platform needs to be measured. ZUPT is applied when a person is walking on the ground while CUPT is to handle the case on a moving platform. In order to apply CUPT, firstly we need to use the stance phase detector to declare whether the foot is stationary, then determine

whether the foot is on a moving platform, measure the constant velocity of the platform and apply CUPT to correct the errors.

4.2.1 CUPT for Elevators

Take an elevator as example, Figure 4.2 shows the measurement of the vertical accelerometer AccZ and the estimated velocity. A pedestrian walked into the elevator and stood still when the elevator was moving. As indicated by the velocity in Figure 4.2, the elevator went through a period of acceleration, kept at constant velocity and decelerated before it was stationary again. The constant velocity appears once the elevator finishes accelerating and before the elevator starts slowing down. In this circumstance, the most significant indicator is the acceleration in the vertical direction.

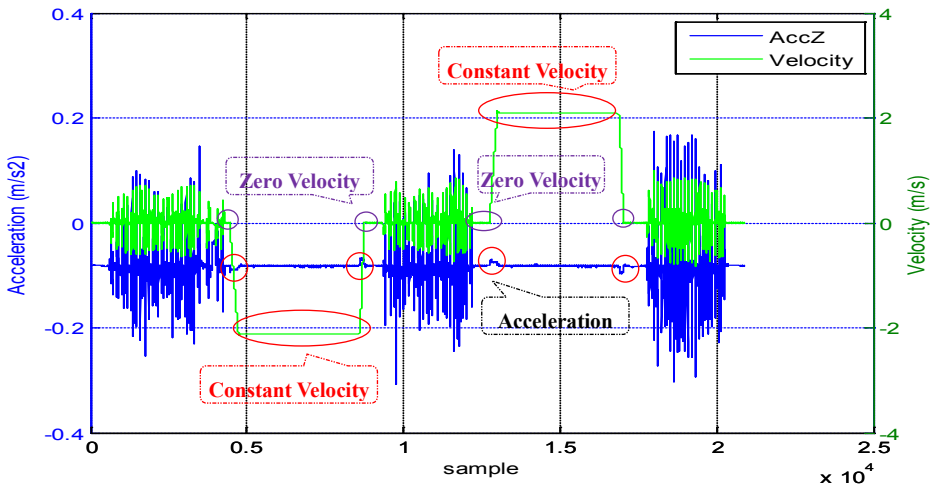


Figure 4.2 Indication of motion in an elevator

When the elevator starts to move, the acceleration in the z axis experiences a jump which can be easily distinguished from the normal gaits. When this special case of acceleration has been detected, the velocity is calculated by the inertial navigation algorithm without velocity measurement update. Since the pedestrian stands in the

elevator, the zero horizontal velocity can be added as a measurement to the EKF, referencing to (4):

$$z = \hat{x} - v_{XY} \tag{4.1}$$

After the elevator finishes accelerating, the velocity keeps constant before the elevator begins to slow down. During this constant velocity section, CUPT is applied to correct navigation errors. So the measurement z is $\hat{x} - v_{constant}$. The decelerating section is similar to the accelerating section, and the zero horizontal velocity update is applied.

4.2.2 CUPT for Escalators

For the case of being on an escalator, the horizontal velocity is the most significant indicator. When a foot steps onto the escalator and keeps pace with the escalator going up or down, a constant velocity is determined by the instance the foot sets on the escalator. So in this case, the measurement z is $\hat{x} - v_{constant}$. Escalators are often powered by constant-speed alternating current motors and move at approximately 0.3–0.6 m/s. Once the horizontal velocity is over a threshold in the stance phase, it is assumed that the foot is on an escalator and we apply CUPT as measurement of the EKF.

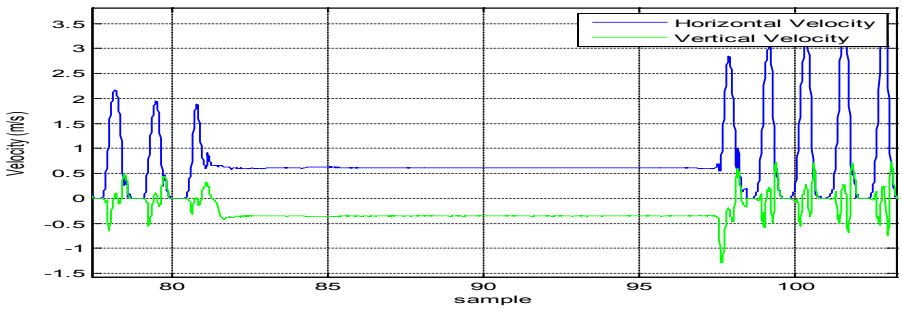


Figure 4.3 Motion in an escalator with CUPT

Figure 4.3 shows the horizontal and vertical velocity of a foot on a moving escalator. We can see that ZUPT is applied when the pedestrian walks on the ground; the observation is zero velocity during the stance phase, while the velocity keeps constant when the pedestrian stands on the escalator and CUPT corrects the velocity error using the constant velocity as the observation.

4.3 Experimental Results

In this section we present results in two scenarios and all the trajectories are closed loop.

4.3.1 Experiments in elevator

The test was conducted in a tower building in UTS which has multi elevators and 27 levels. The subject walked for a while before he went into the elevator. When the door was closed, the subject stopped walking and stayed still until the door opened again. Which level the elevator chose to stop at and which elevator the subject chose to take are all random. Since the start (dark dot) and stop point (blue dot) are the same location in reality, the distance between the dark and blue dot in the figures below represent the relative error.

The three trajectories are as follows:

a) Start walking at level 4; take an elevator up to level 26 followed by walking and a trip down to level 4 by another elevator.

b) Start walking at level 4; take an elevator up to level 26 followed by walking and down to level 2 by the same elevator and then back up to level 4 by another elevator.

c) Start walking at level 4, take an elevator up to level 24 and climb the stairs to level 26, go down to level 16 by another elevator, walk outside the elevator and back down to level 4.

