UNIVERSITY OF CALIFORNIA,
IRVINE

Pedestrian Inertial Navigation — Development of Sensors and Algorithms

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Mechanical and Aerospace Engineering

by

Yusheng Wang

Dissertation Committee:
Professor Andrei M. Shkel, Chair
Professor Solmaz Kia
Professor Tryphon Georgiou

2020

DEDICATION

To the Lord Jesus Christ

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First, I would like to thank Professor Andrei M. Shkel for being my advisor and dissertation committee chair. He provided me the valuable opportunity to conduct research in the area of microelectromechanical inertial sensors and navigation. This journey would not have been possible without his continuous support, guidance, and encouragement throughout the years. I also would like to thank the members of my dissertation committee Professor Solmaz Kia and Professor Tryphon Georgiou for their time to review this work and provide insightful comments.

Next, I would like to thank the colleagues and friends at UCI MicroSystems Laboratory for their support, collaboration, and valuable discussions. More specifically, I would like to thank Dr. Mohammad H. Asadian, Dr. Sina Askari, Dr. Radwan Mohammednoor, Yu-Wei Lin, Danmeng Wang, Daryosh Vatanparvar, and Chi-Shih Jao for their valuable discussion and help in various aspects of the research, as well as Dr. Doruk Senkal, Dr. Joan Giner de Haro, and Dr. Alexandra Efimovskaya for their guidance and help during the first few years of my research.

I would like to thank Dr. Andrei Chernyshoff for his invaluable technical support and insight in inertial navigation algorithm development. I would like to thank Professor Thomas W. Kenny and his research group at Stanford University, especially Dr. Janna Rodriguez and Gabrielle D. Vukasin for their help in the collaboration of epitaxially-encapsulated devices. In addition, I would like to acknowledge staff members of the Integrated Nanosystems Research Facility (INRF), Irvine Materials Research Institute (IMRI), and Institute for Design & Manufacturing Innovation (IDMI) at the University of California, Irvine for technical support over the years.

I also would like to acknowledge the generous funding provided by Defense Advanced Research Projects Agency (DARPA) and National Institute of Standards and Technology (NIST). Specifically, this work has been supported by DARPA under Grants W31P4Q11-1-0006 and N66001-16-1-4021, and NIST under Grant 70NANB17H192.

Finally, I would like to thank my family and friends, who have been supporting me all the time, both mentally and spiritually.

CURRICULUM VITAE

Yusheng Wang

EDUCATION

Doctor of Philosophy in Mechanical and Aerospace Engineering — **June 2020**
University of California, Irvine — *Irvine, CA, USA*

Master of Science in Mechanical and Aerospace Engineering — **December 2016**
University of California, Irvine — *Irvine, CA, USA*

Bachelor of Engineering in Engineering Mechanics — **July 2014**
Tsinghua University — *Beijing, China*

RESEARCH EXPERIENCE

Graduate Research Assistant — **2014–2020**
University of California, Irvine — *Irvine, CA, USA*

Undergraduate Research Intern — **2014**
University of Washington — *Seattle, WA, USA*

Undergraduate Research Assistant — **2012–2014**
Tsinghua University — *Beijing, China*

JOURNAL PUBLICATIONS

Yusheng Wang, Andrei Chernyshoff and Andrei M. Shkel, "Study on Estimation Errors in ZUPT-Aided Pedestrian Inertial Navigation Due to IMU Noises," *IEEE Transactions on Aerospace and Electronic Systems*, doi: 10.1109/TAES.2019.2946506

Mohammad H. Asadian, **Yusheng Wang**, and Andrei Shkel, "Development of 3D Fused Quartz Hemi-toroidal Shells for High-Q Resonators and Gyroscopes," *IEEE Journal of Microelectromechanical Systems*, vol. 28, no. 6, pp. 954-964, 2019.

Yusheng Wang and Andrei M. Shkel, "Adaptive Threshold for Zero-Velocity Detector in ZUPT-Aided Pedestrian Inertial Navigation," *IEEE Sensors Letters*, vol. 3, no. 11, pp. 1-4, 2019.

Yusheng Wang, Daryosh Vatanparvar, Andrei Chernyshoff and Andrei M. Shkel, "Analytical Closed-Form Estimation of Position Error on ZUPT-Augmented Pedestrian Inertial Navigation," *IEEE Sensors Letters*, vol. 2, no. 4, pp. 1-4, 2018.

Yusheng Wang, Mohammad H. Asadian, and Andrei M. Shkel, "Compensation of Frequency Split by Directional Lapping in Fused Quartz Micro Wineglass Resonators," *Journal of Micromechanics and Microengineering*, vol. 28, no. 9, pp. 095001, 2018.

Yusheng Wang, Mohammad H. Asadian, and Andrei M. Shkel, "Modeling the Effect of Imperfections in Glassblown Micro-Wineglass Fused Quartz Resonators," *ASME Journal of Vibration and Acoustics*, vol. 139, no. 4, pp. 040909, 2017.

CONFERENCE PUBLICATIONS

Yusheng Wang and Andrei M. Shkel, "A Review on ZUPT-Aided Pedestrian Inertial Navigation," *27th Saint Petersburg International Conference on Integrated Navigation Systems*, May 25-27, 2020, Saint Petersburg, Russia.

Yusheng Wang, Yu-Wei Lin, Sina Askari, Chi-Shih Jao, and Andrei M. Shkel, "Compensation of Systematic Errors in ZUPT-Aided Pedestrian Inertial Navigation," *IEEE/ION Position Location and Navigation Symposium (PLANS)*, Apr. 20-23, 2020, Portland, OR, USA.

Mohammad H. Asadian, Danmeng Wang, **Yusheng Wang**, and Andrei M. Shkel, "3D Dual-Shell Micro-Resonators for Harsh Environments," *IEEE/ION Position Location and Navigation Symposium (PLANS)*, Apr. 20-23, 2020, Portland, OR, USA.

Chi-Shih Jao, **Yusheng Wang**, Sina Askari, and Andrei M. Shkel, "A Close-formAnalytical Estimation of Vertical Displacement Error in Altimeter/INS Sensor Fusion Systemfor Pedestrian Navigation," *IEEE/ION Position Location and Navigation Symposium (PLANS)*, Apr. 20-23, 2020, Portland, OR, USA.

Chi-Shih Jao, **Yusheng Wang**, and Andrei M. Shkel, "Pedestrian Inertial Navigation System Augmented by Vision-Based Foot-to-foot Relative Position Measurements", *IEEE/ION Position Location and Navigation Symposium (PLANS)*, Apr. 20-23, 2020, Portland, OR, USA.

Sina Askari, Chi-Shih Jao, **Yusheng Wang**, and Andrei M. Shkel, "Learning-Based Calibration Decision System for Bio-Inertial Motion Application," *IEEE Sensors Conference*, Oct 27-30, 2019, Montreal, Canada, pp. 1-4.

Mohammad H. Asadian, **Yusheng Wang**, and Andrei M. Shkel, "Development of Microscale 3D Fused Quartz Hemi-Toroidal Shells for High-Q Resonators and Gyroscopes," *CIVIL-COMP & CIVIL-COMP-OPTI*, Elsevier Publisher, Sep. 16-19, 2019, Riva del Garda, Italy.

Yusheng Wang, Sina Askari, Chi-Shih Jao, and Andrei M. Shkel, "Directional Ranging for Enhanced Performance of Aided Pedestrian Inertial Navigation," *IEEE Inertial Sensors*, Apr. 1-5, 2019, Naples, FL, USA, pp. 39-40.

Sina Askari, Chi-Shih Jao, **Yusheng Wang**, and Andrei M. Shkel, "A Laboratory Testbed for Self-Contained Navigation," *IEEE Inertial Sensors*, Apr. 1-5, 2019, Naples, FL, USA, pp. 41-42.

Yusheng Wang, Sina Askari, and Andrei M. Shkel, "Study on Mounting Position of IMU for Better Accuracy of ZUPT-Aided Pedestrian Inertial Navigation," *IEEE Inertial Sensors*, Apr. 1-5, 2019, Naples, FL, USA, pp. 116-119.

Mohammad H. Asadian, **Yusheng Wang**, Radwan M. Noor, and Andrei M. Shkel, "Design Space Exploration of Hemi-Toroidal Fused Quartz Shell Resonators," *IEEE Inertial Sensors*, Apr. 1-5, 2019, Naples, FL, USA, pp. 151-154.

Mohammad H. Asadian, Sina Askari, **Yusheng Wang**, and Andrei M. Shkel, "Characterization of Energy Dissipation Mechanisms in Dual Foucault Pendulum Gyroscopes," *IEEE Inertial Sensors*, Apr. 1-5, 2019, Naples, FL, USA, pp. 200-201.

Mohammad H. Asadian, **Yusheng Wang**, and Andrei M. Shkel, "Design and Fabrication of 3D Fused Quartz Shell Resonators for Broad Range of Frequencies and Increased Decay Time," *IEEE Sensors Conference*, Oct. 29-31, 2018, New Delhi, India, pp. 1736-1739.

Yusheng Wang, Andrei Chernyshoff, and Andrei M. Shkel, "Error Analysis of ZUPT-Aided Pedestrian Inertial Navigation," *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sep. 24-27, 2018, Nantes, France, pp. 206-212.

Yusheng Wang, Yu-Wei Lin, Janna Rodriguez, Gabrielle D. Vukasin, Dongsuk D. Shin, Hyun-Keun Kwon, David B. Heinz, Yunhan Chen, Dustin D. Gerrard, Thomas W. Kenny, and Andrei M. Shkel, "On Decoupled Quantification of Energy Dissipation Mechanisms in Toroidal Ring Gyroscopes," *Solid-State Sensors, Actuators, and Microsystems Workshop (Hilton Head)*, June 3-7, 2018, Hilton Head Island, SC, USA, pp. 318-321.

Mohammad H. Asadian, **Yusheng Wang**, Sina Askari and Andrei M. Shkel, "Controlled Capacitive Gaps for Electrostatic Actuation and Tuning of 3D Fused Quartz Micro Wineglass Resonator Gyroscope," *IEEE Inertial Sensors*, Mar. 27-30, 2017, Kauai, HI, USA, pp. 19-22.

Yusheng Wang, Mohammad H. Asadian, and Andrei M. Shkel, "Frequency Split Reduction by Directional Lapping of Fused Quartz Micro Wineglass Resonators," *IEEE Inertial Sensors*, Mar. 27-30, 2017, Kauai, HI, USA, pp. 78-81.

Yusheng Wang, Mohammad H. Asadian, Andrei M. Shkel, "Predictive Analytical Model of Fundamental Frequency and Imperfections in Glassblown Fused Quartz Hemi-Toroidal 3D Micro Shells," *IEEE Sensors Conference*, Oct. 30 - Nov. 2, 2016, Orlando, FL, USA, pp. 1-3.

Yusheng Wang and Andrei M. Shkel, "Study on Surface Roughness Improvement of Fused Quartz After Thermal and Chemical Post-Processing," *IEEE Inertial Sensors*, Feb. 22-25, 2016, Laguna Beach, CA, USA, pp. 101-104.

Yusheng Wang and Andrei M. Shkel, "Near-Zero Power Wake Up System Enabled by Ultra High Q-Factor Array of MEMS Resonators," *4th International Symposium on Energy Challenges and Mechanics - working on small scales (ECM4)*, Aug. 11-13, 2015, Aberdeen, Scotland, UK.

ABSTRACT OF THE DISSERTATION

Pedestrian Inertial Navigation — Development of Sensors and Algorithms

By

Yusheng Wang

Doctor of Philosophy in Mechanical and Aerospace Engineering

University of California, Irvine, 2020

Professor Andrei M. Shkel, Chair

This Ph.D. dissertation reports on development of inertial sensors and inertial navigation algorithms for pedestrian inertial navigation applications. Toward enabling such applications, this thesis developed new techniques leading to improvements in gyroscopes and navigation algorithms. The main contributions of this thesis include:

- Improved structural symmetry and quality factor of Fused Quartz micro-wineglass resonators. We developed an analytical model to predict the resonant frequency of wineglass modes of the resonators with error less than 20%. A model to predict the frequency split (Δf) of the device was also developed. Directional lapping method was introduced to reduce the as-fabricated Δf by more than $6\times$ without damaging the integrity of the structure, thus preserving the quality factor of the device. Effects of surface loss were observed and analyzed, and surface reflow was demonstrated to be able to improve the quality factor of the devices. A piezoelectric actuation architecture was explored on the Fused Quartz dual shell resonators, in order to eliminate the metal deposition on the vibratory element to minimize the surface-related energy losses.

- Identified and quantified, for the first time, energy dissipation mechanisms in Micro-Electro-Mechanical System (MEMS) resonators. By controlling the temperature, air

pressure, and surface moisture of the device, viscous air damping, Thermo-Elastic Damping (TED), the anchor loss, and the surface loss of the resonator were manipulated and identified. At room temperature, the quality factors related to viscous air damping, TED, the anchor loss, and the surface loss were experimentally measured to be 625,000, 200,000, 1,350,000, and 1,146,000, respectively. This study provides insight to understanding the dominant mechanism that limits the quality factors of MEMS resonators.

- Developed an analytical model to predict the effect of IMU noise on the navigation solution uncertainty in the Zero-Velocity-Update (ZUPT)-aided pedestrian inertial navigation. A bio-mechanical model for human walking was built in order to numerically simulate the process. A discrepancy of less than 10% was shown between the analytical and numerical results. Experiments have been conducted and the results were on the same level of the analytical prediction. Among many IMU noise contributions, the dominant factor affecting the accuracy of the ZUPT-aided pedestrian inertial navigation was Rate Random Walk (RRW) of the z-axis gyroscope. This result is envisioned to aid in analysis of the effect of errors in sensors, which might lead to a well informed selection of sensors for the task of ZUPT-aided pedestrian inertial navigation.

- Improved implementation for the ZUPT-aided pedestrian inertial navigation algorithm for better navigation accuracy. Adaptive threshold based on shock level was first developed for stance phase detection to improve the adaptivity and reliability of the algorithm, demonstrating more than 12× improvement of navigation accuracy. Then, more than 10× reduction in systematic error of position estimation was demonstrated with a careful characterization of the motion of the foot during the stance phases in walking cycles and calibration of IMU in terms of the gyroscope g-sensitivity. Stochastic errors were reduced by 45% when mounting the IMU on the forefoot instead of on the heel.

This work offers new methods and techniques for further reduction of navigation errors by improving the performance of inertial sensors and inertial navigation algorithms.

Chapter 1

Introduction

Motivation of the research is presented in this chapter. First, the background of pedestrian inertial navigation is introduced. It is followed by a literature review of inertial navigation, in terms of both inertial sensors and navigation algorithms. Finally, the research objectives are stated, and the chapter is concluded with an outline of the dissertation.

1.1 Motivation

Navigation is the process of planning, recording, and controlling the movement of a craft or vehicle from one place to another [1]. It is an ancient subject but also a complex science, and it has been more and more important in modern society. Among many sub-categories of navigation, pedestrian navigation is of high interest in many fields, such as human health monitoring, personal indoor navigation, and localizers for first responder systems. Since these systems are generally operated in complex environments, self-contained types of navigation techniques are preferable to non-self-contained techniques, such as Global Navigation Satellite Systems (GNSS) [2, 3]. One of the most well-understood and widely utilized self-

contained navigation techniques is inertial navigation, where system works solely based on inertial sensor (accelerometers and gyroscopes) readouts, making the system robust to any external jamming, perturbation, or interference.

In order to be compatible to pedestrian inertial navigation, the navigation system has to be compact so that it is portable during human walking or running. Therefore, Micro-Electro-Mechanical System (MEMS) based inertial sensors would be a good candidate for their low Costs, small Sizes and Weights, and low Power consumptions (C-SWaP) [4]. On the one hand, the relative tolerance associated with micro-machining process ($10^{-2} - 10^{-4}$) is generally much lower than macro precision machining process (about 10^{-6}). Fabrication imperfections, such as etching non-uniformity, alignment errors, and surface roughness, will induce both structural asymmetry and extra energy dissipation of the devices. On the other hand, standard post-processing techniques, such as high-precision trimming and polishing [5], are lacking in micro-machining process. As a result, it is necessary to develop post-processing techniques in micro-machining process to compensate for as-fabricated imperfections, in order to achieve a good performance in MEMS-based inertial sensors.

Improvement of the performance of MEMS-based inertial sensors will help to reduce, but not fully eliminate the long-term navigation drifts in the strapdown inertial navigation [6]. For example, with the best on-shelf navigation grade Inertial Measurement Units (IMUs), which typically cost hundreds of thousands dollars per axis, the position uncertainty will reach a nautical mile of error within an hour of navigation. Therefore, besides inertial sensor development, navigation algorithm development is another important approach to suppress the accumulated long-term navigation errors. The key considerations in the algorithm development include:

- Effectively suppressing the navigation error accumulation without introducing extra biases or errors;

- Not relying on external signals to preserve the self-contained nature of inertial navigation;
- Requiring moderate computational power and memory to simplify the overall system for portability.

The navigation algorithm satisfying all the requirements above has not been reported previously, and therefore its development is one of the main focuses in this dissertation.

1.2 Background

A short overview of inertial navigation and common terms used throughout the dissertation are given in this section.

1.2.1 Inertial Navigation

The operation of inertial navigation relies on the measurements of accelerations and angular rates, which can be achieved by accelerometers and gyroscopes, respectively. In a typical IMU, there are three accelerometers and three gyroscopes mounted orthogonal to each other to measure the acceleration and angular rate components along three perpendicular directions.

1.2.1.1 Historical Overview

The earliest concept of inertial sensor was proposed by Bohnenberger in the early 19^{th} century [7]. Then in 1856, the famous Foucault pendulum experiment was demonstrated as the first whole angle mode gyroscope [8], whose output was proportional to the change of angle along

the sensitive axis, instead of the angular rate as in the case of most commercial gyroscopes. However, the first implementation of an inertial navigation system did not occur until the 1940s on the V-2 rockets, and the wide application of inertial navigation started in the late 1960s [9]. In the early implementation of inertial navigation, inertial sensors are fixed on a stabilized platform supported by a gimbal set with rotary joints allowing rotation in three dimensions (Fig. 1.1). The gyroscope readouts are fed back to torque motors that rotate the gimbals so that any external rotational motion can be canceled out and the orientation of the platform does not change. This implementation is still in common use where very accurate navigation data is required, and the weight and volume of the system are not of great concern, such as in submarines. However, the gimbal systems are large and expensive due to their complex mechanical and electrical infrastructure. In the late 1970s, strapdown system was made possible, where inertial sensors are rigidly fixed, or "strapped down" to the system. In this architecture, the mechanical complexity of the platform is greatly reduced at the cost of substantial increase in the computational complexity in the navigation algorithm and a higher requirement on the dynamic range for gyroscopes. However, recent development of microprocessor capabilities and suitable sensors allowed such design to become reality. The smaller size, lighter weight, and better reliability of the system further broaden the applications of the inertial navigation. Comparison of the schematics of algorithmic implementations in gimbal system and strapdown system is shown in Fig. 1.2.

1.2.1.2 Strapdown Inertial Navigation

Reference frames are needed to describe the motion of an object. It is of fundamental importance in the inertial navigation to have a proper and precise definition of reference frames. The commonly used Cartesian coordinate frame in pedestrian inertial navigation is the Navigation frame (n-frame). The n-frame is a local geographic frame with its center at the system's location and its axes aligned with the local North, East, and Down directions.

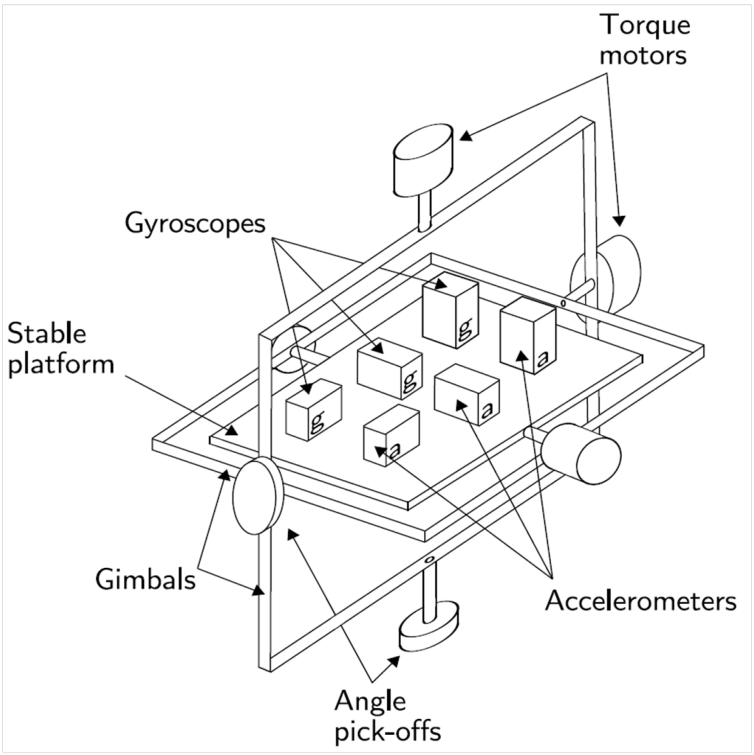

Figure 1.1: A schematic of the IMU implemented on a gimbal system [10].

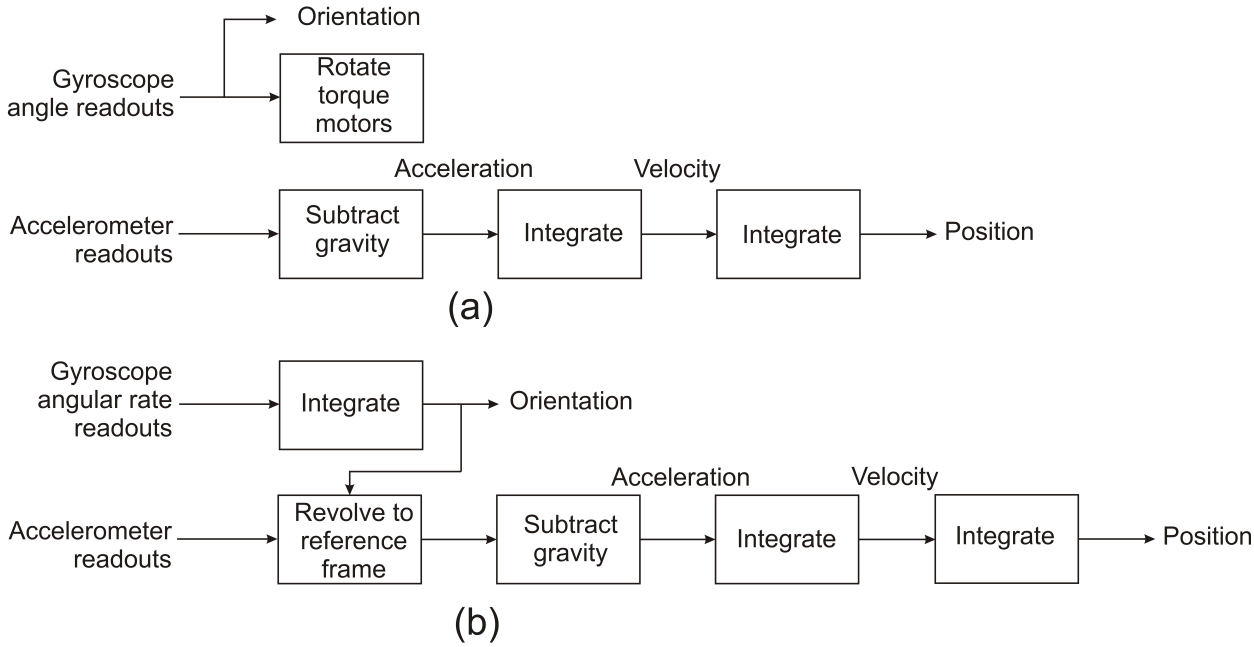

Figure 1.2: Comparison of (a) gimbal inertial navigation algorithm and (b) strapdown inertial navigation algorithm.

Note that the n-frame rotates with the Earth, and the local N, E, and D directions will change. The rate of the orientation change is called transport rate and it will be discussed later. Geographic reference singularities exist over the north and south poles of the Earth in the n-frame.

To calculate the orientation propagation, we introduce the concept of Direction Cosine Matrix (DCM). DCM is a 3×3 matrix defining the relative orientation between different reference frames. The propagation of DCM can be calculated from the angular rate measurements given by the gyroscopes

$$\dot{C}_b^n = C_b^n[\omega_{nb}^b \times], \tag{1.1}$$

where C_b^n is the DCM defining the relative orientation of the frame fixed on the body (b-frame) to the n-frame, $[\cdot\times]$ is the skew-symmetric cross-product-operator, and ω_{nb}^b is the relative angular rate of the b-frame with respect to the n-frame. The superscript denotes the reference frame in which the vector is expressed. ω_{nb}^b can be calculated as

$$\omega_{nb}^b = \omega_{ib}^b - C_n^b(\omega_{ie}^n + \omega_{en}^n), \tag{1.2}$$

where ω_{en}^n is the relative angular rate of the n-frame with respect to the frame fixed on the Earth (e-frame) expressed in the n-frame, and it corresponds to the transport rate mentioned earlier.

Note the inertial sensors measure the motion with respect to an absolutely fixed and stationary frame (i-frame). However in most navigation applications, the quantities of interest are those relative to the e-frame instead of the i-frame. Therefore, a transformation between the e-frame and the i-frame is necessary. Coriolis' theorem is utilized to conduct the time

derivative in the e-frame

$$\left.\frac{d\boldsymbol{r}_{ib}}{dt}\right|_i = \left.\frac{d\boldsymbol{r}_{ib}}{dt}\right|_e + \boldsymbol{\omega}_{ie} \times \boldsymbol{r}_{ib}, \tag{1.3}$$

where $\boldsymbol{r}_{ib}$ is the displacement of the b-frame with respect to the i-frame, and $\boldsymbol{\omega}_{ie}$ is the angular rate of the e-frame with respect to the i-frame, i.e., the Earth's rotation rate. Denote $\boldsymbol{v}_e = \left.\frac{d\boldsymbol{r}_{ib}}{dt}\right|_e$, and take time derivative to (1.3), then the acceleration can be expressed as

$$\boldsymbol{a} = \left.\frac{d^2\boldsymbol{r}_{ib}}{dt^2}\right|_i = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i + \left.\frac{d}{dt}(\boldsymbol{\omega}_{ie} \times \boldsymbol{r}_{ib})\right|_i = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i + \boldsymbol{\omega}_{ie} \times \boldsymbol{v}_e + \boldsymbol{\omega}_{ie} \times (\boldsymbol{\omega}_{ie} \times \boldsymbol{r}_{ib}). \tag{1.4}$$

For velocity propagation, changing rate of $\boldsymbol{v}_e$ in the n-frame can be written as

$$\left.\frac{d\boldsymbol{v}_e}{dt}\right|_n = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i - \boldsymbol{\omega}_{in} \times \boldsymbol{v}_e = \left.\frac{d\boldsymbol{v}_e}{dt}\right|_i - (\boldsymbol{\omega}_{ie} + \boldsymbol{\omega}_{en}) \times \boldsymbol{v}_e. \tag{1.5}$$

Combining (1.4) and (1.5), we obtain

$$\left.\frac{d\boldsymbol{v}_e}{dt}\right|_n = \boldsymbol{a} - (2\boldsymbol{\omega}_{ie} + \boldsymbol{\omega}_{en}) \times \boldsymbol{v}_e - \boldsymbol{\omega}_{ie} \times (\boldsymbol{\omega}_{ie} \times \boldsymbol{r}_{ib}). \tag{1.6}$$

Equation (1.6) can be expressed in the n-frame coordinates as

$$\dot{\boldsymbol{v}}_e^n = C_b^n \boldsymbol{f}^b - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \boldsymbol{v}_e^n - \boldsymbol{\omega}_{ie}^n \times (\boldsymbol{\omega}_{ie}^n \times \boldsymbol{r}_{ib}^n) + \boldsymbol{g}^n. \tag{1.7}$$

Note that in the n-frame inertial navigation mechanism, there are two terms associated with cross product. The first term is due to the rotation of the Earth and the transport rate, and it is called Coriolis acceleration. The second term is a centripetal acceleration due to the Earth's rotation, and it is usually considered as a part of the gravity. The Coriolis acceleration can be neglected in cases where the navigation error caused by the IMU

measurement error is much greater than the Coriolis effect. These cases generally require a short navigation time (less than 10 minutes) and moderate IMU performances (tactical grade or worse). A block diagram of the strapdown inertial navigation mechanism in the n-frame is shown in Fig. 1.3.

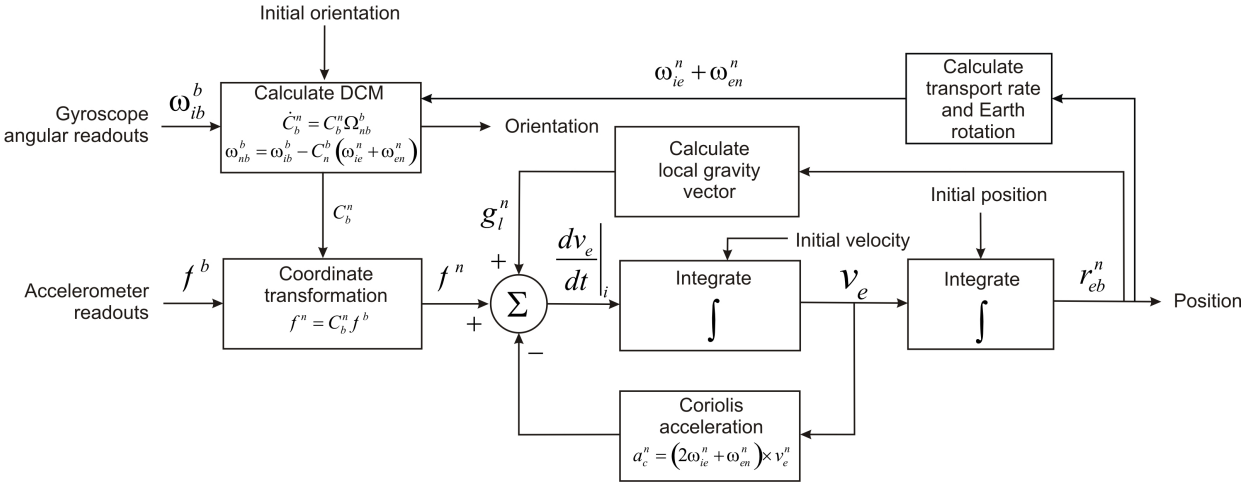

Figure 1.3: Block diagram of strapdown inertial navigation mechanism in the n-frame.

1.2.1.3 Definition of Inertial Sensor Grades

Inertial sensors are categorized by different grades according to their performance characteristics. There is no specific definition on the grades, but inertial sensors can still be generally categorized into four grades: consumer grade, industrial grade, tactical grade, and navigation grade. One of the common standards for categorization is the Bias Instability (BI) of the sensors [11], which indicates how stable the bias of the sensor is over a certain specified period of time. A typical classification of IMUs is listed in Table 1.1.

Table 1.1: Classification of IMU performances in terms of bias instability [12, 13]

	Accelerometer BI	Gyroscope BI	Typical applicable field
Consumer Grade	$>50\text{m}g$	$>100°/\text{h}$	Consumer electronics
Industrial Grade	$1 \sim 50\text{m}g$	$10 \sim 100°/\text{h}$	Automotive industry
Tactical Grade	$0.05 \sim 1\text{m}g$	$0.1 \sim 10°/\text{h}$	Short-term navigation
Navigation Grade	$<0.05\text{m}g$	$<0.01°/\text{h}$	Aeronautics navigation

Consumer Grade The lowest grade of inertial sensors are often referred to as consumer grade. The position error will exceed a meter of error within a few seconds of navigation with unaided consumer grade inertial sensors. Consumer grade inertial sensors are typically used in consumer electronics, such as smart phones, tablets, gaming controllers, and entertainment systems. Most consumer grade inertial sensors are made from MEMS technologies due to the low cost resulted from batch fabrication.

Industrial Grade Industrial grade inertial sensors generally have similar or better noise performances than the consumer grade inertial sensors, but with a better calibration procedure. Due to their relatively low performance, industrial grade IMUs are typically implemented with aiding from other sensors, such as magnetometers and barometers, to be used in navigation-related applications. Estimation methods, such as Extended Kalman Filter (EKF), are typically used to obtain navigation solutions from multiple sensing modalities. Typical applications of industrial grade IMUs include Attitude and Heading Reference System (AHRS), automotive applications, such as Anti-lock Braking System (ABS), active suspension system, and airbag, and aided pedestrian dead-reckoning system. Industrial grade inertial sensors are generally made from MEMS technologies.

Tactical Grade IMUs comprised of tactical grade inertial sensors have the capability of attitude measurement with reasonable errors and are able to conduct short-term navigation, with navigation accuracy on the order of meters within 30s of strapdown inertial navigation. Navigation accuracy on the order of centimeters can be achieved with the integration of GNSS [14]. Tactical grade inertial sensors can be made from MEMS, Fiber Optic Gyroscope (FOG), and Ring Laser Gyroscope (RLG) technologies.

Navigation Grade Navigation grade inertial sensors are some of the highest performance devices that are available for general navigation applications and are applicable for aeronau-

tical navigation with navigation error better than 1 nautical mile per hour. The devices with performance higher than than navigation grade are also available (Navigation +), but are only considered for very specialized types of applications, such as navigation for satellites and submarines. Such a high-end IMU can cost over 1 million dollars and they typically provide navigation solution drifts less than 1 nautical mile per day without any aiding. Navigation grade inertial sensors and above are made from RLG and precision machining technologies.

Table 1.2 summarizes some of the commercially available IMUs regarding their characteristics and classifications. Fig. 1.4 shows the volumes of IMUs of different grades and their navigation errors in 5 minutes without aiding.

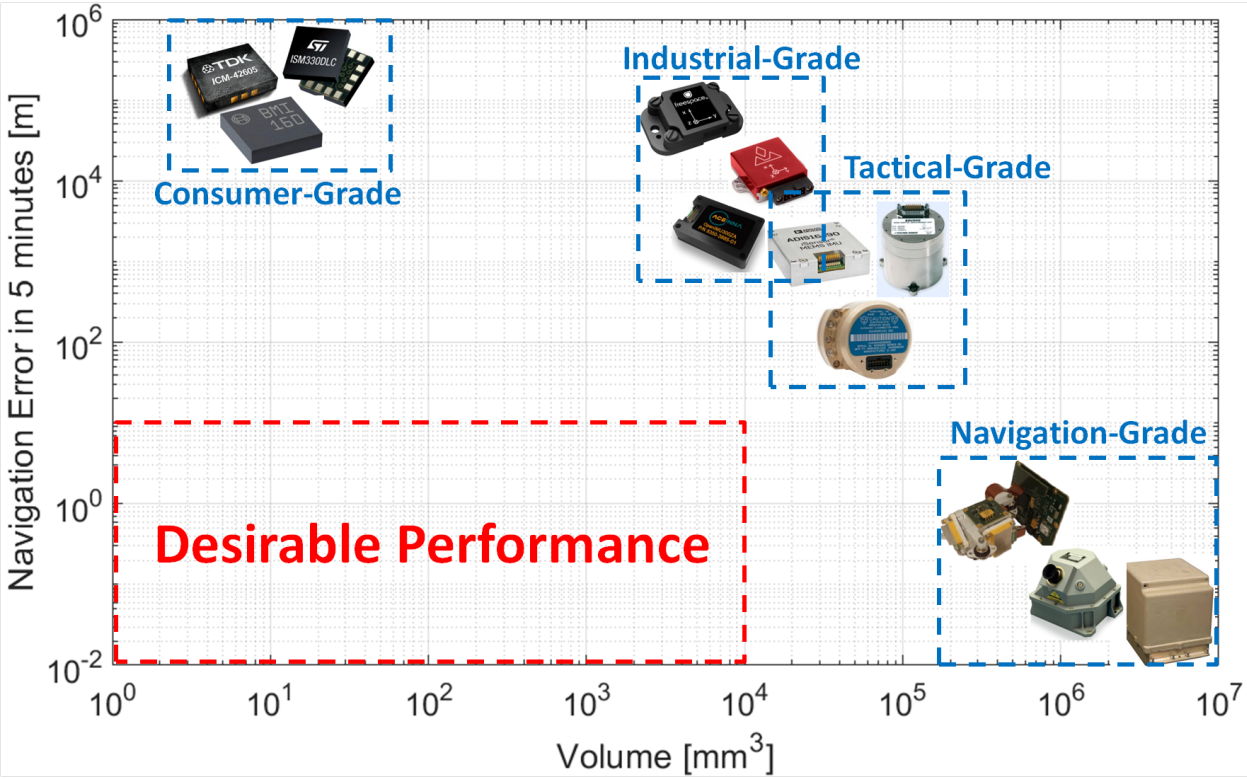

Figure 1.4: Relation between the volumes and the navigation error in 5 minutes of IMUs of different grades. The dashed red box indicates the desired performance for the pedestrian inertial navigation.

Table 1.2: List of some commercial IMUs and their characteristics [15] - [24]

Company	Product	ARW ($^\circ/\sqrt{\text{h}}$)	Gyro BI ($^\circ$/h)	VRW ($\mu g/\sqrt{\text{Hz}}$)	Accel BI (μg)	Volume (mm^3)	Grade
Bosch	BMI160	0.42	252*	180	1800*	6.0	Consumer
STMicroelectronics	ISM330DLC	0.228	270*	75	1800*	6.225	Consumer
TDK InvenSense	ICM-42605	0.228	136.8*	70	700*	6.825	Industrial
Analog Devices	ADI16495	0.12	0.8	13.6	3.2	29,000	Tactical
Honeywell	HG1930	< 0.06	0.25	< 51.8	20	82,000	Tactical
Systron Donner	SDI500	< 0.02	1	100	100	310,000	Tactical
Northrop Grumman	LN-200S	< 0.07	1	35	300	570,000	Tactical
Safran	PRIMUS 400	< 0.002	< 0.01	< 60	< 1000	520,000	Navigation
Honeywell	HG9900	< 0.002	< 0.003	– **	< 10	1,700,000	Navigation
GEM elettronica	IMU-3000	0.0008	0.002	– **	< 40	– **	Navigation

* Data calculated from total RMS noise.
** Data not available.

1.2.1.4 Error Propagation Model

In the inertial navigation algorithm, along with the integration of accelerations and angular rates, all the measurement noises are also integrated and accumulated. As a result, inertial navigation suffers from long-term error accumulations. Error sources include errors from individual inertial sensors, assembly errors of the entire IMU, electronic noises, environment-related errors (temperature, shock, vibration, etc.), and numerical errors. Thus, inertial navigation imposes challenging demands on the system, in terms of the level of errors, to achieve long-term navigation. This partially explains why inertial navigation systems were developed around 100 years later than the development of inertial sensors. It has been shown that without an error-suppressing algorithm, the position error accumulates without bound and approximately proportional to time cubed. For example, for navigation grade IMUs, the navigation error will reach about one nautical mile after an hour of navigation, or equivalently less than 0.01m of navigation error within a minute of navigation. However, for consumer-grade IMUs, which cost few dollars, the navigation error will exceed a meter of error within a few seconds of navigation.

The equations expressing the 3D error propagation in the n-frame can be derived by taking time derivative of (1.1) and (1.7), and neglecting the higher-order terms, if we assume that navigation errors are small compared to the true values. However, an analytical expression for the error of estimation is usually not available due to the unknown and usually complex dynamics of motion during navigation. The results are given without proof below, and more detailed material can be found in [6].

$$\dot{\delta\boldsymbol{\phi}} \approx -\Omega^n_{in}\delta\boldsymbol{\phi} + \delta\boldsymbol{\omega}^n_{in} - C^n_b\delta\boldsymbol{\omega}^b_{ib}, \tag{1.8}$$

$$\dot{\delta\boldsymbol{v}} \approx [\boldsymbol{f}^n\times]\delta\boldsymbol{\phi} + C^n_b\delta\boldsymbol{f}^b - (2\boldsymbol{\omega}^n_{ie} + \boldsymbol{\omega}^n_{en})\times\delta\boldsymbol{v} - (2\delta\boldsymbol{\omega}^n_{ie} + \delta\boldsymbol{\omega}^n_{en})\times\boldsymbol{v} - \delta\boldsymbol{g}, \tag{1.9}$$

$$\dot{\delta\boldsymbol{p}} = \delta\boldsymbol{v}, \tag{1.10}$$

where $\delta\boldsymbol{\phi}$ is the error of orientation estimation in roll, pitch, and yaw angles, $\delta\boldsymbol{v}$ is the error of velocity estimation in the n-frame, and $\delta\boldsymbol{p}$ is the error of position estimation in the n-frame. The above equations can be written in the state-space form

$$\frac{d}{dt}\begin{bmatrix}\delta\boldsymbol{\phi}\\ \delta\boldsymbol{v}\\ \delta\boldsymbol{p}\\ \boldsymbol{b}_g^b\\ \boldsymbol{b}_a^b\end{bmatrix} = \begin{bmatrix} -[\boldsymbol{\omega}_{in}^n\times] & F_{\delta v}^{\delta\dot{\theta}} & 0 & -C_b^n & 0\\ [\boldsymbol{f}^n\times] & C_1 & C_2 & 0 & C_b^n\\ 0 & I & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{\phi}\\ \delta\boldsymbol{v}\\ \delta\boldsymbol{p}\\ \boldsymbol{b}_g^b\\ \boldsymbol{b}_a^b\end{bmatrix} + \begin{bmatrix} C_b^n\cdot\epsilon_{\mathrm{ARW}}\\ C_b^n\cdot\epsilon_{\mathrm{VRW}}\\ 0\\ \epsilon_{\mathrm{RRW}}\\ \epsilon_{\mathrm{AcRW}}\end{bmatrix}, \tag{1.11}$$

where $\boldsymbol{b}_g^b$ is the gyroscope bias expressed in the b-frame, $\boldsymbol{b}_a^b$ is the accelerometer bias expressed in the b-frame, ϵ_{ARW} is the Angle Random Walk of the gyroscopes and ϵ_{VRW} is the Velocity Random Walk of the accelerometers, ϵ_{RRW} and ϵ_{AcRW} are noise terms in the 1st order Markov process in the modeling of the bias $\boldsymbol{b}_g^b$ and $\boldsymbol{b}_a^b$, i.e., RRW and AcRW, respectively, and $F_{\delta v}^{\delta\dot{\theta}}$ is the term related to the transport rate, and it is expressed as

$$F_{\delta v}^{\delta\dot{\theta}} = \frac{1}{R}\begin{bmatrix} 0 & 1 & 0\\ -1 & 0 & 0\\ 0 & -\tan L & 0\end{bmatrix}, \tag{1.12}$$

where R is the Earth radius, L is the latitude of the system, and C_1 and C_2 are the terms related to the Coriolis effects due to the Earth rotation and the transport rate, and they correspond to the third and fourth term of the right-hand-side of (1.9).

Fig. 1.5 shows the simulation result of the propagation of navigation error with different grades of IMUs. The nominal trajectory is a straight line toward the North. For consumer grade IMU (gyroscope BI $> 100^\circ$/h, accelerometer BI $> 50\mathrm{m}g$), the navigation error reaches 10m within 10s. Even for navigation grade IMU (gyroscope BI $< 0.01^\circ$/h, accelerometer BI $< 0.05\mathrm{m}g$), error of position estimation reaches 70m within 1000s, or equivalently about 1

nautical mile per hour, but the speed at which the position estimation error grows increases as time increases. Therefore, we conclude that to achieve the long-term inertial navigation, or to achieve high navigation accuracy with relatively low IMU performances, aiding techniques are needed.

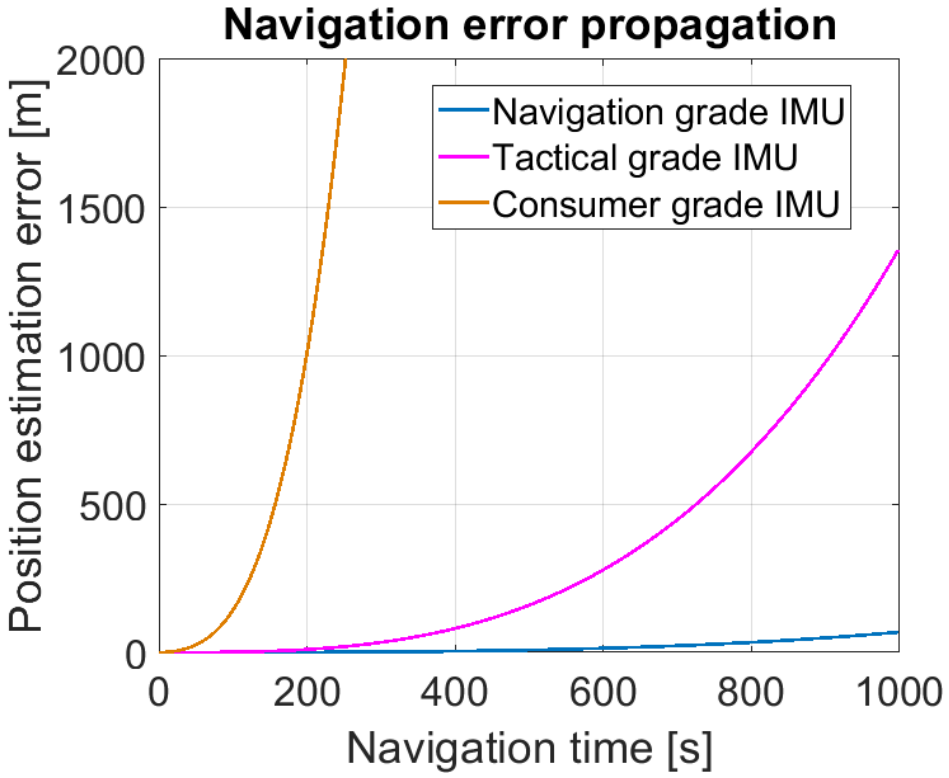

Figure 1.5: Propagation of navigation error with different grades of IMUs.

In most navigation applications, the navigation error is dominated by gyroscopes errors, especially for long-term navigation. Therefore, development of high-performance gyroscope is critical to improve long-term navigation accuracy.

1.2.2 MEMS Vibratory Gyroscopes

Gyroscope is a kind of sensor that measures rotation. Main applications of gyroscopes include automotive rotation detection, platform stabilization, gyrocompassing, and inertial navigation. Gyroscopes can be categorized into different classes according to their operat-

ing physical principle and the involved technology. Some of the gyroscope classes include mechanical gyroscopes, optical gyroscopes, MEMS vibratory gyroscopes, and nuclear magnetic resonance gyroscopes. Their performances and applications are summarized in Fig. 1.6. Since the focus of this dissertation is on pedestrian inertial navigation, where MEMS vibratory gyroscopes are most commonly used due to their small sizes and potential of high performances, we only briefly introduce MEMS vibratory gyroscopes in this section.

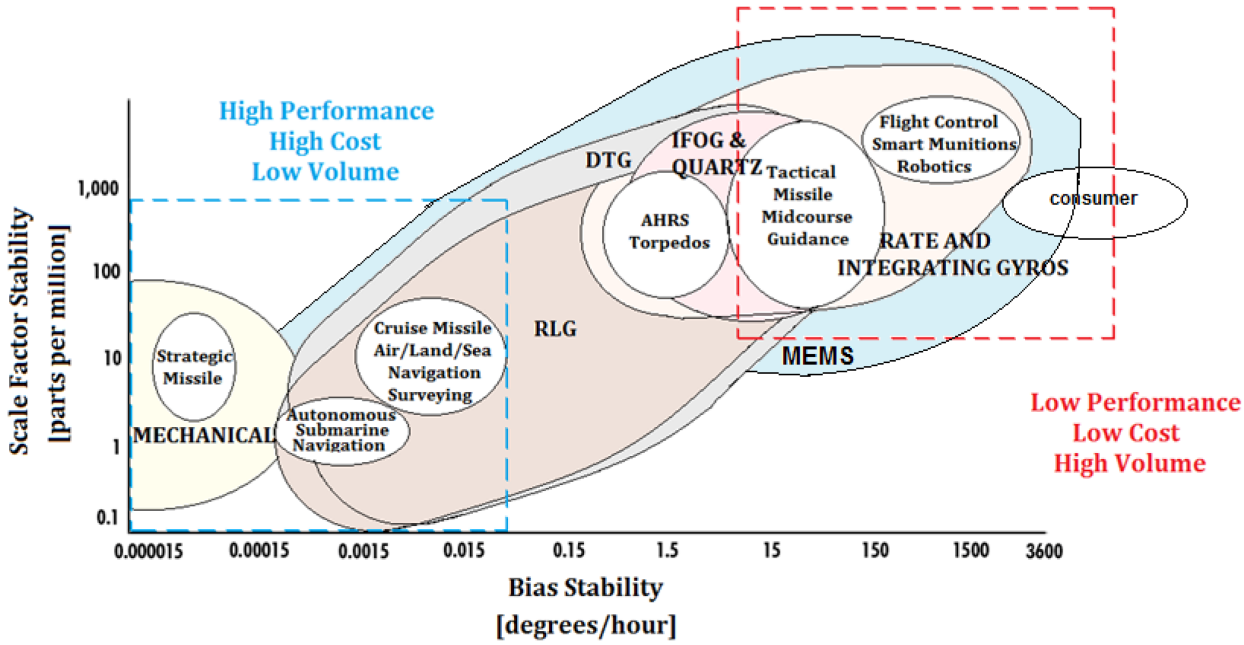

Figure 1.6: Typical performances and applications of different gyroscopes [25].

Unlike mechanical gyroscopes, where spinning rotor and the complicated gimbal system are utilized, MEMS vibratory gyroscopes typically use a vibrating mechanical element, and therefore very small size can be achieved. Although MEMS vibratory gyroscopes may have different geometries to achieve certain goals, they generally can be modeled as a two degree-of-freedom (2-DOF) mass-damper-spring system, as shown in Fig. 1.7. More details on the MEMS vibratory gyroscopes can be found in [26].

MEMS vibratory gyroscopes detect the angular velocity by the Coriolis effect, where a mass moving in a rotating system experiences a force (the Coriolis force) acting perpendicular to the direction of motion and to the axis of rotation. The magnitude of the Coriolis effect is

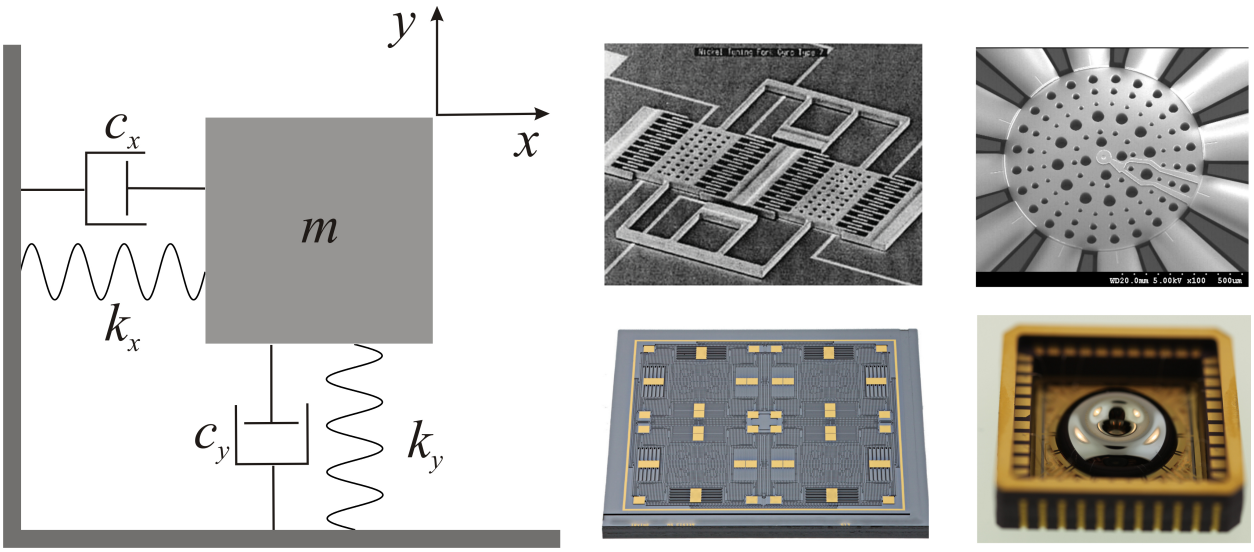

Figure 1.7: Schematics of a MEMS vibratory gyroscope and different possible geometries and configurations [27, 28, 29].

proportional to the angular rate of the frame and the velocity at which the mass moves:

$$\boldsymbol{F}_{\mathrm{c}} = -2m\boldsymbol{\Omega} \times \boldsymbol{v}, \tag{1.13}$$

where m is the mass of the object, $\boldsymbol{\Omega}$ is the angular rate of the frame, and $\boldsymbol{v}$ is the relative velocity of the object with respect to the frame.

1.2.2.1 Allan Deviation

The Allan Deviation (AD) is a method of representing the Root Means Square (RMS) random drift error as a function of averaging time [30]. It is commonly used in the characterization of noises in inertial sensors and clocks. In this section, different types of noises in gyroscopes are introduced and analyzed by AD method.

Quantization Noise It is one of the noises associated with Analog-to-Digital Conversion (ADC). It is caused by the small differences between the actual amplitudes of the analog

signal being sampled and the bit resolution of the analog-to-digital converter. It is a noise with an f^2 Power Spectral Density (PSD) and it shows up on AD [31] with a slope of τ^{-1} (see Fig. 1.8).

Angle Random Walk (ARW) This kind of noise is typically caused by white thermo-mechanical and thermo-electrical noise with much higher frequency than the sampling rate. It is characterized by a white-noise spectrum, meaning it has equal intensity at different frequencies. It shows a slope of $\tau^{-0.5}$ on AD. The effect of this kind of noise is the random walk of angle estimation during navigation.

Bias Instability (BI) The origin of BI is usually the electronics susceptible to random flickering. It is indicated as a sensor bias fluctuation in the data. It has a $1/f$ PSD under the cutoff frequency and it shows a curve on AD with a slope of zero.

Rate Random Walk (RRW) . The origin of RRW is still unknown. It has a $1/f^2$ PSD and shows a curve on AD with a slope of $\tau^{+0.5}$. The effect can be considered as the white noise applied on the bias of the sensor. Therefore, it is also called "random walk."

Drift Rate Ramp This kind of error is related to output change due to temperature change. Strictly speaking, it is not a stochastic error, but we still can analyze it in a similar manner. It has a $1/f^3$ PSD and shows a curve on AD with a slope of τ^{+1}.

1.2.2.2 Operational Modes

There are three operational modes in which a gyroscope can be controlled: open-loop mode, force-to-rebalance mode, and whole angle mode [32], and their ideal responses are shown in

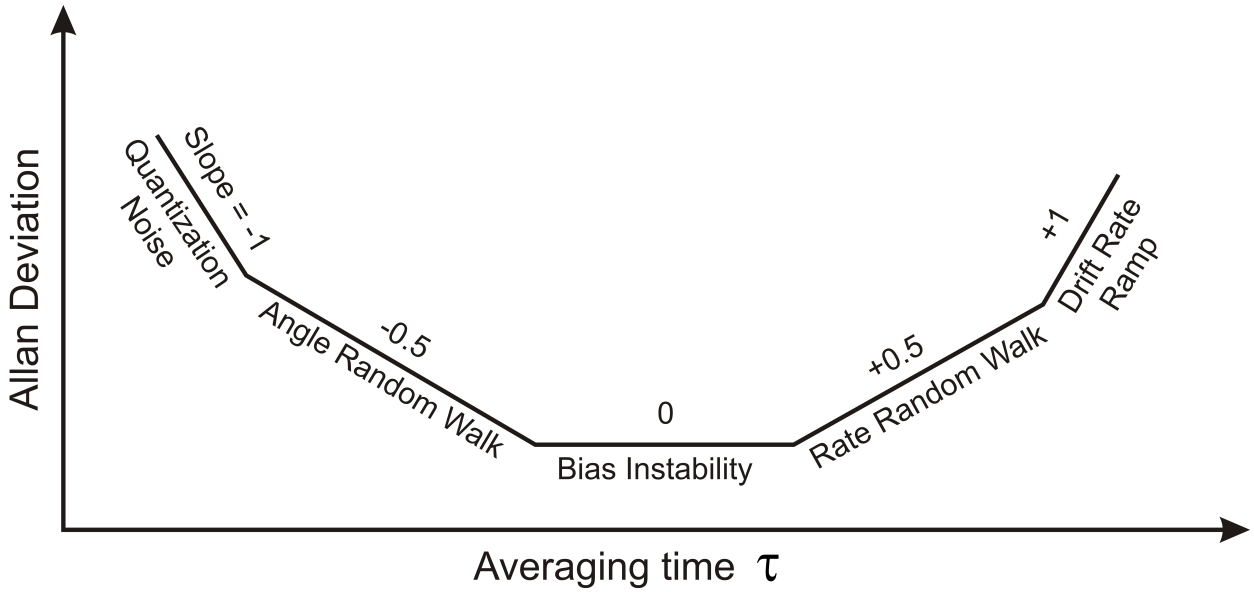

Figure 1.8: A schematic of log-log plot of Allan Deviation.

Fig. 1.9.

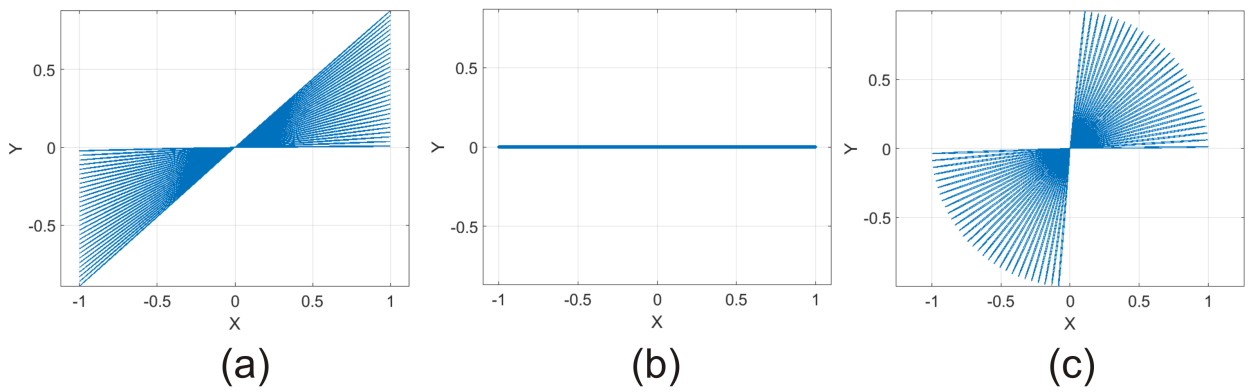

Figure 1.9: Ideal response of a gyroscope operated in (a) open-loop mode, (b) force-to-rebalance mode, and (c) whole angle mode.

Open-Loop Mode The open-loop mode is the simplest gyroscope operational mode, where the proof mass is actuated at resonance with a constant amplitude along the drive axis (x-axis), and no actuation or control is applied along the sense axis (y-axis). If the gyroscope experiences angular rate along its sensitive axis (z-axis in this case), the Coriolis force will take effect and drive the y-axis. It can be shown that the amplitude of motion along the

y-axis is proportional to the angular rate along the z-axis. The open-loop implementation is simple at the cost of small gyroscope bandwidth and low linear measuring range.

Force-to-Rebalance Mode The force-to-rebalance mode has been developed to overcome the limits of the open-loop mode. In the force-to-rebalance mode, control along the drive axis is the same as in the open-loop mode, while the amplitude of motion along the sense axis is suppressed to zero by applying an appropriate drive force. The controlling force to suppress the motion is recorded and it is proportional to the input angular rate. The bandwidth of the gyroscope can be increased to the resonant frequency of the device, at the cost of more complicated control and lower signal-to-noise ratio.

Whole Angle Mode In the whole angle operational mode, the proof mass vibrates freely in the x-y plane, and the orientation of the oscillation precesses due to the Coriolis effect if there is input rotation along the z-axis. Unlike the previous two operational modes, the output of the gyroscopes operated in the whole angle mode is angle, instead of angular rate. As a result, the need for numerical integration of the angular rate into orientation is eliminated during navigation. Besides, gyroscopes operated in the whole angle mode have theoretically unlimited mechanical bandwidth, allowing them to work under severe dynamics.

1.2.2.3 Error Analysis in the Whole Angle Mode

In the ideal case, the gyroscope operated in the whole angle mode should be completely symmetric and have no energy dissipation. The equation of motion can be expressed as

$$\begin{aligned} \ddot{x} + (\omega^2 - \eta'\Omega_z^2)x - \eta\dot{\Omega}_z y &= \frac{F_x}{m_{eq}} + 2\eta\dot{y}\Omega_z, \\ \ddot{y} + (\omega^2 - \eta'\Omega_z^2)y - \eta\dot{\Omega}_z x &= \frac{F_y}{m_{eq}} - 2\eta\dot{x}\Omega_z, \end{aligned} \tag{1.14}$$

where ω is the resonant frequency of the resonator, Ω_z is the input angular rate along the z-axis, m_{eq} is the equivalent mass of the system, η is the angular gain of the gyroscope, η' is the parameter associated with the centrifugal force, and F_x and F_y are the force applied along the x and y axes, respectively.

However, due to design limits and fabrication imperfections, aniso-elasticity and aniso-damping always exist, and they will affect the gyroscope performance. In this section, we briefly introduce the relation between them. More details can be found in [33]. The equation of motion for a non-ideal gyroscope operated in the whole angle mode can be expressed as

$$\begin{aligned}
&\ddot{x} + (\frac{1}{\tau_1} + \frac{1}{\tau_2})\dot{x} + (\frac{1}{\tau_1} - \frac{1}{\tau_2})(\dot{x}\cos 2\theta_\tau + \dot{y}\sin 2\theta_\tau) - \eta\dot{\Omega}_z y \\
&\quad + (\frac{\omega_1^2 + \omega_2^2}{2} - \eta'\Omega_z^2)x - \frac{\omega_1^2 - \omega_2^2}{2}(x\cos 2\theta_\omega + y\sin 2\theta_\omega) = \frac{F_x}{m_{eq}} + 2\eta\dot{y}\Omega_z, \\
&\ddot{y} + (\frac{1}{\tau_1} + \frac{1}{\tau_2})\dot{y} + (\frac{1}{\tau_1} - \frac{1}{\tau_2})(\dot{x}\sin 2\theta_\tau - \dot{y}\cos 2\theta_\tau) - \eta\dot{\Omega}_z x \\
&\quad + (\frac{\omega_1^2 + \omega_2^2}{2} - \eta'\Omega_z^2)y - \frac{\omega_1^2 - \omega_2^2}{2}(x\sin 2\theta_\omega - y\cos 2\theta_\omega) = \frac{F_y}{m_{eq}} - 2\eta\dot{x}\Omega_z,
\end{aligned} \tag{1.15}$$

where ω_1 and ω_2 are the resonant frequencies along the two principal axes of elasticity, respectively, τ_1 and τ_2 are the time constants along the two principal axes of damping, respectively, θ_ω and θ_τ are the orientations of the principal axes of elasticity, and principal axes of damping, respectively. Then, the error propagation of the gyroscope can be expressed as:

$$\begin{aligned}
\dot{E} =& -\frac{2}{\tau_0}E - \Delta\tau^{-1}\cos 2(\theta - \theta_\tau)\sqrt{E^2 - Q^2} - \frac{a}{\omega_0}f_{as}\cos\delta\phi + \frac{q}{\omega_0}(f_{qc}\cos\delta\phi + f_{qs}\sin\delta\phi), \\
\dot{Q} =& -\frac{2}{\tau_0}Q - \Delta\omega\sin 2(\theta - \theta_\theta)\sqrt{E^2 - Q^2} + \frac{a}{\omega_0}(f_{qc}\cos\delta\phi + f_{qs}\sin\delta\phi) - \frac{q}{\omega_0}f_{as}\cos\delta\phi, \\
\dot{\theta} =& -\eta\Omega_z + \frac{1}{2}\Delta\tau^{-1}\sin 2(\theta - \theta_\tau)\frac{E}{\sqrt{E^2 - Q^2}} + \frac{1}{2}\Delta\omega\cos 2(\theta - \theta_\theta)\frac{Q}{\sqrt{E^2 - Q^2}} \\
&- \frac{a}{2\omega_0\sqrt{E^2 - Q^2}}(f_{qs}\cos\delta\phi - f_{qc}\sin\delta\phi) + \frac{q}{2\omega_0\sqrt{E^2 - Q^2}}f_{as}\sin\delta\phi,
\end{aligned} \tag{1.16}$$

where E presents the total energy in the system, Q is the quadrature term, θ is the orientation of oscillation, which is extracted and used as the output of the gyroscope, f_{qs}, f_{qc}, and f_{as} are potential forces used to control the gyroscope, $\delta\phi$ is the phase difference between the oscillation and the reference, and

$$\frac{1}{\tau_0} = \frac{1}{2}(\frac{1}{\tau_1} + \frac{1}{\tau_2}), \quad \Delta\tau^{-1} = \frac{1}{\tau_1} - \frac{1}{\tau_2}, \quad \omega_0 = \sqrt{\frac{\omega_1^2 + \omega_2^2}{2}}, \quad \Delta\omega = \frac{\omega_1^2 - \omega_2^2}{2\omega_0},$$
$$a = \sqrt{\frac{1}{2}(E + \sqrt{E^2 - Q^2})}, \qquad q = \sqrt{\frac{1}{2}(E - \sqrt{E^2 - Q^2})}.$$

Equation (1.16) shows that, in order to achieve ideal performance, we need to make sure that

$$\begin{aligned} \tau_0 &= \infty, \\ \Delta\tau^{-1} &= 0, \\ \Delta\omega &= 0. \end{aligned} \tag{1.17}$$

This leads us to the important criteria regarding the gyroscopes operated in the whole angle mode:

1. The time constant of the resonator τ_0 needs to be maximized, in order to minimize the energy needed to sustain the oscillation;

2. The aniso-damping $\Delta\tau^{-1}$ needs to be minimized, in order to minimize the drift of the output θ;

3. The aniso-elasticity $\Delta\omega$ needs to be minimized, in order to minimize the quadrature term Q and also minimize the drift of the output θ.

1.2.3 Pedestrian Inertial Navigation

Pedestrian inertial navigation has been made possible by the miniaturization of IMUs. For example, Fig. 1.10 shows the IMU developed for the Apollo program in 1960's and a typical modern MEMS-based IMU developed in 2010's. The IMU for the Apollo missions has a volume of 1100in^3 (or 1.8×10^7mm^3) and a weight of 42.5lb [34], whereas the volume of the shown MEMS-bases IMU is 8.55mm^3 and the weight is on the order of tens of milligrams. Six orders of magnitude of both volume and weight reduction has been demonstrated over the past 50 years, due to the MEMS technology and strapdown configuration.

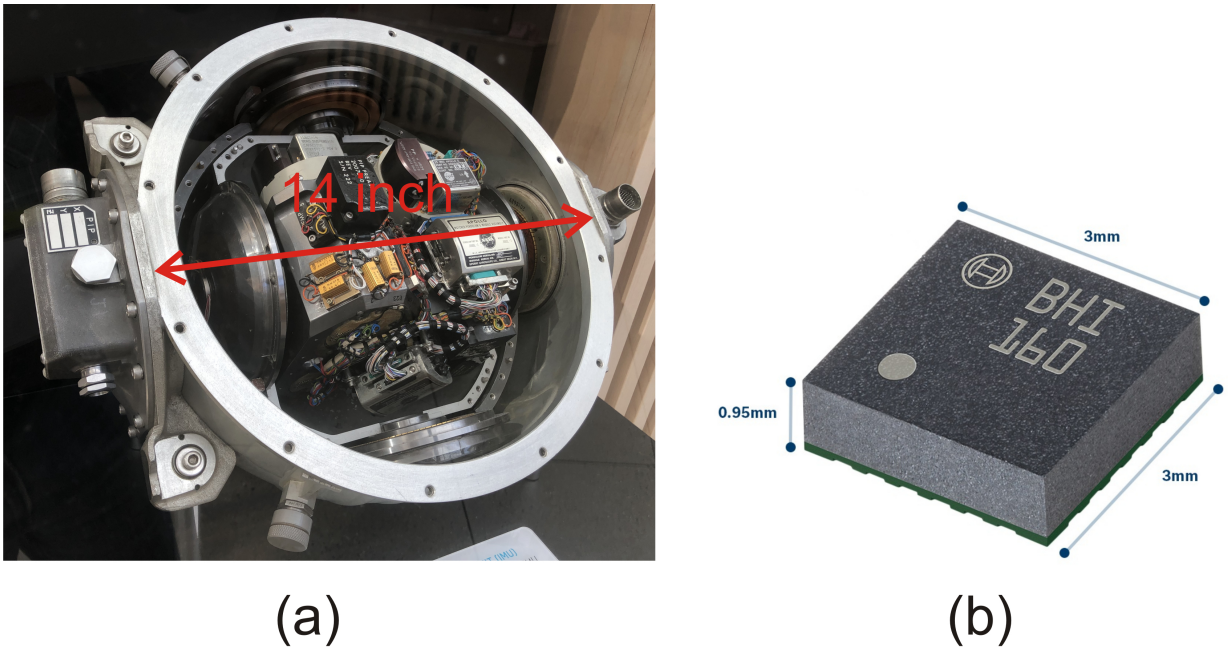

Figure 1.10: Comparison of (a) the IMU developed for the Apollo program in 1960's [35] and (b) a modern MEMS-based IMU developed in 2010's [15].

Due to the limits on the size and weight of the IMUs, most pedestrian inertial navigation systems can only use consumer grade or industrial grade IMUs, and the drift of navigation error is intolerable over long period of time. As a result, aiding techniques are necessary in pedestrian inertial navigation.

1.2.3.1 Aiding techniques

Aiding techniques aim to provide the navigation system more information about the motion in order to improve the navigation accuracy. The extra information can be obtained by some known pattern of motion or other sensing modalities.

In pedestrian inertial navigation, to take advantage of the known pattern of the motion, the IMU can be mounted on a fixed part of the body, such as foot, shank, thigh, waist, shoulder, and head. Among all the mentioned mounting positions, foot is the most commonly used one due to its simple motion during walking. During the stance phase of a walking cycle, the foot periodically returns to stationary state when it is on the floor. The stationary state can be used to limit the long-term velocity and angular rate drift, thus greatly reduce the navigation error. The most commonly used algorithm is the Zero-Velocity-Update (ZUPT) aiding algorithm [36]. In this implementation, IMU is fixed on the foot to perform navigation and detect the stance phase at the same time. Whenever the stance phase is detected, the zero-velocity information of the foot will be input into the system as a pseudo-measurement to compensate for IMU biases and navigation errors, thus reducing the navigation error growth in the system. One of the main advantages of ZUPT is its ability to obtain pseudo-measurement of the velocity, which is otherwise unobservable by IMUs. There are two key parts involved in the ZUPT algorithm: the stance phase detector, and the pseudo-measurement of the motion of the foot. It is also possible to take advantage of the full bio-mechanical model of human gait instead of just the motion of the foot during walking. This approach typically requires multiple IMUs fixed on different parts of human body and relates the recorded motions of different parts through some known relationships derived from the bio-mechanical model [37].

Many aiding techniques have been developed to combine with inertial navigation to improve the navigation accuracy. Magnetometry is one of the most commonly used techniques,

where the heading angle can be extracted from the measurement of the Earth's magnetic field vector, if the roll and pitch angles are available from the measurement of the gravity [38]. Measuring orientation instead of the angular rate, as in most commercial gyroscopes, eliminates the angle drifts, which is a major source of navigation errors in dead-reckoning. However, it requires a "magnetically clean place" to operate, which limits its applicability [39]. Another way of implementing estimation of absolute position is computer vision. Images are captured and matched against the pre-acquired database for localization [40]. This approach potentially requires very large databases of the environment. Simultaneous Location And Mapping (SLAM) method has also been developed, which doesn't require a pre-acquired database. However, this method requires intensive calculation and mostly deals with static environment, and it suffers from odometry drifts [41]. Barometers are widely used to measure the altitude of the system during navigation [42]. This aiding technique is simple but is vulnerable to disturbance of environmental pressure [43], and it is limited to measurements only along the vertical direction.

Ranging technique can be utilized as a self-contained aiding for pedestrian navigation. In this technique, the transmitter sends out a signal (ultrasonic wave or electromagnetic wave) which is received by the receiver. The distance between the transducers can be calculated either by measuring the time of flight (the time difference between transmitting and receiving of the wave), or the decrease of the magnitude of the wave. This technique can be categorized as self-contained if both the transmitter and the receiver are within the system whose state is to be estimated. For example, in the foot-to-foot ranging, the transmitter and receiver are placed on two feet of a person to keep track of the distance between them [44]. In the cooperative localization, ranging technique is applied to measure the distance between multiple agents as a network to improve the overall navigation accuracy of each of the agent [45]. Ranging sensors measure the distance between the sensors and it is therefore a relative measurement, resulting in a limited observability and thus a limited improvement of navigation accuracy [46].

1.2.3.2 Extended Kalman Filter

Kalman filter is a widely applied estimation tool in time series analysis used in fields such as signal processing, navigation, and motion planning. It is the optimal linear estimator for linear system with additive independent white noise [47]. Kalman filter fuses different measurements of the system associated with their measurement uncertainties, and estimates unknown system states that tends to be more accurate than the results obtained from any single measurement. However, most systems are nonlinear in practice, and Extended Kalman filter (EKF) was proposed to linearize the nonlinear system about the estimate of the current mean and covariance.

The EKF works in a system with the following given discrete state transition and measurement models

$$\begin{aligned} \boldsymbol{x}_k &= \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{v}_k, \\ \boldsymbol{z}_k &= \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{w}_k, \end{aligned} \tag{1.18}$$

where k is the time step, $\boldsymbol{f}(\cdot)$ is the state transition function, $\boldsymbol{h}(\cdot)$ is the measurement function, $\boldsymbol{x}_k$ is the system state, $\boldsymbol{u}_k$ is the control input, $\boldsymbol{z}_k$ is the measurement, $\boldsymbol{v}_k$ is the process noise, $\boldsymbol{w}_k$ is the measurement noise, both are zero-mean Gaussian noises with covariance Q_k and R_k, respectively. Note that $\boldsymbol{f}(\cdot)$ and $\boldsymbol{h}(\cdot)$ may not be linear.

There are two steps in the EKF: the predict step and the update step. In the predict step, the EKF provides a state estimate of the current time step (called the *a priori* state estimate) based on the state estimate from the previous time step and the current control input. In the update step, the EKF combines the *a priori* estimate with the current measurement to adjust the state estimate. The adjusted estimate is called the *a posteriori* state estimate. The update step can be skipped if the measurement is unavailable.

The EKF proceeds as follows:

Predict step

a priori state estimate	$\hat{\boldsymbol{x}}_{k\|k-1} = \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1\|k-1}, \boldsymbol{u}_k) + \boldsymbol{v}_k$	(1.19)
a priori estimate error covariance	$P_{k\|k-1} = F_k P_{k-1\|k-1} F_k^{\mathrm{T}} + Q_k$	(1.20)

Update step

Measurement residual	$\boldsymbol{\nu}_k = \boldsymbol{z}_k - \boldsymbol{h}(\hat{\boldsymbol{x}}_{k\|k-1})$	(1.21)
Kalman gain	$W_k = P_{k\|k-1} H_k^{\mathrm{T}} (H_k P_{k\|k-1} H_k^{\mathrm{T}} + R_k)^{-1}$	(1.22)
a posteriori state estimate	$\hat{\boldsymbol{x}}_{k\|k} = \hat{\boldsymbol{x}}_{k\|k-1} + W_k \boldsymbol{\nu}_k$	(1.23)
a posteriori estimate error covariance	$P_{k\|k} = (I - W_k H_k) P_{k\|k-1}$	(1.24)

where $\hat{\boldsymbol{x}}_{m|n}$ denotes the estimate of $\boldsymbol{x}$ at time step m given measurements up to time step n, and $P_{m|n}$ denotes the corresponding estimate covariance, and

$$F_k = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \mid_{\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k}, \quad H_k = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}} \mid_{\hat{\boldsymbol{x}}_{k|k-1}} .$$

1.2.3.3 Application of the EKF in Pedestrian Inertial Navigation

In the pedestrian inertial navigation, the EKF is commonly used to fuse the IMU readouts with other aiding techniques to obtain a better navigation result. In most cases, instead of the navigation state itself, the EKF is designed to work on the navigation errors as the state space, in order to avoid the issues related to the linearization of the highly nonlinear motion dynamics [48].

The diagram of pedestrian inertial navigation with the EKF is presented in Fig. 1.11. The strapdown inertial navigation algorithm is first applied on IMU readouts to predict the cur-

rent navigation states (position, velocity, and orientation), while in the EKF, the navigation errors are used as the system state. Whenever other aiding techniques (for example, ZUPT, bio-mechanical modeling, ranging) are available, their readouts are compared with the predicted navigation states, and the difference is used as the measurement in the EKF to update the system state. The EKF is able to estimate both the IMU errors, which can be used as a feedback to compensate for the raw IMU input, and the navigation errors, which is directly used to update the navigation result.

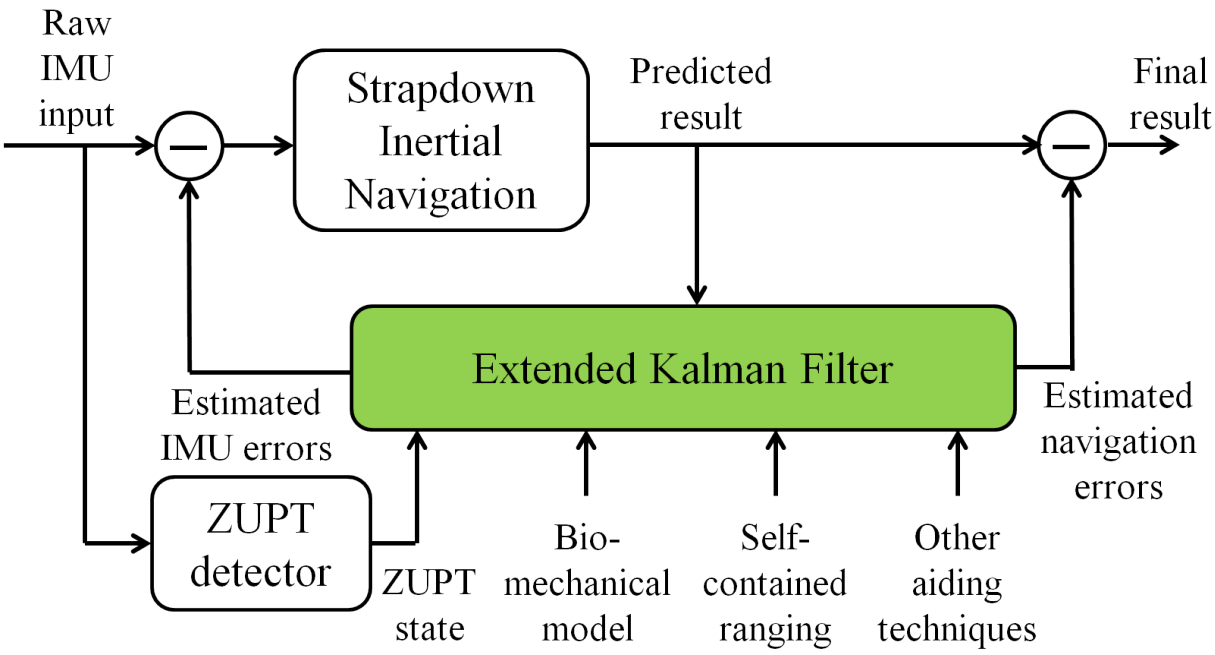

Figure 1.11: Diagram of the pedestrian inertial navigation with EKF.

1.3 Literature Review

This section presents state-of-the-art on methods to improve accuracy of pedestrian inertial navigation. In general, the efforts can be categorized into two aspects: improvement of the performance of inertial sensors, especially gyroscopes, and development of aiding techniques in the navigation system.

1.3.1 Imperfection Compensation Methods for MEMS Devices

In this section, imperfection compensation methods for MEMS devices are discussed as the approach to improve the performance of inertial sensors. The general goal of imperfection compensation is to reduce the frequency mismatch and damping mismatch and to increase the quality factor of the devices.

1.3.1.1 Methods for Frequency Mismatch Reduction

An ideal gyroscope operated in the whole angle mode has two degenerate vibratory modes that share the same mode shape and resonant frequency. Fabrication imperfections, however, will induce the frequency mismatch between the two modes and limit the performance of the device. In this section, we review various methods reported in literature to reduce the frequency mismatch of the device.

First, we discuss the effects of electrostatic biasing for frequency mismatch tuning.

- The success of electrostatic tuning has been experimentally demonstrated on planar axisymmetric resonators, such as ring resonators [49], disk resonators [50, 51], and dual-mass resonators [52]. Model to predict mass and stiffness imperfections based on Nyquist plot has been derived [53]. Tuning capability is related to the capacitive gaps, which are typically defined in two ways: lithography or sacrificial layer. In the first case, capacitive gaps are about 5μm and they are limited by the resolution of lithography. In the other case, a process named HARPSS (High Aspect-Ratio combined Poly and Single-crystal Silicon) was developed and was able to fabricate sub-micron capacitive gaps [54].

- Electrostatic tuning conducted on micro-scale 3D resonators was also investigated. Tuning on hemispherical resonators [55, 56] and hemi-toroidal resonators [57] have been

demonstrated. In all these cases, the capacitive gaps were defined during assembly and ranged from 10μm to 30μm. Conformal capacitive gaps of 7μm was demonstrated on 3D micro shell resonator utilizing electroplated photoresist as sacrificial layer [58]. Large capacitive gaps limit the tuning capability of the system.

Then, we discuss the permanent tuning methods of the devices.

- A method of trimming was proposed and numerically verified on single-ring resonators [59, 60]. The effect of the addition of a number of imperfection masses on a perfect ring was first considered by applying Rayleigh-Ritz approach, assuming that the mode shapes were identical to those of a perfect ring. Then, the inverse (the so-called trimming) problem was deduced. For the special case of trimming a single pair of modes (n=2 mode), analytic solutions for the magnitude and position of the single required trimming mass was derived.

- Laser ablation was experimentally demonstrated on single-ring silicon resonators to tune both out-of-plane and in-plane modes frequency mismatch [61]. A model considering both mass perturbation and stiffness perturbation was derived to guide the experiment. Laser ablation was performed at a wavelength of 532nm, and pulse energies of up to 150mJ with pulse widths of 4ns were available at repetitive rates of up to 20Hz. A spot size of 10μm was demonstrated. Out-of-plane frequency mismatch was reduced from 26Hz to 7Hz. The quality factor of the resonator after laser ablation, however, was reduced due to amorphization, laser induced stress, and microcracking. A significant amount of debris was created around the ablated lines, inducing unwanted errors.

- Mass perturbation method was investigated on axisymmetric multi-ring resonators [62]. The resonator consisted of a set of concentric rings that were affixed to neighboring

rings by a staggered system of spokes. Reservoirs and solder sphere pads were fabricated on the outer layers of spokes for deposition of mass points. Quality factor of the resonators was not influenced by mass point deposition, since the spokes did not deform during vibration. Resonators with wineglass resonant frequency of 14kHz were tuned by this method and the frequency mismatch was reduced from 30Hz to 80mHz. However, it is difficult to apply this method to 3D devices due to its complexity.

- Large reduction in frequency mismatch was demonstrated by ablating mass from the rim of a diamond hemispherical resonator [63]. The diamond tabs at the lip of the resonator were co-fabricated with the hemisphere and coated with Cr/Au, which were laser ablated for tuning. 130fs duration of laser pulse at 400nm wavelength with 100μJ was used. Apertures were used to reduce the power to avoid damaging the diamond tabs under the Au layer. Frequency mismatch was reduced from 35.5Hz to 0.35Hz by this method. The use of tabs prevented any stress induced by trimming from propagating into the hemisphere.

- Frequency split reduction by chemical etching was performed on hemispherical resonators [64]. Analytical model was derived to relate the mass perturbation of hemispherical resonator and the its frequency split. The frequency split was reduced to below 50mHz by controlling the immersion depth, inclination angle, and etching time. Quality factor was demonstrated to remain on the same level of 10,000 in the air. However, no results were reported on the surface finish of the device after chemical etching, and the fact that the quality factor remained the same in the air could not verify the effects of extra surface loss related to the etching. Besides, this method was mainly suitable for macro-machined hemispherical resonators but not compatible with MEMS-based resonators.

- High resolution micro ultrasonic machining (HR-μUSM) process was proposed for post fabrication trimming of complex 3D microstructures made from Fused Quartz (FQ)

[65]. The process was first characterized on flat Fused Quartz samples and demonstrated 30nm surface roughness with mass removal rate of 20ng/min. Boron carbide and tungsten carbide abrasive powders with grain sizes as low as 100nm were used in the slurry to achieve a higher surface quality. Amplitude of the vibration of the tool tip was 1.5μm and the tool wear ratio was less than 4%. The trimming was also performed on bird-bath shells and holes with depth of 18μm, diameter of 60μm, and roughness Sa of 120–150nm were machined.

1.3.1.2 Methods for Quality Factor Improvement

Maximization of quality factor is critical to improve the performance of vibratory MEMS devices [66, 67]. Many factors are found to influence the quality factor of a device, such as viscous air damping, surface related losses, anchor loss, thermo-elastic damping (TED), and internal losses. In this section, we review different damping mechanisms and the methods to eliminate these factors to improve the overall quality factor of the devices.

Viscous Air Damping Viscous air damping is significant in micro-scale level due to the low Reynolds number. Two categories of air damping in 2D structures, squeeze film damping [68, 69] and lateral damping [70], have been thoroughly studied. Viscous air damping is well-understood and can be modeled by either Computational Fluid Dynamics (CFD) for 3D devices with complex structures [71], or some simplified model based on pre-calculated modules and empirical superposition [72]. Since air damping increases rapidly as the resonator's surface-to-volume ratio increases [73], it is typically the dominant damping mechanism for MEMS devices if operated in air. A common method to eliminate the air damping is to operate the devices in vacuum. It has been reported that vacuum level as high as 20μTorr is needed to completely eliminate the effect of air damping for 3D wineglass shells [74].

Thermo-Elastic Damping Thermo-Elastic Damping (TED) is related to the coupling of thermal and elastic deformation fields of the resonator through the Coefficient of Thermal Expansion (CTE) of the material. More specifically, it is caused by the irreversible heat transfer between the stretched regions and compressed regions of the resonator, and its magnitude is related to the structure of the device and material properties, such as CTE. This phenomenon was first identified and modeled in cantilever beams in [75, 76]. For devices with more complicated geometry, such as vented beam resonators [77] and 3D shell resonators [78], Finite Element Analysis (FEA) is able to simulate the model accurately. Cr/Au metal coating has been reported to greatly reduce the quality factor of Fused Quartz micro shell resonators due to the large difference between Fused Quartz and metals [79]. Low CTE materials, such as diamond (1 ppm/°C), ULE glass (0.03 ppm/°C), and Fused Quartz (0.5 ppm/°C), are preferable to reduce TED of the device. Proper structural design can also help to reduce TED. For example, it was numerically demonstrated in [77] that adding vent sections in the clamped end region of cantilever beams can reduce TED by 3 orders of magnitude. Particle Swarm Optimization (PSO) has been proposed to optimize the resonator structural geometry for lower TED for cantilever beam resonators [80] and disk ring resonators [81].

Anchor Loss Anchor loss refers to the acoustic energy radiated into the supporting substrate through the anchor of the resonator [66]. The magnitude of anchor losses is related to the location of the anchor relative to the vibration mode shape. Therefore, anchors are usually placed at the node point of the mode shape to minimize the energy loss. The high accuracy of the anchor loss estimation for MEMS resonators was demonstrated by applying the modified Fourier Transform method [82]. Analytical solution to predict anchor loss has been derived in devices with simple geometries, for example cantilever beams [83] and contour mode resonators [84]. Semi-analytical models have also been developed for a better modeling accuracy [85]. However, for more complicated structures, numerical simulation

is still the only option. To the best of our knowledge, Perfectly Matched Layer (PML) is the only available approach to model the anchor loss phenomenon. PML is an absorbing boundary, which absorbs incoming waves over a wide frequency range for any non-zero angle of incidence [86]. The size of the PML should be on the order of or greater than the solid wavelength in the device to avoid any energy reflected back into the device. Anchor losses in disk resonators [87], bulk mode resonators [88], and hemispherical shell resonators [89] have been predicted by the PML method. Various tuning-folk-type structures have been demonstrated to greatly reduce the anchor loss by canceling the stresses from the two identical parts of the resonator, so that the net force and torque applied on the substrate is almost zero [90, 91, 92, 93]. For wineglass resonators, the anchor loss was minimized by the self-aligned supporting stem at the center of the device [94]. Isolation of the anchor region by spiral or circumferential slots has been numerically demonstrated to reduce the anchor loss [63]. Phononic crystal was proposed for silicon substrate and numerically demonstrated to reduce the anchor loss for Silicon Carbide (SiC) disk resonators [95].

Surface Loss Surface related losses have been observed on micro polysilicon and single-crystal silicon cantilever beams [96, 97], and Fused Quartz fibers [98]. The surface roughness, formation of the oxide layer, coating of metal layer, surface adsorbate, and Sub-Surface Damage (SSD) can all reduce the resonator quality factor. Various methods have been reported to improve the resonator quality factor by improving the surface quality. Flame polishing was reported to increase the quality factor of Fused Quartz fibers by two times [98]. Natural gas flame was used to heat the sample for 15 minutes, so that the surface reached the glass transition temperature. It was believed that surface defects were reduced by flame polishing, so that the quality factor was increased. For single-crystal silicon cantilever beams, heating in nitrogen for 1h at 700 °C was reported to increase the quality factor by a factor of 4 [97]. Forming gas (Ar with 4.25% H_2) was also reported to be effective. The increase of quality factor was believed to be the result of the removal of surface contaminants and

defects. Removal of SiO_2 was also identified as a reason for quality factor improvement, [96]. The quality factor of macro-scale machined Cylindrical Resonator Gyroscope (CRG) was improved by two times using 10:1 NH_4F and HF etching for 2h at the cost of increased surface roughness [99]. 4 hours of annealing at 700 °C was proved to decrease the surface roughness of the dry etched glass sidewall from 900nm to 40nm [100]. The reason was the removal of SSD, which is defined as the residual fractured and deformed material in the near surface region of brittle materials. It is created during the polishing and lapping of the surface [101] and its depth can be estimated by the size of the abrasive used during the grinding operation [102]. Annealing at 700 °C was demonstrated to improve the quality factor in diamond cantilever beams [63]. Ion-beam smoothing has been experimentally demonstrated to improve the surface roughness of diamond resonators, but no quality factor improvement was reported [63].