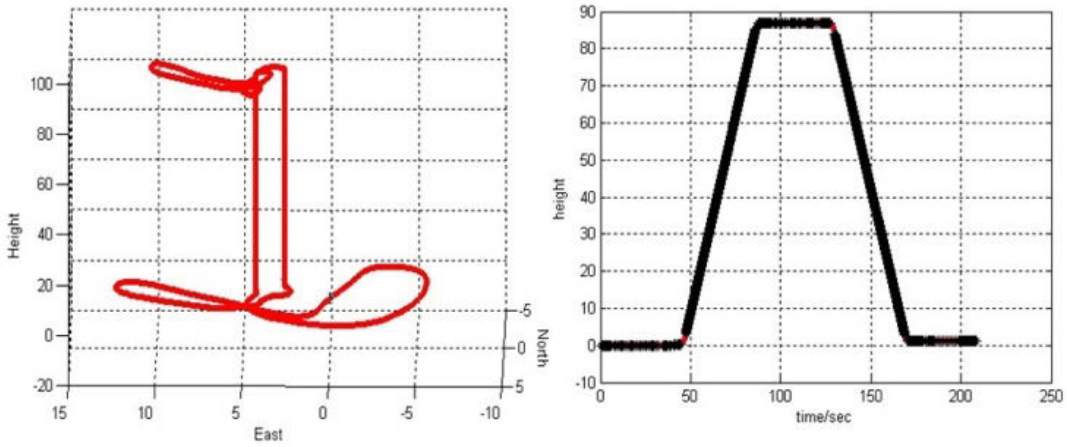


Figure 4.4 Trajectory1 of elevator test

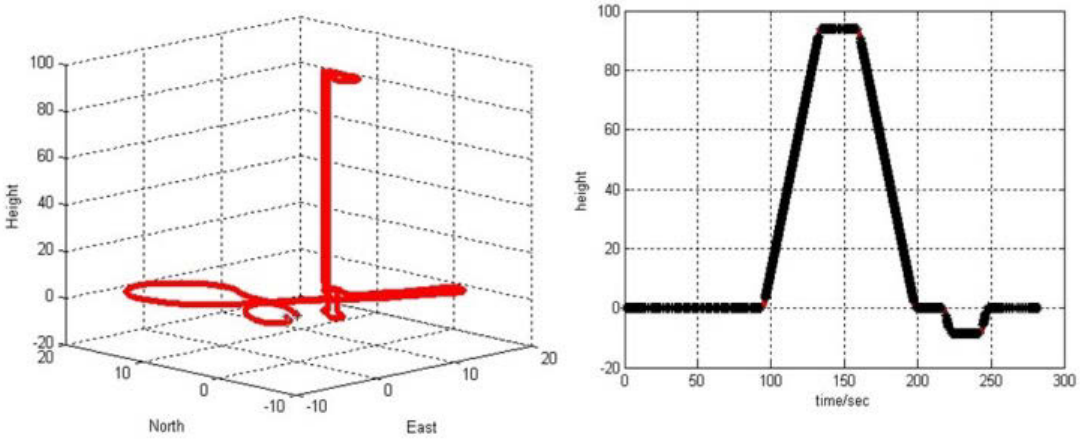


Figure 4.5 Trajectory2 of elevator test

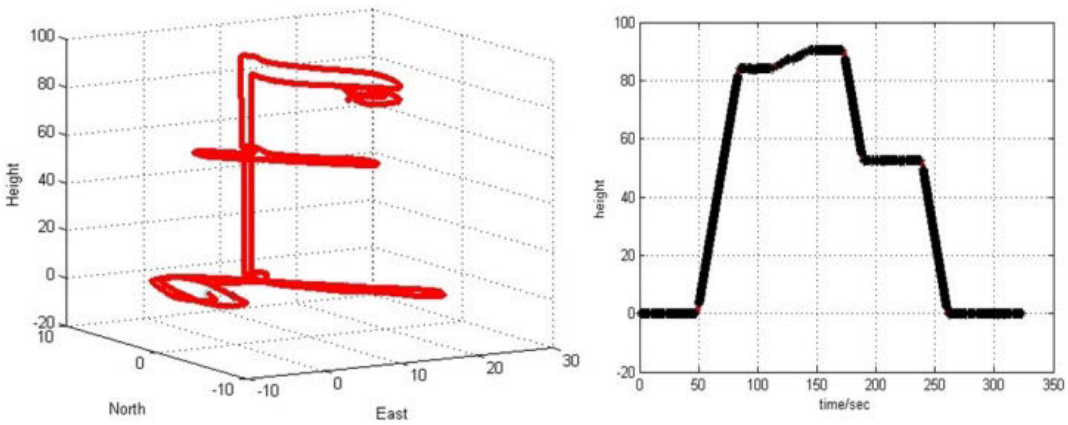


Figure 4.6 Trajectory3 of elevator test

The left figure is the 3D view of the trajectory while the right figure represents the height change over time drew by the red line; the black dots indicate the sample points to which have been applied CUPT or ZUPT.

4.3.2 Experiments on escalator

Starting at the lower level, the subject walked for a while and stood on a rising escalator with the IMU mounted on one foot. Then the subject got off the escalator on the middle level and walked to another rising escalator to the top level, and then back down to the start point via two descending escalators.

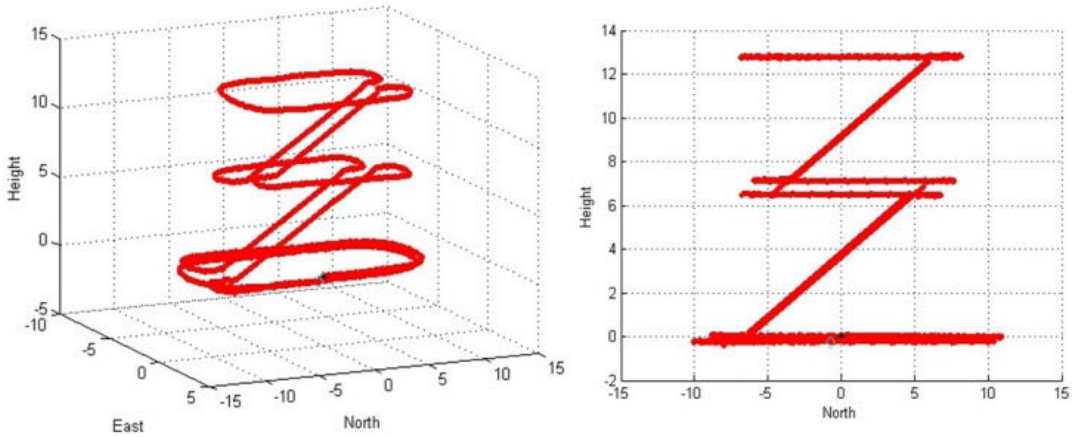


Figure 4.7 Trajectory of escalator test

Figure 4.7 is the 3D view of the trajectory, which clearly represents the four escalators and the three levels walked.

Figure 4.8 shows one section of CUPT computed velocity for the case in an elevator. The elevator went through a period of accelerating, kept at a constant velocity, and then decelerated to zero velocity. The green line draws the velocity of the vertical direction and the blue line represents the raw accelerator measurements on the z axis.

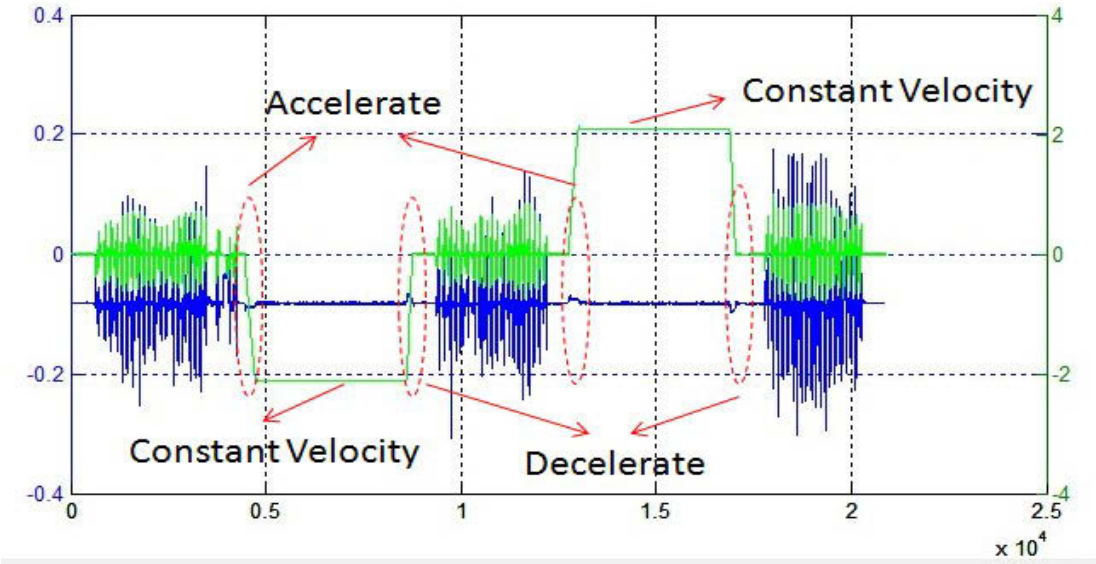


Figure 4.8 Indication of motion in an elevator

Figure 4.9 shows the steps walking on the ground applied ZUPT but no update for standing on the escalator, which results in large errors in velocity without IMU drift correction. Figure 4.10 shows one section of CUPT computed velocity for the same case, which keeps at a constant velocity going along with the escalator until it steps down to the ground. The black dots stand for the sample points that have been detected to be constant velocity or zero velocity, the red, blue and green lines draw the velocity of the north, east and vertical direction respectively.

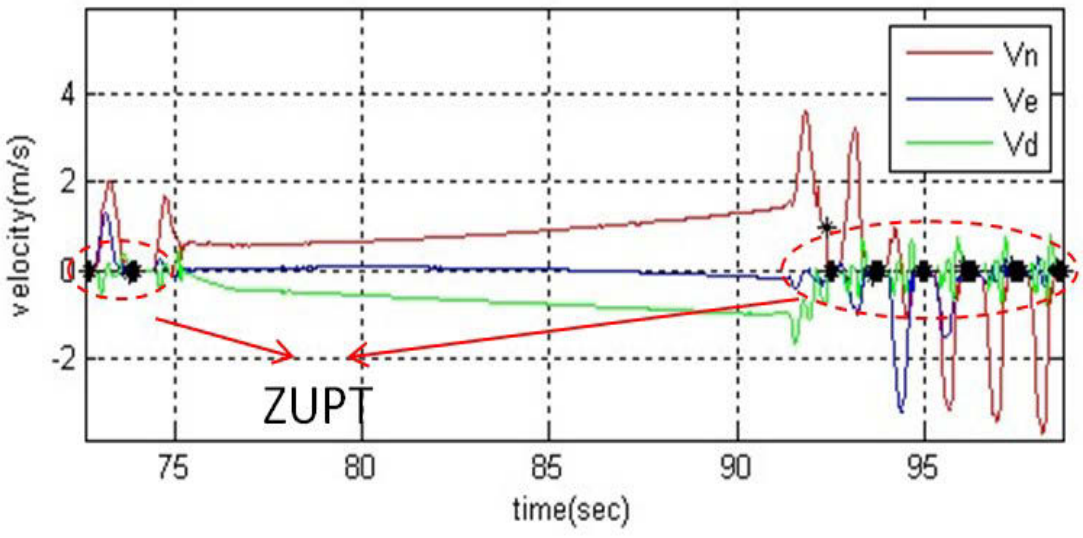


Figure 4.9 Motion in an escalator without update

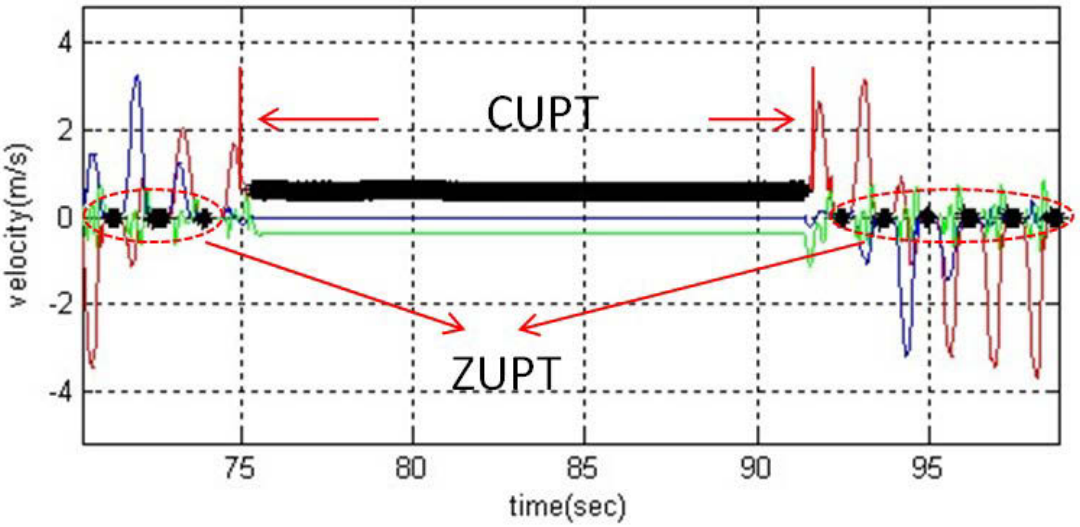


Figure 4.10 Motion in an escalator with CUPT

Table 4.1 and 4.2 summarize the return position errors for elevator and escalator experiments respectively.

Table 4.1 Return position errors of elevator

Elevator	Estimate Distance (m)	Relative (%)	
		X-Y plane	Z direction
Trajectory 1	248.3	0.17%	0.47%
Trajectory 2	324.7	0.39%	0.063%
Trajectory 3	373.0	0.56%	0.02%
Average	**315.3**	**0.37%**	**0.18%**

Table 4.2 Return position errors of escalator

Escalator	Estimate Distance (m)	Relative (%)	
		X-Y plane	Z direction
Trajectory 1	223.3	0.34%	0.11%
Trajectory 2	259.1	0.56%	0.34%
Trajectory 3	232.4	0.64%	0.37%
Trajectory 4	210.8	0.22%	0.10%
Average	**231.4**	**0.44%**	**0.23%**

Although these test results are reasonably good, there is still large room to further improve the proposed algorithm to achieve better performance and handle more

challenging situations. Processing elevator data during accelerating and braking sections, CUPT follows the INS mechanization computing position, velocity and attitude without any update and correction until constant velocity is detected. However, in some cases it is hard to detect constant velocity. The motion in accelerating and braking sections is with variable acceleration. Especially during braking, an elevator firstly slows down to a low speed and keeps constant for a while, then decelerates to zero velocity, which needs more specific analysis. Moreover, the frequency of elevator stops also has an influence on the accuracy of height measurement. As CUPT only implements at the instance when the accelerating period ends, if an elevator stops frequently, its acceleration takes more time and is not obvious for detection, which could result in false or missing constant velocity detection.

While for the escalator, the vertical angle of the escalators was predetermined to compute the vertical velocity. This angle could be measured by the sensor with more sophisticated investigation. Although the height relative error in table 4.2 is small, the horizontal relative error is not so good. This may due to the change of vertical angle at the two ends of escalators where the staircase keeps moving forward but no longer goes upward or downward, where the predetermined angle introduces error for measurement update.

4.4Summary

This chapter presents a CUPT algorithm for a self-contained PDR system which works on a moving platform with good performance. For elevator experiments, the average return position error on the horizontal plane is around 0.37% of the total

distance travelled, the height drift is about 0.18%. The return position error for escalator is a little larger, that is 0.44% on the horizontal plane by average and 0.23% on the vertical. From the test results, we can draw a conclusion that the CUPT algorithm is very efficient in limiting the growth of IMU error. Our robust stance phase detection and constant velocity update algorithm shows satisfactory performance. This CUPT algorithm could give an accurate navigation solution simultaneously as a pedal robot or pedestrian is walking.

Chapter V

STEPWISE SMOOTHING

ZUPT is an effective way to correct low cost IMU errors when it is foot-mounted for pedestrian navigation. The stance phase in steps provides zero velocity measurement for inertial sensor error correction. As the errors of IMU estimated position and velocity grow rapidly with time between each correction, ZUPT applied at each step leads to sharp corrections and discontinuities in the estimated trajectory. For motion analysis and visualization, these large corrections are undesirable. Consequently, the implementation of smoothing for ZUPT-aided INS is considered in order to eliminate the sharp corrections.

To eliminate the sharp corrections and discontinuities, David et.al had developed an open loop step-wise smoothing algorithm to aid ZUPT which can efficiently eliminate the sharp corrections and discontinuities at the end of the step [41]. We are inspired by their work and propose a new stepwise closed loop smoothing algorithm to implement on our previous pedestrian navigation system, which can improve the navigation accuracy and smoothness. Firstly we will introduce the basic principle in smoothing and then discuss the Rauch-Tung-Streibel (RTS) smoother. The smoothing algorithm and segmentation for subsequent data processing will be explained followed by the experiments and results.

5.1Smoothing Review

Smoothing uses the fact that the sensor data can also be used in a reverse time

fashion. The goal of smoothing is to determine an optimal estimate utilizing information before and after the current time. Meditch classified smoothing problems into three categories, i.e. fixed-point, fixed-lag and fixed-interval [42]. Fixed-point smoothing is used when we are interested in the states at specific points such as orbit injection time of a satellite or initial condition of the reaction substance in a chemical process. Fixed-lag smoothing can be used when the existence of a fixed-lag of the estimate does not impose intractable problems. Thus, the method appears attractive primarily in communication and telemetry data reduction problems. Fixed-interval smoothing can be used in most surveying applications because surveying is typically amenable to post-processing where best position information is pursued for all measured points. Hence only fixed-point smoothing will be discussed in more detail [17].

Smoothing is a non-real time estimation method that uses the measurement data before and after time tk, for computing the state estimates at time tk. An optimal smoother can be thought of as a combination of two optimal filters, namely the forward filter and the backward filter. The forward filter provides state estimates based on all the measurements before the current time tk, while the backward filter provide state estimates based on all the measurements after time tk.

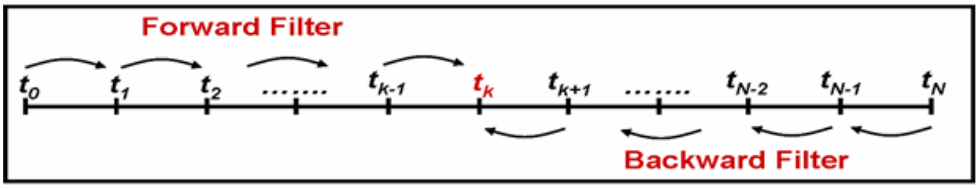


Figure 5.1 Forward and Backward Filters (adapted from [43])

Suppose that we combine a forward estimate $\hat{x}_k^f$ of the state and a backward

estimate $\hat{x}_k^b$ of the state to get a smoothed estimate of x as follows:

$$\hat{x}_k^s = A\hat{x}_k^f + B\hat{x}_k^b \tag{5.1}$$

Where, the 'hat' signifies the estimated state, the superscript 'f' represents the forward filter, 'b' represents the backward filter, 's' represents the smoother. x_k is the state vector and A, B are the weighing matrices.