Internal Loss Internal losses refer to the energy losses due to the imperfections inside the material of the devices, such as impurities, residual stress, defects, and grain boundaries. Metallic impurities, OH, and Cl are three major impurities in Fused Quartz material and it wasn shown that their concentrations could influence the quality factor of the devices [103, 104]. High purity material is needed to achieve the high quality factor. The effects of grain boundaries were demonstrated on polydiamond resonators by achieving 3 times higher quality factor on microcrystalline diamond resonators than nanocrystalline diamond resonators [105]. This indicates the advantage of single crystal and amorphous materials over polycrystalline materials. The effects of annealing was demonstrated on Fused Quartz hemispherical resonators by increasing the quality factor by 50% [106]. Factors that contributes to the increased quality factor were elimination of possible water molecules, fabrication residues, and reduced OH content. The same effect was also observed on Fused Quartz cylindrical resonator [104].

Akhiezer Damping Akhiezer Damping (AKE) is related to the phonon-phonon interaction within the devices, and energy is dissipated through phonon population relaxation [107]. Quality factor due to AKE is inversely proportional to the resonant frequency of the device if the resonant frequency is generally in MHz range, and is independent of the resonant frequency in GHz range [108]. AKE is more dominant in the resonators of high frequencies, and can be neglected in resonators of frequency in kHz range. SiC has been widely used to suppress AKE (almost 30× lower than silicon [109]) for high-frequency resonators due to its large Young's modulus, acoustic velocity, band gap and stable physico-chemical properties in harsh environments [110].

1.3.2 IMU-Based Aiding in Pedestrian Inertial Navigation

Many aiding techniques have been developed in pedestrian inertial navigation. However, in this section, we only cover the IMU-based aiding techniques, with a goal of not adding any extra sensing modalities. Besides applying strapdown inertial navigation algorithm to IMU readouts, many other methods exist to fully exploit the information contained in IMU data. In this section, we review the recent development in this field.

1.3.2.1 Zero-Velocity Update

The first few publications demonstrating the concept, implementation, and experimental results of the zero-velocity update (ZUPT) algorithm include [111, 112, 113], and more details of the algorithm and implementation were reported in [36].

A stance phase detector is needed to detect the event when the foot is on the floor. Mathematically, the stance phase detector can be formulated as a binary hypothesis testing problem [114]. Generalized Likelihood Ratio Test (GLRT) can be conducted to form the detector.

Different hypothesis can be made on the features of the IMU readout during the stance phase, and different stance phase detectors can be developed accordingly. The simplest type is called acceleration moving variance detector, where a stance phase is detected if the variance of the accelerometer readout over a short period of time is small [115]. Equivalently, the change in accelerometer readout can also be directly used as an indicator [36]. Another type of stance phase detector is called acceleration magnitude detector, which determines that the IMU is stationary if the magnitude of the measured specific force vector is close to the gravity [116]. Gyroscope readout can also be used in the stance phase detection. Angular rate energy detector detects the stance phase if the root-mean-square of the gyroscope readout is small [117]. A more widely used detector is called Stance Hypothesis Optimal dEtector (SHOE), which is a combination of acceleration magnitude detector and angular rate energy detector, where both accelerometer and gyroscope readouts are utilized [118]. It has been shown that the SHOE detector outperforms marginally the angular rate energy detector, and the acceleration moving variance detector and the acceleration magnitude detector are not as good as the previous two [114].

For all the detectors mentioned above, determination of the stance phase is conducted by comparing the test statistic to a threshold. However, different gait patterns lead to different gait dynamics, and as a result, different thresholds are required. The simple way is to tune the threshold in an ad-hoc manner to achieve the best performance [119]. However, this is not practical in real navigation applications due to the lack of the ground truth in most navigation applications. Some other alternatives include: adjusting the parameters based on the period of gait cycle (or equivalently, the walking speed) obtained by pre-setting the walking speed [120], and applying Pseudo Wigner-Ville Distribution (SPWVD) to the gyroscope readout to extract gait frequency [121]. Sensor fusion type approach is another option to adaptively detect the stance phases. For example, some reported to mount the pressure sensors in the shoe sole to detect the pressure between the shoe and the floor [122], while others used multiple IMUs to detect the motion of foot, shank, and thigh simultaneously to improve

detection accuracy [37].

After identifying the stance phase, pseudo-measurement on the motion of the foot during the stance phase is fed into the EKF, and different types of pseudo-measurement have been reported. The most commonly used pseudo-measurement is the zero-velocity information, where the velocity of the system is simply set to be zero. The simple pseudo-measurement is associated with a linear measurement model and no extra parameter is needed in the system [123]. More complicated models have been proposed to simulate the motion of the foot during the stance phase. For example in [44], a pure rotation of the foot around a pivot point near the toes was assumed during the stance phase. In this model, the rotational speed during the stance phase could be extracted from the gyroscope readout, and the distance between the pivot point and the IMU could be measured or manually tuned. A better navigation accuracy was demonstrated with the rotational pseudo-measurement during the stance phase at the cost of nonlinear measurement model and more parameters involved. Zero-Angular-Rate-Update (ZARU) has also been proposed, where a zero angular rate of the foot is used as a pseudo-measurement during the stance phases [124]. ZARU provides the system more observability of the yaw angle, but but it is shown to be accurate enough only when the subject stands still [113].

1.3.2.2 Bio-Mechanical Model

Another approach is to take advantage of bio-mechanical model of human gait instead of just the motion of the foot during walking. This approach typically requires multiple IMUs fixed on different parts of human body and relates the recorded motions of different parts through some known relationships derived from the bio-mechanical model. For example, a double-pendulum model for the swing phase and an inverted pendulum model for the stance phase may be utilized. A complete kinematic model was reported and utilized in [125], where 7 IMUs were mounted on two feet, two tibias, two thighs, and pelvis, respectively, so

that the motion of the entire lower body could be modeled and recorded. In this study, the orientations of the six segments of the lower body were first estimated by gyroscopes. Then, foot-mounted IMUs were used to detect the stance phases of the foot motion. Next, the positions of all segments could be estimated after setting the foot segment as root nodes during the stance phases. However, this study was mainly focused on human gait reconstruction, instead of pedestrian inertial navigation.

In [126], a bio-mechanical model of walking was used to improve the navigation accuracy, where two IMUs were mounted on the upper thigh and the foot during navigation, respectively. The orientation of the thigh-mounted IMU was used to relate to the kinematic motion of the leg by a bio-mechanical model, and position estimate error was improved by 50%. Besides adding more information into the system, one of the main advantages of thigh-mounted IMUs and tibia-mounted IMUs over foot-mounted IMUs is that they will experience a much smoother and smaller magnitude of motion, which poses lower requirements on the IMU performance in terms of measurement range, bandwidth, and the coupling between axes.

It is also possible to take advantage of some patterns during walking, instead of the full bio-mechanical model. One simple implementation is to mount two IMUs on two feet, respectively. On top of the two ZUPT-aided inertial navigation on each of the foot, a constraint of maximum allowed separation of the position estimates of the two feet was imposed [127]. In this way, the systematic errors in the drift of heading angles of the two feet can be partially canceled due to their symmetric nature.

In [128], two IMUs were mounted on the knee and waist, respectively. The IMU mounted on the knee was used to count the step numbers and to measure the z-axis acceleration, which was correlated to the stride length. The IMU mounted on the waist was used to track the orientation of the motion. Navigation error of less than 1m was demonstrated over a 100m trajectory. Note that this method is based on stride length estimation instead of strapdown inertial navigation.

1.3.2.3 Machine Learning

More recently, machine learning was explored in Human Activity Recognition (HAR), stride length estimation, and ultimately in the pedestrian inertial navigation.

The first Convolutional Neural Network (CNN) approach to HAR was introduced in [129], where an 1D array representation of the motion signals was used as input, instead of 2D. This method was demonstrated to be more efficient than the state-of-the-art, and also competitive in accuracy. In [130], a deep CNN was applied to multi-modal time-series sensor data for HAR that performs a 2D convolution on the last convolutional layer. The approach was evaluated on two public available HAR datasets and outperformed the other deep learning state-of-the-art methods. However, distinguishing standing from sitting activity was still challenging.

HAR can be used to compensate for IMU errors, and thus improve the accuracy of pedestrian inertial navigation, for example in [131]. Two IMUs were mounted on the foot and the head, respectively. Then, CNN was applied to the IMU readouts to perform motion classification. Finally, the classification result could be used to calibrate the head-mounted IMU by compensating for the gyroscope long-term drift.

In [132], the deep CNN was used to estimate the stride length by mapping stride-specific inertial sensor data to the resulting stride length. An average accuracy and precision of 0.01 ± 5.37cm was demonstrated on a public dataset consisting of 1220 strides from 101 geriatric patients. Combined with yaw angle estimation, this method has the potential of performing inertial navigation without performing any integral to the IMU readouts. Furthermore, robustness of the algorithm was improved by enabling the stride length estimation for atypical strides, such as small steps, side steps, and backward steps [133]. The stride detection rate of 100% for normal walking steps, and about 98% for atypical steps was demonstrated.

HAR can also help to improve the robustness of stance phase detection in the ZUPT-aided pedestrian inertial navigation. In [134], a Support Vector Machine (SVM) classifier using inertial data recorded by a single foot-mounted sensor to differentiate between six motions was trained. Following the motion classification, an adaptive threshold for stance phase detection was derived based on the classification results. A mean test classification accuracy of over 90% was reported. 3× navigation error reduction was demonstrated over a 130m trajectory.

1.4 Research Objective

The goal of this dissertation is to develop high-accuracy self-contained pedestrian inertial navigation system. Such systems are generally composed of two parts: the inertial sensors that measure the motion, and the navigation algorithm that processes the sensor readouts and estimates the motion states. In this dissertation, two corresponding approaches are explored toward this goal, including analysis and compensation of sensor imperfections, and study and reduction of navigation errors.

To achieve long-term pedestrian inertial navigation, development of MEMS-based gyroscopes that achieve low noise, high measuring range, and large bandwidth is the key. Fused Quartz 3D wineglass resonator for the whole angle gyroscope implementation is a good candidate, and it was developed at the University of California, Irvine [74, 135, 136, 137]. However, no effective method to compensate for the as-fabricated imperfections has been developed. In this dissertation, the fabrication imperfections that affect the energy dissipation and structural asymmetry are analyzed and compensated. More specifically, a model to predict the wineglass resonant frequency and frequency split of the device is developed. Then, a method to compensate for the structural asymmetry without damaging the integrity of the device is developed. The surface effects on the overall quality factor are investigated and

methods to reduce the surface losses are explored. Piezoelectric actuation is explored to avoid the metal deposition on the device surface to preserve the high quality factor of the devices. A methodology to experimentally decouple and quantify different energy dissipation mechanisms in MEMS resonators is developed.

Another objective of this dissertation is the development of inertial navigation algorithm to further reduce the navigation errors. ZUPT-aided pedestrian inertial navigation algorithm has been demonstrated as an effective method [36, 138]. This dissertation first develops a close-form analytical model to relate the IMU errors to the navigation errors in the ZUPT-aided pedestrian inertial navigation, which is envisioned to aid in analysis of the effect of errors in sensors, and it might lead to a well informed selection of sensors for the task of the ZUPT-aided pedestrian inertial navigation. Then, improvement of the algorithm, in terms of its implementation, adaptivity, estimation bias, and stochastic noise level is explored to achieve better robustness and accuracy.

1.5 Dissertation Outline

In Chapter 2, compensation of structural asymmetry in 3D Fused Quartz wineglass resonator gyroscope is presented. First, an analytical model based on inextensional assumption is derived to predict the resonant frequency with error less than 20%. Then, a numerical model is developed based on the analytical model to estimate the frequency split of the device. Next, for the first time, a permanent structural asymmetry compensation method is presented. Using the technique, we demonstrate a near 6× reduction of structural asymmetry ($n = 2$ wineglass mode), culminating in reduction of the frequency split from 41Hz to 7Hz. This procedure demonstrates the ability to reduce the structural asymmetry, while not affecting the overall quality factor of the resonators.

In Chapter 3, efforts to reduce the surface-related energy dissipation in 3D Fused Quartz wineglass resonator gyroscope are presented. The effects of different thermal and chemical post-processing on the surface quality of Fused Quartz are studied, including thermal reflow, potassium hydroxide (KOH) etching, Buffered Oxide Etching (BOE), 10:1 Hydrogen Fluoride and Hydrogen Chloride solution (HF/HCl) etching, and RCA-1 surface treatment. Thermal reflow of Fused Quartz at 1300 °C for one hour achieves the best result. The improvement in surface roughness of micro-structures is shown to directly correlate to improvements in the quality factor. An alternative implementation of piezoelectric actuation is explored on Fused Quartz dual shell resonators, in order to avoid the metal coating on the surface to minimize the surface loss. Electrode-shaping is conducted to maximize the transduction rate of piezoelectric actuation, and selective metal deposition by shadow mask is experimentally demonstrated as the enabling technique. The frequency response of the device is modeled to prove the feasibility of the approach.

In Chapter 4, we present a methodology on the quantification of energy dissipation of MEMS resonators and the possibility of quantifying the contribution of each mechanism independently. The main energy dissipation mechanisms in the study include viscous air damping, TED, anchor loss, and surface loss. At room temperature, the quality factors related to viscous air damping, TED, and anchor loss are experimentally measured to be 625,000, 200,000, and 1,350,000, respectively. The effects of moisture-related surface loss have been demonstrated over a 2-year period of time. This study provides more insight to better understand the dominant mechanism that limits the quality factor of MEMS resonators, in order to effectively improve the design and achieve better device performances.

In Chapter 5, a study on the effects of IMU noises on the navigation solution uncertainty in the ZUPT-aided pedestrian inertial navigation is presented. The analytically derived attitude, velocity, and position propagation errors reveal that among many IMU noise terms, the dominant factor affecting the accuracy of ZUPT-aided pedestrian navigation is Rate

Random Walk (RRW) of the z-axis gyroscope. A discrepancy less than 20% between the analytical, numerical, and experimental results supports fidelity of the analytical estimates. This study offers a close-form analytical prediction for the errors of ZUPT-aided pedestrian inertial navigation due to IMU noises.

In Chapter 6, an effort on the reduction of navigation error in the ZUPT-aided pedestrian inertial navigation without adding other sensing modalities is reported. Both stochastic errors and systematic errors are addressed. Adaptive threshold for stance phase detection is developed to improve the robustness of the algorithm, especially in navigation with varying walking and running speeds. A systematic error reduction of more than 10× is demonstrated by compensating the residual velocity during the stance phase and the gyroscope g-sensitivity. Reduction of stochastic errors is demonstrated by Circular Error Probable (CEP) reduction of 45%. The analytical analysis and experimental verification show a great improvement of navigation accuracy and robustness of the developed compensation algorithms.

Finally, Chapter 7 concludes the dissertation with a summary of contributions and future research directions.

Chapter 2

Sensor Development — Structural Asymmetry Compensation

This chapter investigates the structural asymmetry compensation method on Fused Quartz 3D wineglass resonators. In Section 2.1, an analytical model for resonant frequency prediction is derived as the basis of this chapter. Based on this model, a numerical analysis on structural symmetry is presented to study two main factors that will introduce asymmetry to the device in Section 2.2. Then, Section 2.3 presents the directional lapping procedure to compensate for the structural asymmetry. More analysis on the directional lapping procedure is discussed in Section 2.4. The chapter concludes with discussion of the results in Section 2.5.

2.1 Frequency Model

In this section, we introduce an analytical model of hemi-toroidal shell (Fig. 2.1) based on the inextensional wineglass mode shape assumption. The natural frequency of a per-

fectly symmetric shell was derived by applying the Rayleigh's energy method and verified experimentally.

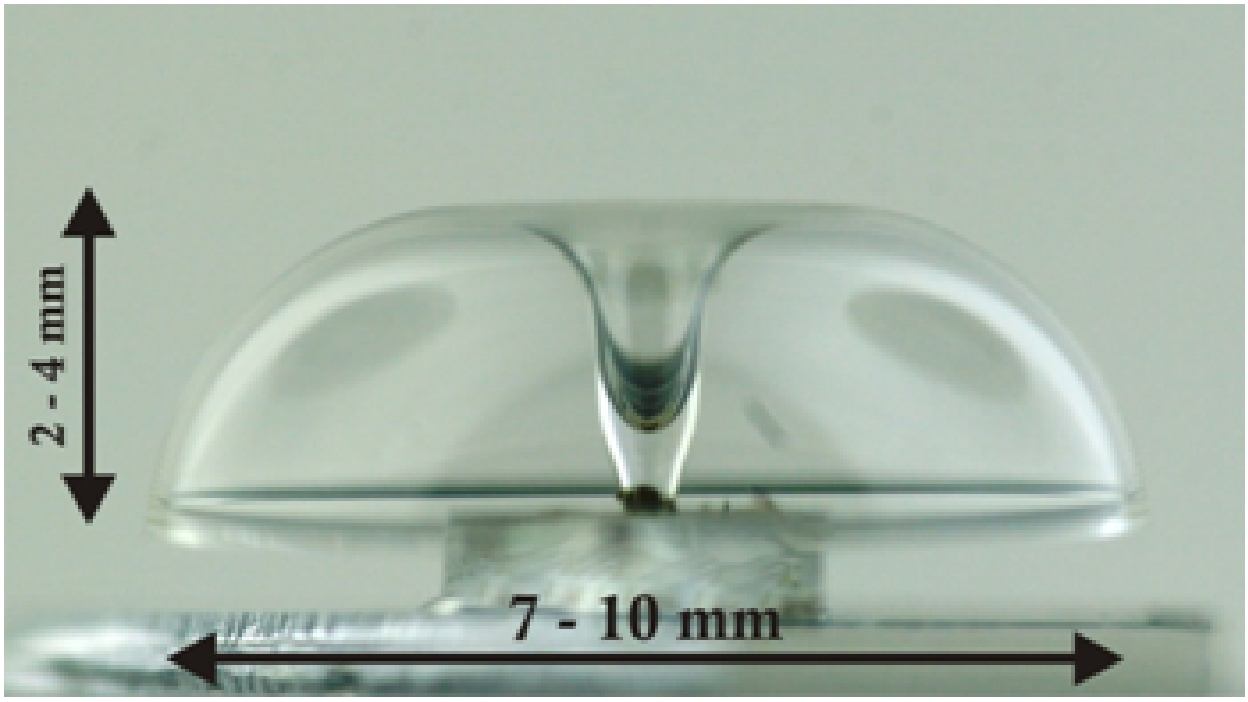

Figure 2.1: Typical geometry of hemi-toroidal shell fabricated using high temperature micro-glassblowing process of fused quartz [139].

2.1.1 Derivation of Mode Shapes

Fig. 2.2 shows a thin hemi-toroidal shell with thickness h and radius R. In spherical coordinates, the shape of the shell can be expressed as $r = 2R\sin\theta$, where r is the radial distance and θ is the polar angle. Since the shell is axisymmetric, r is independent of ϕ, and the latter is defined as the azimuth angle. Motion of the shell structure was assumed to be completely described by the motion of its middle surface. We denoted the local displacement components of the middle surface to be δr, $\delta\theta$, and $\delta\phi$, where δr is the linear displacement along r, while $\delta\theta$ and $\delta\phi$ are the angular displacements along θ and ϕ, respectively. In the

wineglass mode, the displacement components of two matched modes were expressed as

$$\begin{aligned} \delta r &= U(\theta)\sin n\phi\cos\omega t & & \delta r = U(\theta)\cos n\phi\cos\omega t \\ \delta\theta &= V(\theta)\sin n\phi\cos\omega t & \text{and} \quad & \delta\theta = V(\theta)\cos n\phi\cos\omega t \\ \delta\phi &= W(\theta)\cos n\phi\cos\omega t & & \delta\phi = W(\theta)\sin n\phi\cos\omega t \end{aligned} \tag{2.1}$$

where n is called the mode number, ω is the angular frequency of the mode, and U, V, and W are mode shape functions of variable θ [140], and they are to be determined later in this section. The terms in (2.1) containing ϕ define the orientation of the mode shape and the terms containing ω define the frequency. Note that orientations of the mode shapes with respect to the azimuth angle are separated by $\pi/2n$. For $n = 2$ wineglass mode, for example, the orientations of mode shapes are separated by 45 degrees.

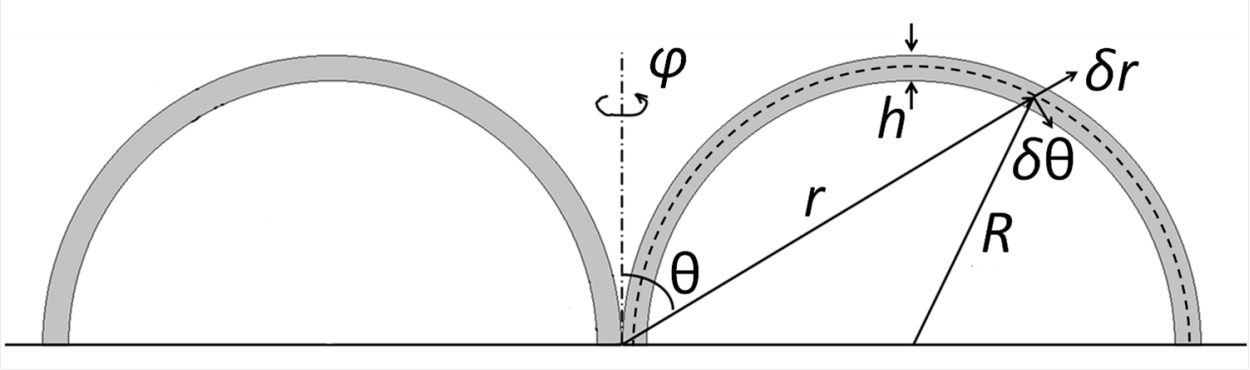

Figure 2.2: Coordinate system, middle surface (dashed line) and parameters of hemi-toroidal shell.

Inextensional assumption was applied to calculate the mode shape functions. It assumed that the strain of the middle surface of the shell remained zero during the deformation. This assumption held if thickness of the shell was much smaller than the other dimensions [140]. In the case of glassblown structures, the thickness of the shell was on the order of 100μm, while the outer radius was on the order of 10mm and the height was about 3mm (Fig. 2.1). Hence, due to the ratio of dimensions (100:1 and 30:1), the inextensional assumption was justifiable to apply for the structures of interest.

Let us consider an arbitrary line element with length ds on the middle surface of the shell at position (r, θ, ϕ) and its length components dr, $d\theta$, and $d\phi$. After deformation of the shell, the position of the element became $(r + \delta r, \theta + \delta\theta, \phi + \delta\phi)$ and its length became $ds + d\delta s$. In the spherical coordinate system, the length of the line element could be expressed as:

$$\begin{aligned}(ds + d\delta s)^2 = (dr + d\delta r)^2 + \\ (r + \delta r)^2 \sin^2(\theta + \delta\theta)(d\phi + d\delta\phi)^2 + \\ (r + \delta r)^2 (d\theta + d\delta\theta)^2.\end{aligned} \tag{2.2}$$

We expanded the differential elements in displacement and geometry with respect to coordinates θ and ϕ as follows:

$$\begin{aligned}d\delta r &= \frac{\partial \delta r}{\partial \theta} d\theta + \frac{\partial \delta r}{\partial \phi} d\phi, \\ d\delta\theta &= \frac{\partial \delta\theta}{\partial \theta} d\theta + \frac{\partial \delta\theta}{\partial \phi} d\phi, \\ d\delta\phi &= \frac{\partial \delta\phi}{\partial \theta} d\theta + \frac{\partial \delta\phi}{\partial \phi} d\phi, \\ dr &= \frac{dr}{d\theta} d\theta.\end{aligned}$$

The radial distance of hemi-torus r was only dependent on the polar angle θ and not related to the azimuth angle ϕ. Therefore, there was no $d\phi$ term in the expansion of dr.

When we considered only the first order terms in (2.2), the resulting equation for the deformation of the line element took the form:

$$\begin{aligned}\frac{d\delta s}{ds} = (\frac{dr}{d\theta}\frac{\partial \delta r}{\partial \theta} + r^2 \frac{\partial \delta\theta}{\partial \theta} + r\delta r)\frac{d\theta^2}{ds^2} + \\ (r\delta r \sin^2\theta + r^2 \sin\theta\cos\theta\delta\theta + r^2 \frac{\partial \delta\phi}{\partial \phi}\sin^2\theta)\frac{d\phi^2}{ds^2} + \\ (\frac{dr}{d\theta}\frac{\partial \delta r}{\partial \phi} + r^2 \frac{\partial \delta\theta}{\partial \phi} + r^2 \frac{\partial \delta\phi}{\partial \theta}\sin^2\theta)\frac{d\theta d\phi}{ds^2}.\end{aligned} \tag{2.3}$$

According to the inextensional assumption, the length of the line element should not change after deformation no matter what values dr, $d\theta$, and $d\phi$ would take, implying that the coefficients of all terms on the right hand side of (2.3) were zero. Substituting (2.2) in (2.3) and canceling the two variables δr and $\delta\theta$ would cancel the common orientation term (sinusoidal term with respect to ϕ) and the frequency term (sinusoidal term with respect to t). The resulting fundamental equation of the mode shape of the hemi-toroidal shell became

$$(\cos^2\theta - \sin^2\theta)\frac{d^2}{d\theta^2}r_\phi + 4\sin\theta\cos\theta\frac{d}{d\theta}r_\phi + \frac{(2s^2-1)}{\sin^2\theta}r_\phi = 0, \tag{2.4}$$

where $r_\phi = rW(\theta)\sin\theta$ represented the linear displacement along the azimuth angle. We inteded to solve (2.4) for r_ϕ instead of $W(\theta)$ to avoid the coefficient of the second order term from being zero when $\theta = 0$. Note that the mode number n was included in (2.4), indicating that the equation allowed to calculate the mode shapes of any order.

Equation (2.4) was a linear second-order ordinary differential equation with varying coefficients. A clamped boundary condition was assumed at $\theta = 0$, corresponding to the attachment of the stem of the device to a substrate. The collocation method was applied to solve the equation numerically [141]. Hermite polynomials of order three were used to approximate the solution. The solution with $n = 2$ was shown by the blue solid line in Fig. 2.3. The red dashed line in Fig. 2.3 was the normalized result from Finite Element Analysis (FEA); COMSOL Multiphysics package was used for the FEA modeling. Convergence of the analysis was achieved with meshing elements on the order of 25,000. The largest error was about 1% of the maximum displacement, where θ was about 0.5rad. The small error indicated the fidelity of the developed analytical model. Note that the amplitude of motion was close to zero when $\theta < \pi/4$, corresponding to the inner part of the device, and the vibratory energy was mostly limited in the outer part of the device. It implied that geometric deviation of the inner part of the real device from the model would only induce a small

disturbance of the resonant frequency of the device, which expanded the range where the model could be applied.

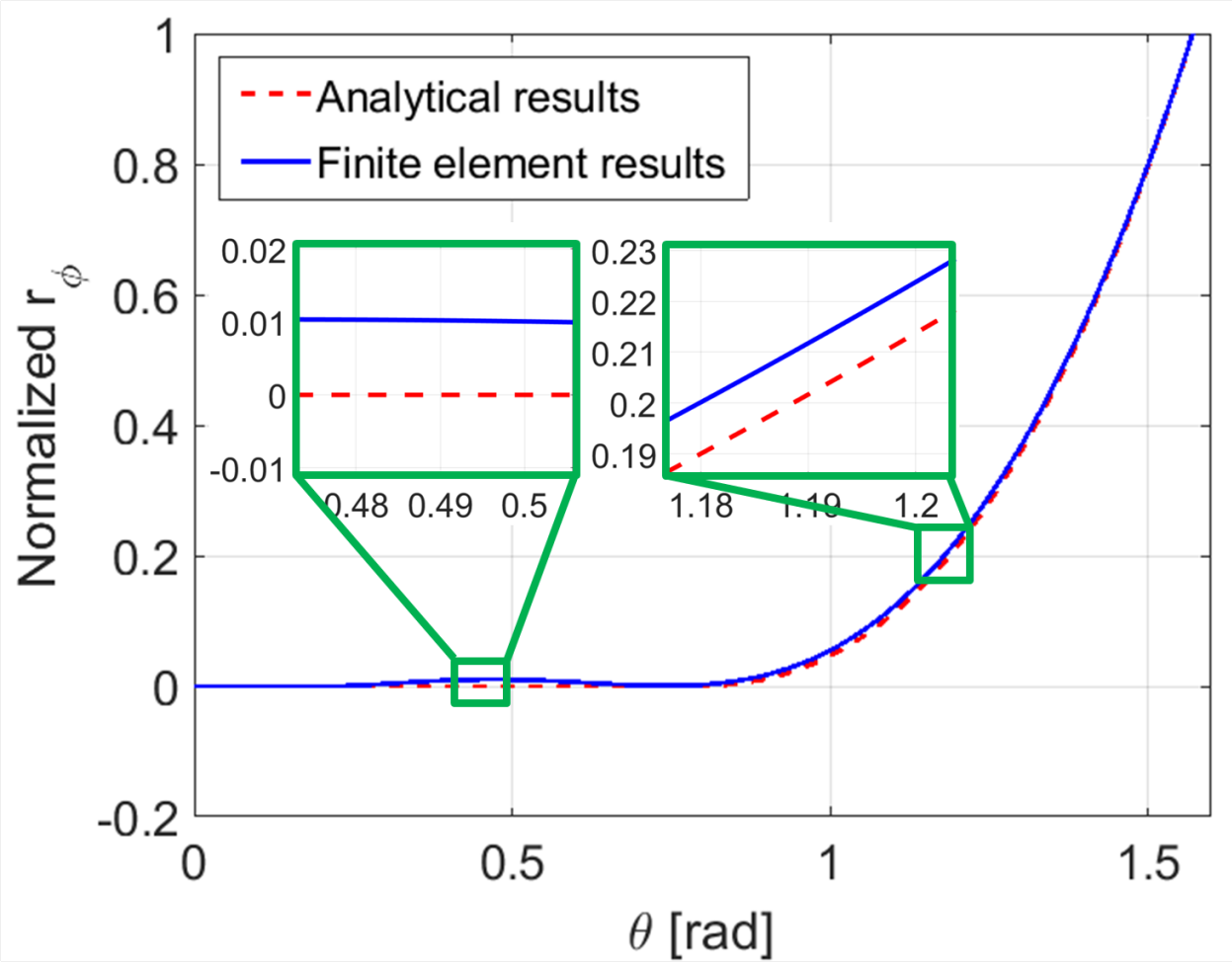

Figure 2.3: Comparison of n=2 mode shapes from analytical model and finite element model. Displacements in ϕ direction were normalized.

2.1.2 Calculation of Resonant Frequency

Resonant frequency of the shell was derived by the Rayleigh's energy method [142]. First, the kinetic energy K_0 and the strain energy U_0 of the shell were calculated based on the

mode shape with an arbitrary amplitude of motion A

$$K_0 = \frac{1}{2}\rho \int_\Omega (\dot{\delta r}^2 + r^2 \dot{\delta\theta}^2 + r^2 \sin^2\theta \dot{\delta\phi}^2) dV, \tag{2.5}$$

$$U_0 = \frac{1}{2} \int_\Omega (\sigma_{11}\epsilon_{11} + \sigma_{22}\epsilon_{22} + \sigma_{12}\epsilon_{12}) dV, \tag{2.6}$$

where Ω is the integration region, which is the shape of the shell. Directions 1 and 2 in (2.6) are the two principal axes of the middle surface, corresponding to θ and ϕ directions. For shells with thicknesses much smaller than the other dimensions, the strain related to the third direction (normal direction) was zero [143]. This was the reason why there were only three terms instead of six in the parenthesis of (2.6). Then, the Lagrangian was expressed as $L = U_{0max} - K_{0max}$, where U_{0max} and K_{0max} are the maximum of the kinetic energy and the strain energy with respect to time, correspondingly. Finally, solving the Rayleigh's equation $\frac{\partial L}{\partial A} = 0$ gave us the angular frequency of the shell.

As shown in Fig. 2.4, the analytical model matched well with the finite element model for the $n = 2$ mode. The error was within 10% for the shell with thickness less than 300μm. Since inextensional assumption could only be applied to thin shells, larger errors for thicker shells were expected. In such cases, the complexity of the model should be increased by eliminating the inextensional assumption and using the full tensor of deformation.

2.1.3 Experimental Verification

Frequencies of micro shells with outer diameter of 7mm and thicknesses of 70μm, 150μm, and 250μm were tested to verify the analytical model. The shells were actuated along the stem by a piezoelectric element attached to the shell by Field's metal, and characterized optically by Laser Doppler Vibrometer (LDV) in a vacuum chamber under pressure on the order of 20μTorr, so that the viscous air damping was completely eliminated [74]. The

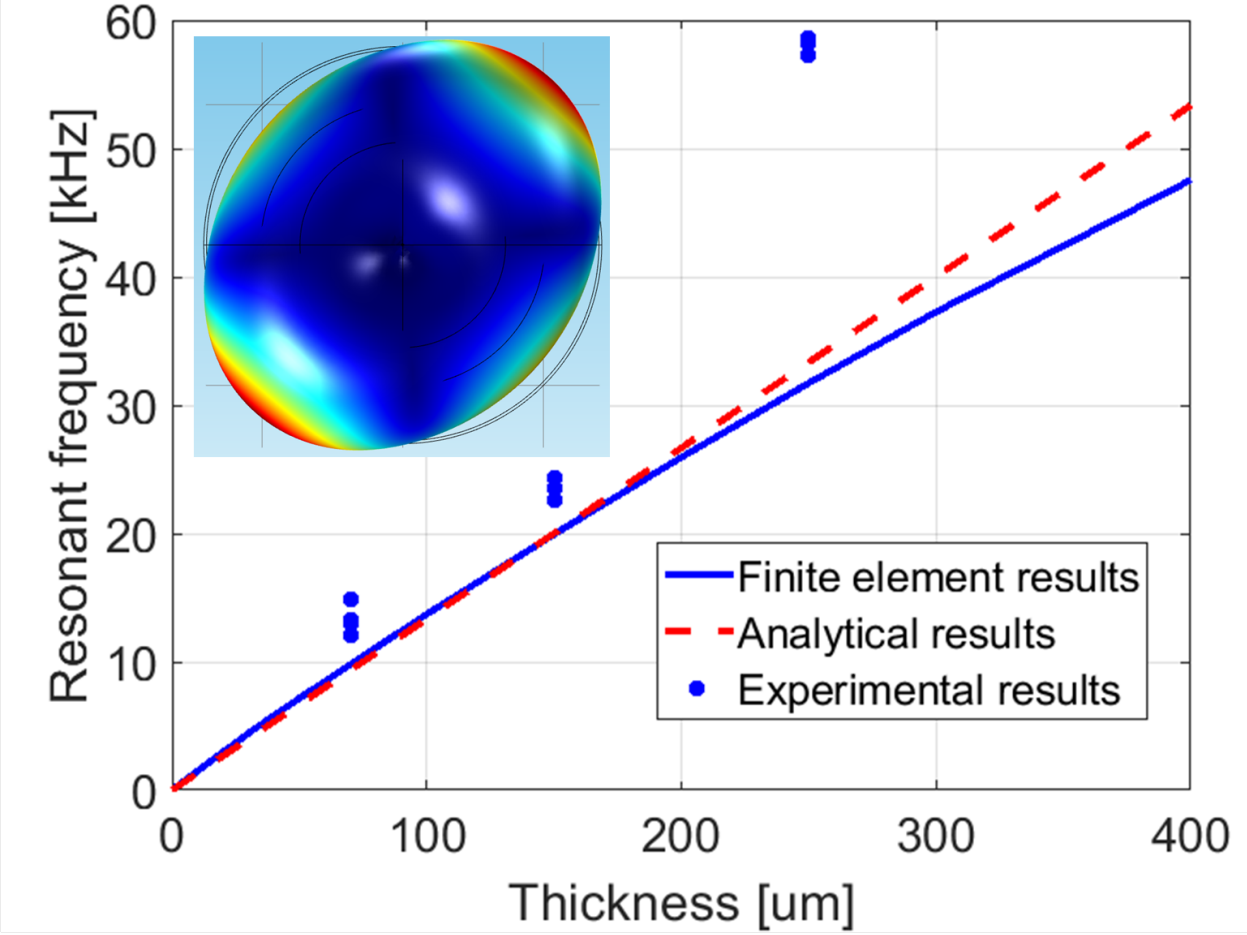

Figure 2.4: Relation between resonant frequency of $n = 2$ mode and the thickness of the shell

experimental setup is shown in Fig. 2.5. The results are presented in Fig. 2.4, showing the maximum errors of about 20%, when the shell thickness was less than 150μm. The errors were possibly due to non-uniformity of the thickness and over-release of the devices. For thicker devices, the errors were relatively larger because the shells were not fully developed during glassblowing and consequently the shape deviated from the hemi-torus.

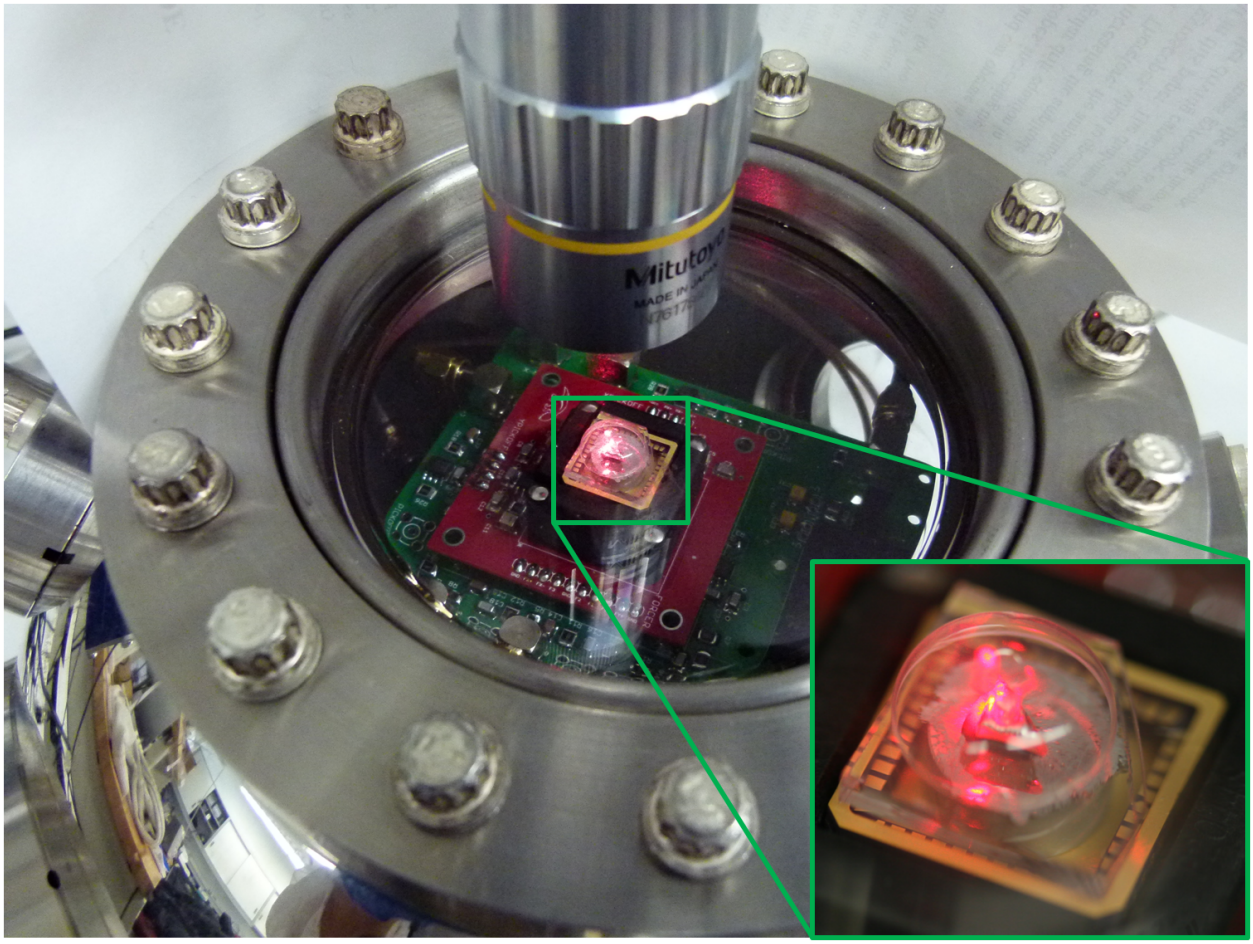

Figure 2.5: Experimental setup to measure the resonant frequency of the device

2.2 Structural Symmetry Analysis

2.2.1 Structural Asymmetry During Glassblowing

Micro-glassblowing technique has been developed as an approach to fabricate 3D fused quartz wineglass resonators (Fig. 2.6) [144]. In this process, a fused quartz wafer is first etched by Hydrogen Fluoride (HF) to create cavities for glassblowing. Then, it is bonded with another fused quartz wafer to seal the cavities. Next, the wafer stack is placed in the high-temperature furnace for glassblowing. Mechanical removal of the substrate is conducted after the blowing in order to release the devices. In this process, atomically smooth and spherical shell structures are experimentally demonstrated with the radial error less than 500ppm [135]. However, many sources of fabrication imperfections may cause structural asymmetry of glassblown wineglass resonators and a change of mode shapes of the resonators, including the temperature non-uniformity in the glassblowing process, the misalignment between the device and the lapping plane during the release, and a thickness variation of the device layer created during the HF wet etching [139]. Among all these factors, the temperature non-uniformity during glassblowing is the most dominant factor. In this section, we present a relation between the temperature non-uniformity and the structural asymmetry of the wineglass resonators.

ANSYS Polyflow was used to simulate the glassblowing process. Generalized Newtonian non-isothermal flow model was applied to simulate the temperature non-uniformity. The temperature-dependent viscosity of Fused Quartz was approximated by Arrhenius exponential model and the parameters were extracted from [146]. In [146], the viscosity of silica was listed in a wide range of temperature from 1200 °C to 2500 °C, matching the temperature during glassblowing (around 1500 °C). In the model, thickness of the device layer before glassblowing was 100μm, the diameter of the cavity was 7mm, and the thickness of the cavity was 400μm (Fig. 2.7). The average glassblowing temperature was set to be 1500 °C.

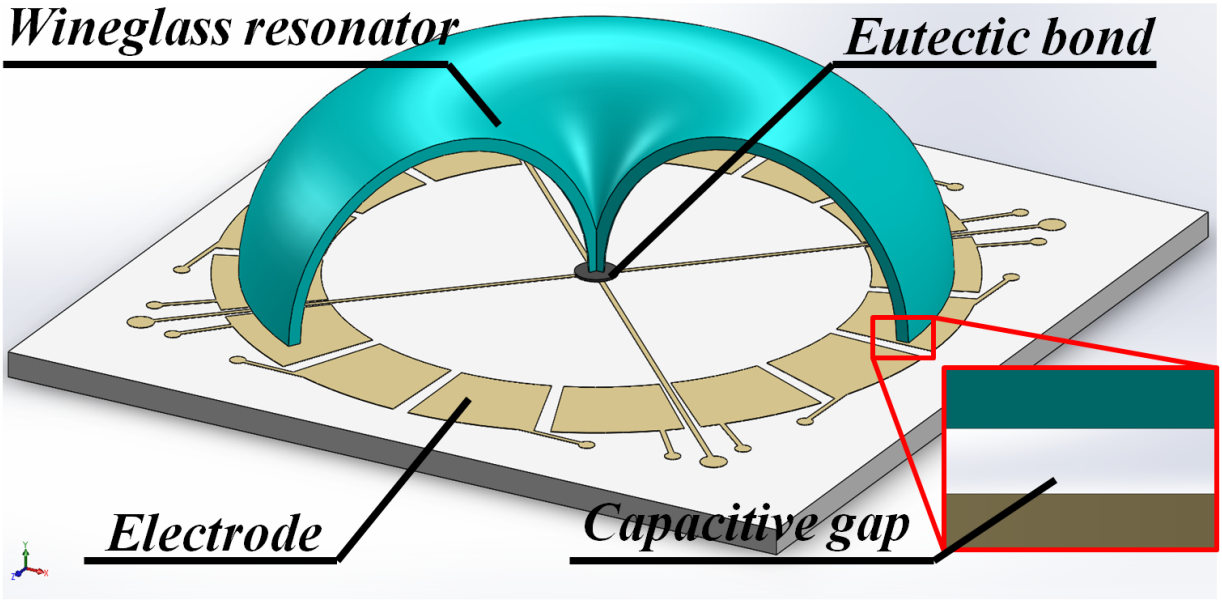

Figure 2.6: Schematics of a micro glassblown wineglass resonator with assembled electrodes [145].

The temperature difference was applied along the x-axis and the gradient was assumed to be constant. To reduce the amount of calculation, the substrate layer of the glass-to-glass stack was not included in the model and only the deformation of the device layer was simulated, since the deformation of the substrate was small compared to that of the device layer. A constant pressure difference of 4bar between the top and bottom of the device layer was applied to model the process of glassblowing. The prediction of glassblowing asymmetry with 20 °C temperature difference between the two sides (+x and -x) of the substrate is shown in Fig. 2.7. The maximum deformation in height on the "+x" side of the blown shell was 4.073mm and the maximum deformation in height on the "-x" side was 3.713mm. The height difference between the two sides was calculated to be 0.36mm. The obtained height difference was due to the viscosity difference caused by temperature gradient through the structure. It was also an indicator of structural asymmetry, which would cause the frequency split of the wineglass resonator. The relation between the temperature difference and the glassblowing height difference is presented in Fig. 2.8. The height difference would be zero and the structure would be axisymmetric if the temperature was uniform. When the

temperature difference between the two sides of the die reached 30 °C, the height difference was around 0.55mm, more than 15% of the net height of the shell structure (excluding the substrate).

2.2.2 Effects of Lapping on Asymmetry

Fabrication imperfections, such as lapping errors, will change the mode shapes of the resonator, as well as induce the frequency mismatch between the two degenerate modes, as shown in Fig. 2.9. On the left side of Fig. 2.9, the blue solid line and the red dashed line represent, in the frequency domain, the responses of two principal directions corresponding to $n = 2$ modes. On the right side of Fig. 2.9, the lines show the distribution of the amplitude of motion for different azimuth angles. Note that the amplitudes at four anti-nodes are not the same due to the structural asymmetry. The orientation of the mode shape that has a lower resonant frequency (blue solid line) is aligned with the direction of asymmetry (vertical direction) and the other mode is separated by 45 degrees. This mode shape corresponds to $n = 2$ mode of the resonator. The information about the mode shape can be utilized as an indicator of the magnitude and orientation of the imperfections.

The effects of lapping errors on the mode shapes of wineglass resonators were demonstrated by FEA, and then verified experimentally. Identical wineglass resonators with different lapping angular errors were modeled by COMSOL MultiPhysics package. In the model, the thickness of the resonators was 100μm and the outer diameter was 7mm. The results are shown in Fig. 2.10. Yellow lines represent the fitted sinusoidal envelops of the mode shapes. When the lapping error β was zero, which corresponded to a perfect device, the amplitudes of motion at anti-nodes were the same and the orientations of the mode shapes were random due to axisymmetric nature of the device. As β increased, the mode shapes were aligned with the direction of lapping imperfection, which was the y-axis in this model, and they deviated

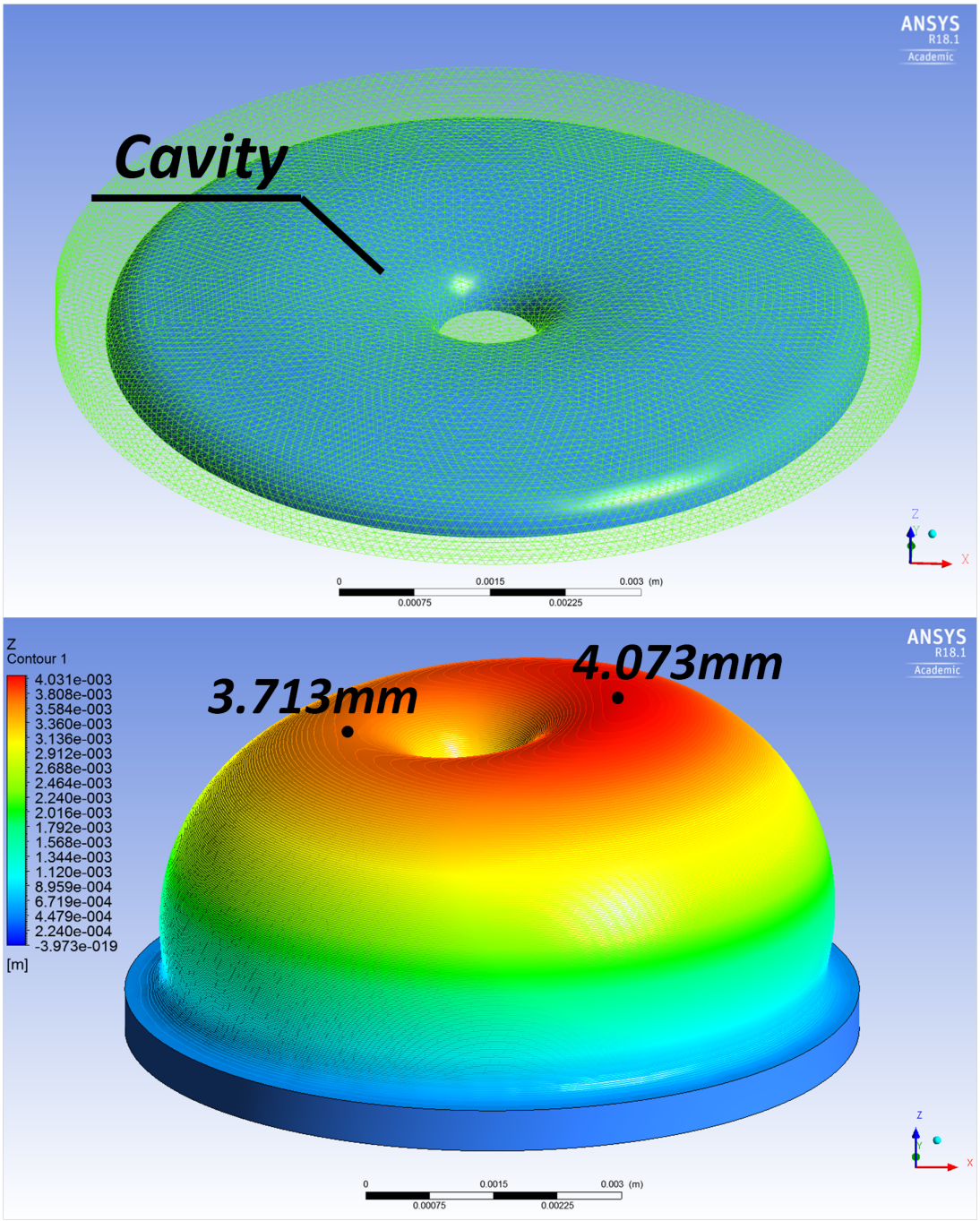

Figure 2.7: Shape of the model (a) before blowing, and (b) after blowing. Cavity was indicated by the solid blue part. The modeling showed a source of possible imperfections in glassblown structure, which needed to be compensated. Temperature non-uniformity in furnace during the glassblowing process resulted in asymmetry of the structure. The temperature difference between the two sides of the substrate was set to be 20 °C. The maximum deformations in height on the "+x" and "-x" side of the blown shell were 4.073mm and 3.713mm, respectively.

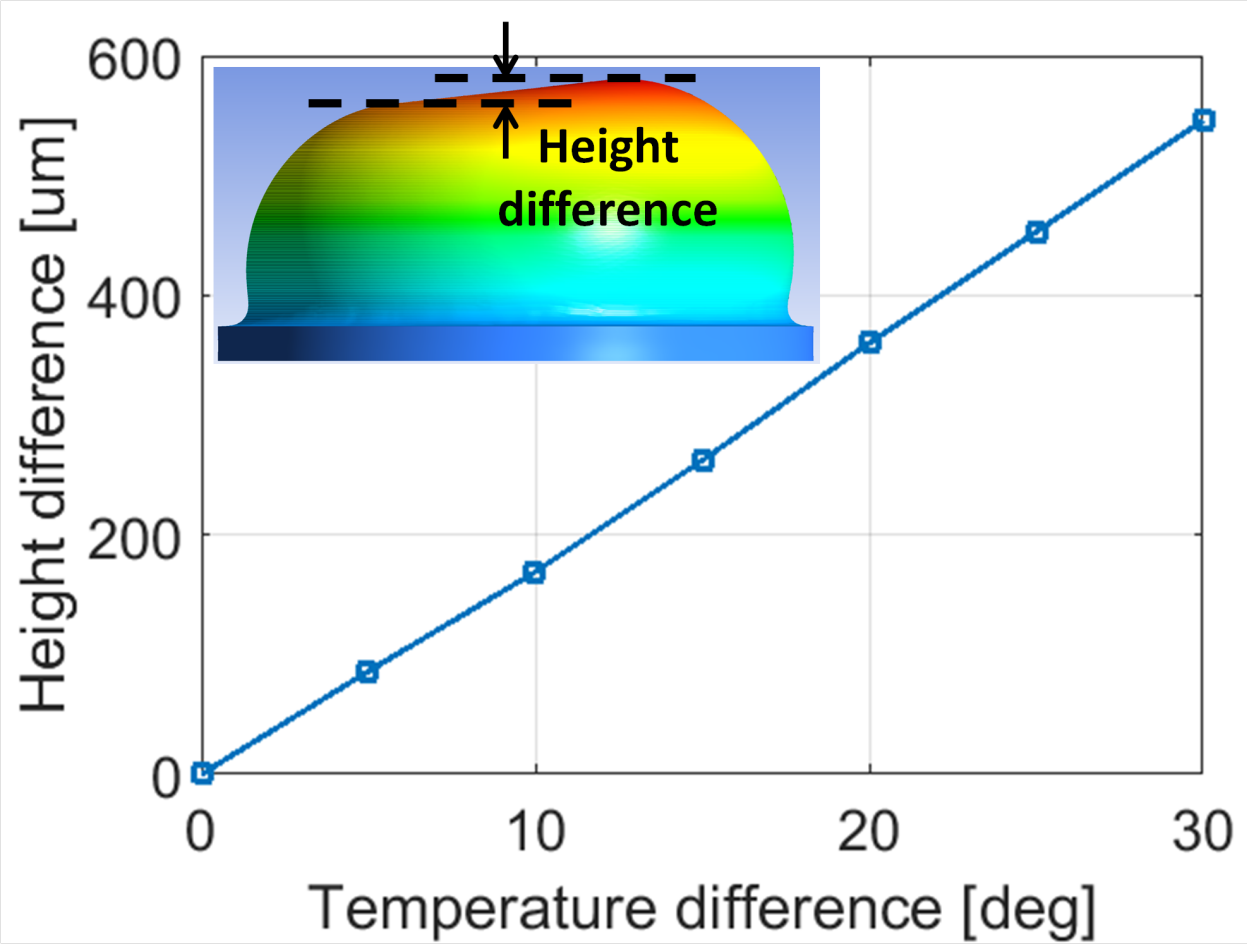

Figure 2.8: Relation between the temperature difference and the height difference. The inserted plot is the side view of the glassblown structure.

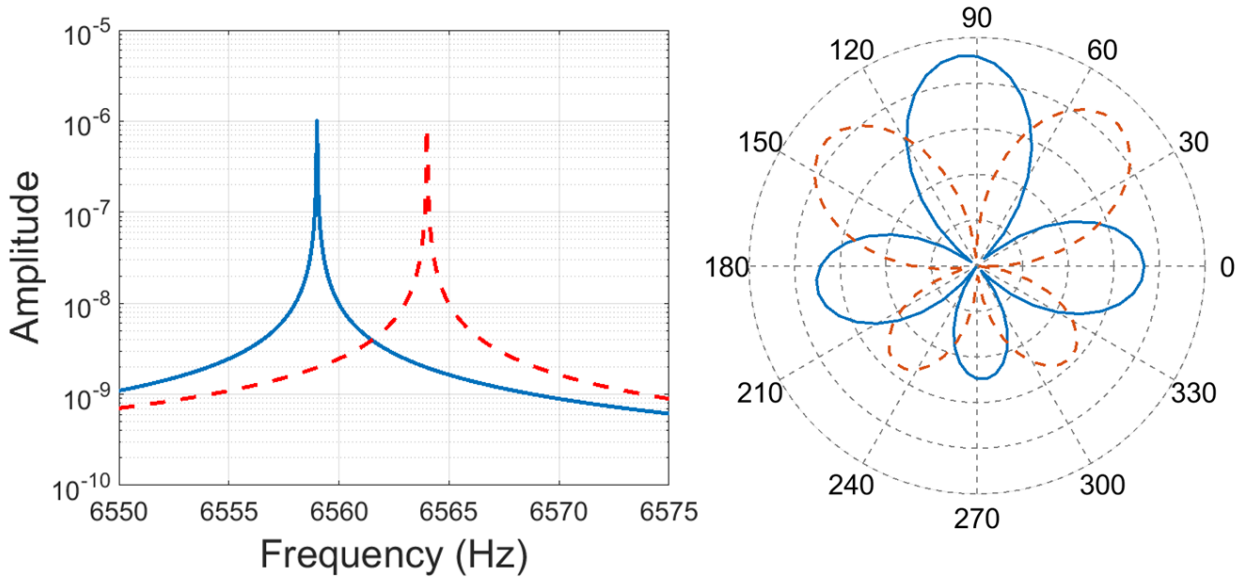

Figure 2.9: Effects of fabrication imperfections on the resonant frequency and mode shape ($n = 2$ in this case) of the wineglass resonator.

from the case in which the amplitudes at anti-nodes were the same. It is also noticed that the azimuth amplitude of the mode shapes still followed the sinusoidal distribution. The ratio between the largest amplitude and the smallest amplitude of the sinusoidal envelop increased as β increased. The relation between the lapping angle and the ratio of amplitudes is presented in Fig. 2.11, which shows an exponential relation between the angular lapping error β and the amplitude ratio. This relation not only helped to evaluate the deviated mode shape of the structure in the frequency split model, but also allowed to estimate the lapping angle needed in our algorithm to compensate for the structural asymmetry of a wineglass resonator.

2.2.3 Directional Lapping Analysis

The idea of directional lapping is to adjust structural asymmetry of the as-fabricated asymmetric structure by lapping the rim of the shell to compensate for the structural asymmetry. In this section, a model is derived to fully understand the effects of directional lapping on

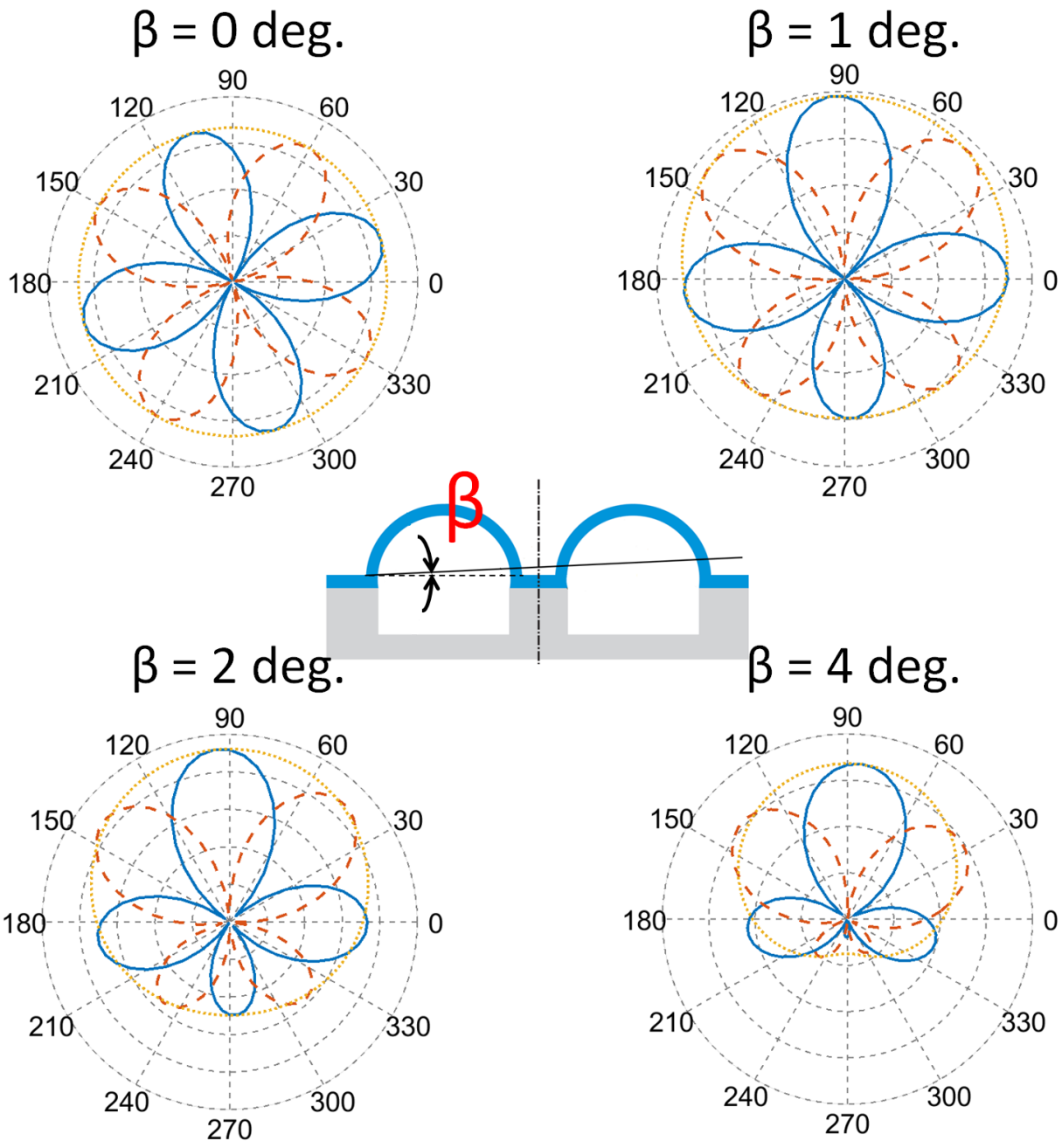

Figure 2.10: The effects of lapping angle on the mode shape of the wineglass resonator. Solid blue lines and dashed red lines are the azimuth amplitude distribution of the mode shapes. Yellow dotted lines are the sinusoidal envelops of the mode shapes.

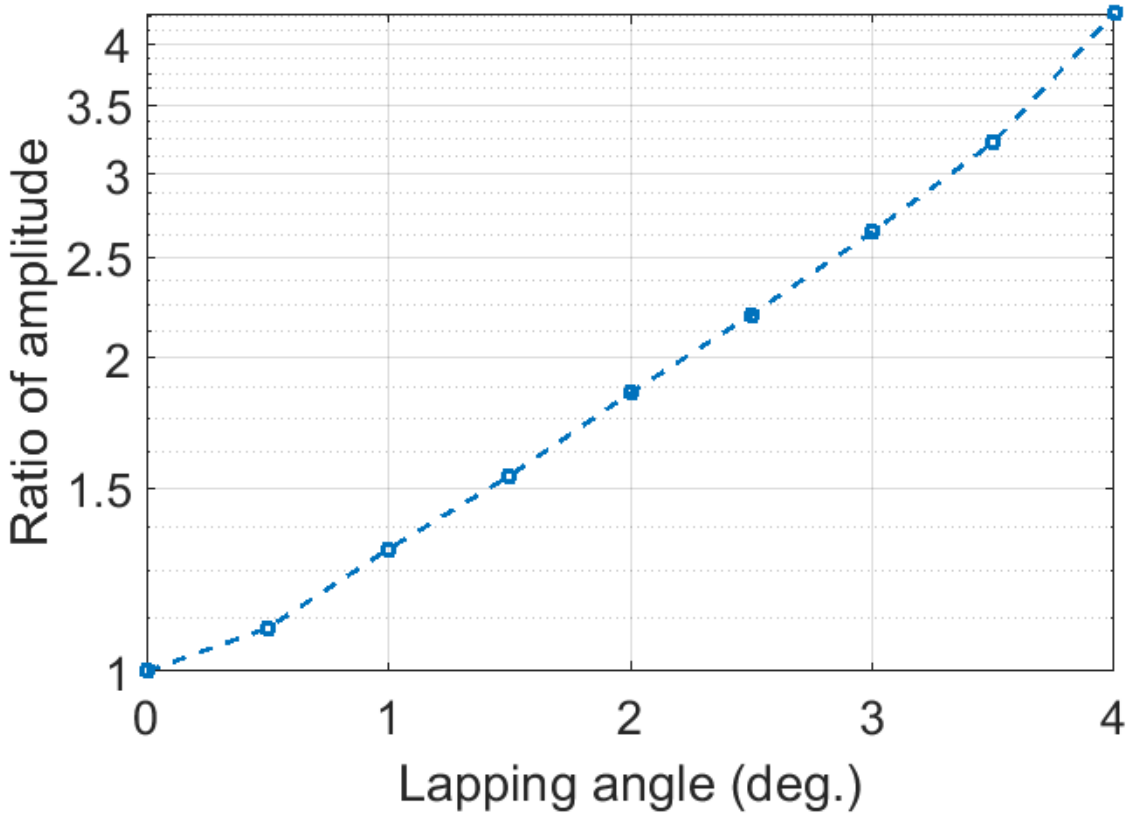

Figure 2.11: Exponential relation between the lapping angle and the ratio between the largest amplitude and the smallest amplitude of the sinusoidal envelop of mode shapes for the wineglass resonator.

the structural asymmetry.

Mode shape of an ideal hemi-toroidal wineglass resonator has been analytically derived in Section 2.1. For asymmetric shells, deviations of mode shapes from the ideal case exist due to lapping errors [145]. Therefore, to numerically estimate the frequency split due to lapping errors, not only the region of integration need to be adjusted, but also the deviations of the mode shapes need to be considered in the model. During the release step, the lapping will not stop precisely in the desirable location when the structure is released. This is called overlapping of the shell and will influence the frequency of the devices. In our analysis, overlapping and the thickness variation of the shell along radial direction were also considered:

1. Mode shape deviation is the most dominant factor in the frequency split analysis. It was assumed that the amplitude distribution along the radial direction remained the same as in an ideal shell, while the amplitude distribution along the tangential direction was sinusoidal with respect to the azimuth angle. The ratio between the largest amplitude and the smallest amplitude of the mode shape depended on the lapping error and followed the relation presented in Fig. 2.11.

2. Thickness variation of the structure is inevitable in the glassblowing process since the stretching of the material is much greater at the top of the shell than around the rim. Therefore, the thickness at the rim of the structure is much larger than at the top. As a result, the lapping analysis will yield smaller effects if a uniform thickness is assumed. To develop a more accurate model, two devices were vertically lapped and their thickness distributions were measured. The experimental results are shown in Fig. 2.12. The horizontal axis is the θ angle, and the vertical axis is the normalized structural thickness with respect to the thickness at the top of the shell. Thickness distributions of the two devices matched each other well. The thickness distribution showed a thinner structure at the top and a thicker structure at the stem and rim of the shell. Only the outer part, corresponding to $\theta > 90^\circ$, of the wineglass thickness

was estimated, since the amplitude of motion was close to zero for the inner part, [147]. We applied the following function to fit the curve

$$t = t_0 * exp((\theta - 90)^3/1.2e6), \tag{2.7}$$

where t_0 is the thickness of the wineglass at the top.

3. Overlapping of the structure is another factor that affects the frequency split prediction. A moderate overlapping is necessary to make sure the rim of the wineglass is still in a plane. Therefore, 50μm of overlapping was assumed for each lapping process in the model.

Frequencies of two $n = 2$ modes were calculated based on the Rayleigh's energy method, as discussed previously. In this model, the structural thickness at the top of the wineglass was assumed to be 70μm and the outer diameter was 7mm, matching the experimentally tested devices. The results correlating the angular lapping error and the frequency split are shown in Fig. 2.13. A good match was achieved between the model and experimental results. Around 50Hz frequency split was induced by a lapping error of 3°, i.e. 3° of directional lapping would have the capability of compensating the frequency split on the order of 50Hz, since the directional lapping procedure could be considered as a reverse process of lapping error. Fig. 2.13 also shows that the frequency split of the wineglass resonator is less sensitive to lapping errors, when the errors are small. This indicated that the directional lapping process would become more robust to errors, such as misalignment, as the frequency split was reduced. Less than 10Hz of frequency split could be easily achieved if the error of lapping angle was controlled within 1°. This would greatly facilitate the subsequent electrostatic fine tuning of the wineglass resonators.

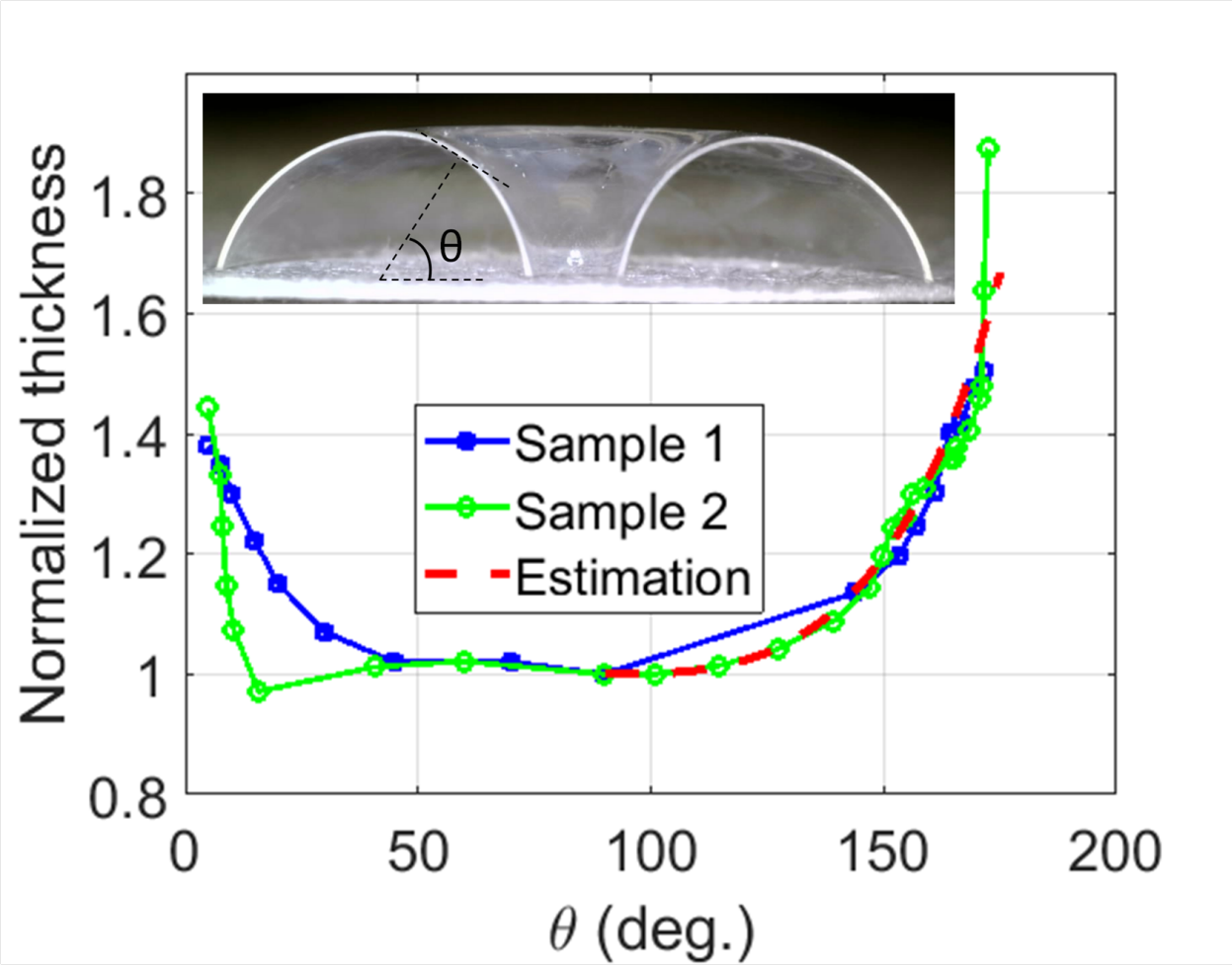

Figure 2.12: The cross section of the device and the thickness distribution of two samples. Structural thickness was normalized with respect to the thickness of the wineglass at the top.

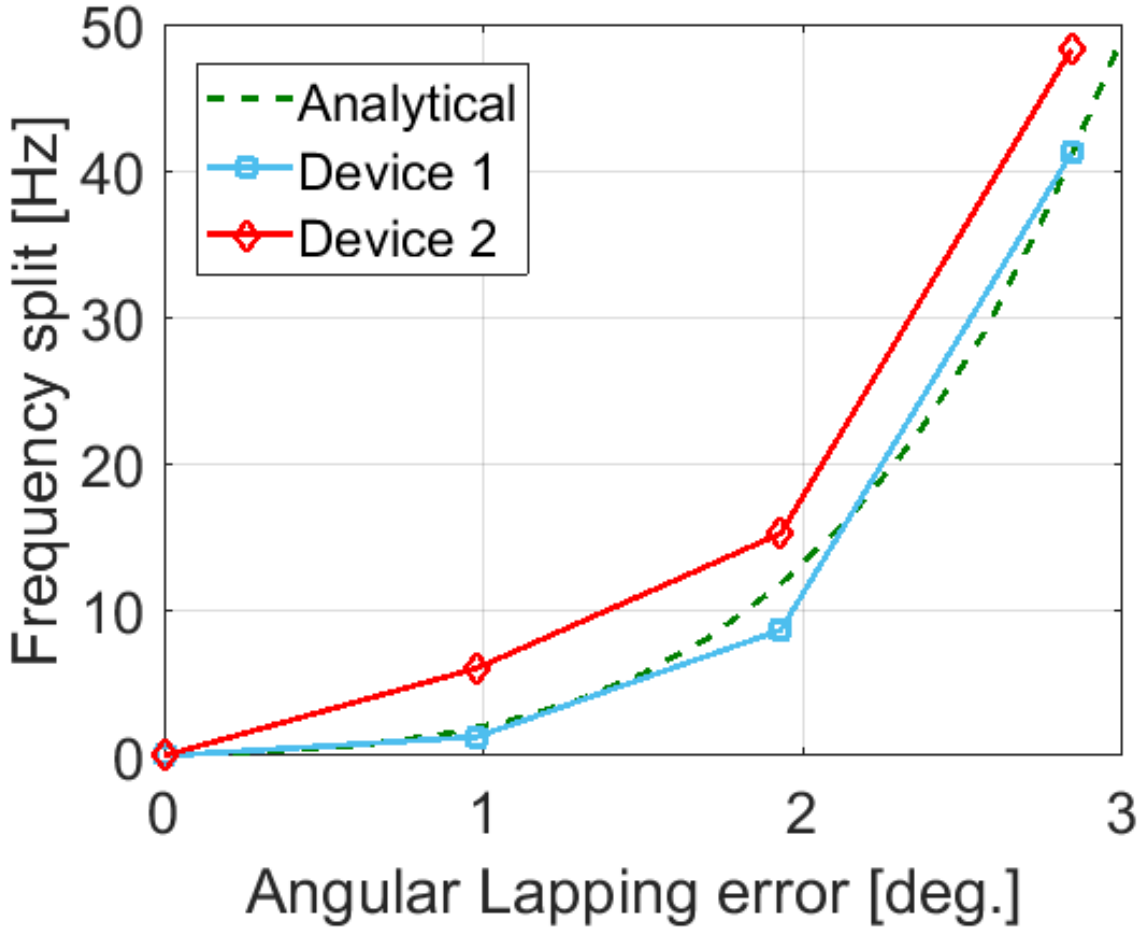

Figure 2.13: Relation between the angular lapping error and the frequency split of the wine-glass resonator. A good match between the model and experimental results was achieved.

2.3 Directional Lapping and Frequency Split

There were two steps in the directional lapping to reduce the frequency split of a micro wine-glass resonator. The first step was to determine the orientation of asymmetry by measuring the mode shapes of the resonator. (In this study we focused on $n = 2$ mode, but higher order modes could be considered as well). In the second step, the permanent tuning was performed by directional lapping based on the information about asymmetry of the structure.

2.3.1 Asymmetry Identification

If we take the amplitudes of two degenerate modes as variables, 3D micro glassblown resonator vibrating in $n = 2$ wineglass modes can be modeled as a 2 degree-of-freedom system, although it is a continuous structure instead of a lumped mass-spring system. In the mode of oscillation, two Principal Axes of Elasticity (PAE) can be identified, where the frequency of vibration will be the lowest if orientation of the mode shape is aligned with one of the PAE, and the frequency will be the highest if the mode shape is aligned with the second PAE. Since $n = 2$ wineglass mode is the most commonly used mode in 3D axisymmetric resonator gyroscopes, we only considered the PAE of $n = 2$ mode in this study.

An experimental setup was developed to determine the orientation of PAE, and it is shown in Fig. 2.14. The wineglass resonator was temporarily attached to a piezoelectric stack by Field's metal, allowing to actuate the device during the procedure of parameter identification. Then, the device with the piezoelectric stack was attached to a rotary stage, which was controlled by a servo-motor. The device was excited by the piezoelectric stack along the stem of the wineglass resonator and its response to actuation was measured by Laser Doppler Vibrometer (LDV) pointed to the outer edge of the wineglass resonator. The rotary stage was rotated by 10° after each measurement, so that the amplitude at different azimuth angles was measured to obtain the full information about the mode shape. This step of the experiment was conducted in air since the quality factor was not a critical parameter in this step. The quality factor of the resonator in the air was on the order of hundreds, corresponding to a damping ratio of smaller than 0.01. Therefore, the effects of the possible asymmetric damping on the shift of modes could be neglected.

The in-plane amplitude of motion along the outer edge of the device is shown in Fig. 2.15. Red dots are the experimental results and the blue dashed line is the fitted curve. A principal axis of elasticity for $n = 2$ mode was identified according to the mode shape and was shown

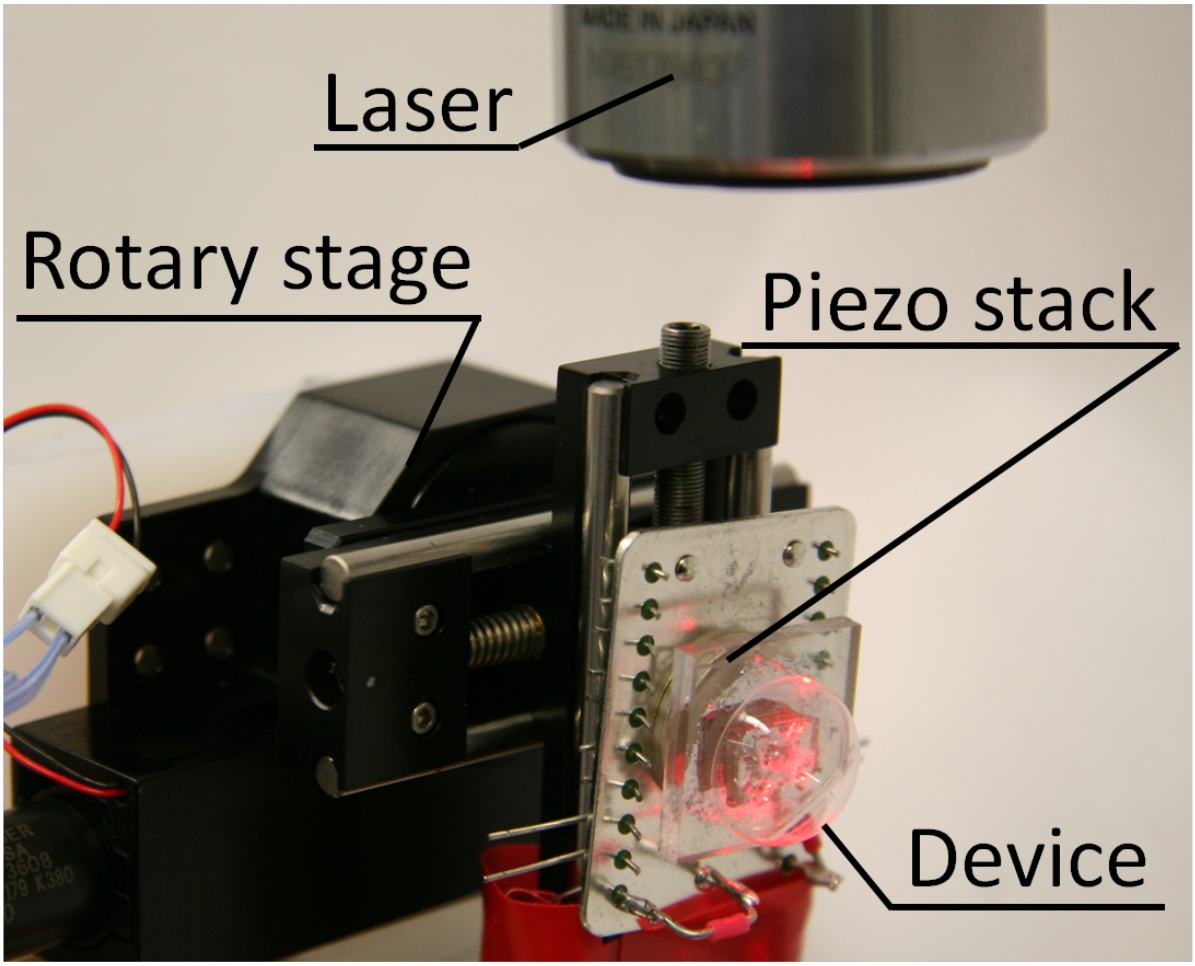

Figure 2.14: Experimental setup to determine the orientation of PAE of the wineglass resonator.

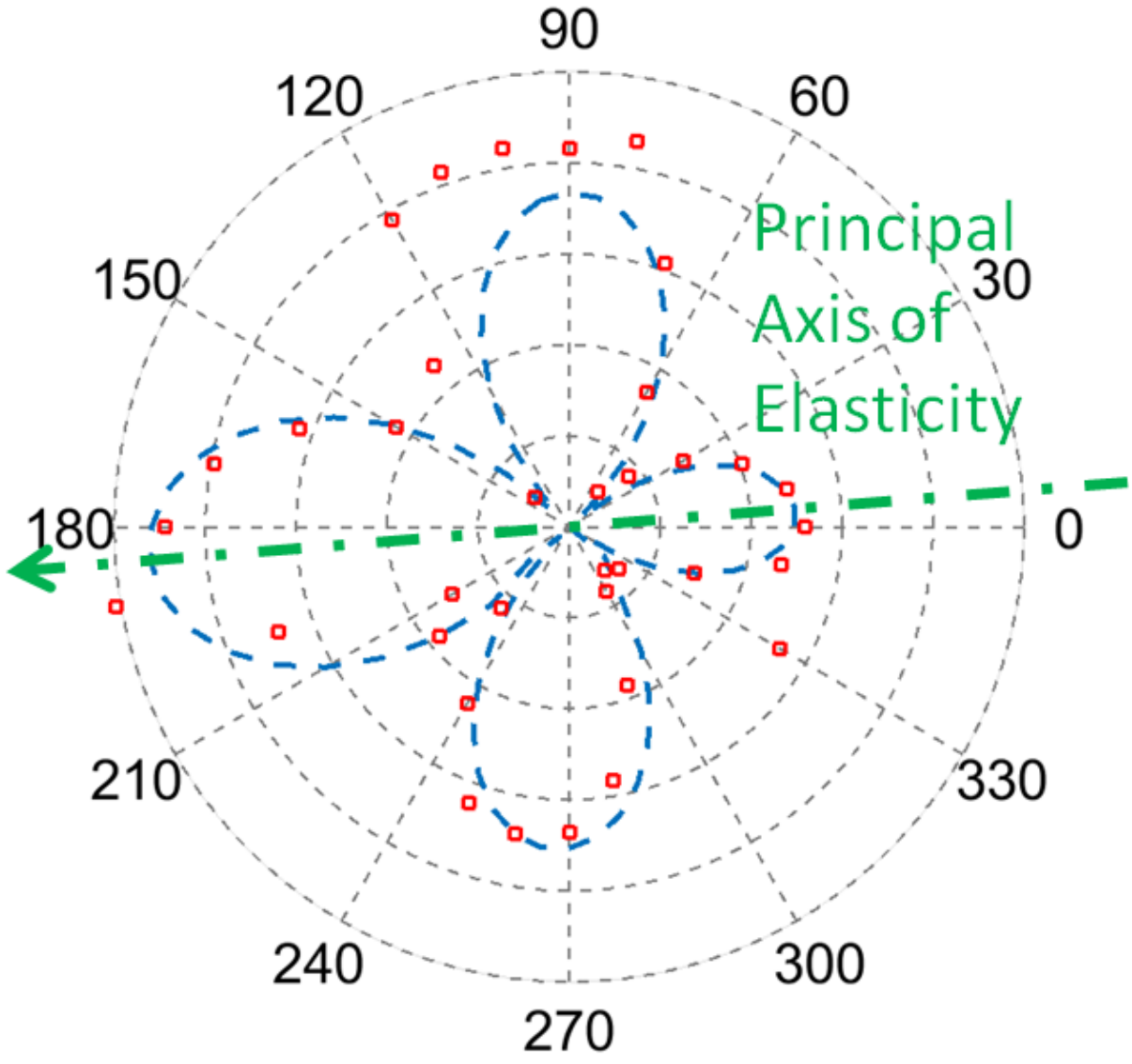

Figure 2.15: Experimental result of the mode shape of the wineglass resonator and the identification of PAE.

by a green dashed arrow. The arrow also shows the orientation of the structural asymmetry, and therefore, the anticipated directional lapping should be aligned with this direction to compensate for the structural asymmetry.

2.3.2 Asymmetry Reduction

In this section, we demonstrated that the directional lapping could be utilized for reduction of asymmetry in wineglass resonators.

Special lapping fixtures were designed and 3D printed to perform the directional lapping. These fixtures were designed so that lapping angle of 1° could be introduced by each iteration of lapping. The angular misalignment with our current setup was estimated to be less than 10° and possible error sources included LDV measurements of the amplitude, fabrication of lapping fixtures, and attachment of the device to the lapping fixture. During the directional lapping, one side of the wineglass resonator would get contacted to the lapping film and be lapped first. To be able to apply the out-of-plane electrode architecture (illustrated in Figure 2.6), a moderate overlapping was necessary to make sure the rim of the wineglass was still in a plane, even though it might affect the efficiency of the whole process. A solid model of the fixture is shown in Fig. 2.16. The cavity in the middle of the fixture was designed to hold the device and four height references were designed to precisely control the lapping angle. The vertical resolution of the 3D printer was within 30μm, corresponding to a lapping angle of about 0.1°, and thus it guaranteed the control of the lapping angle with the step accuracy of 0.1°. The directional lapping procedure was exactly the same as the shell release procedure, Fig. 2.16, except that it utilized the designed fixture for directional lapping. The described procedure did not reduce the quality factor of the device after the compensation since the same level of surface roughness could be achieved as in the release process (less than 1nm Sa).

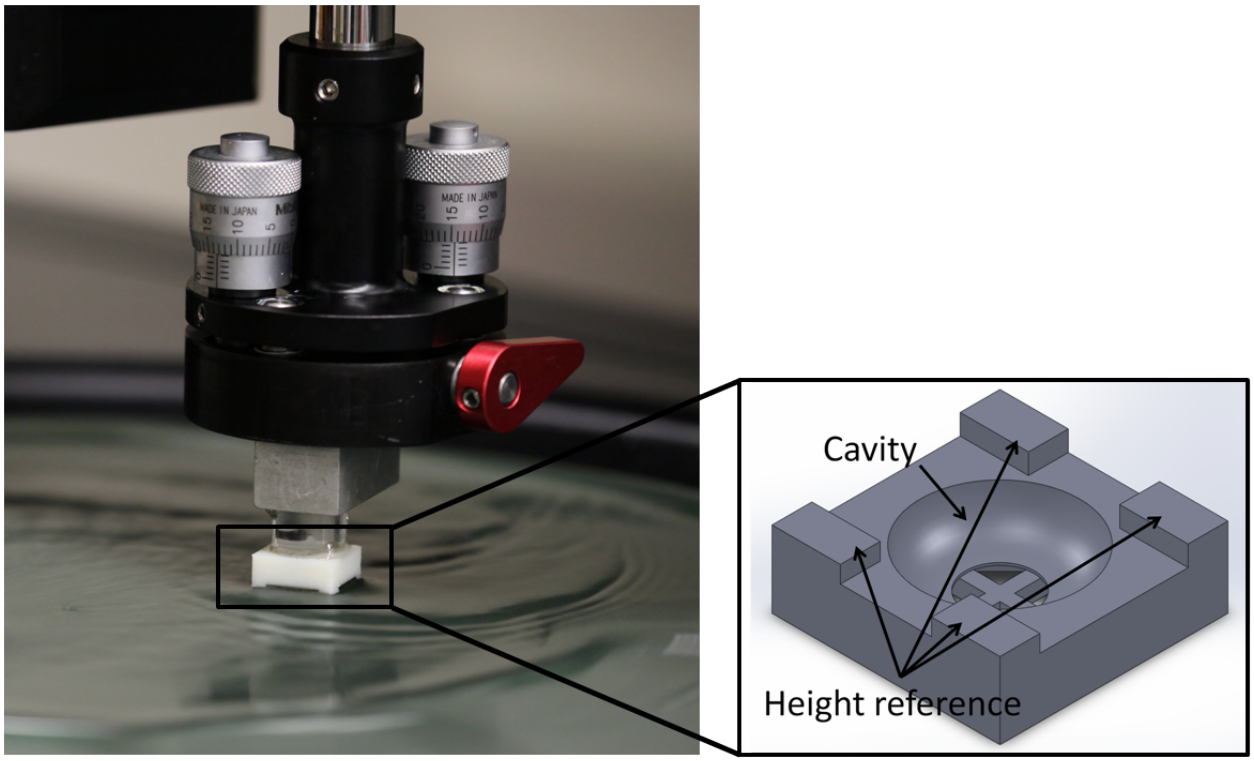

Figure 2.16: Experimental setup to conduct the directional lapping of the wineglass resonator and a SolidWorks model of the lapping fixture.