By replacing each state estimate by its true value plus the estimation error in Equation (5.1), an equation in terms of state error vector can be derived and is given by Equation (5.2):

$$\delta\hat{x}_k^s = (A + B - I)x_k + A\,\delta\hat{x}_k^f + B\,\delta\hat{x}_k^b \tag{5.2}$$

where δ represents error perturbations. The corresponding error covariance matrix associated with Equation (5.2) is given by:

$$P_k^s = A\,P_k^s\,A^T + B\,P_k^b\,B^T \tag{5.3}$$

where P_k is the state error covariance matrix at time tk.

For the smoother to be unbiased, the first term in Equation (5.2) should be zero [44], as stated by Equation (5.4):

$$A+B-I = 0 \tag{5.4}$$

By minimizing the smoother error covariance, and using the relation given by Equation (5.4), the weight matrices A and B are obtained as given by Equations (5.5) and (5.6)

$$A = P_k^b\,(P_k^f + P_k^b)^{-1} \tag{5.5}$$

$$\mathrm{B} = \mathrm{A} - \mathrm{I} = P_k^f \, (P_k^f + P_k^b)^{-1} \tag{5.6}$$

Thus by substituting Equations (5.5) and (5.6) into Equation (5.3), the expression for the smoother error covariance is obtained, and is given by Equation (5.7):

$$P_k^{s\,-1} = P_k^{f\,-1} + P_k^{b\,-1} \tag{5.7}$$

$$\hat{x}_k^s = P_k^s \left(P_k^{f\,-1} \hat{x}_k^f + P_k^{b\,-1} \hat{x}_k^b \right)$$

$$= P_k^s \left(P_k^{f\,-1} \hat{x}_k^f + P_k^{b\,-1} \hat{x}_k^f - P_k^{b\,-1} \hat{x}_k^f + P_k^{b\,-1} \hat{x}_k^b \right)$$

$$= \hat{x}_k^f + P_k^s P_k^{b\,-1} (\hat{x}_k^b - \hat{x}_k^f)$$

The above implies that the optimal smoothed information is always better than the information in either the forward or backward filter. Hence optimal smoothing will always improve the estimation error.

Smoothing is widely used in a GPS/INS integrated system which is typically useful during GPS outages. The principle of smoothing during GPS outages is explained in Figure 5.2. The INS operates in a prediction mode during GPS outages. When a filter operates in prediction mode, the estimation error grows since there are no measurement updates and thus the errors keep accumulating. The forward INS filter will accumulate error forward in time (brown line), while the backward filter will also accumulate error (green line), but backward in time. Thus, when the two solutions are combined using the above algorithm, a smoothed estimate is obtained (red line). Clearly, smoothing reduces the errors significantly during the data outage interval, relative to either of the forward or backward estimate.

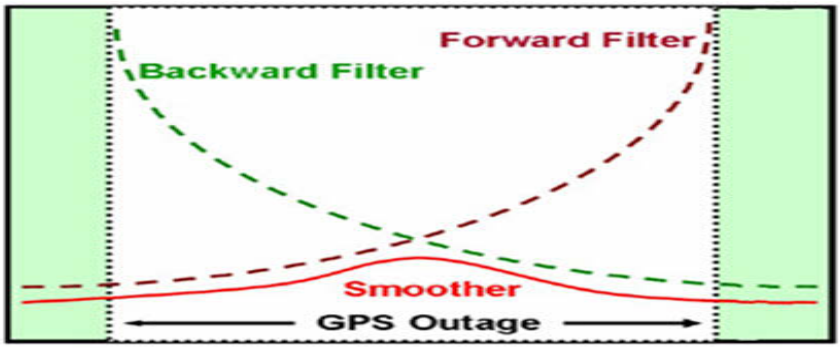


Figure 5.2 Errors during GPS outage (adapted from [45])

5.2RTS Smoother

The RTS smoother is a two-pass smoothing algorithm. The forward pass is the same as the regular Kalman filter algorithm. During the prediction and update steps, the states and covariance are saved for use in the backwards pass. After the forward pass, the RTS smoothing begins from the end of the mission and moves back to the starting point of the data set. In the backwards pass, the smoothed state estimates and covariance propagate using the states saved in the forward pass [46]. The smoother is initialized with the forward filter's last epoch states and covariance at the end of forward processing, as stated by:

$$\delta\hat{x}_{k+1}^{s} = \delta\hat{x}_{k+1}^{f+}$$

$$\mathrm{P}_{k+1}^{s} = \mathrm{P}_{k+1}^{f+}$$

Once initialized, the smoothed estimate of the states is computed using the equations below.

$$\delta\hat{x}_{k}^{s} = \delta\hat{x}_{k}^{f+} + G_k(\delta\hat{x}_{k+1}^{s} - \delta\hat{x}_{k+1}^{f-}) \tag{5.8}$$

$$\mathrm{P}_k^s = \mathrm{P}_k^{f+} + G_k(\mathrm{P}_{k+1}^s - \mathrm{P}_{k+1}^{f-})G_k{}^T \tag{5.9}$$

$$G_k = \mathrm{P}_k^{f+}\Phi_{k+1,k}^T(\mathrm{P}_{k+1}^{f-})^{-1} \tag{5.10}$$

Where '+' and '-' denote the estimated and the predicted variables, and are the error states and their variances stored in the forward filter, $\Phi_{\mathrm{k+1,k}}^{\mathrm{T}}$ is the transition matrix from time $\mathrm{t_k}$ to $\mathrm{t_{k+1}}$.

Equations (5.8) through (5.10) combined are referred to as the backward processing equations. Since in our 24 error-state filter, the state vector is actually a state error vector, the above set of equations results in a series of smoothed corrections for each epoch that should be applied to the navigation solution computed using the forward filter [45]. Ultimately, the smoothed solution is computed using Equation (5.11):

$$\hat{x}_k^s = \hat{x}_k^{f+} + (\delta\hat{x}_k^s - \delta\hat{x}_k^{f+}) \tag{5.11}$$

With each backward sweep, the forward filter estimate is updated to yield an improved smoothed estimate, which is based on all the measurement data [47].

5.3Step-wise Segmentation

The implementation of the RTS smoother has some limitations. First, it is commonly applied for post processing. Second, a lot of memory is required to store the predicted and updated information [48]. In order to achieve near-real-time smoothing and reduce the storage burden, one intuitive way is to take profit of the detected zero velocity intervals and divide the step with them.

In order to get near-real-time smoothing, proper segmentation rules should be applied. The principle for our data segmentation is similar to the one proposed by [41]. They use velocity error covariance as a threshold. The choice of the threshold value is based on typical values from the training data of random walking. So it is not suitable for scenarios that involve running. Moreover, it is not applicable for navigation on a moving platform, such as an elevator or escalator, where the velocity of a foot in stance phase may not converge to zero but a constant velocity. In addition, the fixed time constant as 30 samples that they applied may work properly in normal walking but fail occasionally in running, since the duration of the stance phase in running may be shorter than 30 samples and it will cause miss detection. Furthermore, if the person alternates between walking and running, the segmentation rule based on velocity error covariance will easily get confused and become invalid.

We proposed a new segmentation rule by combining both the footstep detector and swing phase detector. The energy of rotation threshold T (refer to equation 3.5) is used to avoid false segmentation in swing phases, which makes the rule more flexible to meet the complex environment.

The segmentation point should be in the stance phase. Here we have tried three methods to get the segmentation point, the first one is to use the middle point of the stance phase, the second one is the sample point with smallest T during the stance phase and the third one is the sample point with T under a certain small threshold. According to our tests, these three methods show little difference in the final smoothing results, which means that we do not need to find where the exact point converges to zero, as

long as the segmentation point is within the stance phase. The third method is used as the segmentation rule in the following closed-loop smoothing.

The footstep detector is used not only to detect the push off which indicates the start of a swing phase, but also the toe off section when the foot is leaving the ground. The segmentation point should be during the stance phase. Here we have tried three methods to get the segmentation point, the first one is to use the middle point of the stance phase, the second one is the sample point with smallest T during the stance phase and the third one is the sample point meet T under a certain small threshold. According to our tests, these three methods have no big differences in affecting the final smooth result, which means that we do not need to find where the exact point converges to zero, as long as the segmentation point is within the stance phase and there is a certain level of trust that the zero velocity detection is correct. So in the following chapter, we use the third method as our segmentation rule to analysis.