After the directional lapping, the devices were cleaned by solvent and RCA-1, and characterized in the vacuum chamber by LDV to show the reduction of frequency split. Then, the identification of PAE was conducted again and another directional lapping was applied to the device along the updated direction of PAE to further reduce the frequency split of the device, until the frequency split reached its minimum. The goal of the subsequent identification of PAE was to compensate for any possible errors introduced by the previous directional lapping process which might arise, for example, due to the misalignment between the direction of asymmetry and the direction of lapping, thus made the procedure less sensitive to these possible errors than directly measuring and lapping the device just once

2.3.3 Directional Lapping Results

Three devices with initial frequency splits of around 50Hz were used to illustrate results of this study. Fig. 2.17 shows initial results of directional lapping, illustrating the same trend of frequency split reduction due to lapping along the PAE, and thus revealing feasibility of the method. Note that the frequency split of device #1 was increased after the third directional lapping, indicating that the frequency split of the device reached minimum when the lapping angle was between 2° and 3°. Our best result was achieved on the device #1, as shown in Fig. 2.17. A near 6 times of frequency split reduction was demonstrated after 2° of directional lapping by reducing the frequency split from 41Hz to 7Hz, as presented in Fig. 2.18. The resonant frequency of the device was also reduced from 5.2kHz to 4.6kHz during the process.

Fig. 2.15 shows the amplitude of motion at the outer edge of the device #1 before compensation. The ratio between the largest amplitude of anti-node of the fitted curve (around 4.5) and the smallest amplitude of anti-node (around 2.5) was 1.8. According to Fig. 2.11, the ratio of 1.8 corresponded to the lapping angle of about 2°. This agreed with the experimental result that the frequency split was reduced to the minimum for lapping angle between 2° and 3°. The close match between the prediction and the experimental result verified the relation between the ratio of amplitudes and the lapping angle, which was derived from the finite element analysis.

Another advantage of the directional lapping method was that this method was based on the mechanical lapping process, and as a result, it was compatible with most materials and no special design of the structure was needed to apply the compensation. The directional lapping method could be applied not only to Fused Quartz wineglass resonators, but also to almost all kinds of micro-scale 3D structures.

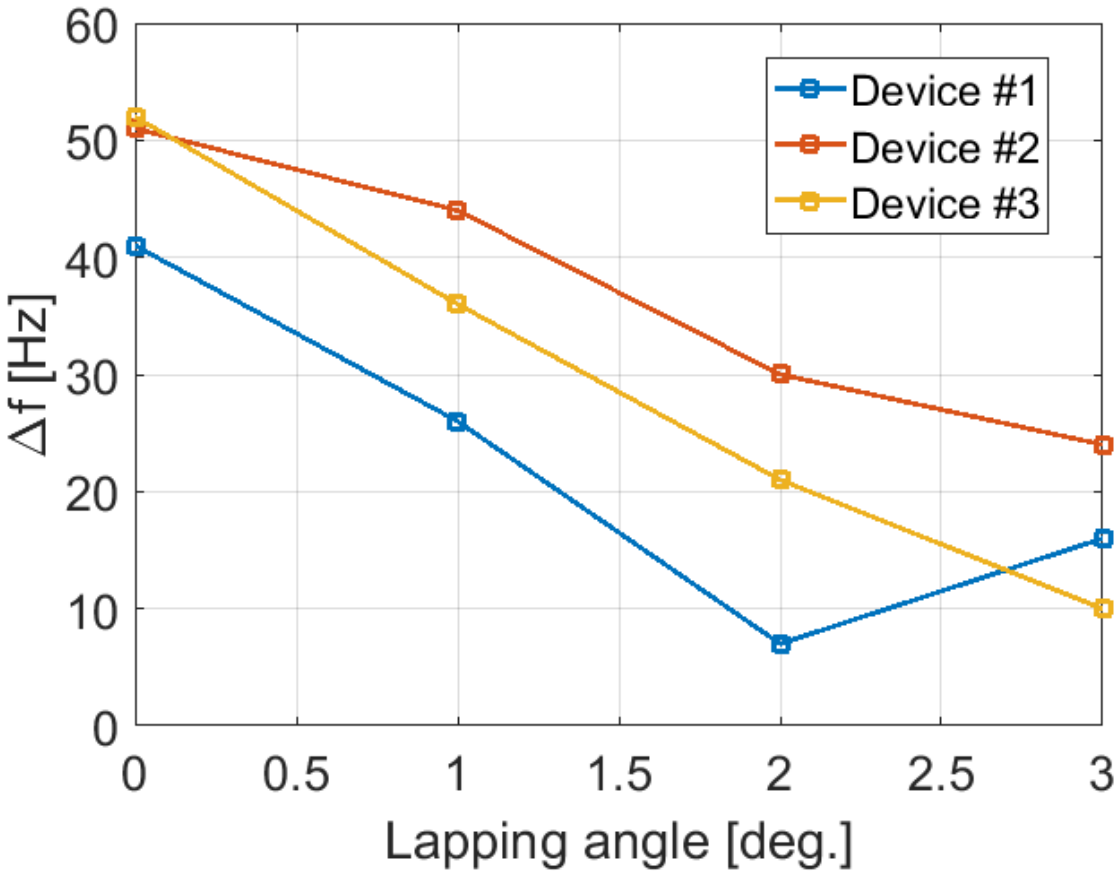

Figure 2.17: Experimental results demonstrating the directional lapping on three samples of wineglass resonators. The same trend of frequency split reduction was observed, confirming feasibility of the method.

2.4 Directional Lapping and Quality Factor

The directional lapping procedure was based on the mechanical lapping procedure. Therefore, no additional energy loss mechanism, such as surface losses, would be introduced due to the directional lapping procedure. A higher structural symmetry after compensation would lead to a lower anchor loss, and therefore, the directional lapping process was hypothesized not to reduce the overall quality factor of the wineglass resonators. The effects of TED were also analyzed in this section.

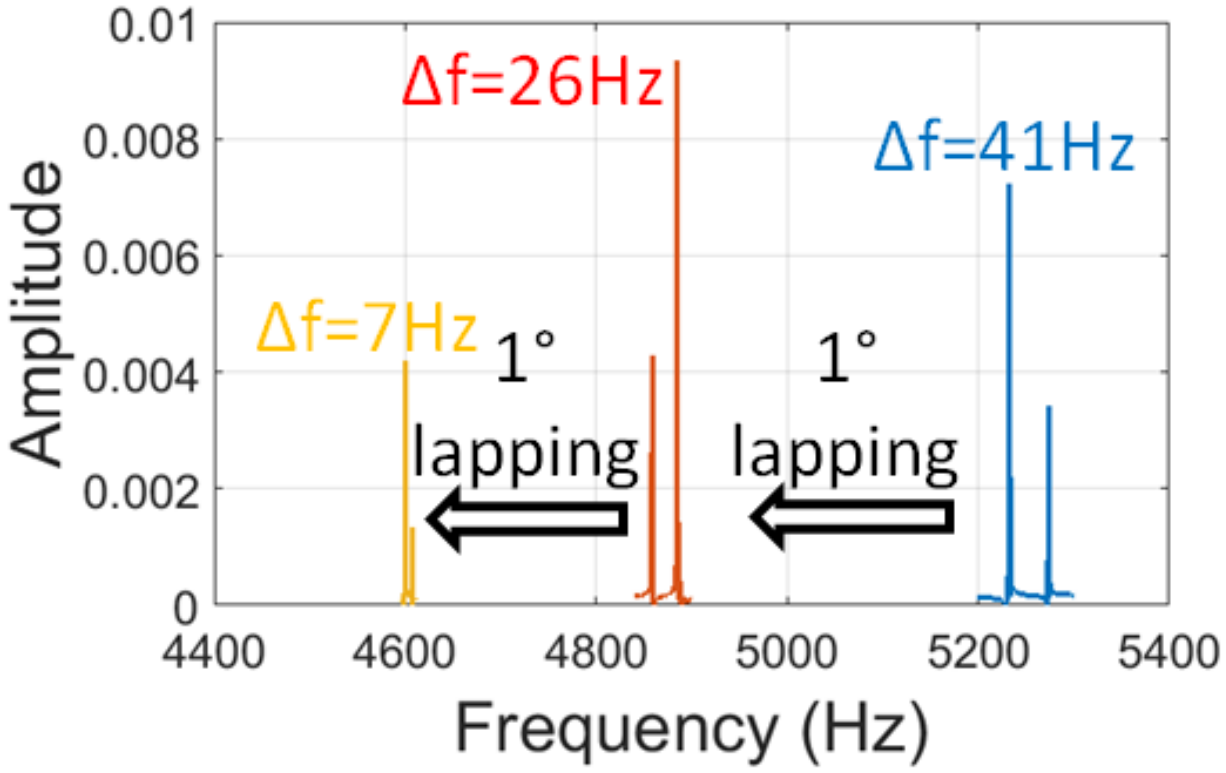

Figure 2.18: A near 6 times of frequency split reduction has been demonstrated after 2 degrees of directional lapping, reducing the frequency split from 41Hz to 7Hz.

2.4.1 Thermo-Elastic Damping Analysis

It is believed that TED is one of the dominant factors that limit the overall quality factor of a Fused Quartz wineglass resonator [74].

In this study, we analyzed the effects of directional lapping on TED by COMSOL MultiPhysics package. Thermoelastic damping of wineglass resonators with different lapping angles was simulated at room temperature (300K). Dimensions of the resonators were the same as described in previous section. The results were shown in Fig. 2.19. The quality factor dropped from 41.3 million to 33.4 million after an lapping error of 5° was introduced. Since directional lapping was a procedure of reducing the lapping errors, the quality factor of wineglass resonator was expected to increase after the directional lapping procedure.

2.4.2 Anchor Loss Analysis

In this study, we applied the PML approach to study the effects of directional lapping on the anchor loss of the wineglass resonators by COMSOL MultiPhysics package. Hemispherical PML with a device at its center is highly effective in absorbing acoustic waves in the radial direction [89], and therefore it was chosen for this study. The working frequencies of the resonators were between 5kHz and 10kHz. The wavelength of the acoustic wave generated by the vibration of the resonator in the glass substrate was less than 1m. Therefore, we set the radius of PML to be 1m. Dimensions of the devices were the same as described in previous section. Details of the model were presented in Fig. 2.20. The size of the PML was much larger than the actual size of the substrate due to the relatively low resonant frequency of the device. It did not contradict the real case since the function of PML was just to fully absorb the vibration and it did not correspond to any physical object in the experimental setup.

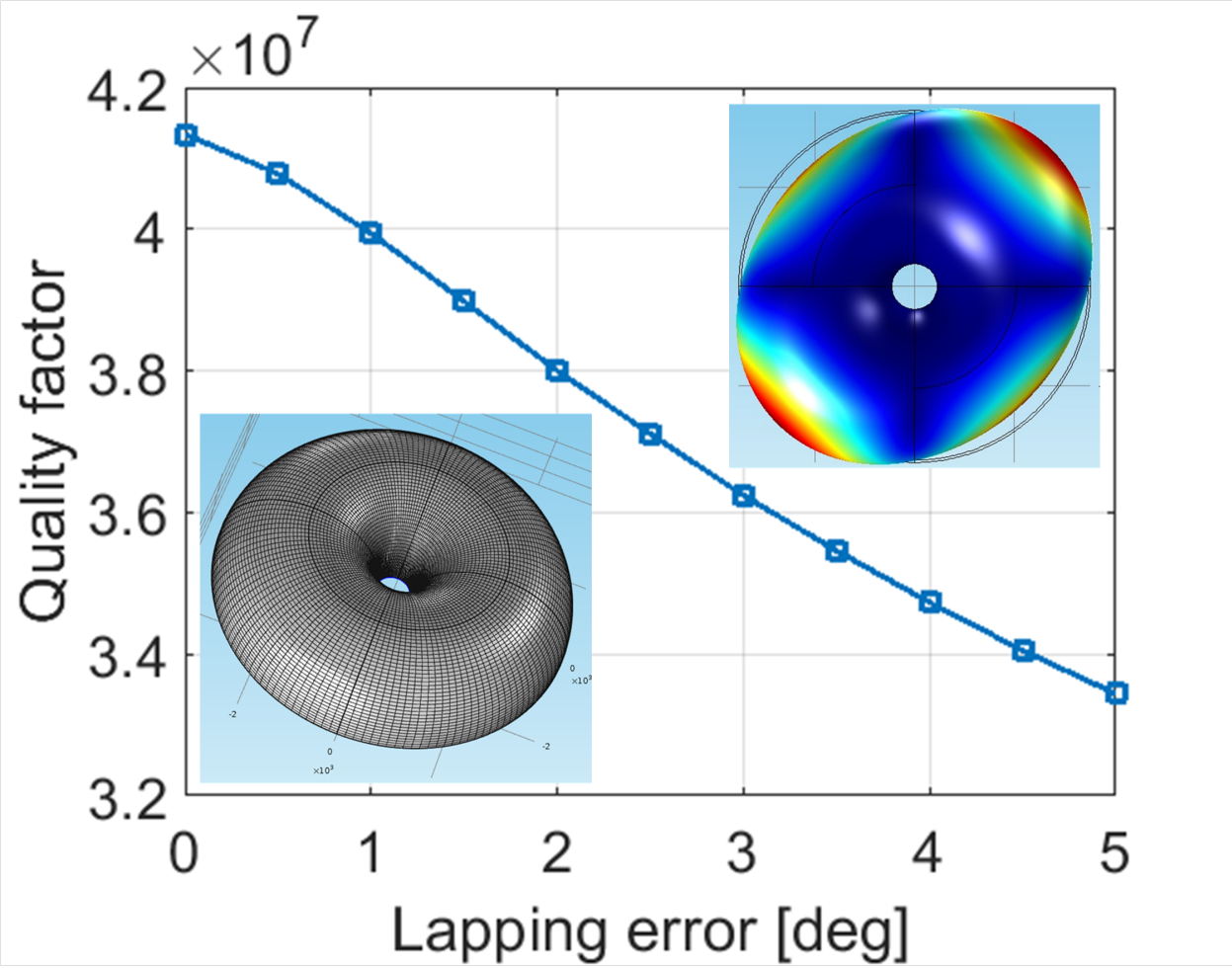

Figure 2.19: Relation between the lapping angle and TED limit of the resonator. Meshing and the mode shape of the wineglass resonator were also presented. The quality factor dropped from 41.3 million to 33.4 million with a lapping error of 5°.

Meshing quality was critical in this study since the dimensions of the device (70μm in thickness) were 3 to 4 orders of magnitude smaller than those of the PML (1m in radius). The shell geometry was discretized with tetrahedron meshing elements of the size on the order of 50μm. Meshing distribution inside PML, however, changed from very small elements in the region close to the shell geometry (on the order of 100μm) to very large elements at the boundary (around 0.2m). The ratio of the size of the largest element to the smallest element was 2000 in this model.

Anchor loss of an ideal axisymmetric wineglass resonator was first calculated. The quality factor due to the anchor loss was 19.6 billion and it was referred as Q_0 in this section. Mesh elements, varying from 7.59×10^4 to 1.37×10^5, confirmed the convergence of the model with a tolerance of 5%, Fig. 2.21. Then, angular lapping errors were introduced in the shell structure and the quality factor was calculated again. The relation between the quality factor and lapping error is shown in Fig. 2.22. A 400 times of reduction in the quality factor due to the anchor loss was numerically demonstrated with a lapping error of 5°. Since directional lapping procedure resulted in the reduction of the lapping error, the increase in quality factor due to anchor loss was expected. However, due to the balanced nature of the structure, the quality factor limit due to the anchor loss was much higher than the TED limit of the device. Therefore, the increase of the overall quality factor due to lapping was not expected to be observed experimentally.

2.4.3 Surface Finish Analysis

Surface loss is another important factor that affects the quality factor of the wineglass resonator. The surface roughness has been demonstrated to affect the surface loss [148]. To eliminate the effect from our experiments, we kept the surface roughness of rim of the devices the same after the compensation process, in order to keep the quality factor constant.

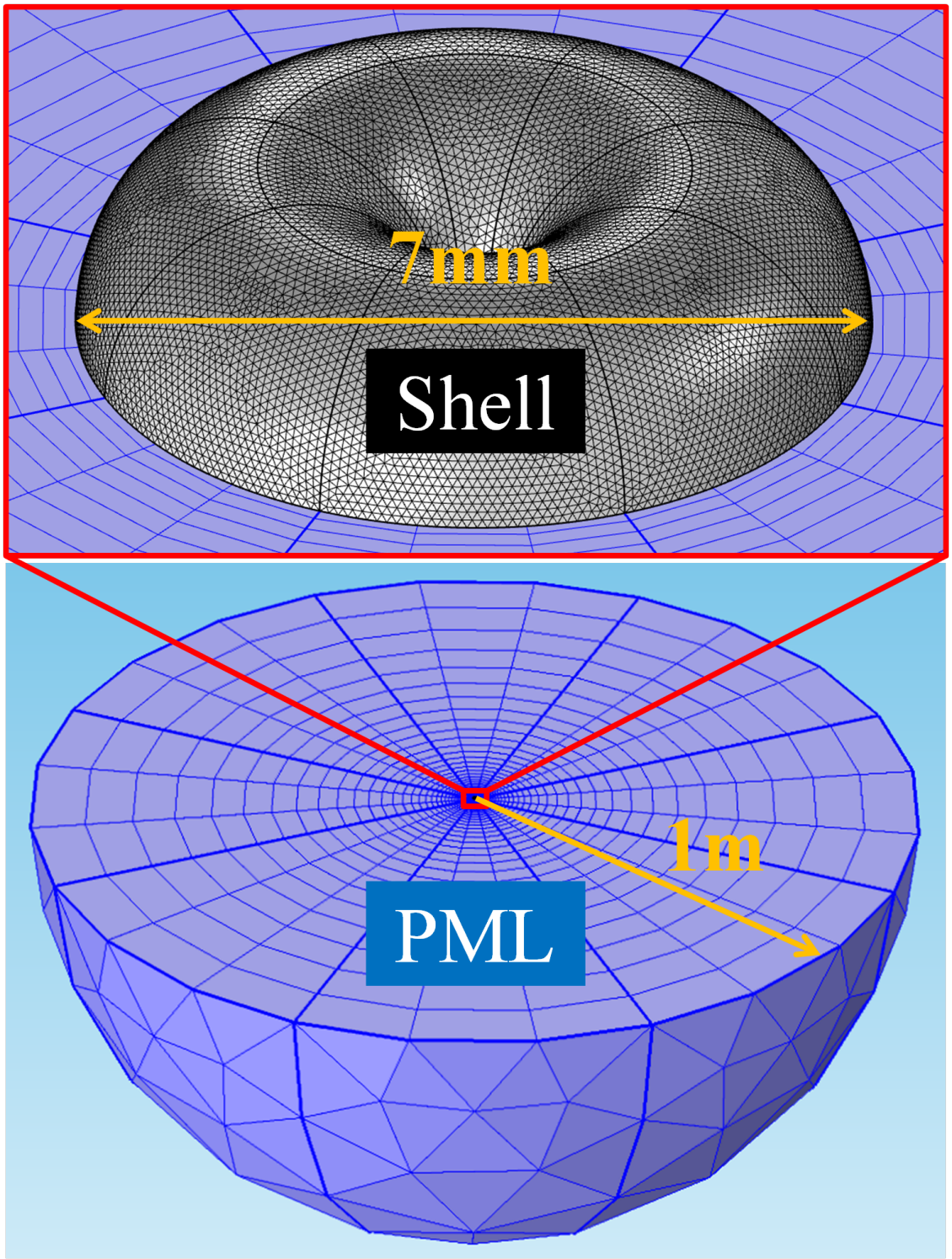

Figure 2.20: PML model developed in COMSOL MultiPhysics package for anchor loss simulation. The blue region is the PML and the gray region is the device. Varying mesh distribution in the PML is also shown.

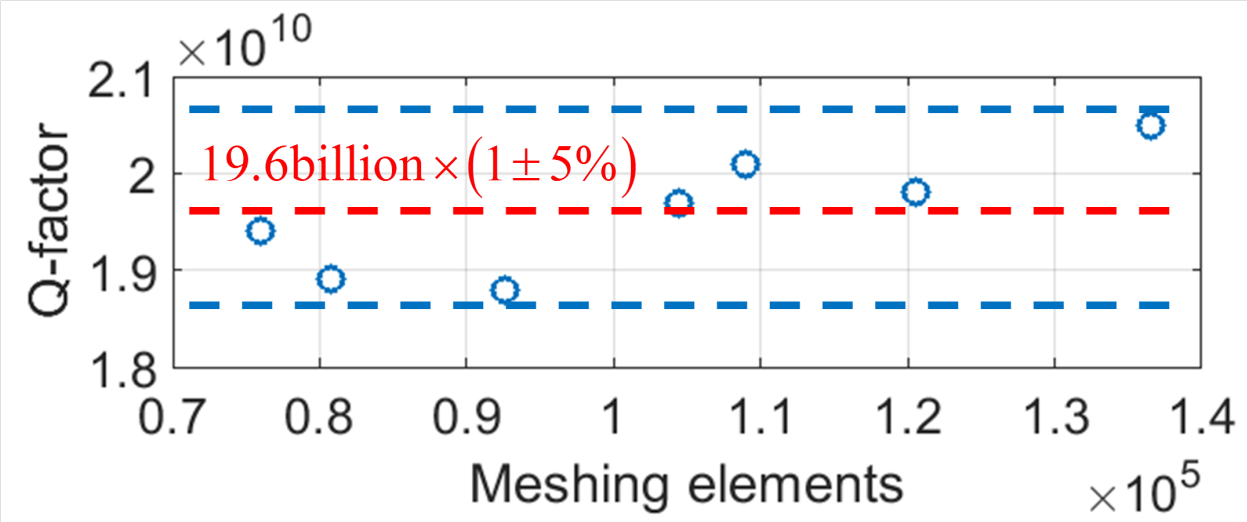

Figure 2.21: Relation between the meshing element number and the FEM results. Mesh elements, varying from 7.59×10^4 to 1.37×10^5, confirmed the convergence of the model with a tolerance of 5%.

The directional lapping procedure is based on the mechanical removal of the material as in the mechanical lapping process. As a result, the surface finish at the rim will not be degraded after the compensation process and the overall quality factor will not be affected. To experimentally prove this conclusion, the surface roughness of the rim of a wineglass resonator was measured before and after the directional lapping process. Atomic Force Microscope (AFM) from Pacific Nanotechnology (Nano-R) was used to measure the surface roughness of the samples. The scan area was selected to be 2μm × 2μm. Four different points on the rim were measured and the average was calculated. The sample was cleaned by a standard solvent cleaning (acetone, isopropanol, and methanol) before each scan. The AFM was run in a close contact mode, using a 10nm radius probe tip (Agilent U3120A). The results are listed in Table 2.1. The average surface roughness was 0.89nm and 0.86nm before and after directional lapping procedure, respectively. No obvious surface roughness change could be observed. AFM images before and after the compensation procedure are shown in Fig. 2.23. Trenches with the depth of around 10nm were observed in both measurements, indicating a similar level of surface finish.

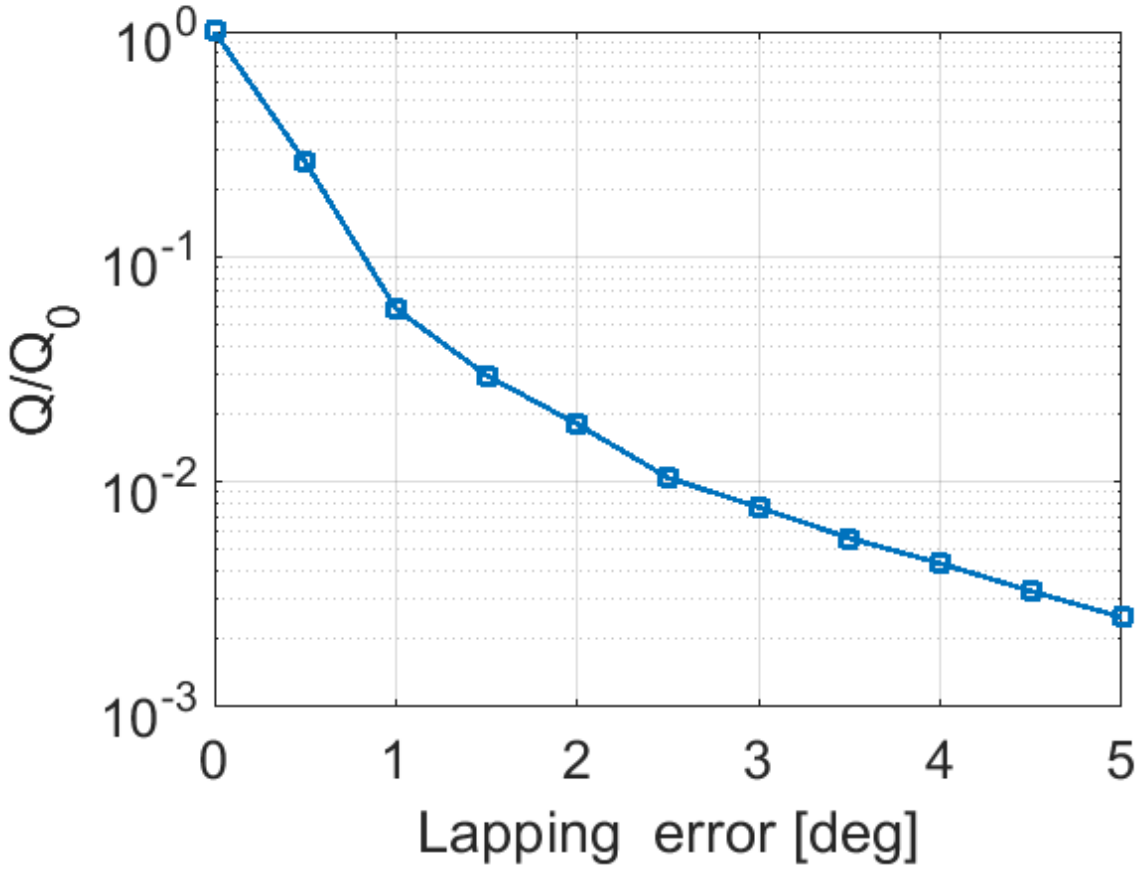

Figure 2.22: Relation between the lapping angle and the quality factor due to anchor loss of wineglass resonators. A 400 times of reduction in the quality factor due to anchor loss was numerically predicted with a lapping error of 5°.

Table 2.1: Surface roughness of rim of the shell before and after compensation

Sample status	**Surface roughness Sa [nm]**				
	***Point* 1**	***Point* 2**	***Point* 3**	***Point* 4**	***Average***
Before lapping	0.75	0.85	1.18	0.76	0.89
After lapping	1.17	0.64	0.92	0.70	0.86

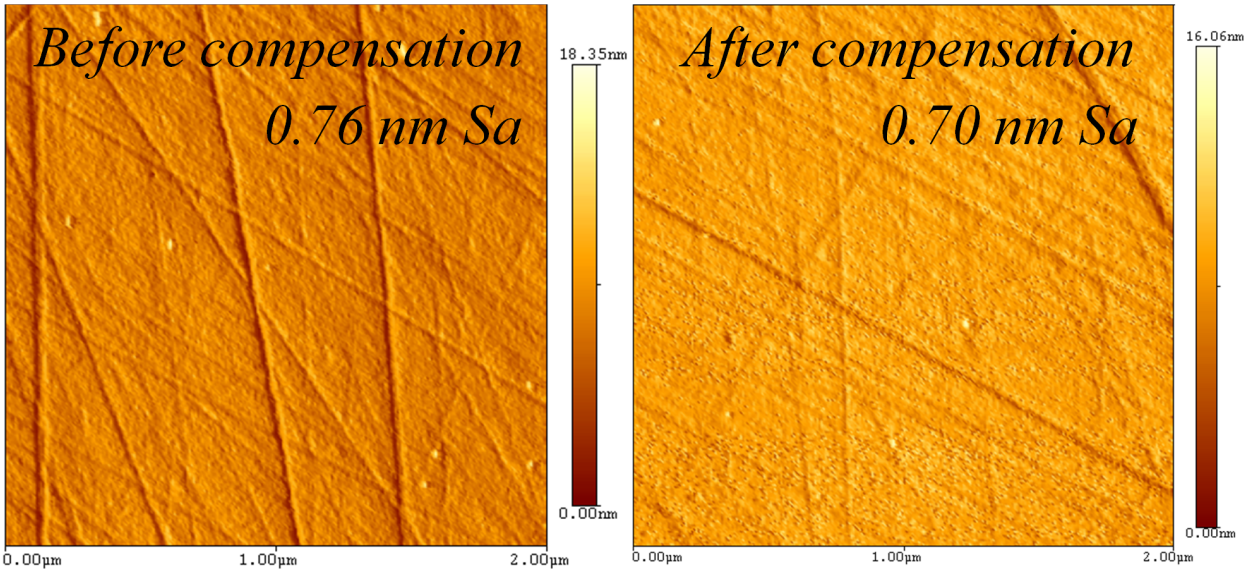

Figure 2.23: AFM images of the surface of wineglass rim before and after compensation. Images show that the quality of surfaces did not degrade after the compensation.

2.5 Conclusions

In this chapter, for the first time, we introduced and experimentally demonstrated a process for permanent structural compensation of the frequency split of 3D shell structures. We showed that the process was naturally integrated in the fabrication sequence, and thus completely compatible with MEMS fabrication. Near 6× of frequency split reduction, from 41Hz to 7Hz, was demonstrated on fused quartz wineglass resonators. Surface finish of the device after the compensation remained the same as before compensation, indicating that the process did not degrade the overall quality factor of the resonators.

The permanent structural asymmetry compensation procedure demonstrated in this chapter is envisioned as a step in the three-step fabrication and tuning sequence, where a precision

shell is fabricated, directionally lapped, and finally electrostatically fine-tuned. Due to limitations of electronics, it is challenging to control the large magnitude of tuning voltage with required high precision [136]. However, the directional lapping procedure is able to greatly reduce the initial frequency asymmetry of MEMS devices, and therefore only relatively low tuning voltages are required to accurately tune the devices. As a result, the structural asymmetry compensation method developed in this chapter helps to reduce the circuit-induced noise in the MEMS resonators and gyroscopes, and ultimately helps to improve the navigation accuracy for pedestrian inertial navigation.

Chapter 3

Sensor Development — Energy Dissipation Reduction

This chapter investigates the surface effects on the energy dissipation of Fused Quartz 3D wineglass resonators. Two main methods are proposed in the chapter to reduce the surface-related energy dissipation. In Section 3.1, the effects of surface roughness caused by different thermal and chemical post-processing techniques on the Fused Quartz surface are analyzed, and correlated to the quality factor of the device. Then in Section 3.2, an implementation of piezoelectric actuation is proposed, with a goal of avoiding metal deposition on the device surface and thus reducing the surface-related energy dissipation. Finally, the chapter is concluded with Section 3.3.

3.1 Post-Processing for Surface Quality

To increase the quality factor of the Fused Quartz micro wineglass resonators, different thermal and chemical post-processing techniques are tested to optimize the surface quality

of the device and the results are discussed in this section. In our study, the effects of thermal and chemical post-processes were initially tested on blank Fused Quartz wafers. Then, the effects of thermal reflow were tested on curved 3D Fused Quartz structures to show that the conclusions derived from blank samples were applicable to curved surfaces.

3.1.1 Experimental Procedure on Flat Samples

In our initial experiments, Fused Quartz wafers were first diced into 5mm by 5mm samples before different post-processing methods were conducted on the samples. The tested post-processing methods included: thermal reflow, Potassium hydroxide (KOH) etching, Buffered Oxide Etching (BOE), 10:1 hydrogen fluoride and hydrogen chloride solution (HF/HCl) etching, and RCA-1 surface treatment. Atomic Force Microscope (AFM) from Pacific Nanotechnology (Nano-R) was used to measure the surface roughness of samples and the scan area was $10\mu m \times 10\mu m$ for blank Fused Quartz samples. All samples were measured at three different points to guarantee the reliability of the measurement. The samples were cleaned by a standard solvent cleaning procedure (acetone, IPA, methanol) before each scan. The AFM was run in a close contact mode, using a 10nm radius probe tip (Agilent U3120A).

3.1.2 Results and Discussion

3.1.2.1 Chemical Post-Processing

Fused Quartz wafers typically go through a number of chemical processing steps before they are fabricated into wineglass resonators. All these processing steps have an influence on the surface quality of samples. Typical processes include 48% HF etching, 45% KOH hard mask removal, and RCA-1 cleaning. Some processes also include 10:1 HF/HCl etching and BOE. The effects of all these treatments on the surface quality of Fused Quartz were

investigated and reported in this section. Each treatment was applied to samples for the time duration that was typical for the fabrication process of wineglass resonator. The results are summarized in Table 3.1.

Table 3.1: Effects of different chemical treatments on surface roughness.

Comparison	**Sample status**	**Surface roughness Sa* (nm)**			
		Point 1	***Point 2***	***Point 3***	***Averaged***
1	Original	25.7	25.2	22.6	24.5
	10min KOH	50.5	47.7	47.8	48.7
2	Original	1.2	1.1	1.2	1.2
	30s BOE	5.4	5.2	5.5	5.4
	5h HF	1.7	1.3	1.0	1.3
	5h 10:1 HF/HCl	10.7	12.0	9.4	10.7
3	Original	7.5	5.6	5.8	6.3
	20min RCA-1	4.3	5.1	4.8	4.7

*Sa denotes the arithmetic average of a height function of a surface

Table 3.1 shows that both KOH etching and BOE deteriorated the surface of Fused Quartz, and HF etching created a lower surface roughness than 10:1 HF/HCl solution, while RCA-1 surface treatment improved the surface quality of Fused Quartz samples by reducing the averaged surface roughness from 6.3nm Sa to 4.7nm Sa. To uncover the active components of RCA-1 solution and to understand mechanisms of the effect, three Fused Quartz samples with the same surface quality were placed into RCA-1 solution (volume ratio of 1:1:5 for 27% NH_4OH, 30% H_2O_2, and DI water), H_2O_2 solution (0:1:6), and NH_4OH solution (1:0:6), respectively. The reaction temperature was controlled at 80 °C and the reaction time was 20 minutes for all three samples. The results are shown in Table 3.2.

Table 3.2: Effects of RCA-1 and its components

Sample status	**Surface roughness Sa (nm)**			
	Point 1	***Point 2***	***Point 3***	***Averaged***
Original	7.5	5.6	5.8	6.3
RCA-1	4.3	5.1	4.8	4.7
H_2O_2	7.5	6.1	6.2	6.6
NH_4OH	6.2	5.5	5.7	5.8

The results showed that H_2O_2 or NH_4OH alone did not greatly change the surface quality, however, their combination reduced the averaged surface roughness from 6.3nm Sa to 4.7nm Sa. According to [149], the SiO_2 etching rate was on the order of 0.1nm/min and it increased with higher NH_4OH/H_2O_2 ratio and higher temperature. This result suggested that the surface roughness of a sample might become worse if a higher etching rate was applied with certain etchants.

3.1.2.2 Thermal Post-Processing

Fused Quartz is an amorphous material that does not have a precisely defined melting temperature. As the temperature increases, the material becomes soft and its viscosity decreases. If the temperature of treatment is high enough, the surface tension of Fused Quartz can overcome viscosity and therefore can minimize the surface area, like in a liquid. Using the reflow phenomenon, the surface roughness can be improved.

Reflow temperature was one of the critical parameters during the reflow of 3D devices. On the one hand, the reflow temperature needed to be high enough, so that the surface tension could overcome viscosity and smoothen the surface. On the other hand, the temperature cannot be too high, so that the sample did not deform and lose its original symmetry due to gravity. Fused Quartz wineglass resonators were reflowed for 30 minutes at different temperatures, ranging from 1100 °C to 1400 °C, with 1300 °C turning out to be the highest temperature at which no obvious deformation was observed. Therefore, the reflow temperature was set to be 1300 °C for all experiments.

The change of surface roughness over time was studied. The thermal reflow at 1300 °C was applied to blank Fused Quartz samples with the same surface roughness for different time durations. The results are shown in Fig. 3.1. Each sample was measured at three different points and the red line represents an average value. The surface roughness was reduced by

increasing the time of treatment and the smoothening effect mainly took place in the first 30 minutes. It could be concluded from Fig. 3.1 that the reflow time of 1 hour was sufficient, reducing the averaged surface roughness from 24.5nm to 1.9nm.

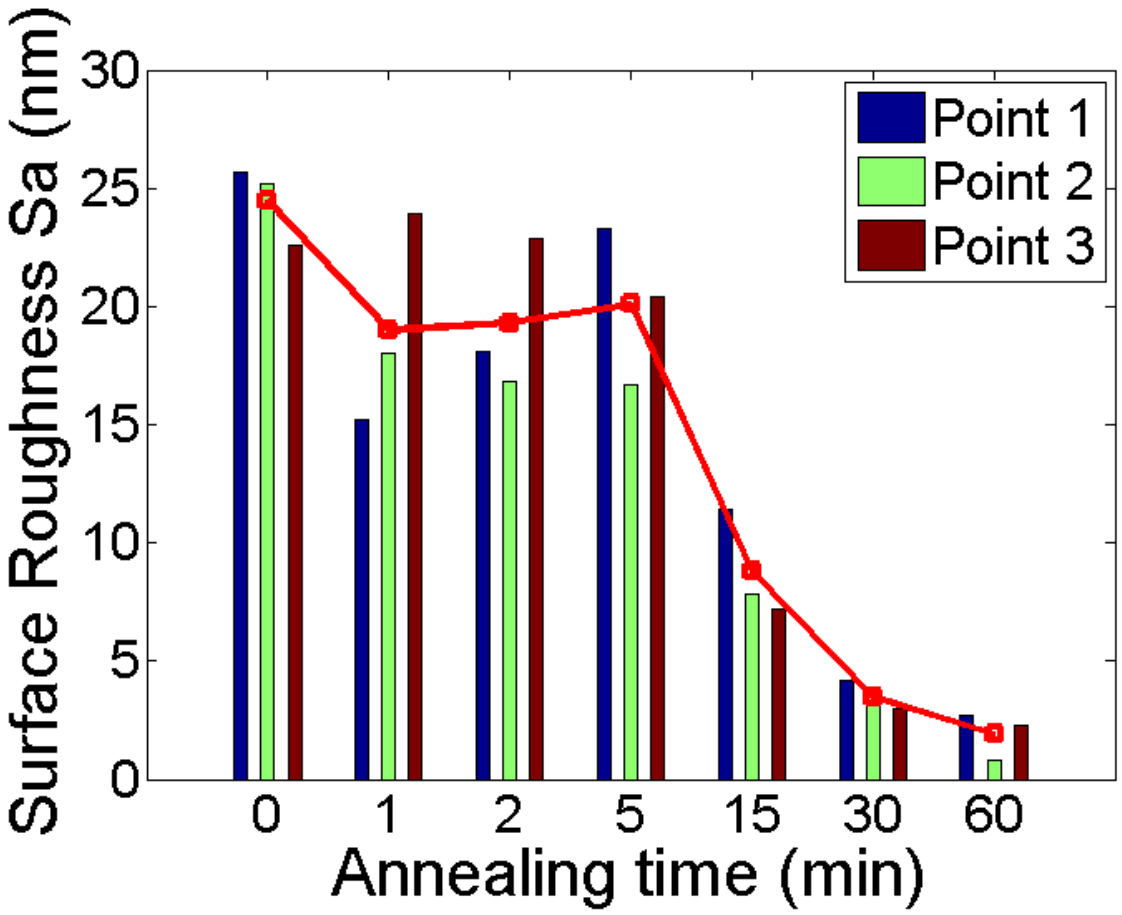

Figure 3.1: Surface roughness changes over reflow time. Reflow temperature was 1300 °C.

Fig. 3.1 shows that the averaged surface roughness did not change significantly during the first five minutes, but was greatly reduced with an extended duration of the experiment. To understand the process during the first five minutes, the Power Spectral Density (PSD) of the surface was analyzed. PSD of a surface is related to the 2-dimensional Fourier Transform of the surface height function and it contains more information than an averaged surface roughness. PSD is expressed as

$$P(\mathbf{k}) = \frac{1}{(2\pi)^2} \left| \int f(\mathbf{x}) e^{-i\mathbf{k}\cdot\mathbf{x}} d\mathbf{x} \right|^2, \tag{3.1}$$

where k is the wave number, x is the position vector, and $f(x)$ is the height function of the surface. Fig. 3.2 shows the PSD of original surface and surfaces after reflowing for 1min, 2min, and 5min. It shows that these surfaces could be modeled as self-affined [150] and surface roughness exponent α could be extracted to characterize the surfaces. The roughness exponent characterized the short-range roughness of a self-affined surface and it ranged from 0 to 1. A small value of α implied a rougher local surface, as observed in Fig. 3.3 [150]. For self-affined surfaces, the PSD and surface roughness exponent had a relation

$$P(k) \propto k^{-2-2\alpha}, \text{ for } k \gg \xi^{-1}, \tag{3.2}$$

where ξ is the lateral correlation length of the surface. Fig. 3.2 shows that the surface roughness exponent increased from 0.458 to 0.671, indicating that the short-range roughness was improved by reflow, although the general averaged surface roughness did not change.

Histograms of the original surface and surfaces after reflowing for 5min, 15min, and 60min are shown in Fig. 3.4, which is a characterization of the long-term surface morphology. The x-axis is the height and the y-axis is the percentage of the area at a corresponding height. The shape and position of the distribution peak in histogram did not change in the first five minutes and only very high elevation regions of the surface were affected. This confirmed that only improvement of the short-range roughness happened during this time period. After the first five minutes, the peak became more pronounced and shifted to the left, indicating a better surface quality.

3.1.3 Experiments on Curved Samples

All the previous results were reported for blank Fused Quartz samples. To show that the same effects could be observed on 3D wineglass structures, the surfaces were measured before and after the thermal reflow. Two glassblown samples from the same batch were tested before

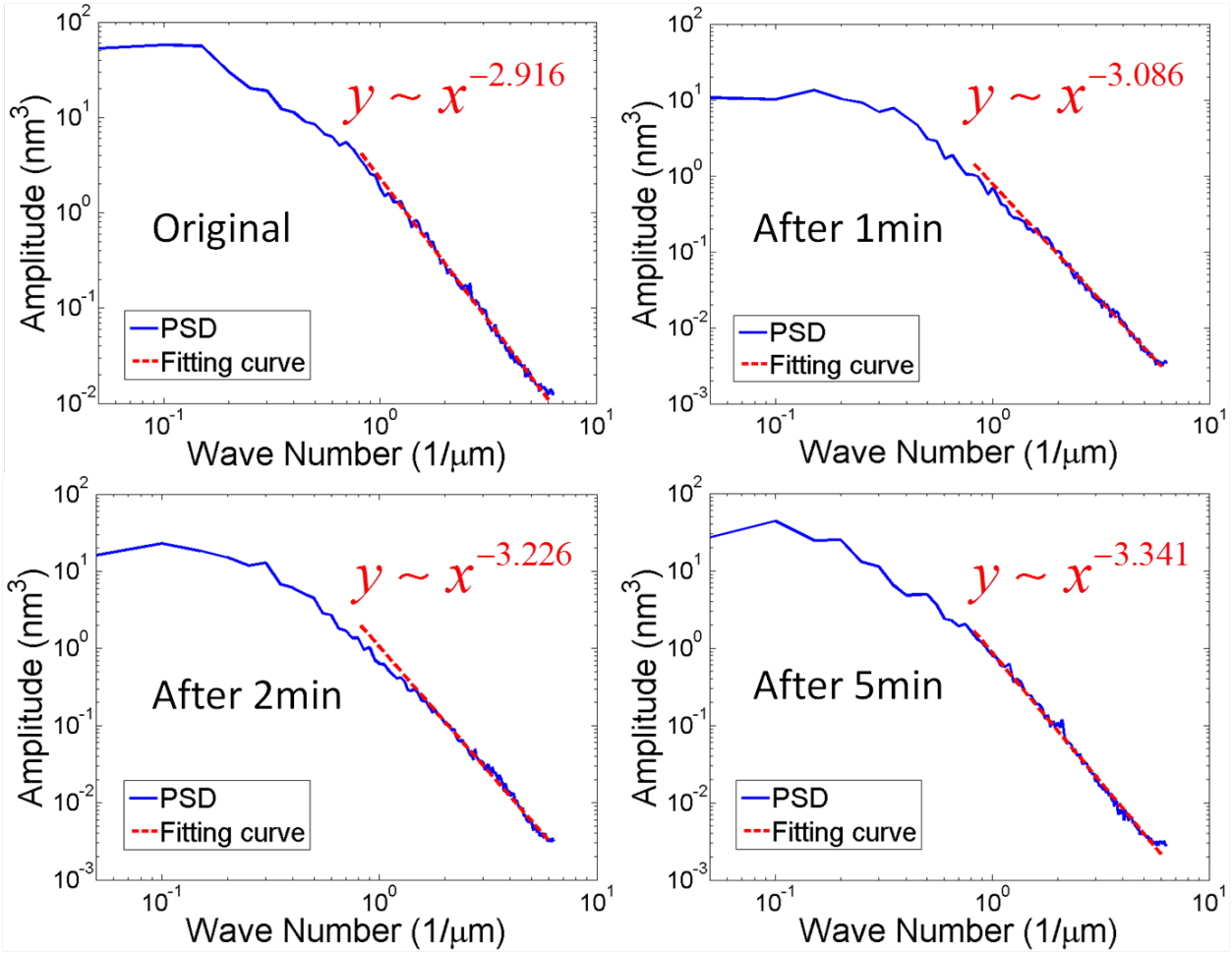

Figure 3.2: Power spectral densities of the original surface and surfaces after reflow for 1min, 2min, and 5min

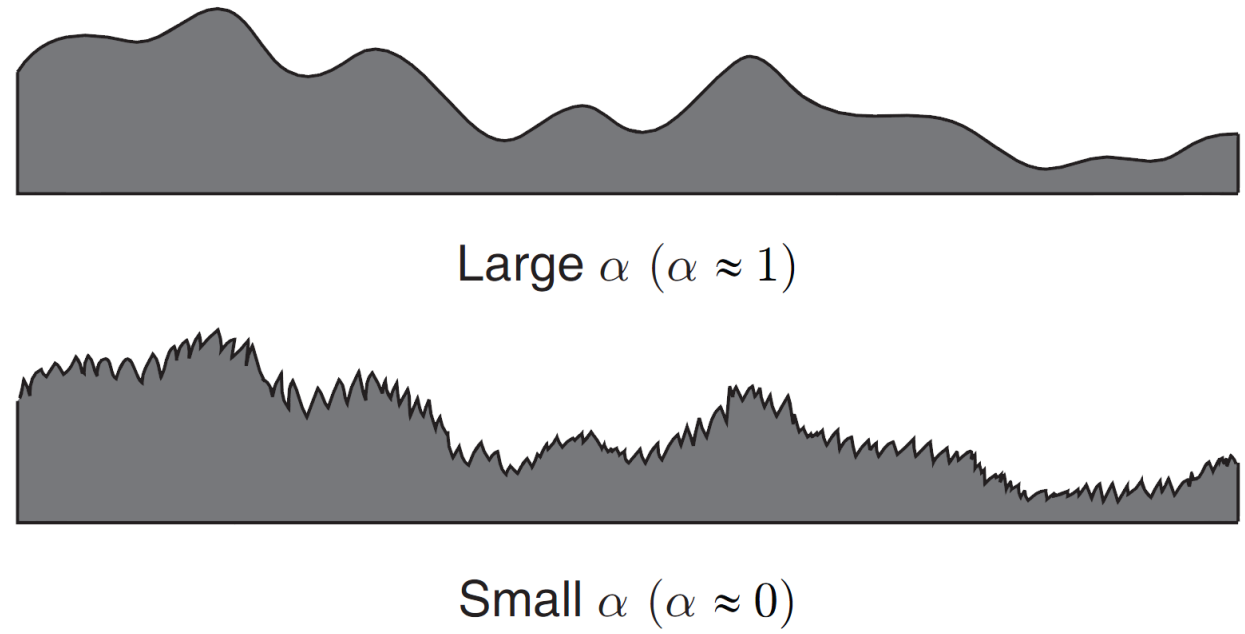

Figure 3.3: A comparison of the local surface morphology for similar surfaces with different values of α. A smaller value of α implies a rougher local surface, where α lies between 0 and 1 [150].

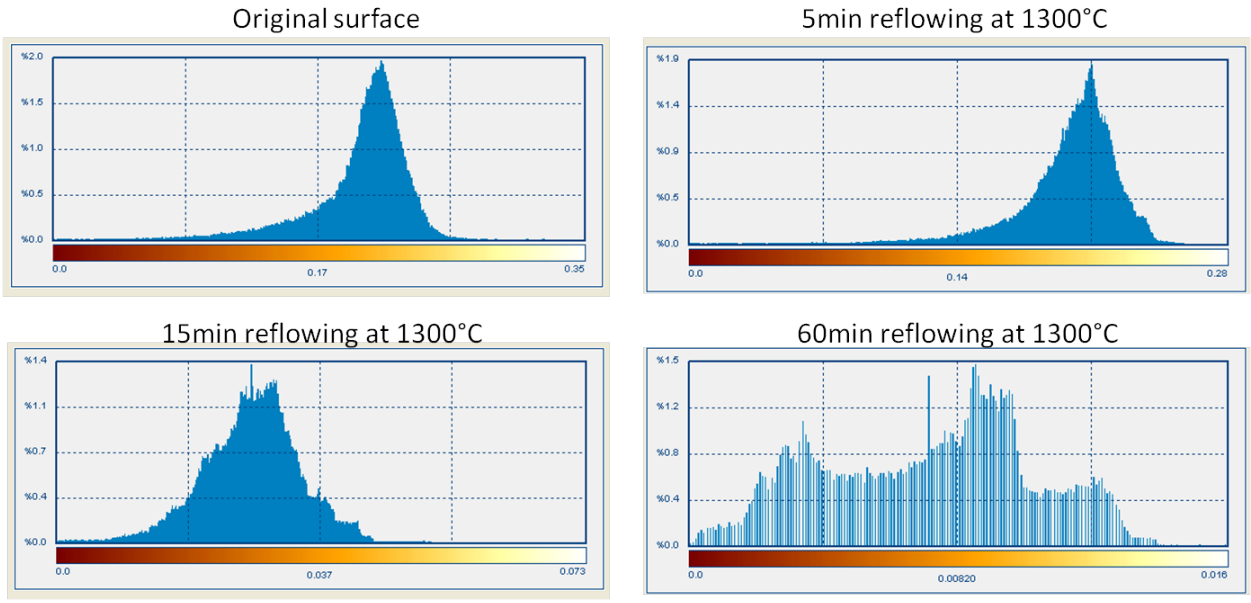

Figure 3.4: Histograms of the original surface and surfaces after reflow for 5min, 15min, and 60min, showing an improvement of long-term surface roughness.

and after the 1 hour reflow at 1300 °C. The devices were diced into smaller pieces to make sure no other parts of the sample except for the point being measured would be in contact with the AFM tip. The results are shown in Fig. 3.5. The averaged surface roughness was reduced from 1.9nm to 0.19nm. The same smoothening effect was observed in the 3D structures, implying that all the conclusions above were applicable to curved surfaces. Besides, the surface height distribution was lower and more concentrated, indicating a better surface quality after the thermal reflow.

3.1.4 Surface Roughness and Quality Factor

One of the goals of this study was to further improve the quality factor of Fused Quartz wineglass resonators. In this section, the quality factors of resonators with different surface roughnesses were measured, showing a direct correlation between improvement in the quality factor and an improved surface quality.

The experimental procedure for measurement of the quality factor of wineglass resonators was exactly the same as the measurement of resonant frequency, as discussed in Chapter 2. All devices were cleaned by solvent and RCA-1 solution. Next, the devices were placed in dehydration furnace for 45 minutes to remove all absorbates on the surface before characterization.

Three Fused Quartz wineglass resonators were tested in this study. First, the resonators were characterized and the quality factors were measured. Then, 10 minutes of KOH etching was applied to the resonators to roughen the surface and the quality factors were measured again. Then, the resonators were thermally reflowed at 1300 °C for 1 hour to improve the surface roughness and the quality factors were measured for the third time. The results of this experiment are shown in Table 3.3. A direct correlation between the quality factor of resonators and surface roughness could be derived. The reduction of quality factor related

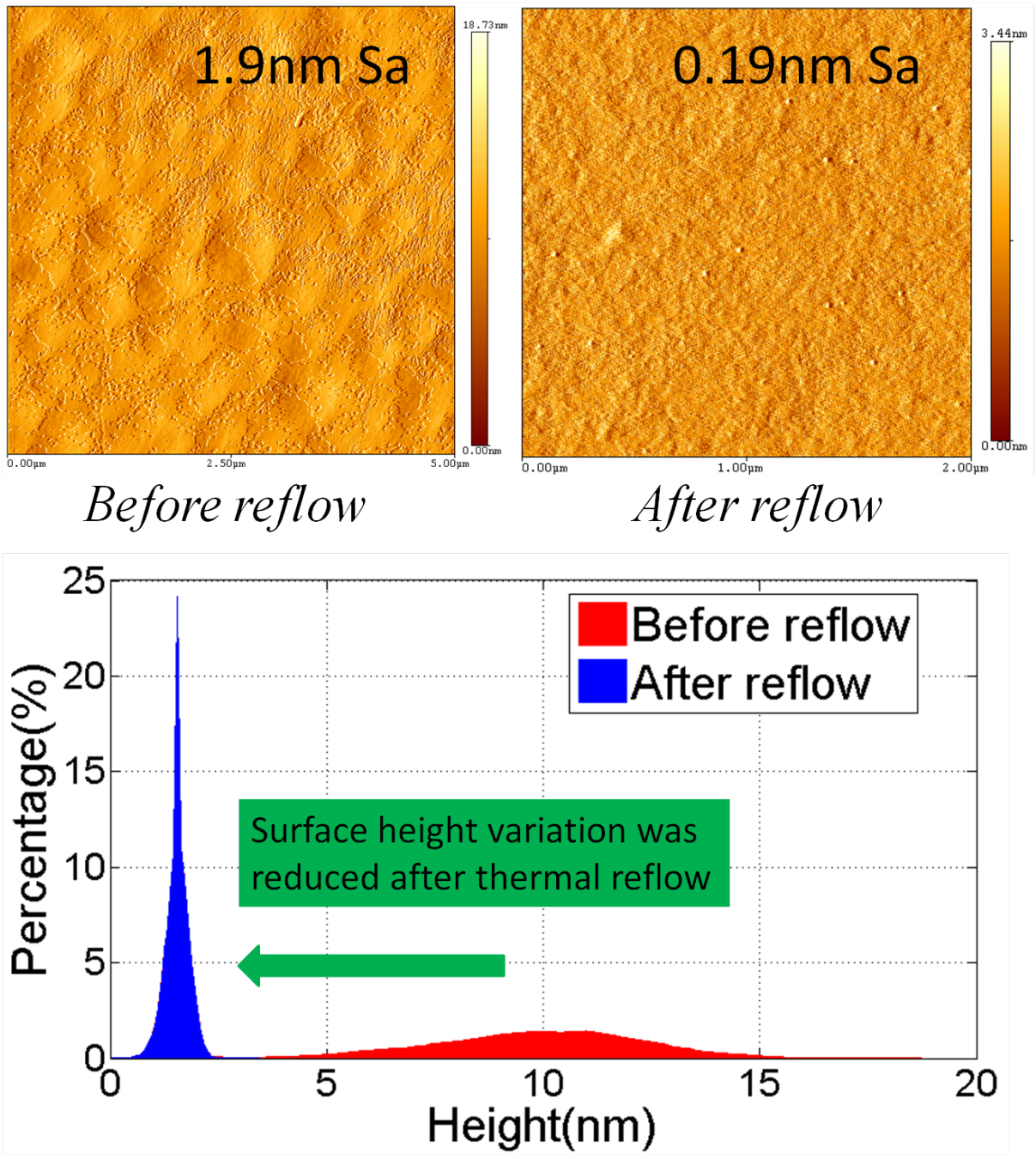

Figure 3.5: AFM images of the surface of 3D Fused Quartz structures and histograms before and after reflow. The averaged surface roughness was reduced from 1.9nm to 0.19nm and surface height distribution was lower and more concentrated, indicating a better surface quality.

to KOH etching was 17,000 on average and the increase of quality factor due to thermal reflow was about 6,000. It indicated that a part of the roughening effects due to KOH etching could be compensated by thermal reflow. All other loss mechanisms were remained the same, leading to the conclusion that changes in quality factor were due to surface losses. Quality factors of devices with higher original quality factors were not completely restored probably due to imperfections created during the reflow process, such as the deformation and handling of the devices during the reflow.

Table 3.3: Quality factors of Fused Quartz resonators after treatments

Sample number	**Quality factor**		
	Original	**After KOH**	**After reflow**
1	49,900	25,600	30,200
2	60,100	42,500	46,300
3	16,100	6,300	15,800

3.2 Piezoelectric Actuation for High Quality Factor

Metal deposition on the Fused Quartz resonators has been demonstrated to greatly reduce the device's quality factor due to a large mismatch of CTE between metal and Fused Quartz [79]. In this section, we explore the possibility of piezoelectric actuation in the Fused Quartz dual shell resonators in order to avoid the metal coating. Simulation and proposed fabrication process are also presented.

3.2.1 Fused Quartz Dual Shell Resonator

Fused Quartz dual shell resonator (Fig. 3.6) has been developed in the MicroSystems Laboratory at the University of California, Irvine [151, 152]. Micro-glassblowing technique is also involved in the fabrication process, and some of the advantages of such device over single shell resonators include:

- High shock survivability of over $50,000g$ due to the extra support of the cap shell;
- Co-fabricated package utilizing the cap shell;
- High fabrication yield due to the easy handling of the device during fabrication and characterization.

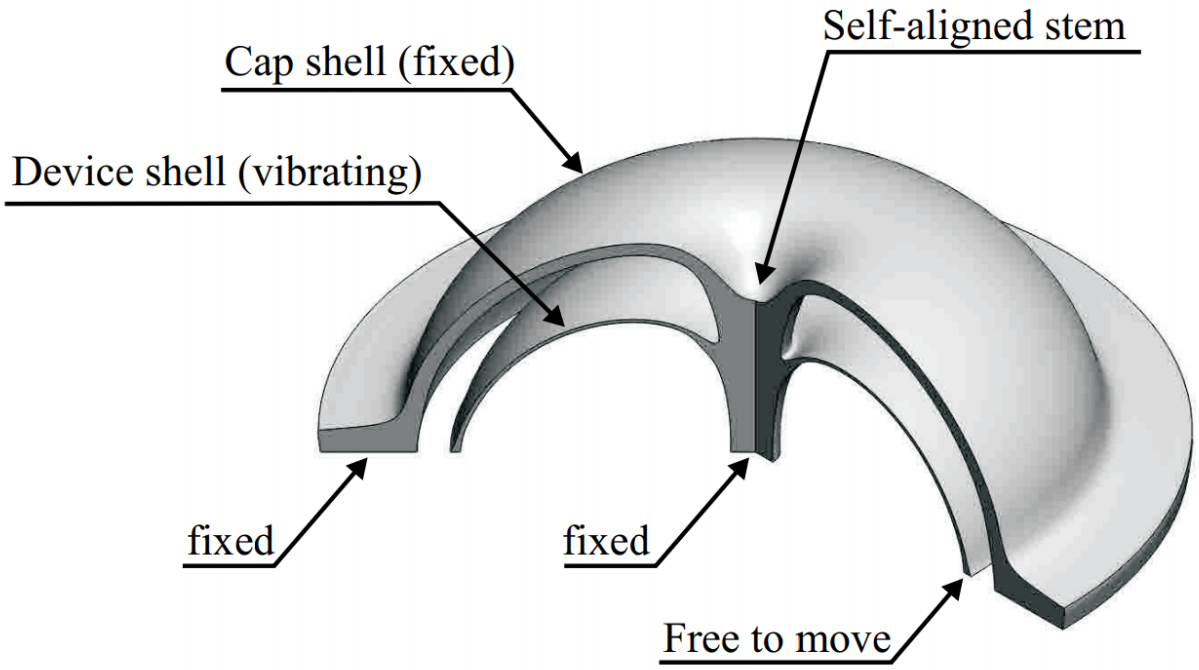

Figure 3.6: Schematic of the dual shell resonator [152].

Quality factor as high as 1.1 million has been demonstrated for $n = 2$ wineglass mode before metal coating [153]. However, in order to perform electrostatic actuation and sensing, metal coating is necessary for electrical conductivity, which typically reduces the overall quality factor by $10\times$. In this section, we explore an alternative actuation-sensing implementation that will eliminate the metal coating process, therefore keeping the high quality factor of the devices. Specifically, we explore the possibility of actuation of the device shell by depositing and driving the piezoelectric layer on the cap shell.

3.2.2 Piezoelectric Actuation

In this section, we numerically demonstrated the possibility of piezoelectric actuation by transduction of the vibration from the cap shell to the device shell. Only numerical and experimental results are reported in this section, while theoretical details are described in Section B.1 and B.2.

The first step was to optimize the geometry of the piezo electrodes for maximal transduction of vibration, so that lower driving voltage was needed to achieve large resonant displacement. The optimization technique was reported in [154]. The general idea was to deposit top electrode on the surface of the resonator where the signs of the principal stress were the same. By taking advantage of the same sign of the principal stress, both the e_{31} and e_{32} piezoelectric stress constants would excite or inhibit the mode, and as a result, the driving effect of the piezoelectric layer would reach its maximum. A typical geometry of the dual shell resonator was used in the modeling as a demonstration, and the resonant frequency for $n = 2$ wineglass mode was 8.5kHz. The principal stress on the outer surface of the cap shell for $n = 2$ wineglass mode is shown in Fig. 3.7, and the electrodes could be deposited on the red area or blue area to maximize the transduction rate. In the case of differential driving configuration, two sets of electrodes could be deposited on the red and blue area, respectively.

A method called Rapid Analytical-FEA Technique (RAFT) was used to simulate the frequency response of the dual shell resonator. This method was first reported in [155]. The first step of this method was to apply the FEA to the device geometry to analyze the information of different modes, such as resonant frequency, effective stiffness, and energy stored due to the piezoelectric effect. The second step was to use the analytical model to combine the information of different modes to generate the frequency response of the resonator. It was worth noticing that in this model, the quality factor of each mode should be manually

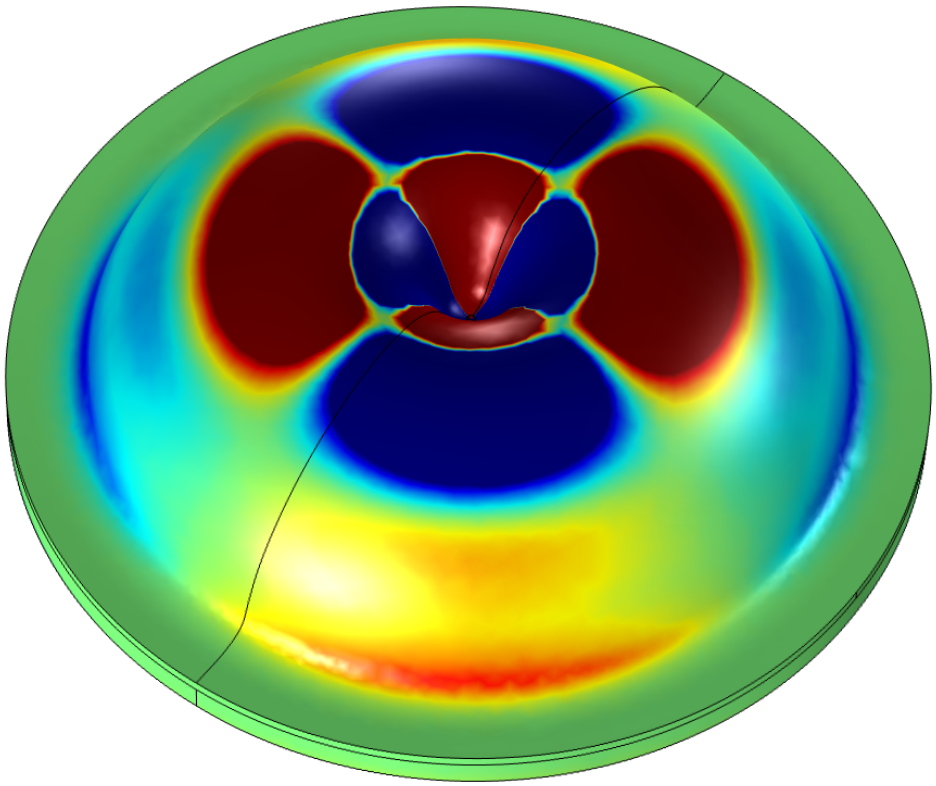

Figure 3.7: Principal stress on the outer surface of the cap shell for $n = 2$ wineglass mode.

assigned to generate the frequency response. For the $n = 2$ wineglass mode, a conservative quality factor value of 1×10^5 was assumed for the demonstration. The assumed quality factor for each mode was defined by their typical values obtained from experiments, and they were listed in Table 3.4. The resonant frequencies of different modes were obtained from simulation. An AC driving voltage of 10V was assumed. With these parameters applied, and assuming the optimal piezoelectric electrode shape, the frequency sweep response was calculated and presented in Fig. 3.8. All the major modes (the first 7 modes) have been captured. The predicted maximum displacement of the resonator was on the order of 10μm, which was large enough to be detected optically.

Table 3.4: Assumed parameters of first seven modes of the dual shell resonator

Mode	$n = 2$	$n = 1$	Out-of -plane	$n = 3$	Rotation	$n = 4$	$n = 5$
Frequency [kHz]	8.5	10.2	19.7	20.0	28.3	36.0	55.2
Quality factor	1×10^5	1×10^4	1×10^4	5×10^5	1×10^3	1×10^6	3×10^6

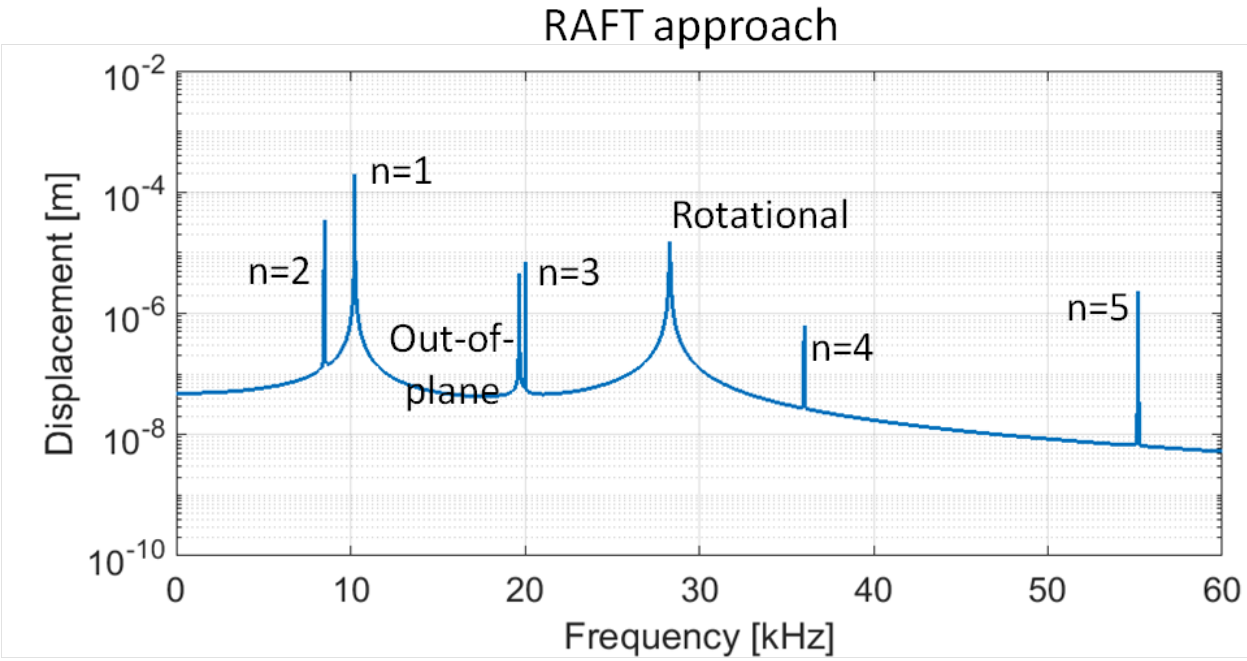

Figure 3.8: Frequency response of a typical dual shell resonator with the optimal piezo electrode shape.

Different electrode designs were compared in terms of their transduction rate, and the result is presented in Fig. 3.9. The x-axis is the applied AC voltage, and the y-axis is the corresponding amplitude of $n = 2$ wineglass mode. The optimized electrode pattern (d) showed the highest transduction rate with amplitude about 20μm at 10V driving voltage, verifying that it was the optimal design in terms of transduction rate.

3.2.3 Piezoelectric Layer Fabrication Process

Next, we present the fabrication aspect of the device. The proposed fabrication process for piezoelectric layer deposition on the cap shell is shown in Fig. 3.10. After the dual shell is formed in the micro-glassblowing step, the first step is surface cleaning by standard RCA-1 solution or Piranha solution to remove any organic residue before subsequent deposition process. In the deposition process, Titanium films is sputter deposited on the surface and then annealed in Oxygen to form the Titanium dioxide layer as the adhesion layer. The

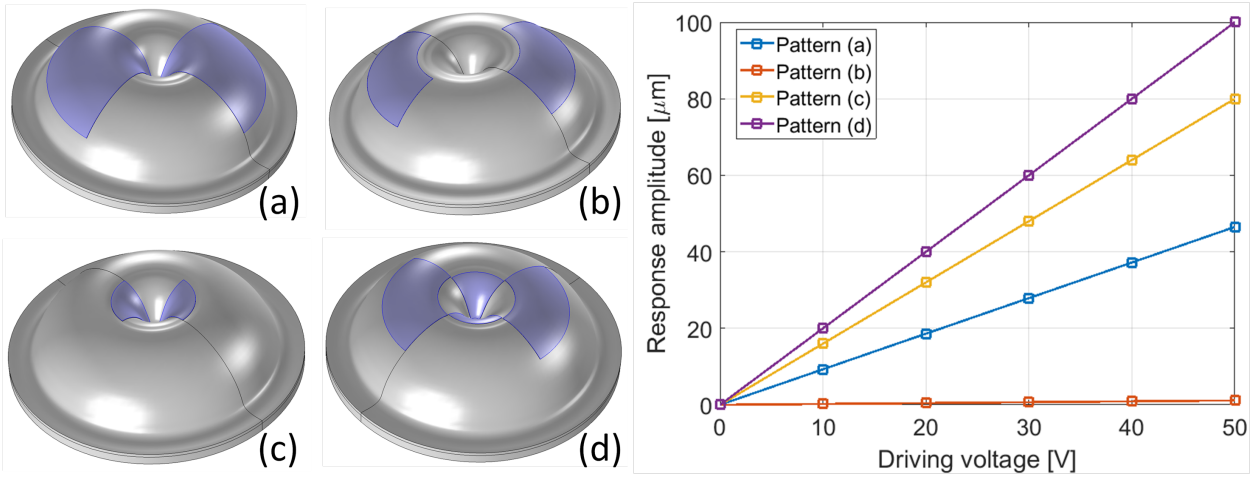

Figure 3.9: The comparison of different electrode shape designs.

thickness of the adhesion layer is typically 200nm. Then, highly textured (111) Platinum layer is sputter deposited on the adhesion layer as the bottom electrode. The thickness of Platinum layer is on the order of 100nm. Next, a 1μm PZT layer with a Zr/Ti ratio of 52/48 can be deposited using a chemical solution deposition method with a crystallization temperature of 700 °C. However, other piezoelectric materials, such as Aluminum Nitride (AlN), can be used for this purpose. The following step is for selective top electrode coating by shadow mask. The top electrode is also 100nm Platinum layer deposited by evaporation or sputtering, the exact thickness of the deposited material is not critical. Finally, the dual shell resonator can be released by lapping away the substrate and assembled to a ceramic package for further operation. A detailed process of piezoelectric stack fabrication on flat surface has been reported in [156]. Traditionally on flat devices, the top electrode, the piezoelectric layer, and the bottom electrode can be patterned by lift-off process or ion milling. However, in this study, a different technique is needed due to the curved surface of the structure. We experimentally demonstrated that shadow mask method could be used to achieve the goal.

Schematic of the shadow mask method for selective metal deposition on dual shell structure is shown in Fig. 3.11. At the bottom is the dual shell structure, on the top is the shadow mask, and in the middle is the frame used to align the shadow mask with the structure to

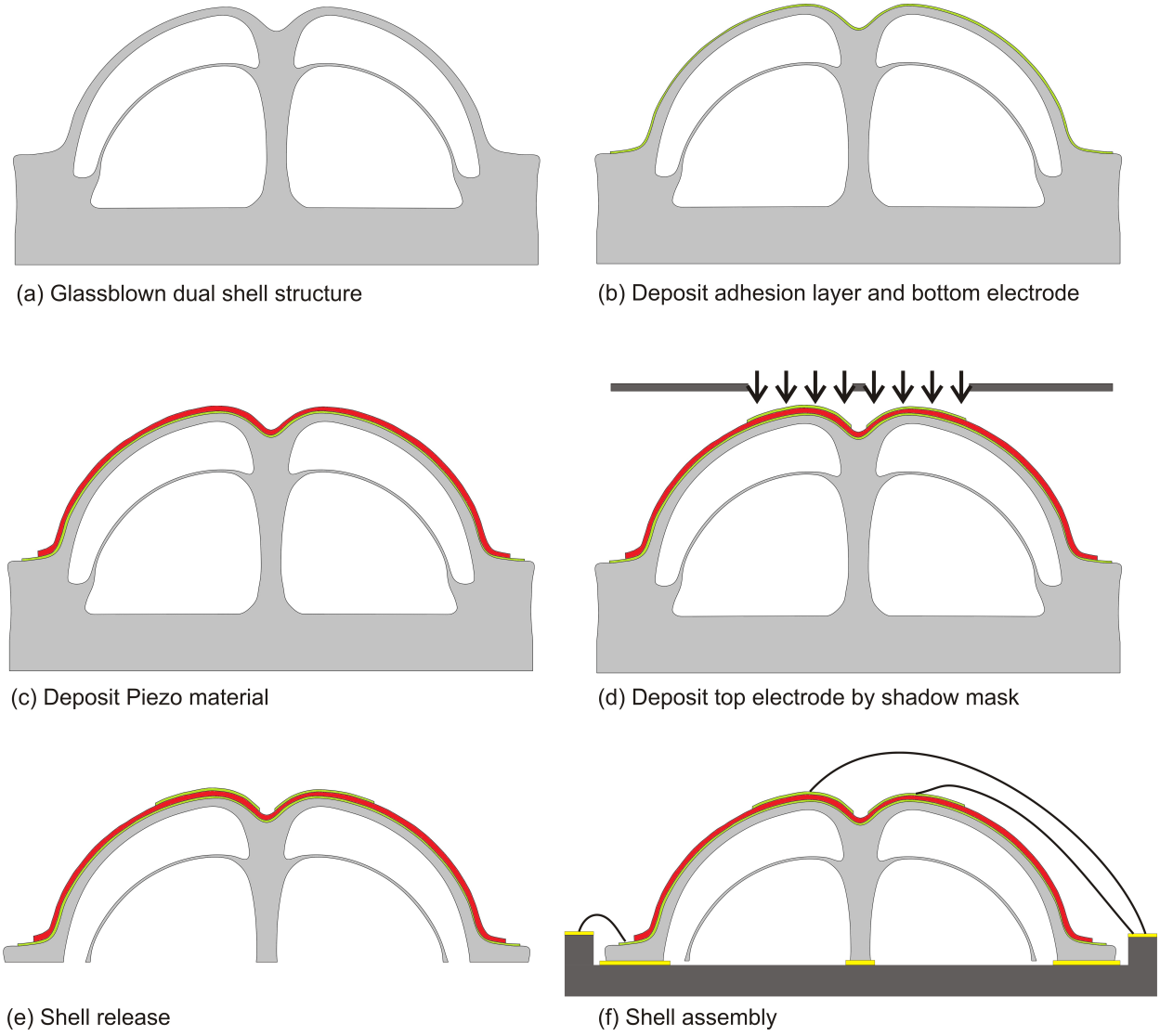

Figure 3.10: Process flow for piezo layer deposition on the cap shell.

be deposited. The shadow mask could be batch-fabricated by Deep Reactive Ion Etching (DRIE) of prime silicon wafers. Different shapes of opening could be designed to optimize the shape of the top electrode. The frame could be machined, for example, from aluminum or graphite to withstand the high temperature during deposition and also eliminate out-gassing. The shadow mask method was experimentally demonstrated to yield the result. The designed shape of the top electrode (the blue area on the left) and the deposition result (the dark area on the right) are shown in Fig. 3.12. A good agreement between the design and the result indicated the feasibility of the method.

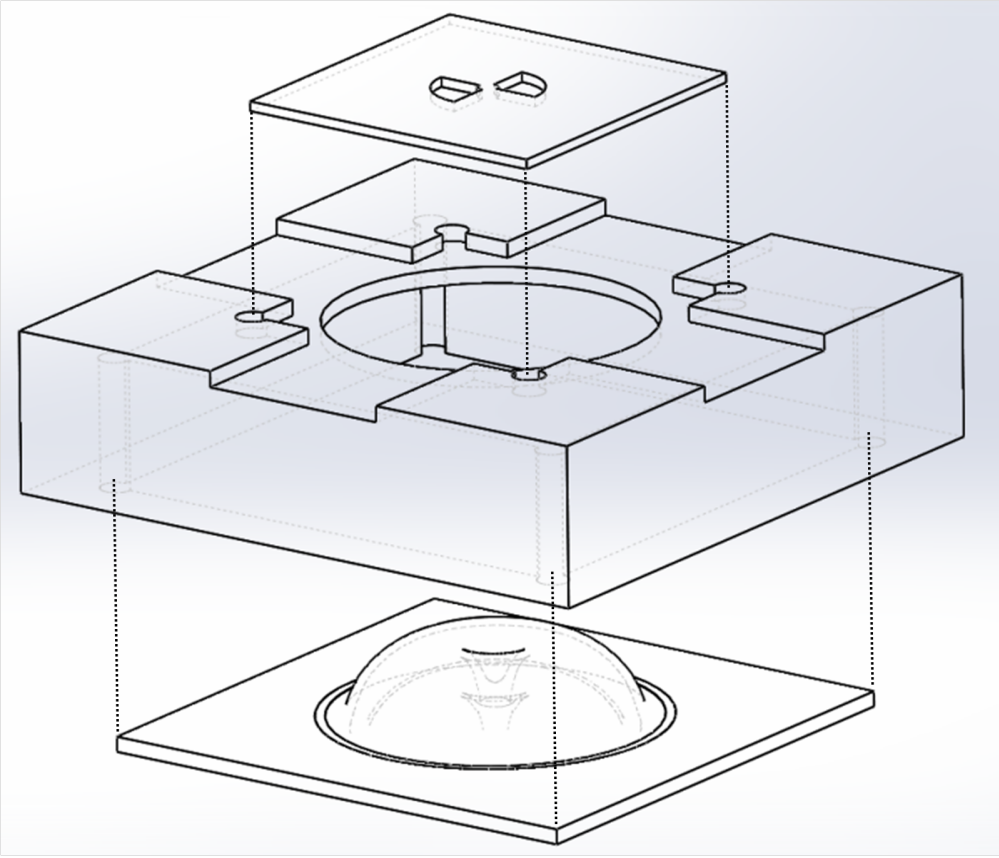

Figure 3.11: Schematic of the shadow mask method for selective metal coating on curved surface.

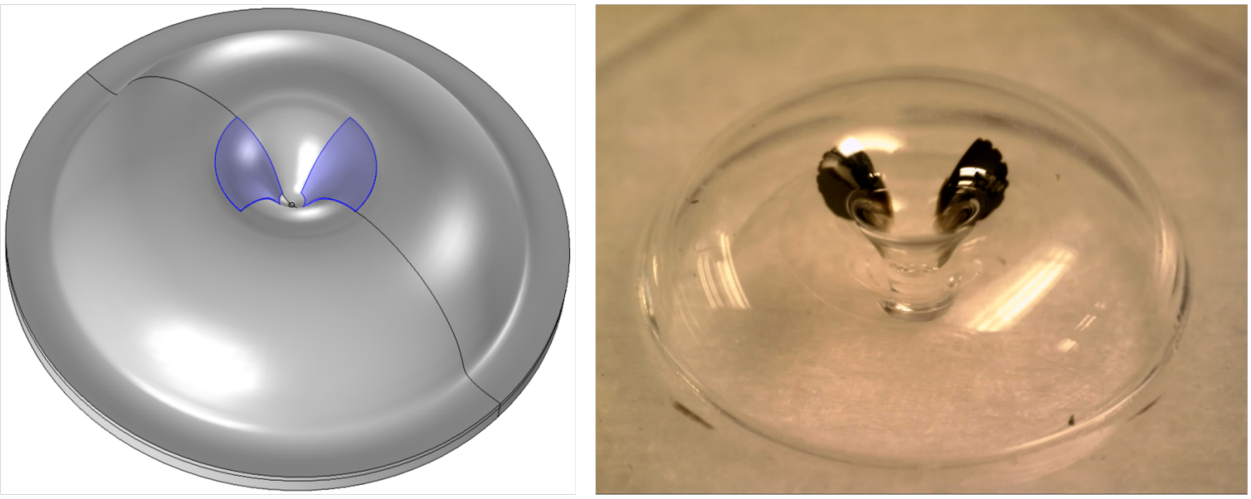

Figure 3.12: Comparison of the designed pattern and the experimentally obtained pattern.

3.3 Conclusions

In this chapter, methods to reduce the surface effect related energy dissipation were explored. To reduce the surface roughness, high-temperature reflow was demonstrated and the energy dissipation in Fused Quartz 3D wineglass resonators was reduced. The effects of chemical and thermal post-processing on the surface quality of Fused Quartz samples were studied. Thermal reflow at 1300 °C for 1 hour showed the best result, reducing the averaged surface roughness from 24.5nm Sa to 1.9nm Sa. To avoid the energy loss caused by metal coating, piezoelectric actuation was explored for Fused Quartz dual shell resonator. Shadow mask method was experimentally demonstrated to be able to selectively deposit metal on curved surfaces. Electrode-shaping was conducted to maximize the transduction rate of the piezoelectric actuation, and the frequency response of the device was simulated to verify effects and prove the feasibility of the approach.

Due to the low CTE of Fused Quartz and the balanced structural geometry, the TED and anchor loss has been minimized in the Fused Quartz 3D wineglass resonators. Thus, the surface loss is typically the dominant energy dissipation mechanism in the device. Therefore, the techniques developed and demonstrated in this chapter have the potential of greatly

improving the overall quality factor of the device, and ultimately improving the navigation accuracy for pedestrian inertial navigation. No experimental demonstration of this type of actuation has been demonstrated, and this remains the topic of future research.

Chapter 4

Sensor Development — Energy Dissipation Quantification

In this chapter, a methodology for analysis of energy dissipation mechanisms in MEMS devices and the possibility of quantifying the contribution of each mechanism independently are developed and demonstrated. The experimental setup is first reported in Section 4.1. Next in Section 4.2, thorough analysis and quantification on major energy dissipation mechanisms, including viscous air damping, TED, anchor loss, and surface loss, are presented. The following is a discussion on the obtained results in Section 4.3. Finally, the chapter is concluded with Section 4.4.

4.1 Experimental Setup

The Toroidal Ring Gyroscope (TRG) was selected as the test platform for its structural simplicity and a demonstrated potential for high performance. Such designs were first reported in [157, 158]. It consists of a concentric ring structure, an outer anchor that encircles the

rings, and an inner electrode assembly. The outer anchor structure leads to a more robust support of the structure compared to devices with a central anchor. The device was fabricated using the Epi-Seal process [159], which utilized epitaxially grown silicon to seal the device layer at extremely high temperatures, resulting in an ultra-clean wafer-level sealing. Surface loss of the device could be minimized with a smoother surface due to silicon migration at high temperature during the sealing process [160]. The device was fabricated on single-crystal $< 111 >$ silicon wafers to achieve a better structural symmetry, enabling the operation in the $n = 2$ mode. The Scanning Electron Microscope (SEM) image of the device is shown in Fig. 4.1. The diameter of the ring structure was 1000μm and it included 20 concentric rings with a width of 6μm. The rings were connected to each other with 6 spokes in between two consecutive rings. The thickness of the device was 60μm and the gaps in between the rings were 1.5μm. The designed resonant frequency of the $n = 2$ mode of the TRG was 68.7kHz and the experimentally measured frequency was 54.5kHz. The difference in frequency was possibly due to the over-etching of the device. One of the benefits of this specific design was that the $n = 2$ mode was the mode with the lowest natural frequency. It came before all the out-of-plane modes because of the large thickness-to-width ratio of the rings. It also came before the $n = 1$ rocking mode due to the phenomenon of mode ordering [161]. This order of the modes helped to improve the common mode rejection of external accelerations.

To be able to adjust the air pressure and temperature at which the device was operated, a few steps were taken. First, Focused Ion Beam (FIB) was used to form a small hole through the cap to vent the entire cavity (the inset of Fig. 4.3). The hole was drilled in between the electrodes to avoid damaging the device, and diameter of the hole was about 10μm. Next, the die was attached and wire-bonded to a Leadless Chip Carrier (LCC) package. Then, the package was placed into the vacuum chamber of LakeShore FWP6 Probe Station [162], where the operational temperature could range from 4.5K to 475K, and 6 probes were available for signal transmission between the device and the circuit outside the probe station system. The

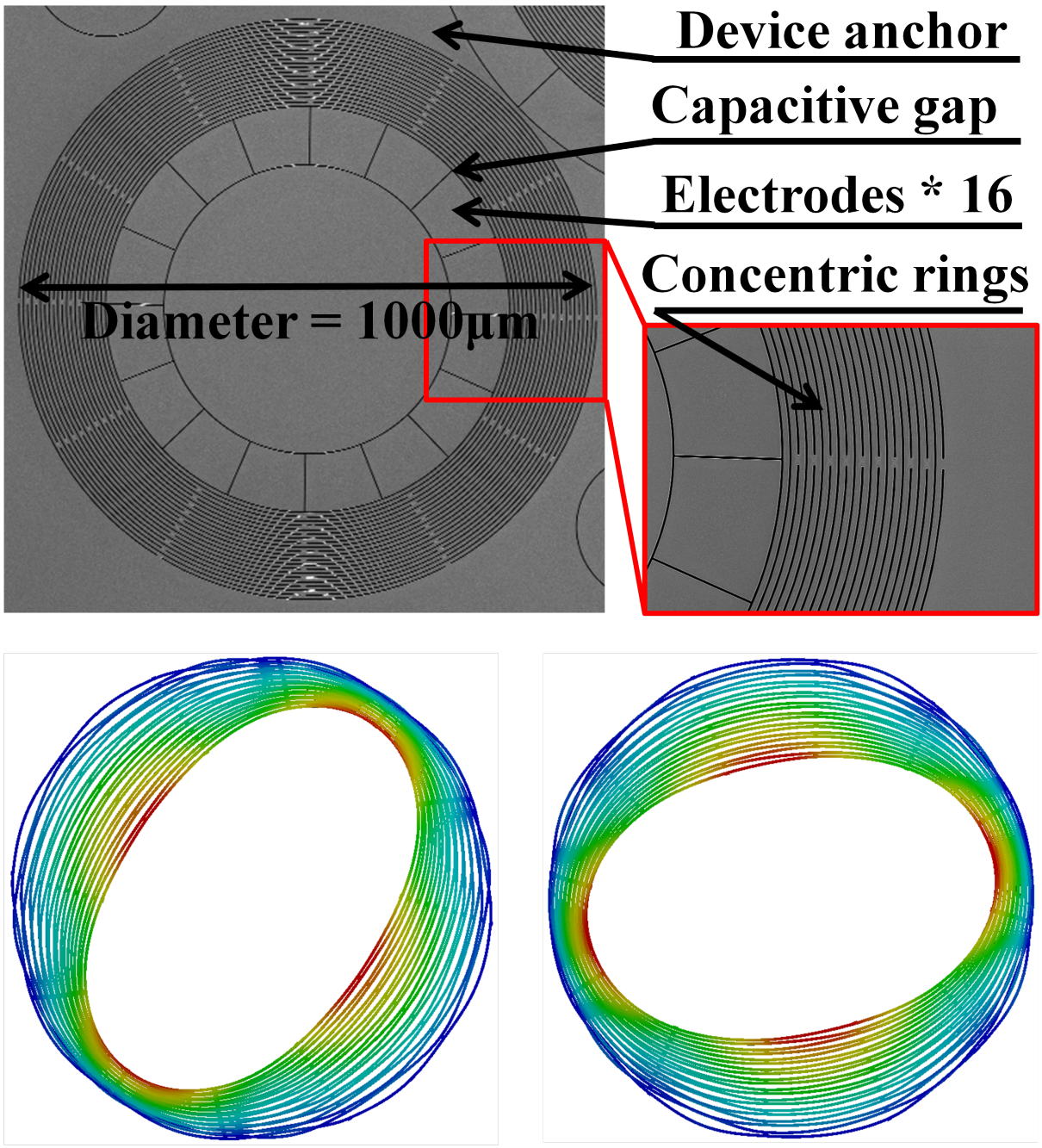

Figure 4.1: Details of the device used in this study and the $n = 2$ mode shapes.

experimental setup is shown in Fig. 4.2. Liquid Nitrogen was used to cool down the vacuum chamber, and a PID controller was used to control the heater installed in the chamber to stabilize the temperature.

Quality factor of the device was extracted by ring-down method, where the device was actuated at resonance by a Phase-Locked Loop (PLL), and the resonant frequency f was recorded; after the magnitude of motion became constant, the actuating AC voltage was removed; next, the time it took until the magnitude of motion was decayed to $1/e$ of the original level was measured as the ring-down time τ. The quality factor could be calculated as

$$Q = \pi \tau f. \tag{4.1}$$

Ring-down method provided a better measurement of the quality factor compared with the frequency sweep method, especially for resonators with high quality factors and low resonant frequencies. In the method of frequency sweep, it was necessary to measure the natural bandwidth of the resonator, which was typically small for high-Q devices. Thus, any measurement errors in frequency might cause a large error in the quality factor. Besides, due to the small bandwidth of the device, the frequency sweep had to be conducted slowly to wait for transient-state response to die out, such that an accurate frequency peak could be obtained.

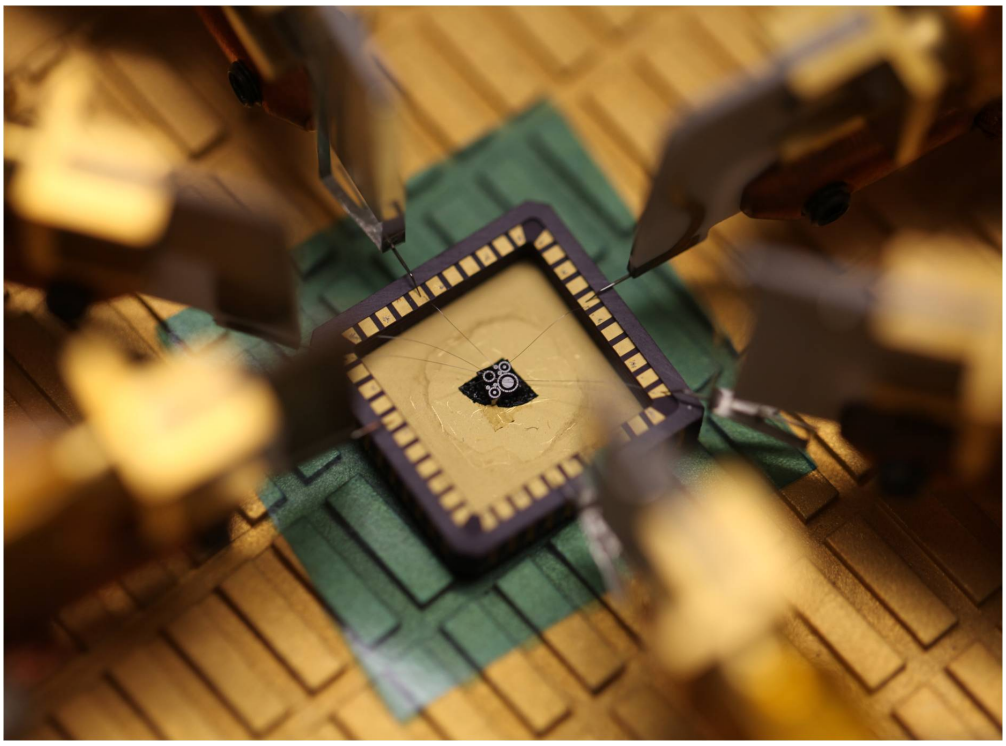

Figure 4.2: Experimental setup used in this study. The device was attached and wire-bonded to an LCC package, which was placed in the vacuum chamber of the probe station system, and six probes were used for driving and sensing of the device.

4.2 Energy dissipation Analysis

4.2.1 Viscous Air Damping

Viscous air damping of a resonator is related to its geometry and the pressure under which the device is operated. Although devices fabricated in the Epi-Seal process are sealed with pressure near 0.1Pa [159], the narrow gaps (1.5μm) and a low frequency motion of the rings of the resonator will induce some viscous damping, whose effect may be dominant especially if other dissipation mechanisms are suppressed.

It is impossible to directly measure the pressure inside the cavity of the Epi-Seal devices. Therefore, we utilized the overall quality factor of the resonator as an indicator of the pressure. The quality factor of the sealed device was first measured to be 128,000 at room temperature. Next, we used the Focused Ion Beam (FIB) to form a small hole through the cap, exposing the interior of the resonator to the surrounding atmosphere, as shown in Fig. 4.3. The device was then placed into a vacuum chamber with adjustable air pressure and the quality factor of the device was measured as a function of pressure at room temperature. The relation between the air pressure and the quality factor of the device is shown in Fig. 4.3. The figure shows that the quality factor of the device reached 170,000 with air pressure below 1mPa, indicating that viscous air damping was overwhelmed by other dissipation mechanisms at pressures below 1mPa. The original quality factor of 128,000 for the sealed device corresponded to a pressure level of 40mPa, providing a good estimate for the pressure in the cavity of the device before venting. To suppress the effects of viscous air damping on the overall quality factor, we tested the vented device at pressures below 0.1mPa in the following sections.

By comparing the quality factor of the device before and after FIB drilling of the cap over a wide temperature range, the magnitude of air damping of the device before venting the cap

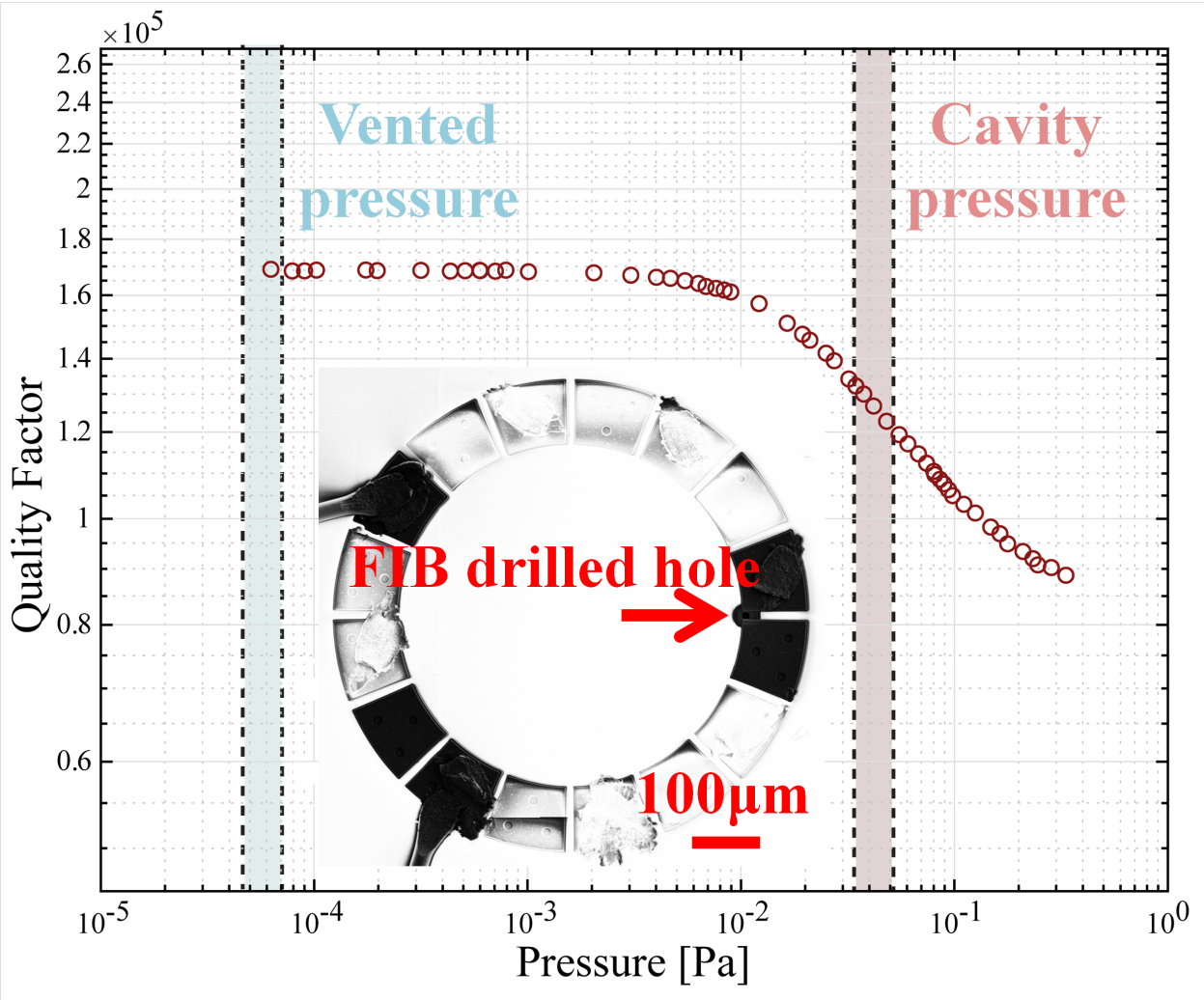

Figure 4.3: Relation between the quality factor of TRG and the air pressure at room temperature. Inset is the SEM image of the device with the hole drilled by FIB.

could be estimated. The result is presented in Fig. 4.4. The red dots are the experimental results measured before drilling the cap, and the blue dots are the data obtained after venting the package and pumping the vacuum chamber to below 0.1mPa. The device showed a higher quality factor in better vacuum, as expected from the earlier pressure sweep results in Fig. 4.3. The difference in quality factor could be calculated, and it is shown by the yellow dots. Since the only difference between the two measurements was the air pressure in which TRG was operated, the quality factor difference was contributed to viscous air damping. The relation between the viscous air damping and the temperature of the device could be expressed as

$$Q_{\text{air}} \propto \frac{\omega\sqrt{k_B T}}{p} = \frac{\omega\sqrt{k_B}V}{nR}\frac{1}{\sqrt{T}}, \tag{4.2}$$

where ω is the resonant frequency of the device, k_B is the Boltzmann constant, T is the temperature, p is the air pressure in the cavity, which can be related to temperature by the ideal gas law: $pV = nRT$, where V is the volume of the cavity, n is the amount of substance of the gas, and R is the ideal gas constant. If the changes of resonant frequency and the volume of the cavity are neglected, then the quality factor due to air damping is inversely proportional to the square root of temperature. The fitted curve of viscous air damping is shown by the purple dotted line in Fig. 4.4. The quality factor due to viscous air damping was estimated to be 625,000 at room temperature (300K) before venting the cavity. It was reasonable to assume that all TRG devices from the same batch would experience the same level of air damping, since they were of the same geometry and operated under the same air pressure.

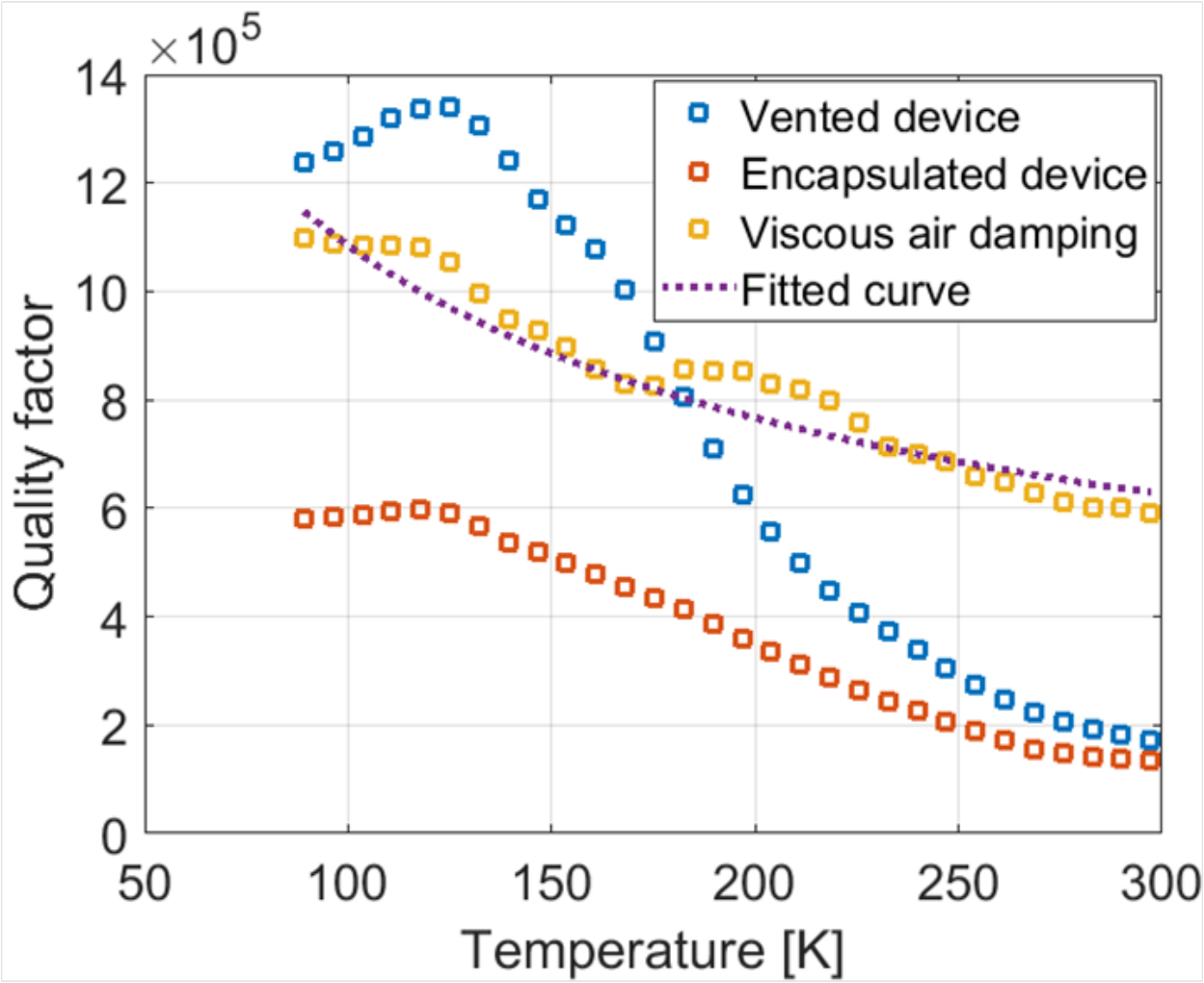

Figure 4.4: Relation between the quality factor of TRG and temperature before (red dots) and after (blue dots) FIB drilling of the cap. The yellow dots show the calculated magnitude of viscous air damping at different temperatures.

4.2.2 Thermo-Elastic Damping

Thermo-elastic damping is related to the exchange of energy between temperature and elastic deformation fields in resonating structures. The coupling term between the two fields is the CTE of the material. It has been shown that the magnitude of TED in a cantilever beam is proportional to the square of CTE of the material

$$Q_{\mathrm{TED}} \propto \frac{\rho C_p}{\alpha^2 T E}, \tag{4.3}$$

where ρ is the density, C_p is the heat capacity, α is CTE, T is the temperature, and E is Young's modulus of the material. For the TRG devices used in this study, the general principle of TED through CTE-coupling between the temperature and elastic deformation fields still applied, although they had a more complicated geometry. Silicon is a material whose CTE is strongly dependent on the temperature and it crosses zero at temperature of near 123K [163]. A numerical simulation was conducted by FEA in COMSOL MultiPhysics to predict the TED of TRG at different temperatures. The thermo-mechanical module was utilized to model the coupling of thermal and elastic deformation fields. Using the full set of temperature-dependent material parameters reported in [163], prediction of the temperature dependence of the TED contribution to the quality factor was built, and the result is shown in Fig. 4.5. The blue solid line and the red solid line present the CTE and TED over temperature, respectively. At room temperature, the model predicted a Q of 200,000, which was in reasonable agreement with the experimental result of 170,000, and Q_{TED} reached infinite at around 120K due to a zero CTE. Because of these effects, the TED contribution to the overall quality factor as a function of temperature should have a strong and recognizable signature, making it possible to quantify TED in the resonator.

The signature of the effects of TED on the overall quality factor of TRG could be clearly seen in Fig. 4.4, especially after eliminating viscous air damping. The maximum quality

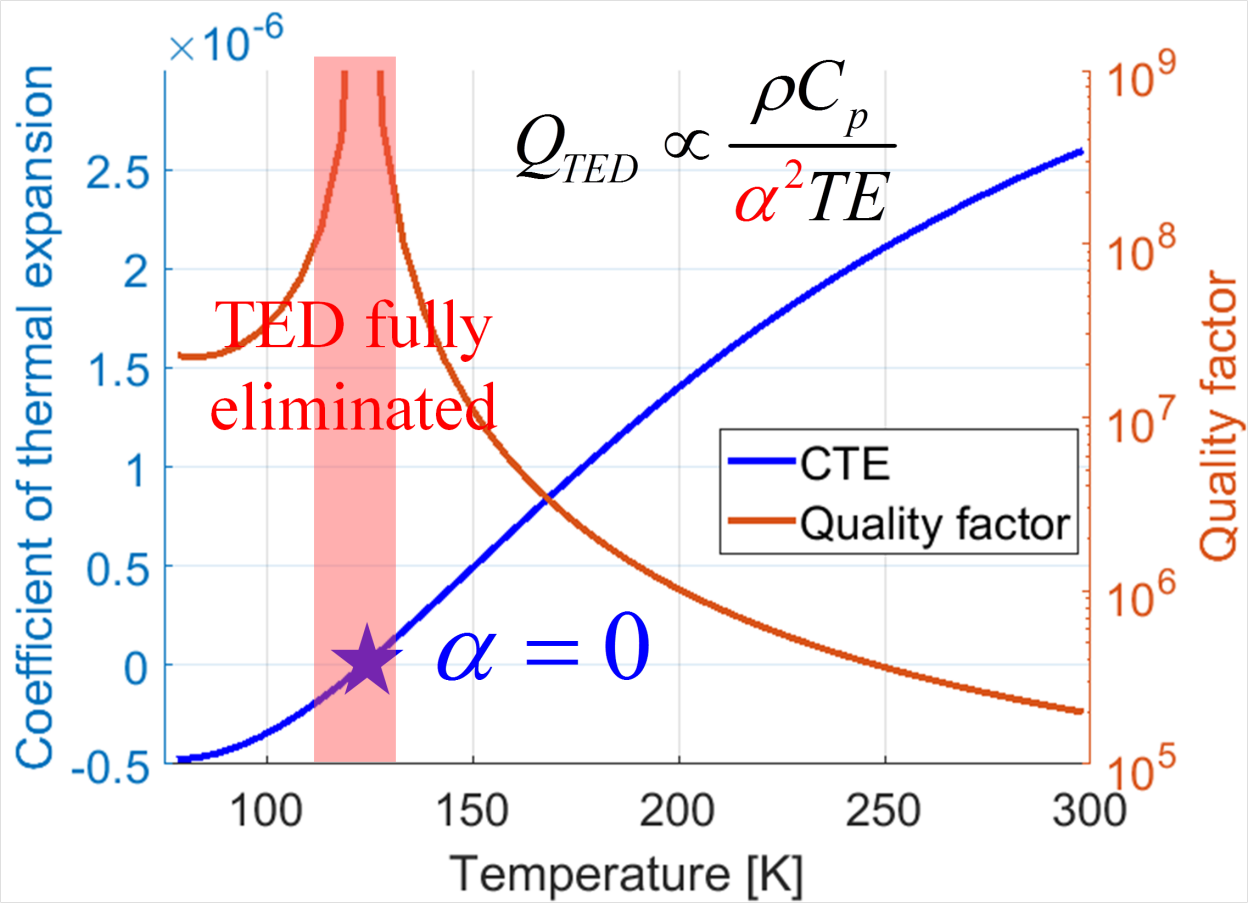

Figure 4.5: Simulation result of Q_{TED} over temperature.

factor for the vented device rose to 1,350,000 at temperature of around 123K, and then decreased again at even lower temperatures. This was the expected signature of a strong contribution from TED. The same signature was present in the data from the sealed device, but the additional contribution from viscous air damping suppressed the signature. At the peak of the $Q - T$ curve for the vented device, we effectively suppressed the contributions from viscous damping and TED, leaving the anchor loss as the most likely remaining source of dissipation. However, this data alone was not sufficient to positively identify the anchor loss as the limiting mechanism at $Q = 1,350,000$.

4.2.3 Anchor Loss

All of the most commonly discussed models for anchor loss depended only on material's mechanical properties, such as Young's modulus, and geometry. In this study, the relevant material properties and geometries were almost invariant over a large temperature range, so we hypothesized that the anchor loss was temperature-invariant. To confirm this hypothesis, the resonant frequency of the device was measured over a large temperature range from 80K to 300K. The results are shown in Fig. 4.6. An overall frequency shift of only 64Hz was demonstrated over a 220K temperature range with the highest resonant frequency at 150K. A frequency shift of $<$ 1300ppm indicated a weak temperature dependence of the mechanical properties of silicon, such as Young's modulus and Poisson's ratio.

In order to confirm that the dissipation mechanism limiting the quality factor to 1,350,000 was due to the anchor loss, we turned to the FEA simulation for anchor loss analysis in COMSOL MultiPhysics Package with the PML model. To model the anchor loss, the device was assumed to be anchored to an semi-infinite elastic half-space, such that all vibratory energy transmitted from the device to the substrate would not be reflected back and thus could be assumed to be lost. In this study, the PML was built in a cylinder shape with

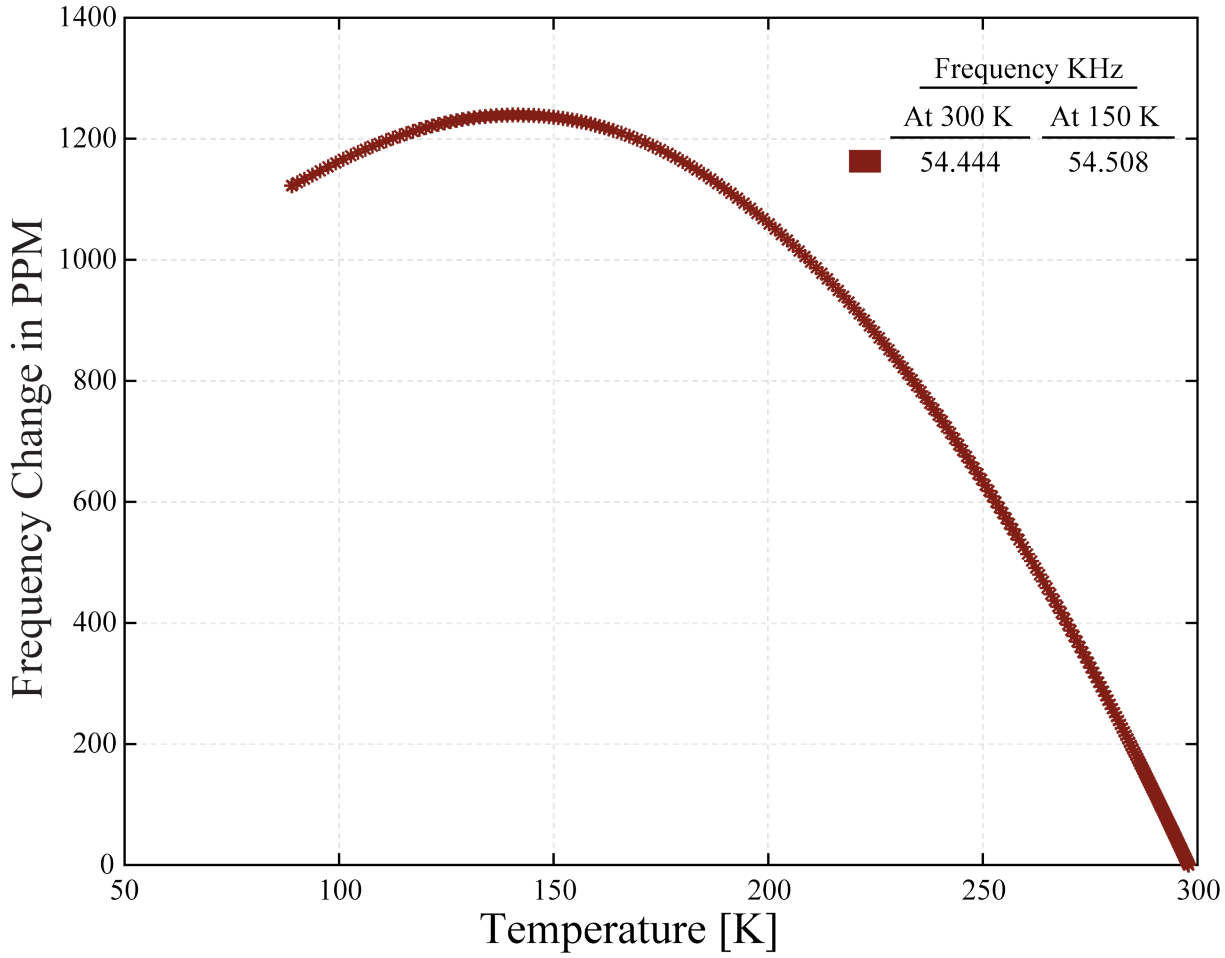

Figure 4.6: The relation between the temperature and the $n = 2$ resonant frequency of the encapsulated device.

the height of 15cm and diameter of 30cm. Even though these dimensions far exceeded the dimensions of the physical die, they were selected to be 20% greater than the wavelength in silicon at 68kHz (0.12m) to guarantee that waves would decay before reaching the PML boundary. The PML model is presented in Fig. 4.7, and the PML is shown by the blue part. Setting the current geometry of PML and the scaling factor of PML to be 1, different meshing densities with number of Degree Of Freedom (DOF) ranging from 0.9 million to 1.4 million were tested to confirm the convergence of the modeling. An averaged quality factor of 1,320,000 was predicted, with meshing errors of less than 5%. This result agreed with the highest quality factor of the device after eliminating the air damping and TED (Fig. 4.4). The close match between the numerical simulation and the experimental results verified that the anchor loss was the dominant energy dissipation mechanism.

Electrostatic tuning of the resonator is necessary to adjust its resonant frequency, and it can be considered as a negative spring applied on the device [53]. Therefore, applying tuning voltage on the resonator will change the balance of the device, and as a result will also change the anchor loss of the device. In this section, we experimentally demonstrated the effect.

In our study, DC bias voltage was applied on the driving electrodes for both driving and tuning of the device. Therefore, driving architecture also determined how tuning voltages would be applied to the device. Two commonly used driving methods were single-sided driving and differential driving. In the single-sided driving architecture, one electrode was used for electrostatic driving of the device, as shown by the highlighted electrode in Fig. 4.8 (a). In the differential driving architecture, two electrodes were used for device actuation in a push-pull way. It was achieved by applying two AC signals of the same magnitude but of opposite phase on two sets electrodes, with one setting for pushing and the other pulling the device. Compared to single-sided driving, differential driving could only actuate the modes where the effects of the two driving electrodes did not cancel each other, and it could generally lead to a more balanced motion. Since the mode of interest in this study was $n = 2$

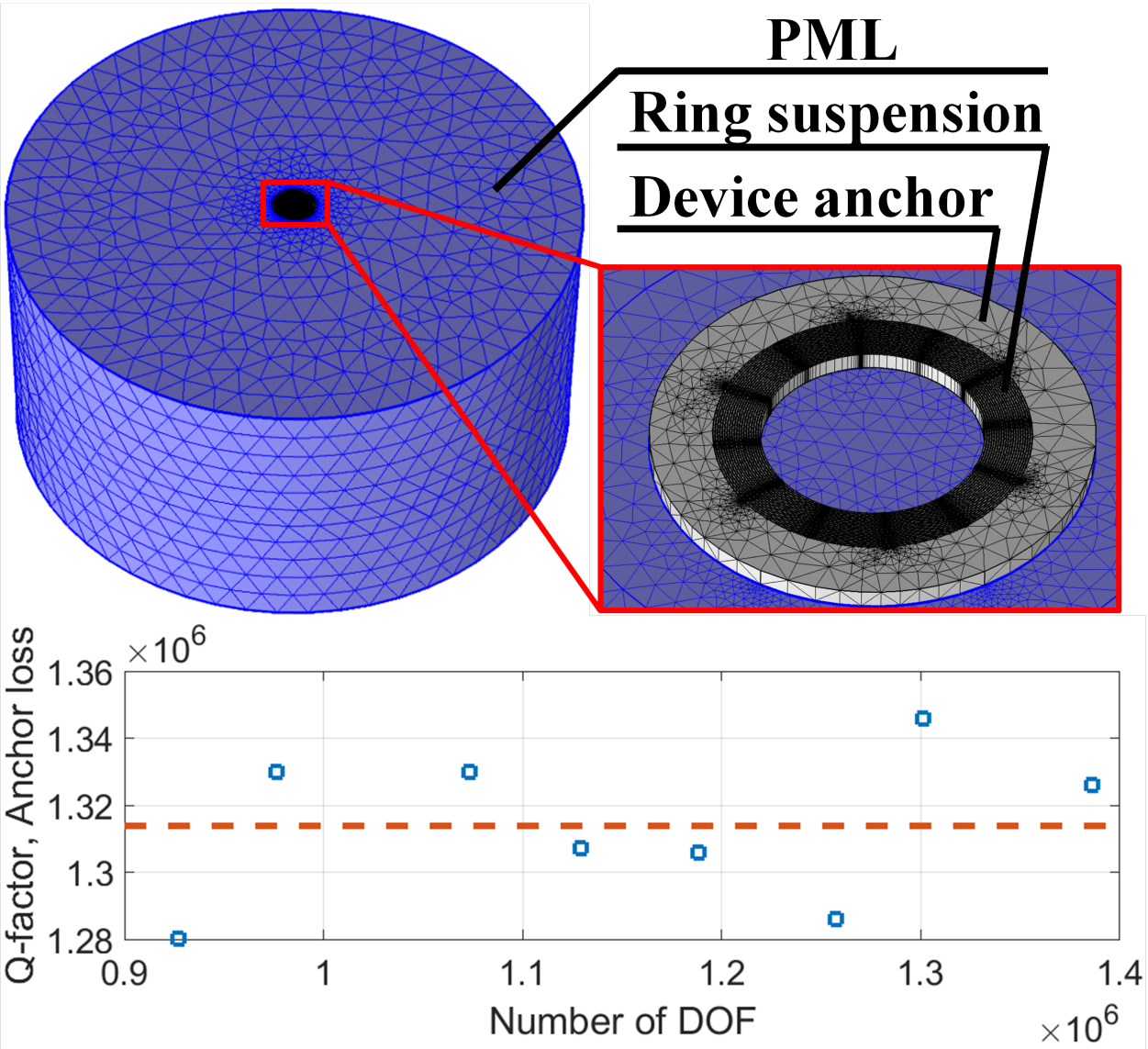

Figure 4.7: Anchor loss model in COMSOL MultiPhysics. The blue part is the PML.

mode, we used two electrodes that were 90° apart from each other, as shown in Fig. 4.8 (b).

To experimentally demonstrate the effect of electrostatic tuning on the anchor loss, it was necessary to make the anchor loss dominant over other energy dissipation mechanisms, such as air damping and TED. Therefore, the device was operated in high vacuum (pressure lower than 0.1mPa) and at a temperature of 123K. The results are presented in Fig. 4.8. For both single-sided driving and differential driving, decrease of the quality factor of TRG was observed as the tuning voltage increased. To fairly compare the effects of electrostatic tuning in two driving/tuning architectures, the relation between the quality factor and the frequency change of the device is shown in Fig. 4.8. A similar trend could be observed for single-sided driving (blue dots in Fig. 4.8) and differential driving (green dots in Fig. 4.8). Quality factor dropped from 1,350,000 to 700,000 as the resonant frequency was tuned by 50Hz. The change of the quality factor due to electrostatic tuning indicated that the anchor loss was the dominant energy dissipation mechanism in the resonator.

In the differential driving architecture for $n = 2$ mode, the two electrodes were placed 90° from each other. Therefore, the constant part of the electrostatic force applied by the electrodes could not cancel each other. As a result, larger tuning voltage would cause greater unbalanced motion, and the anchor loss would be increased. In order to actuate and tune the resonator without affecting the quality factor, we proposed to use the two electrodes that were 180° apart from each other, as shown in Fig. 4.8 (c). In this case, the driving electrodes faced each other, thus the constant part of the electrostatic force would cancel each other, and the anchor loss would not be reduced due to the extra unbalance caused by tuning. The experimental results are presented by the red dots in Fig. 4.8. The overall quality factor remained almost constant with large tuning voltage applied. The drop in quality factor was probably due to the non-linearity caused by the high voltage applied (about 10V).

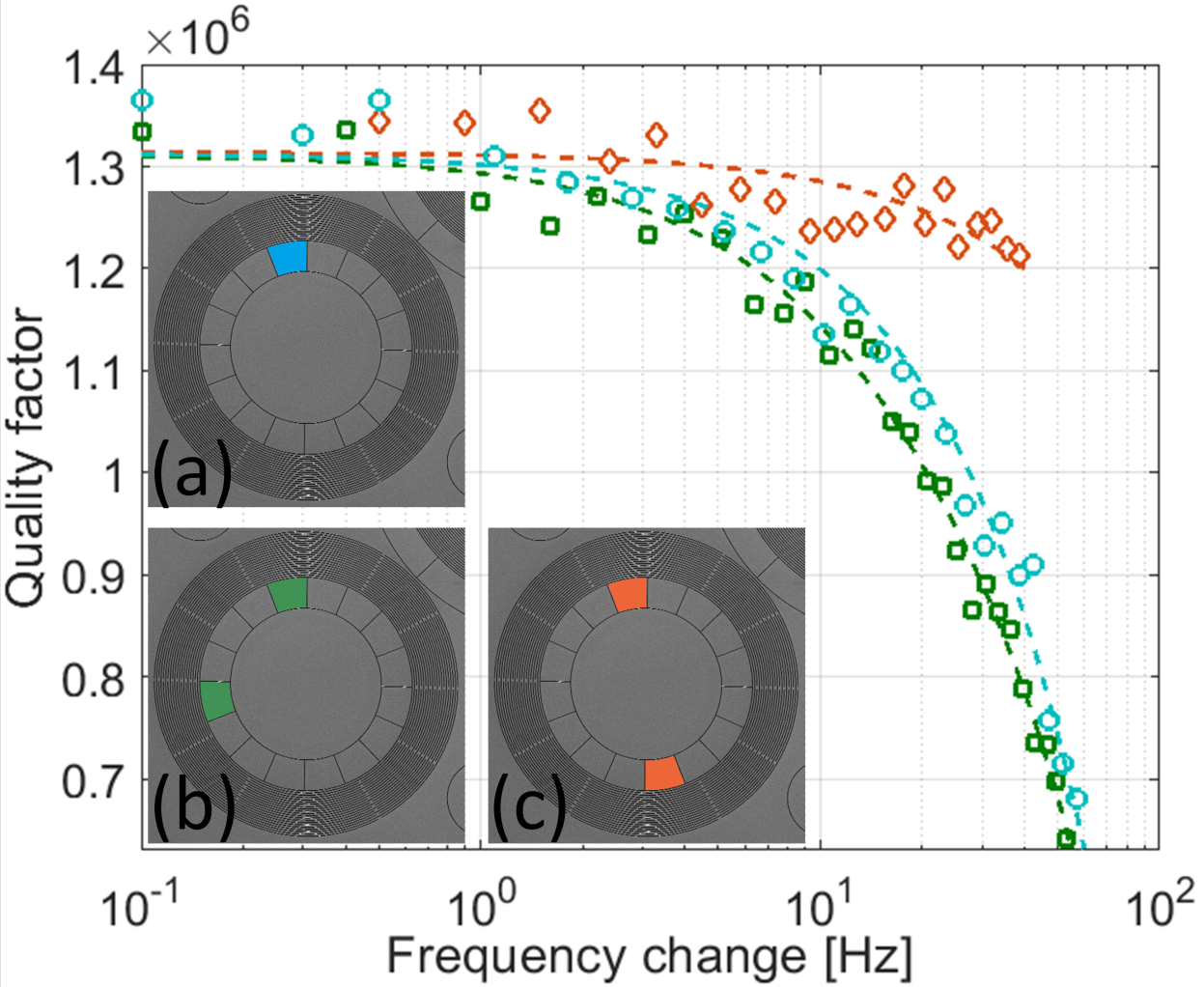

Figure 4.8: Relation between the quality factor of TRG and frequency change due to electrostatic tuning under different driving architectures, which are shown by the insets. Different colors correspond to different driving architectures. The data points are experimental data, and the dashed lines are the fitting curves.

4.2.4 Surface Loss

Due to the fabrication process of epi-seal devices, the as-fabricated surface loss has been minimized. However, due to the venting of the device cavity in out study, the device was exposed to the external environment. A decrease of the quality factor was observed after FIB drilling of the cap layer. The quality factor was reduced from 1,350,000 to 620,000 at the temperature of 123K. The results are presented in Fig. 4.9. Since other energy dissipation mechanisms, such as air damping, TED, the anchor loss, have not changed over the two years, the drop in the overall quality factor was contributed to the surface loss. Changes of the surface quality were not possible, since the temperature was not high enough to change the surface topology, and there was no obvious frequency change observed. Contamination on the surface was also unlikely, because the size of the drilling hole was too small (about 10μm) for dusts to get through. The only possible reason was the moisture on the surface of the device.

To verify the assumption and remove the moisture, the device was placed in a bake-out furnace for high-temperature bake-out in vacuum. After 12 hours of bake-out at 220 °C, the overall quality factor was increased to 860,000 at the temperature of 123K (shown by the green square dots in Fig. 4.9). After another 12-hour bake-out at a higher temperature of 425 °C, the overall quality factor was further increased to 1,090,000 at the temperature of 123K (yellow square dots in Fig. 4.9). Finally, after another 30-hour bake-out at 425 °C, the quality factor was restored to the original level of 1,350,000 at 123K (red square dots in Fig. 4.9), indicating that the effects of surface loss were completely eliminated. The quality factors of the device due to surface loss at different conditions were calculated and the results are summarized in Table 4.1.

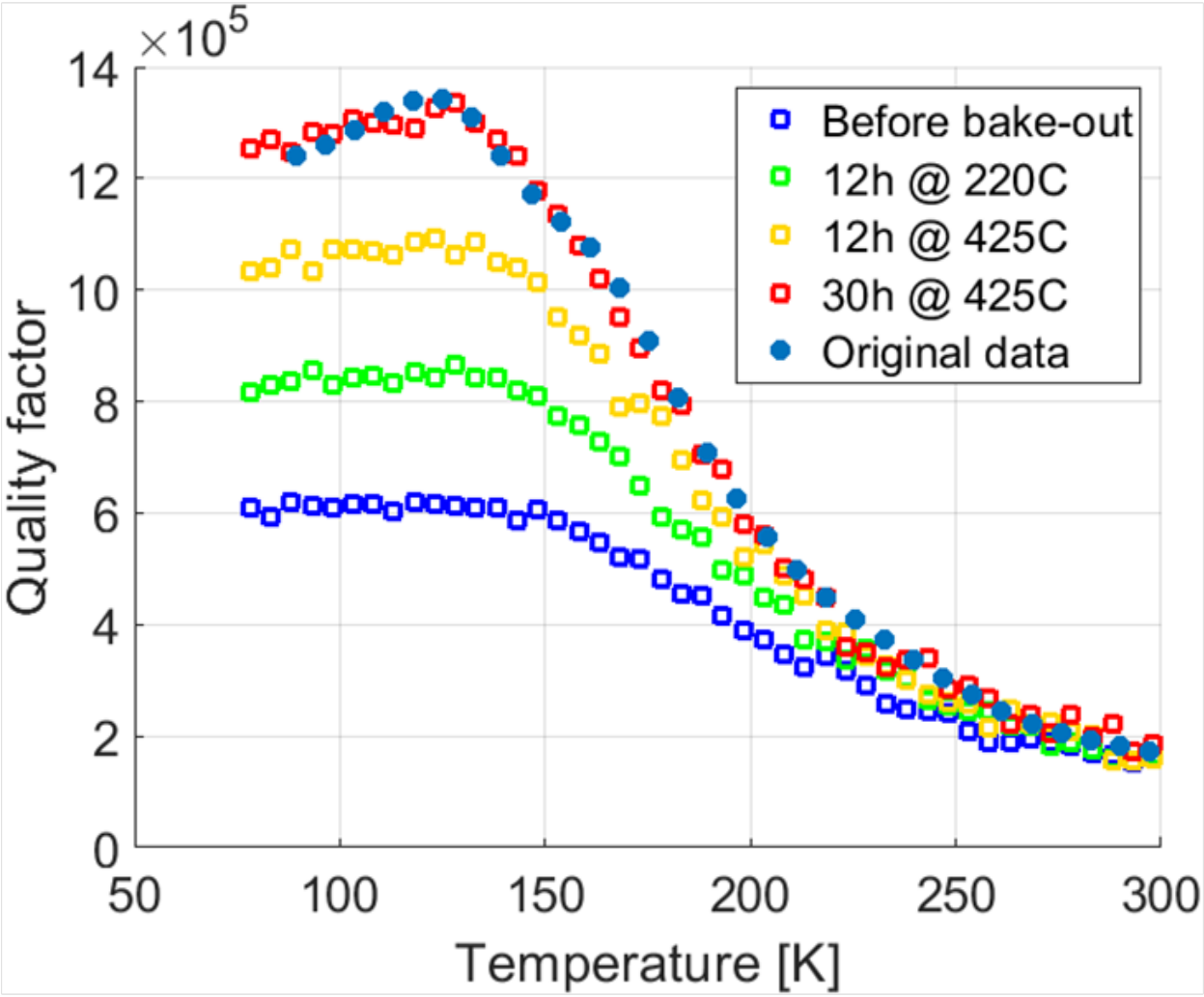

Figure 4.9: Relation between the quality factor of TRG and operating temperature under different conditions, showing the effects of surface loss on the quality factor.

Table 4.1: Summary of the quality factor of TRG due to surface loss under different conditions

Device Condition	Q_{surface}
Before venting the cavity	N/A
2 years after venting	1,146,000
12h bake-out @ 220 °C	2,369,000
12h bake-out @ 425 °C	5,660,000
30h bake-out @ 425 °C	N/A

4.3 Results Discussion

In Section 4.2, we decoupled and analyzed some of the major energy dissipation mechanisms in TRG, and in this section, we quantify and compare the effects of each of them.

The quality factors of TRG due to major energy dissipation mechanisms (viscous air damping, TED, and anchor loss) at room temperature and low temperature (123K) are listed in Table 4.2. Table 4.2 clearly shows the contribution of each energy dissipation mechanism to the overall quality factor of the device. To the best of our knowledge, this study is the first to experimentally quantify all major energy dissipation mechanisms in a MEMS resonator. Note the dominant energy dissipation may change under different conditions even for the same device. For example, before venting the cavity of the device, the overall quality factor of the device was dominated by TED at room temperature. After cooling down the temperature to 123K, the dominant energy dissipation became viscous air damping due to the elimination of TED. Then, the cavity was vented to operate the device in a better vacuum, and the anchor loss became dominant. However, after two years of venting the cavity, because of the introduction of the moisture related surface loss, the quality factor dropped and it was dominated by both the anchor loss and the surface loss. Finally, we demonstrated that high-temperature bake-out of the device could eliminate the moisture, and anchor loss became the only dominant energy dissipation mechanism again.

Table 4.2: Summary of the quality factor of TRG due to major energy dissipation mechanisms at different temperature

T	300K	123K
Q_{air}	625,000	978,000
Q_{TED}	200,000	Eliminated due to zero CTE
Q_{anchor}	1,350,000	1,350,000
Q_{overall}	128,000	567,000

Since the major energy dissipation mechanism may change even for the same device, it is necessary to identify the dominant one and optimize the design with regard to it, in order to

effectively improve the overall quality factor of the device. For example, in the case of TRG operated at room temperature, it will not help to improve the design for a lower anchor loss while keeping TED at the same level.

4.4 Conclusions

In this chapter, a methodology for analysis of energy dissipation mechanisms in MEMS devices and the possibility of quantifying the contribution of each mechanism independently were developed and demonstrated. The effects of viscous air damping, TED, anchor loss, and surface loss on the overall quality factor of the device were studied and experimentally estimated, and their magnitudes at room temperature were measured to be 625,000, 200,000, 1,350,000, and 1,146,000, respectively. Relation between the anchor loss and electrostatic tuning was also explored. We experimentally demonstrated that unbalanced electrostatic tuning would increase the anchor loss, while the balanced tuning would not affect the anchor loss. High-temperature bake-out process was developed to eliminate the moisture-related surface loss.

This work provides more insight into energy dissipation of MEMS devices by thoroughly decoupling and quantifying different dissipation mechanisms, which is necessary to better understand the dominant mechanism that limits the quality factor of MEMS devices, in order to effectively improve the design to achieve higher device performances.

Chapter 5

Algorithm Development — Navigation Error Estimation

In this chapter, an analytical model relating the IMU errors and navigation errors in the ZUPT-aided pedestrian inertial navigation is developed. In Section 5.1, a 2D bio-mechanical model to simulate human gait is presented to better understand human walking dynamics and also serves as the basis for the following numerical simulations. Next, Section 5.2 introduces the implementation of the Extended Kalman Filter (EKF) in the algorithm. In Section 5.3, derivation of the model is explained in detail, and the analytical results are explicitly presented. The following are the numerical and experimental verification of the model in Section 5.4. Finally, the chapter is concluded with Section 5.5.

5.1 Human Gait Bio-Mechanical Model

In this section, we develop an approach for generation of the foot trajectory. Such bio-mechanical models are necessary for analytical prediction of errors in ZUPT-aided pedestrian

navigation.

Human ambulatory gait models are multi-dimensional due to the complex kinematic and dynamic relations between many parts of human body involved during walking. In this study, our focus was only on the trajectories of two feet instead of the whole-body motion. Therefore, a few assumptions were used to simplify the human gait model:

1. The motion of each leg is two-dimensional and parallel to each other, indicating no rotation occurs at the pelvis and no horizontal rotation occurs at the ankles;
2. The dimensions of both legs are identical;
3. The pattern and duration of each step are identical;
4. The floor is flat, resulting in no accumulation of altitude changes during walking;
5. The trajectory is straight; no turning or stopping happens during the navigation.

In the following parts of this section, we first extracted the foot motion from the joints rotation in the torso coordinate frame. Next, based on the human gait analysis, the expression of the foot motion was transferred from the torso frame to the navigation frame. Finally, a parameterization was applied to generate a new trajectory with higher order of continuity and while preserving all the key characteristics of the foot motion.

5.1.1 Foot Motion in Torso Frame

The torso frame is a coordinate frame that is fixed to the body trunk. In the torso frame, only the relative motion with respect to the trunk is studied.

Joint movement has been widely studied for pathological purposes and the angle data are typically extracted by a high-speed camera or wearable sensors [164]. A pattern of joint

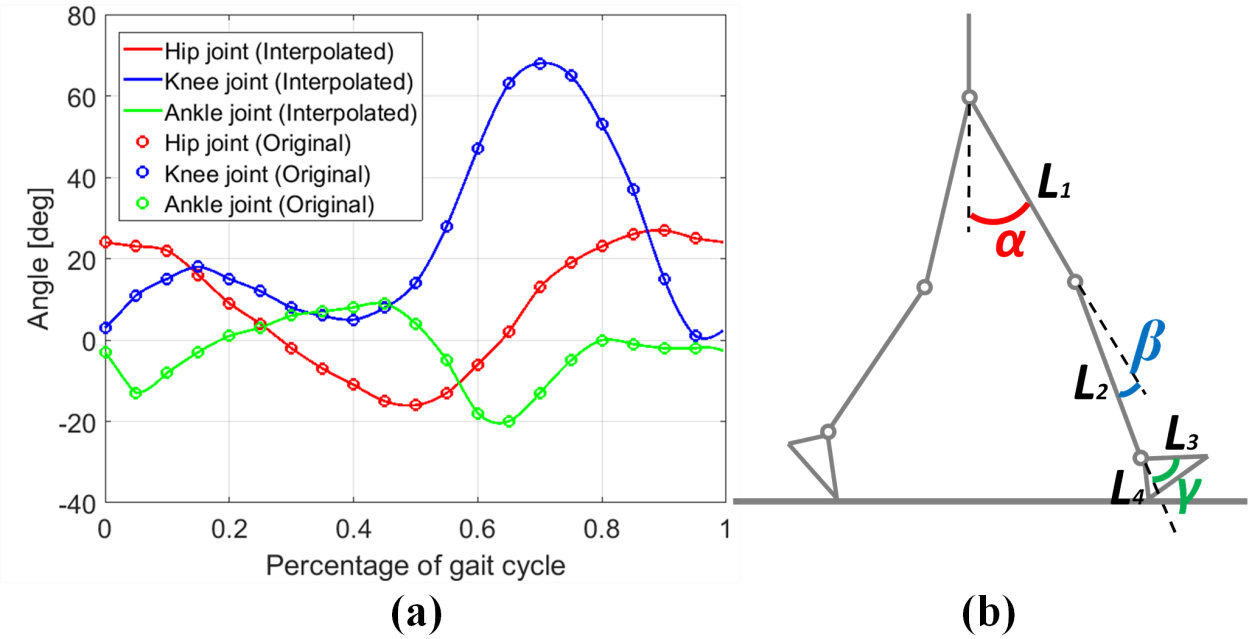

Figure 5.1: (a) Interpolation of joint movement data (left) and (b) simplified human leg model (right).

angle changes was reported in [165] and was reproduced in Fig. 5.1(a). A simplified human leg model is shown in Fig. 5.1(b). The leg was modeled as two bars with femur length of 50cm and tibia length of 45cm. The foot was modeled as a triangle with the side lengths of 4cm, 13cm, and 16cm, respectively. The parameters were determined by a typical male subject with a height of 180cm. Position of the forefoot in the torso frame was expressed as

$$x_{\text{forefoot}} = L_1 \sin\alpha + L_2 \sin(\alpha - \beta) + L_3 \sin(\alpha - \beta + \gamma), \tag{5.1}$$

$$y_{\text{forefoot}} = L_1 \cos\alpha + L_2 \cos(\alpha - \beta) + L_3 \cos(\alpha - \beta + \gamma). \tag{5.2}$$

The corresponding parameters are shown in Fig 5.1(b). The position of another foot could be calculated by shifting the time by half of a cycle since we assumed that every step was identical.

5.1.2 Foot Motion in Navigation Frame

The navigation frame is the coordinate frame that is fixed on the ground with axes pointing to the north, east, and down directions, respectively. In this frame, the motion of foot with respect to the ground was first studied.

To transfer a foot motion from the torso frame to the navigation frame, the gait analysis was necessary to establish stationary points as a reference in different phases of the gait cycle. Each gait cycle was divided in two phases: stance and swing. The stance phase is a period during which the foot is on the ground. The swing phase is a period when the foot is in the air for the limb advancement [166].

We assumed that each gait cycle began when the left heel contacted the ground (heel strike). During the first 15% of the gait cycle, the left heel was assumed to be stationary and the foot rotated around it (heel rocker) until the whole foot touched the ground. During 15% to 40% of the gait cycle, the whole left foot was on the ground and stationary, and the left ankle joint rotated for limb advancement (ankle rocker). This was also the time period when zero-velocity update was applied as pseudo-measurements to the EKF. For 40% to 60% of the gait cycle, the left heel began to rise and this stage ended when the left foot was off the ground. In this stage, the left foot rotated with respect to the forefoot, which we assumed to be stationary (forefoot rocker) [167]. The following part of the gait cycle was symmetric to the previous part since we assumed that every step was identical. Phases of the gait cycle are presented in Fig. 5.2.

After establishing different stationary points in different phases of the gait cycle, we could extract position of the body trunk with respect to the ground. The foot motion could be superimposed on top of the torso motion to obtain the foot motion in the navigation frame.

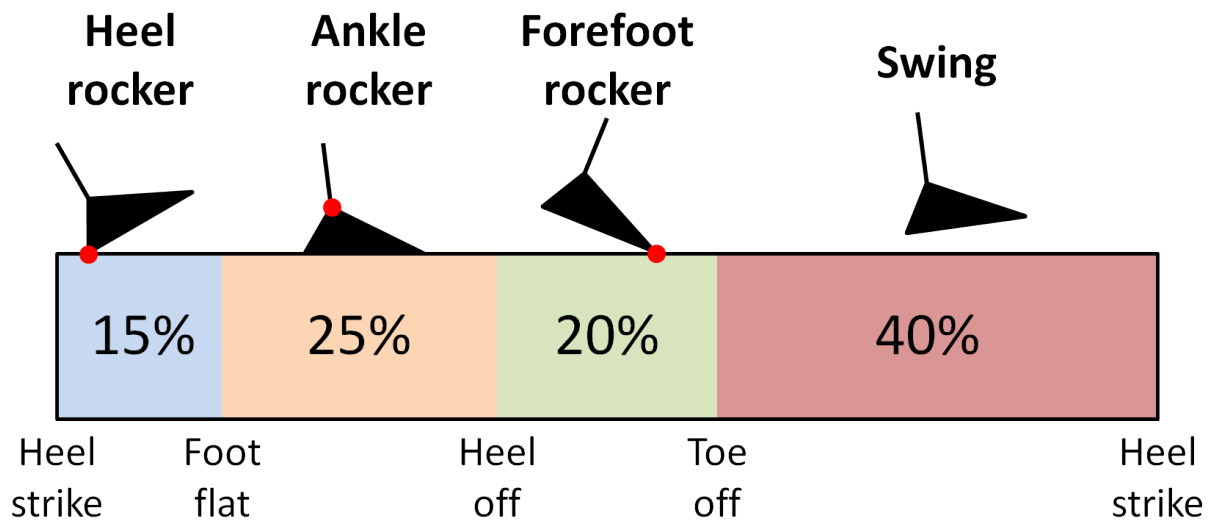

Figure 5.2: Human ambulatory gait analysis. Red dots are the stationary points in different phases of one gait cycle.

5.1.3 Parameterization of Trajectory

Abrupt changes of the reference point from the heel to the ankle and to the forefoot would create discontinuity in the trajectory, especially in terms of velocity and acceleration, as depicted in Fig. 5.3. The discontinuities of acceleration resulted in discontinuous accelerometer readouts, which would cause numerical problems in the algorithm. Therefore, parameterization was needed to generate a new trajectory with a higher order of continuity.

The new trajectory to be generated did not have to strictly follow the angle data for each joint and the linkage relations, but ambulatory characteristics should be preserved, especially zero velocity and angular rate during time period of the ankle rocker.

The velocity along the trajectory was parameterized to guarantee the continuity of both displacement and acceleration. Key points were first selected to characterize the IMU velocity along the horizontal and vertical directions. For parameterization along the vertical direction, the integral of velocity for a gait cycle should be zero to make sure the altitude did not change over one gait cycle. This was achieved by adjusting velocity values at some of the key points.

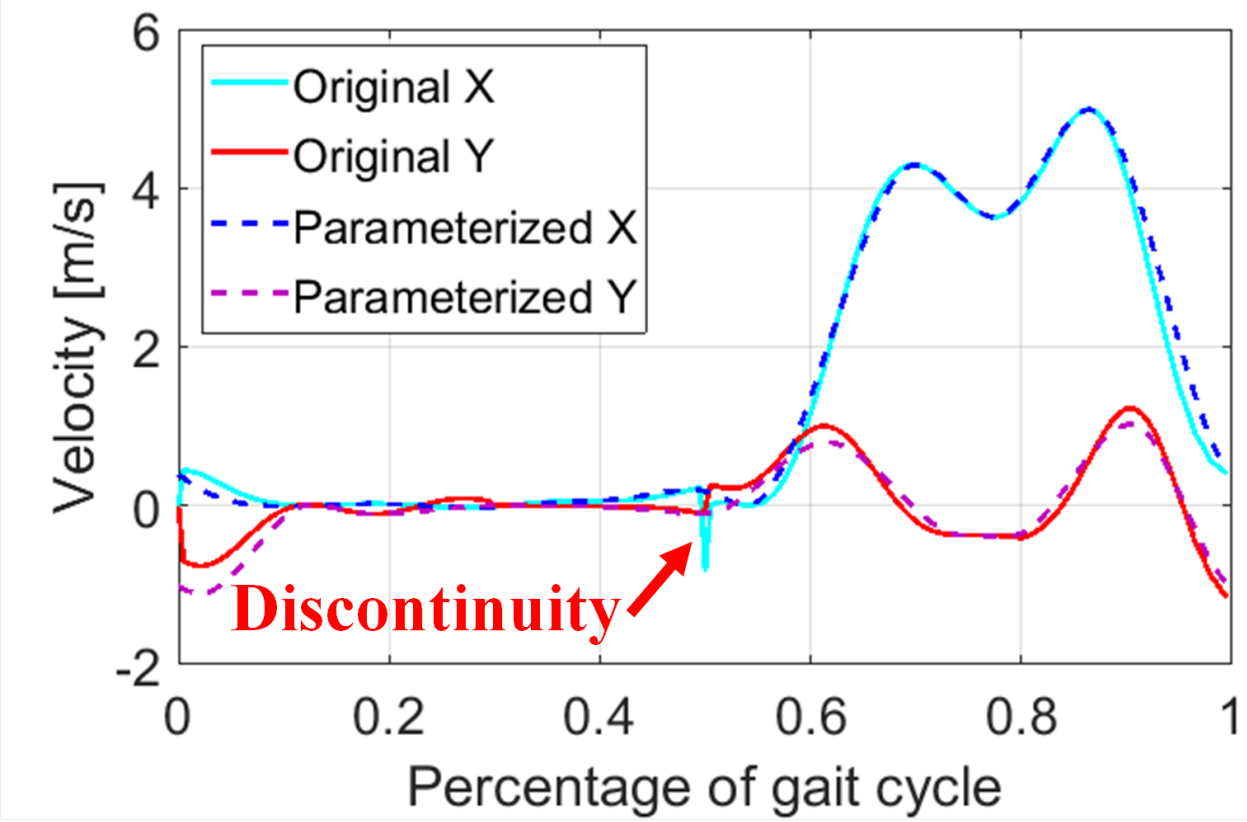

Figure 5.3: Velocity of the parameterized trajectory. A close match was demonstrated and discontinuities were eliminated.

The parameterization results are shown in Fig. 5.3. The generated velocities (dashed lines) closely followed the original values without losing any characteristics and also eliminated the discontinuity that otherwise would happen in the middle (50%) of the gait cycle, corresponding to a shift of the reference point from the left forefoot to the right heel. The trajectory in terms of position was obtained by integrating the velocity and the results are shown in Fig. 5.4. A close match was demonstrated for displacement along the horizontal direction. For displacement along the vertical direction, the difference was purposefully introduced to guarantee that the altitude of foot did not change after one gait cycle.

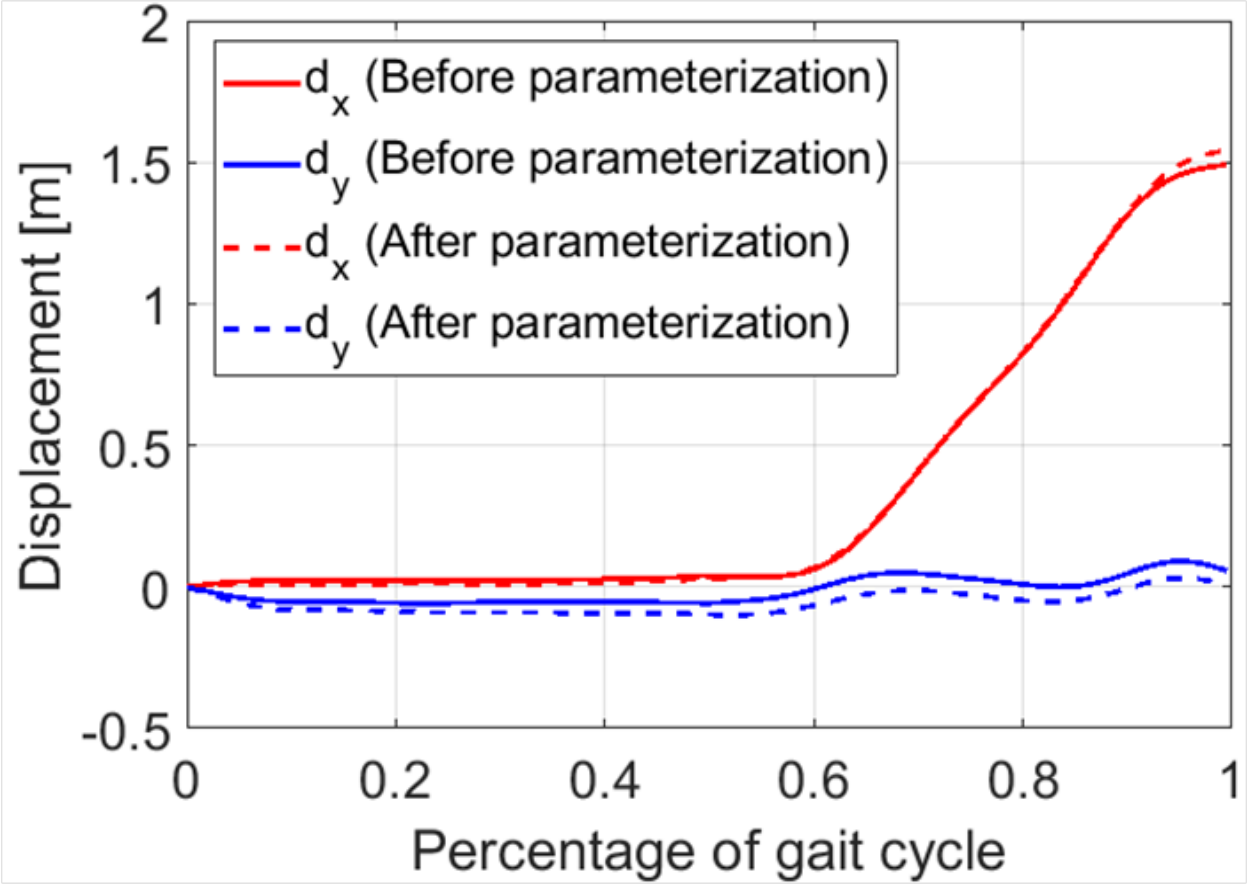

Figure 5.4: Displacement of the parameterized trajectory. A close match was demonstrated for displacement along the x direction (horizontal). Difference between the displacements along y direction (vertical) was to guarantee the displacement continuity in between gait cycles.

5.2 Implementation of ZUPT-Aided Algorithm

The EKF implementation in the ZUPT-aided pedestrian inertial navigation has been briefly introduced in Section 1.2.1.4 and Section 1.2.3.2, and more details will be presented in this section in order to conduct further analysis in the following sections.

The ZUPT-aided navigation algorithm takes advantage of the stance phase in human gait cycle to compensate for the IMU drifts. The effects of the aiding on navigation are demonstrated in Fig. 5.5. Due to stochastic and deterministic errors of the IMU, velocity of the foot will drift without usage of the ZUPT aiding, as presented by the blue line. However,

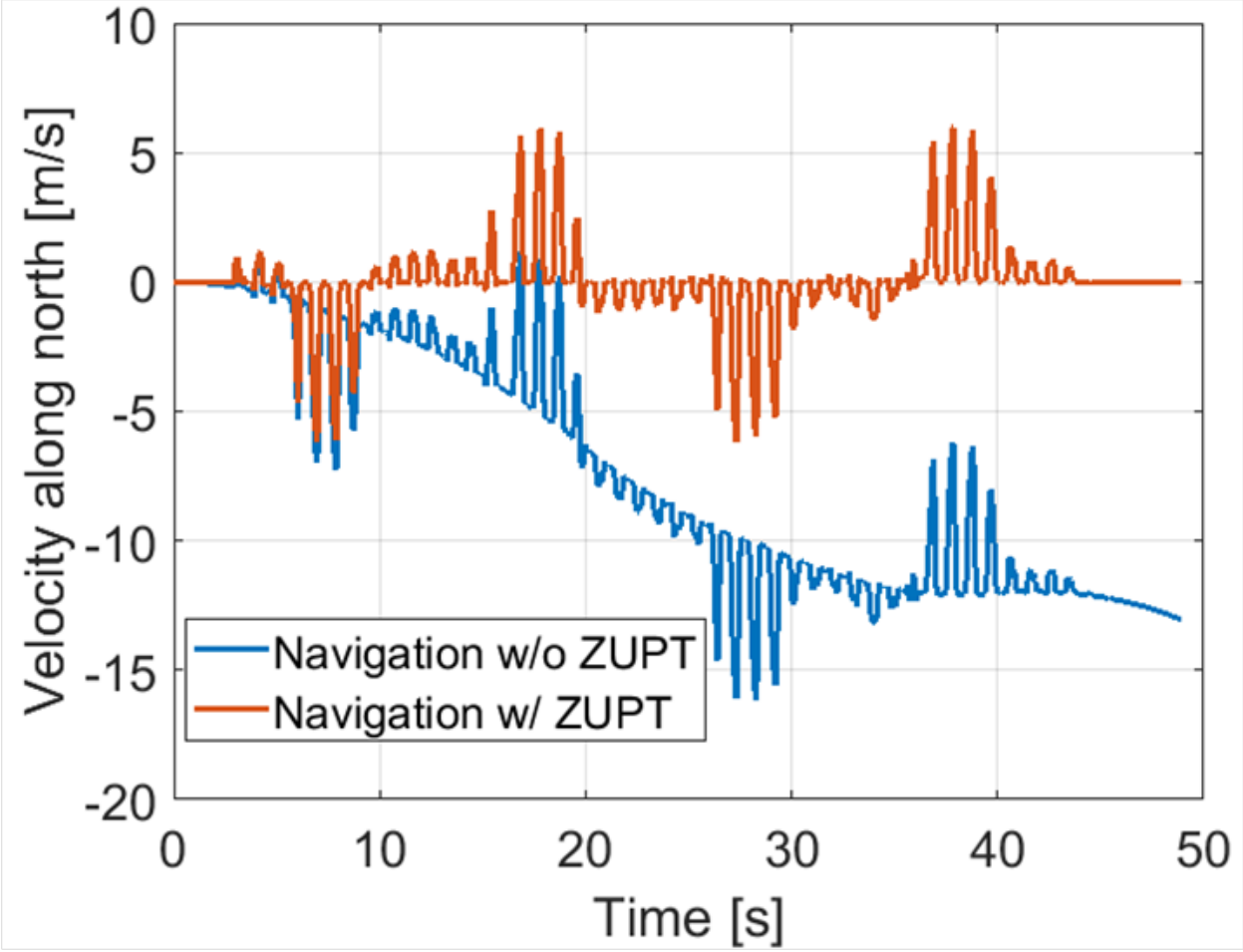

Figure 5.5: The comparison between velocity estimations with and without ZUPT-aided navigation algorithm. Changes in sign correspond to changes in direction of motion.

the ZUPT-aided inertial navigation algorithm helped to set the velocity of the foot close to zero during the stance phase and greatly reduced the effects of the IMU biases.

A standard strapdown Inertial Navigation System (INS) mechanization in the navigation frame was implemented, and the details are introduced in Section 1.2.1.2. Drift correction was performed by implementing the EKF operating on the error states

$$\delta\vec{x} = [\delta\vec{\theta}^{\mathrm{T}}, \delta\vec{v}_n^{\mathrm{T}}, \delta\vec{s}_n^{\mathrm{T}}, \delta\vec{x}_g^{\mathrm{T}}, \delta\vec{x}_a^{\mathrm{T}}]^{\mathrm{T}}, \tag{5.3}$$

where $\delta\vec{\theta}$ is the three-axis attitude error in the navigation frame, $\delta\vec{v}_n$ and $\delta\vec{s}_n$ are the vectors of velocity and position errors along the North, East, and Down directions of the navigation

coordinate frame, $\delta\vec{x}_g$ is the gyroscope states (12-element vector) modeling gyroscope bias, scale factor error, rotational and non-orthognality misalignment, and $\delta\vec{x}_a$ is the accelerometer states (9-element vector) modeling accelerometer bias, scale factor error and non-orthognality misalignment [168]. A full dynamic error model could be approximated by

$$\delta\dot{\vec{x}} = \begin{bmatrix} -[\vec{\omega}_i^n\times] & F_{\delta v}^{\delta\dot{\theta}} & 0 & -F_g & 0 \\ [\vec{f}^n\times] & C_1 & C_2 & 0 & F_a \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \delta\vec{x} + \begin{bmatrix} C_{s_g}^n \cdot \epsilon_{\text{ARW}} \\ C_{s_a}^n \cdot \epsilon_{\text{VRW}} \\ 0 \\ \epsilon_{b_g} \\ \epsilon_{b_a} \end{bmatrix}, \tag{5.4}$$

where $[\vec{\omega}_i^n\times]$ and $[\vec{f}^n\times]$ are the skew-symmetric cross-product-operators of angular rate of the navigation frame relative to the inertial frame expressed in the navigation frame, and of accelerometer output in the navigation frame, respectively. I is the identity matrix. $F_{\delta v}^{\delta\dot{\theta}}$ is the term related to transport rate, C_1 and C_2 are the terms related to the Coriolis effects due to the Earth rotation and the transport rate, $C_{s_g}^n$ and $C_{s_a}^n$ are the Direction Cosine Matrices (DCM) from the navigation frame to the coordinate frames of accelerometers and gyroscopes, respectively. F_g and F_a are matrices (3 by 12) and (3 by 9) modeling the linearized dynamics of the states $\delta\vec{x}_g$ and $\delta\vec{x}_a$, respectively, ϵ_{ARW} is the Angle Random Walk of the gyroscopes and ϵ_{VRW} is the Velocity Random Walk of the accelerometers. ϵ_{b_g} and ϵ_{b_a} are noise terms in the 1st order Markov process in the modeling of the states $\delta\vec{x}_g$ and $\delta\vec{x}_a$, respectively [169].

For a typical IMU, scale factor errors and misalignments vary slowly during a single pedestrian inertial navigation process and can be approximated by bias errors. We assumed that the calibration process before the navigation was able to remove all deterministic biases. Therefore, in this work, only stochastic bias errors and white noises (ARW for gyroscopes and VRW for accelerometers) were considered. The Earth rotation and the transport rate were neglected in this study. The simplification of the error model yielded a shorter system

state

$$\delta\vec{x} = [\delta\vec{\theta}^T, \delta\vec{v}_n^T, \delta\vec{s}_n^T, \delta\vec{b}_g^T, \delta\vec{b}_a^T]^T, \tag{5.5}$$

where $\delta\vec{b}_g$ is the bias of three gyroscopes and $\delta\vec{b}_g$ is the bias of three accelerometers. Note that biases were not systematic errors. With these corrections, the dynamic error model became

$$\delta\dot{\vec{x}} = \begin{bmatrix} 0 & 0 & 0 & -C_b^n & 0 \\ [\vec{f}^n\times] & 0 & 0 & 0 & C_b^n \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \delta\vec{x} + \begin{bmatrix} C_b^n \cdot \epsilon_{\text{ARW}} \\ C_b^n \cdot \epsilon_{\text{VRW}} \\ 0 \\ \epsilon_{\text{RRW}} \\ \epsilon_{\text{AcRW}} \end{bmatrix} \triangleq A\delta\vec{x} + B, \tag{5.6}$$

where C_b^n is the DCM from the navigation frame to the body frame, which was assumed to be aligned with the sensor frame, ϵ_{RRW} is Rate Random Walk (RRW) of the gyroscopes, and ϵ_{AcRW} is Accelerometer Random Walk of the accelerometers.

For each time step, predict step in the EKF was necessary: besides calculating the system states (position, velocity, and attitude) in the standard strapdown navigation algorithm, *a priori* error covariance was propagated using (1.20)

$$P_{k+1|k} = F_k P_{k|k} F_k^T + Q_k, \tag{5.7}$$

where F and Q are defined as

$$F = exp\{A \cdot \Delta t\} \approx I + A \cdot \Delta t,$$
$$Q = Var\{BB^{\text{T}}\} \cdot \Delta t,$$

where Δt is the time duration of each accumulation in (5.7), and B is the process noise defined in (5.6). In the discrete form, the system state update could be expressed as

$$\delta\vec{x}_{k+1|k} = F_k \cdot \delta\vec{x}_{k|k}.$$

To activate the update step of the EKF, a zero-velocity detector was needed to detect the stance phase in each gait cycle. Standard Stance Hypothesis Optimal dEtector (SHOE) was applied. A proper combination of the time interval for the detector W, and the threshold T was needed to minimize the probability of false detection at the cost of some miss-detection of the zero-velocity events. In this study, W was set to be 5 and T was set to be 3×10^4. The stance phase was detected if the variance was lower than the threshold value

$$\text{ZUPT} = H\{\frac{1}{W}\sum_{k=1}^{W}(\frac{||\ \boldsymbol{y}_k^a - g \cdot \bar{\boldsymbol{y}_{\boldsymbol{n}}^{\boldsymbol{a}}}\ ||^2}{\sigma_a^2} + \frac{||\ \boldsymbol{y}_k^\omega\ ||^2}{\sigma_\omega^2}) - T\},$$

where ZUPT is the logical indicator of the detector, $H\{\cdot\}$ is the Heaviside step function, $\boldsymbol{y}_k^a$ and $\boldsymbol{y}_k^\omega$ are accelerometer and gyroscope readouts at time step k, respectively, $\bar{\boldsymbol{y}_{\boldsymbol{n}}^{\boldsymbol{a}}}$ is the normalized and averaged accelerometer readout within the measurement window, σ_a and σ_ω are the white noise level of the accelerometers and gyroscopes of the IMU, respectively.

When the stance phase was detected, zero-velocity update was applied as pseudo-measurements, and velocity in the system state was considered as the measurement residual $\boldsymbol{v}_k$ to update the state estimation

$$\vec{v}_k = \begin{bmatrix} 0 & I_{3\times3} & 0 & 0 & 0 \end{bmatrix} \cdot \delta\vec{x}_k + \vec{w}_k \triangleq H \cdot \delta\vec{x}_k + \vec{w}_k,$$

where H is called the observation matrix, and $\vec{w}_k$ is the measurement noise, which was mainly due to non-zero velocity of the IMU during the stance phase [170]. The covariance of $\vec{w}_k$ was denoted by R_k. In most studies, $\vec{w}_k$ was assumed to be white with constant and

isotropic standard deviation r, which was generally set in the range from 0.001m/s to 0.1m/s [36, 118, 170, 171]. The value r was also called velocity uncertainty. Therefore, the noise covariance matrix could be expressed as $R_k = r^2 I_{3\times 3}$. In this study, we set the velocity uncertainty to be 0.02m/s unless otherwise stated. This value matched the experimental data reported in [170], and more details are presented in Section 6.3.3.

After the EKF received the measurement information, it updated the system state with

$$\delta\vec{x}_{k+1|k+1} = \delta\vec{x}_{k+1|k} + K_{k+1}\vec{v}_{k+1}, \tag{5.8}$$

$$K_{k+1} = P_{k+1|k}H_{k+1}^T(H_{k+1}P_{k+1|k}H_{k+1}^T + R_{k+1})^{-1}, \tag{5.9}$$

$$P_{k+1|k+1} = (I - K_{k+1}H_{k+1})P_{k+1|k}. \tag{5.10}$$

5.3 Navigation Error Analysis

The navigation errors in the ZUPT-aided navigation algorithm came mainly from two major sources: systematic modeling errors and IMU noises. We emphasized that in this section we only analyzed the navigation errors caused by IMU noises. In this section, we quantitatively analyzed the navigation errors in terms of angle, velocity, and position.

5.3.1 Starting Point

A typical propagation of the error in attitude estimation in ZUPT-aided pedestrian inertial navigation and its covariance are presented in Fig. 5.6. A similar phenomenon could be observed for the velocity error propagation as well. A few conclusions could be drawn from propagation of errors in attitude estimation:

1. Although the propagated error was random due to the stochastic nature of noise (red

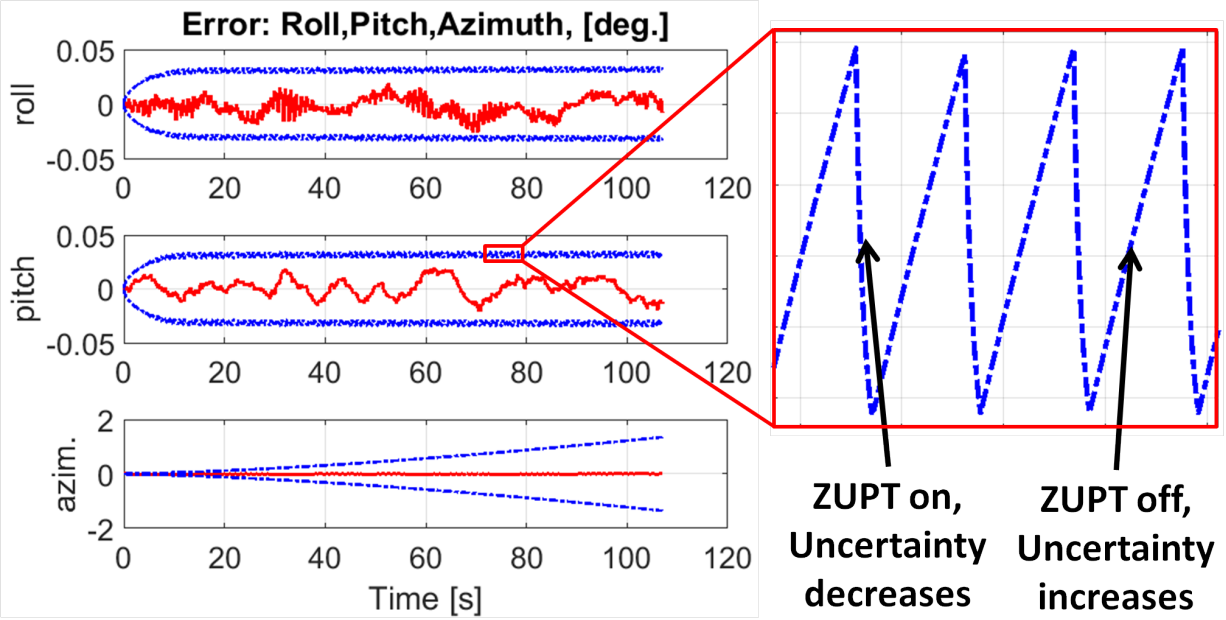

Figure 5.6: A typical propagation of errors in attitude estimations in ZUPT-aided pedestrian inertial navigation. The red solid lines are the actual estimation errors, and the blue dashed lines are the 3σ uncertainty of estimation. Azimuth angle (heading) was the only important EKF state that was not observable from zero-velocity measurements.

solid lines in Fig. 5.6), the error covariances (bounds) followed a pattern (blue dashed lines in Fig. 5.6);

2. For roll and pitch angles, the covariance reached a constant and the same level with some fluctuations, but covariance of azimuth angle propagated as $t^{1.5}$ due to RRW and lack of observability in azimuth angle [172];

3. The covariances were reduced if the update step was activated during the stance phase, and they were increased in the prediction step during the swing phase.

A starting point of the analysis was an observation that covariances of attitude and velocity reached a stable level with some fluctuation in the long run of the ZUPT-aided navigation algorithm [173], as shown by the blue dashed lines in Fig. 5.6. This observation indicated that in a whole gait cycle, the amount of the covariance increase in the prediction step was equal to the amount of the covariance decrease in the update step. Following this observation,

we combined the parameters related to ZUPT and the IMU parameters to estimate the overall navigation solution uncertainty. This combination enabled us to fully analyze the system behavior and extract the covariance of the error in the system state estimation. For simplicity of derivation, we assumed in the analysis a straight line trajectory toward the North. We also assumed a 2D motion of the foot: the foot only moved along the North and Down directions, and roll and azimuth angles were zero. In case of any other trajectory shapes, the analytical expression for the navigation error might be different, but the general conclusions would still hold.

5.3.2 Covariance Increase During Swing Phase

Covariance increased during the swing phase due to the noise in IMU readouts. The *a priori* covariance propagated according to (5.7). To differentiate the errors terms in position, velocity, and orientation, we expanded (5.7) into 3×3 sub-blocks, suppressed subscripts indicating the time steps for simplicity, and used new subscripts to indicate the index of sub-blocks. In this way, subscript 1 corresponded to the angle error, and subscript 2 corresponded to the velocity error, and subscript 3 corresponded to the position error. The *a priori* covariance propagation of angle in a single time step could be estimated by

$$P_{11}^{\text{priori}} \approx P_{11} + Q_{11} - (C_b^n P_{41} + P_{14} C_b^{n\text{T}}) \cdot \Delta t + C_b^n P_{44} C_b^{n\text{T}} \cdot \Delta t^2, \tag{5.11}$$

where Δt is the length of a time step. $C_b^n P_{41}$ and $P_{14} C_b^{n\text{T}}$ are symmetric with respect to each other and share the same on-diagonal terms. Note that in Section 5.3.2 and 5.3.3, terms in the form of P_{mn} and $P_{mn}(j,k)$ stand for the posterior covariance obtained from previous update step, or *a priori* covariance from previous prediction step if there is no update in the previous step. The last term on the right hand side of (5.11) could be neglected since the sampling rate was high (typically above 100Hz). The high sampling frequency also

helped to mitigate the error caused by applying the EKF to nonlinear problems. Due to the assumption that the foot motion was 2D, DCM could be expressed by

$$C_b^n = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix},$$

where θ is the pitch angle of the foot. Since the orientation covariance propagations of the two horizontal directions (roll and pitch) were identical, we only needed to focus on one of them. In this study, we selected $P_{11}(1,1)$, which corresponded to the roll angle, and its *a priori* covariance increase during the whole gait cycle was

$$P_{11}^{\text{priori}}(1,1) \approx P_{11}(1,1) + (ARW^2 - 2a_c P_{41}(1,1)) \cdot t_{\text{stride}}, \tag{5.12}$$

where t_{stride} is the time duration of a gait cycle, a_c is the averaged value of $\cos\theta$ during the whole gait cycle and it was estimated to be around 0.84 for a normal human gait pattern [165]. $P_{41}(3,1)$ was neglected because it was much smaller than $P_{41}(1,1)$.

Covariance propagation of the velocity estimation error could be analyzed in a similar way based on (5.7):

$$P_{22}^{\text{priori}} \approx P_{22} + Q_{22} + \{[\vec{f}^n\times]P_{12} + P_{21}[\vec{f}^n\times]^{\text{T}} + C_b^n P_{52} + P_{25}C_b^{n\text{T}}\}\Delta t. \tag{5.13}$$

P_{12} and P_{21} were symmetric with respect to each other. Integration of (5.13) over a whole gait cycle led to the increase of the covariance of velocity estimation during a single gait cycle. $[\vec{f}^n\times]$ was composed of two parts: the constant acceleration $\vec{g}$ and the fast-varying acceleration $\vec{a}_m$ caused by motion. The latter could be neglected in the integration because P_{12} was a relatively slowly-varying term compared to $\vec{a}_m$, and therefore, P_{12} could be considered constant in the integral of their multiplication in (5.13) and taken out of the integral.

Therefore, the expression became an integral of the acceleration $\vec{a}_m$, which equaled zero since the velocity returned to its original value in a complete gait cycle. The terms $P_{52}(1,1)$ and $P_{52}(1,3)$ were much smaller than $P_{12}(1,2)$ and thus could be neglected. Therefore, a total *a priori* covariance increase of the velocity error along the North could be expressed as

$$P_{22}^{\text{priori}}(1,1) \approx P_{22}(1,1) + (VRW^2 - 2g \cdot P_{12}(1,2)) \cdot t_{\text{stride}}, \tag{5.14}$$

where $P_{12}(1,2)$ is the covariance between rotation along the North and the velocity along the East, and g is the gravitational acceleration. $P_{12}(1,2)$ is an important parameter because it corresponds to coupling between the angular rate error and the velocity error. Schuler pendulum, for example, is one of the effects of this term [6]. To complete the analysis, we also needed to calculate the covariance increase of $P_{12}(1,2)$. The covariance propagation was described as

$$P_{12}^{\text{priori}}(1,2) \approx P_{12}(1,2) - g \cdot P_{11}(1,1) \cdot t_{\text{stride}}. \tag{5.15}$$

The sub-block in the covariance matrix that corresponded to the position estimation error was P_{33} and its propagation in the prediction step could be expressed as

$$P_{33}^{\text{priori}} = P_{33} + (P_{23} + P_{32}) \cdot \Delta t + P_{22} \cdot \Delta t^2. \tag{5.16}$$

The position estimation uncertainties along the North and the East were represented by $P_{33}(1,1)$ and $P_{33}(2,2)$, respectively, and they only depended on the propagation of $P_{23}(1,1)$ and $P_{23}(2,2)$, which corresponded to the coupling between the velocity errors and position

errors. Propagations of the coupling terms were expressed as

$$P_{23}^{\text{priori}}(1,1) \approx P_{23}(1,1) + [P_{22}(1,1) + (g - a_D)P_{13}(2,1)] \cdot \Delta t, \tag{5.17}$$

$$P_{23}^{\text{priori}}(2,2) \approx P_{23}(2,2) + [P_{22}(2,2) + (g - a_D)P_{13}(2,1) - a_N P_{13}(3,2)] \cdot \Delta t, \tag{5.18}$$

where a_N is the acceleration along the North, and a_D is the acceleration toward the Down direction. Note that terms related to P_{53} were neglected and not shown in these equations. The reason why a_N and a_D cannot be neglected as in (5.14) will be explained later in this section. The only difference between the two directions was the last term in (5.18).

Similarly, propagations of covariance of other terms were

$$P_{12}^{\text{priori}}(3,2) = P_{12}(3,2) - P_{11}(3,3) \cdot a_N \cdot \Delta t, \tag{5.19}$$

$$P_{13}^{\text{priori}}(2,1) = P_{13}(2,1) + P_{12}(2,1) \cdot \Delta t, \tag{5.20}$$

$$P_{13}^{\text{priori}}(3,2) = P_{13}(3,2) + [P_{12}(3,2) + \sin\theta P_{43}(1,2) - \cos\theta P_{43}(3,2)] \cdot \Delta t, \tag{5.21}$$

$$P_{41}^{\text{priori}}(1,1) = P_{41}(1,1) - P_{44}(1,1)\cos\theta \cdot \Delta t, \tag{5.22}$$

$$P_{41}^{\text{priori}}(1,3) = P_{41}(1,3) + P_{44}(1,1)\sin\theta \cdot \Delta t, \tag{5.23}$$

$$P_{41}^{\text{priori}}(3,3) = P_{41}(3,3) - P_{44}(3,3)\cos\theta \cdot \Delta t, \tag{5.24}$$

$$P_{42}^{\text{priori}}(1,2) = P_{42}(1,2) + P_{41}(1,1) \cdot (-g + a_D) \cdot \Delta t, \tag{5.25}$$

$$P_{42}^{\text{priori}}(3,2) = P_{42}(3,2) - P_{41}(3,3) \cdot a_N \cdot \Delta t, \tag{5.26}$$

$$P_{43}^{\text{priori}}(1,2) = P_{43}(1,2) + P_{42}(1,2) \cdot \Delta t, \tag{5.27}$$

$$P_{43}^{\text{priori}}(3,2) = P_{43}(3,2) + P_{42}(3,2) \cdot a_N \cdot \Delta t, \tag{5.28}$$

$$P_{44}^{\text{priori}}(2,2) = P_{44}(2,2) + RRW^2 \cdot \Delta t. \tag{5.29}$$

5.3.3 Covariance Decrease During the Stance Phase

During the stance phase, the ZUPT-aided navigation algorithm compensated for the IMU errors/noise, and therefore reduced the covariance of the state estimation. The amount of the total reduction could be calculated based on (5.9) and (5.10).

We first analyzed the covariance of the angle estimation. For each time step during the stance phase, the *a posteriori* covariance change could be expressed as

$$\begin{aligned} P_{11}^{\text{posteriori}}(1,1) &= P_{11}(1,1) - \frac{P_{12}(1,1)P_{21}(1,1)}{P_{22}(1,1)+r^2} - \frac{P_{12}(1,2)P_{21}(2,1)}{P_{22}(2,2)+r^2} - \frac{P_{12}(1,3)P_{21}(3,1)}{P_{22}(3,3)+r^2} \\ &\approx P_{11}(1,1) - \frac{P_{12}(1,2)^2}{r^2}. \end{aligned} \tag{5.30}$$

In the strapdown inertial navigation mechanization, the rotation along the North is strongly coupled with the acceleration along the East due to the gravity. Therefore, $P_{12}(1,1)$ and $P_{12}(1,3)$ were much smaller than $P_{12}(1,2)$ and could be neglected. The velocity measurement uncertainty was generally much greater than velocity error induced by IMU noises in ZUPT-aided navigation process [174, 175]. As a result, P_{22} in the denominator was smaller than r^2 and could be neglected.

Similarly, *a posteriori* covariance of other terms that were needed in the derivation could be

calculated as

$$P_{12}^{\text{posteriori}}(1,2) = P_{12}(1,2) - P_{12}(1,2) \cdot P_{22}(2,2)/r^2, \tag{5.31}$$

$$P_{22}^{\text{posteriori}}(2,2) = P_{22}(2,2) - P_{22}(2,2)^2/r^2, \tag{5.32}$$

$$P_{13}^{\text{posteriori}}(3,2) = P_{13}(3,2) - P_{23}(2,2) \cdot P_{12}(3,2)/r^2, \tag{5.33}$$

$$P_{13}^{\text{posteriori}}(2,1) = P_{13}(2,1) - P_{23}(1,1) \cdot P_{12}(2,1)/r^2, \tag{5.34}$$

$$P_{33}^{\text{posteriori}}(1,1) = P_{33}(1,1) - P_{23}(1,1)^2/r^2, \tag{5.35}$$

$$P_{22}^{\text{posteriori}}(1,1) = P_{22}(1,1) - P_{22}(1,1)^2/r^2, \tag{5.36}$$

$$P_{33}^{\text{posteriori}}(2,2) = P_{33}(2,2) - P_{23}(2,2)^2/r^2, \tag{5.37}$$

$$P_{44}^{\text{posteriori}}(2,2) = P_{44}(2,2) - P_{42}(2,1)^2/r^2, \tag{5.38}$$

$$P_{42}^{\text{posteriori}}(1,2) = P_{42}(1,2) - P_{42}(1,2)P_{22}(2,2)/r^2. \tag{5.39}$$

$$P_{41}^{\text{posteriori}}(1,1) = P_{41}(1,1) - P_{42}(1,2)P_{21}(2,1)/r^2. \tag{5.40}$$

5.3.4 Covariance Level Estimation

Since ZUPT-aided navigation algorithm has limited observability of azimuth angle as it is shown in Fig. 5.6, the propagation of error in the azimuth angle and in the z-axis gyroscope bias is the same as in strapdown inertial navigation

$$P_{44}(3,3) = RRW^2 \cdot t, \tag{5.41}$$

$$P_{11}(3,3) = ARW^2 \cdot t + \frac{RRW^2}{3} \cdot t^3, \tag{5.42}$$

where t is the total navigation time. In Section 5.3.4, terms in the form of P_{mn} and $P_{mn}(j,k)$ stand for the predicted continuous covariance bounds. Besides, in this section, we only focused on the level of covariance in the long term, and neglected the variance change within a single gait cycle. Therefore, we did not distinguish *a priori* and *a posteriori* covariances

anymore.