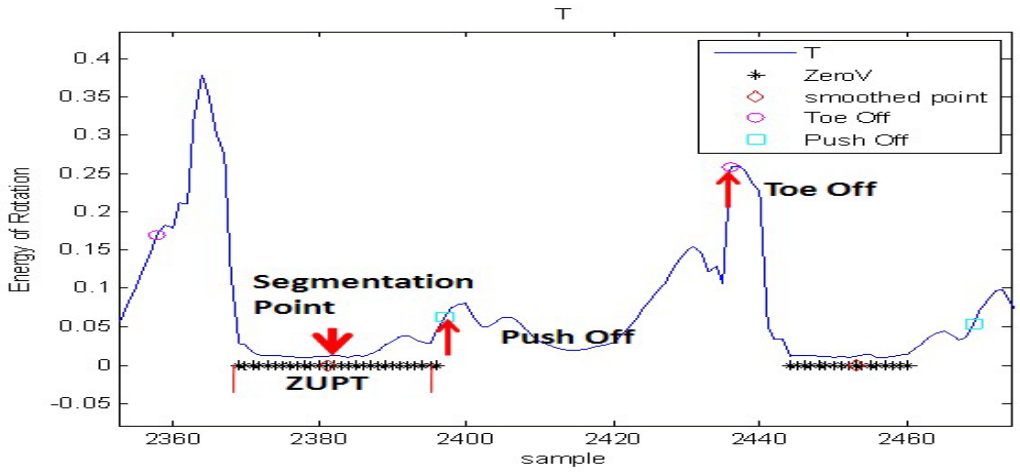


Figure 5.3 Segmentation Rule

As for the special case on the lift, when the person stands in the lift and keeps stationary relative to the lift going up or down, the observation can be constant position

in the horizontal plane, while when the lift keeps at constant velocity, we use CUPT to update the error states. But it is important to notice that in this case the velocity will not converge to zero but to constant velocity when CUPT applied.

5.4Closed Loop Smoothing

In our ZUPT aided INS implementation, the predicted EKF error states are always reset to zero. During measurement updates, the estimated error states are corrected and fed back to compensate the estimated navigation states. However, in a closed loop smoothing aided ZUPT, the EKF runs as normal but the predicted error states need to be propagated and no longer reset to zero. The estimated states and covariance are updated only when ZUPT are applied, the smoothing is applied over the error states. After the step segmentation is performed, the estimated states are combined with the smoothed error states to get the final navigation results. Figure 5.4 summarizes the implementation procedures for the closed-loop smoothing.

Loop: For n=1 to the end of the data

Time Update	INS mechanization Error states and error covariance propagate	$\hat{x}_k^-$ $\delta\hat{x}_k^- = F_{k,k-1}\delta\hat{x}_{k-1}^+$ $P_k^- = F_{k,k-1}P_{k-1}^+F_{k,k-1}^T + Q_{k-1}$
Footstep detector	Detect start of the step	$T(k_1) < T1$
Measurement Update If ZUPT	Error states and error covariance update Compensate navigation states	$K_k = P_k^-H_k^T(H_kP_k^-H_k^T + R_k)^{-1}$ $v_k = z_k - H_k\delta\hat{x}_k^-$ $\delta\hat{x}_k^+ = \delta\hat{x}_k^- + K_kv_k$ $P_k^+ = (I - K_kH_k)P_k^-$ $\hat{x}_k^+ = \hat{x}_k^- + \delta\hat{x}_k^+$
Footstep detector	Detect swing phase	$T(k_2) < T2$
Segmentation	Segmentation rule Decide: go back to time update/ leave loop	$k_1 < k < k\underline{T}(k) < T3$
After loop	Smoothing of the error states Compensate navigation states Go back to loop and go on processing	$\delta\hat{x}_{k\|N}$ (Equation 9) $\hat{x}_{k\|N} = \hat{x}_k^+ + \delta\hat{x}_{k\|N}$

Figure 5.4 Step wise close loop smoothed ZUPT aided INS

5.5Experimental Results

All experimental tests have been conducted in UTS, including walking, running and taking escalator and elevator. The figures below show the achieved smoothing effect compared with the estimated trajectory.

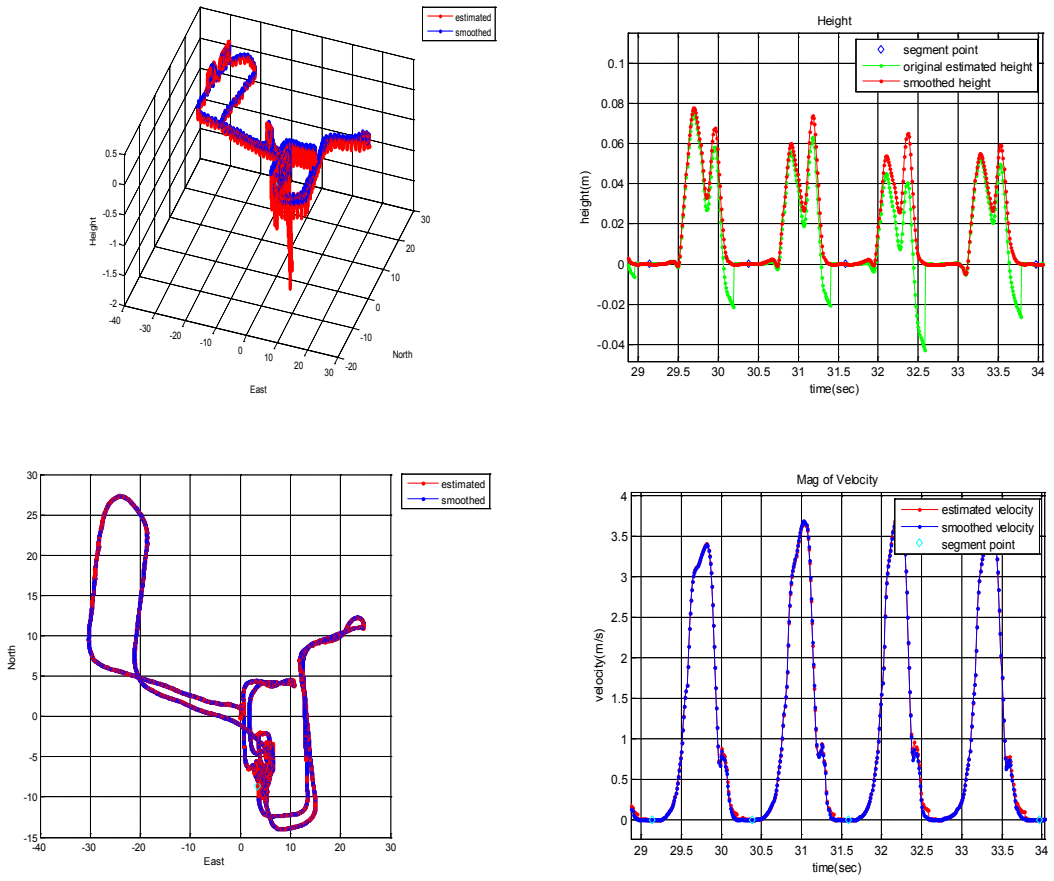


Figure 5.5 Effect of smoothing over a walking trajectory

Figure 5.5 and 5.6 show the effect of smoothing over a walking and running trajectory respectively. From the height and velocity analysis, we can see that when the first ZUPT is applied, the correction applied over the estimated trajectory results in a big jump appearing in every step. After smoothing, these big jumps can be eliminated thus the trajectory is smoother than the customary one.

Compared with walking, the magnitude of running velocity may vary and not

converge to exactly zero as walking during the stance phase.