A combination of (5.29) and (5.38) gave

$$RRW^2 \cdot t_{\text{stride}} = \frac{P_{42}(2,1)^2}{r^2} \cdot N_{\text{stance}}. \tag{5.43}$$

Since $N_{\text{stance}} = f_s \cdot t_{\text{stance}}$, where f_s is the sampling frequency of the IMU, P_{42} could be expressed as

$$P_{42}(1,2) = -RRW[\frac{r^2 \cdot t_{\text{stride}}}{f_s \cdot t_{\text{stance}}}]^{\frac{1}{2}}. \tag{5.44}$$

The minus sign in the equation was due to the fact that a positive gyroscope bias along the East would cause a negative velocity estimation error along the North.

Similarly, a combination of (5.25) and (5.39) gave

$$-P_{44}(1,1) \cdot g \cdot t_{\text{stride}} = \frac{P_{42}(1,2)P_{22}(2,2)}{r^2} f_s \cdot t_{\text{stance}}. \tag{5.45}$$

A combination of (5.12) and (5.30) gave

$$[ARW^2 - 2a_c P_{41}(1,1)] \cdot t_{\text{stride}} = \frac{P_{12}(1,2)^2}{r^2} f_s \cdot t_{\text{stance}}. \tag{5.46}$$

A combination of (5.14) and (5.36) gave

$$[VRW^2 - 2g \cdot P_{12}(1,2)] \cdot t_{\text{stride}} = \frac{P_{22}(1,1)^2}{r^2} f_s \cdot t_{\text{stance}}. \tag{5.47}$$

From (5.44)-(5.47) we were able to calculate $P_{22}(1,1)$, which was the root of a quartic

equation

$$ax^4 + bx^2 + cx + d = 0 \tag{5.48}$$

with coefficients expressed as

$$a = [\frac{f_s \cdot t_{\text{stance}}}{2gr^2 \cdot t_{\text{stride}}}]^2, \qquad b = -\frac{f_s \cdot t_{\text{stance}} VRW^2}{2g^2 r^2 \cdot t_{\text{stride}}},$$
$$c = -2a_c \frac{RRW}{g}\sqrt{\frac{r^2 t_{\text{stride}}}{f_s t_{\text{stance}}}}, \qquad d = \frac{VRW^4}{4g^2} - \frac{ARW^2 r^2 t_{\text{stride}}}{f_s t_{\text{stance}}}.$$

An analytical solution to (5.48) exists but it is too complicated and not instructive to write it here. Therefore, instead of searching for the analytical expression, we calculated the solution numerically in this study. Note $P_{22}(1,1)$ was the term in the covariance matrix that corresponded to uncertainty of the velocity estimation along the East. The velocity uncertainty was simply $\sigma_v = \sqrt{P_{22}(2,2)}$. From the equations above, $P_{12}(1,2)$ and $P_{41}(1,1)$ could be calculated as

$$P_{12}(1,2) = -[ARW^2 \frac{r^2 t_{\text{stride}}}{f_s t_{\text{stance}}} + 2a_c \frac{RRW \cdot \sigma_v^2}{g}\sqrt{\frac{r^2 t_{\text{stride}}}{f_s t_{\text{stance}}}}]^{\frac{1}{2}}, \tag{5.49}$$

$$P_{41}(1,1) = -\frac{RRW \cdot \sigma_v^2}{g}\sqrt{\frac{f_s t_{\text{stance}}}{r^2 t_{\text{stride}}}}. \tag{5.50}$$

A combination of (5.22) and (5.40) gave

$$P_{44}(1,1) a_c \cdot t_{\text{stride}} = \frac{P_{42}(1,2) P_{21}(2,1)}{r^2} f_s \cdot t_{\text{stance}}, \tag{5.51}$$

i.e.

$$P_{44}(1,1) = \frac{P_{42}(1,2)P_{21}(2,1)}{a_c}\frac{f_s \cdot t_{\text{stance}}}{r^2 \cdot t_{\text{stride}}} = [(\frac{RRW \cdot ARW}{a_c})^2 + \frac{2\sigma_v^2 RRW^3}{a_c \cdot g}\sqrt{\frac{f_s t_{\text{stance}}}{r^2 t_{\text{stride}}}}]^{\frac{1}{2}}. \tag{5.52}$$

The uncertainty of gyroscope bias along the North was $\sigma_{g_N} = \sqrt{P_{44}(1,1)}$.

Attitude estimation covariance was obtained by combining (5.15) and (5.31)

$$-P_{11}(1,1) \cdot g \cdot t_{\text{stride}} = \frac{P_{12}(1,2)P_{22}(2,2)}{r^2} f_s \cdot t_{\text{stance}}, \tag{5.53}$$

i.e.

$$P_{11}(1,1) = -\frac{P_{12}(1,2)P_{22}(2,2)}{g}\frac{f_s \cdot t_{\text{stance}}}{r^2 \cdot t_{\text{stride}}}. \tag{5.54}$$

The uncertainty of the attitude estimation along the North was $\sigma_\theta = \sqrt{P_{11}(1,1)}$.

To estimate the position uncertainty, we analyzed the propagation of $P_{12}(3,2)$ first. Equation (5.19) showed that the propagation of $P_{12}(3,2)$ was related to the acceleration along the North a_N and the azimuth angle uncertainty $P_{11}(3,3)$. In a single gait cycle, $P_{11}(3,3)$ could be considered as constant since the duration of one gait cycle was relatively short (around 1s). Thus, $P_{12}(3,2)$ was an integral of a_N, i.e., the real velocity of IMU along the North $v_N(t)$. Therefore, $P_{12}(3,2)$ returned to a near-zero value when the update step began, and as a result, the update step had little effect on $P_{12}(3,2)$ since its value was already close to zero. $P_{12}(3,2)$ could be expressed as

$$P_{12}(3,2) \approx -P_{11}(3,3) \cdot v_N(t) = -(ARW^2 t + \frac{RRW^2}{3}t^3) \cdot v_N(t). \tag{5.55}$$

Similarly, several other terms that were necessary in the derivation could be calculated as

$$
\begin{aligned}
P_{41}(3,3) &= -\int_0^t P_{44}(3,3)\cdot\cos\theta\cdot d\tau = -\frac{RRW^2}{2}\cdot a_c\cdot t^2,\\
P_{41}(1,3) &= \int_0^t P_{44}(1,1)\cdot\sin\theta\cdot d\tau = \sigma_{g_N}^2\cdot a_s\cdot t,\\
P_{42}(1,2) &= -\int_0^t P_{41}(1,3)\cdot a_N\cdot d\tau = -\sigma_{g_N}^2\cdot a_s\cdot t\cdot v_N(t),\\
P_{42}(3,2) &= -\int_0^t P_{41}(3,3)\cdot a_N\cdot d\tau = \frac{RRW^2}{2}\cdot a_c\cdot t^2\cdot v_N(t),\\
P_{43}(1,2) &= \int_0^t P_{42}(1,2)\cdot d\tau = -\int_0^t \sigma_{g_N}^2\cdot a_s\cdot t\cdot v_N(t)\cdot d\tau\\
&\approx -\sum_i \sigma_{g_N}^2\cdot a_s\cdot t_i \int\limits_{\text{cycle } i} v_N(t)\cdot d\tau\\
&= -\sum_i \sigma_{g_N}^2\cdot a_s\cdot t_i s_N = -\frac{1}{2}\sigma_{g_N}^2\cdot a_s\cdot t^2 s_N,\\
P_{43}(3,2) &= \int_0^t P_{42}(3,2)\cdot d\tau = \frac{RRW^2}{6}\cdot a_c\cdot s_N\cdot t^3,\\
P_{13}(3,2) &= \int_0^t [P_{12}(3,2) + P_{43}(1,2)\sin\theta - P_{43}(3,2)\cos\theta]d\tau\\
&= -(\frac{ARW^2}{2}t^2 + \frac{RRW^2}{12}t^4)\cdot s_N - \frac{1}{6}\sigma_{g_N}^2\cdot a_s^2\cdot s_N\cdot t^3 - \frac{RRW^2}{24}\cdot a_c^2\cdot s_N\cdot t^4,
\end{aligned}
$$

where a_s is the average value of $\sin\theta$ over the whole gait cycle, and s_N is the stride length of the human gait. In the equation for $P_{43}(1,2)$, the integral over the whole navigation process was calculated as the summation of the integral over each gait cycle i. Besides, in the integral of each gait cycle, t was approximated as a constant t_i and moved out of the integral, because the relative changing rate of v_N was much larger than that of t in a single gait cycle.

Then, we estimated the level of P_{23} since it was related to the propagation of P_{33}, both in the prediction step (indicated by (5.16)) and in the update step (indicated by (5.35) and (5.37)).

From our extensive observation of many cases, the value of $P_{13}(2,1)$ remained at a constant

level during the navigation. Therefore, a combination of (5.20) and (5.34) yielded

$$P_{12}(2,1)t_{\text{stride}} = N_{\text{stance}}P_{23}(1,1)\cdot P_{12}(2,1)/r^2, \tag{5.56}$$

or equivalently represented as

$$P_{23}(1,1) = \frac{r^2\cdot t_{\text{stride}}}{N_{\text{stance}}} = \frac{r^2\cdot t_{\text{stride}}}{f_s\cdot t_{\text{stance}}}. \tag{5.57}$$

Comparing (5.17) and (5.18) yielded

$$P_{23}(2,2) = P_{23}(1,1) - \int_0^t a_N P_{13}(3,2)\cdot d\tau \approx \frac{r^2\cdot t_{\text{stride}}}{f_s\cdot t_{\text{stance}}} - P_{13}(3,2)\cdot v_N(t). \tag{5.58}$$

Now we explain the reason why the acceleration caused by foot motion cannot be neglected in (5.17) and (5.18). The position uncertainty $P_{33}(2,2)$ was derived by integrating $P_{23}(2,2)$ twice, and the acceleration term a_N would be transformed to the displacement term s_N. Therefore, even though the velocity v_N returned to zero after a full gait cycle, its integral, displacement s_N, cannot be neglected. The acceleration term in (5.14), however, was only integrated once to obtain the final result for velocity uncertainty. As a result, neglecting the acceleration term would not introduce large errors but would only lose information about some fluctuations within a single gait cycle.

Combining (5.16) and (5.35), we could obtain the full increment of $P_{33}(1,1)$, which corresponded to the square of position uncertainty along the trajectory, during a complete gait cycle

$$\Delta P_{33}(1,1) = 2P_{23}(1,1)\cdot t_{\text{stride}} - \frac{P_{23}(1,1)^2}{r^2}\cdot N_{\text{stance}} \approx (2-\frac{t_{\text{stride}}}{4})\frac{r^2\cdot t_{\text{stride}}}{f_s\cdot t_{\text{stance}}}\cdot t_{\text{stride}}. \tag{5.59}$$

Therefore, the propagation of $P_{33}(1,1)$ could be expressed as

$$P_{33}(1,1) = (2 - \frac{t_{\text{stride}}}{4}) \frac{r^2 \cdot t_{\text{stride}}}{f_s \cdot t_{\text{stance}}} \cdot t. \tag{5.60}$$

The propagation of $P_{33}(2,2)$ could be derived similarly as

$$P_{33}(2,2) = (2 - \frac{t_{\text{stride}}}{4}) \frac{t_{\text{stride}} r^2}{t_{\text{stance}} f_s} \cdot t + \frac{1}{3} ARW^2 s_N^2 \cdot t^3 + \frac{a_s^2}{12} \sigma_{g_N}^2 s_N^2 \cdot t^4 + (\frac{1}{30} + \frac{a_c^2}{60}) RRW^2 s_N^2 \cdot t^5. \tag{5.61}$$

Set $\sigma_{\shortparallel} = \sqrt{P_{33}(1,1)}$, and $\sigma_{\perp} = \sqrt{P_{33}(2,2)}$. $\sigma_{\shortparallel}$ and $\sigma_{\perp}$ were the position estimation uncertainties parallel and perpendicular to the trajectory, and they corresponded to 1.2 times of the semi-major and semi-minor axes of Circular Error Probable (CEP), respectively.

Equations (5.48), (5.42), (5.54), (5.60), and (5.61) fully described the uncertainty of navigation results due to IMU noises with respect to angle, velocity, and position.

5.3.5 Observations

1. ARW, VRW, and RRW all affected the final navigation uncertainties, for example in (5.48); higher noise level resulted in larger errors.

2. The velocity measurement uncertainty r played an important role in the final results; lower r indicated a higher reliability and weight of the zero-update information in the EKF, resulting in a better navigation accuracy. However, this value was determined by human gait pattern and the type of floor [170]. Therefore, it should be adjusted according to the experiment and cannot be set arbitrarily.

3. The position uncertainty along the trajectory was dominated by the velocity measurement uncertainty r in the EKF and was proportional to square root of the navigation time.

4. The position uncertainty perpendicular to the trajectory depended on many parameters (see (5.61)). However, it was dominated by RRW and was proportional to the navigation time of the power of 2.5, in the case of long-term navigation.

5. Human gait pattern affected the navigation errors. It was reflected in the ratio between duration of the stance phase and the whole gait cycle and the average of sine and cosine value of the pitch angle, for example in (5.46). A higher percentage of the stance phase during the gait cycle gave the EKF more data to compensate for the IMU noises and reduced the overall navigation errors.

6. AcRW was not included in the model. This was due to our assumption that the propagated velocity covariance during the swing phase was much smaller than the velocity measurement uncertainty r. This conclusion agreed with the argument in [174].

7. The results were only approximations of the navigation errors due to assumptions and approximations made during the derivation, e.g. 2D foot motion, moderate IMU performance, high IMU sampling rate, straight trajectory, etc. Validity of the approximations would be demonstrated in the following section.

5.4 Verification of Analysis

5.4.1 Numerical Verification

Simulations were conducted to verify the derived analytical expressions. First, a trajectory of foot toward the North and the corresponding IMU readouts were generated based on a human gait analysis reported in Section 5.2. Then, the numerical results were compared to analytical expressions. The generated trajectory was a straight line toward the North

containing 100 steps. The total time duration of the trajectory was 53.6s and the total length of the trajectory was 77m.

5.4.1.1 Effect of ARW

We first studied the influence of the ARW of gyroscopes on the navigation errors. We swept the ARW value from $0.01\,^\circ/\sqrt{hr}$ to $10\,^\circ/\sqrt{hr}$ (from near navigation grade to consumer grade), while keeping other parameters constant. VRW of accelerometers was set to be $0.14mg/\sqrt{Hz}$ (industrial grade), RRW of gyroscopes was set to be $0.048\,^\circ/s/\sqrt{hr}$, and the sampling frequency was selected to be 800Hz. The simulation results are presented in Fig. 5.7. The upper plot shows a relation between the ARW and the velocity estimation uncertainty and the lower plot shows a relation between the ARW and the angle estimation uncertainty. Notice that the angle estimation uncertainty was only for roll and pitch angle, since yaw angle was unobservable in the EKF and propagated according to (5.42). In both plots, the blue lines are analytical results, and the red error bars are simulation results. The simulation results were a range instead of a value because covariances of the estimation errors fluctuated during the navigation (see Fig. 5.6). The upper and lower bounds of the error bars showed the amplitude of fluctuation and the square showed an average value of the fluctuation.

A close match between the analytical and simulation results verified validity of the analysis. Fig. 5.7 shows that both velocity estimation uncertainty and angle estimation uncertainty were not affected by ARW when its value was smaller than $0.1\,^\circ/\sqrt{hr}$. One possible reason was that in this case the navigation uncertainty was dominated by other errors, such as VRW and RRW, therefore it was independent of ARW value. The lower bound of the fluctuation of velocity uncertainty was almost not affected by ARW either. This was because the lower bound of the velocity uncertainty was limited by the velocity measurement uncertainty set in the EKF, which was fixed in this model. It was also noticed that fluctuation of the angle uncertainty was much smaller than the velocity uncertainty. The reason was that the velocity

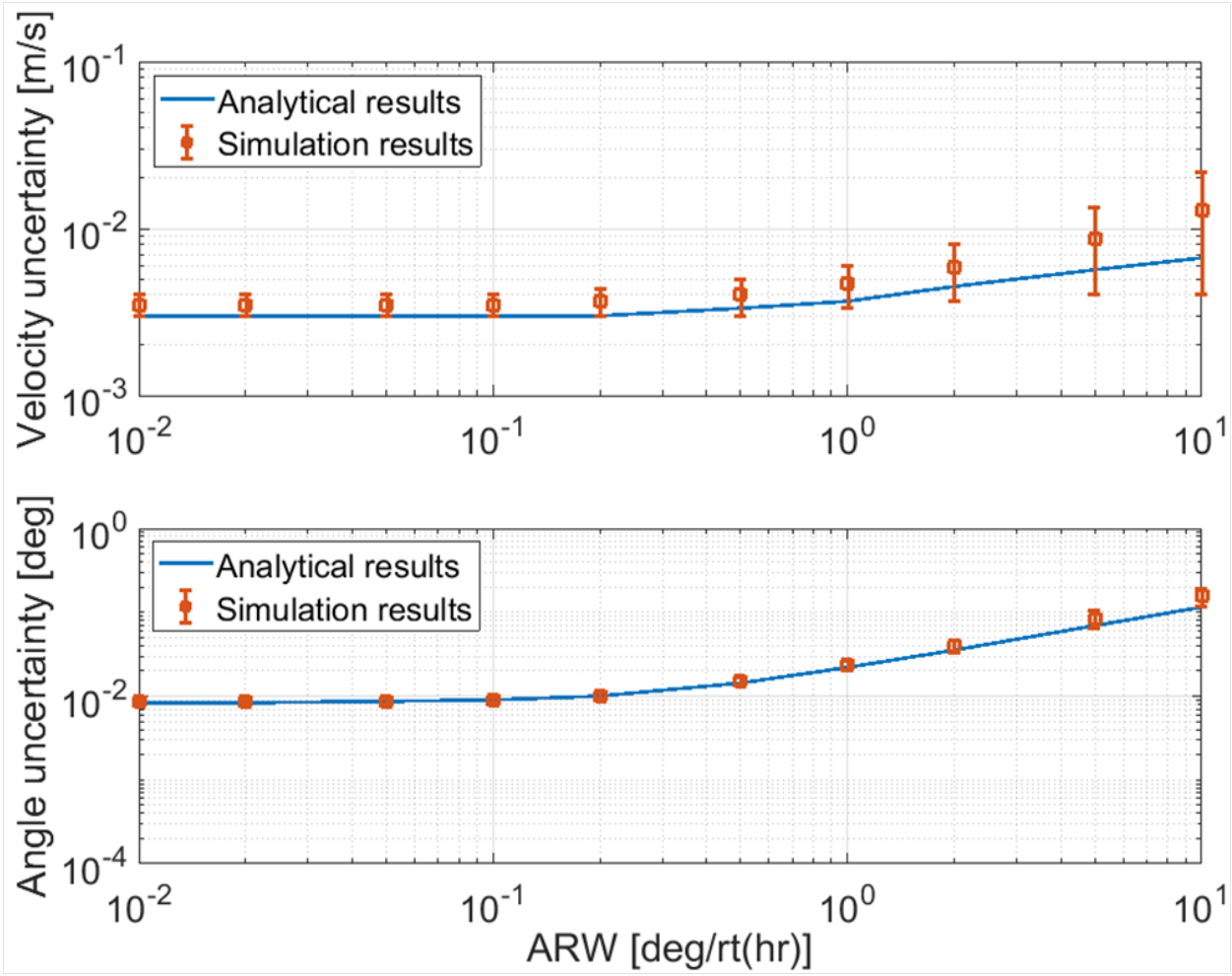

Figure 5.7: Effects of ARW of the gyroscopes on the velocity and angle estimation errors in the ZUPT-aided inertial navigation algorithm.

was directly observable in the ZUPT-aided navigation algorithm, and therefore, the EKF could directly estimate the velocity value and reduce the velocity uncertainty. The angle estimation, however, was achieved through coupling the velocity and angle, and as a result, the observability was reduced.

5.4.1.2 Effect of VRW

Similarly, we swept the VRW value of accelerometers from $0.01mg/\sqrt{Hz}$ to $10mg/\sqrt{Hz}$, while keeping ARW of the gyroscope to be $0.21^\circ/\sqrt{hr}$ (industrial grade) and RRW to be

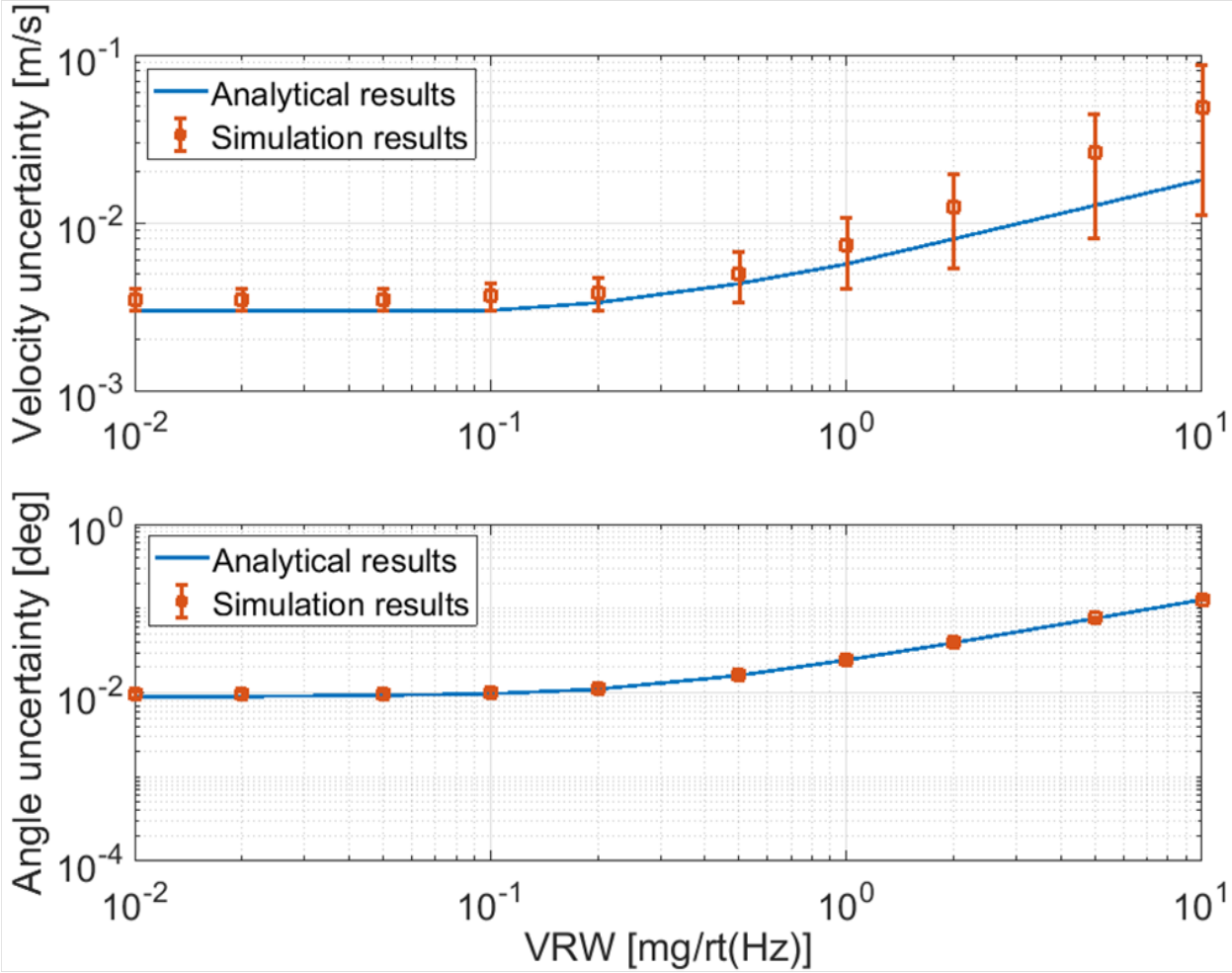

Figure 5.8: Effects of VRW of the accelerometers on the velocity and angle estimation errors in the ZUPT-aided inertial navigation algorithm.

$0.048\,^\circ/s/\sqrt{hr}$. The results are shown in Fig. 5.8. As expected, the curves became flat when VRW was small, since the navigation error was dominated by gyroscope errors in this range.

5.4.1.3 Effect of RRW

As indicated in (5.61), RRW was the major error source that affects the navigation accuracy. We swept the RRW value of gyroscopes from $6 \times 10^{-4}\,^\circ/s/\sqrt{hr}$ to $0.6\,^\circ/s/\sqrt{hr}$, while keeping ARW of the gyroscope to be $0.21\,^\circ/\sqrt{hr}$ and VRW to be $0.14mg/\sqrt{Hz}$. The influence of RRW on the velocity and angle estimation errors is shown in Fig. 5.9.

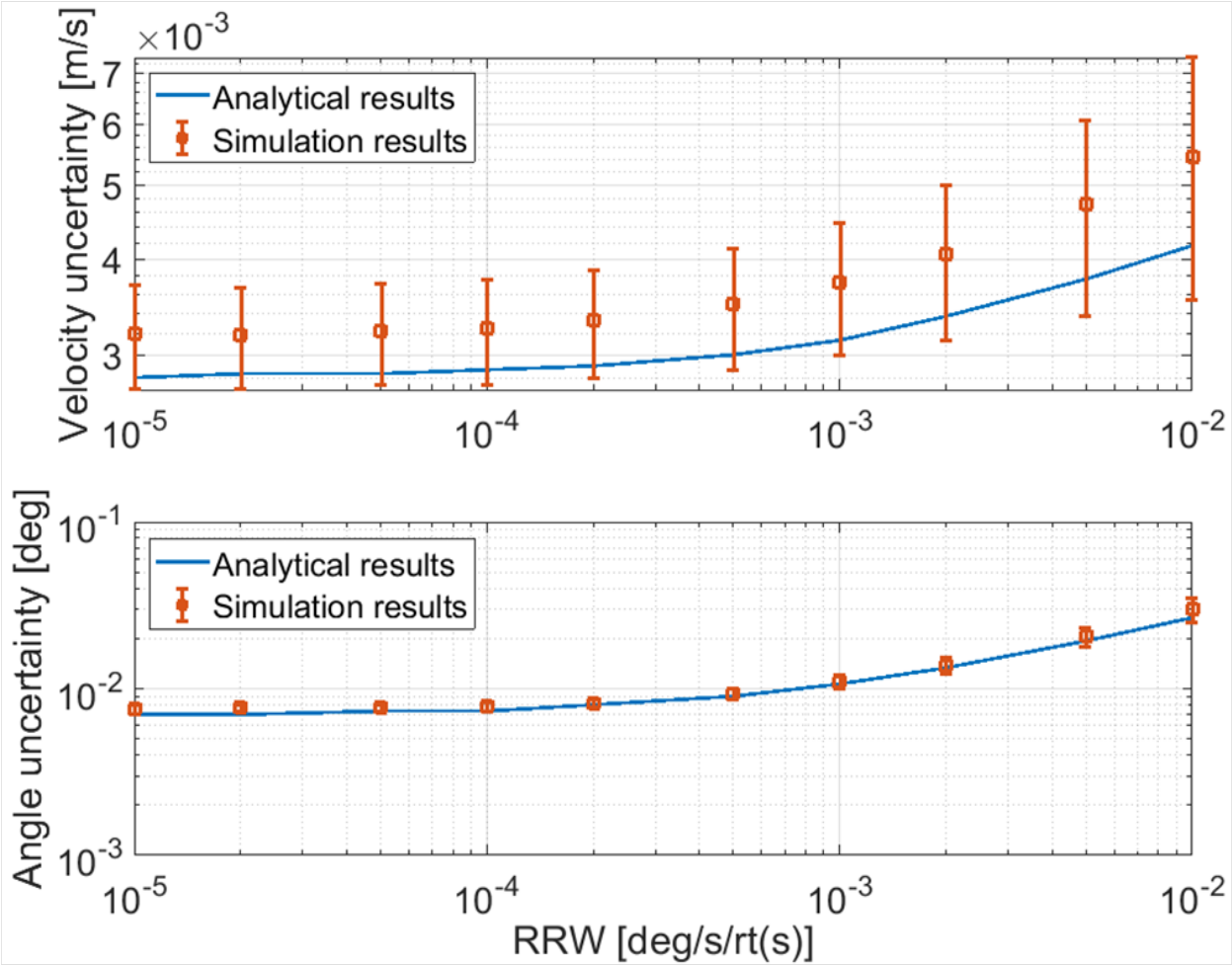

Figure 5.9: Effects of RRW of the gyroscopes on the velocity and angle estimation errors in the ZUPT-aided inertial navigation algorithm.

Fig. 5.10 shows the relation between the position uncertainty and RRW. A difference within 10% was demonstrated between the analytical results and the numerical results. Note that the position uncertainty perpendicular to the trajectory was not affected by RRW, but dominated by velocity uncertainty during the stance phase. As a result, a lower velocity measurement uncertainty would be desirable for a better navigation accuracy. The following considerations could help to reduce the velocity measurement uncertainty and improve the overall navigation accuracy: (1) a stiffer shoe with less deformation during walking, (2) a better position to attach the IMU, so that the IMU could be more stationary during the stance phase, and (3) shock absorber on the shoes to prevent strong shocks between the shoe

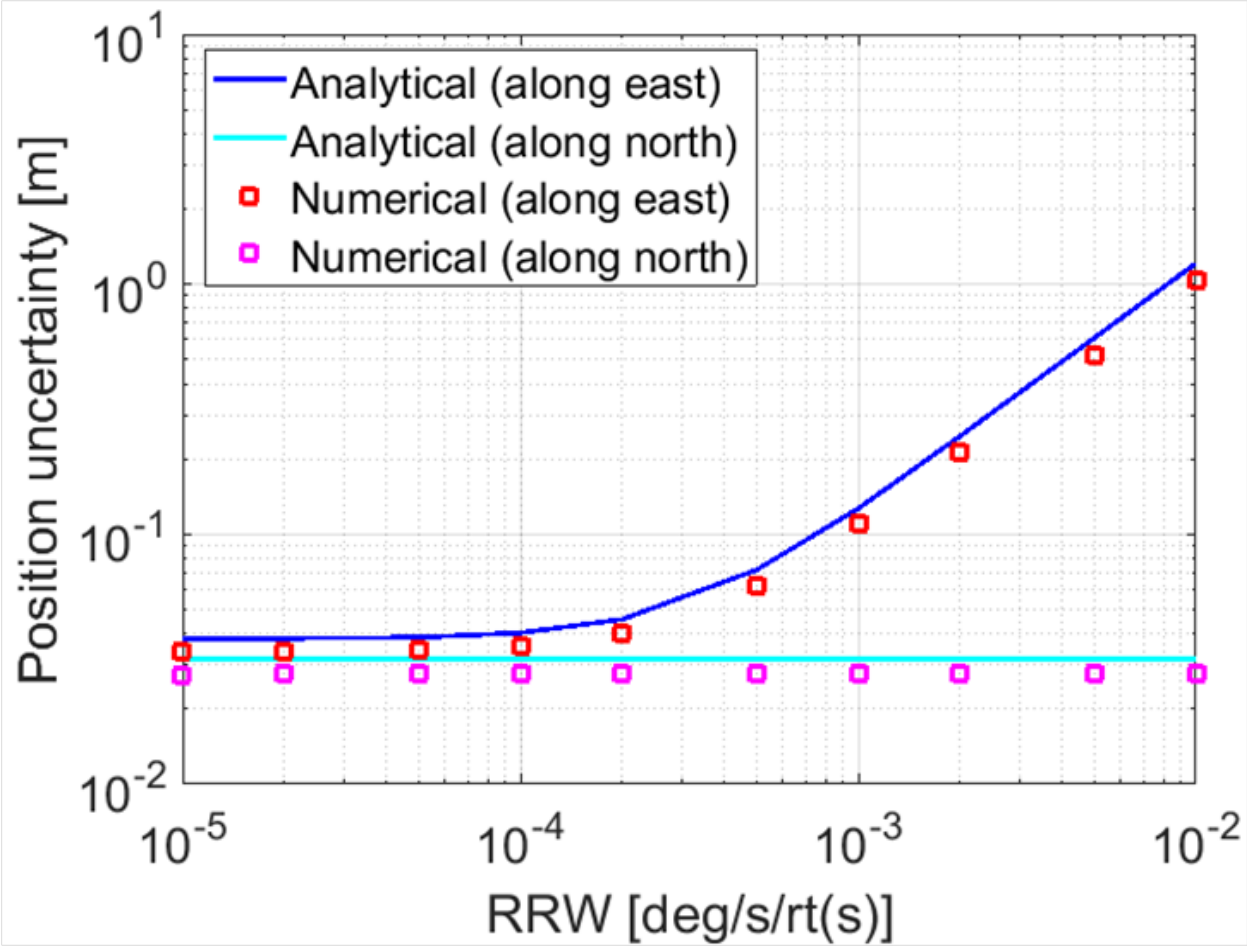

Figure 5.10: Relation between RRW of gyroscopes and the position estimation uncertainties.

and the ground [176].

5.4.2 Experimental Verification

A VectorNav VN-200 INS (Industrial grade) was mounted on the right shoe by a 3D-printed fixture (Fig. 5.12) and IMU readouts were collected during walking. The uncompensated IMU data instead of the compensated ones were used to avoid any possible affects of the filtering algorithm in the INS on the overall navigation accuracy. Allan deviations of the accelerometer and gyroscope readouts were collected to confirm the performance of the IMU [30], and the result is shown in Fig. 5.11. ARW, VRW, and RRW of the IMU were

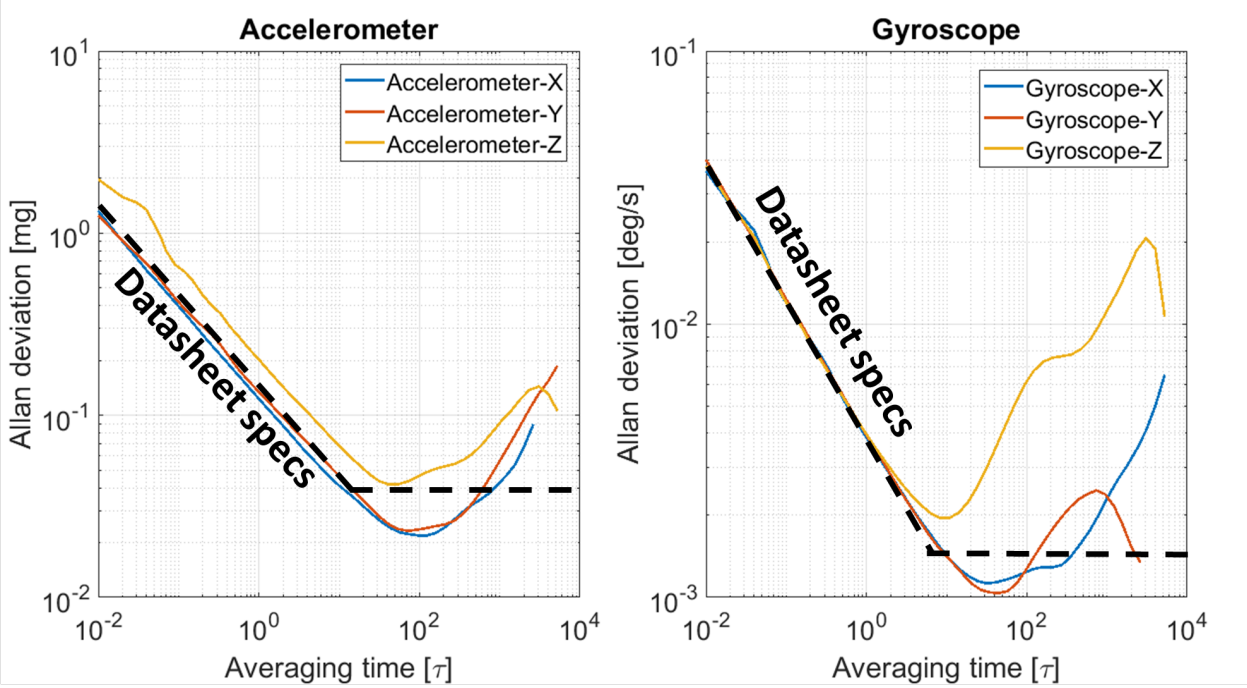

Figure 5.11: Allan deviation plot of the IMU used in this study. The result was compared to the datasheet specs [177].

$0.21\,°/\sqrt{hr}$, $0.14mg/\sqrt{Hz}$, and $0.048\,°/s/\sqrt{hr}$, respectively. Sampling frequency was set to be 800Hz (maximum sampling rate of the IMU) to capture the high-frequency component of the motion. The length of the straight line trajectory was around 100m, and the total navigation time was around 110s. During the first 10s of each run, the foot was stationary for the initial calibration of IMU and calculation of the roll and pitch angles. Magnetometer was used to determine the initial orientation of the foot. IMU data for 40 trajectories were collected to obtain a relatively accurate position uncertainty during the navigation.

The navigation error results of 40 trajectories are shown in Fig. 5.12. In all cases, the nominal trajectory was a straight line trajectory toward the North. All estimated trajectories exhibited a drift to the right side, and the averaged drift value was 1.82m. This phenomenon was a result of the systematic error due to the non-zero velocity of the foot during the stance phase and the gyroscope g-sensitivity, and it has been reported in [174]. This drift was the result of a systematic errors, and it will be discussed in detail in Chapter 6.

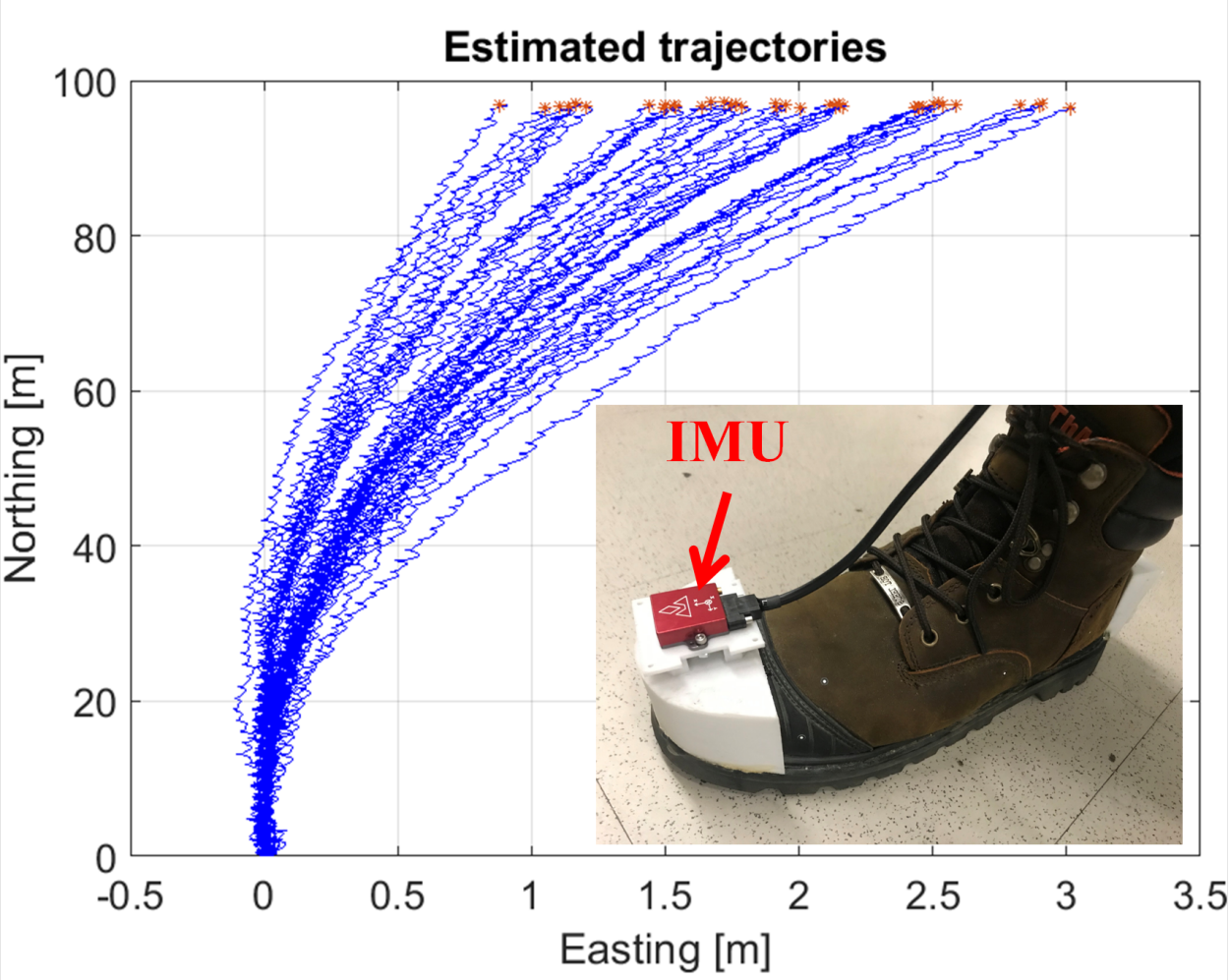

Figure 5.12: The navigation error results of 40 trajectories. The averaged time duration was about 110s, including the initial calibration. Note that scales for the two axes were different to highlight the effect of error accumulation.

A zoomed-in view of the ending points of 40 trajectories is shown in Fig. 5.13. The ending points were distributed in a rectangular area with the length of 2.2m and width of 0.8m. Based on IMU performances from Fig. 5.11 and analytical results from (5.60) and (5.61), the uncertainty of position estimation could be calculated as $\sigma_{\shortparallel} = 0.07m$, and $\sigma_{\perp} = 0.43m$. Assuming the position error was normally distributed, then 99% of the points should be in an ellipse with the major axis of $6\sigma_{\perp} = 2.58m$ and the minor axis of $6\sigma_{\shortparallel} = 0.42m$. The analytical expression was within 20% of the experimental result along the direction perpendicular to the trajectory, showing a good agreement. As for the direction along the trajectory, the

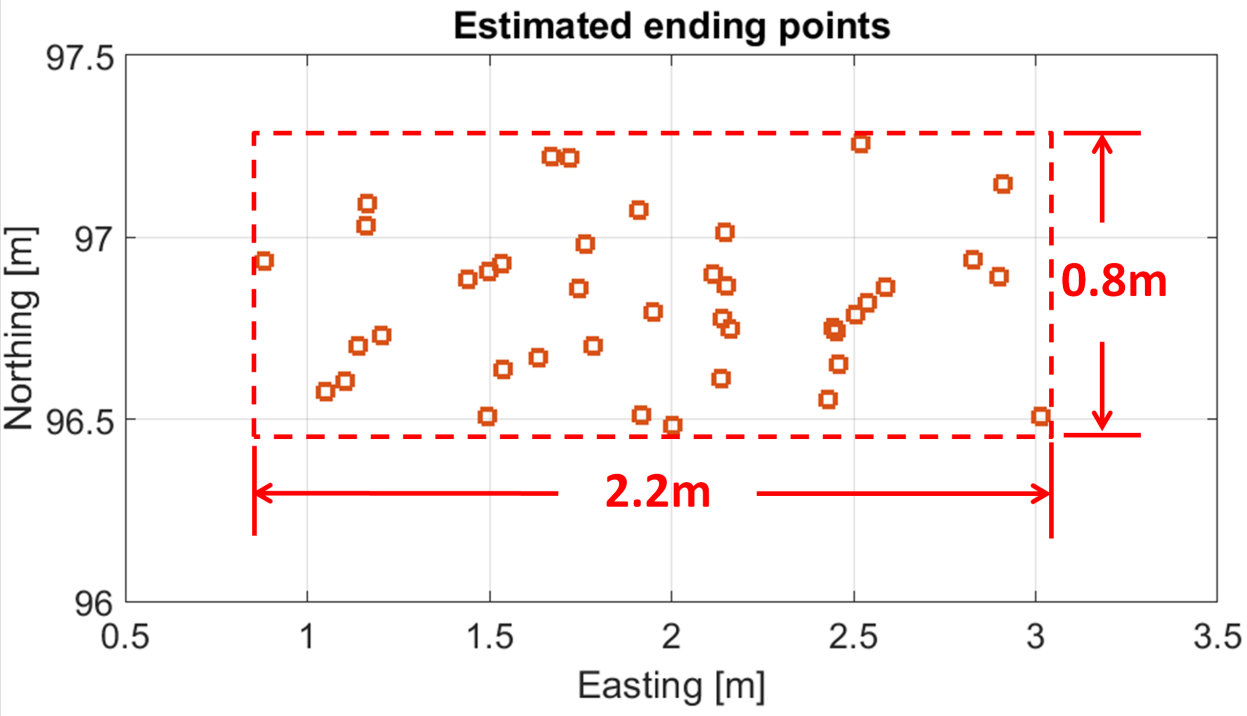

Figure 5.13: Ending points of 40 trajectories. All data points were in a rectangular area with the length of 2.2m and width of 0.8m.

analytical result was about 50% smaller than the experimental result, possibly due to the systematic modeling errors, or the velocity uncertainty during the stance phases. The shock and vibrations of the IMU when the foot touched the ground were also not considered in the analytical model, and they would also introduce extra errors. Similar phenomenon of larger navigation errors with IMU mounted on the foot has been reported previously in [44].

5.5 Conclusions

In this chapter, an analytical model correlating the IMU errors and the navigation errors in the ZUPT-aided pedestrian inertial navigation was presented. Analytical expression of the uncertainty of the estimation of angle, velocity, position, and IMU stochastic biases were derived and confirmed with discrepancy of 10% with numerical simulation and 20% with experimental results for the direction perpendicular to the trajectory. Rate Random Walk

of the z-axis gyroscope was found to be the main factor that affects the navigation accuracy. We believe that a relatively small discrepancy between the analytical result and the numerical simulation is indicative of the accuracy of the analysis, while a relatively larger discrepancy between the analytical result and the experiments are likely the result of systematic modeling errors, such as biased and correlated process noises, nonlinear navigation dynamics, and the zero-velocity assumption. This will be addressed in the next chapter.

This study estimates the magnitude of navigation errors due to IMU errors, laying a basis for error analysis in pedestrian inertial navigation. It is envisioned to aid in analysis of the effect of errors in sensors, which might lead to a well informed selection of sensors for the task of ZUPT-aided pedestrian inertial navigation.

Chapter 6

Algorithm Development — Navigation Error Reduction

In this chapter, methods to reduce the navigation error in the ZUPT-aided pedestrian inertial navigation without adding extra sensing modalities are presented. Both systematic bias and stochastic noise in the navigation error are greatly reduced. In Section 6.1, an adaptive stance phase detection is presented, being able to adapt the algorithm to different walking or running speeds. Next, Section 6.2 introduces systematic error reduction by compensation of residual velocity during the stance phase and gyroscope g-sensitivity. In Section 6.3, experimental demonstration of stochastic noise reduction is presented by optimizing IMU mounting positions. Then in Section 6.4, a combination of the aforementioned compensation methods is demonstrated, and their effects are discussed. Finally, the chapter is concluded with Section 6.5.

6.1 Adaptive Stance Phase Detection

Adaptive stance phase detection is necessary in the ZUPT-aided pedestrian inertial navigation to make the algorithm more robust, especially in a complex navigation environment. In this section, we develop an adaptive threshold based on the Bayesian approach to enable the detector to work with various walking or running speeds [178].

6.1.1 Zero-Velocity Detector

A zero-velocity detector can be mathematically expressed as a binary hypothesis test, where the detector can choose between the two hypotheses: the IMU is moving (H_0) or the IMU is stationary (H_1). A common approach is to apply the Neyman-Pearson theorem and compare the likelihood ratio with some pre-defined threshold γ [179]: choose H_1 if

$$L(\boldsymbol{z}_n) = \frac{p(\boldsymbol{z}_n \mid H_1)}{p(\boldsymbol{z}_n \mid H_0)} > \gamma, \tag{6.1}$$

where $\boldsymbol{z}_k = \{\boldsymbol{y}_k\}_{k=n}^{n+N-1}$ is the N consecutive IMU readouts between time index n and $n+N-1$, and $L(\cdot)$ is the likelihood ratio of probability of the measurement.

Stance Hypothesis Optimal dEtector (SHOE) is one of the most commonly used zero-velocity detectors [118]. This detector is based on the fact that the foot is almost stationary during the stance phase, and therefore the magnitude of the specific force is equal to the gravity, and the angular rate is close to zero. However, many parameters that determine the Probability Density Functions (PDF) of observations are unknown due to a complicated dynamics of the human gait [175]. One possible solution is to replace the unknown parameters with their Maximum Likelihood (ML) estimates, and this method is called the Generalized Likelihood

Ratio Test (GLRT) [180]. Using this method, the test statistic can be expressed as

$$L'_{\mathrm{ML}}(\boldsymbol{z}_n) = -\frac{2}{N} log(L_{\mathrm{ML}}(\boldsymbol{z}_n)) = \frac{1}{N} \sum_{k=n}^{n+N-1} \frac{1}{\sigma_a^2} \mid\mid \boldsymbol{y}_k^a - g\frac{\bar{\boldsymbol{y}}^a}{\mid\mid \bar{\boldsymbol{y}}^a \mid\mid} \mid\mid^2 + \frac{1}{\sigma_\omega^2} \mid\mid \boldsymbol{y}_k^\omega \mid\mid^2, \tag{6.2}$$

where $\boldsymbol{y}_k^a$ and $\boldsymbol{y}_k^\omega$ are the accelerometer and the gyroscope readouts at time index k, respectively, $\bar{\boldsymbol{y}}^a$ is the averaged value of the N consecutive accelerometer readouts, σ_a and σ_ω are related to the white noise level of the accelerometer and the gyroscope, and g is the gravity.

We can then state the GLRT as: choose H_1 if

$$L'_{\mathrm{ML}}(\boldsymbol{z}_n) < \gamma', \tag{6.3}$$

where γ' is the threshold to be determined.

6.1.2 Adaptive Threshold Determination

For different gait patterns, the distributions of $\boldsymbol{z}_n$ are also different, and therefore different thresholds are needed. In this section, we derived an adaptive threshold based on a time-dependent cost function. The main goal of the adaptive threshold was to adjust the ZUPT detector to different walking or running patterns, in order to minimize the extra navigation errors by erroneously applying ZUPT.

There were three general goals of adaptive threshold:

- Limit the probability of false alarm. False alarm happens if the detector determines the IMU is stationary while the foot is actually moving. False alarm will cause KF to erroneously set the velocity close to zero, which greatly degrades the results of navigation.
- Minimize the probability of miss-detection. Stance phase is the time period when

zero-velocity information can be utilized to suppress the navigation error growth. Miss-detection will reduce the chance of compensation, therefore increase the overall navigation error.

- Adjust the threshold parameters automatically to fit different gait dynamics and maintain a proper amount of zero-velocity updates.

Fig. 6.1 shows a typical test statistic $L'_{\mathrm{ML}}(\boldsymbol{z}_n)$ for different gait dynamics. There were six different gait dynamics shown in Fig. 6.1, corresponding to walking at the pace of 80, 90, 100, 110, and 120 steps per minute, and running at the pace of 160 steps per minute, respectively. High test statistic indicated that the IMU was moving while a lower value showed that the IMU was close to a stationary state. Fig. 6.1 shows that the test statistic was around 50 when the foot was stationary, and this value was mostly related to IMU noises. The red dashed lines are the averaged values of test statistic during the stance phase related to different gait dynamics, ranging from the lowest of 4×10^4 to the highest of 6×10^5. Note that these values were much higher than the test statistic when the foot was stationary on the floor, indicating that the foot was actually not stationary during the stance phase. Therefore, an excessive use of zero-velocity update would cause a degradation to the overall navigation accuracy [174]. Since the lowest values of the test statistic were different with varying walking or running speeds, an adaptive threshold was necessary to make the ZUPT detection more robust, especially in the case where a varying walking speed was involved.

The Bayesian likelihood ratio test states that the threshold can be expressed as

$$\gamma = \frac{p(H_0)}{p(H_1)} \cdot \frac{c_{10} - c_{00}}{c_{01} - c_{11}}, \tag{6.4}$$

where $p(H_0)/p(H_1)$ is the prior probability of the hypotheses, c_{00}, c_{11}, c_{10}, and c_{01} are the cost functions of correct detection of the swing phase, correct detection of the stance phase, a false alarm, and a miss-detection, respectively. We assumed a uniform prior probability since no

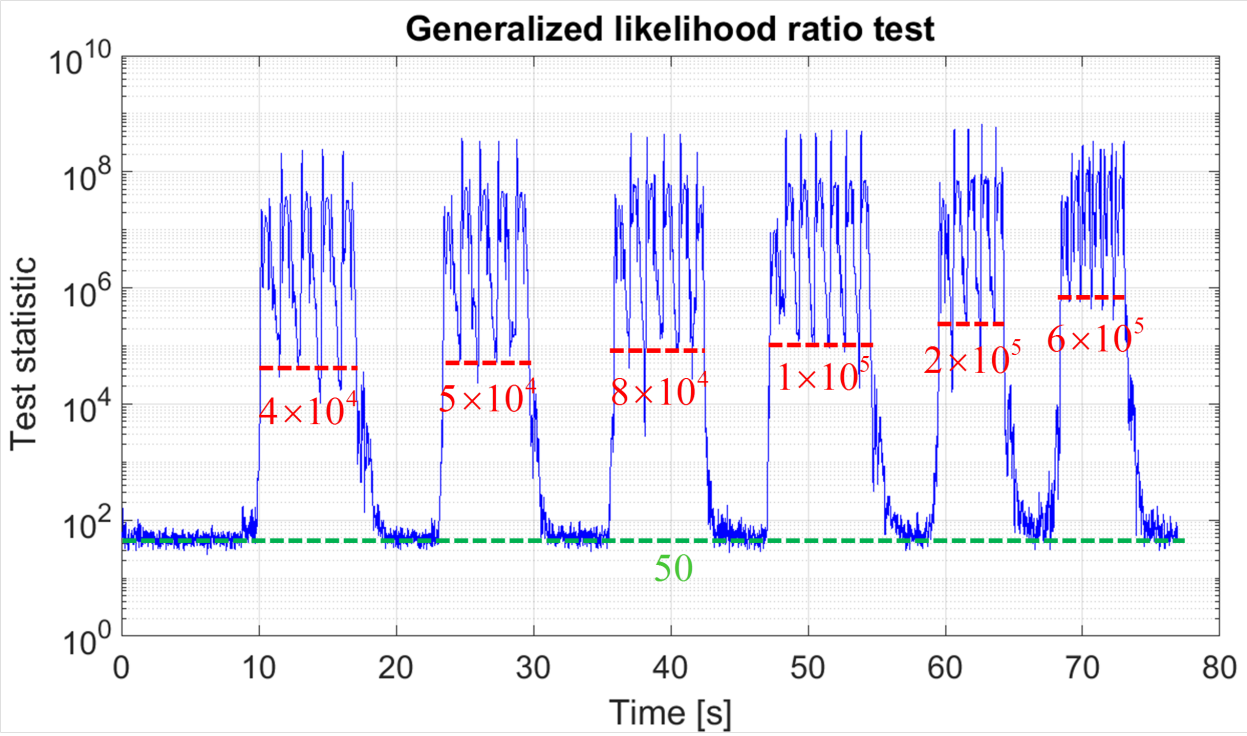

Figure 6.1: The blue line is a typical test statistic for different walking and running paces. The red dashed lines show the test statistic levels during the stance phase with different gait paces, and the green dashed line shows the test statistic level when standing still.

other information of the motion was available, and a zero cost for correct detections. Thus, the threshold equaled the ratio of the cost function of the false alarm and miss-detection.

In a miss-detection, the stance phase of the foot is not detected, and therefore the zero-velocity information is not fused in the system to suppress the error propagation. The associated cost is time-dependent, since the navigation error accumulates as a polynomial with respect to time without any error suppression [173]. Therefore, it is proper to assume a polynomial cost function for miss-detection instead of an exponential cost function reported in [178]. On the other hand, the false alarm is the case where the zero-velocity information is fused into the system while the foot is still moving. The cost of a false alarm is related to the actual velocity of the foot, and it is relatively random and time-independent compared to the cost of a miss-detection [178]. Therefore, a constant cost function was assumed for the false alarm. To summarize, the ratio of the cost of a false alarm to the cost of a miss-detection could be expressed in a polynomial form

$$\gamma = \frac{c_{10}}{c_{01}} = \alpha_1 \cdot \Delta t^{-\theta_1}, \tag{6.5}$$

where Δt is the time difference between the previous ZUPT event and the current time step, and α_1 and θ_1 are design parameters to be decided. The threshold γ' could be defined as

$$\gamma' = -\frac{2}{N} log(\gamma) = -\frac{2}{N}(log(\alpha_1) - \theta_1 \cdot log(\Delta t)) \triangleq \theta \cdot log(\Delta t) + \alpha. \tag{6.6}$$

The value of γ' was low immediately after detection of the last stance phase, since the time interval Δt was small. The physical interpretation was that the probability of detecting another stance phase was low, according to (6.6), since we did not expect two stance phases to be very close to each other.

Another advantage of polynomial cost can be shown from (6.6). Note that (6.6) would be

in the form of $\theta \cdot \Delta t + \alpha$, if an exponential cost were to be used. For normal gait patterns, the range of Δt is typically around 1s. The slope of $log(\Delta t)$, which is associated with the polynomial cost, is similar to that of Δt, which is in turn associated with the exponential cost, when Δt is around 1s, but the slope is much larger if Δt is smaller than 1s. Therefore, a similar performance of stance phase detection could be expected with better robustness against false alarm in between the two stance phases.

The threshold γ' would increase at a speed defined by θ as Δt increases, and α biases the overall level of the threshold. Ideally, θ would be defined such that γ' increases to the level of the test statistic during the stance phase in one gait cycle. This requires an estimation of the level of the test statistic during the stance phase, which is directly related to the gait frequency. In this study, we proposed to take advantage of the shock level that the IMU experiences during the heel strike as an indicator to estimate the real-time gait frequency.

As the step pace increases, the minimum test statistic increases, as well as the shock level that the IMU experiences during the heel strike. The relation between the shock level and the minimum test statistic is shown in Fig. 6.2. The dots correspond to data from different gait cycles, the red solid line is the fitted curve, and the pink dashed lines are 1σ interval of the fitting. An exponential formula could be used to approximate the relation, and the parameter θ in (6.6) could be defined as

$$\theta = \epsilon \cdot \exp(0.0307 \times Shock + 8.6348), \tag{6.7}$$

where $Shock$ is the shock level during the heel strike, ϵ is the parameter that can be adjusted to achieve a proper length of the stance phase, and it was set to be 3.5 in this study. The parameter α adjusts the overall threshold level to reduce the probability of miss-detection and to improve the algorithm robustness. Advantages of using the shock level during the heel strike to extract gait frequency include:

- Priori knowledge of the gait frequency is not needed;
- Ability to continuously track the gait frequency with no lag;
- Less amount of computation than FFT or machine learning.

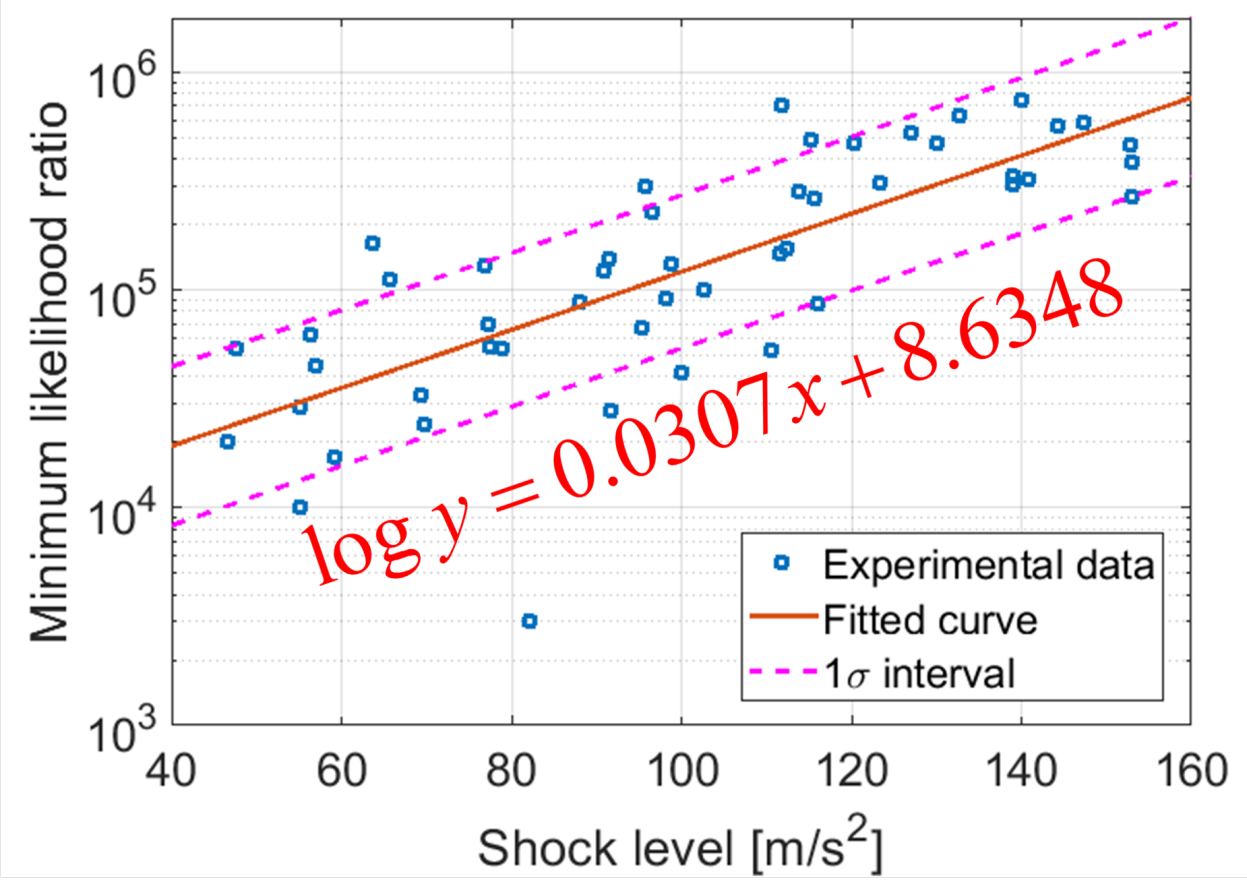

Figure 6.2: The relation between the shock level and the minimum test statistic in the same gait cycle. The dots correspond to data from different gait cycles, the red solid line is a fitted curve, and the pink dashed lines are 1σ intervals.

Note that the adaptive threshold in (6.6) is related to δt, indicating that the threshold will drop immediately after a stance phase is detected. In this case, only one time step of the stance phase can be determined by the detector (shown by the discrete red points in Fig. 6.3). This will deteriorate the navigation accuracy since not all stance phase events are utilized. To ensure that enough zero-velocity updates could be implemented to suppress the navigation error, we proposed to hold the threshold γ' until it became smaller than the test

statistic again, instead of allowing it to drop back as (6.6) indicates. The effect of holding is shown in Fig. 6.3. The threshold level remained constant during the stance phase (indicated by the yellow bar), which enabled us to detect the whole stance phase, instead of discrete time instances during the stance phase.

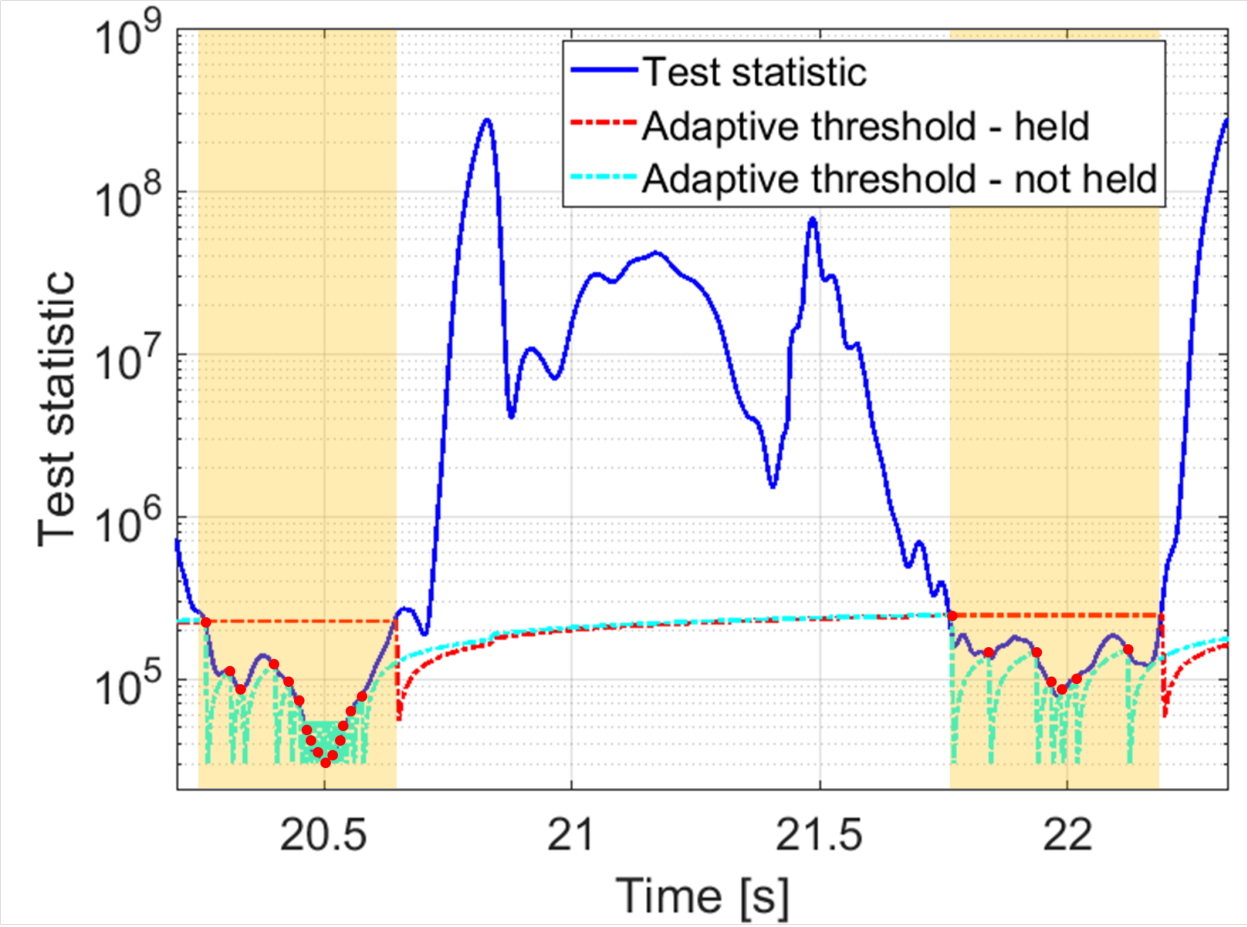

Figure 6.3: The red dashed line and the cyan dashed line are adaptive thresholds with and without an artificial holding, respectively. The red dots indicate the stance phases detected by the threshold without holding, while the stance phases detected by the threshold with holding is shown by the yellow bar.

6.1.3 Experimental Verification

Experiments were conducted to verify the effects of the adaptive threshold. A straight trajectory of 75m was used in the experiment. The IMU was rigidly mounted at the forefoot

and the sampling rate was 200Hz. During the navigation, the subject first stood still for about 12s, then walked at a pace of 84 steps per minute for about 15s. It was followed by running at a pace of 160 steps per minute for about 15s and walking for another 20s. At last, the subject stood for about 5s. The position propagation, accelerometer readouts, generalized likelihood ratio test, and the navigation results are presented in Fig. 6.4. Fig. 6.4 (a) shows the estimated position over time, and the velocity difference between walking and running could be clearly observed. Fig. 6.4 (b) shows the specific force of the IMU. A much higher shock level of about 150m/s^2 during running was observed, exceeding the level of about 50m/s^2 during walking. Fig. 6.4 (c) shows the test statistic (blue solid line) and the adaptive threshold (red dashed line). The threshold successfully captured the changes in dynamics during walking and running, and the threshold value was about 2×10^5 (green dashed line) and 2×10^6 (black dashed line), respectively. Fig. 6.4 (d) shows the comparison of estimated trajectories with different threshold settings. In the case of fixed threshold, the stance phase during running cannot be detected, if the threshold was set to be 2×10^5, which was proper for walking. Thus, the estimated trajectory drifted away soon after the subject started running. On the other hand, if the threshold was set to be 2×10^6 to detect the stance phase during running, too many zero-velocity updates were imposed during walking, and the estimated trajectory became much shorter than the ground truth by 12m, corresponding to 16% of the total trajectory length. By applying adaptive threshold for the stance phase detection, the navigation was reduced to 3m. The effect of holding the threshold during the stance phase was shown by further reducing the navigation error from 3m to 1m.

Navigation Root-Mean-Square Error (RMSE) has been analyzed with respect to different fixed thresholds for the same trajectory, and the result is presented in Fig. 6.5. The minimum RMSE with a fixed threshold was 0.98m, corresponding to a threshold 1.65×10^6. This value was between the adaptive threshold values for walking and running, and therefore it could be considered as a trade-off: some stance phases during running could be detected, while not too much ZUPT would be imposed during walking. The RMSE for adaptive threshold was

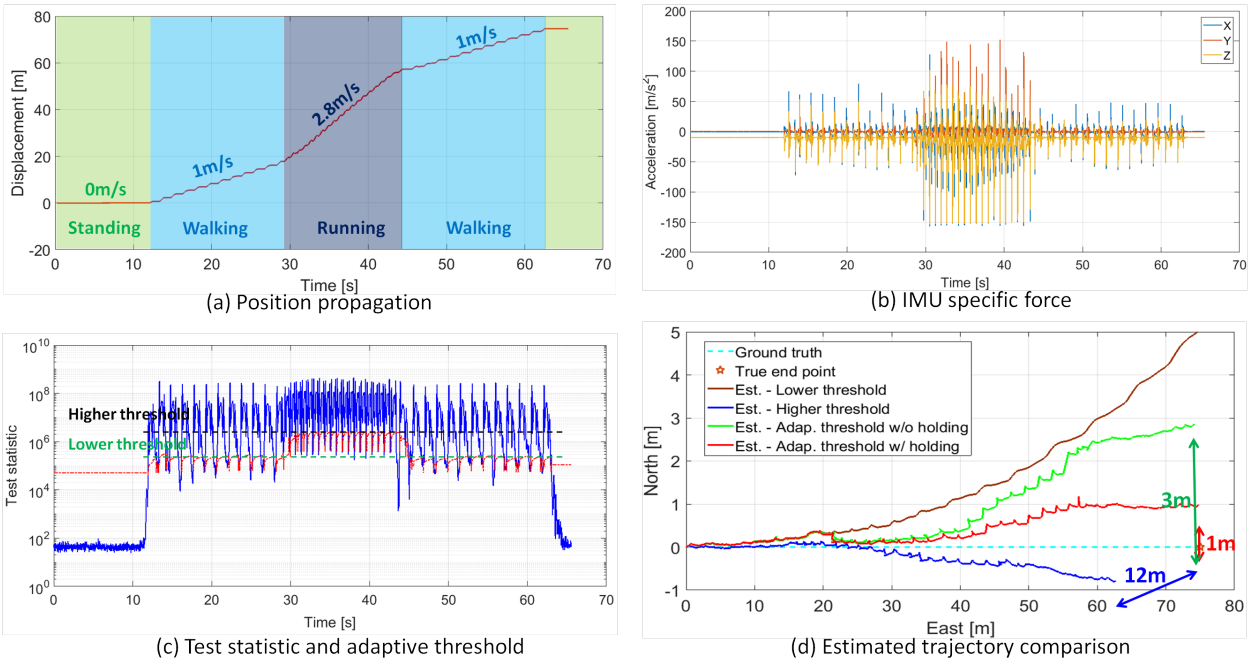

Figure 6.4: Position propagation, specific force of the IMU, generalized likelihood ratio test, and the navigation results of the experiment. Note that the x and y axis scalings in (d) are different.

0.61m, and it was lower than the lowest RMSE achieved by any fixed threshold, showing the advantage of adaptive threshold for ZUPT detection. Note that the optimal fixed threshold was related to the many parameters, such as walking speed, floor type, and walking pattern, and therefore it was typically not available in most navigation scenarios, and could only be determined empirically. As a result, we expect that adaptive threshold will generally perform much better than any fixed threshold, especially in the case where walking or running paces are changing during navigation, as in the case shown in Fig. 6.4.

6.2 Systematic Error Compensation

After deriving an adaptive threshold for the stance phase detection, the next step is to compensate for systematic errors induced by the zero-velocity assumption during the stance phase.

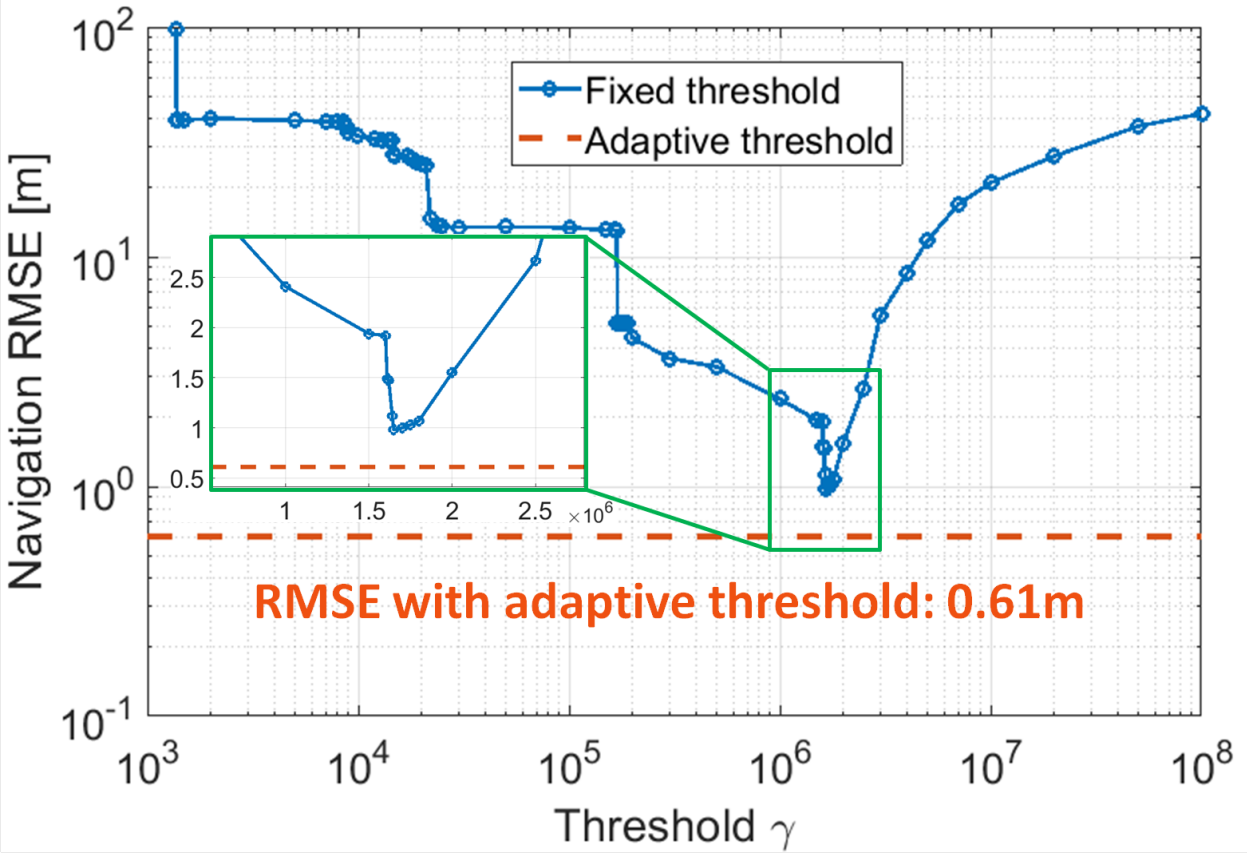

Figure 6.5: Relation between the navigation RMSE and fixed threshold level is shown by the blue solid line. The navigation RMSE achieved by adaptive threshold is shown by the red dashed line.

6.2.1 Error Source Analysis

Two major errors in the ZUPT-aided pedestrian inertial navigation are the underestimate of the length of the trajectory and the systematic drift of orientation of the trajectory. In this section, we identify and describe the sources of these errors.

6.2.1.1 Trajectory Length Underestimate

It has been demonstrated that underestimate of the trajectory length is related to the zero-velocity assumption of the foot during the stance phase [174]. Even with an adaptive threshold for the stance phase detection, underestimate of the length of the trajectory still exists.

To quantitatively analyze the relation between the ZUPT assumption and the underestimate of the trajectory length, the motion of the foot during the stance phase needs to be recorded and analyzed. A magnetic motion tracking system (Polhemus PATRIOT) was used in this study. There were two parts in the system: a magnetic source placed on the floor as a reference, and a tracker mounted by a custom fixture on top of the IMU (VectorNav VN-200). The experimental setup is shown in Fig. 6.6. The tracking system was able to track the relative position between the tracker and the source with a nominal resolution of 1mm at a sampling frequency of 60Hz [181]. Velocity of the foot was derived by taking derivative of the relative position with respect to time.

Seventy gait cycles were recorded with a walking pace of approximately 84 steps per minute toward the North, and the results are shown in Fig. 6.7. The red solid lines are the averaged velocities along three directions. The stance phase corresponded approximately to the time period between 0.5s and 0.9s (indicated by the red box), and the velocity of the foot was close to zero during the stance phase. However, Fig. 6.8 shows the zoomed-in view of the velocity during the stance phase. The red solid lines are the averaged velocities of the foot,

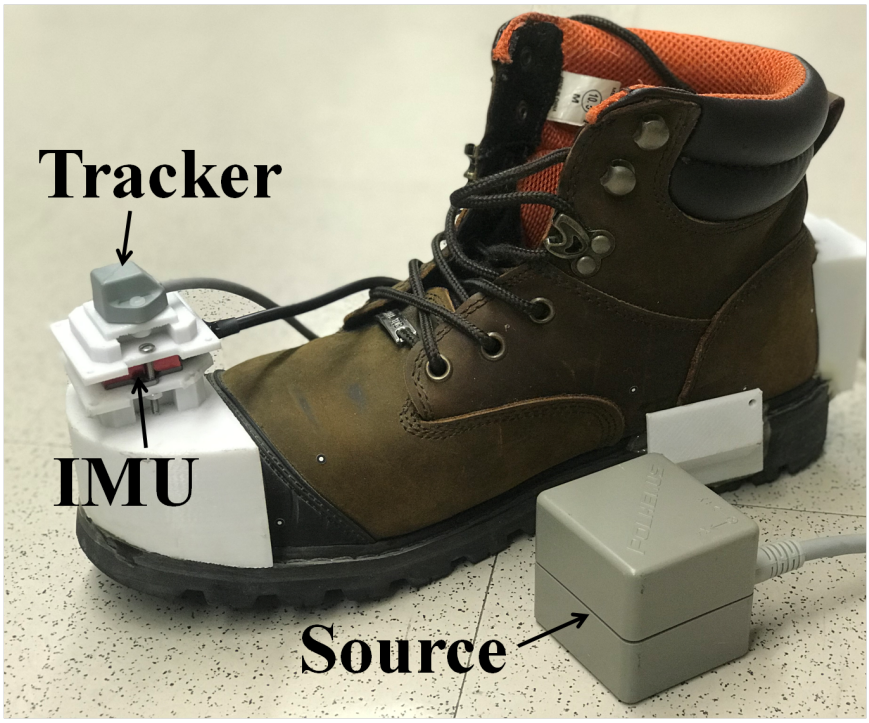

Figure 6.6: Experimental setup to record the motion of the foot during the stance phase.

and the yellow dashed lines correspond to zero velocity. A residual velocity on the order of 0.01m/s was clearly observed, and therefore, assuming the non-zero residual velocity during the stance phase to be zero would introduce a systematic error. The blue dashed lines are the 1σ interval of the velocity distribution, showing a velocity uncertainty about 0.02m/s during the stance phase. The noisy individual measurement was due to the relatively low sampling rate of the motion tracker and the derivative operation to extract the velocity from the distance. Note that the fluctuation caused by the derivative operation would increase the measured velocity variance.

Underestimate of the trajectory length was directly related to the residual velocity during the stance phase. However, the residual velocity was not constant during the stance phase. Therefore, its average value was related to the length of the stance phase as determined by the ZUPT detector. Fig. 6.9 shows the test statistics of the same 70 steps recorded previously, and the red solid line is an averaged value.

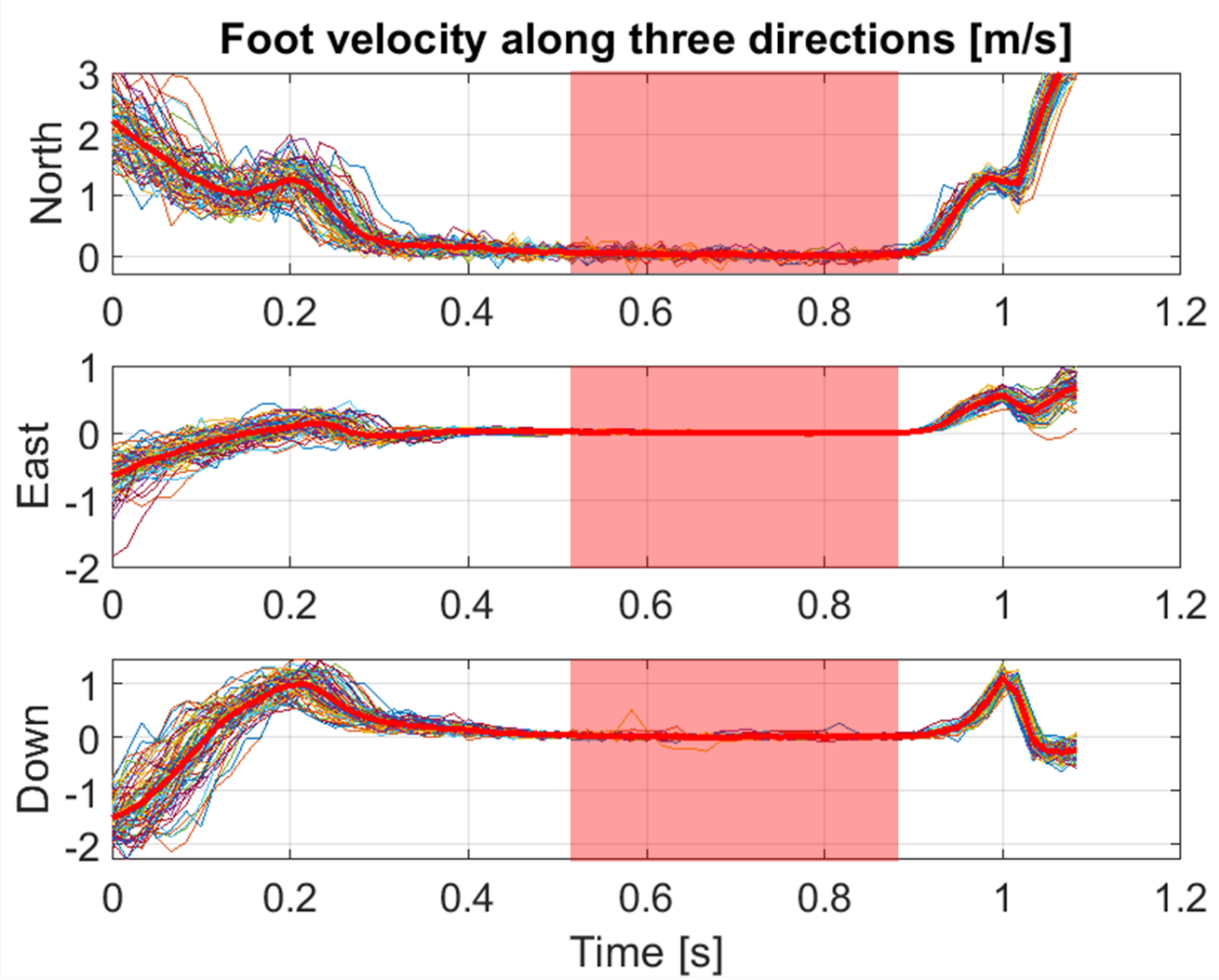

Figure 6.7: Velocity of the foot along three directions during a gait cycle. The red solid lines are the averaged velocity along three directions.

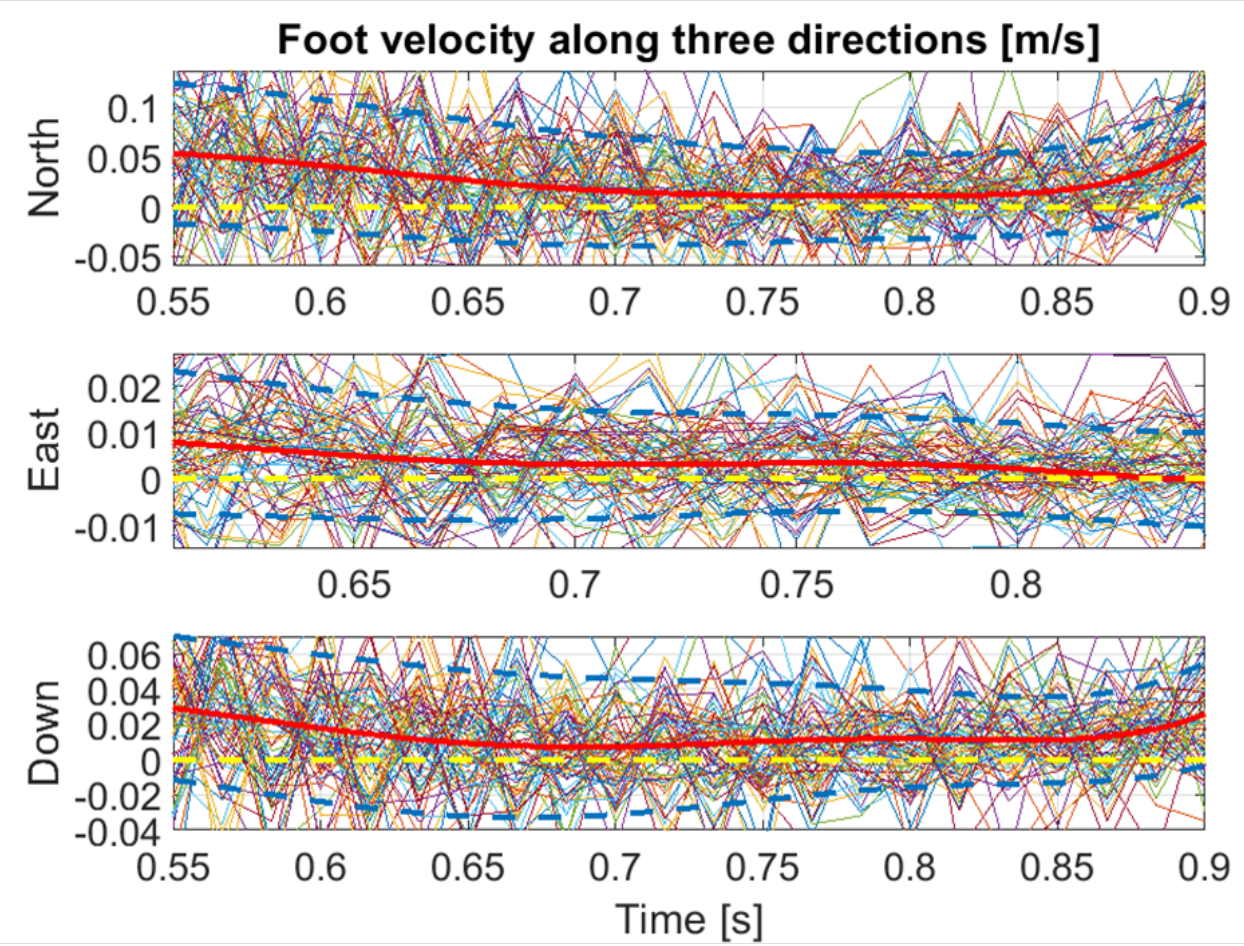

Figure 6.8: Zoomed-in view of the velocity of the foot during the stance phase. The yellow dashed lines correspond to zero-velocity state, and the blue dashed lines are the 1σ range of the velocity distribution.

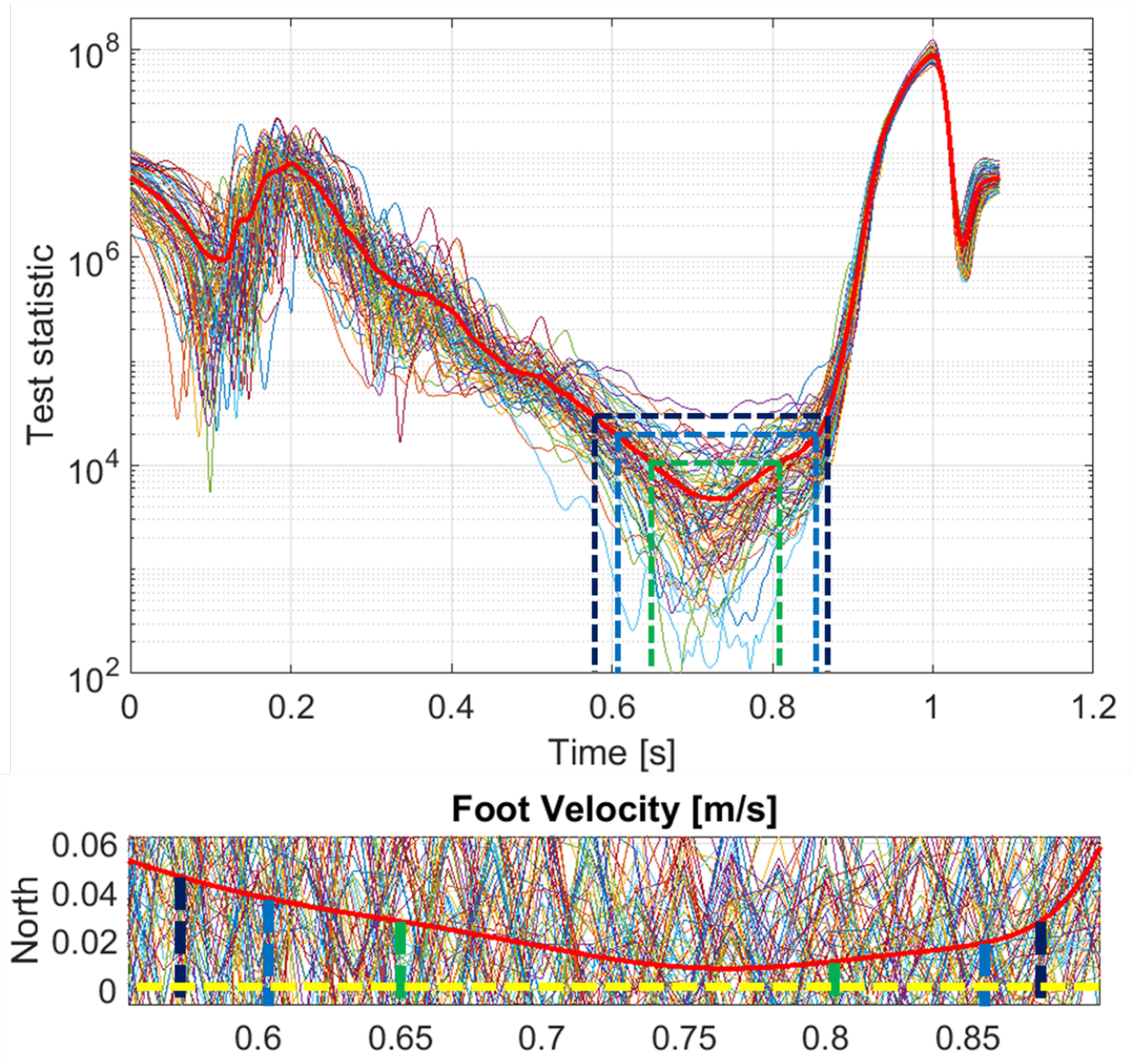

Figure 6.9: The upper figure shows the test statistics of the same 70 steps recorded previously. Red solid line is an averaged value. The lower figure shows the residual velocity of the foot along the trajectory during the stance phase. The green, light blue, and dark blue dash lines correspond to threshold levels of 1×10^4, 2×10^4, and 3×10^4, respectively.