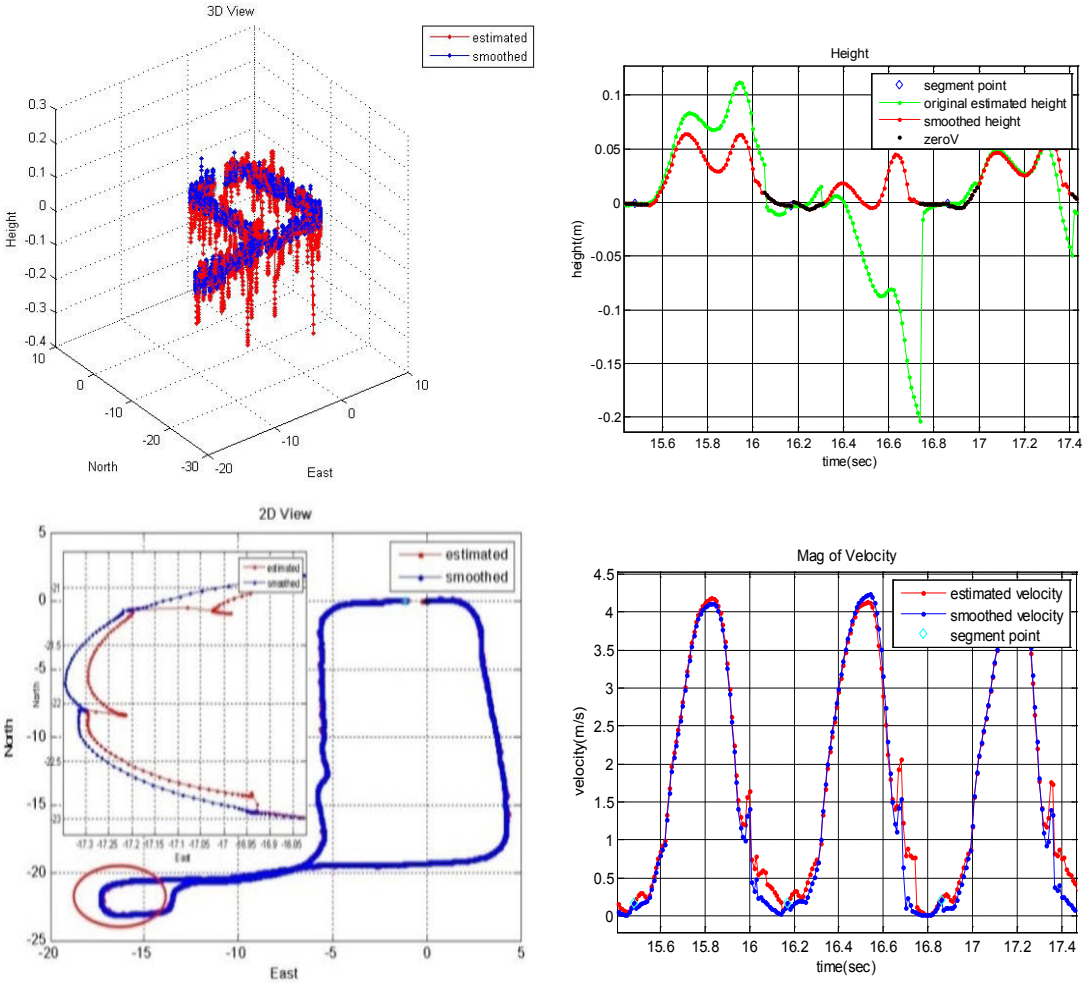
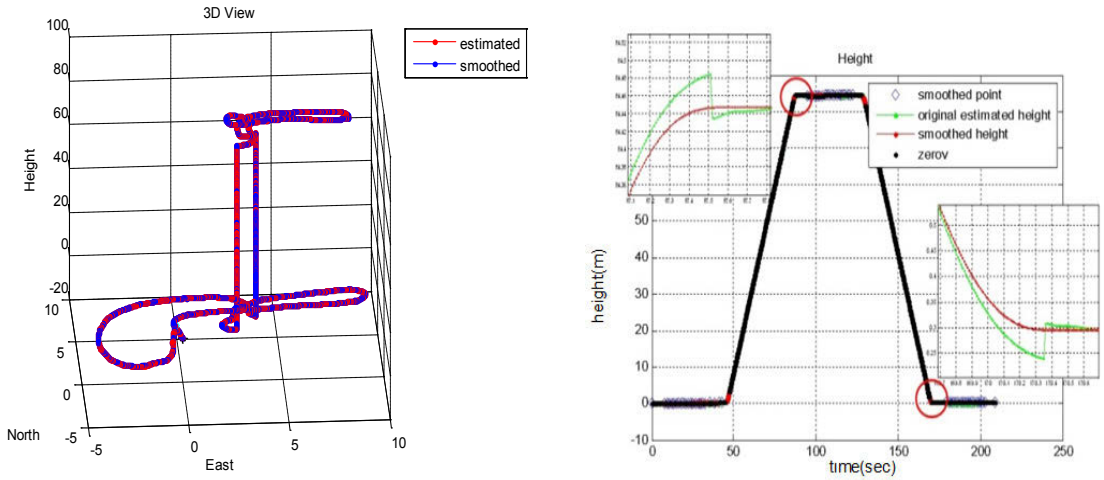


Figure 5.6 Effect of smoothing over a running trajectory



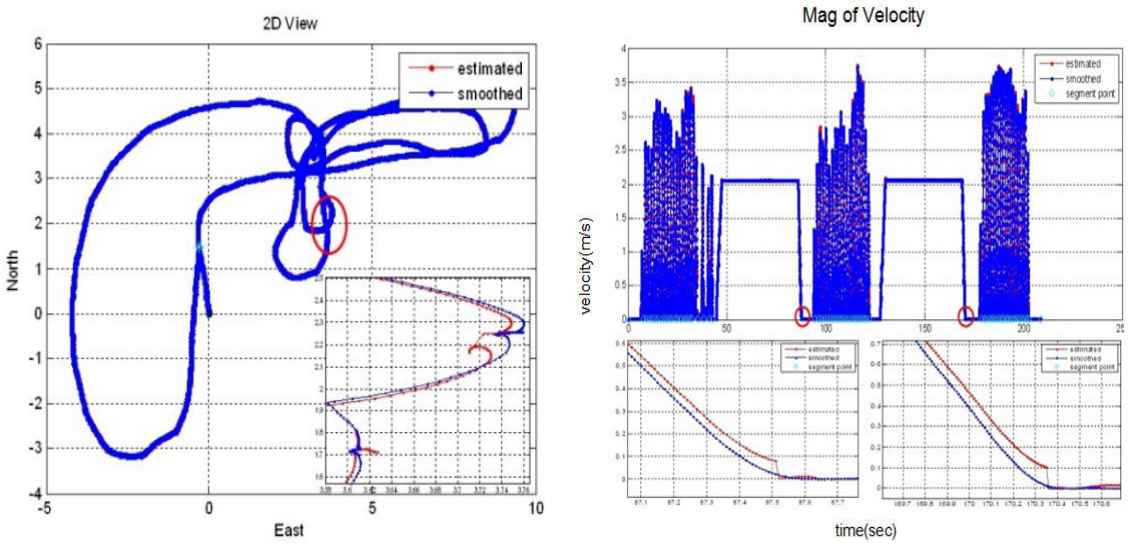


Figure 5.7 Effect of smoothing over a taking elevator trajectory

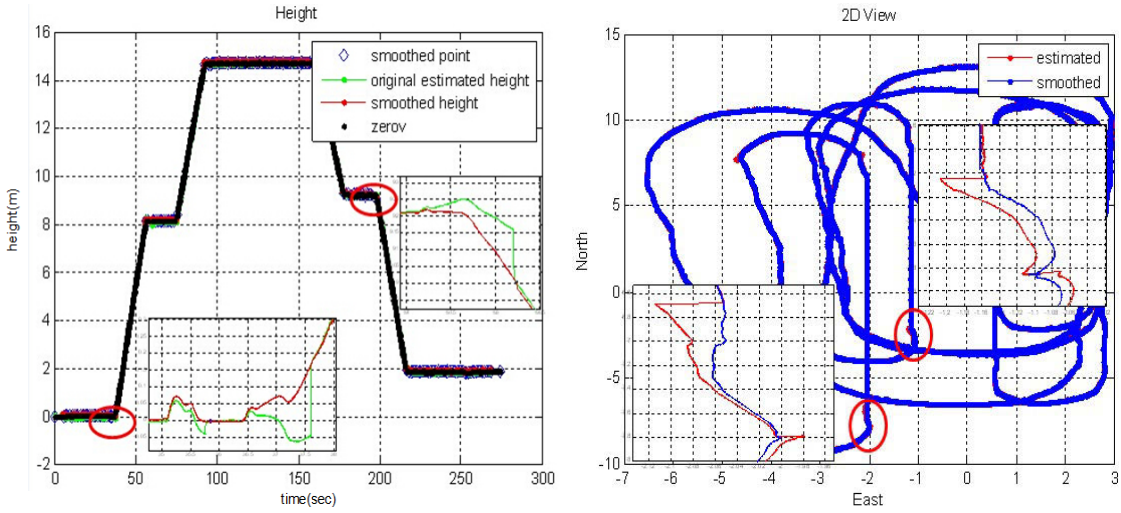


Figure 5.8 Effect of smoothing over an escalator trajectory

For the cases in the elevator and escalator, CUPT is applied to let velocity not converge to zero as with ZUPT, but a constant velocity when the pedestrian stands on the moving platform. It is also obvious to see that the smoothing algorithm can significantly improve the smoothness of the trajectory especially when the elevator finishes accelerating or decelerating and the pedestrian gets on or off the escalator.

5.6Summary

In this chapter, we have proposed a new step wise closed loop smoothing to help to eliminate the sharp correction at the sample points that ZUPT applied and thus improve the smoothness of the trajectory. Since ZUPT applies in each step, a segmentation rule is proposed for a step-wise data processing. This rule is based on the energy of rotation. By considering the start and the end of the step, a smoothed closed loop step wise ZUPT aided INS pedestrian navigation system is proposed. Experimental results have shown the impact of the smoothing that this algorithm can efficiently eliminate the discontinuities at the end of each step.

Chapter VI

CONCLUSION AND FUTURE WORK

6.1Conclusion

The main objective of the research presented in the thesis was to develop an autonomous Pedestrian Navigation System feasible for a complex building environment and which uses only a low-cost IMU, aided with only ZUPTs.

Chapter 3 mainly introduces the INS mechanization and ZUPT aided INS. We propose a novel dynamic and more robust algorithm to detect the stance phases during walking which could work in both 2D and 3D environments with good performance. In order to handle more complex gait behaviour like running, we develop a footstep detector which uses the concept of human gait cycles and a swing phase detector that can efficiently prevent false detection in the swing phase. There is still has some space for further improvement. For example, integrating the system with a building map that could constrain where the turns can occur. It is also possible to add an extra sensor; we can add a magnetometer as a measurement to prevent heading drift, or force sensors beneath the boot to improve the accuracy of stance phase detection. This would be particularly useful in the running case since it is still a challenge to detect the stance phase in every step without miss detection.

Chapter 4 gives a detailed description of the innovation algorithm called CUPT to aid the pedestrian navigation system. This technique is based on the known ZUPT technique which mainly deals with the case of a subject is on a moving escalator or

elevator. This contribution has broadened the practical use of a pedestrian navigation system with only IMU used in a modern building environment. The limitations are the pre-determined escalator angle as 30 degrees and simplified elevator motion model. Thus further improvement can be focused on the mathematical model for both elevator and escalator.

Chapter 5 is an alternative method to the existed smoothing method proposed by [41]. Our closed-loop smoothing algorithm is useful to eliminate the sharp corrections due to ZUPT correction. From further investigation, we found that these sharp discontinuities may affect the final positioning accuracy as an accumulated factor. This inference needs more compact mathematical analysis.

6.2 Future Work

6.2.1 Vision Aided Navigation

Visual-aiding has been used already for decades in navigation of robots and unmanned vehicles. In navigation application, the focus of the research has been mainly in systems using a priori formed databases. The databases contain images of recognizable features in the surroundings attached with position information. When a match between images in the database and the ones taken by a pedestrian is found, the absolute position may be obtained. The database based procedure is however laborious due to the a priori preparations and is restricted to the predefined region. Integrating visual-aiding information with the measurements from other sensors has become a research topic only in the recent years.

Cameras can be used to provide measurements such as translation and rotation between frames by tracking features in successive images. In the future we may look at the use of aiding measurements from a camera attached to an IMU where the user is walking with the device held out in front of them with the camera pointing approximately towards the ground. This is a typical use of smart phones where the user is reading navigation information from the display. The camera therefore has a view of the ground beneath, and immediately in front of the user. Sequential images can then be used to compute the 3-dimensional body frame translation direction of the camera as well as 3-D rotation. The images can be captured at a relatively low rate (a few per second) provided sufficient common features exist between successive frames. From a single camera alone, the body-frame translation can only be computed up-to an unknown scale factor; however if the height of the camera above the ground is known and can be assumed as unchanged, the absolute velocity of the camera can be computed. This measurement is used to correct the drift of the IMU-derived position.

In addition, vision/GPS/INS integration will be an important part. For vision/GPS/IMU integration, the IMU errors and the internal parameters of the vision sensor can be corrected when a GPS signal is available. We still need to have a lot of relevant work to focus on these academic problems, for example, how to distinguish the GPS signal is reliable or not from the information we have, how to use GPS data to correct the vision sensor's internal parameters and how vision information can be used to correct IMU's errors when GPS is unavailable?

On-line camera intrinsic parameter self-calibration is also an interesting topic. In

the review of GPS/INS theory, it was shown that inertial sensor errors could be recovered in a loosely-coupled GPS/INS navigation filter. A reasonable extension to the GPS/visual odometry (VO) work would be used for the on-line estimation of the camera intrinsic parameters, especially for cases where the focal length and distortion parameters vary with time (e.g. a zoom camera).

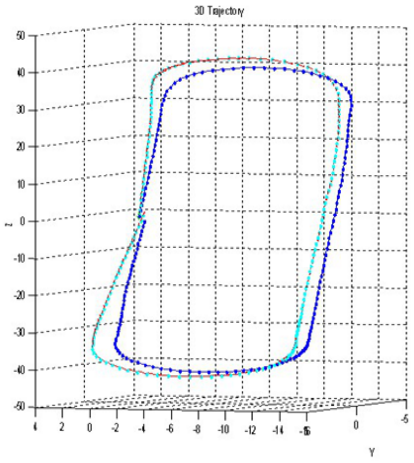


Figure 6.1 VO trajectory

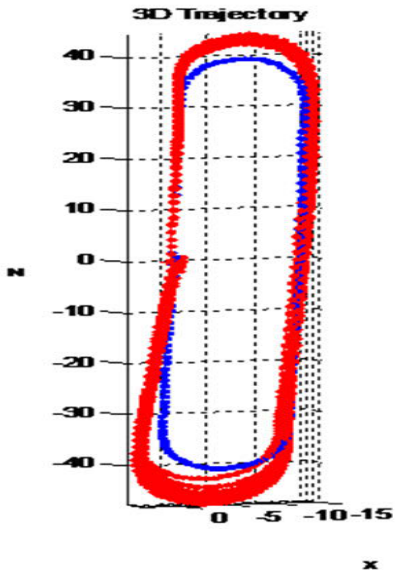


Figure 6.2 3D VO trajectory repeated for 10 times

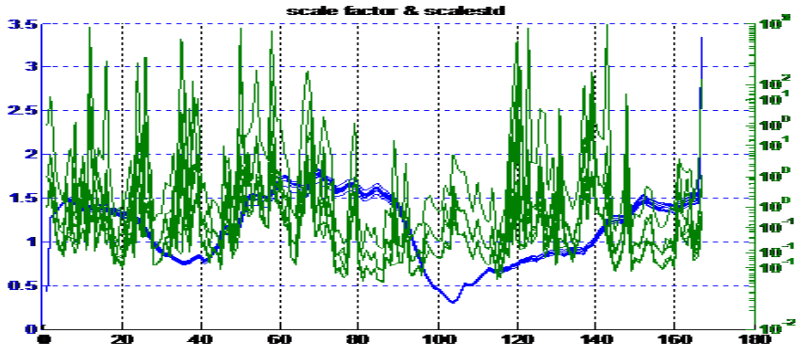


Figure 6.3 Scale factor and the standard deviation

Further on, from our current research on VO, we know that the drift in VO is caused by the random choosing of the matched points to create the fundamental matrix as well as noise. The figure 6.1 is the result of our previous VO which drifted from the ground truth. From the same data repeated for 10 times as shown in figure 6.2 we can see the repeatability can be assured. Thus we can reduce the uncertainty to a certain limited range, where the green lines represent the scale factor and the blue line represents their standard deviation in figure 6.3. We have a bold guess that this kind of regular drift is caused by the uncertainty of the given one-off calibrated intrinsic parameters when the repeatability is at a limited range. It will be a challenge to derive an error propagation model of the intrinsic parameters' uncertainty, from which it could be how the uncertainties affect the drift which would be an innovation for on-line camera calibration.

6.2.2 Integration Algorithm Optimization

Optimization problems occur frequently in fusion research. The simplest example is the association problem when observations from multiple sensors are to be assigned to various tracks in a multi-target tracker. System performance depends on

solving this problem fast and accurately. Sensor fusion optimization is a big challenge and I may develop novel methods to achieve this in my further research.

In the experimental implementation for INS algorithms, the performance of the filter was influenced by the choice of the process noise attributes. Normally we choose noise by experience or engineering judgement. If we can identify the process noise from the collected raw data, then we can create a self-tuning filter. In addition, quantization and colored noises error modelling for inertial sensors and adaptive EKF is also an important topic.

Other non-linear filters such as unscented Kalman filter (UKF) and particle filter (PF) perform over the traditional EKF in navigation applications. Learning these advanced filter methods and applying them to my multi-sensor system can help to get better navigation solution.

REFERENCE

[1] Y. S. Suh and S. Park, "Pedestrian inertial navigation with gait phase detection assisted zero velocity updating," presented at the Autonomous Robots and Agents, 2009. ICARA 2009. 4th International Conference on Autonomous Robots and Agents, Wellington, New Zealand, 2009.

[2] Fu, Qing, and Guenther Retscher. "Another look indoors GPS+ RFID." GPS World 20(3) (2009).