The level of the test statistic was not a constant even when the foot was in the stance phase. Therefore, the length of the detected stance phase was related to the value of the pre-defined threshold. For example, the detected stance phase was between 0.65s and 0.8s, if the threshold was set to 1×10^4 (the dashed green line in Fig. 6.9). If the threshold was increased to 3×10^4, the detected stance phase was between 0.57s and 0.86s (the dashed dark blue line in Fig. 6.9). A longer detected stance phase led to a higher averaged residual velocity of the foot during the stance phase, and therefore a higher systematic error would be introduced.

To experimentally verify the effects of residual velocity during the stance phase, IMU data were recorded for ten straight trajectories with length of 100m. The walking pace was 84 steps per minute. For each of the trajectory, thresholds ranging from 1×10^4 to 5×10^4 were applied in the ZUPT detector, and the underestimate of the trajectory length was recorded and shown in Fig. 6.10. The red solid line is the result of the previous analysis, and the thinner lines are experimental results. A good match was demonstrated, verifying that the residual velocity during the stance phase was the major factor that led to the underestimate of the trajectory length.

6.2.1.2 Trajectory Orientation Drift

Trajectory orientation drift in the ZUPT-aided pedestrian inertial navigation is believed to be related to the g-sensitivity of gyroscopes [182, 183]. Due to a severe dynamics of the foot during walking, the heading angle error is demonstrated to accumulate at a rate of $135\,^\circ$/h in [44], even though the gyroscope bias instability is only $3\,^\circ$/h.

To relate the trajectory orientation drift and the heading angle drift, IMU data were recorded for a straight line trajectory of 550m toward the North. The experimentally recorded trajectory is shown by the blue solid line in Fig. 6.11, showing a drift to the right. The estimated

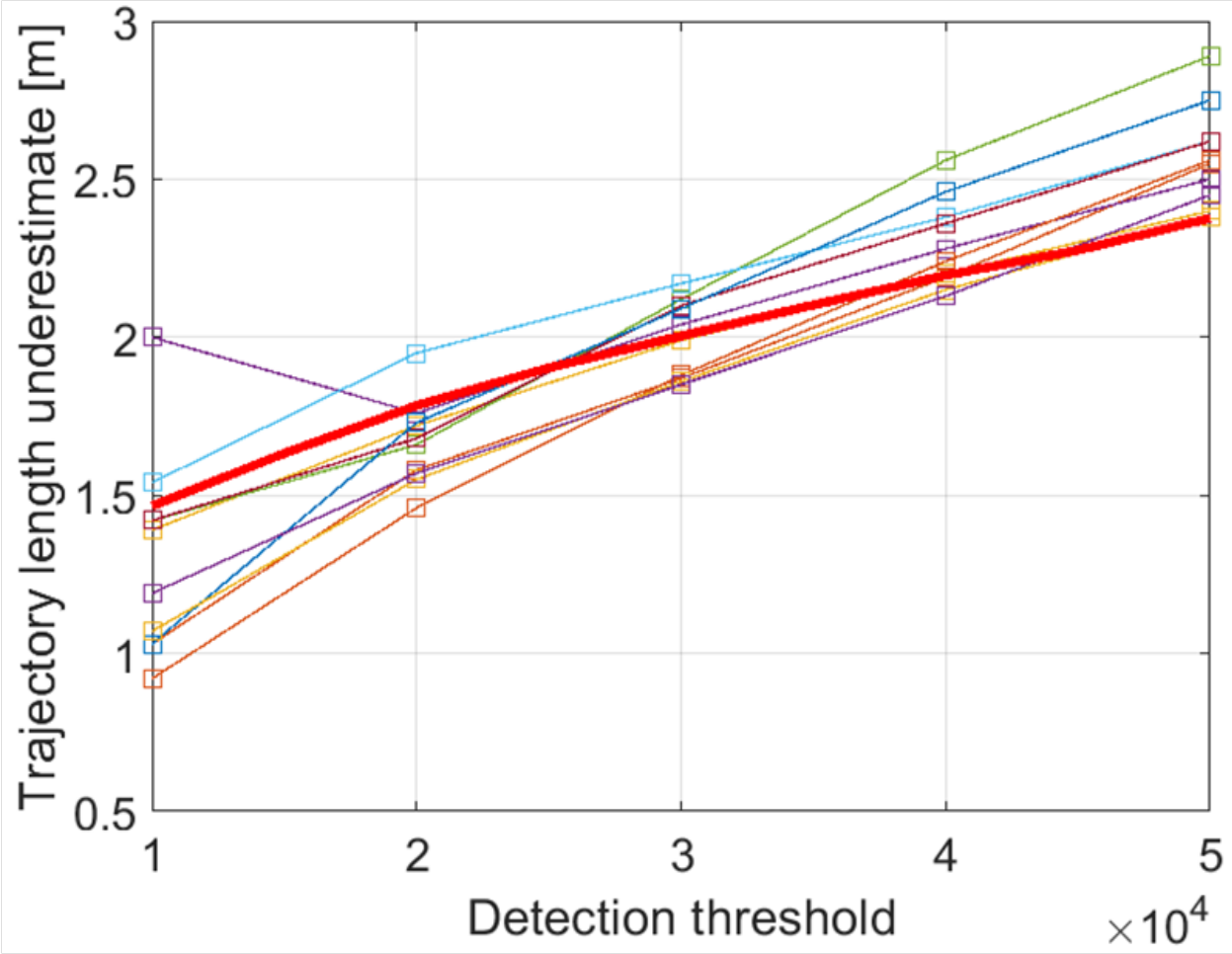

Figure 6.10: Relation between the underestimate of trajectory length and the ZUPT detection threshold. The red solid line is the result of the previous analysis, and the thinner lines are experimental results from 10 different runs.

heading angle is shown in the inset in Fig. 6.11, showing a drift at the rate of 0.028 °/s. The red dashed line shows an analytically generated trajectory assuming a constant speed and a heading angle increase at the rate of 0.028 °/s. The experimental and generated trajectories matched each other with a difference within 10m, indicating that the heading angle drift was the major factor that leads to the trajectory orientation drift.

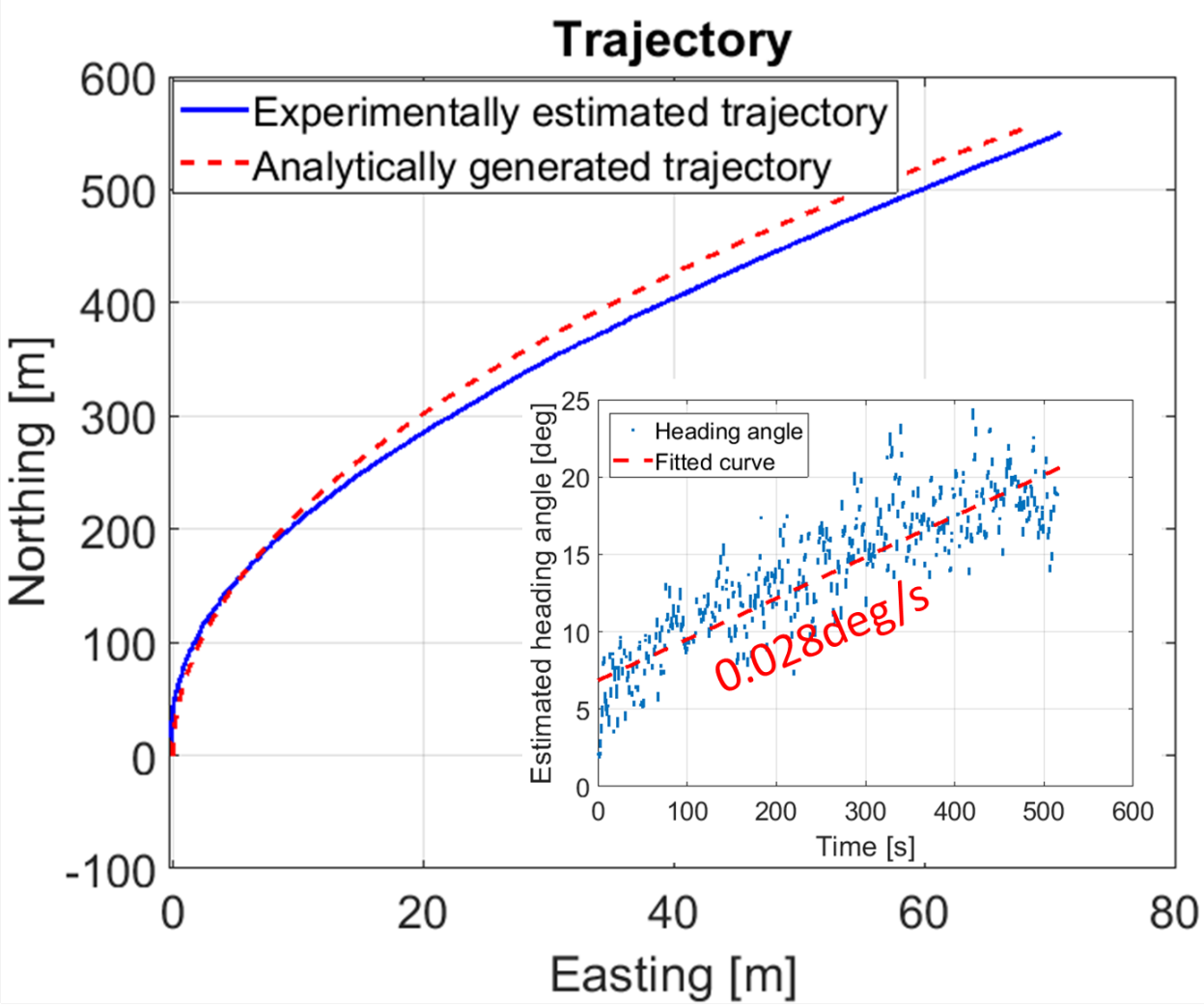

Figure 6.11: The blue solid line is an estimated trajectory, and the red dashed line is an analytically generated trajectory with heading angle increasing at a rate of 0.028 °/s. Note that the scales for the x and y axes are different. The inset shows that the estimated heading angle increases at a rate of 0.028 °/s.

To understand the reason for the heading angle drift, we fully calibrated the IMU to obtain not only the gyroscope and accelerometer biases, but also the misalignment and gyroscope

g-sensitivity. Such data were not typically provided for consumer or industrial grade IMUs. During calibration, the IMU was rigidly mounted on a tilt table to achieve different orientations, and the tilt table was mounted on a single-axis rate table (IDEAL AEROSMITH 1270VS) to generate a constant reference rotation. The experimental setup is shown in Fig. 6.12 (a). A standard IMU calibration procedure was followed [184], and the calibration results were as follows

$$b_a = \begin{bmatrix} -0.025 \\ -0.0176 \\ 0.1955 \end{bmatrix}, \qquad M_a = \begin{bmatrix} 1.0020 & -0.0083 & -0.0042 \\ 0.0055 & 0.9986 & 0.0051 \\ 0.0067 & -0.0039 & 0.9964 \end{bmatrix},$$

$$b_g = \begin{bmatrix} -0.0893 \\ 0.0375 \\ -0.0412 \end{bmatrix}, \qquad M_g = \begin{bmatrix} 0.9972 & -0.0041 & -0.0067 \\ 0.0041 & 0.9972 & 0.0052 \\ 0.0067 & -0.0027 & 1.0019 \end{bmatrix},$$

$$G_g = \begin{bmatrix} 0.0041 & 0.0002 & -0.0005 \\ 0.0002 & 0.0025 & 0.0002 \\ -0.0005 & -0.0006 & -0.0022 \end{bmatrix}.$$

where b_a is the accelerometer bias in $\mathrm{m/s^2}$, b_g is the gyroscope bias in $^\circ$/s, M_a is the accelerometer misalignment matrix, M_g is the gyroscope misalignment matrix, and G_g is the gyroscope g-sensitivity matrix in $^\circ/\mathrm{s}/(\mathrm{m/s^2})$. These values were used in the IMU readout compensation before fed into the navigation algorithm in the following studies. Note that the g-sensitivity of the gyroscope was on the order of $0.002\,^\circ/\mathrm{s}/(\mathrm{m/s^2})$. A shock on the order of $10g$, which was typical for a foot-mounted IMU, would cause a gyroscope bias of $0.2\,^\circ/\mathrm{s}/(\mathrm{m/s^2})$ and resulted in a large navigation error, if not compensated.

Note that g-sensitivity of gyroscopes was obtained in a static condition. Since the IMU would experience severe dynamics during navigation, a measurement of the gyroscope g-sensitivity

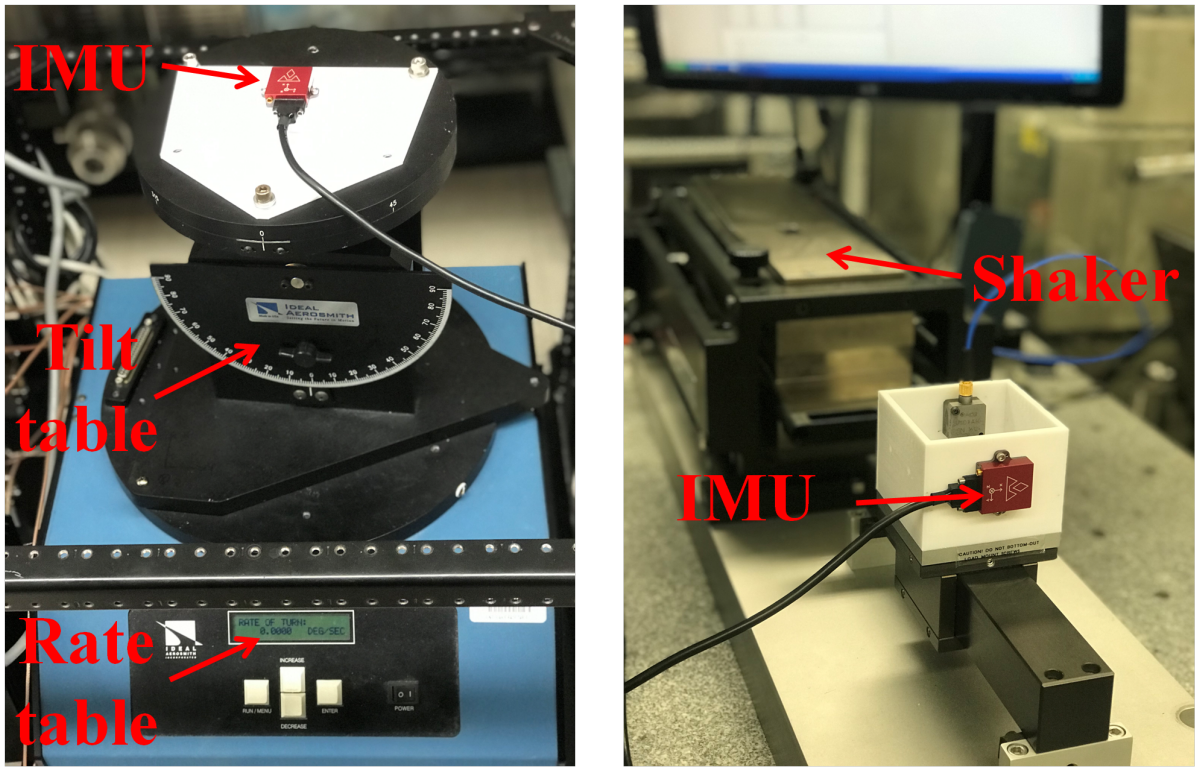

Figure 6.12: (a) Experimental setup to statically calibrate IMU; (b) experimental setup to measure the relation between gyroscope g-sensitivity and acceleration frequency.

in dynamic conditions was also necessary. To achieve it, the IMU was rigidly mounted on a shaker (APS Dynamics APS-500), and the gyroscope readouts were recorded while the shaker generated vibrations with different frequencies, ranging from 10Hz to 160Hz. Three independent measurements were conducted to guarantee repeatability. A relation between the gyroscope g-sensitivity and the vibration frequency is shown in Fig. 6.13. Gyroscope g-sensitivity remained relatively stable around $0.0022\,^{\circ}/\mathrm{s}/(\mathrm{m/s^2})$ until the vibration frequency was above 140Hz. Inset is the FFT of the z-axis accelerometer readout during a typical walking of 2min, and the spectrum reacheed close to zero with frequencies higher than 80Hz. Therefore, the gyroscope g-sensitivity could be considered constant for the whole frequency range in the case of pedestrian navigation.

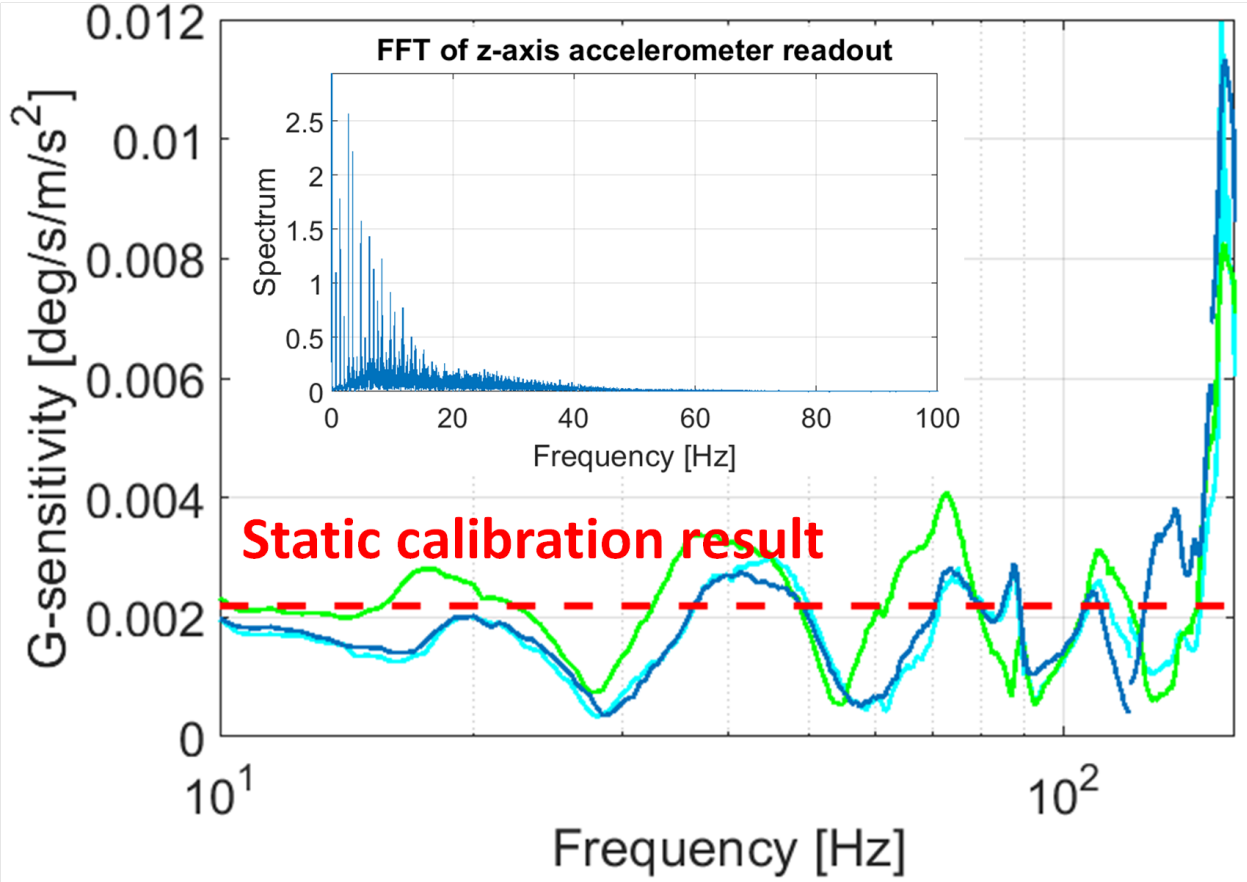

Figure 6.13: Relation between the gyroscope g-sensitivity and the vibration frequency obtained from 3 independent measurements. The red dashed line is the gyroscope g-sensitivity measured in static calibration. Inset is the FFT of the z-axis accelerometer readout during a typical walking of 2min.

6.2.2 Systematic Error Compensation Results

Two steps were taken to compensate for systematic errors identified in this paper: (1) calibrated the IMU readouts to remove the effects of sensor biases, misalignment, and especially gyroscope g-sensitivity; (2) set the pseudo-measurement of the velocity of the foot during the stance phase according to the gait pattern – characterized in Section 6.2.1.1 – instead of zero.

Experiments were conducted to verify the effects of compensation. The IMU was mounted on top of the toes. A straight line trajectory of 99.6m was used, and 40 sets of data were recorded in total. The navigation results with and without the compensation are shown in Fig. 6.14. It could be seen that the drift in trajectory orientation was compensated, while the compensation effects along the trajectory could not be seen clearly due to the scale. Fig. 6.15 shows the ending points of the 40 trajectories with and without compensation. The dashed lines are the 3σ boundaries of the results. Note that they were approximately of the same size, since they were the result of stochastic noise, which was not compensated in this section. Note that the navigation error demonstrated in Fig. 6.15 agreed with the model developed in Chapter 5, indicating that IMU noise was the dominant navigation error source. The averaged navigation error was 3.23m without any compensation (red dots in Fig. 6.15). The majority part of the navigation error perpendicular to the trajectory could be canceled by calibrating the IMU, and the averaged navigation error was reduced to 2.08m (yellow dots in Fig. 6.15). After implementing the residual velocity compensation, the navigation error along the trajectory was compensated, and the averaged navigation error was reduced to 0.31m (blue dots in Fig. 6.15), demonstrating a more than $10\times$ improvement. After systematic error compensation, the error caused by IMU noises became dominant, thus requiring the improvement in the IMU performance to improve the overall navigation accuracy.

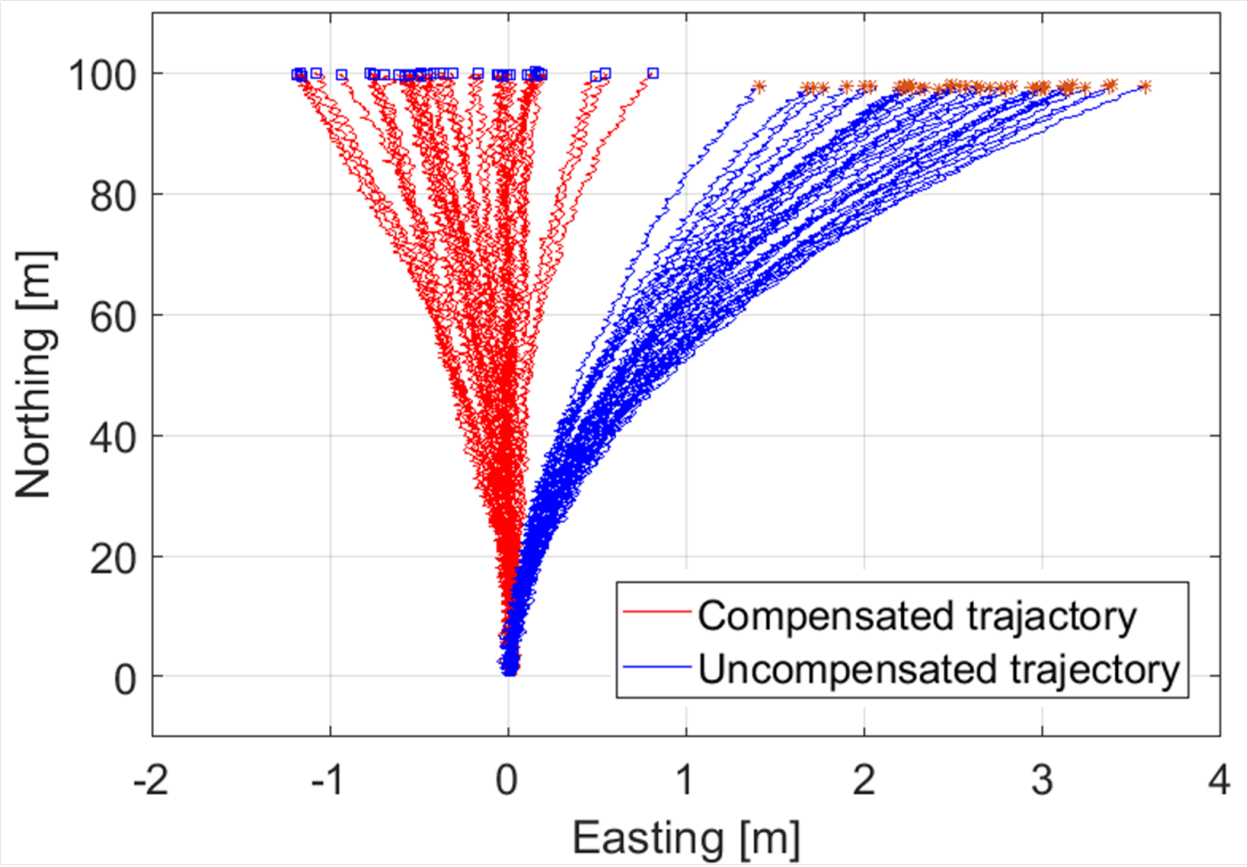

Figure 6.14: Comparison of trajectories with and without systematic error compensation. Note that the scales for x and y axes are different.

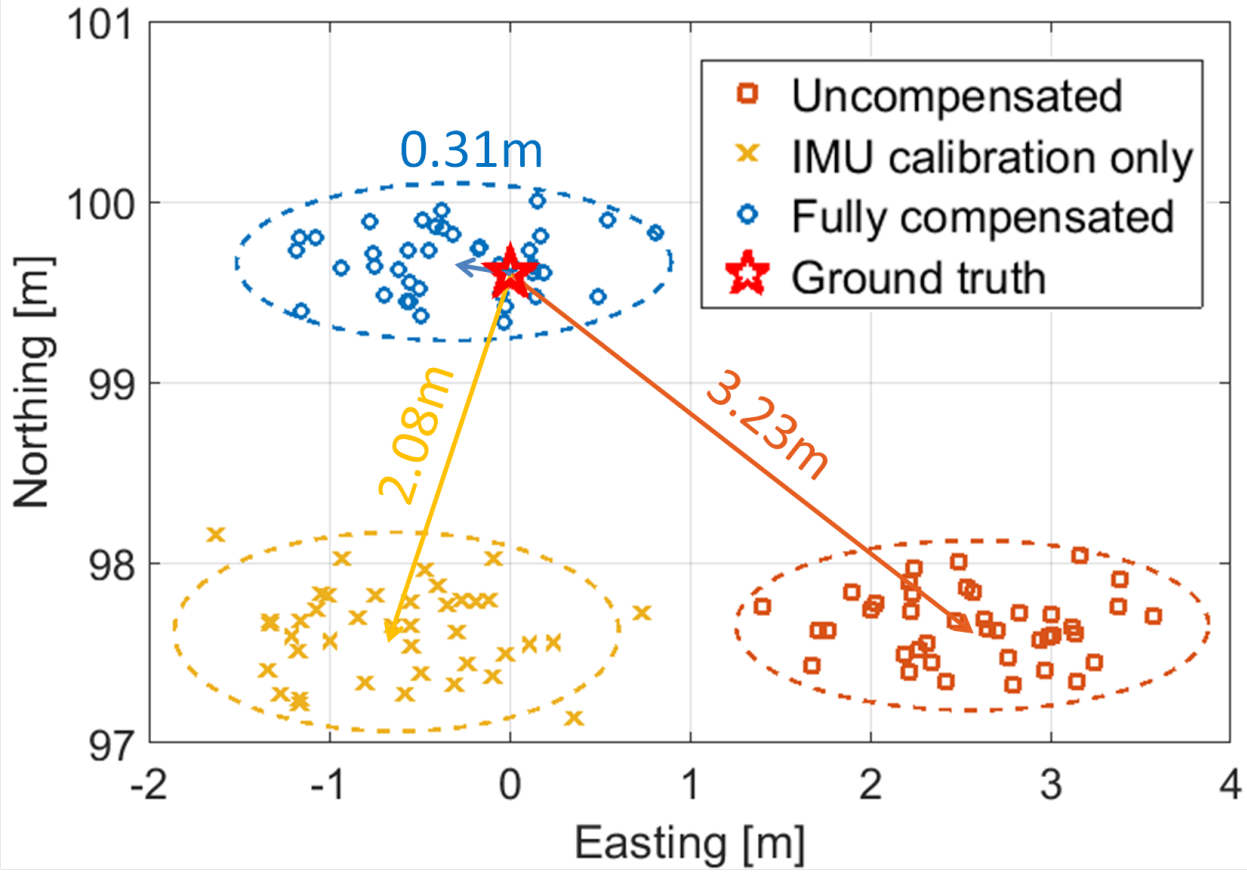

Figure 6.15: Comparison of the end points with and without systematic error compensation. The dashed lines are the 3σ boundaries of the results.

6.3 Stochastic Noise Reduction

Stochastic noise mainly comes from two sources: IMU readout noise and the random component of residual velocity of the foot during the stance phase. The first source is related to IMU performance, and methods to reduce them have been discussed in Chapter 2 and Chapter 3. However, in this chapter, we demonstrate that the second noise source can be reduced by optimizing the mounting position of the IMU on the foot.

6.3.1 Data Collection

Two identical industrial grade IMUs (VectorNav VN-200 IMU) were rigidly mounted above the forefoot and behind the heel of the boot, respectively, to collect data of the motion of the forefoot and the heel simultaneously. Noise characteristics of the two IMUs were first estimated by Allan Deviation analysis and compared to the datasheet [177]. The results are presented in Fig. 6.16, showing that the two IMUs had the same noise level, eliminating some possible discrepancies in experiment due to IMU characteristics.

With IMUs mounted on the forefoot and behind the heel of the boot, experiments were conducted by different subjects, each of which walked on different floor types at different speeds, assuring validity of conclusions. For each experiment, a trajectory with 600 strides (1200 steps) was recorded. The walking pace was fixed with the help of a metronome for a better post-processing result, but the step counter was not used in estimation of the navigation accuracy. Floor types, such as hard floor, grass lawn, and sand floor, were tested. Walking paces ranged from 84 to 112 steps per minute. Trajectories, such as walking upstairs and downstairs, were also investigated. Four different walking patterns by four different subjects were tested in this study. IMU readouts were collected during walking and then analyzed in post-processing.

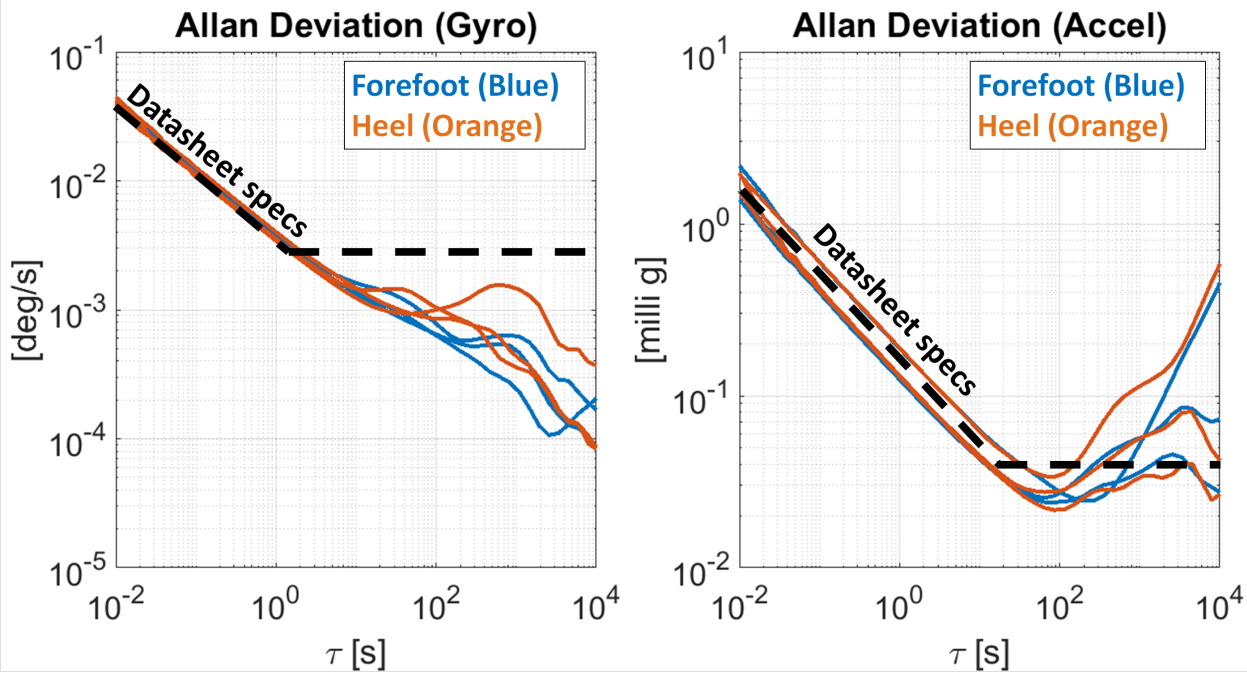

Figure 6.16: Noise characteristics of the IMUs used in this study.

6.3.2 Data Averaging

Collected IMU data from the forefoot and the heel were processed to compare the two mounting positions. IMU data were first averaged to reduce the IMU noise and extract parameters, such as length of the stance phase and the shock level during walking.

In this process, IMU data of 600 gait cycles were averaged. The main purpose of averaging was to remove the majority of the white noise for a better extraction of motion features. A zero-velocity detector was applied to the averaged data to determine the stance phase during the gait cycle. The results are shown in Fig. 6.17. On the left are the ZUPT state and the averaged IMU data from the forefoot. On the right of Fig. 6.17 are the readouts of the IMU located behind the heel. ZUPT state was derived for both mounting positions with the same threshold. For IMUs mounted on the forefoot and behind the heel, Fig. 6.17 shows an average time duration of the stance phase of 0.498s and 0.363s, and the shock experienced by IMU was on the level of 80m/s^2 and 150m/s^2, respectively. Both a

longer stance phase and a lower level of shock yielded a better navigation accuracy for the IMU mounted on the forefoot. Interruption of the stance phase for the heel showed that the IMU was moving during the stance phase, indicating a less stable stance phase if IMU was mounted behind the heel. One disadvantage of forefoot as the IMU mounting position was that the maximum gyroscope readout was about 800 °/s compared to the 450 °/s if the IMU was mounted behind the heel. In this study, the gyroscopes of IMU had a maximum measuring range of 2000 °/s, and therefore in this case, the higher magnitude of gyroscope readout was not an issue. However, the choice of IMU with a sufficient measuring range was an important consideration for this application.

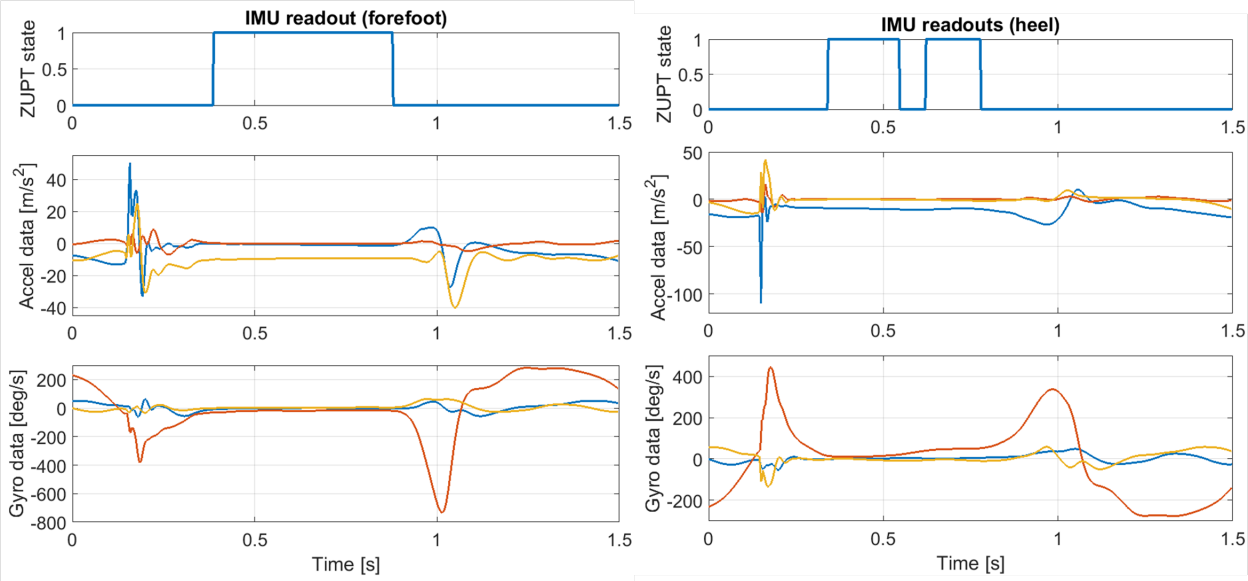

Figure 6.17: Comparison of averaged IMU data and ZUPT states from IMUs mounted on the forefoot and behind the heel. Stance phase was identified when ZUPT state was equal to 1.

Bandwidth of most MEMS-based inertial sensors was typically not high enough for foot-mounted inertial navigation due to the shock during the heel-strike [174]. However, in this study, the bandwidth of the sensors was 250Hz, and the effects of the limited bandwidth might be negligible [185].

6.3.3 Velocity Uncertainty Determination

After collecting the IMU data in the previous section, in this section, we studied the distribution of the IMU data to understand the noise introduced in the zero-velocity update during the stance phase. More specifically, the velocity uncertainty during the stance phase was studied in this section.

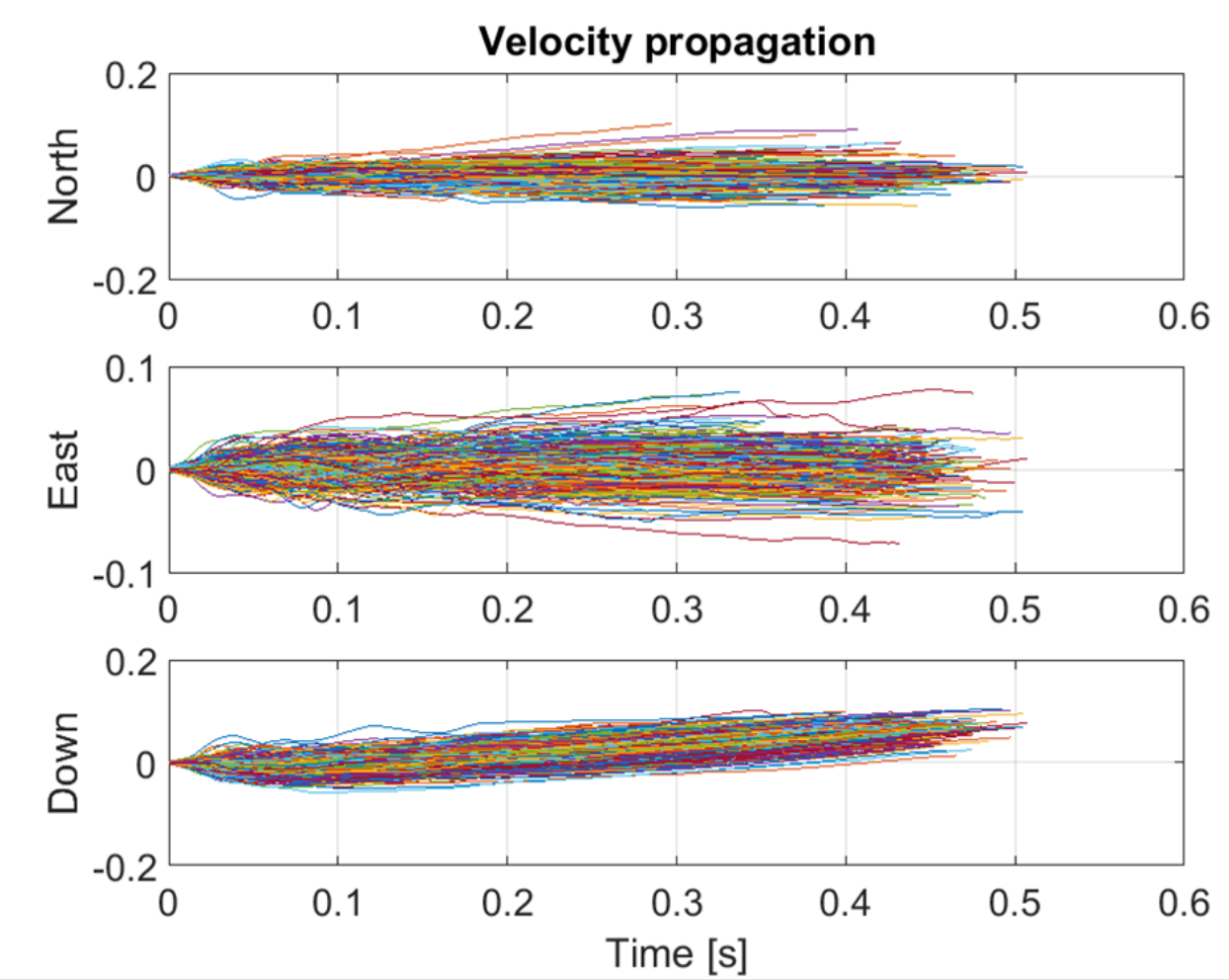

Figure 6.18: Velocity propagation along three orthogonal directions during the 600 stance phases.

To estimate the velocity uncertainty during the stance phase, we first implemented the ZUPT-aided inertial navigation algorithm to the IMU data to estimate the trajectory and to extract the stance phases. Next, a free strapdown navigation algorithm was applied to

the IMU readouts during the stance phases, assuming a zero initial velocity. The initial orientation of the IMU during the stance phases was obtained from previous navigation results. Velocity propagations during the stance phases are shown in Fig. 6.18. The average length of the stance phase in this case was around 0.48s, which was comparable to results discussed in Section 6.3.2. Finally, the distribution of the final velocity was analyzed (Fig. 6.19), and its standard deviation was calculated to be 0.016m/s and 0.022m/s for the IMU mounted on the forefoot and behind the heel, respectively. These values also matched the results shown in Fig. 6.8. Note that the bias of the final velocity distribution along the Down direction was about 0.06m/s; this was probably due to the velocity difference at the beginning and at the end of the stance phases. The standard deviation was set to be the velocity uncertainty during the stance phase. A lower velocity uncertainty for the forefoot showed an advantage of the forefoot as the IMU mounting position. The final velocity distribution was of a bell shape, indicating that it could be approximated as a Gaussian distribution. Note that the focus in this section was only on the stochastic component of the motion of the foot during the stance phase. As a result, the value of the assumed initial velocity for each stance phase could be arbitrarily selected, and it would not affect the distribution of final velocity of the stance phase.

Next, we calculated the velocity uncertainty caused by accelerometer's and gyroscope's white noise [10]

$$\Delta v_{\text{accel}} = VRW \cdot \sqrt{t} \approx 1 \times 10^{-3} m/s, \tag{6.8}$$

$$\Delta v_{\text{gyro}} = \sqrt{\frac{1}{3}} ARW \cdot g \cdot t^{3/2} \approx 8 \times 10^{-5} m/s, \tag{6.9}$$

where Δv_{accel} is the velocity uncertainty caused by accelerometer's white noise, Δv_{gyro} is the velocity uncertainty caused by gyroscope's white noise, t is the length of the stance phase, and g is the gravity. According to (6.8) and (6.9), the extracted velocity uncertainty during the stance phase was not related to the IMU noises, since the noise was orders of magnitude

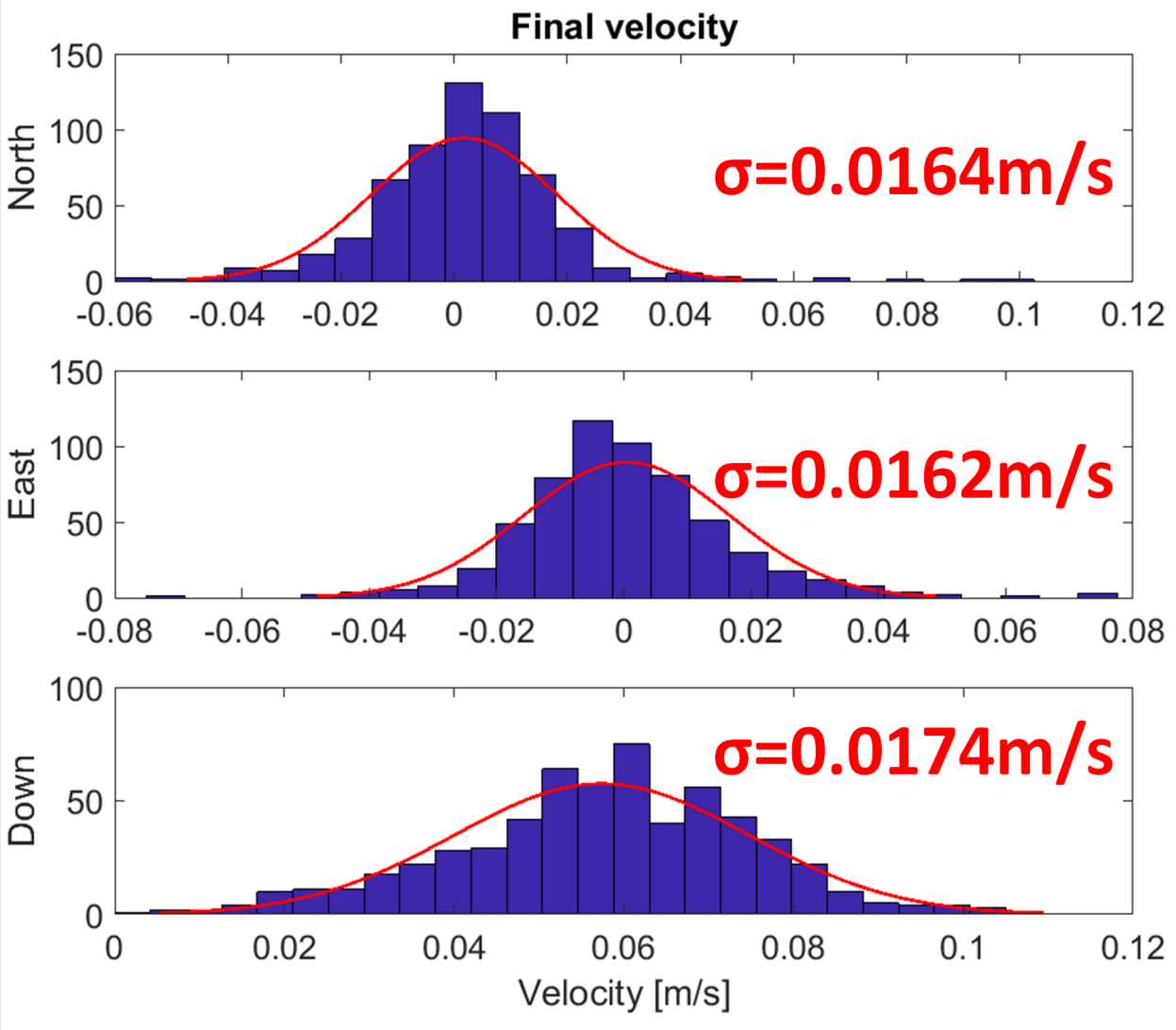

Figure 6.19: Distribution of the final velocity along three orthogonal directions during 600 stance phases. Standard deviation was extracted as the average velocity uncertainty during the stance phase.

Table 6.1: Stance Phase Analysis Summary with Different Floor Types

Floor Type	**Step Pace [step/min]**	**Velocity Uncertainty [m/s]**		**Stance Phase Length [s]**	
		Heel	*Forefoot*	*Heel*	*Forefoot*
Hard Floor	84	0.022	0.016	0.36	0.50
	100	0.025	0.020	0.33	0.39
	112	0.029	0.024	0.29	0.34
Grass Lawn	84	0.046	0.032	0.48	0.55
	100	0.052	0.035	0.38	0.45
	112	0.055	0.045	0.34	0.39
Sand	84	0.060	0.048	0.51	0.48
	100	0.076	0.050	0.37	0.37
	112	0.095	0.051	0.31	0.32

lower than what was needed to introduce velocity uncertainty on the order of 0.01m/s.

6.3.4 Data Processing Summary

The stance phase analysis with different step paces and floor types is presented in Table 6.1. The velocity uncertainty during the stance phase was highest when walking on the sand, then on the grass lawn, and it was lowest when walking on the hard floor. This result was expected, since more uncertainty would be induced when walking on softer floors. The same conclusion was drawn in all nine scenarios that the forefoot was a better IMU mounting position than the heel with an average of 20% lower velocity uncertainty and about 20% longer stance phase period.

Table 6.2 lists the analysis of walking upstairs and downstairs, while Table 6.3 lists the analysis of walking on a hard floor by different subjects (different walking patterns). A similar result was obtained for all cases, concluding that a longer stance phase and a lower velocity uncertainty should be expected to be achieved with IMU mounted on the forefoot.

Table 6.2: Stance Phase Analysis Summary with Different Trajectories

Trajectory	**Step Pace [step/min]**	**Velocity Uncertainty [m/s]**		**Stance Phase Length [s]**	
		Heel	*Forefoot*	*Heel*	*Forefoot*
Upstairs	84	0.086	0.042	0.51	0.53
	100	0.088	0.038	0.39	0.45
	112	0.086	0.029	0.33	0.39
Downstairs	84	0.084	0.055	0.48	0.58
	100	0.083	0.042	0.31	0.42
	112	0.080	0.038	0.27	0.30

Table 6.3: Stance Phase Analysis Summary with Different Subjects

Subject Number	**Step Pace [step/min]**	**Velocity Uncertainty [m/s]**		**Stance Phase Length [s]**	
		Heel	*Forefoot*	*Heel*	*Forefoot*
Subject 1	84	0.022	0.016	0.36	0.50
	100	0.025	0.020	0.33	0.39
	112	0.029	0.024	0.29	0.34
Subject 2	84	0.044	0.028	0.49	0.50
	100	0.040	0.026	0.29	0.31
	112	0.034	0.020	0.27	0.27
Subject 3	84	0.052	0.035	0.48	0.52
	100	0.071	0.026	0.34	0.37
	112	0.022	0.020	0.28	0.30
Subject 4	84	0.029	0.026	0.38	0.42
	100	0.040	0.034	0.28	0.33
	112	0.031	0.021	0.27	0.28

6.3.5 Experimental Verification

Next, a direct correlation between the IMU mounting position and the navigation accuracy was presented to compare the forefoot and the heel as IMU mounting positions.

A circular path with a diameter of 8m and 10 laps was used as a trajectory to experimentally demonstrate the effect of the IMU mounting position on the Circular Error Probable (CEP). The trajectory was a close loop to facilitate the extraction of navigation position errors. Thirty-four tests were recorded with an average navigation time of 260s, and the navigation errors for all tests are presented in Fig. 6.20. The CEP was reduced from 1.79m to 0.96m

by using the data from the IMU mounted on the forefoot instead of behind the heel.

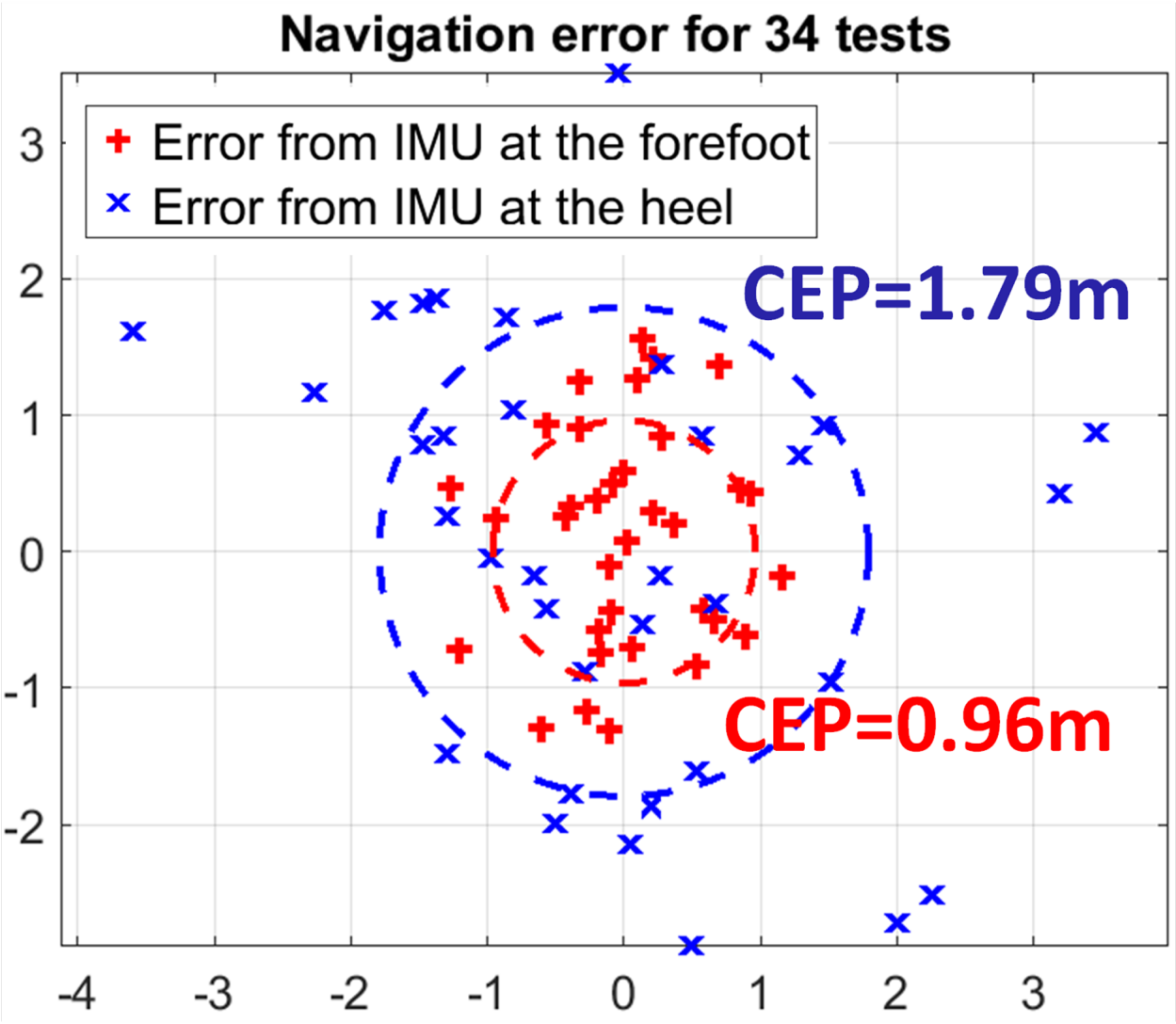

Figure 6.20: Navigation error of 34 tests of the same circular trajectory.

A comparison of estimation results from IMU mounted at the forefoot and the heel is presented in Fig. 6.21. It could be clearly observed that the estimated trajectory was smoother with IMU mounted at the forefoot, and it was due to a smaller position correction during the stance phase, indicating less noise accumulated during the swing phases. To better interpret the performance of the EKF, we characterized the innovation of the EKF. Innovation, or measurement residual, is defined as the difference between the measurement and its prediction [186]. In the case of the ZUPT-aided pedestrian inertial navigation, innovation is the

difference between estimated velocity of the foot during the stance phases and the pseudo-measurement of the velocity of the foot. The distribution of the innovation from the same experiment is shown in the lower part of Fig. 6.21. A lower innovation covariance was shown in the data from the IMU mounted at the forefoot, indicating a better navigation accuracy. Note that some innovation values were outside of 3σ envelop, and they were probably related to the false alarm in the ZUPT detection.

6.4 Combination of Compensation Methods

In this section, the aforementioned three compensation methods were combined together to achieve optimal navigation performance. The effects of each compensation method were also compared. A closed-loop trajectory was selected with the end point being the same as the start point. The trajectory length was about 880m, and the total navigation time was about 12.5 minutes, including 1 minute of running and 11.5 minutes of walking. The altitude change during the navigation was about 8m. Two identical industrial grade IMUs were fixed at the forefoot and the heel, respectively. The navigation results with different compensation methods are presented in Fig. 6.22. Due to a lack of ground truth, instead of RMSE, the final position error was used to characterize the navigation results, which is listed in Table 6.4. The three rows list the final position errors with no compensation, adaptive threshold for ZUPT detection only, and all compensations; and they correspond to the dotted lines, dashed lines, and solid lines in Fig. 6.22, respectively. The two columns present the final position errors using data from the forefoot-mounted IMU and heel-mounted IMU, corresponding to the blue lines and red lines in Fig. 6.22, respectively. Both columns show smaller final position errors with adaptive stance phase detection, residual velocity compensation, and gyroscope g-sensitivity compensation implemented, indicating the effectiveness of the proposed compensation methods. Note that the values in the first column are of the same

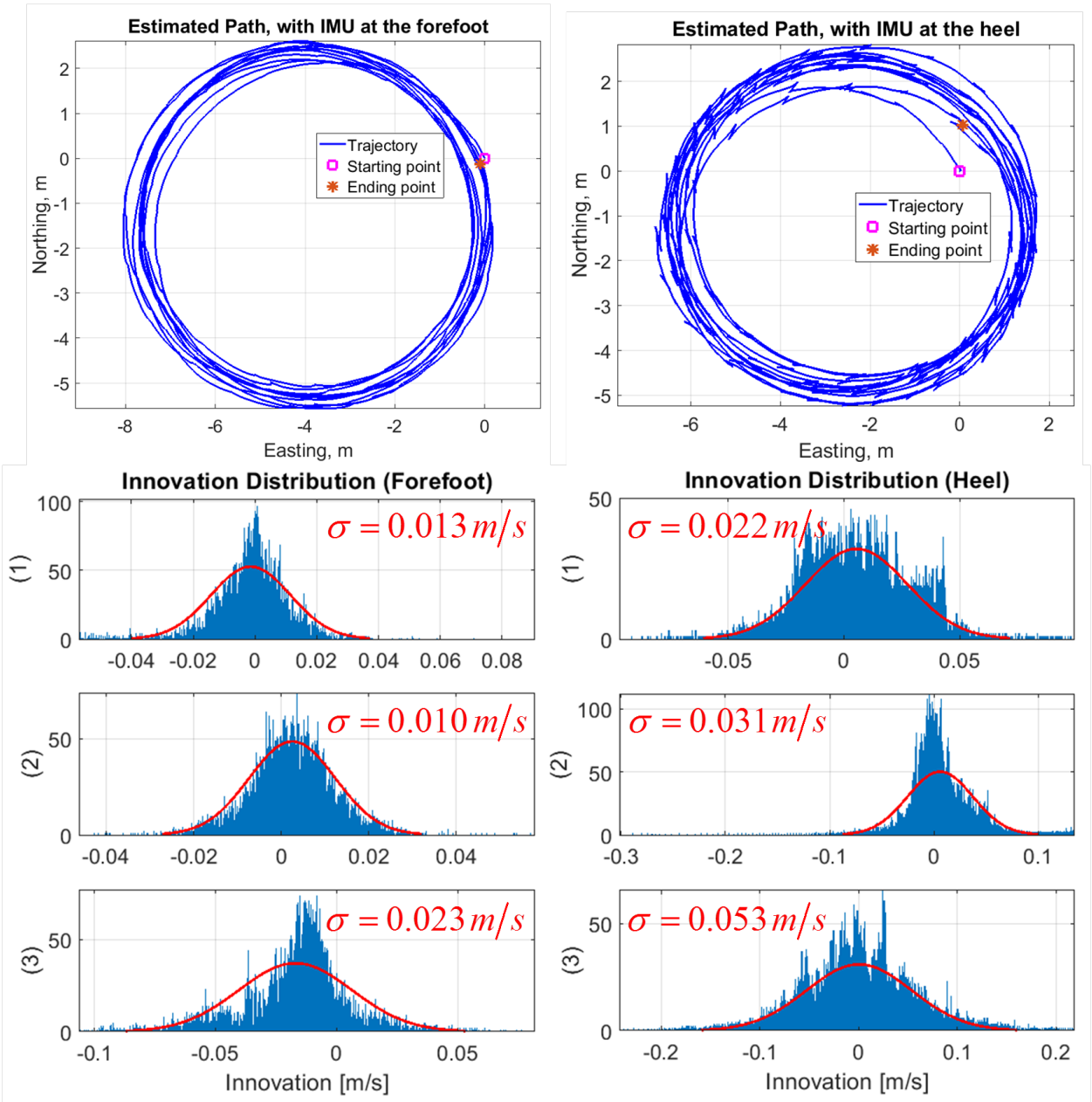

Figure 6.21: Comparison of estimated trajectories and innovations from IMU mounted at the forefoot (left) and the heel (right).

order or smaller than the values in the second column. We believe it was related to a smaller stochastic noise with the IMU mounted at the forefoot. The lowest navigation error of 7.5m was achieved by the forefoot-mounted IMU with all compensation methods implemented. A more than 10× navigation error reduction was demonstrated compared with the error of 76.7m achieved by a fixed threshold for ZUPT detection and without any compensation.

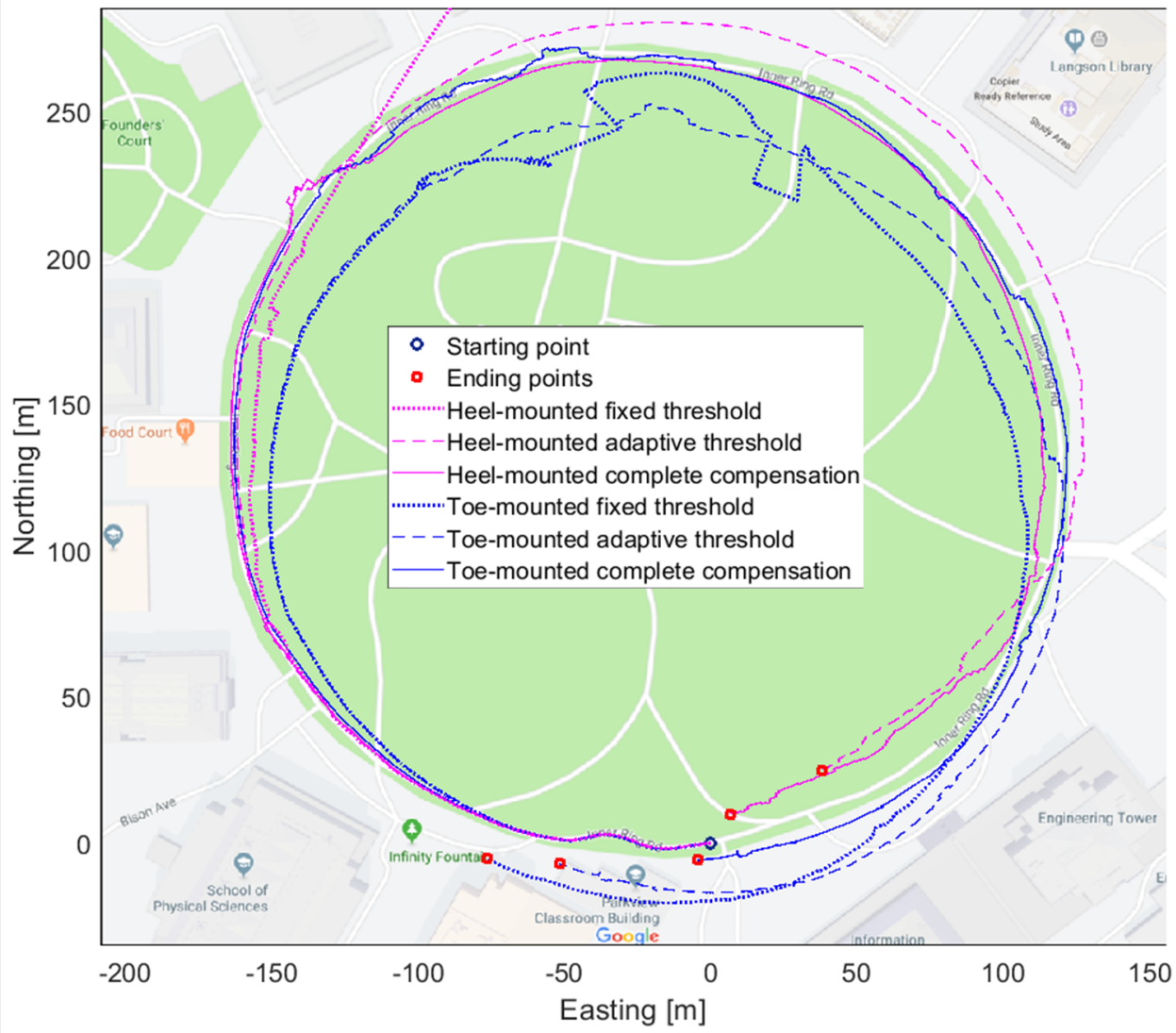

Figure 6.22: Navigation results with different IMU mounting positions and compensation implementations.

Table 6.4: Final Navigation Errors with Different Compensations

	Forefoot	Heel
Inertial navigation	192km	53km
ZUPT only	76.7m	3667m
Adaptive ZUPT only	52.2m	45.4m
All compensations	7.5m	11.9m

6.5 Conclusions

In this chapter, we presented a comprehensive study on reduction of navigation errors, both systematic and stochastic, in the ZUPT-aided pedestrian inertial navigation without adding extra sensing modalities. Adaptive threshold for the stance phase detection improved both the accuracy and the robustness of the navigation algorithm, especially in cases where varying walking or running speed was involved. Compensation of residual velocity during the stance phase and gyroscope g-sensitivity were demonstrated to reduce the systematic error over 10×. Mounting the IMU on the forefoot instead of on the heel was shown to reduce the stochastic error, or equivalently CEP, by 50%. A combination of all the aforementioned techniques was demonstrated in a complex environment, involving walking, running, uphills, and downhills, with total reduction of navigation error over 10×, showing a great improvement of navigation accuracy and robustness of the developed compensation algorithms.

This work eliminates the majority of the errors caused by the ZUPT implementation, which would otherwise be the dominant error source in the ZUPT-aided pedestrian inertial navigation. The IMU error becomes dominant factor in the navigation error after implementing all the compensation methods developed in this chapter, providing the possibility of further reducing the navigation error by improving the performance of inertial sensors. Therefore, the work reported in this chapter is envisioned as a foundation of further navigation system development, including improvement of inertial sensor performance, and addition of other aiding mechanisms to the system, such as altimeter, ultra-sonic ranging, signals of opportunities, and cooperative localization.

Chapter 7

Conclusions

Toward the goal of high accuracy pedestrian inertial navigation, this dissertation presents both inertial sensor development and navigation algorithm development in an effort to address the challenges associated with imperfection-induced errors in gyroscopes as well as the accumulation of sensor errors in the inertial navigation. The majority of work on the sensor development was toward compensation of fabrication imperfections in the Fused Quartz 3D wineglass resonator gyroscopes, with the goal of reducing as-fabricated structural asymmetry and energy dissipation. A methodology to identify, decouple, and quantify different energy dissipation mechanisms in MEMS resonators was developed. On the algorithm development side, an analytical model to quantify the effects of IMU errors in the ZUPT-aided pedestrian inertial navigation was developed as the basis. Navigation error reduction methods were thoroughly explored to eliminate the majority of the system-induced errors, providing the possibility of further reducing the navigation error by improving the performance of inertial sensors. Specific contributions of the dissertation are summarized below.

7.1 Contributions of the Dissertation

1. Analytical and numerical models to predict the resonant frequency and frequency mismatch of Fused Quartz 3D wineglass resonators with error less than 20% was developed. Mode shapes of all wineglass modes were first calculated based on inextentional assumption, and the resonant frequencies were extracted by Rayleigh's method. Based on this model, major factors that affect the structural asymmetry, including non-uniform temperature during glassblowing and misaligned lapping, were analyzed. This modeling work laid a foundation for the following imperfection compensation efforts.

2. Frequency split reduction of 6× from 41Hz to 7Hz on Fused Quartz 3D wineglass resonators by directional lapping was reported with the potential of not affecting the overall quality factor of the device. A method to identify the principal axes of elasticity was first implemented by measuring the mode shape using Laser Doppler Vibrometer. Then, directional lapping was conducted along the identified axes to compensated for as-fabricated structural asymmetry. Numerical simulation confirmed that the TED and the anchor loss would not be changed by the directional lapping procedure, while the fact that the surface roughness of the rim indicated that the surface loss was not increased either. The directional lapping procedure is able to greatly reduce the initial frequency asymmetry of MEMS devices, and therefore only relatively low tuning voltages are required to accurately tune the devices, and the circuit-induced noise in MEMS resonators and gyroscopes can be reduced.

3. Energy dissipation reduction due to surface roughness improvement from 24.5nm Sa to 1.9nm Sa on Fused Quartz 3D wineglass resonators was reported. The effects of chemical and thermal post-processing on the surface quality of fused quartz samples were studied, including thermal reflow, potassium hydroxide (KOH) etching, Buffered Oxide Etching (BOE), 10:1 Hydrogen Fluoride and Hydrogen Chloride solution (HF/HCl) etching, and RCA-1 surface treatment. Thermal reflow at 1300 °C for 1 hour showed

the best result, and an improved quality factor of the device was correlated to the smoother surface.

4. Piezoelectric actuation on Fused Quartz dual shell resonators was proposed and developed, in order to avoid the metal deposition on the device surface and keep the high quality factor of the Fused Quartz devices. Piezo electrode geometry was first optimized to achieve highest transduction rate, and then the frequency response of the device was simulated, predicting a resonant with amplitude of 2μm with 1V AC voltage applied to PZT layer for $n = 2$ mode. As a key part in patterning the piezo electrode on the shell surface, shadow mask method was proposed and experimentally demonstrated to selectively deposit metal layer on curved surfaces.

5. For the first time, methodology to analyze and quantify different energy dissipation mechanisms in MEMS resonators was demonstrated. Toroidal Ring Gyroscopes (TRG) were designed and characterized as the test platform. At room temperature, the quality factor related to viscous air damping, TED, and anchor loss was experimentally measured to be 625,000, 200,000, and 1,350,000, respectively. The effects of moisture-related surface loss were estimated and eliminated by high-temperature back-out process. Relation between the anchor loss and electrostatic tuning was also explored. We experimentally demonstrated that unbalanced electrostatic tuning would increase the anchor loss, while the balanced tuning would not affect the anchor loss. This study provides more insight to better understand the dominant mechanism that limits the quality factors in MEMS resonators, in order to effectively improve the design and achieve better device performances.

6. An analytical model to relate the IMU error to the navigation error in the ZUPT-aided pedestrian inertial navigation was presented with error less than 20%. A bio-mechanical model of human gait was first developed. The analytically derived attitude, velocity, and position propagation errors reveal that among many IMU noise terms, the dom-

inant factor affecting the accuracy of the ZUPT-aided pedestrian inertial navigation was Rate Random Walk (RRW) of the z-axis gyroscope. Simulation and experiments were conducted to confirm the analytical results. This study is envisioned to aid in analysis of the effect of errors in sensors, which might lead to a well informed selection of sensors for the task of the ZUPT-aided pedestrian inertial navigation.

7. Navigation error reduction of more than 10× in the ZUPT-aided pedestrian inertial navigation was demonstrated in complex navigation environment (walking, running, uphills, and downhills) without adding any extra sensing modalities. The adaptive threshold for ZUPT detection helped to improve both the accuracy and the robustness of the algorithm, especially in complicated environment with varying walking and running speeds. In addition, as the two major sources of systematic errors, residual velocity of the foot during the stance phase and gyroscope g-sensitivity were identified and compensated. A more than 10× reduction of systematic errors was experimentally demonstrated, which otherwise would be even greater than the error caused by the IMU noise. Furthermore, forefoot and heel were compared as IMU mounting positions, and a nearly 2× reduction of the stochastic navigation error was demonstrated by mounting the IMU on top of the forefoot instead of behind the heel.

7.2 Future Research Directions

Optical Detection of Dual Shell Resonator

In Chapter 3, piezoelectric actuation of dual shell resonator was proposed and explored on Fused Quartz dual shell resonators, in order to eliminate metal deposition of device surface to preserve the high quality factor of the devices. However, sensing architecture has not been explored. Optical sensing seems to be a natural choice to avoid metal deposition.

The technique reported in [187] could be adapted for the dual shell resonators, with the ability to measure the amplitude of vibration of the shell from multiple points simultaneously with a resolution on the order of 1nm. The multiple-point measurement may provide more information about the motion of the device, thus the possibility of a better control the device to operate in the whole angle mode.

Frequency Tuning of Dual Shell Resonator

Frequency tuning is necessary for mode-matched gyroscopes in order to achieve small frequency mismatch between the degenerated modes. However, electrostatic tuning of Fused Quartz devices also requires metal deposition for electrical conductivity. Lead Zirconate Titanate (PZT) piezoelectric frequency tuning is a possible alternative due to the high piezoelectric coefficient of the PZT material. Possible PZT deposition locations include the stem and the cap shell. Optimization of electrode design is necessary to maximize the tuning effects and tuning flexibility.

Whole Angle Operation of Dual Shell Resonator

After development of piezoelectric actuation, optical detection, and piezoelectric tuning on the dual shell resonator, whole angle operation can be implemented to take advantage of the high quality factor of the devices. Navigation grade gyroscope performance is expected with the quality factor on the order of millions and the frequency split lower than 0.1Hz.

Machine Learning on Parameter Identification

In Chapter 6, adaptive threshold was developed for stance phase detection with different walking speeds. However, there are many other factors that affect the parameters used in the ZUPT-aided pedestrian inertial navigation algorithm, such as floor types, the weight and size of the subject, and gait patterns. These factor will affect the threshold needed to identify the stance phase, the residual velocity during the stance phase, and the velocity uncertainty of the foot during the stance phase. Therefore, a more comprehensive study is necessary to develop a uniformly applicable navigation algorithm. Machine learning approach can be adapted for identification of gait patterns, gait speeds, and floor types. The robustness of the algorithm can be further improved by automatically identifying different combinations of gait parameters.

Ultimate Navigation Chip

In this dissertation, only IMU was explored as the sensing modality for pedestrian inertial navigation. However, as mentioned in Chapter 5, some navigation states, such as yaw angle, is not observable in such implementations. Therefore, in order to further improve the navigation accuracy, sensor fusion technique can be utilized, incorporating other sensing mechanisms, such as barometer, magnetometer, ultrasonic ranging, computer vision, signals of opportunities, and cooperative localization. Navigation error less than 1m is projected during navigation for few hours, if the information from different sensing modalities can be incorporated properly. Such a pedestrian navigation system is envisioned to be miniaturized to the size of an apple seed by micro-fabrication technologies.

Bibliography

[1] N. Bowditch, *The American Practical Navigator*, 2002 Bicentennial Edition, National Imagery and Mapping Agency, Bethesda, Maryland.

[2] M. Kourogi, N. Sakata, T. Okuma, and T. Kurata, "Indoor/outdoor pedestrian navigation with an embedded GPS/RFID/self-contained sensor system," *International Conference on Artificial Reality and Telexistence*, Springer, Berlin, Heidelberg, 2006, pp. 1310-1321.

[3] X. Yun, E. R. Bachmann, H. Moore, and J. Calusdian, "Self-contained position tracking of human movement using small inertial/magnetic sensor modules," *IEEE International Conference on Robotics and Automation*, Apr. 10-14, 2007, Rome, Italy, pp. 2526-2533.

[4] M. Perlmutter, and L. Robin, "High-performance, low cost inertial MEMS: A market in motion!" *IEEE/ION Position, Location and Navigation Symposium*, Apr. 23-26, 2012, Myrtle Beach, SC, USA.

[5] D. M. Rozelle, "The hemispherical resonator ryro: from wineglass to the planets," *Proc. 19th AAS/AIAA Space Flight Mechanics Meeting*, Savannah, Georgia, USA, February 8-12, 2009.

[6] D. Titterton, and J. Weston, *Strapdown inertial navigation technology*, 2nd ed., vol. 207. AIAA, 2004, pp. 30.

[7] J. Wagner, and A. Trierenberg, "The machine of Bohnenberger: Bicentennial of the gyro with cardanic suspension," *Proceedings in Applied Mathematics and Mechanics*, vol. 10, no. 1, pp. 659-660, 2010.

[8] I. P. Prikhodko, S. A. Zotov, A. A. Trusov, and A. M. Shkel, "Foucault pendulum on a chip: Rate integrating silicon MEMS gyroscope," *Sensors and Actuators A: Physical*, Vol. 177, pp. 67-78, 2012.

[9] D. Tazartes, "An historical perspective on inertial navigation systems," *IEEE International Symposium on Inertial Sensors and Systems (ISISS)*, Feb. 25-26, 2014, Laguna Beach, CA, USA.

[10] O. J. Woodman, *An introduction to inertial navigation*, No. UCAM-CL-TR-696. University of Cambridge, Computer Laboratory, 2007.

[11] H. C. Lefevre, *The fiber-optic gyroscope*, 2nd ed., Artech house, 2014.

[12] V. Passaro, A. Cuccovillo, L. Vaiani, M. De Carlo, and C. E. Campanella, "Gyroscope technology and applications: A review in the industrial perspective," *Sensors* vol. 17, no. 10, pp. 2284, 2017.

[13] N. Yazdi, F. Ayazi. and K. Najafi, "Micromachined inertial sensors," *Proceedings of the IEEE*, vol. 86, no. 8, pp. 1640-1659, 1998.

[14] M. G. Petovello, M. E. Cannon, and G. Lachapelle, "Benefits of Using a Tactical-Grade IMU for High-Accuracy Positioning," *Navigation* vol. 51, no. 1, pp. 1-12, 2004.

[15] Bosch BMI160 Datasheet. https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BMI160-DS000.pdf

[16] StMicroelectronics ISM330DLC Datasheet. https://www.st.com/resource/en/datasheet/ism330dlc.pdf

[17] InvenSense ICM-42605 Datasheet. http://www.invensense.com/wp-content/uploads/2019/04/DS-ICM-42605v1-2.pdf

[18] Analog Devices ADI16495 Datasheet. https://www.analog.com/media/en/technical-documentation/data-sheets/ADIS16495.pdf

[19] Honeywell HG1930 Datasheet. https://aerospace.honeywell.com/en/~/media/aerospace/files/brochures/n61-1637-000-000-hg1930inertialmeasurementunit-bro.pdf

[20] Systron Donner SDI500 Datasheet. https://www.systron.com/sites/default/files/965755_m_sdi500_brochure_0.pdf

[21] Northrop Grumman LN-200S Datasheet. https://www.northropgrumman.com/Capabilities/LN200sInertial/Documents/LN200S.pdf

[22] Safran PRIMUS 400 Datasheet. https://www.safran-electronics-defense.com/security/navigation-systems

[23] Honeywell HG9900 Datasheet. https://aerospace.honeywell.com/en/~/media/aerospace/files/brochures/n61-1638-000-000-hg9900inertialmeasurementunit-bro.pdf

[24] GEM elettronica IMU-3000 Datasheet. http://www.gemrad.com/imu-3000/

[25] V. Passaro, A. Cuccovillo, L. Vaiani, M. De Carlo, and C. E. Campanella, "Gyroscope technology and applications: A review in the industrial perspective," *Sensors*, vol. 17, no. 10, pp. 2284, 2017.

[26] C. Acar, and A. M. Shkel, *MEMS vibratory gyroscopes: structural approaches to improve robustness*, Springer Science & Business Media, 2008.

[27] S. Nasiri, "A critical review of MEMS gyroscopes technology and commercialization status," InvenSense, 2005.

[28] A. A. Trusov, G. Atikyan, D. M. Rozelle, A. D. Meyer, S. A. Zotov, B. R. Simon, and A. M. Shkel, "Flat is not dead: Current and future performance of Si-MEMS quad mass gyro (QMG) system," *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, May 5-8, 2014, Monterey, CA, USA, pp. 252-258.

[29] H. Johari, and F. Ayazi, "High-frequency capacitive disk gyroscopes in (100) and (111) silicon," *IEEE 20th International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 21-25, 2007, Hyogo, Japan, pp. 47-50.

[30] N. El-Sheimy, H. Hou, and X. Niu, "Analysis and modeling of inertial sensors using Allan variance," *IEEE Transactions on instrumentation and measurement*, vol. 57, no. 1, pp. 140-149, 2008.

[31] *IEEE Std 962-1997 (R2003) Standard Specification Format Guide and Test Procedure for Single-Axis Interferometric Fiber Optic Gyros, Annex C.* IEEE, 2003.

[32] A. M. Shkel, "Type I and type II micromachined vibratory gyroscopes," *IEEE/ION Position, Location, And Navigation Symposium (PLANS)*, Apr. 25-27, 2006, Coronado, CA, USA, pp. 586-593.

[33] D. D. Lynch, "Vibratory Gyro Analysis by The Method of Averaging," *Saint Petersburg Conference on Gyroscopic Technology and Navigation*, May 24-25, 1995, Saint Petersburg, Russia, pp. 26-34.

[34] P. F. Jopling, and W. A. Stameris. "Apollo guidance, navigation and control-Design survey of the Apollo inertial subsystem," 1970.

[35] Wikipedia: Inertial Measurement Unit. https://en.wikipedia.org/wiki/Inertial_measurement_unit.

[36] E. Foxlin, "Pedestrian tracking with shoe-mounted inertial sensors," *IEEE Computer Graphics and Applications*, vol. 25, no. 6, pp. 38-46, 2005.

[37] M. Grimmer, K. Schmidt, J. E. Duarte, L. Neuner, G. Koginov, and R. Riener, "Stance and swing detection based on the angular velocity of lower limb segments during walking," *Frontiers in Neurorobotics*, vol. 13, pp. 57, 2019.

[38] Y. Geng, R. Martins, and J. Sousa, "Accuracy analysis of DVL/IMU/magnetometer integrated navigation system using different IMUs in AUV," *IEEE ICCA 2010*, June 9-11, 2010, Xiamen, China, pp. 516-521.

[39] F. Goldenberg, "Geomagnetic navigation beyond the magnetic compass," *IEEE/ION Position, Location, and Navigation Symposium*, Apr. 25-27, 2006, Coronado, CA, USA, pp. 684-694.

[40] M. Kourogi, and T. Kurata, "Personal positioning based on walking locomotion analysis with self-contained sensors and a wearable camera," *IEEE/ACM International Symposium on Mixed and Augmented Reality*, Oct. 7-10, 2003, Tokyo, Japan, pp. 103.

[41] S. Thrun, "Simultaneous localization and mapping," *Robotics and cognitive approaches to spatial mapping*, Springer, Berlin, Heidelberg, 2007, pp. 13-41.

[42] M. Romanovas, V. Goridko, L. Klingbeil, M. Bourouah, A. Al-Jawad, M. Traechtler, and Y. Manoli, "Pedestrian indoor localization using foot mounted inertial sensors in combination with a magnetometer, a barometer and RFID," *Progress in Location-Based Services*, Springer, Berlin, Heidelberg, 2013, pp. 151-172.

[43] J. Parviainen, L. Kantola, and J. Collin, "Differential barometry in personal navigation," *2008 IEEE/ION Position, Location and Navigation Symposium*, May 6–8, 2008, Monterey, CA, USA, pp. 148-152.

[44] M. Laverne, M. George, D. Lord, A. Kelly, and T. Mukherjee, "Experimental validation of foot to foot range measurements in pedestrian tracking," *ION GNSS Conference*, Sep. 19-23, 2011, Portland, OR, USA.

[45] F. Olsson, J. Rantakokko, and J. Nygards, "Cooperative localization using a foot-mounted inertial navigation system and ultrawideband ranging," *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct. 27-30, 2014, Busan, Korea.

[46] Antonelli, G., Arrichiello, F., Chiaverini, S., and Sukhatme, G. S., "Observability analysis of relative localization for AUVs based on ranging and depth measurements," *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, May 3-7, 2010, pp. 4276-4281.

[47] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[48] X. Hou, Y. Yang, F. Li, and Z. Jing, "Kalman filter based on error state variables in SINS + GPS navigation application," *IEEE International Conference on Information Science and Technology*, Nanjing, China, Mar. 26-28, 2011, pp. 770-773.

[49] F. Ayazi, and K. Najafi, "A HARPSS polysilicon vibrating ring gyroscope," *IEEE Journal of Microelectromechanical Systems*, vol. 10, no. 2, pp. 169-179, 2001.

[50] A. D. Challoner, H. Ge Howard, and J. Y. Liu, "Boeing disc resonator gyroscope," *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, May 5-8 2014, Monterey, CA, USA, pp. 504-514.

[51] S. Nitzan, C. H. Ahn, T-H. Su, M. Li, E. J. Ng, S. Wang, Z. M. Yang, G. O'brien, B. E. Boser, T. W. Kenny, and D. A. Horsley, "Epitaxially-encapsulated polysilicon disk resonator gyroscope," *IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 20-24, 2013, Taipei, Taiwan, pp. 625-628.

[52] A. Efimovskaya, D. Wang, Y.-W. Lin, and A. M. Shkel, "Electrostatic compensation of structural imperfections in dynamically amplified dual-mass gyroscope," *Sensors and Actuators A: Physical*, vol. 275, pp. 99-108, 2018.

[53] B. J. Gallacher, J. Hedley, J. S. Burdess, A. J. Harris, A. Rickard, and D. O. King, "Electrostatic correction of structural imperfections present in a microring gyroscope," *IEEE Journal of Microelectromechanical Systems*, vol. 14, no. 2, pp. 221-234, 2005.

[54] F. Ayazi, and K. Najafi, "High aspect-ratio combined poly and single-crystal silicon (HARPSS) MEMS technology," *IEEE Journal of Microelectromechanical Systems*, vol. 9, no. 3, pp. 288-294, 2000.

[55] P. Shao, C. L. Mayberry, X. Gao, V. Tavassoli, and F. Ayazi, "A polysilicon microhemispherical resonating gyroscope," *IEEE Journal of Microelectromechanical Systems*, vol. 23, no. 4, 762-764, 2014.

[56] P. Taheri-Tehrani, T. Su, A. Heidari, G. Jaramillo, C. Yang, S. Akhbari, H. Najar, S. Nitzan, D. Saito, L. Lin, and D. A. Horsley, "Micro-scale diamond hemispherical resonator gyroscope," *Solid-State Sensors, Actuators, and Microsystems Workshop*, June 8-12, 2014, Hilton Head Island, SC, USA, pp. 289-292.

[57] J. Y. Cho, J.-K. Woo, J. Yan, R. L. Peterson, and K. Najafi, "Fused-Silica Micro Birdbath Resonator Gyroscope (μ-BRG)," *IEEE Journal of Microelectromechanical Systems*, vol. 23, no. 1, pp. 66-77, 2013.

[58] J. Cho, T. Nagourney, A. Darvishian, and K. Najafi, "Ultra conformal high aspect-ratio small-gap capacitive electrode formation technology for 3D micro shell resonators," *IEEE 30th International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 22-26, 2017, Las Vegas, NV, USA, pp. 1169-1172.

[59] A. K. Rourke, S. McWilliam, and C. H. J. Fox, "Multi-mode trimming of imperfect rings," *Journal of Sound and Vibration*, vol. 248, no. 4, pp. 695-724, 2001.

[60] A. K. Rourke, S. McWilliam, and C. H. J. Fox, "Multi-mode trimming of imperfect thin rings using masses at pre-selected locations," *Journal of sound and vibration*, vol. 256, no. 2, pp. 319-345, 2002.

[61] B. J. Gallacher, J. Hedley, J. S. Burdess, A. J. Harris, and M. E. McNie, "Multimodal tuning of a vibrating ring using laser ablation," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 217, no. 5, pp. 557-576, 2003.

[62] D. M. Schwartz, D. Kim, P. Stupar, J. DeNatale, and R. T. M'Closkey. "Modal parameter tuning of an axisymmetric resonator via mass perturbation," *IEEE Journal of Microelectromechanical Systems*, vol. 24, no. 3, pp. 545-555, 2015.

[63] J. J. Bernstein, M. G. Bancu, J. M. Bauer, E. H. Cook, P. Kumar, E. Newton, T. Nyinjee, G. E. Perlin, J. A. Ricker, W. A. Teynor, and M. S. Weinberg, "High Q diamond hemispherical resonators: fabrication and energy loss mechanisms," *Journal of Micromechanics and Microengineering*, vol. 25, no. 8, pp. 085006, 2015.

[64] Y. Wang, Y. Pan, T. Qu, Y. Jia, K. Yang, and H. Luo, "Decreasing Frequency Splits of Hemispherical Resonators by Chemical Etching," *Sensors*, vol. 18, no. 11 pp. 3772, 2018.

[65] A. Viswanath, T. Li, and Y. Gianchandani, "High resolution micro ultrasonic machining for trimming 3D microstructures," *Journal of Micromechanics and Microengineering*, vol. 24, no. 6, pp. 065017, 2014.