[3] Biswas, J., and Veloso, M.. “Wifi localization and navigation for autonomous indoor mobile robots”. 2010 IEEE International Conference on Robotics and Automation (ICRA),(pp. 4379-4384).

[4] S. Godha and G. Lachapelle, "Foot mounted inertial system for pedestrian navigation," Meas. Sci. Technol, vol. 19 2008.

[5] Beauregard, Stéphane, and Martin Klepal. "Indoor PDR performance enhancement using minimal map information and particle filters." 2008 IEEE/ION Position, Location and Navigation Symposium.

[6] Aoki, H., Schiele, B. and Pentland, A. (1999). “Real-time personal positioning system for wearable computer”. The 3rd IEEE International Symposium on Wearable Computers (ISWC 1999), Washington, DC, USA.

[7] Du S, Huang B, Gao Y. “An integrated MEMS IMU/camera system for pedestrian indoor navigation using smartphones”, In 2012 Proceedings China Satellite Navigation Conference (CSNC), pp 669-678. Springer Berlin Heidelberg, 2012.

[8] Hide, C., Botterill, T. and Andreotti, M. (2010). Low cost vision-aided IMU for pedestrian navigation. In Ubiquitous Positioning Indoor Navigation and Location Based Service (UPINLBS), pp. 1-7, Kirkkonummi, Finland.

[9] Savage, P.G. 2000a, Strapdown Analytics: Part I. Strapdown Associates; Rogers, R.M. 2000, Applied mathematics in integrated navigation systems

[10] D. Titterton and J. Weston. Strapdown Inertial Navigation Technology. The American Institute of Aeronautics and Astronautics, 2nd edition, 2004.

[11] Roetenberg, D. (2006). Inertial and magnetic sensing of human motion.Ph.D. thesis, Enschede.

[12] Mezentsev, O., 2005. Sensor aiding of HSGPS pedestrian navigation. Ph.D. University of Calgary

[13] O. Woodman, “An introduction to inertial navigation”, Technical Report 696, University of Cambridge, Computer Laboratory, 2007

[14] Weinberg, M. S. & Kourepenis, A., 2006. “Error sources in in-plane silicon tuning-fork MEMS gyroscopes”. Journal of Microelectromechanical Systems, 15, 479-491

[15] Oliver J. Woodman, “Pedestrian localisation for indoor environments”, PhD thesis, 2010

[16] Farrell, J. & Barth, M., 2008. Aide navigation: GPS with high rate sensors; Groves, P. D., 2008. Principles of GNSS, inertial, and multi-sensor Integrated Navigation Systems

[17] Shin, E. H., 2005. Estimation techniques for low cost inertial navigation， Ph.D. University of Calgary.

[18] Grejner-Brzezinska, D. A., Toth, C. K. and Yi, Y. (2001). Bridging GPS gaps in urban canyons: can ZUPT really help? Proceedings of the 58th Annual Meeting of the Institute of Navigation and CIGTF 21st Guidance Test Symposium, pp. 231-240.

[19] Grewal, M. S. & Andrews, A. P., 2008. Kalman filtering: theory and practice using MATLAB. 3rd ed. New York: Wiley

[20] Grover, Robert, and P. Y. C. Hwang. "Introduction to random signals and applied Kalman filtering." Willey, New York (1992).

[21] Callmer J. Autonomous Localization in Unknown Environments[J]. 2013.

[22] Woodman, O., and Harle, R. Pedestrian localisation for indoor environments. In Proceedings of the 10th international conference on Ubiquitous computing (pp. 114-123). ACM.

[23] Foxlin, E. (2005). Pedestrian tracking with shoe-mounted inertial sensors. IEEE Computer Graphics and Applications, 25(6), 38-46.

[24] Castaneda, N. and Lamy-Perbal, S. (2010). An improved shoe-mounted inertial navigation system. 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Campus Science City, ETH Zurich.

[25] Cho, S. Y. and Park, C. G. (2006). MEMS based pedestrian navigation system. Journal of Navigation, 59(1), 135-153.

[26] Huang, C., Liao, Z. and Zhao, L. (2010). Synergism of INS and PDR in self-contained pedestrian tracking with a miniature sensor module. IEEE Sensors Journal, 10(8), 1349-1359.

[27] Feliz, R., Zalama, E. and García-Bermejo, J. G. (2009). Pedestrian tracking using inertial sensors. Journal of Physical Agents, 3(1), 35-43.

[28] Bebek, O., Suster, M. A., Rajgopal, S., Fu, M. J., Huang, X., Çavuşoğlu, M. C. and Mastrangelo, C. H. (2010). Personal navigation via high-resolution gait-corrected inertial measurement units. The IEEE Transactions on Instrumentation and Measurement, 59(11), 3018 – 3027

[29] Jiménez, A. R., Seco, F., Prieto, J. C. and Guevara, J. (2010). Indoor pedestrian navigation using an INS/EKF framework for yaw drift reduction and a foot-mounted IMU. 7th Workshop on Positioning Navigation and Communication (WPNC), pp.135 – 143, Dresden, German.

[30] Cho, S. Y., Lee, K. W., Park, C. G. and Lee, J. G. (2003). A personal navigation system using low-cost MEMS/GPS/Fluxgate. The Proceedings of the 59th Annual Meeting of the Institute of Navigation and CIGTF 22nd Guidance Test Symposium, Albuquerque, NM.

[31] Wang, Hui, et al. "WLAN-based pedestrian tracking using particle filters and low-cost MEMS sensors." IEEE 4th Workshop on Positioning, Navigation and Communication, 2007.

[32] Suh, Y. S. and Park, S. (2009). Pedestrian inertial navigation with gait phase detection assisted zero velocity updating. 4th International Conference on Autonomous Robots and Agents, Wellington, New Zealand.

[33] Park, S. K. and Suh, Y. S. (2010). A zero velocity detection algorithm using inertial sensors for pedestrian navigation systems. Sensors, 10(10), 9163-9178.

[34] Yun, X., Bachmann, E. R., Moore, H. and Calusdian, J. (2007). Self-contained position tracking of human movement using small inertial/magnetic sensor modules. 2007 IEEE International Conference on Robotics and Automation (ICRA), Roma, Italy.

[35] Skog, I., Nilsson, J. O. and Handel, P. I. (2010). Evaluation of zero-velocity detectors for foot-mounted inertial navigation systems. 2010 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Campus Science City, ETH Zurich.

[36] Ojeda, L. and Borenstein, J. (2007). Non-GPS navigation for security personnel and first responders. Journal of Navigation, 60(3), 391-407.

[37] Abdulrahim, K., Hide, C., Moore, T. and Hill, C. (2012). Using constraints for shoe mounted indoor pedestrian navigation. Journal of Navigation 65(1), 15-28.

[38] S. Rajagopal, "Personal dead reckoning system with shoe mounted inertial sensors," Master of Science, Electrical and Electronic Engineering KTH, Stockholm, Sweden, 2008.

[39] Sidney. Kwakkel, "Human Lower Limb Kinematics Using GPS/INS",Master of Philosophy, Department of Geomatics Engineering, The University of Calgary, December, 2008.

[40] Angermann, M., Robertson, P., Kemptner, T. and Khider, M. (2010). A high precision reference data set for pedestrian navigation using foot-mounted inertial sensors. 2010 IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN), Zurich, Switzerland, pp. 1-6.

[41] Colomar S. D., Nilsson J. O., Handel P. (2012). Smoothing for ZUPT-aided INSs. 2012 IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN), Sydney, Australia.

[42] Meditch, S. (1969) Stochastic Optimal Linear Estimation and Control, McGraw-Hill, New York.

[43] Godha, S. "Performance evaluation of low cost MEMS-based IMU integrated with GPS for land vehicle navigation application." UCGE report20239 (2006).

[44] Scherzinger, B. (2004) Estimation with Application to Navigation, ENGO 699.11-Course Notes, Department of Geomatics Engineering, University of Calgary, Canada

[45] Hide, C. D. and T. Moore (2004) “Low Cost Sensors, High Quality Integration,” in Proceedings of NAV/AIS04, Location and Timing Applications, London.

[46] Rauch, H. E., Striebel, C. T., & Tung, F. (1965). Maximum likelihood estimates of linear dynamic systems. AIAA journal, 3(8), 1445-1450.

[47] Brown, R. G., & Hwang, P. Y. Introduction to random signals and applied kalman filtering. 1997. NY John Wiley and Sons.

[48] Chiang, K. W., Duong, T. T., Liao, J. K., Lai, Y. C., Chang, C. C., Cai, J. M. and Huang, S. C. (2012). On-line smoothing for an integrated navigation system with low-cost MEMS inertial sensors. Sensors 12(12), 17372-17389.