[66] M. Weinberg, R. Candler, S. Chandorkar, J. Varsanik, T. Kenny, and A. Duwel, "Energy Loss in MEMS Resonators and the Impact on Inertial and RF Devices," *Solid-State Sensors, Actuators and Microsystems Conference (TRANSDUCERS)*, June 21-25, 2009, Denver, CO, USA, pp. 688-695.

[67] S. Ghaffari, E. J. Ng, C. H. Ahn, Y. Yang, S. Wang, V. A. Hong, and T. W. Kenny, "Accurate modeling of quality factor behavior of complex silicon MEMS resonators," *Journal of Microelectromechanical Systems*, vol. 24, no. 2, pp. 276-288, 2014.

[68] W. S. Griffin, H. H. Richardson, and S. Yamanami, "A study of fluid squeeze film damping," *Journal of Basic Engineering*, vol. 88, no. 2, pp. 451-456, 1966.

[69] J. J. Blech, "On isothermal squeeze films," *Journal of Lubrication Technology*, vol. 105, no. 4, 615-620, 1983.

[70] Y. H. Cho, A. P. Pisano, and R. T. Howe, "Viscous damping model for laterally oscillating microstructures," *IEEE Journal of Microelectromechanical Systems*, vol. 3, no. 2, pp. 81-87, 1994.

[71] M. Turowski, Z. Chen, and A. Przekwas, "High-fidelity and behavioral simulation of air damping in MEMS," *Proc. Modeling and Simulation of Microsystems*, Apr. 19–21, 1999, San Juan, Puerto Rico, vol. 99, pp. 241.

[72] P. Fedeli, A. Frangi, G. Laghi, G. Langfelder, and G. Gattere, "Near vacuum gas damping in MEMS: Simplified modeling", *IEEE Journal of Microelectromechanical Systems*, vol. 26, no. 3, pp. 632-642, 2017.

[73] J. Yang, T. Ono, and M. Esashi, "Energy dissipation in submicrometer thick single-crystal silicon cantilevers," *IEEE Journal of Microelectromechanical Systems*, vol. 11, no. 6, pp. 775-783, 2002.

[74] D. Senkal, M. J. Ahamed, M. H. Asadian, S. Askari, and A, M. Shkel, "Demonstration of 1 Million Q-Factor on Microglassblown Wineglass Resonators With Out-of-Plane Electrostatic Transduction," *IEEE/ASME Journal of Microelectromechanical Systems*, vol. 24, no. 1, pp. 29-37, 2015.

[75] C. Zener, "Internal friction in solids I. Theory of internal friction in reeds," *Physical Review*, vol. 52, no. 3, p.230, 1937.

[76] C. Zener, "Internal friction in solids II. General theory of thermoelastic internal friction," *Physical Review*, vol. 53, no. 1, p.90, 1938.

[77] X. Guo, Y-B. Yi, and S. Pourkamali, "A finite element analysis of thermoelastic damping in vented MEMS beam resonators," *International Journal of Mechanical Sciences*, vol. 74, pp. 73-82, 2013.

[78] A. Darvishian, T. Nagourney, J. Y. Cho, B. Shiari and K. Najafi, "Thermoelastic Dissipation in Micromachined Birdbath Shell Resonators," *IEEE Journal of Microelectromechanical Systems*, vol. 26, no. 4, pp. 758-772, 2017.

[79] T. Nagourney, J. Y. Cho, A. Darvishian, B. Shiari, and K. Najafi, "Effect of metal annealing on the Q-factor of metal-coated fused silica micro shell resonators," *IEEE International Symposium on Inertial Sensors and Systems (ISISS)*, Mar. 23-26, 2015, Hapuna Beach, HI, USA, pp. 1-5.

[80] J. J. Lake, A. E. Duwel, and R. N. Candler, "Particle swarm optimization for design of slotted MEMS resonators with low thermoelastic dissipation," *IEEE Journal of Microelectromechanical Systems*, vol. 23, no. 2, pp. 364-371, 2014.

[81] Q. Li, D. Xiao, X. Zhou, Z. Hou, Y. Xu, and X. Wu, "Quality Factor Improvement in the Disk Resonator Gyroscope by Optimizing the Spoke Length Distribution," *IEEE Journal of Microelectromechanical Systems*, vol. 27, no. 3, pp. 414-423, 2018.

[82] Y. H. Park and K. C. Park, "High-fidelity modeling of MEMS resonators. Part I. Anchor loss mechanisms through substrate", *IEEE Journal of Microelectromechanical Systems*, vol. 13, no. 2, pp. 238-247, 2004.

[83] Z. Hao, A. Erbil, and F. Ayazi, "An analytical model for support loss in micromachined beam resonators with in-plane flexural vibrations," *Sensors and Actuators A: Physical*, vol. 109, no. 1-2, pp. 156-164, 2003.

[84] J. Segovia-Fernandez, and G. Piazza, "Analytical and numerical methods to model anchor losses in 65-MHz AlN contour mode resonators," *IEEE Journal of Microelectromechanical Systems*, vol. 25, no. 3, pp. 459-468, 2016.

[85] C. Schaal, R. M'Closkey, and A. Mal, "Semi-analytical modeling of anchor loss in plate-mounted resonators," *Ultrasonics*, vol. 82, pp. 304-312, 2018.

[86] D. S. Bindel, and S. Govindjee, "Elastic PMLs for resonator anchor loss simulation," *International Journal for Numerical Methods in Engineering*, vol. 64, no. 6, pp. 789-818, 2005.

[87] D. S. Binder, E. Quevy, T. Koyama, S. Govindjee, J. W. Demmel, and R. T. Howe, "Anchor loss simulation in resonators," *IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 30 - Feb. 3, 2005, Miami Beach, FL, USA, pp. 133-136.

[88] D. D. Gerrard, E. J. Ng, C. H. Ahn, V. A. Hong, Y. Yang, and T. W. Kenny, "Modeling the effect of anchor geometry on the quality factor of bulk mode resonators," *IEEE International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*, June 21-25, 2015, Anchorage, AK, USA, pp. 1997-2000.

[89] A. Darvishian, B. Shiari, J. Y. Cho, T. Nagourney, and K. Najafi, "Anchor loss in hemispherical shell resonators," *IEEE Journal of Microelectromechanical Systems*, vol. 26, no. 1, pp. 51-66, 2017.

[90] A. A. Trusov, I. P. Prikhodko, S. A. Zotov, and A. M. Shkel, "Low-Dissipation Silicon Tuning Fork Gyroscopes for Rate and Whole Angle Measurements," *IEEE Sensors Journal*, vol. 11, no. 11, pp. 2763-2770, 2011.

[91] M. F. Zaman, A. Sharma, and F. Ayazi, "High performance matched-mode tuning fork gyroscope," *IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 22-26, 2006, Istanbul, Turkey, pp. 66-69.

[92] B. R. Simon, A. A. Trusov, and A. M. Shkel, "Anti-phase mode isolation in tuning-fork MEMS using a lever coupling design," *IEEE SENSORS Conference*, Oct. 28-31, 2012, Taipei, Taiwan, pp. 1-4.

[93] B. Johnson, K. Christ, D. Endean, B. Mohr, R. Supino, H. French, and E. Cabuz, "Tuning fork MEMS gyroscope for precision northfinding," *IEEE DGON Inertial Sensors and Systems Symposium (ISS)*, Spe. 22-23, 2015, Karlsruhe, Germany, pp. 1-10.

[94] D. Senkal, M. J. Ahamed, S. Askari, and A. M. Shkel, "1 Million Q-Factor Demonstrated on Micro-Glassblown Fused Silica Wineglass Resonators with Out-of-Plane Electrostatic Transduction," *Solid-State Sensors, Actuators, and Microsystems Workshop*, June 8-12, 2014, Hilton Head Island, South Carolina, USA.

[95] B. Hamelin, J. Yang, A. Daruwalla, H. Wen, and F.Ayazi, "Monocrystalline Silicon Carbide Disk Resonators on Phononic Crystals with Ultra-Low Dissipation Bulk Acoustic Wave Modes," *Scientific Reports*, vol. 9, no. 1, pp. 1-8, 2019.

[96] J. Yang, T. Ono, and M. Esashi, "Surface effects and high quality factors in ultrathin single-crystal silicon cantilevers," *Applied Physics Letters*, vol. 77, no. 23, pp. 3860-3862, 2000.

[97] K. Y. Yasumura, T. D. Stowe, E. M. Chow, T. Pfafman, T. W. Kenny, B. C. Stipe, and D. Rugar, "Quality factors in micron-and submicron-thick cantilevers," *IEEE Journal of Microelectromechanical Systems*, vol. 9, no. 1, pp. 117-125, 2000.

[98] S. D. Penn, G. M. Harry, A. M. Gretarsson, S. E. Kittelberger, P. R. Saulson, J. J. Schiller, J. R. Smith, and S. O. Swords, "High quality factor measured in fused silica," *Review of Scientific Instruments*, vol. 72, no. 9, pp. 3670-3673, 2001.

[99] Y. Pan, D. Wang, Y. Wang, J. Liu, S. Wu, T. Qu, K. Yang, and H. Luo, "Monolithic Cylindrical Fused Silica Resonators with High Q Factors," *Sensors*, vol. 16, no. 8, pp. 1185, 2016.

[100] M. J. Ahamed, D. Senkal, and A. M. Shkel, "Improvement of side-wall roughness in deep glass etched MEMS vibratory sensors," *IEEE International Symposium on Inertial Sensors and Systems (ISISS)*, Feb. 25-26, 2014, Laguna Beach, CA, USA, pp. 1-2.

[101] P. E. Miller, T. I. Suratwala, L. L. Wong, M. D. Feit, J. A. Menapace, P. J. Davis, and R. A. Steele, "The distribution of subsurface damage in fused silica," *Boulder Damage Symposium XXXVII: Annual Symposium on Optical Materials for High Power Lasers*, Sep. 19-21, 2005, Boulder, CO, USA, pp. 599101.

[102] J. Lambropoulos, "From abrasive size to subsurface damage in grinding," *Optical Fabrication and Testing*, p. OMA6, 2000.

[103] W. J. Startin, M. A. Beilby, and P. R. Saulson, "Mechanical quality factors of fused silica resonators," *Review of Scientific Instruments*, vol. 69, pp. 3681-3689, 1998.

[104] K. Numata, S. Otsuka, M. Ando, and K. Tsubono, "Intrinsic losses in various kinds of fused silica," *Classical and Quantum Gravity*, vol. 19, no. 7, pp. 1697-1702, 2002.

[105] M. Akgul, R. Schneider, Z. Ren, G. Chandler, V. Yeh, and C. T-C. Nguyen, "Hot filament CVD conductive microcrystalline diamond for high Q, high acoustic velocity micromechanical resonators," *Joint Conference of the IEEE International Frequency Control and the European Frequency and Time Forum (FCS) Proceedings*, May 2-5, 2011, San Francisco, CA, USA, USA, pp. 1-6.

[106] M. J. Ahamed, D. Senkal, and A. M. Shkel, "Effect of annealing on mechanical quality factor of fused quartz hemispherical resonator," *International Symposium on Inertial Sensors and Systems (ISISS)*, Feb. 25-26, 2014, Laguna Beach, CA, USA.

[107] S. A. Chandorkar, M. Agarwal, R. Melamud, R. N. Candler, K. E. Goodson, and T. W. Kenny, "Limits of quality factor in bulk-mode micromechanical resonators," *IEEE International Conference on Micro Electro Mechanical Systems*, Jan 13-17, 2008, Wuhan, China, pp. 74-77.

[108] R. Tabrizian, M. Rais-Zadeh, and F. Ayazi, "Effect of phonon interactions on limiting the fQ product of micromechanical resonators," *IEEE International Solid-State Sensors,*

Actuators and Microsystems Conference (TRANSDUCERS), June 21-26, 2009, Denver, CO, USA, pp. 2131-2134.

[109] F. Ayazi, L. Sorenson, and R. Tabrizian, "Energy dissipation in micromechanical resonators," *Micro-and Nanotechnology Sensors, Systems, and Applications III*, vol. 8031, p. 803119, 2011.

[110] D. R. Myers, K. B. Cheng, B. Jamshidi, R. G. Azevedo, D. G. Senesky, L. Chen, M. Mehregany, M. B. Wijesundara, and A. P. Pisano, "Silicon carbide resonant tuning fork for microsensing applications in high-temperature and high G-shock environments," *Journal of Micro/Nanolithography, MEMS, and MOEMS*, vol. 8, no. 2, pp. 021116, 2009.

[111] L. Sher, "Personal inertial navigation system (PINS)," DARPA, 1996.

[112] L. J. Hutchings, "System and method for measuring movement of objects," US Patent No. 5,724,265, 1998.

[113] J. Elwell, "Inertial navigation for the urban warrior," *AEROSENSE '99*, Apr. 5-9, 1999, Orlando, FL, USA, pp. 196-204.

[114] I. Skog, P. Händel, J. O. Nilsson, and J. Rantakokko, "Zero-velocity detection — An algorithm evaluation," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 11, pp. 2657-2666, 2010.

[115] S. Godha, and G. Lachapelle, "Foot mounted inertial system for pedestrian navigation," *Measurement Science and Technology*, vol. 19, no. 7, pp. 075202, 2008.

[116] B. Krach, and P. Robertson, "Integration of foot-mounted inertial sensors into a Bayesian location estimation framework," *IEEE Workshop on Positioning, Navigation and Communication*, Mar. 27, 2008, Hannover, Germany, pp. 55-61.

[117] R. Feliz, E. Zalama, and J. G. Garcia-Bermejo, "Pedestrian tracking using inertial sensors," *Journal of Physical Agents*, vol. 3, no. 1, pp. 35–43, 2009.

[118] I. Skog, J. O. Nilsson, and P. Händel, "Evaluation of zero-velocity detectors for foot-mounted inertial navigation systems," *IEEE International Conference on In Indoor Positioning and Indoor Navigation (IPIN)*, Sept 15-17, 2010, Zurich, Switzerland.

[119] X. Meng, Z.-Q. Zhang, J.-K. Wu, W.-C. Wong, and H. Yu, "Self-contained pedestrian tracking during normal walking using an inertial/magnetic sensor module," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 3, pp. 892-899, 2013.

[120] J. Wahlstrom, I. Skog, F. Gustafsson, A. Markham, and N. Trigoni, "Zero-Velocity Detection — A Bayesian Approach to Adaptive Thresholding," *IEEE Sensors Letters*, vol. 3, no. 6, pp. 1-4, 2019.

[121] X. Tian, J. Chen, Y. Han, J. Shang, and N. Li, "A novel zero velocity interval detection algorithm for self-contained pedestrian navigation system with inertial sensors," *Sensors*, vol.16, no. 10, pp. 1578, 2016.

[122] M. Ma, Q. Song, Y. Li, and Z. Zhou, "A zero velocity intervals detection algorithm based on sensor fusion for indoor pedestrian navigation," *IEEE Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Dec. 15-17, 2017, Chengdu, China, pp. 418-423.

[123] A. Ramanandan, A. Chen, and J. A. Farrell, "Inertial navigation aiding by stationary updates," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no.1, pp. 235-248, 2012.

[124] S. Rajagopal, "Personal dead reckoning system with shoe mounted inertial sensors," Master dissertation, KTH Royal Institute of Technology, 2008.

[125] A. Ahmadi, F. Destelle, L. Unzueta, D. S. Monaghan, M. T. Linaza, K. Moran, and N. E. O'Connor, "3D human gait reconstruction and monitoring using body-worn inertial sensors and kinematic modeling," *IEEE Sensors Journal*, vol. 16, no. 24, pp. 8823-8831, 2016.

[126] D. B. Ahmed, and K. Metzger, "Wearable-based pedestrian inertial navigation with constraints based on biomechanical models," *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, Apr. 23-26, 2018, Monterey, CA, USA.

[127] I. Skog, J. O. Nilsson, D. Zachariah, and P. Händel, "Fusing the information from two navigation systems using an upper bound on their maximum spatial separation," *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Nov. 13-15, 2012, Sydney, Australia.

[128] Z. Liu, and C.-H. Won, "Knee and waist attached gyroscopes for personal navigation: Comparison of knee, waist and foot attached inertial sensors," *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, May 4-6, 2010, Indian Wells, CA, USA, pp. 375-381.

[129] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," *International Conference on Web-Age Information Management*, June 16-18, 2014, Macau, China, pp. 298-310.

[130] P. Kasnesis, C. Z. Patrikakis, and I. S. Venieris, "PerceptionNet: a deep convolutional neural network for late sensor fusion," *SAI Intelligent Systems Conference*, Sep. 6-7, 2018, London, UK, pp. 101-119.

[131] S. Askari, C.-S. Jao, Y. Wang, and A. M. Shkel, "Learning-Based Calibration Decision System for Bio-Inertial Motion Application," *IEEE SENSORS Conference*, Oct. 27-30, 2019, Montreal, Canada, pp. 1-4.

[132] J. Hannink, T. Kautz, C. F. Pasluosta, J. Barth, S. Schülein, K.-G. Gaßmann, J. Klucken, and Bj. M. Eskofier, "Mobile stride length estimation with deep convolutional neural networks," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 2, pp. 354-362, 2017.

[133] B. Beaufils F. Chazal, M. Grelet, and B. Michel, "Robust pedestrian trajectory reconstruction from inertial sensor," *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sep. 30 - Oct. 3, 2019, Pisa, Italy, pp. 1-8.

[134] B. Wagstaff, V. Peretroukhin, and J. Kelly, "Improving foot-mounted inertial navigation through real-time motion classification," *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sep. 18-21, 2017, Sapporo, Japan.

[135] D. Senkal, M. J. Ahamed, A. A. Trusov, A. M. Shkel, "High Temperature Microglassblowing Process Demonstrated on Fused Quartz and ULE TSG," *Sensors and Actuators A: Physical*,vol. 201, pp. 525-531, 2013.

[136] M. H. Asadian, Y. Wang, S. Askari, and A. M. Shkel, "Controlled Capacitive Gaps for Electrostatic Actuation and Tuning of 3D Fused Quartz Micro Wineglass Resonator Gyroscope," *IEEE International Symposium on Inertial Sensors and Systems (Inertial)*, Mar. 27–30, 2017, Kauai, HI, USA.

[137] M. H. Asadian, Y. Wang, and A. M. Shkel, "Development of 3D Fused Quartz Hemi-Toroidal Shells for High-Q Resonators and Gyroscopes," *IEEE Journal of Microelectromechanical Systems*, vol. 28, no. 6, pp. 954-964, 2019

[138] J.-O. Nilsson, A. K. Gupta, and P. Händel, "Foot-mounted inertial navigation made easy," *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Oct. 27-30, 2014, Busan, South Korea, pp. 24-29.

[139] Y. Wang, M. H. Asadian, and A. M. Shkel, "Predictive Analytical Model of Fundamental Frequency and Imperfections in Glassblown Fused Quartz Hemi-Toroidal 3D Micro

Shells," *IEEE Sensors Conference*, October 30-November 2, 2016, Orlando, Florida, USA.

[140] J. W. E. B. Rayleigh, *The theory of Sound*, Vol. 2, Macmillan, 1896.

[141] N. Bellomo, B. Lods, R. Revelli, and L. Ridolfi, *Generalized collocation methods: solutions to nonlinear problems*, Springer Science & Business Media, 2007.

[142] S.-Y. Choi, and J.-H. Kim, "Natural frequency split estimation for inextensional vibration of imperfect hemispherical shell," *Journal of Sound and Vibration*, vol. 330, no. 9, pp. 2094-2106, 2011.

[143] W. Soedel, *Vibrations of shells and plates*." 3rd. ed., CRC Press, 2004.

[144] A. Trusov, D. Senkal, and A. M. Shkel, "Microfabrication of high quality three dimensional structures using wafer-level glassblowing of fused quartz and ultra low expansion glasses," US Patent No. 9,139,417

[145] Y. Wang, M. H. Asadian, and A. M. Shkel, "Frequency Split Reduction by Directional Lapping of Fused Quartz Micro Wineglass Resonators", *IEEE International Symposium on Inertial Sensors and Systems (Inertial)*, March 27-30, 2017, Kauai, HI, USA.

[146] R. H. Doremus, "Viscosity of silica," *Journal of Applied Physics*, vol. 92, no. 12, pp. 7619-7629, 2002.

[147] Y. Wang, M. H. Asadian, and A. M. Shkel, "Modeling the Effect of Imperfections in Glassblown Micro-Wineglass Fused Quartz Resonators," *ASME Journal of Vibration and Acoustics*, vol. 139, no. 4, pp. 040909, 2017.

[148] Y. Wang, A. M. Shkel, "Study on Surface Roughness Improvement of Fused Quartz After Thermal and Chemical Post-Processing", *IEEE International Symposium on Inertial Sensors and Systems*, February 22-25, 2016, Laguna Beach, CA, USA.

[149] H. Kaigawa, K. Yamamoto, and Y. Shigematsu, "Etching of thermally grown SiO2 by NH4OH in mixture of NH4OH and H2O2 cleaning solution," *Japanese Journal of Applied Physics*, vol. 33, no. 7, pp. 4080, 1994.

[150] M. Pelliccione, and T.-M. Lu, *Evolution of thin film morphology*, Springer Series in Materials Science, vol. 108, Springer-Verlag Berlin, 2008.

[151] A. M. Shkel and D. Senkal, "Environmentally robust micro-wineglass gyroscope"," US Patent No. 9,429,428 B2.

[152] M. H. Asadian and A. M. Shkel, "Fused Quartz Dual Shell Resonators," *IEEE International Symposium on Inertial Sensors and Systems (Inertial)*, Apr. 1-5, 2019, Naples, FL, USA.

[153] M. H. Asadian, Y. Wang, D. Wang, A. M. Shkel, "High-Q 3D Dual-Shell Resonators for High Shock Environments," *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, Apr. 20-23, 2020, Portland, OR, USA.

[154] J. S. Pulskamp, S. S. Bedair, R. G. Polcawich, G. L. Smith, J. Martin, B. Power, and S. A. Bhave, "Electrode-shaping for the excitation and detection of permitted arbitrary modes in arbitrary geometries in piezoelectric resonators," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 59, no. 5, pp. 1043-1060, 2012.

[155] J. M. Puder, J. S. Pulskamp, R. Q. Rudy, C. Cassella, M. Rinaldi, G. Chen, S. A. Bhave, and R. G. Polcawich, "Rapid harmonic analysis of piezoelectric mems resonators," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 65, no. 6, pp. 979-990, 2018.

[156] D. M. Potrepka, G. R. Fox, L. M. Sanchez, and R. G. Polcawich, "Pt/TiO2 growth templates for enhanced PZT films and MEMS devices," *MRS Online Proceedings Library Archive*, vol. 1299, 2011.

[157] D. Senkal, S. Askari, M. J. Ahamed, E. Ng, V. Hong, Y. Yang, C. H. Ahn, T. W. Kenny, A. M. Shkel, "100k Q-Factor Toroidal Ring Gyroscope Implemented in Wafer-level Epitaxial Silicon Encapsulation Process," *IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 26-30, 2014, San Francisco, CA, USA.

[158] D. Senkal, E.J. Ng, V. Hong, Y.Yang, C.H. Ahn, Thomas W. Kenny, Andrei M. Shkel, "Parametric Drive of a Toroidal MEMS Rate Integrating Gyroscope Demonstrating $<$ 20 ppm Scale Factor Stability," *IEEE International Conference on Micro Electro Mechanical Systems (MEMS)*, Jan. 18 - 22, 2015, Estoril, Portugal.

[159] Y. Yang, E. J. Ng, Y. Chen, I. B. Flader, and T. W. Kenny, "A Unified Epi-seal Process for Fabrication of High-stability Microelectromechanical Devices," *IEEE Journal of Microelectromechanical Systems*, vol. 25, no. 3, pp. 489-497, 2016.

[160] Y. Yang, E. J. Ng, Y. Chen, I. B. Flader, C. H. Ahn, V. A. Hong, and T. W. Kenny, "A unified epi-seal process for resonators and inertial sensors," *IEEE International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*, June 21-25, 2015, Anchorage, AK, USA, pp. 1326-1329.

[161] A. Efimovskaya, D. Wang, Y.-W. Lin, and Andrei M. Shkel, "On Ordering of Fundamental Wineglass Modes in Toroidal Ring Gyroscope," *IEEE Sensors Conference*, Oct. 30 - Nov. 2, 2016, Orlando, FL, USA.

[162] https://www.lakeshore.com/docs/default-source/product-downloads/fwp6.pdf?sfvrsn=9b1ad3b3_1

[163] T. Middelmann, A. Walkov, G. Bartl, and R. Schodel, "Thermal expansion coefficient of single-crystal silicon from 7 K to 293 K," *Physical Review B*, vol. 92, no. 17, pp. 174113, 2015.

[164] W. Tao, T. Liu, R. Zheng, and H. Feng, "Gait analysis using wearable sensors," *Sensors*, vol. 12, no. 2, pp. 2255-2283, 2012.

[165] M. P. Murray, A. B. Drought, and R. C. Kory, "Walking gait of normal man," *Journal of Bone & Joint Surgery*, vol. 46, pp. 335-360, 1964.

[166] J. Perry, and J. R. Davids, "Gait analysis: normal and pathological function," *Journal of Pediatric Orthopaedics*, vol. 12, no. 6, pp. 815, 1992.

[167] M. W. Whittle, *Gait analysis: an introduction*, 3rd ed., Oxford: Butterworth-Heinemann, 2002.

[168] P. G. Savage, *Strapdown analytics*, 2nd ed., Maple Plain, MN: Strapdown Associates, 2007.

[169] J. K. Huddle, "Theory and performance for position and gravity survey with an inertial system," *Journal of Guidance, Control, and Dynamics*, vol. 1, no. 3, pp. 183-188, 1978.

[170] Y. Wang, S. Askari, and A. M. Shkel, "Study on mounting position of IMU for better accuracy of ZUPT-adied pedestrian inertial navigation," *IEEE International Symposium on Inertial Sensors and Systems (Inertial)*, Apr. 1-5, 2019, Naples, FL, USA.

[171] M. Ren, K. Pan, Y. Liu, H. Guo, X. Zhang, and P. Wang, "A novel pedestrian navigation algorithm for a foot-mounted inertial-sensor-based system," *Sensors*, vol. 16, no. 1, pp.139, 2016.

[172] A. R. Jimenez, F. Seco, J. C. Prieto, and J. Guevara, "Indoor pedestrian navigation using an INS/EKF framework for yaw drift reduction and a foot-mounted IMU," *IEEE Workshop on Positioning Navigation and Communication (WPNC)*, Mar. 11-12, 2010, Dresden, Germany.

[173] Y. Wang, D. Vatanparvar, A. Chernyshoff, and A. M. Shkel, "Analytical Closed-Form Estimation of Position Error on ZUPT-Augmented Pedestrian Inertial Navigation," *IEEE Sensors Letters*, vol. 2, no. 4, pp. 1-4, 2018.

[174] J. O. Nilsson, I. Skog, and P. Händel, "A note on the limitations of ZUPTs and the implications on sensor error modeling," *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Nov. 13-15, 2012, Sydney, Australia.

[175] Y. Wang, A. Chernyshoff, and A. M. Shkel, "Error analysis of ZUPT-aided pedestrian inertial navigation," *IEEE International Conference on Indoor Positioning and Indoor Navigaiton (IPIN)*, Sep. 24-27, 2018, Nantes, France.

[176] Y. Wang, A. Chernyshoff, and A. M. Shkel, "Study on Estimation Errors in ZUPT-Aided Pedestrian Inertial Navigation Due to IMU Noises," *IEEE Transactions on Aerospace and Electronic Systems*, doi: 10.1109/TAES.2019.2946506.

[177] VectorNav VN-200 GPS-Aided Inertial Navigation System Product Brief. https://www.vectornav.com/docs/default-source/documentation/vn-200-documentation/PB-12-0003.pdf?sfvrsn=749ee6b9_13

[178] Y. Wang, and A. M. Shkel, "Adaptive Threshold for Zero-Velocity Detector in ZUPT-Aided Pedestrian Inertial Navigation," *IEEE Sensors Letters*, vol. 3, no. 11, pp. 1-4, 2019.

[179] H. L. Van Trees, *Detection, estimation, and modulation theory, part I: detection, estimation, and linear modulation theory*, 2nd ed. John Wiley & Sons, 2004.

[180] O. Zeitouni, J. Ziv, and N. Merhav, "When is the generalized likelihood ratio test optimal?" *IEEE Transactions on Information Theory*, vol. 38, no. 5, pp. 1597-1602, 1992.

[181] Polhemus PATRIOT two-sensor 6-DOF tracker: https://polhemus.com/_assets/img/PATRIOT_brochure.pdf

[182] J. B. Bancroft, and G. Lachapelle, "Estimating MEMS gyroscope g-sensitivity errors

in foot mounted navigation," *IEEE Ubiquitous Positioning, Indoor Navigation, and Location Based Service (UPINLBS)*, Oct. 3-4, 2012, Helsinki, Finland.

[183] Z. Zhu, and S. Wang, "A Novel Step Length Estimator Based on Foot-Mounted MEMS Sensors," *Sensors*, vol. 18, no. 12, pp. 4447, 2018.

[184] A. B. Chatfield, *Fundamentals of high accuracy inertial navigation*, American Institute of Aeronautics and Astronautics, 1997, eISBN: 978-1-60086-646-3.

[185] Yusheng Wang and Andrei M. Shkel, "A Review on ZUPT-Aided Pedestrian Inertial Navigation," *27th Saint Petersburg International Conference on Integrated Navigation Systems*, May 25-27, 2020, Saint Petersburg, Russia.

[186] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*, John Wiley & Sons, 2001.

[187] J. Kilpatrick, A. Apostol, and V. Markov, "MEMS characterization with conformal imaging vibrometry (CIV)," *IEEE International Symposium on Inertial Sensors and Systems (ISISS)*, Feb. 25-26, 2014, Laguna Beach, CA, USA.

Appendix A

Lab procedures

The developed procedures related to the fabrication of Fused Quartz wineglass 3D resonators are listed in this section.

A.1 Plasma-Aided Glass-to-Glass Wafer Bonding

Overview	
Input	Wafers in separate petri dishes
Consumables	Chemicals for Piranha and RCA-1 bath, acetone, IPA, DI water
Equipment	Wafer cleaning equipment, Teflon wafer holders, PlasmaTherm 790 RIE, AML-AWB Wafer Bonder (Optional)
Output	Bonded wafer pairs

A.1.1 Purpose

This process is used to bond two Fused Quartz wafers to form wafer stacks. This is a critical step in the fabrication of Fused Quartz wineglass resonators. The advantage of plasma-aided bonding is that no intermediate material is needed as the bonding interface, thus, the wafer stack can survive under high temperature.

A.1.2 Process

A.1.2.1 Wafer Cleaning

Wafer cleaning is critical in this process, since the process is super sensitive to dust, or any contamination on the surface. Typical residue may include photoresist, polymer, and grease from previous fabrication steps. Wafers must be thoroughly cleaned before they are bonded. Solvent cleaning (Acetone and IPA), Piranha cleaning, and RCA-1 cleaning should be conducted in sequence to completely remove all possible residue on the surface. After the wafers are cleaned, store them in clean Teflon wafer holders with DI water separately.

A.1.2.2 Surface Activation

1. Thoroughly wipe the bench with Acetone and IPA before using it.

2. Place a piece of wipe on the bench, take one wafer out of poly-container and place it on the wipe. Dry the tweezers and the wafer with Nitrogen flow. Place the wafer on another piece of wipe and dry it again. Transfer the wafer to a clean petri dish (with wipe under the wafer to prevent stiction). Do the same with the second wafer.

3. Place two wafers into the chamber of PlasmaTherm 790 RIE. Load the process "O2ASH-Sh". The time for the process is 3 minutes. Take out the wafers and put them back

to the petri dishes after the process is finished.

4. Take one wafer out of the petri dish (in the bench of course). Rinse it under running DI water for 1 minute. Then dry it as in step 2. Then put the wafer back to the petri dish to avoid any dust falling on its surface. Do the same for the second one.

A.1.2.3 Wafer Bonding

1. Take one wafer out of the petri dish (preferably the thinner one) and put it on a piece of clean wipe. Take out the other wafer and flip it so that the activated surface is at bottom. Carefully place it on top of the other wafer. Make sure the top wafer does not slide on the other wafer. Press the top wafer with hand. Ideally, the bonded area should spread if the surfaces are clean and flat. Fringes can be observed in the unbonded area due to interference of the light. This step can also be finished with AML-AWB Wafer Bonder for a better alignment.

2. Press the stack to squeeze the unbonded area manually to improve the yield of the bonding process. The bonded wafer stack is shown in Fig. A.1.

A.1.2.4 Wafer Annealing

Wafer annealing is to strengthen the bonding between the wafers. Place the stack under room temperature for one day before conducting high-temperature annealing. The temperature for annealing is 400 °C with temperature ramp of 50 °C/h. Anneal the stack for 1 day before dicing.

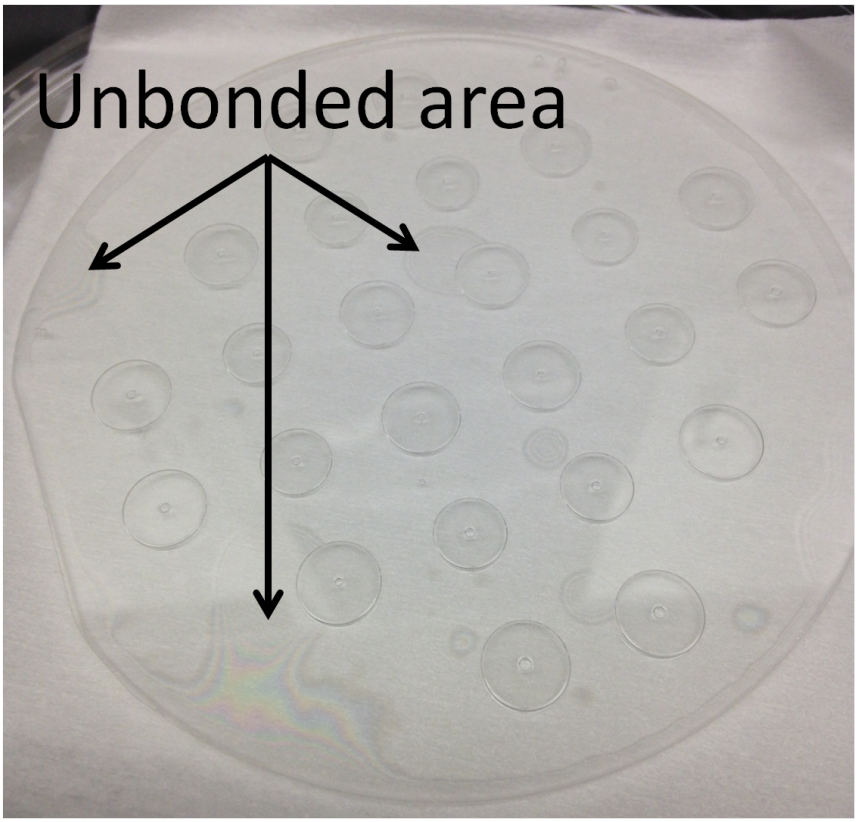

Figure A.1: A picture of the bonded wafer. The unbonded area are indicated by the fringes.

A.1.3 Remarks

1. This process is super sensitive to surface quality. Send the wafer back to the vendor for another fine polishing if needed.
2. It is relatively harder to see if the stem area is bonded or not. Do not press on the cavity too hard to avoid breaking the wafer, since the wafer is thinner in that area.
3. If the yield of bonding is too low (<80%), it is an option to separate the two wafers. However, the separated wafers need to be cleaned and activated again before they can be bonded again.

A.2 High-Temperature Micro-Glassblowing

Overview	
Input	Bonded wafer stacks
Consumables	Acetone, IPA, DI water
Equipment	MTI GSL-1500X high-temperature furnace
Output	Glassblown devices

A.2.1 Purpose

This process is used in the fabrication of Fused Quartz micro-wineglass resonators. It takes advantage of the drop of viscosity of Fused Quartz material and the increase of pressure of air trapped in the pre-etched cavity at high temperature. If the pressure in the cavity is large enough to overcome the viscosity of material, the device layer will deform and form the shape of 3D shell.

A.2.2 Process

1. Plant the plug. Turn on the switch on the right side of the furnace. Turn on the lock on the panel.

2. There are two panels on the front side of the furnace. Use the left one. Press the "A/M" button to edit the temperature profile. It should show "C 01" on top and "0" at bottom. Press the button on the left side. It should show "t 01" on top and "30" at bottom. This means the initial temperature is 0 °C and it takes 30 minutes for the temperature in the furnace to get to next temperature point. Press the button on the left side again. It shows "C 02" on top and the second temperature point at bottom. Press the "A/M" button to select the digit that you want to edit and use up (RUN/HOLD) and down (STOP) button to make change. Each time and temperature set point is shown in Table A.1.

3. It is critical to put the fused silica stack in the very middle of the furnace so that the temperature gradient in the furnace will not influence the shell. Measure the depth of the chamber before it gets hot. Estimate the position of the sample. Make a mark on the trail of the holder accordingly.

4. Press the green "Turn-on" button on the right side of the panel to begin heating. Make sure you are using ceramic tube rather than fused silica one. Place and fix the boat in the tube. Push the whole handler into the chamber. The heating-up phase takes about 3 hours.

5. Rinse the wafer stacks with Acetone, IPA, and DI water to remove all organic residue or dusts on it. Use air-gun to dry the wafer stacks.

6. After the temperature reaches the set point, long press the "RUN/HOLD" button to keep the temperature constant.

7. Pull out the holder slowly to avoid huge thermal shock. Use three piece of fused silica to build a bridge and put the sample on it. This is to mitigate the huge temperature gradient along the direction of the holder DO NOT look into the chamber with naked eyes when it is hot.

8. Push the holder into the chamber until the marker is aligned. The temperature will drop to about 1400 °C and grows back. The whole process should take 3-4 minutes. The timing of glassblowing is related to the size and thickness of the sample and is mainly based on experience. Push the holder into the furnace after taking off the sample.

9. Repeat step 7 and step 8 to conduct glassblowing to all samples.

10. After all samples are glassblown, long press the "RUN/HOLD" button resume the program. Press the leftmost button and it shows "4" screen. Press "STOP" button to change it to "5" so that the program directly goes to step 5, which is cooling-down phase. After the temperature is below 300 °C, press the red "Turn-off" button, turn off the lock, turn off the switch, and plug off the furnace.

Table A.1: Parameters for temperature profile setup

Step	01	02	03	04	05	06	07
C	0	300	300	1500	1500	0	0
T	30	30	120	120	100	300	N/A

A.2.3 Remarks

1. Temperature for glassblowing is extremely high. Always use a thermal glove when placing sample on the boat and taking it off. Always use the eye field to look inside the furnace.

2. The boat can be setup in different ways. Ceramic rods are preferable to ceramic tubes for better resistance to thermal shock and lower thermal conduction. All efforts should be made to mitigate the temperature gradient along the direction of the holder, which is THE dominant factor that affects the structural symmetry of the glassblown devices.

3. The temperature of the furnace should be set up in a way that the heating elements in the chamber will not break due to thermal shock. The temperature profile against time is shown in Fig. A.2. In the heating-up phase, the temperature increases from room temperature to 300 °C in 30 minutes and remain at that temperature for 30 minutes. Then the temperature increases to 1500 °C in 120 minutes and keep the temperature constant.

4. This process can be conducted in batch or in single devices. The only limiting factor is the size of the furnace if set properly.

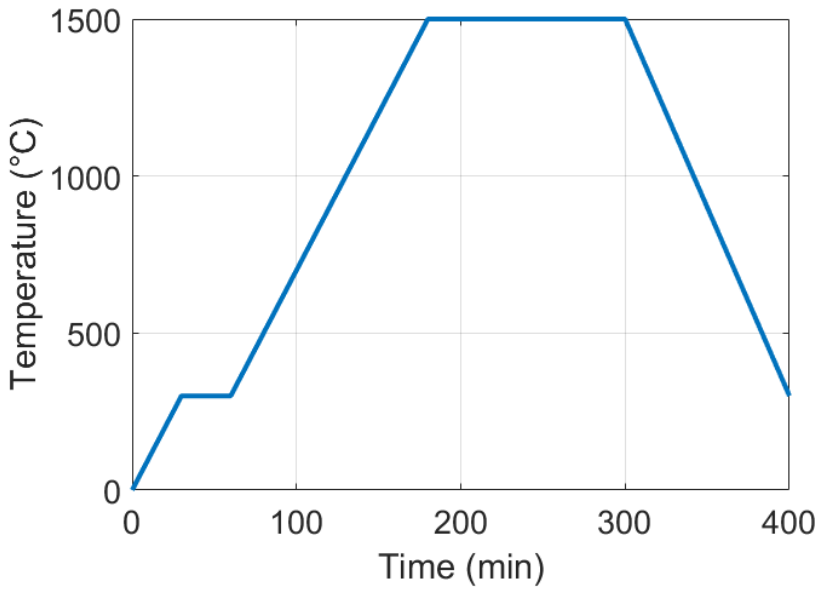

Figure A.2: The temperature profile of the furnace for glassblowing.

A.3 Mechanical Lapping of Glassblown Shells

Overview	
Input	Glassblown shells
Consumables	Acetone, IPA, DI water, SPI Crystalbond 509, Crystalbond Stripper, foam tipped swabs
Equipment	Allied High Tech Lapping machine, hotplate, Leica microscope, BRANSON 1800 ultrasonic cleaner, lapping fixture 3D printed by RapidTech
Output	Lapped glassblown shells

A.3.1 Purpose

This process removes the substrate of the 3D glassblown shell with low roughness on the released surface.

A.3.2 Process

A.3.2.1 Lapping Machine Calibration (Optional)

Calibration of the lapping machine is needed before any extremely precise lapping, or any wafer-level lapping. It is recommended to do alignment at least twice a year to establish perpendicular and parallel reference surfaces to which samples are secured. The procedure of calibration can be found in the operation manual of the lapping machine, pp. 20-27.

A.3.2.2 Shell Mounting

1. Clean the back side of the lapping fixture to make sure it is free of residue and crystalbond.

2. Turn on the hotplate and set the temperature to be 150 °C. Place the lapping fixture on the hotplate. Apply a small amount of crystalbond on top of the lapping fixture. After the crystalbond melts, place the shell to be lapped on top. The substrate should be on the top side to be removed in the following steps. Press the shell so that it fully contacts the fixture surface.

3. Remove the lapping fixture with the sample from the hotplate. Wait for the whole stack to cool down.

A.3.2.3 Lapping Machine Setup

1. Remove the lid and the splash ring from the lapping machine. Make sure the surface of the platen base and the back side of the platen is clean and dry before proceeding. If not, use cleanroom wipes and IPA to remove water, residue, or rust. Install aluminum platen #10-1010M with the grit lapping pad on it. Place the platen on the base gently and press the platen to make sure they are in good contact. The orientation of the platen is not critical for this step.

2. Turn on the power supply at back of the lapping machine. Turn on the water tap and the switch on the bottom right corner of the control panel.

3. Long-press the "OSC" button and release it after hearing two consecutive beeps. The light on the button will become yellow. Make sure it shows "1:1" on the right screen, meaning the oscillation speed is 1, which is lowest. If it is not, press the arrow buttons in the middle to adjust it. Then press the "OSC" button again to stop editing. The

light on the button should turn green, which means the oscillation motion is on. Press it again to turn it off. Do the same for the rotation motion ("Full" button next to it).

4. Press "1","0", and "0", and then press "ENTER". It should show "100" on the left screen, meaning the rotation speed of the platen is 100rpm. Press the green button on the left to begin the rotation of the platen; press the red button to stop it.

5. Use the load adjustment screw to change the load level to full (on the right side of the machine, use a mirror to see it).

6. Align the position-thumbscrew and secure it. Install the lapping fixture on the arm and make sure the edge of the fixture is aligned with the reference edge. Then secure the lapping fixture by rotating the cam-lock lever.

Figure A.3: Details of the lapping machine.

A.3.2.4 Shell Lapping

1. Use the vertical adjustment knob to raise the arm before rotating the spindle riser to lower the arm, in order to make sure the sample does not touch the grit lapping pad. Turn on the water and rotation of the platen. Adjust the height of the arm so that

the sample just touches the grit lapping pad. Set both the digital indicators to be zero and lower the arm. Turn on the rotation and oscillation of the arm.

2. Lap the shell until the cavity is opened. Rotate the spindle riser to raise the lapping fixture. Press the red button to stop lapping. The overlapping in this step is not critical, but should still be avoided. The opening of the cavity should be large enough to place more crystalbond into the cavity.

3. Remove the lapping fixture from the lapping machine and place it back on the hotplate again. Use small pieces of crystalbond to fill the cavity. This step is to reinforce the stem and the rim of the shell, such that they don't break during lapping.

4. Remove the lapping fixture from the hotplate. Wait for the whole stack to cool down.

5. Keep lapping the shell with the grit lapping pad until about 250μm to full releasing.

6. Place the shell under microscope to observe the lapped surface.

7. Remove the aluminum platen #10-1010M and replace with the platen #10-1010. Clean the bottom side of the platen if necessary. There is a letter "A" stamped on the platen base. Make sure it is aligned to the "A" on the back side of the aluminum platen.

8. Press the Coolant button at the bottom left corner so water will flow out of the water coolant. Spread water on the platen then press the button again to stop the water. Put a 30μm diamond lapping film (green) on the platen and make sure the rough surface is on top. Use the squeezer to squeeze out the water between the film and the platen so that the film is attached to the platen by surface tension. Rotate the platen if needed (NOT by hand).

9. Continue lapping for about 150μm. Then observe the lapping surface under microscope. A smoother surface should be observed without any major scratches.

10. Change to 15μm diamond lapping film, and lap the shell until the it is partially released (indicated by the appearing of the crescent shape of crystalbond near the edge of the shell shown in Fig. A.4). The lapping amount should be around 100μm for this step. Overlapping should be avoided in this step, such that the resonant frequency of the release device is closed to the design value. Observe the lapped surface.

11. Change back to aluminum platen #10-1010M, and place the cloth lapping pad on top. Then, apply 3μm diamond slurry on the lapping pad and start lapping. Lapping in this step should be at least half on hour. It is hard to measure the lapped distance using cloth lapping pad, however, an obvious change of the surface quality should be seen under the microscope. No crystalbond or scratch should be observed on the rim of the shell. Foam tipped swabs can be used to remove extra crystalbond on the rim with acetone or IPA.

12. Change to another cloth lapping pad, apply 0.5μm diamond slurry, and start lapping. This is the final lapping step, and ideally, no scratches or imperfections should be observed under microscope after this step. A comparison of the lapped surface after rough lapping and fine lapping is presented in Fig. A.5.

A.3.2.5 Shell Cleaning

1. Place the lapping fixture on the hotplate at 150 °C. Remove the lapped shell from the fixture after the crystalbond softens.

2. Place the shell in acetone overnight to remove the crystalbond. Put a lid on the container to avoid evaporation.

3. Pour crystal bond stripper into the container of the ultrasonic cleaner, and set the temperature to be 60 °C. Take the shell out of acetone and rinse it under IPA and ID

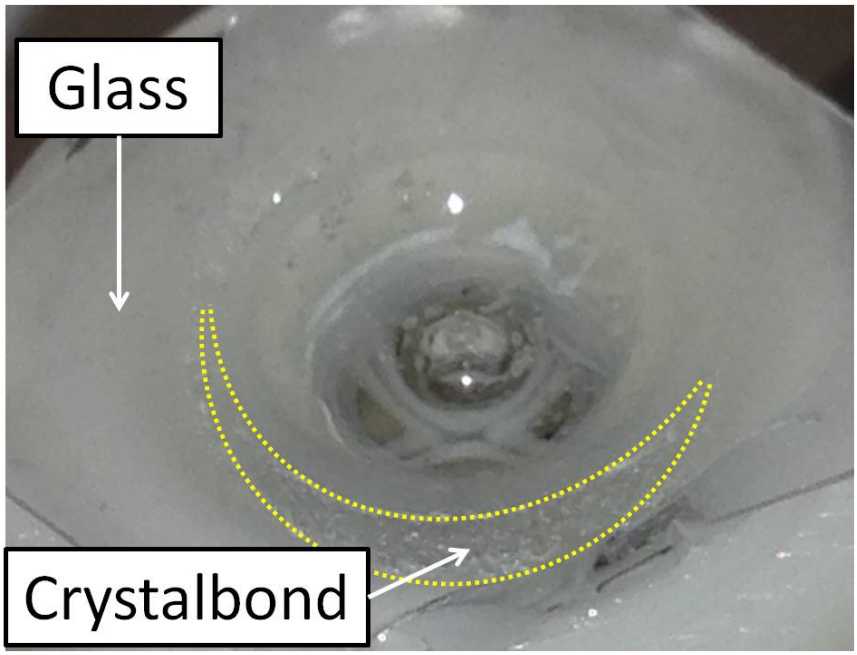

Figure A.4: The crescent shape indicating partial release of the shell.

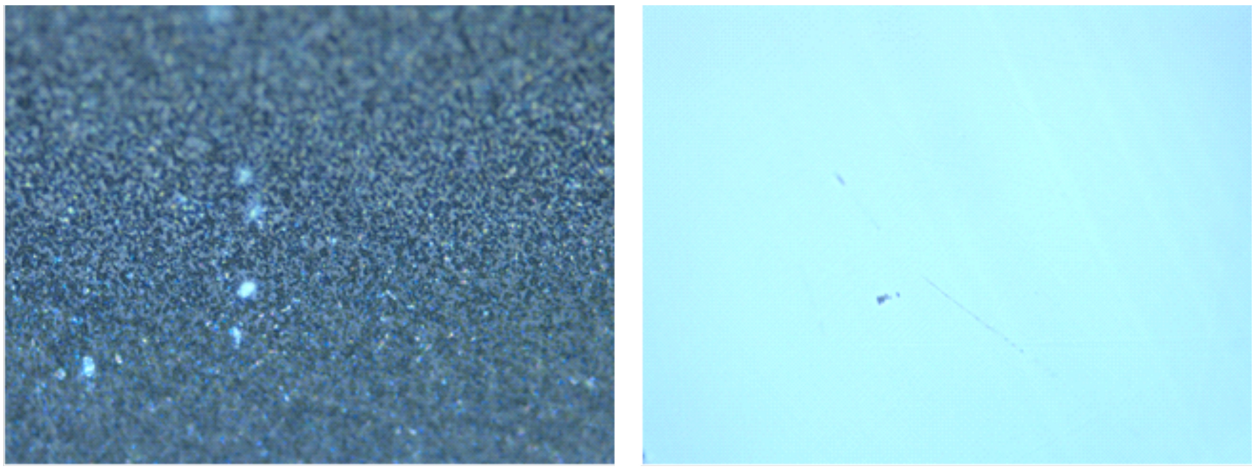

Figure A.5: Comparison of the surface after 15µm lapping and 0.5µm lapping. Magnification of 50× is used under microscope.

water, then, place it in the ultrasonic cleaner. Turn on the cleaner. Clean the shell for 1 hour.

4. Take the shell out of the cleaner, rinse it under acetone, IPA, and DI water, dry it on the hotplate, and observe it under microscope. No crystalbond residue should be observed. Place the shell back to the ultrasonic cleaner for further cleaning if necessary.

5. Clean the shell in piranha bath and RCA-1 bath, dry it in the dehydration furnace, and store it in the dry box.

A.3.3 Remarks

1. This process can be used for the lapping of both single shells, dual shells, and thinning of the wafer. Batch lapping of the shells is achievable by using different lapping fixtures.

2. Not too much crystalbond should be used to attach the shell to the lapping fixture, or it is very likely that the shell will be misaligned with respect to the fixture.

3. Bubbles attached on the surface of the cavity should be avoided when reinforcing the stem and the rim of the shell. Placing the lapping fixture with the shell on the hotplate at high temperature for long time helps to get rid of the bubbles.

4. Overlapping of the shell should be avoided in all lapping steps, especially when using the 15μm diamond lapping film. The lapping surface should be checked at least every 5 minutes when the cavity is about to open.

5. A thorough cleaning of the shells is critical after lapping, since many different materials are involved during lapping, and residue will be left on the surface, which will ultimately reduce the overall quality factor of the shell if not cleaned.

Appendix B

Piezoelectric Simulation

The modeling techniques to design the shape of the electrodes in the piezoelectric resonators and to simulate their frequency response are briefly described in this section.

B.1 Electrode-Shaping in Piezoelectric Resonators

Electrode-shaping is to properly design the resonator electrodes to selectively such that the desired resonant modes can be excited and detected. More details can be found in [154].

Assuming a thin piezoelectric layer on the surface of the resonator, the differential volume elements on the surface can be considered to be in the state of plane stress in the x-y plane (see Fig. B.1). For most commonly used piezoelectric materials, such as PZT, AlN, and ZnO, their piezoelectric coefficients e_{31} and e_{32} are the same. Thus, the piezoelectric differential element is in a state of bi-axial stress, with equal and invariant stress along any direction. Coordinate transformation can be applied to extract the principal stress of the resonator

differential element:

$$\sigma_{1,2} = \frac{\sigma_x + \sigma_y}{2} \pm \sqrt{(\frac{\sigma_x - \sigma_y}{2})^2 + \tau_{xy}^2}, \tag{B.1}$$

where σ_1 and σ_2 are two principal stress along the first and second principal axes.

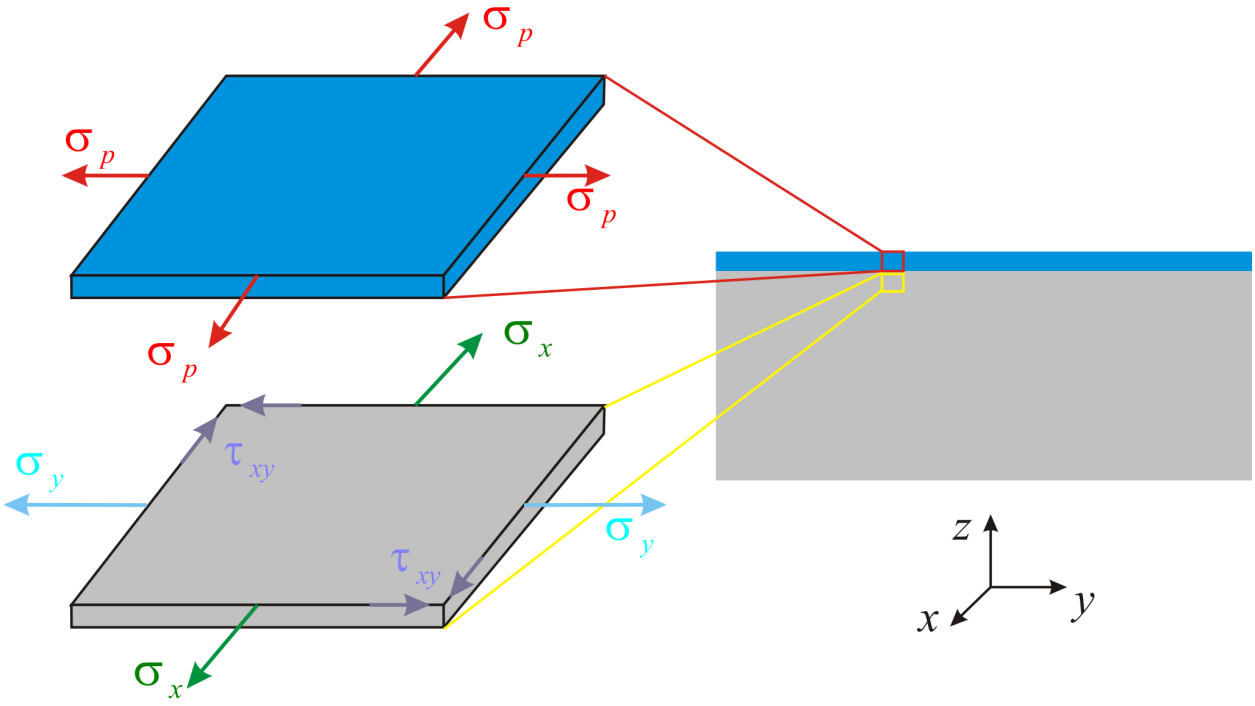

Figure B.1: Stress of differential volume elements on the surface of the piezoelectric resonator.

The resonant mode will be locally excited when the piezoelectrically induced stresses are in-phase with the modal stresses, i.e., when the signs of σ_1, σ_2, and σ_p are the same. Therefore, we can define a parameter β to describe the electrode design:

$$\beta = \begin{cases} sign(\sigma_1)sign(\sigma_p), & \text{if } sign(\sigma_1) = sign(\sigma_2), \\ 0, & \text{if } sign(\sigma_1) \neq sign(\sigma_2). \end{cases} \tag{B.2}$$

In the area where $\beta = 0$, the local force cancellation will occur since σ_1 and σ_2 are of different signs. However, for the are corresponding to $\beta = 1$ and $\beta = -1$, the piezoelectrically induced

stress will excite or inhibit the resonance, respectively. Therefore, one set of the electrode can be deposited in the area where $\beta = 1$ to excite the resonance, while another set of the electrode can be deposited in the area where $\beta = -1$ to form the differential driving architecture.

B.2 Frequency Response of Piezoelectric Resonators

In this section, a technique to simulate the frequency response of the piezoelectric resonators is introduced. One of the purpose of such simulation is to study the transduction efficiency of piezoelectric actuation. More details can be found in [155].

The basis of the frequency response analysis is the Modified Butterworth Van-Dyke (mBVD) model, where piezoelectric resonators can be considered as an equivalent circuit model, and resonant modes are assumed to be independent of each other, and a full frequency response can be obtained by superimposing the response of each mode.

In a quasi-static simulation, there are three types of internal energy stored in a piezoelectric resonator: purely elastic energy U_e, purely electric energy U_d, and the energy due to the piezoelectric effect U_p. In this study, only U_p is considered, which includes the elastic energy generated when an electric field is applied to the piezoelectric material, as well as the capacitive energy due to the deformed piezoelectric material. Since U_p consists of two parts, it can be modeled as a lumped spring with a modal force and a lumped capacitor with an applied voltage, and these two parts have the same magnitude. Then, the total energy can be expressed as

$$\frac{1}{2}F_m\delta + \frac{1}{2}Q\Phi = \Phi\delta \int_V \nabla\phi^{\mathrm{T}} \boldsymbol{e}^t \boldsymbol{S}_n dV, \tag{B.3}$$

where F_m is the modal force, δ is the modal displacement, Q is the charge in the modal

capacitor, Φ is the applied voltage, $\boldsymbol{e}^t$ piezoelectric stress matrix, ϕ is the normalized electric potential field, $\boldsymbol{S}_n$ is the normalized strain field, and V is the volume of the piezoelectric material. Then the modal force can be calculated as

$$F_m = \Phi \int_V \boldsymbol{e}^t \nabla\phi \boldsymbol{S}_n dV. \tag{B.4}$$

The effective stiffness of the mode can be calculated as

$$k_m = \int_V \boldsymbol{S}_n^{\mathrm{T}} \boldsymbol{c}^E \boldsymbol{S}_n dV, \tag{B.5}$$

where $\boldsymbol{c^E}$ is the stiffness matrix. Then, the static modal displacement is

$$\delta = \frac{F_m}{k_m} = \Phi \frac{\int_V \boldsymbol{e}^t \nabla\phi \boldsymbol{S}_n dV}{\int_V \boldsymbol{S}_n^{\mathrm{T}} \boldsymbol{c}^E \boldsymbol{S}_n dV}. \tag{B.6}$$

Then the vibration of the piezoelectric resonator can be expressed as

$$u = \delta \cdot u_n \cdot \frac{e^{j\omega t}}{(1 - \frac{\omega^2}{\omega_n^2}) + \frac{j\omega}{Q_m \omega_n}}, \tag{B.7}$$

where u_n is the normalized mode shape, ω_n is the resonant frequency of the mode, Q_m is the quality factor of the mode, and ω is the driving frequency.

Appendix C

MATLAB Code for ZUPT-Aided Navigation

MATLAB code developed for human gait bio-mechanical model and ZUPT-aided pedestrian inertial navigation are listed with the involved sub-functions. A MATLAB-based GUI to demonstrate the effects of IMU errors on the navigation error in the ZUPT-aided pedestrian inertial navigation is also included.

C.1 Code for Human Gait Bio-Mechanical Model

MATLAB code used to generate the trajectories of both feet during human walking and corresponding IMU readouts is listed below. The results are reported in Section 5.1 and [175]. More details of the strapdown inertial navigation algorithm can be found in Section 1.2.1.2 and [6]. The original MATLAB code file can be found in `\\nitride.eng.uci.edu\NIST\Code_Yusheng\Human_gait_model`.

```
% This code is to generate trajectroy of foot by gait analysis
% Each gait cycle starts with left foot contact
clc;
clear;
close all;
currDir = pwd;
addpath([currDir, '\lib']);
s=settings();
global simdata;
d2r = pi/180;
r2d = 180/pi;
M = 50; % The number of total strides
%% --------------- Part.I   Foot trajectory generation --------------------
% 1. Joint angle recording
Time = 1.071;  % Time period
N = 200; % Number of data points in a gait cycle
Ts = Time/N;  % Time step length
% Joint angles in degree
Hip_o = [24 23 22 16 9 4 -2 -7 -11 -15 -16 -13 -6 2 13 19 23 26 27 25 24];
Knee_o = [3 11 15 18 15 12 8 6 5 8 14 28 47 63 68 65 53 37 15 1 3];
Ankle_o = [-3 -13 -8 -3 1 3 6 7 8 9 4 -5 -18 -20 -13 -5 0 -1 -2 -2 -3];
T = 0: Ts: Time-Ts;
P = 0:0.005:1-0.005;
% 40th order interpolation
x = -14:1:36;
x_in = 1:0.1:20.9;
y1 = [Hip_o(1, 6:20) Hip_o Hip_o(2:16)];
y2 = [Knee_o(1, 6:20) Knee_o Knee_o(2:16)];
y3 = [Ankle_o(1, 6:20) Ankle_o Ankle_o(2:16)];
k = 40;
p1 = polyfit(x, y1, k);
p2 = polyfit(x, y2, k);
```

```
33 p3 = polyfit(x, y3, k);
34 Hip = polyval(p1, x_in);
35 Knee = polyval(p2, x_in);
36 Ankle = polyval(p3, x_in);
37 angle_foot = Hip - Knee + Ankle;
38 % Plot interpolation results
39 figure
40 plot(x_in, Hip);
41 hold on
42 plot(x_in, Knee);
43 plot(x_in, Ankle);
44 plot(x, y1, 'o');
45 plot(x, y2, 'o');
46 plot(x, y3, 'o');
47 % 2. Velocity parameterization
48 % vx direction  [percentage velocity acceleration]
49 x = [0 0.4 -11.1; 0.125 -3.2e-3 0; 0.185 0.0237 0; 0.275 -0.0206 0;...
50     0.395 0.0623 0; 0.49 0.2 0; 0.54 7.5e-3 0; 0.7 4.292 0;...
51     0.775 3.63 0; 0.865 4.988 0; 1 0.4 -11.1];
52 % vy direction  [percentage velocity acceleration]
53 y = [0 -0.314 -5.98; 0.02 -0.409 0; 0.125 0.0128 0; 0.185 -0.104 0;...
54     0.3 0 0; 0.4 -9.6e-3 -0.084; 0.5 -0.1 0; 0.615 1-0.2 0;...
55     0.745 -0.373 0; 0.79 -0.378 0; 0.905 1.227-0.2 0; 1 -0.664-0.35 -5.98];
56 p_x = zeros(length(x)-1, 4);
57 p_y = zeros(length(y)-1, 4);
58 x_f_l_d_i = zeros(N, 1);
59 y_f_l_d_i = zeros(N, 1);
60 % Interpolation of velocity
61 for i = 1:length(p_x)
62     p_x(i, :) = cubic_interpolation(x(i, :), x(i+1, :));
63     temp_l = [x(i, 1):0.005:x(i+1, 1)];
64     x_f_l_d_i(x(i, 1)/0.005+1:x(i+1, 1)/0.005+1) = polyval(p_x(i, :), ...
         temp_l);
```

```
end
for i = 1:length(p_y)
    p_y(i, :) = cubic_interpolation(y(i, :), y(i+1, :));
    temp_l = [y(i, 1):0.005:y(i+1, 1)];
    y_f_l_d_i(y(i, 1)/0.005+1:y(i+1, 1)/0.005+1) = polyval(p_y(i, :), ...
        temp_l);
end
% Displacement integration
x_f_l_i = cumsum(x_f_l_d_i(1:200))*Ts;
y_f_l_i = cumsum(y_f_l_d_i(1:200))*Ts;
figure
plot(P, x_f_l_i, 'r--');
hold on; grid on;
plot(P, y_f_l_i, 'r--');
xlabel('Percentage of gait cycle');
ylabel('Displacement [m]')
% Acceleration
x_f_l_dd_i = diff(x_f_l_d_i)/Ts;  % 200*1
y_f_l_dd_i = diff(y_f_l_d_i)/Ts;  % 200*1
figure
hold on; grid on;
plot(P, x_f_l_dd_i, 'r');
plot(P, y_f_l_dd_i, 'b');
xlabel('Percentage of gait cycle');
ylabel('Acceleration [m/s^2]')
% To generate longer trajectory based on single gait North East Down
O = zeros(200, 1);
trajectory_l = [x_f_l_i O y_f_l_i O angle_foot'*d2r O]';
temp_l = trajectory_l;
x_f_l_d_i = x_f_l_d_i(1:200);
y_f_l_d_i = y_f_l_d_i(1:200);
v_x = x_f_l_d_i;
v_y = y_f_l_d_i;
```

```
97  for i = 1:M-1
98      trajectory_l = [trajectory_l temp_l];
99      x_f_l_d_i = [x_f_l_d_i; v_x];
100     y_f_l_d_i = [y_f_l_d_i; v_y];
101 end
102 % Right foot position
103 d = 0.2;  % Distance between the two feet
104 x_ini = (x_f_l_i(1) - x_f_l_i(200))/2;
105 x_r = [x_ini+x_f_l_i(101:200)' -x_ini+x_f_l_i(1:100)']';
106 y_r = [y_f_l_i(101:200)' y_f_l_i(1:100)']';
107 angle_r = [angle_foot(101:200) angle_foot(1:100)]';
108 trajectory_r = [x_r ones(200, 1)*d y_r O angle_r*pi/180 O]';
109 % Velocity
110 x_f_r_d_i = [x_f_l_d_i(101:200); x_f_l_d_i(1:100)];
111 y_f_r_d_i = [y_f_l_d_i(101:200); y_f_l_d_i(1:100)];
112 temp_r = trajectory_r;
113 v_x = x_f_r_d_i;
114 v_y = y_f_r_d_i;
115 for i = 1:M-1
116     trajectory_r = [trajectory_r temp_r];
117     x_f_r_d_i = [x_f_r_d_i; v_x];
118     y_f_r_d_i = [y_f_r_d_i; v_y];
119 end
120 % Load data into true structure
121 a = simdata.a; % Earth radius
122 e2  = simdata.e2;  % EarthEccentricitySq
123 % Left foot
124 true.Pn_l     = trajectory_l(1:3,:);
125 true.rpyDeg_l = trajectory_l(4:6,:)*r2d;
126 true.dt     = Ts;
127 true.t      = ((1:size(true.Pn_l,2))-1)*true.dt;
128 true.q_b2n_l = Imu2YPRdeg_to_qb2n(true.rpyDeg_l(1,:),...
129                 true.rpyDeg_l(2,:),true.rpyDeg_l(3,:));
```

```
130 true.h_l  = true.Pn_l(3,:);
131 true.Vn_l = [x_f_l_d_i'; zeros(1, 200*M); -y_f_l_d_i'];  % North east down
132 lonDeg =  (simdata.longitude+d/a) * r2d;
133 latDeg =  simdata.latitude * r2d;
134 q_e2N0_l =  lonLatDegTo_q_e2N(lonDeg, latDeg);
135 true.q_e2N_l = integVel_into_q_e2N(q_e2N0_l, true.h_l, true.Vn_l, true.dt);
136 [true.lonDeg_l, true.latDeg_l] = q_e2N_toLonLatDeg(true.q_e2N_l);
137 true.LLA_l(1,:) = true.lonDeg_l*d2r;
138 true.LLA_l(2,:) = true.latDeg_l*d2r;
139 true.LLA_l(3,:) = true.h_l;
140 true.Northing_l = (true.LLA_l(2,:)-true.LLA_l(2,1))*a;
141 true.Easting_l  =(true.LLA_l(1,:)-true.LLA_l(1,1)).*cos(true.LLA_l(2,:))*a;
142 true.Down_l     =  -(true.LLA_l(3,:)-true.LLA_l(3,1));
143 % Right foot
144 true.Pn_r     = trajectory_r(1:3,:)*1;
145 true.rpyDeg_r = trajectory_r(4:6,:)*180/pi*1;
146 true.t        = ((1:size(true.Pn_r,2))-1)*true.dt;
147 true.q_b2n_r  = Imu2YPRdeg_to_qb2n(true.rpyDeg_r(1,:),...
148                 true.rpyDeg_r(2,:),true.rpyDeg_r(3,:));
149 true.h_r  = true.Pn_r(3,:);
150 true.Vn_r = [x_f_r_d_i'; zeros(1, 200*M); -y_f_r_d_i'];  % North east down
151 q_e2N0_r =  lonLatDegTo_q_e2N(lonDeg, latDeg);
152 true.q_e2N_r = integVel_into_q_e2N(q_e2N0_r, true.h_r, true.Vn_r, true.dt);
153 [true.lonDeg_r, true.latDeg_r] = q_e2N_toLonLatDeg(true.q_e2N_r);
154 true.LLA_r(1,:) = true.lonDeg_r*d2r;
155 true.LLA_r(2,:) = true.latDeg_r*d2r;
156 true.LLA_r(3,:) = true.h_r;
157 %
158 true.Northing_r = (true.LLA_r(2,:)-true.LLA_r(2,1))*a;
159 true.Easting_r  =(true.LLA_r(1,:)-true.LLA_r(1,1)).*cos(true.LLA_r(2,:))*a;
160 true.Down_r     =  -(true.LLA_r(3,:)-true.LLA_r(3,1));
161 %% ------------------ Part.II   IMU data generation -----------------------
162 % Left foot
```

```
dth_l = quatToAngl(q_mult(true.q_b2n_l(:,1:end-1),...
                    q_inv(true.q_b2n_l(:,2:end))));
dth_l = [dth_l(:,1), dth_l];
r_wrtE_e_l = LLA2rWrtEinE(true.LLA_l);
r_e_n_l    = quatRot(true.q_e2N_l,r_wrtE_e_l);
w_e2i_n_l  = zeros(3,length(true.t));
g_n_l      = zeros(3,length(true.t));
dg_centropital_l = zeros(3,length(true.t));
for i=1:length(true.t)
    w_e2i_n_l(:,i) = earthRateInBody(0,0,0, true.LLA_l(2,i)*180/pi);
    g_n_l(:,i) = gravityModel(true.LLA_l(2,i), norm(r_e_n_l(:,i)), a, e2);
    dg_centropital_l(:,i) = cross(w_e2i_n_l(:,i),...
                            cross(w_e2i_n_l(:,i), r_e_n_l(:,i)));
end
gl_n_l  = g_n_l - dg_centropital_l;
w_n2e_n_l(1,:) =  true.Vn_l(2,:)./a;
w_n2e_n_l(2,:) = -true.Vn_l(1,:)./a;
w_n2e_n_l(3,:) =  w_n2e_n_l(2,:)*0;
w_aux_l = 2 * w_e2i_n_l  +  w_n2e_n_l;
aux1_l = zeros(3,length(true.t));
for i=1:length(true.t)
    aux1_l(:,i) = cross(w_aux_l(:,i), true.Vn_l(:,i));
end
aux2_l = aux1_l - gl_n_l;
dvn_l = [zeros(3,1),diff(true.Vn_l,1,2)];
f_n_l = dvn_l/true.dt + aux2_l;
f_b_l = quatRot(q_inv(true.q_b2n_l),f_n_l);
dv_l  = f_b_l*true.dt;
w_n2i_n_l = w_n2e_n_l + w_e2i_n_l;
w_n2i_b_l = quatRot(q_inv(true.q_b2n_l),w_n2i_n_l);
dth_l = dth_l + w_n2i_b_l*true.dt;
% Right foot
dth_r = quatToAngl(q_mult(true.q_b2n_r(:,1:end-1),...
```

```
196                     q_inv(true.q_b2n_r(:,2:end))));
197 dth_r = [dth_r(:,1), dth_r];
198 r_wrtE_e_r = LLA2rWrtEinE(true.LLA_r);
199 r_e_n_r    = quatRot(true.q_e2N_r,r_wrtE_e_r);
200 w_e2i_n_r  = zeros(3,length(true.t));
201 g_n_r      = zeros(3,length(true.t));
202 dg_centropital_r = zeros(3,length(true.t));
203 for i=1:length(true.t)
204     w_e2i_n_r(:,i) = earthRateInBody(0,0,0, true.LLA_r(2,i)*180/pi);
205     g_n_r(:,i) = gravityModel(true.LLA_r(2,i), norm(r_e_n_r(:,i)), a, e2);
206     dg_centropital_r(:,i) = cross(w_e2i_n_r(:,i), ...
207                             cross(w_e2i_n_r(:,i), r_e_n_r(:,i)));
208 end
209 gl_n_r  = g_n_r - dg_centropital_r;
210 w_n2e_n_r(1,:) =  true.Vn_r(2,:)./a;
211 w_n2e_n_r(2,:) = -true.Vn_r(1,:)./a;
212 w_n2e_n_r(3,:) =  w_n2e_n_r(2,:)*0;
213 w_aux_r = 2 * w_e2i_n_r  +  w_n2e_n_r;
214 aux1_r = zeros(3,length(true.t));
215 for i=1:length(true.t)
216     aux1_r(:,i) = cross(w_aux_r(:,i), true.Vn_r(:,i));
217 end
218 aux2_r = aux1_r - gl_n_r;
219 dvn_r = [zeros(3,1),diff(true.Vn_r,1,2)];
220 f_n_r = dvn_r/true.dt + aux2_r;
221 f_b_r = quatRot(q_inv(true.q_b2n_r),f_n_r);
222 dv_r  = f_b_r*true.dt;
223 w_n2i_n_r = w_n2e_n_r + w_e2i_n_r;
224 w_n2i_b_r = quatRot(q_inv(true.q_b2n_r),w_n2i_n_r);
225 dth_r = dth_r + w_n2i_b_r*true.dt;
226
227 figure
228 plot(true.t, dth_l/true.dt);
```

```
229 xlabel('Time [s]');
230 ylabel('Gyro readout [deg/s]')
231 grid on
232
233 figure
234 plot(true.t, f_b_l);
235 xlabel('Time [s]');
236 ylabel('Accel readout [m/s^2]')
237 grid on
238 %% ------- Part.III   Predicted trajectory based on noiseless data --------
239 % Left foot
240 input.q_b2n           = true.q_b2n_l(:,1);
241 input.q_e2n           = true.q_e2N_l(:,1);
242 input.LLA             = true.LLA_l(:,1);
243 input.v_nWrtE_n       = true.Vn_l(:,1);
244 pred.q_b2n_l(:,1)     = input.q_b2n;
245 pred.q_e2n_l(:,1)     = input.q_e2n;
246 pred.LLA_l(:,1)       = input.LLA;
247 pred.v_nWrtE_n_l(:,1) = input.v_nWrtE_n;
248 pred.t                = true.t;
249 sensor.dt             = true.dt;
250 for i=2:length(pred.t)
251     sensor.w_b2i_b = dth_l(:,i)/sensor.dt;
252     sensor.f_b     = dv_l(:,i)/sensor.dt;
253     input = navSLN(sensor, input);
254     pred.q_b2n_l(:,i)     = input.q_b2n;
255     pred.q_e2n_l(:,i)     = input.q_e2n;
256     pred.LLA_l(:,i)       = input.LLA;
257     pred.v_nWrtE_n_l(:,i) = input.v_nWrtE_n;
258 end
259 pred.Northing_l =  (pred.LLA_l(2,:)-pred.LLA_l(2,1))*a;
260 pred.Easting_l  =  ...
       (pred.LLA_l(1,:)-pred.LLA_l(1,1)).*cos(pred.LLA_l(2,:))*a;
```

```
pred.Down_l     = -(pred.LLA_l(3,:)-pred.LLA_l(3,1));
[roll_deg_l, pitch_deg_l, yaw_deg_l] = qb2nImu2YPRdeg(pred.q_b2n_l);
pred.rpyDeg_l = [roll_deg_l; pitch_deg_l; yaw_deg_l];
% Right foot
input.q_b2n          = true.q_b2n_r(:,1);
input.q_e2n          = true.q_e2N_r(:,1);
input.LLA            = true.LLA_r(:,1);
input.v_nWrtE_n      = true.Vn_r(:,1);
input.Δ_b_Prev   = zeros(3,1);
input.Δ_n_Prev   = zeros(3,1);
input.Δ_n2e_Prev = zeros(3,1);
pred.q_b2n_r(:,1) = input.q_b2n;
pred.q_e2n_r(:,1) = input.q_e2n;
pred.LLA_r(:,1)   = input.LLA;
pred.v_nWrtE_n_r(:,1)    = input.v_nWrtE_n;
pred.t           = true.t;
sensor.dt        = true.dt;
for i=2:length(pred.t)
    sensor.w_b2i_b = dth_r(:,i)/sensor.dt;
    sensor.f_b     = dv_r(:,i)/sensor.dt;
    input = navSLN(sensor, input);
    pred.q_b2n_r(:,i)     = input.q_b2n;
    pred.q_e2n_r(:,i)     = input.q_e2n;
    pred.LLA_r(:,i)       = input.LLA;
    pred.v_nWrtE_n_r(:,i) = input.v_nWrtE_n;
end
pred.Northing_r =  (pred.LLA_r(2,:)-pred.LLA_r(2,1))*simdata.a;
pred.Easting_r  =  ...
    (pred.LLA_r(1,:)-pred.LLA_r(1,1)).*cos(pred.LLA_r(2,:))*simdata.a;
pred.Down_r     = -(pred.LLA_r(3,:)-pred.LLA_r(3,1));
[roll_deg_r, pitch_deg_r, yaw_deg_r] = qb2nImu2YPRdeg(pred.q_b2n_r);
pred.rpyDeg_r = [roll_deg_r; pitch_deg_r; yaw_deg_r];

```

```
293 %% -------------------- Part.IV   Add sensor error ------------------------
294 gyro.sigma_AWN = 2e-4*d2r*0;                % rad
295 gyro.sigma_ARW = simdata.sigma_g/sqrt(simdata.Ts);
296 gyro.sigma_RRW = simdata.gyro_bias_driving_noise/sqrt(simdata.Ts);
297 gyro.Bias      = simdata.sigma_initial_gyro_bias';
298 accl.sigma_VWN   = 4.5e-4*0;    % m/s
299 accl.sigma_VRW   = simdata.sigma_a/sqrt(simdata.Ts) * 1;
300 accl.sigma_AcRW  = simdata.acc_bias_driving_noise/sqrt(simdata.Ts) * 1;
301 accl.Bias      = simdata.sigma_initial_acc_bias';
302 N = length(pred.t);
303 gyro.dth_error_l = randn(3,N)*gyro.sigma_ARW*sqrt(true.dt) + ...
304                    randn(3,N)*gyro.sigma_AWN + gyro.Bias*true.dt + ...
305                    cumsum(randn(3,N),2)*gyro.sigma_RRW*sqrt(true.dt)*true.dt;
306 accl.dv_error_l = randn(3,N)*accl.sigma_VRW*sqrt(true.dt) + ...
307                   randn(3,N)*accl.sigma_VWN + accl.Bias*true.dt +...
308                   cumsum(randn(3,N),2)*accl.sigma_AcRW*sqrt(true.dt)*true.dt;
309 gyro.dth_error_r = randn(3,N)*gyro.sigma_ARW*sqrt(true.dt) + ...
310                    randn(3,N)*gyro.sigma_AWN + gyro.Bias*true.dt + ...
311                    cumsum(randn(3,N),2)*gyro.sigma_RRW*sqrt(true.dt)*true.dt;
312 accl.dv_error_r = randn(3,N)*accl.sigma_VRW*sqrt(true.dt) + ...
313                   randn(3,N)*accl.sigma_VWN + accl.Bias*true.dt +...
314                   cumsum(randn(3,N),2)*accl.sigma_AcRW*sqrt(true.dt)*true.dt;
315 meas.dth_l = dth_l + gyro.dth_error_l;
316 meas.dv_l  = dv_l  + accl.dv_error_l;
317 u_l = [meas.dth_l; meas.dv_l]./sensor.dt; % Noisy IMU reaout for left foot
318 meas.dth_r = dth_r + gyro.dth_error_r;
319 meas.dv_r  = dv_r  + accl.dv_error_r;
320 u_r = [meas.dth_r; meas.dv_r]./sensor.dt; % Noisy IMU reaout for right foot
```

C.2 Code for ZUPT-Aided Navigation

MATLAB code for ZUPT-aided pedestrian inertial navigation algorithm is listed below. More details can be found in Section 1.2.3.2, Section 5.2, and [176]. The original MATLAB code files can be found in `\\nitride.eng.uci.edu\NIST\Code_Yusheng\ZUPT_algorithm`.

```
% This is the main code for ZUPT-aided pedestrian inertial navigation
clc;
clear;
close all;
currDir = pwd;
addpath([currDir, '\lib']);
s=settings();
global simdata;
d2r = pi/180;
r2d = 180/pi;
%% ------------------------ Initialization ------------------------------
m = 1;  % Number of averaging of raw data to adjust sampling rate.
cal = 9600; % Number of initial time steps for calibration
zupt_switch = 1; % ZUPT switch 1 is on, 0 is off
comp1 = 1; % G-sensitivity compensation. 1 is compensated, 0 is not.
comp2 = 1; % Residaul velocity compensation. 1 is compensated, 0 is not
% Load data. row 1-3: accel data in m/s^2, row 4-6: gyro data in deg/s
load('100m_new_uncomp_40.mat');
% IMU data compensation
if (comp1 == 1)
    u = IMU_compensation_27(u, cal);
end
% Averaging the data to reduce the sampling rate if necessary
v = zeros(6, round(length(u)/m));
for i = 1:m:length(u)-m
```

```
26      for j = 1:6
27          v(j, (i-1)/m+1) = mean(u(j, i:i+m-1));
28      end
29  end
30  v(4:6, :) = v(4:6, :) * d2r;  % convert to rad/s
31  v = v';
32  N = length(v);
33  cal = round(cal/m);
34  % gravity
35  a      = simdata.a; % Earth major axis
36  e2     = simdata.e2;  % EarthEccentricitySq
37  cLat   = cos(simdata.latitude);
38  sLat   = sin(simdata.latitude);
39  ax     = 1-e2*sLat^2;
40  R_N    = a/sqrt(ax);
41  R_M    = R_N*(1-e2)/ax;
42  h      = simdata.altitude;
43  r_e_n  = [-R_N*e2*sLat*cLat; 0; -R_N*ax - h];
44  g_n    = gravityModel(simdata.latitude, norm(r_e_n), a, e2);
45  g      = norm(g_n);
46  g      = 9.796; % or just use other values
47  % Calculate inital roll and pitch angles
48  f=mean(v(2:cal,1:3));
49  roll=atan2(-f(2),-f(3));
50  pitch=atan2(f(1),sqrt(f(2)^2+f(3)^2));
51  roll_Deg = roll * r2d;
52  pitch_Deg = pitch * r2d;
53  yaw_Deg = simdata.init_heading * r2d;
54  q_b2n_0  = Imu2YPRdeg_to_qb2n(roll_Deg,pitch_Deg,yaw_Deg);
55  % Navigation state initialization
56  latDeg = simdata.latitude * r2d;
57  lonDeg = simdata.longitude * r2d;
58  h = simdata.altitude;
```

```
59 V_0 = simdata.init_vel; % Initial velocity
60 dt = simdata.Ts * m; % Time step length
61 true.t      = ((1:N)-1)*dt;
62 q_e2N =  lonLatDegTo_q_e2N(lonDeg, latDeg);
63 LLA_0 = [simdata.longitude; simdata.latitude; h];
64 w_e2i_n        = earthRateInBody(0,0,0, LLA_0(2)*r2d);
65 input.q_b2n     = q_b2n_0;
66 input.q_e2n     = q_e2N;
67 input.LLA       = LLA_0;
68 input.v_nWrtE_n = V_0;
69 sensor.dt       = dt;
70 true.dt         = dt;
71 est.t           = true.t;
72 est.q_b2n       = [input.q_b2n(:,1), zeros(4,N-1)];
73 est.q_e2n       = [input.q_e2n(:,1), zeros(4,N-1)];
74 est.LLA         = [input.LLA(:,1),   zeros(3,N-1)];
75 est.v_nWrtE_n   = [input.v_nWrtE_n(:,1),    zeros(3,N-1)];
76 LR              = zeros(1, N);  % Likelihood ratio
77 gamma           =  zeros(1, N);  % Likelihood ratio test threshold
78 xi              = zeros(1, N);
79 shock           = zeros(1, N);
80 gyro.sigma_AWN  = 2e-4*d2r*0;               % Angle White Noise in rad
81 gyro.sigma_ARW  = simdata.sigma_g/sqrt(simdata.Ts) * 1;
82 gyro.sigma_RRW  = simdata.gyro_bias_driving_noise/sqrt(simdata.Ts) * 1;
83 gyro.Bias       = simdata.sigma_initial_gyro_bias';
84 accl.sigma_VWN  = 4.5e-4*0;    % Velocity White Noise in m/s
85 accl.sigma_VRW  = simdata.sigma_a/sqrt(simdata.Ts) * 1;
86 accl.sigma_AcRW = simdata.acc_bias_driving_noise/sqrt(simdata.Ts) * 1;
87 accl.Bias       = simdata.sigma_initial_acc_bias';
88 e3 = [1,1,1];
89 Q_diag = [gyro.sigma_ARW^2*e3, accl.sigma_VRW^2*e3, e3*0, ...
90           gyro.sigma_RRW^2*e3, accl.sigma_AcRW^2*e3]*true.dt;
91 % System state: position, velocity, orientation, gyro bias, accel bias
```

```
92  x = [simdata.sigma_initial_att, simdata.sigma_initial_vel,...
93      simdata.sigma_initial_pos, simdata.sigma_initial_gyro_bias/3,...
94      simdata.sigma_initial_acc_bias/3] * 1;
95  P = diag(x.^2);
96  H         = zeros(3,size(P,1));
97  H(:,4:6) = eye(3,3); % Observation matrix
98  R         = diag(simdata.sigma_vel.^2);
99  aB         = zeros(3,1);
100 gB         = zeros(3,1);
101 dx         = zeros(size(P,1),1);
102 Id         = eye(size(P));
103 zupt       = zeros(1,N);  % ZUPTing marker
104 T          = zeros(1,N);   % Test statistics of ZUPTing detector
105 W          = simdata.Window_size;  % ZUPTing window size
106 O33        = zeros(3,3);
107 I33        = eye(3,3);
108 A11        = -skew(w_e2i_n);
109 input_cal = input;
110 in         = 0;  % Last time index when ZUPT is on
111 t_i        = 1.5/simdata.Ts; % Index range to for the maximum shock level
112 %% ------------------------- Navigation ---------------------------------
113 % First few second stationary for calibration
114 for i=2:cal
115     % z-axis gyro can be compensated or not due to observability issues
116     sensor.w_b2i_b = v(i, 4:6)' - gB; %.*[1; 1; 0];
117     sensor.f_b     = v(i, 1:3)' - aB;
118     if(i-t_i<1)
119         shock(i) = max(abs(v(1:i, 3)));  % Calculate shock level
120     else
121         shock(i) = max(abs(v(i-t_i:i, 3)));
122     end
123     if i > W - 1
124 %       [zupt(i-W+1:i),LR(i)]=SHOE_detector(v(i-W+1:i, :)',simdata.factor);
```

```
125         [zupt(i), LR(i), gamma(i)] = adaptive_detector(v(i-W+1:i, :)',...
126             xi(i-1), (i-in)*dt, zupt(i-1), gamma(i-1), shock(i));
127     end
128     if zupt(i) == 1
129         in = i;   % Record the index for last ZUPT
130     end
131     input_cal = navSLN(sensor, input_cal); % Stradown navigation algorithm
132     A14 = -quat2dcos(input_cal.q_b2n);
133     A21 =  skew(-A14*sensor.f_b);
134     A = [A11 O33 O33 A14  O33
135          A21 O33 O33 O33 -A14
136          O33 I33 O33 O33  O33
137          O33 O33 O33 O33  O33
138          O33 O33 O33 O33  O33];
139     F = expm(A*sensor.dt);
140     P = F*P*F' + diag(Q_diag);
141     if (zupt(i) == 1 && zupt_switch==1)
142 %         z = quatToAngl(q_mult(q_inv(pred.q_b2n(:,i)),input.q_b2n));% ZAPT
143         z  = input_cal.v_nWrtE_n; % ZUPT
144         K  = (P*H')/(H*P*H'+R);
145         P  = (Id-K*H)*P;
146         dx = K*z;
147         gB = gB + dx(10:12); % Update gyro bias
148         aB = aB + dx(13:15); % Update accel bias
149         input_cal.LLA(2) = input_cal.LLA(2) - dx(7)/a;
150         input_cal.LLA(1) = input_cal.LLA(1)-dx(8)/a/cos(input_cal.LLA(2));
151         input_cal.LLA(3) = input_cal.LLA(3) + dx(9);
152         input_cal.q_e2n  = lonLatDegTo_q_e2N...
153             (input_cal.LLA(1)*r2d, input_cal.LLA(2)*r2d);
154         input_cal.v_nWrtE_n = input_cal.v_nWrtE_n - dx(4:6);
155         input_cal.q_b2n  = q_mult(input_cal.q_b2n, theta2quat(-dx(1:3)));
156     end
157     P                   = (P+P')/2;
```

```
158     est.q_b2n(:,i)     = input_cal.q_b2n;
159     est.q_e2n(:,i)     = input_cal.q_e2n;
160     est.LLA(:,i)       = input_cal.LLA;
161     est.v_nWrtE_n(:,i) = input_cal.v_nWrtE_n;
162     kf.dx(:,i)         = dx;
163     kf.gB(:,i)         = gB;
164     kf.aB(:,i)         = aB;
165     kf.diagP(:,i)      = diag(P);
166     % Uncertainty of estimated velocity
167     xi(i) = input_cal.v_nWrtE_n'/ P(4:6, 4:6)*input_cal.v_nWrtE_n;
168 end
169 % Start moving
170 for i=cal+1:length(est.t)
171     sensor.w_b2i_b = v(i, 4:6)' - gB; %.*[1; 1; 0];
172     sensor.f_b     = v(i, 1:3)'  - aB;
173     z              = zeros(3, 1);
174     dx             = dx.*0;
175     if(i-t_i<1)
176         shock(i) = max(abs(v(1:i, 3)));
177     else
178         shock(i) = max(abs(v(i-t_i:i, 3)));
179     end
180     if i > W - 1
181 %       [zupt(i-W+1:i),LR(i)]=SHOE_detector(v(i-W+1:i, :)',simdata.factor);
182         [zupt(i), LR(i), gamma(i)] = adaptive_detector(v(i-W+1:i, :)', ...
183             xi(i-1), (i-in)*dt, zupt(i-1), gamma(i-1), shock(i));
184     end
185     if zupt(i) == 1
186         in = i;
187     end
188     input = navSLN(sensor, input);
189     A14 = -quat2dcos(input.q_b2n);
190     A21 =  skew(-A14*sensor.f_b);
```

```
191     A = [A11 O33 O33 A14  O33
192          A21 O33 O33 O33 -A14
193          O33 I33 O33 O33  O33
194          O33 O33 O33 O33  O33
195          O33 O33 O33 O33  O33];
196     F = expm(A*sensor.dt);
197     P = F*P*F' + diag(Q_diag);
198     if (zupt(i) == 1 && zupt_switch == 1)
199         %z = quatToAngl(q_mult(q_inv(pred.q_b2n(:,i)),input.q_b2n));
200         if (comp2 == 1 && LR(i)>1000) % Walking
201             z = input.v_nWrtE_n - quat2dcos(input.q_b2n)*simdata.dv;
202         else
203             z = input.v_nWrtE_n;
204         end
205         K  = (P*H')/(H*P*H'+R);
206         P  = (Id-K*H)*P;
207         dx = K*z;
208         gB = gB + dx(10:12);
209         aB = aB + dx(13:15);
210         input.LLA(2)     = input.LLA(2) - dx(7)/simdata.a;
211         input.LLA(1)     = input.LLA(1)-dx(8)/simdata.a/cos(input.LLA(2));
212         input.LLA(3)     = input.LLA(3) + dx(9);
213         input.q_e2n =lonLatDegTo_q_e2N(input.LLA(1)*r2d, input.LLA(2)*r2d);
214         input.v_nWrtE_n  = input.v_nWrtE_n - dx(4:6);
215         input.q_b2n      = q_mult(input.q_b2n, theta2quat(-dx(1:3)));
216     end
217     P  = (P+P')/2;
218     est.q_b2n(:,i)     = input.q_b2n;
219     est.q_e2n(:,i)     = input.q_e2n;
220     est.LLA(:,i)       = input.LLA;
221     est.v_nWrtE_n(:,i) = input.v_nWrtE_n;
222     kf.dx(:,i)         = dx;
223     kf.gB(:,i)         = gB;
```

```
224     kf.aB(:,i)          = aB;
225     kf.diagP(:,i)       = diag(P);
226     kf.z(:, i)          = z;
227     xi(i) = input.v_nWrtE_n'/ P(4:6, 4:6)*input.v_nWrtE_n;
228 end
229 est.Northing =  (est.LLA(2,:)-LLA_0(2,1))*simdata.a;
230 est.Easting  =  (est.LLA(1,:)-LLA_0(1,1)).*cos(est.LLA(2,:))*simdata.a;
231 est.Down     = -(est.LLA(3,:)-LLA_0(3,1));
232 [roll_deg, pitch_deg, yaw_deg] = qb2nImu2YPRdeg(est.q_b2n);
233 est.rpyDeg = [roll_deg; pitch_deg; yaw_deg];
234 est.rpyDeg = wrapTo180(est.rpyDeg);
235 %% ---------------------- Results Plotting -------------------------------
236 if (1)
237     figure
238     plot( est.t,  shock); grid; hold on;
239     ttlMsg = 'Shock level';
240     title(ttlMsg);
241     xlabel('Time [s]');
242     ylabel('Acceleration [m/s^2]');
243     set(gcf, 'Name', ttlMsg);
244 end
245 % ------------------------------------------------------------------------
246 if (1)
247     figure
248     plot( est.t,  xi); grid; hold on;
249     ttlMsg = 'Innovation covariance';
250     title(ttlMsg);
251     xlabel('Time [s]');
252     ylabel('Innovation');
253     set(gcf, 'Name', ttlMsg);
254 end
255 % ------------------------------------------------------------------------
256 if (1)
```

```
257     figure
258     subplot(3,1,1);
259     plot( est.t,  v(:, 1)); grid; hold on;
260     plot( est.t,  v(:, 2));
261     plot( est.t,  v(:, 3));
262     ylabel('Acceleration [m/s^2]');
263     subplot(3,1,2);
264     plot( est.t,  v(:, 4)*r2d); grid; hold on;
265     plot( est.t,  v(:, 5)*r2d);
266     plot( est.t,  v(:, 6)*r2d);
267     ylabel('Angular rate [deg/s]');
268     subplot(3,1,3);
269     plot( true.t,  zupt, 'r');
270     ylabel('ZUPT state');
271     xlabel('Time [s]');
272     ttlMsg = 'IMU readouts and ZUPT';
273     set(gcf, 'Name', ttlMsg);
274 end
275 % ------------------------------------------------------------------------
276 if (1)
277     yL = {'roll','pitch','azim.'};
278     figure
279     for i=1:3
280         subplot(3,1,i);
281         plot( est.t(cal+1:end), est.rpyDeg(i,cal+1:end), 'b');grid; hold on;
282         if (i==1)
283             ttlMsg = 'Roll,Pitch,Azim. [deg]';
284             title(ttlMsg);
285         end
286         ylabel(yL{i});
287     end
288     xlabel('Time [s]')
289     set(gcf, 'Name', ttlMsg);
```

```
290 end
291 % --------------------------------------------------------------------------
292 if (1)
293     yL = {'roll','pitch','azim.'};
294     figure
295     for i=1:3
296         subplot(3,1,i);
297         plot( est.t,    3*sqrt(kf.diagP(i, :))*r2d,  'b');grid; hold on;
298         plot( est.t,   -3*sqrt(kf.diagP(i, :))*r2d,  'b');
299         if (i==1)
300             ttlMsg = 'Euler angle uncertainty [deg]';
301             title(ttlMsg);
302         end
303         ylabel(yL{i});
304     end
305     xlabel('Time [s]')
306     set(gcf, 'Name', ttlMsg);
307 end
308 % --------------------------------------------------------------------------
309 if (1)
310     yL = {'(1)','(2)','(3)'};
311     figure
312     subplot(4,1,1);
313     plot(true.t(cal+1:end), zupt(cal+1:end), 'r');
314     hold on;
315     ttlMsg = 'ZUPT state';
316     title(ttlMsg);
317     for i=1:3
318         subplot(4,1,i+1);
319         plot( est.t(cal+1:end), est.v_nWrtE_n(i,cal+1:end), 'b');
320         grid on; hold on;
321         if (i==1)
322             ttlMsg = 'Velocity NED [m/s]';
```

```
323             title(ttlMsg);
324         end
325         ylabel(yL{i});
326     end
327     xlabel('Time [s]')
328     set(gcf, 'Name', ttlMsg);
329 end
330 % ------------------------------------------------------------------------
331 if (1)
332     yL = {'(1)','(2)','(3)'};
333     figure
334     for i=1:3
335         subplot(3,1,i);
336         plot( est.t(cal+1:end), kf.dx(3+i,cal+1:end), 'r');grid on; hold on;
337         plot( est.t(cal+1:end), kf.z(i,cal+1:end), 'b');
338         if (i==1)
339             ttlMsg = 'Velocity Correction NED [m/s]';
340             title(ttlMsg);
341         end
342         ylabel(yL{i});
343     end
344     xlabel('Time [s]')
345     set(gcf, 'Name', ttlMsg);
346 end
347 % ------------------------------------------------------------------------
348 if (1)
349     yL = {'(1)','(2)','(3)'};
350     figure
351     for i=1:3
352         subplot(3,1,i);
353         plot( est.t(cal+1:end), kf.dx(6+i,cal+1:end), 'r');
354         grid on; hold on;
355         plot( est.t(cal+1:end), kf.z(i,cal+1:end), 'b');
```

```
356         if (i==1)
357             ttlMsg = 'Position Correction NED [m]';
358             title(ttlMsg);
359         end
360         ylabel(yL{i});
361     end
362     xlabel('Time [s]')
363     set(gcf, 'Name', ttlMsg);
364 end
365 % ------------------------------------------------------------------------
366 if (1)
367     yL = {'(1)','(2)','(3)'};
368     figure
369     for i=1:3
370         subplot(3,1,i);
371         plot(est.t(cal+1:end), kf.dx(i,cal+1:end)*r2d, 'r');
372         grid on; hold on;
373         plot(est.t(cal+1:end), kf.z(i,cal+1:end), 'b');
374         if (i==1)
375             ttlMsg = 'Angle Correction [deg]';
376             title(ttlMsg);
377         end
378         ylabel(yL{i});
379     end
380     xlabel('Time [s]')
381     set(gcf, 'Name', ttlMsg);
382 end
383 % ------------------------------------------------------------------------
384 if (1)
385     yL = {'(1)','(2)','(3)'};
386     figure
387     subplot(3,1,1);
388     for i=1:3
```

```
389         subplot(3,1,i);
390         grid; hold on;
391         plot( est.t(cal+1:end),  3*sqrt(kf.diagP(i+3, cal+1:end)), 'b');
392         plot( est.t(cal+1:end), -3*sqrt(kf.diagP(i+3, cal+1:end)), 'b');
393         if (i==1)
394             ttlMsg = 'Error: Velocity NED [m/s]';
395             title(ttlMsg);
396         end
397         ylabel(yL{i});
398     end
399     xlabel('Time [s]')
400     set(gcf, 'Name', ttlMsg);
401 end
402 % ------------------------------------------------------------------------
403 if (1)
404     figure
405     semilogy( est.t,  LR, 'b');
406     grid on; hold on;
407     semilogy( est.t,  gamma, 'r-.');
408     xlabel('Time [s]');
409     ylabel('Likelihood ratio');
410     ttlMsg = 'Likelihood ratio';
411     title(ttlMsg);
412     set(gcf, 'Name', ttlMsg);
413 end
414 % ------------------------------------------------------------------------
415 if (1)
416     figure
417     plot( est.Easting(cal+1:end),  est.Northing(cal+1:end), 'b');
418     grid on; hold on; axis equal;
419     plot(est.Easting(cal+1), est.Northing(cal+1), 's');
420     plot(est.Easting(end), est.Northing(end), '*');
421     xlabel('Easting [m]');
```

```
422     ylabel('Northing [m]');
423     ttlMsg = 'Estimated and True Path, Northing-Easting [m]';
424     title(ttlMsg);
425     set(gcf, 'Name', ttlMsg);
426 end
427 % --------------------------------------------------------------------------
428 if (1)
429     figure
430     plot( est.t(cal+1:end),  est.Northing(cal+1:end), 'b');
431     grid on; hold on;
432     plot( est.t(cal+1:end),  est.Easting(cal+1:end), 'r');
433     ttlMsg = 'Displacement along east and north';
434     title(ttlMsg);
435     xlabel('Time [s]');
436     ylabel('Displacement [m]');
437     legend('Northing', 'Easting');
438     set(gcf, 'Name', ttlMsg);
439 end
440 % --------------------------------------------------------------------------
441 if (1)
442    yL = {'(1)','(2)','(3)'};
443    figure
444    subplot(4,1,1);
445    plot( true.t(cal+1:end),  zupt(cal+1:end), 'r'); hold on; grid on;
446    ttlMsg = 'ZUPT state';
447    title(ttlMsg);
448    for i=1:3
449        subplot(4,1,i+1);
450        plot( est.t(cal+1:end),  3*sqrt(kf.diagP(i+6, cal+1:end)), 'b');
451        grid on; hold on;
452        plot( est.t(cal+1:end), -3*sqrt(kf.diagP(i+6, cal+1:end)), 'b');
453        if (i==1)
454            ttlMsg = 'Error: Position NED [m]';
```

```
455             title(ttlMsg);
456         end
457         ylabel(yL{i});
458     end
459     xlabel('Time [s]')
460     set(gcf, 'Name', ttlMsg);
461 end
462 % ------------------------------------------------------------------------
463 if (1)
464     figure
465     plot( est.t(cal+1:end),  est.LLA(3,cal+1:end), 'b'); grid on; hold on;
466     xlabel('Time [sec]');
467     ylabel('Down [m]');
468     ttlMsg = 'Estimated and True Altitude [m]';
469     title(ttlMsg);
470     set(gcf, 'Name', ttlMsg);
471 end
472 % ------------------------------------------------------------------------
473 if (1)
474     yL = {'1','2','3'};
475     figure
476     for i=1:3
477         subplot(3,1,i);
478         plot( est.t,  kf.aB(i,:), 'r'); grid on; hold on
479         plot(est.t,  sqrt(kf.diagP(12+i,:))*3, 'b');
480         plot(est.t, -sqrt(kf.diagP(12+i,:))*3, 'b');
481         if (i==1)
482             ttlMsg = 'Accel bias [m/s^2]';
483             title(ttlMsg);
484         end
485         ylabel(yL{i});
486     end
487     xlabel('Time [s]')
```

```
488     set(gcf, 'Name', ttlMsg);
489 end
490 % ------------------------------------------------------------------------
491 if (1)
492     yL = {'1','2','3'};
493     figure
494     for i=1:3
495         subplot(3,1,i);
496         plot( est.t,  kf.gB(i,:) *r2d*3600, 'r'); grid on; hold on;
497         plot(est.t,  sqrt(kf.diagP(9+i,:))*r2d*3600*3, 'b');
498         plot(est.t, -sqrt(kf.diagP(9+i,:))*r2d*3600*3, 'b');
499         if (i==1)
500             ttlMsg = 'Error: Gyro bias [deg/h]';
501             title(ttlMsg);
502         end
503         ylabel(yL{i});
504     end
505     xlabel('Time [s]')
506     set(gcf, 'Name', ttlMsg);
507 end
```

C.3 Code for Sub-Functions

This section lists all sub-functions used in the code. The purpose and the descriptions of the input and output variables are also included.

C.3.1 adaptive_detector

```
function [zupt,LR,thre2] = adaptive_detector(u, xi, dt, ZUPT, thre1, shock)
% ------------------------------------------------------------------------
% This function calculates adaptive threshld for ZUPT detector
% Ref: Wang, et al. IEEE Sensors Letters, vol. 3, no. 11, pp. 1-4, 2019.
% Input: u:     IMU readouts at current time step
%        xi:    Uncertainty of estimated velocity
%        dt:    Time differnce beween last time ZUPT is on and current step
%        ZUPT:  ZUPT state of previous time step
%        thre1: Threshold of previous time step
%        shock: Maximum shock level of last step
% Output: zupt:  Detected ZUPT state. 1 is stance and 0 is swing
%         LR:    Likelihood ratio
%         thre2: Actual adaptive threshold
% ------------------------------------------------------------------------
    global simdata;
    alpha = simdata.alpha;
    theta = simdata.theta;
    beta = simdata.beta*0;
    p = [0.0364 7.9276];  % Derived from experimental results
    p = [0.032 7.9276];  % Adjusted according to differnet subjects
    temp = exp(p(1)*shock + p(2));
    [r, c] = size(u);
    W=simdata.Window_size;
    sigma2_a = (simdata.sigma_a/simdata.Ts)^2;
    sigma2_g = (simdata.sigma_g/simdata.Ts)^2;
    g = 9.796;
    u_n = mean(u(1:3, :), 2);
    u_n = u_n / norm(u_n);  % Unit vector along the specific force
    for i = 1:3
        u(i, :) = u(i, :) - g*u_n(i);
    end
    total = sum(sum(u(1:3,:).^2))/sigma2_a+sum(sum(u(4:6,:).^2))/sigma2_g;
    total = total/c;
```

```
34      thre2 = alpha + theta*temp*log(dt/0.005) - beta*xi;
35      % Hold the threshold during the stance phase
36      if (ZUPT == 0)
37          thre2 = alpha + theta*temp*log(dt/0.005) - beta*xi;
38      elseif ZUPT ==1
39          thre2 = thre1;
40      end
41      if(thre2 < 1000)
42          thre2 = 1000;
43      end
44      if(total < thre2)
45          zupt = 1;
46      else
47          zupt = 0;
48      end
49      LR = total;
50  end
```

C.3.2 b321tang

```
1   function [roll_deg, pitch_deg, yaw_deg] = b321tang(b)
2   % ------------------------------------------------------------------------
3   % This function converts DCM to Euler angles
4   % Ref: Titterton et al. Strapdown inertial navigation technology. (3.66)
5   % Input: b: DCM obtained by 321 rotation sequence, i.e. yaw-pitch-roll
6   % Output: roll_deg:  'roll' angle about 1-axis (x-axis) in deg,
7   %         pitch_deg: 'pitch' angle about 2-axis (y-axis) in deg,
8   %         yaw_deg:   'yaw' angle about 3-axis (z-axis) in deg,
9   % ------------------------------------------------------------------------
10      r2d = 180/pi;
```

```
11      roll_deg  = atan2(b(2,3), b(3,3)) * r2d;
12      aux = sqrt(1.0 - b(1,3)^2 );
13      pitch_deg = atan2(-b(1,3), aux) * r2d;
14      yaw_deg   = atan2(b(1,2), b(1,1)) * r2d;
15  end
```

C.3.3 Ce2n_toLLA

```
1   function [Lon_deg, Lat_deg] = Ce2n_toLLA(C_e2n)
2   % ------------------------------------------------------------------------
3   % This function calculates longgitude and latitude from q_e2n
4   % Input: C_e2n: quaternion from e-frame to n-frame
5   % Output: Lon_deg: Longitude in deg
6   %         Lat_deg: Latitude in deg
7   % ------------------------------------------------------------------------
8       D13 = C_e2n(3,1);
9       D23 = C_e2n(3,2);
10      Lon_deg = atan2(-D23,-D13)*180/pi;
11      D31 = C_e2n(1,3);
12      D32 = C_e2n(2,3);
13      D33 = C_e2n(3,3);
14      aux = sqrt(D31*D31 + D32*D32);
15      Lat_deg = atan2(-D33, aux)*180/pi;
16  end
```

C.3.4 cubic_interpolation

```
function p = cubic_interpolation( d1, d2 )
% ------------------------------------------------------------------------
% This funciton calculates the coefficient of the cubic
% interpolation, given the value and slope of both ends
% Input:  d1&d2  1*3 array [x coordinate, y coordinate, dy/dx]
% Output: p      1*4 array [a b c d]
% ------------------------------------------------------------------------
    x1 = d1(1);
    x2 = d2(1);
    y1 = d1(2);
    y2 = d2(2);
    y1_d = d1(3);
    y2_d = d2(3);
    A = [x1^3   x1^2 x1 1;...
         x2^3   x2^2 x2 1;...
         3*x1^2 2*x1 1  0;...
         3*x2^2 2*x2 1  0];
    b = [y1; y2; y1_d; y2_d];
    p = A\b;
    p = p';
end
```

C.3.5 dcos2quat

```
function [q,a1,a2,a3,a4,b,ei] = dcos2quat( d, enblNormalization )
% ------------------------------------------------------------------------
% This function converts DCM to quaternion.
% Ref: Shepperd, Journal of Guidance and Control, 1 (3), pp. 223-224, 1978.
% Ipnut: d:                 DCM mapping frame A to frame B
%        enblNormalization: normalizaiton swtich
```

```
% Output: q: quaternion dexcribing the orientation of B with respect to A.
% ------------------------------------------------------------------------
    if (nargin == 1)
        enblNormalization = 1;
    end
    q = zeros(4,1);
    a1 = d(1,1) + d(2,2) + d(3,3);
    a2 = d(1,1) - d(2,2) - d(3,3);
    a3 = -d(1,1) + d(2,2) - d(3,3);
    a4 = -d(1,1) - d(2,2) + d(3,3);
    b  = max([a1; a2; a3; a4 ]);
    ei = 0.5*sqrt(1 + b);
    inv_4ei = 0.25/ei;

    if b == a1
        q(1) = (d(2,3) - d(3,2))*inv_4ei;
        q(2) = (d(3,1) - d(1,3))*inv_4ei;
        q(3) = (d(1,2) - d(2,1))*inv_4ei;
        q(4) = ei;
    elseif b == a2
        q(1) = ei;
        q(2) = (d(1,2) + d(2,1))*inv_4ei;
        q(3) = (d(3,1) + d(1,3))*inv_4ei;
        q(4) = (d(2,3) - d(3,2))*inv_4ei;
    elseif b == a3
        q(1) = (d(1,2) + d(2,1))*inv_4ei;
        q(2) = ei;
        q(3) = (d(2,3) + d(3,2))*inv_4ei;
        q(4) = (d(3,1) - d(1,3))*inv_4ei;
    else
        q(1) = (d(3,1) + d(1,3))*inv_4ei;
        q(2) = (d(3,2) + d(2,3))*inv_4ei;
        q(3) = ei;
```

```
40          q(4) = (d(1,2) - d(2,1))*inv_4ei;
41      end
42      % ensure q is canonical
43      fix = sign(q(4));
44      if fix==0
45          fix=1;
46      end
47      q(1) = fix*q(1);
48      q(2) = fix*q(2);
49      q(3) = fix*q(3);
50      q(4) = fix*q(4);
51      if (enblNormalization == 1)
52          if (abs(norm(q)) > 1e-40)
53             q = q/norm(q);
54          else
55             q = [0;0;0;1];
56          end
57      end
```

C.3.6 earthRateInBody

```
1  function w_e2i_b = earthRateInBody(roll,pitch,yaw,latitude)
2  % ------------------------------------------------------------------------
3  % This function calculates the earth rotation in body frame
4  % Input: roll, pitch, and yaw are Euler angle in deg
5  %        latitude is the latitude in deg
6  % Output: w_e2i_b is the Earth rotation vector described in b-frame
7  % ------------------------------------------------------------------------
8  global simdata
9  Omega = simdata.earthrate;
```

```
d2r = pi/180;

lat_rad   = latitude*d2r;
roll_rad  = roll*d2r;
pitch_rad = pitch*d2r;
yaw_rad   = yaw*d2r;
w_e2i_n = [cos(lat_rad);0;-sin(lat_rad)] * Omega;
C_roll  = trueCosine([roll_rad;0;0]);
C_pitch = trueCosine([0;pitch_rad;0]);
C_yaw   = trueCosine([0;0;yaw_rad]);

C_n2b = C_roll * C_pitch * C_yaw;

w_e2i_b = C_n2b * w_e2i_n;
```

C.3.7 euler1

```
function m = euler1(i,a)
% ------------------------------------------------------------------------
% This fuction calculates elementary DCM of rotation along one axis
% Input: i: the rotation axis, can be 1, 2, or 3
%        a: rotation angle in rad
% Output: m: DCM
% ------------------------------------------------------------------------
    ca=cos(a);
    sa=sin(a);
    if i==1 % rotation about x-axis
       m=[1, 0, 0; 0, ca, sa; 0,-sa, ca];
    end
    if i==2 % rotation about y-axis
```

```
14          m=[ca, 0,-sa; 0, 1, 0; sa, 0, ca];
15      end
16      if i==3 % rotation about z-axis
17          m=[ ca, sa, 0; -sa, ca, 0; 0, 0, 1];
18      end
19      if i>3 || i<1
20        error('euler called with bad number for axis');
21      end
22  end
```

C.3.8 gravityModel

```
1   function g = gravityModel(Lat, P, a, e2)
2   % -------------------------------------------------------------------------
3   % This function calculates the local gravity vector
4   % Ref: WGS84 Gravity Model
5   % Input: Lat: Latitude in rad
6   %        P:   distance to the Earth center in m
7   %        a:   Earth radius in m
8   %        e2:  Earth Eccentricity
9   % Output: g: gravity vector
10  % -------------------------------------------------------------------------
11      GM  = 3.986005e14; % m^3/s^2 GravitationalConstant
12      c20 = -sqrt(5)*4.8416685e-4;
13      Latc = atan((1-e2)*tan(Lat));
14      % geocentric
15      P2 = P*P;
16      a2 = a*a;
17      ax = 3*c20*a2/P2;
18      GMoverP2 = GM/P2;
```

```
    sLatc = sin(Latc);
    cLatc = cos(Latc);
    g_n = GMoverP2 * ax*sLatc*cLatc;
    g_d =  GMoverP2 * (1 + ax/2*(3*sLatc^2-1));
    % geodetic
    Alpha = Lat - Latc;
    cAlpha = cos(Alpha);
    sAlpha = sin(Alpha);
    g_N =  g_n*cAlpha + g_d*sAlpha;
    g_D = -g_n*sAlpha + g_d*cAlpha;
    g = [g_N; 0; g_D];
end
```

C.3.9 Imu2YPRdeg_to_qb2n

```
function [ q_b2n ] = Imu2YPRdeg_to_qb2n(roll_deg, pitch_deg, yaw_deg)
% -------------------------------------------------------------------------
% This function converts roll, pitch, yaw angles to quaternion
% Input: roll_deg, pitch_dg, yaw_deg are Euler angles in deg
% Output: q_b2n quaternion from n-frame to b-frame, in order of 3-2-1
% -------------------------------------------------------------------------
    d2r = pi/180;
    N1 = length(roll_deg);
    N2 = length(pitch_deg);
    N3 = length(yaw_deg);
    N = max([N1,N2,N3]);
    q_b2n = zeros(4,N);
    for i=1:N
        Cz = euler1(3,yaw_deg(i)*d2r);  % Convert to DCM first
        Cy = euler1(2,pitch_deg(i)*d2r);
```

```
16          Cx = euler1(1,roll_deg(i)*d2r);
17          C_n2b = Cx*Cy*Cz;
18          q = dcos2quat( C_n2b');  % Convert from DCM to quaternion
19          q_b2n(:,i) =  q_canonicalize(q);
20      end
21  end
```

C.3.10 IMU_compensation_27

```
1   function output = IMU_compensation_27 (input, cal)
2   % ------------------------------------------------------------------------
3   % This function calibrates IMU readouts
4   % Calibration parameters are for VN-200 28727, tested on Oct. 09, 2019
5   % Input: input: 1-3 row are accel in m/s^2, and 4-6 rows are gyro in deg/s
6   %        cal:   Length of initial calibration to remove bias
7   % Output: ouput: Calibrated IMU readout, format is the same as input
8   % ------------------------------------------------------------------------
9       N = length(input); % Data length
10      output = input;
11      %  Accel calibration
12      % Accel bias
13      b_a = [0.0945; -0.0989; -0.0398];
14      % Accel misalignment
15      M_a = [ 1.0104   0.0061   0.0062; ...
16             -0.0034   0.9941   0.0113; ...
17             -0.0104  -0.0230   0.9754];
18      % Compensate
19      for i = 1:N
20          output(1:3, i) = M_a\(input(1:3, i)-b_a);
21      end
```

```
    % Gyro calibration
    b_g = mean(input(4:6, 1:cal), 2);
    % Gyro misalignment
    M_g = [ 0.9991  0.0006 -0.0031; ...
           -0.0029  0.9979  0.0067; ...
           -0.0004 -0.0055  1.0000];
    % Gyro g-sensitivity   % in deg/s/m/s^2
    G_g = [0.0042 -0.0001 -0.0012; ...
           0.0001  0.0020 -0.0006; ...
          -0.0010 -0.0002 -0.0027];
    % Compensate
    for i = 1:N
        output(4:6, i) = M_g\(input(4:6, i)-b_g-G_g*output(1:3, i));
    end
    % Secondary compensation
    omega = mean(output(4:6, 1:cal), 2);  % Averaged gyro readout
    output(4, :) = output(4, :) - omega(1);
    output(5, :) = output(5, :) - omega(2);
    output(6, :) = output(6, :) - omega(3);
end
```

C.3.11 integVel_into_q_e2N

```
function q_e2N = integVel_into_q_e2N(q_e2N0, h, Vn, dt)
% ------------------------------------------------------------------------
% This function calculates the quaternion considering transport rate
% Input: q_e2N0: Original quaternion
%        h:      Altitude in [m]
%        Vn:     Velocity in n-frame [m/s]
%        dt:     time step length
```

```
% Ourput: q_e2N: Output quaternion
% ------------------------------------------------------------------------
    global simdata
    a = simdata.a + h;
    dth = [Vn(2,:)./a; -Vn(1,:)./a;  zeros(1,length(h))]*dt;
    dth = (dth + [dth(:,2:end), dth(:,end)])/2;
    dq = theta2quat(dth);
    q_e2N = zeros(4,length(h));
    q_e2N(:,1) = q_e2N0;
    for i=2:length(h)
        q_e2N(:,i) = q_mult(q_e2N(:,i-1), dq(:,i-1));
    end
end
```

C.3.12 LLA2rWrtEinE

```
function r_wrtE_e = LLA2rWrtEinE(LLA, ...
                    EarthSemiMajorAxis, EarthEccentricitySq )
% ------------------------------------------------------------------------
% This function calculates the position in e-frame based on LLA
% Note, WGS84 model is used
% Input: LLA:                Longtitude, latitude, and altitude, in ...
    rad, m
%        EarthSemiMajorAxis:  Earth amjor axis
%        EarthEccentricitySq: Earth Eccentricity Square
% Ourput: r_wrtE_e: Output position vector
% ------------------------------------------------------------------------
    global simdata
    if (nargin == 1)
       EarthSemiMajorAxis  = simdata.a;
```

```
14          EarthEccentricitySq = simdata.e2;
15      end
16      cLon  = cos(LLA(1,:));
17      sLon  = sin(LLA(1,:));
18      cLat  = cos(LLA(2,:));
19      sLat  = sin(LLA(2,:));
20      R_N   = EarthSemiMajorAxis./sqrt(1-EarthEccentricitySq*sLat.^2);
21      h = LLA(3,:);
22      r_wrtE_e = [(R_N + h).*cLat.*cLon;
23                  (R_N + h).*cLat.*sLon;
24                  (R_N*(1-EarthEccentricitySq) + h).*sLat];
25  end
```

C.3.13 lonLatDegTo_q_e2N

```
1   function q_e2N = lonLatDegTo_q_e2N(lonDeg, latDeg)
2   % -------------------------------------------------------------------------
3   % This function calculates quaternion from longitude and latitude
4   % Input: lonDeg: Longitude in deg
5   %        latDeg: Latitude in deg
6   % Output: q_e2N: quaternion from e-frame to n-frame
7   % -------------------------------------------------------------------------
8       q_e2N = zeros(4,length(lonDeg));
9       for i=1:length(lonDeg)
10          cLon = cosd( lonDeg(i) ); sLon = sind( lonDeg(i) );
11          cLat = cosd( latDeg(i) ); sLat = sind( latDeg(i) );
12          C_e2N = [-sLat*cLon  -sLat*sLon  cLat;
13                   -sLon        cLon       0;
14                   -cLon*cLat  -cLat*sLon  -sLat]; % Calculate DCM first
15          q_e2N = dcos2quat(C_e2N); % Convert from DCM to quaternion
```

```
16      end
17  end
```

C.3.14 navSLN

```
1  function output = navSLN( sensor, input )
2  % ------------------------------------------------------------------------
3  % This function conducts strapdown inertial navigation algorithm
4  % Input: sensor: IMU data: gyro readout:  w_b2i_b
5  %                          accel readout: f_b
6  %                          time step:     dt
7  %        input: initial state: orientation quaternion:        q_b2n
8  %                              position quaternion:           q_e2n
9  %                              Longitude, latitude, altitude: LLA
10 %                              velocity:                      v_nWrtE_n
11 % Output: output: final state, strucutre the same as input
12 % ------------------------------------------------------------------------
13 global simdata;
14 w_b2i_b    = sensor.w_b2i_b;
15 f_b        = sensor.f_b;
16 dt         = sensor.dt;
17 q_b2n      = input.q_b2n;
18 q_e2n      = input.q_e2n;
19 LLA        = input.LLA;  % [lonRad; latRad; alt]
20 v_nWrtE_n  = input.v_nWrtE_n;
21
22 Omega = simdata.earthrate;           % rad/s (earth's rate)
23 a     = simdata.a;               % m ( semi-major axis)
24 e2    = simdata.e2;
25 cLat = cos(LLA(2));
```

```
26 sLat = sin(LLA(2));
27 w_e2i_n = Omega*[cLat; 0; -sLat];
28 ax   = 1-e2*sLat^2;
29 R_N  = a./sqrt(ax);
30 R_M  = R_N*(1-e2)/ax;
31 h    = LLA(3);
32 dotLLA(3) = -v_nWrtE_n(3);
33 w_n2e_n(1,:) =  v_nWrtE_n(2)/a;
34 w_n2e_n(2,:) = -v_nWrtE_n(1)/a;
35 w_n2e_n(3,:) =   0;
36 % ------------------------ Attitude -------------------------------------
37 dTheta_b      = w_b2i_b * dt;
38 q = qintegrator(q_inv(q_b2n), dTheta_b, 0);
39 q_b2nNm_wErth = q_inv(q);
40 w_n2i_n       = w_e2i_n + w_n2e_n;
41 dTheta_n      = w_n2i_n * dt;
42 q_b2n = qintegrator(q_b2nNm_wErth, dTheta_n,0);
43 % -------------------------- Vel.& Pos. ---------------------------------
44 % Gravity
45 r_e_n          = [-R_N*e2*sLat*cLat; 0; -R_N*ax - h];
46 g_n            = gravityModel(LLA(2), norm(r_e_n), a, e2);
47 dg_centropital = cross(w_e2i_n, cross(w_e2i_n, r_e_n));
48 gl_n           = g_n - dg_centropital;
49 % Velocity change
50 w_aux     = 2 * w_e2i_n  +  w_n2e_n;
51 f_n       = quatRot(q_b2n, f_b);
52 v_nWrtE_n = v_nWrtE_n  + (f_n - cross(w_aux, v_nWrtE_n) + gl_n) * dt ;
53 % Position change
54 dotLLA(3) = -v_nWrtE_n(3);
55 w_n2e_n(1,:) =  v_nWrtE_n(2)/a;
56 w_n2e_n(2,:) = -v_nWrtE_n(1)/a;
57 w_n2e_n(3,:) =   0;
58 dTheta_n2e_n = w_n2e_n * dt;
```

```
59 q_e2n = qintegrator(q_e2n, dTheta_n2e_n,0);
60 [lonDeg, latDeg] = q_e2N_toLonLatDeg(q_e2n);
61 LLA(1) = lonDeg*pi/180;
62 LLA(2) = latDeg*pi/180;
63 LLA(3) = LLA(3) +  dotLLA(3) * dt;
64 % ---------------------------- Output ------------------------------------
65 output.q_b2n          = q_b2n;
66 output.q_e2n          = q_e2n;
67 output.LLA            = LLA;
68 output.v_nWrtE_n      = v_nWrtE_n;
```

C.3.15 q_canonicalize

```
1  function  q_out = q_canonicalize(q_in)
2  % ------------------------------------------------------------------------
3  % This function makes sure the scalar part of the quaternion is positive
4  % Input: q_in: Input quaternion, defined as: a*i + b*j + c*k + d
5  % Output: q_out: output quaternion, defined as: a*i + b*j + c*k + d
6  % ------------------------------------------------------------------------
7      q_out = q_in;
8      k = find(q_in(4,:) < 0);
9      if isempty(k)
10         return
11     else
12         q_out(:,k) = -1 * q_out(:,k);
13     end
14 end
```

C.3.16 q_e2N_toLonLatDeg

```
function [lonDeg, latDeg] = q_e2N_toLonLatDeg(q_e2N)
% -------------------------------------------------------------------------
% This function calculates longitude and latitude from quaternion
% Input: q_e2N: quaternion from e-frame to n-frame
% Output: lonDeg: Longitude in deg
%         latDeg: Latitude in deg
% -------------------------------------------------------------------------
    N = size(q_e2N,2);
    lonDeg = zeros(1,N);
    latDeg = zeros(1,N);
    for i=1:N
        C_e2n = quat2dcos(q_e2N(:,i));
        [lonDeg(i), latDeg(i)] = Ce2n_toLLA(C_e2n);
    end
end
```

C.3.17 q_inv

```
function  q_out = q_inv(q_in)
% -------------------------------------------------------------------------
% This function calculates the inverse of a quaternion
% Input: q_in: Input quaternion, defined as: a*i + b*j + c*k + d
% Output: q_out: output quaternion, defined as: a*i + b*j + c*k + d
% -------------------------------------------------------------------------
    q_out = q_in;
    q_out(1:3,:) = -1 * q_out(1:3,:);
end
```

C.3.18 q_mult

```
function c_out = q_mult(a, b)
% ------------------------------------------------------------------------
% This function calculates the multiplication of two quaternions
% Ref: Titterton et al. Strapdown inertial navigation technology. (3.55)
% Input: a: input quaternion 1
%        b: input quaternion 2
% Output: c: output quaternion
% ------------------------------------------------------------------------
    a1 = a(1,:);
    a2 = a(2,:);
    a3 = a(3,:);
    a4 = a(4,:);
    b1 = b(1,:);
    b2 = b(2,:);
    b3 = b(3,:);
    b4 = b(4,:);
    d1 =   a4.*b1  -  a3.*b2  +  a2.*b3  +  a1.*b4 ;
    d2 =   a3.*b1  +  a4.*b2  -  a1.*b3  +  a2.*b4 ;
    d3 = - a2.*b1  +  a1.*b2  +  a4.*b3  +  a3.*b4 ;
    d4 = - a1.*b1  -  a2.*b2  -  a3.*b3  +  a4.*b4 ;
    c = [d1(:)'; d2(:)'; d3(:)'; d4(:)'];
    c_out = q_canonicalize( c );
end
```

C.3.19 qb2nImu2YPRdeg

```
function [roll_deg, pitch_deg, yaw_deg] = qb2nImu2YPRdeg(q_b2n, C_bRef2b)
```

```
% ------------------------------------------------------------------------
% This function calculates Euler angles from quaternion
% Input: q_b2n:    quaternion from n-frame to b-frame in order 3-2-1
%        C_bRef2b: an optional constant offset of the b-frame
% Output: roll_deg, pitch_deg, and yaw_deg are Euler angles in deg
% ------------------------------------------------------------------------
    if (nargin==1)
       C_bRef2b = eye(3,3);
    end
    [n1,N] = size(q_b2n);
    roll_deg  = zeros(1,N);
    pitch_deg = roll_deg;
    yaw_deg   = roll_deg;
    for i=1:N
        nq = norm(q_b2n(:,i));
        q_b2n_use = q_b2n(:,i);
        if (nq>0.0)
           q_b2n_use = q_b2n_use/nq;
        end
        C_b2n = quat2dcos( q_b2n_use );
        C_bRef2n = C_b2n * C_bRef2b;
        [roll_deg(i), pitch_deg(i), yaw_deg(i)] = b321tang( C_bRef2n' );
    end
end
```

C.3.20 qintegrator

```
function q_o = qintegrator(q, dTheta, flag)
% ------------------------------------------------------------------------
% This function calculates the quaternion after an angle increment
```

```
% Ref: Titterton et al. Strapdown inertial navigation technology. (3.61)
% Input : q:      initial quaternion
%         dTheta: current angle increment
%         flag:   methods to calculate, default is 0
% Output : q_o: propagated quaternion
% ------------------------------------------------------------------------
    if (nargin < 3)
       flag = 0;
    end
    nrm_dth = norm(dTheta);
    if (nrm_dth < 1e-40)
        e = zeros(3,1);
    else
        e = dTheta/nrm_dth;
    end
    switch (flag)
        case 0  % Direct quaternion multiplication
             dq0 = [e(:)*sin(nrm_dth/2); cos(nrm_dth/2)];
             q_o = q_mult(q, dq0);
        case 1  % Quaternion propagation
             E = [0     e(3)  -e(2)  e(1);
                 -e(3)   0     e(1)  e(2);
                  e(2) -e(1)    0    e(3);
                 -e(1) -e(2)  -e(3)   0];
             dTh = E*nrm_dth;
             q_o = q + 0.5*dTh*q;
        case 2   % Quaternion multiplication in matrix form
             E = [0     e(3)  -e(2)  e(1);
                 -e(3)   0     e(1)  e(2);
                  e(2) -e(1)    0    e(3);
                 -e(1) -e(2)  -e(3)   0];
             q_o = (eye(4,4)*cos(nrm_dth/2)+sin(nrm_dth/2)*E)*q;
    end
```

```
37      q_o = q_o * (1.5 - 0.5 * q_o'*q_o ); % To make the norm ...
            approximately 1
38  end
```

C.3.21 quat2dcos

```
1   function DCM = quat2dcos(q)
2   % ------------------------------------------------------------------------
3   % This function calculates DCM from quaternion
4   % Ref: Titterton et al. Strapdown inertial navigation technology. (3.63)
5   % Input: q: quaternions in the form of a*i + b*j + c*k + d
6   % Output: DCM
7   % ------------------------------------------------------------------------
8       q1  = q(1);
9       q2  = q(2);
10      q3  = q(3);
11      q4  = q(4);
12      a11 = q1*q1 - q2*q2 - q3*q3 + q4*q4;
13      a12 = 2*(q1*q2 + q3*q4);
14      a13 = 2*(q1*q3 - q2*q4);
15      a21 = 2*(q1*q2 - q3*q4);
16      a22 = -q1*q1 + q2*q2 - q3*q3 + q4*q4;
17      a23 = 2*(q2*q3 + q1*q4);
18      a31 = 2*(q1*q3 + q2*q4);
19      a32 = 2*(q2*q3 - q1*q4);
20      a33 = -q1*q1 - q2*q2 + q3*q3 + q4*q4;
21      DCM = [a11 a12 a13;
22             a21 a22 a23;
23             a31 a32 a33];
24  end
```

C.3.22 quatRot

```
function [ W ] = quatRot(q, V)
% ------------------------------------------------------------------------
% This function calculates vector transformation by a quaternion: W = q ...
    * V
% Ref: Titterton et al. Strapdown inertial navigation technology. (3.59)
% Input: q: quaternions in the form of a*i + b*j + c*k + d
%        V: input 3D vector
% Output: W: output 3D vector
% ------------------------------------------------------------------------
    q0 = q(4,:);
    q1 = q(1,:);
    q2 = q(2,:);
    q3 = q(3,:);
    ax0  = 2*q0.^2;
    ax12 = 2*q1.*q2;
    ax30 = 2*q3.*q0;
    ax23 = 2*q2.*q3;
    ax20 = 2*q2.*q0;
    ax13 = 2*q1.*q3;
    ax10 = 2*q1.*q0;
    Q11 = ax0 - 1 + 2*q1.^2;
    Q12 = ax12 + ax30;
    Q13 = ax13 - ax20;
    Q21 = ax12 - ax30;
    Q22 = ax0 - 1 + 2*q2.^2;
    Q23 = ax23 + ax10;
    Q31 = ax13 + ax20;
    Q32 = ax23 - ax10;
    Q33 = ax0 - 1 + 2*q3.^2;
    X = V';
```

```
30      w1 = Q11(:).* X(:,1) + Q12(:).* X(:,2) + Q13(:).*X(:,3);
31      w2 = Q21(:).* X(:,1) + Q22(:).* X(:,2) + Q23(:).*X(:,3);
32      w3 = Q31(:).* X(:,1) + Q32(:).* X(:,2) + Q33(:).*X(:,3);
33      W = [w1(:)'; w2(:)'; w3(:)'];
34  end
```

C.3.23 quatToAngl

```
1   function thetaArray = quatToAngl(qArray)
2   % ------------------------------------------------------------------------
3   % This function converts quaternion to rotation vector
4   % Input: qArray: quaternion in the form of a*i + b*j + c*k + d
5   % Output: thetaArray: rotation vector in rad
6   % ------------------------------------------------------------------------
7       N = size(qArray,2);
8       thetaArray = zeros(3,N);
9       for i=1:N
10          q = qArray(:,i);
11          x = q(1:3);
12          xNorm = norm(x);
13          theta = zeros(3,1);
14          if xNorm>0
15              angl = atan2(xNorm, q(4))*2;
16              theta = x/xNorm*angl;
17          end
18          thetaArray(:,i) = theta(:);
19      end
20  end
```

C.3.24 settings

```
function s=settings
% This function is for setting the parameters used in the navigation
% The output s is useless
    s=0;
    global simdata;
    % General parameters
    simdata.Ts        = 1/800; % Sampling period [s]
    simdata.earthrate = 7.292115060085166e-5; % Earth rotation rate [rad/s]
    simdata.a         = 6378137; % Earth radius [m]
    simdata.e2        = 6.694380004260835e-3;  % Earth Eccentricity Square
    % VectorNav VN-200 noise characteristics
    simdata.sigma_a                 = 0.14e-3*9.81*sqrt(simdata.Ts); % VRW
    simdata.sigma_g                 = 0.21*pi/180/60*sqrt(simdata.Ts);% ARW
    simdata.acc_bias_driving_noise  = 4.9e-6*9.81*sqrt(simdata.Ts);   % RRW
    simdata.gyro_bias_driving_noise = 8e-4*pi/180*sqrt(simdata.Ts); % AcRW
    % Navigation initialization
    simdata.altitude                = 17; % Initial altitude [m]
    simdata.latitude                = 33.68*pi/180; % Initial latitude
    simdata.longitude               = -117.83*pi/180; % Initial longitude
    simdata.init_heading            = -0.7*pi/180; % Initial yaw [rad]
    simdata.init_vel                = [0 0 0]'; % Initial velocity [m/s]
    simdata.sigma_initial_acc_bias  = 0e-6*9.81*ones(1,3);  % Accel bias
    simdata.sigma_initial_gyro_bias = 0*pi/180/3600*ones(1,3); % Gyro bias
    simdata.sigma_initial_pos       = 1e-3*ones(1,3); % Position [m]
    simdata.sigma_initial_vel       = 1e-3*ones(1,3); % Velocity [m/s]
    simdata.sigma_initial_att       = 0 * pi/180*ones(1,3); % Attitude
    % ZUPT parameters
    simdata.Window_size = 5; % Window size for ZUPT detector [samples]
    simdata.dv          = [0.0177; 0.0034; 0.0104];%Residual velocity [m/s]
    simdata.sigma_vel   = [1 1 1]*0.02; % Measurement uncertainty [m/s]
```

```
31      simdata.factor      = 2e4; % Threshold for SHOE detector
32      simdata.alpha       = -15e4;  % Bias, Parameters for adaptive threshold
33      simdata.theta       = 2.0;   % Speed at which threshold increases
34      simdata.beta        = 10;   % Priori
35  end
```

C.3.25 SHOE_detector

```
1   function [zupt, LR] =SHOE_detector(u, threshold)
2   % ------------------------------------------------------------------------
3   % This function carries out the SHOE detector
4   % Ref: Skog et al. IEEE Trans. on Biomed Eng, 57 (11), pp. 2657-2666, 2010.
5   % Input: u:     IMU readouts at current time step
6   %        xi:    Uncertainty of estimated velocity
7   %        dt:    Time differnce beween last time ZUPT is on and current step
8   %        ZUPT:  ZUPT state of previous time step
9   %        thre1: Threshold of previous time step
10  %        shock: Maximum shock level of last step
11  % Output: zupt:  Detected ZUPT state. 1 is stance and 0 is swing
12  %         LR:    Likelihood ratio
13  %         thre2: Actual adaptive threshold
14  % ------------------------------------------------------------------------
15      global simdata;
16      [r, c] = size(u);
17      W=simdata.Window_size;
18      sigma2_a = (simdata.sigma_a/simdata.Ts)^2;
19      sigma2_g = (simdata.sigma_g/simdata.Ts)^2;
20      g = 9.796;
21      u_n = mean(u(1:3, :), 2);
22      u_n = u_n / norm(u_n);  % Unit vector along the specific force
```

```
23      for i = 1:3
24          u(i, :) = u(i, :) - g*u_n(i);
25      end
26      total = sum(sum(u(1:3, :).^2))/sigma2_a+sum(sum(u(4:6,:).^2))/sigma2_g;
27      total = total/c;
28      if(total < threshold)
29          zupt = ones(1, simdata.Window_size);
30      else
31          zupt = zeros(1, simdata.Window_size);
32      end
33      LR = total;
34  end
```

C.3.26 skew

```
1  function S = skew(x)
2  % ------------------------------------------------------------------------
3  % This function forms the skew-symmetric matrix of a vector
4  % Input: x: 3*1 vector
5  % Output: S: skew-symmetric matrix
6  % ------------------------------------------------------------------------
7      S = [0      -x(3)   x(2);
8           x(3)      0    -x(1);
9          -x(2)     x(1)    0 ];
10 end
```

C.3.27 theta2quat

```
function q = theta2quat(theta)
% ------------------------------------------------------------------------
% This function converts rotation vector to quaternion
% Ref: Titterton et al. Strapdown inertial navigation technology. (3.53)
% Input: theta: rotation vector in rad, NOT Euler angle!
% Output: q: quaternion in the form of a*i + b*j + c*k + d
% ------------------------------------------------------------------------
    x = theta;
    xNorm = sqrt(x(1,:).^2+x(2,:).^2+x(3,:).^2);
    k=find(xNorm>0);
    rotAxs = zeros(size(x));
    if ¬isempty(k)
        rotAxs(:,k) = [x(1,k)./xNorm(k);x(2,k)./xNorm(k);x(3,k)./xNorm(k)];
    end
    sAnglDiv2 = sin(xNorm/2);
    cAnglDiv2 = cos(xNorm/2);
    q = [ rotAxs(1,:).* sAnglDiv2; rotAxs(2,:).* sAnglDiv2;
          rotAxs(3,:).* sAnglDiv2; cAnglDiv2];
end
```

C.3.28 trueCosine

```
function C = trueCosine(v)
% ------------------------------------------------------------------------
% This function calculates DCM from a rotation vector
% Ref: Rodrigues' Rotation Formula
% Input: v: rotation vector in rad
% Output: C: DCM
% ------------------------------------------------------------------------
```

```
    I3  = eye(3,3);
    norm_v = norm(v);
    if norm_v < 1e-15
        C = I3;
        return
    end
    e_th = v/norm_v;
    cosTh = cos(norm_v);
    sinTh = sin(norm_v);
    C = cosTh*I3 + (1-cosTh)*(e_th*e_th') - sinTh*skew(e_th);
end
```

C.3.29 zero_velocity_detector

```
function zupt =zero_velocity_detector(u, sigma_g, sigma_a, Ts)
% -------------------------------------------------------------------------
% This function carries out the IMU energy detector
% Note: this detector only considers the varince of accel ang gyro readouts
% Input: u:       IMU readouts at current time step
%        sigma_a: Accel noise level
%        sigma_g: Gyro noise level
%        Ts:      Time step length
% Output: zupt:  Detected ZUPT state. 1 is stance and 0 is swing
% -------------------------------------------------------------------------
    W=3;
    sigma2_a = (sigma_a/Ts)^2;
    sigma2_g = (sigma_g/Ts)^2;
    tmp = var(u, 0, 2);
    total = 0;
    for i = 1:3
```

```
17          total = total + tmp(i)/sigma2_g;
18          total = total + tmp(i+3)/sigma2_a;
19      end
20      total = total/6;
21      if(total < 2)
22          zupt = ones(1, W);
23      else
24          zupt = zeros(1, W);
25      end
26  end
```

C.4 MATALB GUI for Demonstration

A MATLAB-based GUI was developed to demonstrate the effects of IMU errors on the navigation accuracy. The adjustable parameters include: ARW, VRW, RRW, IMU sampling rate, length of the trajectory, and ZUPT switch. The majority of the code is adapted from Section C.1, but revised to work in the MATLAB App environment. A screenshot of the developed GUI is presented in Fig. C.1. The original MATLAB code files and GUI file can be found in `\\nitride.eng.uci.edu\NIST\Code_Yusheng\Matlab_demo`.

```
1  classdef Demo_code < matlab.apps.AppBase
2
3      % Properties that correspond to app components
4      properties (Access = public)
5          UIFigure                        matlab.ui.Figure
6          UIAxes                          matlab.ui.control.UIAxes
7          ARWdegrthLabel                  matlab.ui.control.Label
8          ARW                             matlab.ui.control.Knob
9          CalculateButton                 matlab.ui.control.Button
```

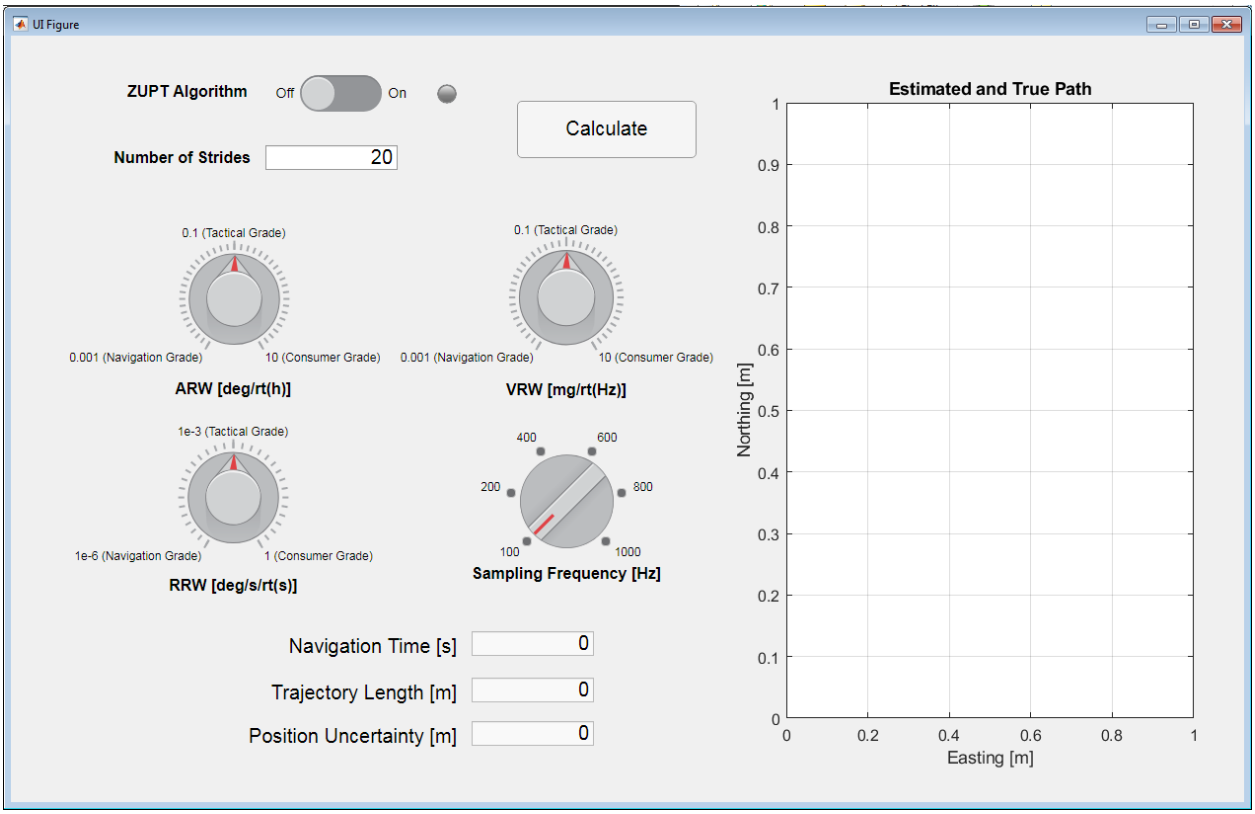

Figure C.1: Screenshot of the developed GUI.

```
10          NavigationTimesEditFieldLabel  matlab.ui.control.Label
11          NavigationTimes                matlab.ui.control.NumericEditField
12          VRWmgrtHzLabel                 matlab.ui.control.Label
13          VRW                            matlab.ui.control.Knob
14          SwitchLabel                    matlab.ui.control.Label
15          Switch                         matlab.ui.control.Switch
16          ZUPTLabel                      matlab.ui.control.Label
17          RRWdegsrtsLabel                matlab.ui.control.Label
18          RRW                            matlab.ui.control.Knob
19          SamplingFrequencyHzKnobLabel   matlab.ui.control.Label
20          SamplingFrequency              matlab.ui.control.DiscreteKnob
21          PositionUncertaintymLabel      matlab.ui.control.Label
22          Positionuncertainty            matlab.ui.control.NumericEditField
23          TrajectoryLengthmEditFieldLabel  matlab.ui.control.Label
24          TrajectoryLength               matlab.ui.control.NumericEditField
25          NumberofStridesLabel           matlab.ui.control.Label
26          NumberofSteps                  matlab.ui.control.NumericEditField
27          Lamp                           matlab.ui.control.Lamp
28      end
29
30      methods (Access = private)
31          function Calculate(app)
32              currDir = pwd;
33              addpath([currDir, '\lib']);
34              s=settings();
35              d2r = pi/180;
36              r2d = 180/pi;
37              M = app.NumberofSteps.Value; % The number of total steps
38              Time = 1.071;  % Time period
39              N = str2num(app.SamplingFrequency.Value);
40              Ts = Time/N;  % Time step
41              a = 6378137;
42              latitude=33.68*pi/180;
```

```
43          longitude=-117.83*pi/180;
44          sigma_a=10^(app.VRW.Value) *9.81e-3*sqrt(Ts); % m/s/sqrt(s)
45          sigma_g=10^(app.ARW.Value) *pi/180/60*sqrt(Ts);   % deg/sqrt(h)
46          sigma_RRW=10^(app.RRW.Value)*pi/180;      %[m/s]
47          sigma_vel=[1 1 1]*0.02;      %[m/s]
48          % -----------------  Bio-mechanial model  -----------------
49          Hip_o = [24 23 22 16 9 4 -2 -7 -11 -15 -16 -13 ...
50              -6 2 13 19 23 26 27 25 24];
51          Knee_o = [3 11 15 18 15 12 8 6 5 8 14 28 47 63 ...
52              68 65 53 37 15 1 3];
53          Ankle_o = [-3 -13 -8 -3 1 3 6 7 8 9 4 -5 -18 -20 ...
54
55          T = linspace(0, Time, N+1);
56          T = T(1:N);
57          x = linspace(-14, 36, 51);
58          x_in = linspace(1, 20.9, N);
59          y1 = [Hip_o(1, 6:20) Hip_o Hip_o(2:16)];
60          y2 = [Knee_o(1, 6:20) Knee_o Knee_o(2:16)];
61          y3 = [Ankle_o(1, 6:20) Ankle_o Ankle_o(2:16)];
62          k = 40;
63          p1 = polyfit(x, y1, k);
64          p2 = polyfit(x, y2, k);
65          p3 = polyfit(x, y3, k);
66          Hip = polyval(p1, x_in);
67          Knee = polyval(p2, x_in);
68          Ankle = polyval(p3, x_in);
69          angle_foot = Hip - Knee + Ankle;
70          % Angle [percentage velocity acceleration]
71          w_i = [0 -290 -2523.4; 0.02 -293.5 0; 0.1 0 0; 0.3 0 0; ...
72              0.385 -72.2 0; 0.415 -72.2 0; 0.56 -463.6 0; 0.74 409.8 ...
73              0; 0.81 327.1 0; 0.885 417.8+20 0; 1 -280 -2523.4];
74          p_w = zeros(length(w_i)-1, 4);
75          angular_velocity_i = zeros(1, N);
```

```
76              for i = 1:length(p_w)
77                  p_w(i, :) = cubic_interpolation(w_i(i, :), w_i(i+1, :));
78                  temp_l = [w_i(i, 1):(1/N):w_i(i+1, 1)];
79                  angular_velocity_i(w_i(i, 1)*N+1:w_i(i+1, 1)*N+1) =...
80                      polyval(p_w(i, :), temp_l);
81              end
82              angle_foot_i = cumsum(angular_velocity_i(1:N))*Ts;
83              angle_foot_i = angle_foot_i - angle_foot_i(1) + angle_foot(1);
84              % vx direction  [percentage velocity acceleration]
85              x = [0 0.4 -11.1; 0.125 0 0; 0.275 0 0; 0.395 0.0623 0;...
86                  0.49 0.2 0; 0.54 7.5e-3 0; 0.7 4.292 0; 0.775 3.63 0;...
87                  0.865 4.988 0; 1 0.4 -11.1];
88              % vy direction  [percentage velocity acceleration]
89              y = [0 -0.564 -5.98; 0.02 -0.759 0; 0.125 0 0; 0.3 0 0;...
90                  0.4 -9.6e-3 -0.084; 0.5 -0.1 0; 0.615 1-0.2 0; 0.745 ...
91                  -0.273 0; 0.79 -0.198 0; 0.905 0.827 0; 1 -0.564 -5.98];
92              p_x = zeros(length(x)-1, 4);
93              p_y = zeros(length(y)-1, 4);
94              x_f_d_i = zeros(N, 1);
95              y_f_d_i = zeros(N, 1);
96              for i = 1:length(p_x)
97                  p_x(i, :) = cubic_interpolation(x(i, :), x(i+1, :));
98                  temp = [x(i, 1):(1/N):x(i+1, 1)];
99                  x_f_d_i(x(i, 1)*N+1:x(i+1, 1)*N+1)=polyval(p_x(i, :),temp);
100             end
101             for i = 1:length(p_y)
102                 p_y(i, :) = cubic_interpolation(y(i, :), y(i+1, :));
103                 temp = [y(i, 1):(1/N):y(i+1, 1)];
104                 y_f_d_i(y(i, 1)*N+1:y(i+1, 1)*N+1)=polyval(p_y(i, :),temp);
105             end
106             x_f_i = cumsum(x_f_d_i(1:N))*Ts;
107             y_f_i = cumsum(y_f_d_i(1:N))*Ts;
108             x_f_dd_i = diff(x_f_d_i)/Ts;  % 200*1
```

```
109             y_f_dd_i = diff(y_f_d_i)/Ts;  % 200*1
110             trajectory = [x_f_i zeros(N, 1) y_f_i zeros(N, 1) ...
111                 angle_foot_i'*pi/180 zeros(N, 1)]';
112             temp = trajectory;
113             x_f_d_i = x_f_d_i(1:N);
114             y_f_d_i = y_f_d_i(1:N);
115             v_x = x_f_d_i;
116             v_y = y_f_d_i;
117             for i = 1:M-1
118                 trajectory = [trajectory temp];
119                 x_f_d_i = [x_f_d_i; v_x];
120                 y_f_d_i = [y_f_d_i; v_y];
121             end
122             true.Pn     = trajectory(1:3,:)*1;
123             true.rpyDeg = trajectory(4:6,:)*180/pi*1;
124             true.dt     = Ts;
125             true.t      = ((1:size(true.Pn,2))-1)*true.dt;
126             true.q_b2n  = Imu2YPRdeg_to_qb2n(true.rpyDeg(1,:),...
127                 true.rpyDeg(2,:),true.rpyDeg(3,:));
128             true.h  = true.Pn(3,:);
129             true.Vn = [x_f_d_i'; zeros(1, N*M); -y_f_d_i'];
130             lonDeg =  longitude *180/pi;
131             latDeg =  latitude *180/pi;
132             q_e2N0 =  lonLatDegTo_q_e2N(lonDeg, latDeg);
133             true.q_e2N=integVel_into_q_e2N(q_e2N0,true.h,true.Vn,true.dt);
134             [true.lonDeg, true.latDeg] = q_e2N_toLonLatDeg(true.q_e2N);
135             true.LLA(1,:) = true.lonDeg*d2r;
136             true.LLA(2,:) = true.latDeg*d2r;
137             true.LLA(3,:) = true.h;
138             true.Northing = (true.LLA(2,:)-true.LLA(2,1))*a;
139             true.Easting  = (true.LLA(1,:)-true.LLA(1,1)).* ...
140                 cos(true.LLA(2,:))*a;
141             true.Down     =  -(true.LLA(3,:)-true.LLA(3,1));
```

```
142             % -----------------  Calculate IMU readouts  -----------------
143             dth = quatToAngl(q_mult(true.q_b2n(:,1:end-1), ...
144                 q_inv(true.q_b2n(:,2:end))));
145             dth = [dth(:,1), dth];
146             e2  = 6.694380004260835e-3;  % EarthEccentricitySq
147             r_wrtE_e = LLA2rWrtEinE(true.LLA);
148             r_e_n    = quatRot(true.q_e2N,r_wrtE_e);
149             w_e2i_n  = zeros(3,length(true.t));
150             g_n      = zeros(3,length(true.t));
151             dg_centropital = zeros(3,length(true.t));
152             for i=1:length(true.t)
153                 w_e2i_n(:,i)=earthRateInBody(0,0,0, true.LLA(2,i)*180/pi);
154                 g_n(:,i)=gravityModel(true.LLA(2,i),norm(r_e_n(:,i)),a,e2);
155                 dg_centropital(:,i) = cross(w_e2i_n(:,i), ...
156                     cross(w_e2i_n(:,i), r_e_n(:,i)));
157             end
158             gl_n  = g_n - dg_centropital;
159             w_n2e_n(1,:) =  true.Vn(2,:)./a;
160             w_n2e_n(2,:) = -true.Vn(1,:)./a;
161             w_n2e_n(3,:) =  w_n2e_n(2,:)*0;
162             w_aux = 2 * w_e2i_n  +  w_n2e_n;
163             aux1 = zeros(3,length(true.t));
164             for i=1:length(true.t)
165                 aux1(:,i) = cross(w_aux(:,i), true.Vn(:,i));
166             end
167             aux2 = aux1 - gl_n;
168             dvn = [zeros(3,1),diff(true.Vn,1,2)];
169             f_n = dvn/true.dt + aux2;
170             f_b = quatRot(q_inv(true.q_b2n),f_n);
171             dv  = f_b*true.dt;
172             w_n2i_n = w_n2e_n + w_e2i_n;
173             w_n2i_b = quatRot(q_inv(true.q_b2n),w_n2i_n);
174             dth = dth + w_n2i_b*true.dt;
```

```
175             % -----------------  Calculate ground truth  -----------------
176             input.q_b2n          = true.q_b2n(:,1);
177             input.q_e2n          = true.q_e2N(:,1);
178             input.LLA            = true.LLA(:,1);
179             input.v_nWrtE_n      = true.Vn(:,1);
180             input.Δ_b_Prev   = zeros(3,1);
181             input.Δ_n_Prev   = zeros(3,1);
182             input.Δ_n2e_Prev = zeros(3,1);
183             %
184             pred.q_b2n(:,1) = input.q_b2n;
185             pred.q_e2n(:,1) = input.q_e2n;
186             pred.LLA(:,1)   = input.LLA;
187             pred.v_nWrtE_n(:,1)    = input.v_nWrtE_n;
188             pred.t          = true.t;
189             sensor.dt       = true.dt;
190             for i=2:length(pred.t)
191                 sensor.w_b2i_b = dth(:,i)/sensor.dt;
192                 sensor.f_b     = dv(:,i)/sensor.dt;
193                 input = navSLN_ZUPT(sensor, input);
194                 pred.q_b2n(:,i)     = input.q_b2n;
195                 pred.q_e2n(:,i)     = input.q_e2n;
196                 pred.LLA(:,i)       = input.LLA;
197                 pred.v_nWrtE_n(:,i) = input.v_nWrtE_n;
198             end
199             pred.Northing =  (pred.LLA(2,:)-pred.LLA(2,1))*a;
200             pred.Easting  =  (pred.LLA(1,:)-pred.LLA(1,1)).* ...
201                 cos(pred.LLA(2,:))*a;
202             pred.Down     = -(pred.LLA(3,:)-pred.LLA(3,1));
203             [roll_deg, pitch_deg, yaw_deg] = qb2nImu2YPRdeg(pred.q_b2n);
204             pred.rpyDeg = [roll_deg; pitch_deg; yaw_deg];
205             % -----------------  Add sensor noise  -----------------
206             gyro.sigma_AWN = 2e-4*d2r*0;               % rad
207             gyro.sigma_ARW = sigma_g/sqrt(Ts);           % rad/sqrt(s)
```

```
208             gyro.sigma_RRW = sigma_RRW;   %  (rad/s)/sqrt(s)
209             gyro.Bias      = 0;          % rad/s
210             accl.sigma_VWN   = 4.5e-4*0;    % m/s
211             accl.sigma_VRW   = sigma_a/sqrt(Ts) * 1;        % m/s/sqrt(s)
212             accl.sigma_AcRW  = 0;         % m/s^2/sqrt(s)
213             accl.Bias        = 0;  % m/s^2
214             N = length(pred.t);
215             gyro.dth_error = randn(3,N)*gyro.sigma_ARW*sqrt(true.dt) + ...
216                              randn(3,N)*gyro.sigma_AWN + ...
217                              gyro.Bias*true.dt + ...
218                              cumsum(randn(3,N),2)*gyro.sigma_RRW ...
219                              *sqrt(true.dt)*true.dt;
220             accl.dv_error = randn(3,N)*accl.sigma_VRW*sqrt(true.dt) + ...
221                             randn(3,N)*accl.sigma_VWN + ...
222                             accl.Bias*true.dt +...
223                             cumsum(randn(3,N),2)* ...
224                             accl.sigma_AcRW*sqrt(true.dt)*true.dt;
225             meas.dth = dth + gyro.dth_error;
226             meas.dv  = dv  + accl.dv_error;
227             u = [meas.dth; meas.dv]./sensor.dt;
228             % -----------------  Perform navigation  -----------------
229             input.q_b2n          = pred.q_b2n(:,1);
230             input.q_e2n          = pred.q_e2n(:,1);
231             input.LLA            = pred.LLA(:,1);
232             input.v_nWrtE_n      = pred.v_nWrtE_n(:,1);
233             input.Δ_b_Prev   = zeros(3,1);
234             input.Δ_n_Prev   = zeros(3,1);
235             input.Δ_n2e_Prev = zeros(3,1);
236             sensor.dt = true.dt;
237             est.t = true.t;
238             n = length(est.t);
239             est.q_b2n     = [input.q_b2n(:,1), zeros(4,n-1)];
240             est.q_e2n     = [input.q_e2n(:,1), zeros(4,n-1)];
```

```
241             est.LLA       = [input.LLA(:,1),   zeros(3,n-1)];
242             est.v_nWrtE_n = [input.v_nWrtE_n(:,1),    zeros(3,n-1)];
243             e3 = [1,1,1];
244             Q_diag = [gyro.sigma_ARW^2*e3, accl.sigma_VRW^2*e3, e3*0, ...
245                       gyro.sigma_RRW^2*e3, accl.sigma_AcRW^2*e3]*true.dt;
246             x = zeros(1, 15);
247             P = diag(x.^2);
248             aB = zeros(3,1);
249             gB = zeros(3,1);
250             Id = eye(size(P));
251             zupt = zeros(1,n);  % ZUPTing marker
252             T = zeros(1,n);   % Test statistics of ZUPTing detector
253             W = 3;  % ZUPTing window size
254             O33 = zeros(3,3);
255             I33 = eye(3,3);
256             A11 = -skew(w_e2i_n(:,1));
257             H = zeros(3,size(P,1));
258             H(:,4:6) = eye(3,3);
259             %
260             R = diag(sigma_vel.^2);
261             sensor.dt       = true.dt;
262             for i=2:length(est.t)
263                 sensor.w_b2i_b = meas.dth(:,i)/sensor.dt - gB;
264                 sensor.f_b     = meas.dv(:,i)/sensor.dt  - aB;
265                 if i < n - W + 2
266                     zupt(i:i+W-1)=zero_velocity_detector(u(:, i:i+W-1), ...
267                         sigma_g, sigma_a, Ts);
268                 end
269                 input = navSLN_ZUPT(sensor, input);
270                 A14 = -quat2dcos(input.q_b2n);
271                 A21 =  skew(-A14*sensor.f_b);
272                 A   = [A11   O33 O33 A14    O33;...
273                        A21   O33 O33 O33   -A14;...
```

```
274                     O33   I33 O33 O33    O33;...
275                     O33   O33 O33 O33    O33;...
276                     O33   O33 O33 O33    O33];
277                 F = expm(A*sensor.dt);
278                 P = F*P*F' + diag(Q_diag);
279                 dx = zeros(15, 1);
280                 if strcmp(app.Switch.Value, 'On')
281                     zupt_switch = 1;
282                 elseif strcmp(app.Switch.Value, 'Off')
283                     zupt_switch = 10;
284                 end
285                     if (zupt(i) == zupt_switch)
286                         z = input.v_nWrtE_n - pred.v_nWrtE_n(:,i);
287                         K  = (P*H')/(H*P*H'+R);
288                         P  = (Id-K*H)*P;
289                         dx = K*z;
290                     end
291                     gB = gB + dx(10:12);
292                     aB = aB + dx(13:15);
293                     input.LLA(1)     = input.LLA(1) - dx(8)/a;
294                     input.LLA(2)     = input.LLA(2) - dx(7)/a;
295                     input.LLA(3)     = input.LLA(3) - dx(9);
296                     input.q_e2n   = lonLatDegTo_q_e2N(input.LLA(1)*r2d, ...
297                         input.LLA(2)*r2d);
298                     input.v_nWrtE_n  = input.v_nWrtE_n - dx(4:6);
299                     input.q_b2n      = q_mult(input.q_b2n, ...
300                         theta2quat(-dx(1:3)));
301                     P  = (P+P')/2;
302                     est.q_b2n(:,i)     = input.q_b2n;
303                     est.q_e2n(:,i)     = input.q_e2n;
304                     est.LLA(:,i)       = input.LLA;
305                     est.v_nWrtE_n(:,i) = input.v_nWrtE_n;
306                     kf.dx(:,i)     = dx;
```

```
307                     kf.gB(:,i)     = gB;
308                     kf.aB(:,i)     = aB;
309                     kf.diagP(:,i)  = diag(P);
310             end
311             est.Northing =  (est.LLA(2,:)-pred.LLA(2,1))*a;
312             est.Easting  =  (est.LLA(1,:)-pred.LLA(1,1)).* ...
313                 cos(pred.LLA(2,:))*a;
314             est.Down     = -(est.LLA(3,:)-pred.LLA(3,1));
315             %
316             [roll_deg, pitch_deg, yaw_deg] = qb2nImu2YPRdeg(est.q_b2n);
317             est.rpyDeg = [roll_deg; pitch_deg; yaw_deg];
318             %
319             est.rpyDeg(1,:) = wrapTo180(est.rpyDeg(1,:));
320             est.rpyDeg(2,:) = wrapTo180(est.rpyDeg(2,:));
321             est.rpyDeg(3,:) = wrapTo180(est.rpyDeg(3,:));
322             %
323             pred.rpyDeg(1,:) = wrapTo180(pred.rpyDeg(1,:));
324             pred.rpyDeg(2,:) = wrapTo180(pred.rpyDeg(2,:));
325             pred.rpyDeg(3,:) = wrapTo180(pred.rpyDeg(3,:));
326             if (1)
327                 plot( app.UIAxes, est.Easting,  est.Northing, 'b', ...
328                     pred.Easting, pred.Northing,'r--');
329                 legend( app.UIAxes, 'Estimated path', 'True path');
330             end
331             app.TrajectoryLength.Value = pred.Northing(end);
332             app.NavigationTimes.Value = app.NumberofSteps.Value * 1.071;
333             app.Positionuncertainty.Value = sqrt(kf.diagP(7, end)+ ...
334                 kf.diagP(8, end))*4;
335         end
336     end
337
338     % Callbacks that handle component events
339     methods (Access = private)
```

```
340
341         % Button pushed function: CalculateButton
342         function CalculateButtonPushed(app, event)
343             Calculate(app);
344         end
345
346         % Value changed function: Switch
347         function SwitchValueChanged(app, event)
348             if strcmp(app.Switch.Value, 'On')
349                 app.Lamp.Color = [0 1 0];
350             elseif strcmp(app.Switch.Value, 'Off')
351                 app.Lamp.Color = [0.5 0.5 0.5];
352             end
353         end
354     end
355
356     % Component initialization
357     methods (Access = private)
358
359         % Create UIFigure and components
360         function createComponents(app)
361
362             % Create UIFigure and hide until all components are created
363             app.UIFigure = uifigure('Visible', 'off');
364             app.UIFigure.Position = [100 100 1250 776];
365             app.UIFigure.Name = 'UI Figure';
366
367             % Create UIAxes
368             app.UIAxes = uiaxes(app.UIFigure);
369             title(app.UIAxes, 'Estimated and True Path')
370             xlabel(app.UIAxes, 'Easting [m]')
371             ylabel(app.UIAxes, 'Northing [m]')
372             app.UIAxes.FontSize = 16;
```

```
373             app.UIAxes.GridAlpha = 0.15;
374             app.UIAxes.MinorGridAlpha = 0.25;
375             app.UIAxes.Box = 'on';
376             app.UIAxes.XGrid = 'on';
377             app.UIAxes.YGrid = 'on';
378             app.UIAxes.Position = [732 36 479 701];
379
380             % Create ARWdegrthLabel
381             app.ARWdegrthLabel = uilabel(app.UIFigure);
382             app.ARWdegrthLabel.HorizontalAlignment = 'center';
383             app.ARWdegrthLabel.VerticalAlignment = 'top';
384             app.ARWdegrthLabel.FontSize = 16;
385             app.ARWdegrthLabel.FontWeight = 'bold';
386             app.ARWdegrthLabel.Position = [155.5 407 142 22];
387             app.ARWdegrthLabel.Text = 'ARW [deg/rt(h)]';
388
389             % Create ARW
390             app.ARW = uiknob(app.UIFigure, 'continuous');
391             app.ARW.Limits = [-3 1];
392             app.ARW.MajorTicks = [-3 -1 1];
393             app.ARW.MajorTickLabels = {'0.001 (Navigation Grade)', ...
394                 '0.1 (Tactical Grade)', '10 (Consumer Grade)'};
395             app.ARW.Position = [181 464 93 93];
396             app.ARW.Value = -1;
397
398             % Create CalculateButton
399             app.CalculateButton = uibutton(app.UIFigure, 'push');
400             app.CalculateButton.ButtonPushedFcn = ...
401                 createCallbackFcn(app, @CalculateButtonPushed, true);
402             app.CalculateButton.FontSize = 20;
403             app.CalculateButton.Position = [514 653 182 58];
404             app.CalculateButton.Text = 'Calculate';
405
```

```
406             % Create NavigationTimesEditFieldLabel
407             app.NavigationTimesEditFieldLabel = uilabel(app.UIFigure);
408             app.NavigationTimesEditFieldLabel.HorizontalAlignment ='right';
409             app.NavigationTimesEditFieldLabel.VerticalAlignment = 'top';
410             app.NavigationTimesEditFieldLabel.FontSize = 20;
411             app.NavigationTimesEditFieldLabel.Position = [276 145 177 26];
412             app.NavigationTimesEditFieldLabel.Text = 'Navigation Time [s]';
413
414             % Create NavigationTimes
415             app.NavigationTimes = uieditfield(app.UIFigure, 'numeric');
416             app.NavigationTimes.Editable = 'off';
417             app.NavigationTimes.FontSize = 20;
418             app.NavigationTimes.Position = [468 149 124 25];
419
420             % Create VRWmgrtHzLabel
421             app.VRWmgrtHzLabel = uilabel(app.UIFigure);
422             app.VRWmgrtHzLabel.HorizontalAlignment = 'center';
423             app.VRWmgrtHzLabel.VerticalAlignment = 'top';
424             app.VRWmgrtHzLabel.FontSize = 16;
425             app.VRWmgrtHzLabel.FontWeight = 'bold';
426             app.VRWmgrtHzLabel.Position = [490.5 406 147 22];
427             app.VRWmgrtHzLabel.Text = 'VRW [mg/rt(Hz)]';
428
429             % Create VRW
430             app.VRW = uiknob(app.UIFigure, 'continuous');
431             app.VRW.Limits = [-3 1];
432             app.VRW.MajorTicks = [-3 -1 1];
433             app.VRW.MajorTickLabels = {'0.001 (Navigation Grade)', ...
434                 '0.1 (Tactical Grade)', '10 (Consumer Grade)'};
435             app.VRW.Position = [516 464 96 96];
436             app.VRW.Value = -1;
437
438             % Create SwitchLabel
```

```
439             app.SwitchLabel = uilabel(app.UIFigure);
440             app.SwitchLabel.HorizontalAlignment = 'center';
441             app.SwitchLabel.VerticalAlignment = 'top';
442             app.SwitchLabel.FontSize = 16;
443             app.SwitchLabel.FontWeight = 'bold';
444             app.SwitchLabel.Position = [324 665 25 20];
445             app.SwitchLabel.Text = '';
446 
447             % Create Switch
448             app.Switch = uiswitch(app.UIFigure, 'slider');
449             app.Switch.ValueChangedFcn = createCallbackFcn(app, ...
450                 @SwitchValueChanged, true);
451             app.Switch.FontSize = 14;
452             app.Switch.Position = [294 700 85 37];
453 
454             % Create ZUPTLabel
455             app.ZUPTLabel = uilabel(app.UIFigure);
456             app.ZUPTLabel.VerticalAlignment = 'top';
457             app.ZUPTLabel.FontSize = 16;
458             app.ZUPTLabel.FontWeight = 'bold';
459             app.ZUPTLabel.Position = [119 710 127 20];
460             app.ZUPTLabel.Text = 'ZUPT Algorithm';
461 
462             % Create RRWdegsrtsLabel
463             app.RRWdegsrtsLabel = uilabel(app.UIFigure);
464             app.RRWdegsrtsLabel.HorizontalAlignment = 'center';
465             app.RRWdegsrtsLabel.VerticalAlignment = 'top';
466             app.RRWdegsrtsLabel.FontSize = 16;
467             app.RRWdegsrtsLabel.FontWeight = 'bold';
468             app.RRWdegsrtsLabel.Position = [125.5 208 202 22];
469             app.RRWdegsrtsLabel.Text = 'RRW [deg/s/rt(s)]';
470 
471             % Create RRW
```

```
472             app.RRW = uiknob(app.UIFigure, 'continuous');
473             app.RRW.Limits = [-6 0];
474             app.RRW.MajorTicks = [-6 -3 0];
475             app.RRW.MajorTickLabels = {'1e-6 (Navigation Grade)', ...
476                 '1e-3 (Tactical Grade)', '1 (Consumer Grade)'};
477             app.RRW.Position = [180 263 93 93];
478             app.RRW.Value = -3;
479 
480             % Create SamplingFrequencyHzKnobLabel
481             app.SamplingFrequencyHzKnobLabel = uilabel(app.UIFigure);
482             app.SamplingFrequencyHzKnobLabel.HorizontalAlignment ='center';
483             app.SamplingFrequencyHzKnobLabel.VerticalAlignment = 'top';
484             app.SamplingFrequencyHzKnobLabel.FontSize = 16;
485             app.SamplingFrequencyHzKnobLabel.FontWeight = 'bold';
486             app.SamplingFrequencyHzKnobLabel.Position = [464.5 222 199 20];
487             app.SamplingFrequencyHzKnobLabel.Text = ...
488                 'Sampling Frequency [Hz]';
489 
490             % Create SamplingFrequency
491             app.SamplingFrequency = uiknob(app.UIFigure, 'discrete');
492             app.SamplingFrequency.Items = {'100', '200', '400',...
493                 '600', '800', '1000'};
494             app.SamplingFrequency.Position = [517 258 95 95];
495             app.SamplingFrequency.Value = '100';
496 
497             % Create PositionUncertaintymLabel
498             app.PositionUncertaintymLabel = uilabel(app.UIFigure);
499             app.PositionUncertaintymLabel.HorizontalAlignment = 'right';
500             app.PositionUncertaintymLabel.VerticalAlignment = 'top';
501             app.PositionUncertaintymLabel.FontSize = 20;
502             app.PositionUncertaintymLabel.Position = [235 54 218 26];
503             app.PositionUncertaintymLabel.Text ='Position Uncertainty [m]';
504 
```

```
505             % Create Positionuncertainty
506             app.Positionuncertainty = uieditfield(app.UIFigure, 'numeric');
507             app.Positionuncertainty.Editable = 'off';
508             app.Positionuncertainty.FontSize = 20;
509             app.Positionuncertainty.Position = [468 58 124 25];
510
511             % Create TrajectoryLengthmEditFieldLabel
512             app.TrajectoryLengthmEditFieldLabel = uilabel(app.UIFigure);
513             app.TrajectoryLengthmEditFieldLabel.HorizontalAlignment ...
514                 = 'right';
515             app.TrajectoryLengthmEditFieldLabel.VerticalAlignment = 'top';
516             app.TrajectoryLengthmEditFieldLabel.FontSize = 20;
517             app.TrajectoryLengthmEditFieldLabel.Position = [258 98 195 26];
518             app.TrajectoryLengthmEditFieldLabel.Text ...
519                 = 'Trajectory Length [m]';
520
521             % Create TrajectoryLength
522             app.TrajectoryLength = uieditfield(app.UIFigure, 'numeric');
523             app.TrajectoryLength.Editable = 'off';
524             app.TrajectoryLength.FontSize = 20;
525             app.TrajectoryLength.Position = [468 102 124 25];
526
527             % Create NumberofStridesLabel
528             app.NumberofStridesLabel = uilabel(app.UIFigure);
529             app.NumberofStridesLabel.HorizontalAlignment = 'right';
530             app.NumberofStridesLabel.VerticalAlignment = 'top';
531             app.NumberofStridesLabel.FontSize = 16;
532             app.NumberofStridesLabel.FontWeight = 'bold';
533             app.NumberofStridesLabel.Position = [100 643 144 20];
534             app.NumberofStridesLabel.Text = 'Number of Strides';
535
536             % Create NumberofSteps
537             app.NumberofSteps = uieditfield(app.UIFigure, 'numeric');
```

```
538             app.NumberofSteps.Limits = [1 500];
539             app.NumberofSteps.FontSize = 20;
540             app.NumberofSteps.Position = [259 641 134 25];
541             app.NumberofSteps.Value = 20;
542 
543             % Create Lamp
544             app.Lamp = uilamp(app.UIFigure);
545             app.Lamp.Position = [433 708 20 20];
546             app.Lamp.Color = [0.502 0.502 0.502];
547 
548             % Show the figure after all components are created
549             app.UIFigure.Visible = 'on';
550         end
551     end
552 
553     % App creation and deletion
554     methods (Access = public)
555 
556         % Construct app
557         function app = Demo_2
558 
559             % Create UIFigure and components
560             createComponents(app)
561 
562             % Register the app with App Designer
563             registerApp(app, app.UIFigure)
564 
565             if nargout == 0
566                 clear app
567             end
568         end
569 
570         % Code that executes before app deletion
```

```
571             function delete(app)
572
573                 % Delete UIFigure when app is deleted
574                 delete(app.UIFigure)
575             end
576         end
577     end
```

Appendix D

List of Vendors

Allied High Tech Products, Inc.

Location: Rancho Dominguez, CA, USA

Phone: 800-675-1118

Email: info@alliedhightech.com

The release of the Fused Quartz shell resonators, and the directional lapping procedure require the Multiprep Polishing System from Allied High Tech Products. All lapping related products, such as lapping films, slurry, lubricants, and lapping fixtures, are available from this vendor. Maintenance and repair of the Multiprep Polishing System are also available.

American Cleanstat

Location: Irvine, CA, USA

Phone: 714-533-2827

Foam Tipped Swabs for shell cleaning during lapping are available from American Cleanstat.

Angstrom Engineering, Inc.

Location: Southern Ontario, Canada

Phone: 800-695-8270

Email: sales@angstromengineering.com

Angstrom Engineering, Inc. specializes in material deposition equipment and supplies, including Physical Vapor Deposition (PVD), Chemical Vapor Deposition (CVD), and evaporation. Relative supplies for Angstrom Engineering EvoVac Glovebox Evaporator at Irvine Materials Research Institute (IMRI), including crucibles, metal targets, are available from Angstrom Engineering, Inc.

Boot Barn

Location: Lake Forest, CA, USA

Phone: 949-455-0211

Steel-toe boots for foot-mounted navigation systems are available from Boot Barn.

Coining Inc.

Location: Montvale, NJ, USA

Phone: 201-791-4020

Email: coining.sales@ametek.com

Coining, Inc. provides preforms for Au-Sn eutectic bonding.

Electro Tech Machining

Location: Long Beach, CA, USA

Phone: 562-436-9281

Email: etm@etm-lb.com

Electro Tech Machining manufactures customized graphite parts. Possible applications include die-attachment, shell assembly, vacuum packaging, and shadow mask for deposition.

Front Range Photomask Co. LLC

Location: Lake Havasu City, AZ, USA

Phone: 928-733-6217

Email: masks@frontrange-photomask.com

Front Range Photomask Co. LLC provides photomask fabrication service with accuracy ranging from 0.2μm to 1μm. File conversion service from DXF file to PDF file is available.

Keysight Technologies, Inc.

Location: Santa Rosa, CA, USA

Phone: 800-829-4444

Email: usa_orders@keysight.com

AFM probes are purchased from Keysight Technologies, Inc.

L.P. Glassblowing, Inc.

Location: Santa Clara, CA, USA

Phone: 408-988-7561

Email: sales@lpglassblowing.com

L.P. Glassblowing, Inc. specializes in high-precision quartzware and glass Products. Welding, polishing, machining and repair of quartzware are available.

Mark Optics, Inc.

Location: Santa Ana, CA, USA

Phone: 714-545-6684

Email: quote@markoptics.com

Mark Optics, Inc. is a custom optics provider that specializes in glass wafers. Available materials include Pyrex, Fused Quartz, Borosilicate Glass (BSG), and Ultra Low Expansion Titania Silicate Glass (ULE TSG). Wafer polishing service for etched wafers is also available at Mark Optics, Inc.

Marvell Nanofabrication Laboratory

Location: Berkeley, CA, USA

Phone: 510-809-8600

Email: rhemp@berkeley.edu

Marvell Nanofabrication Laboratory at UC Berkeley is a cleanroom facility providing microfabrication services. Specifically, double-side 2μm doped Poly-Si LPCVD is necessary as the hard mask during HF etching of the Fused Quartz wafers. The thickness of the wafers can be as high as 2mm.

MTI Corporation

Location: Richmond, CA, USA

Phone: 510-525-3070

Email: info@mtixtl.com

The high-temperature glassblowing furnace is from MTI Corporation. Repair and accessories, such as ceramic vacuum tubes, rods, crucibles, heating elements, fuses, are available.

NPOS Technologies, Inc.

Location: Pasadena, CA, USA

Phone: 626-471-1207

Email: info@npos-usa.com

NPOS Technologies, Inc. offers precision semiconductor assembly equipments for packaging

applications. Purchase, maintenance, and repair of UniTemp RSS-450-110 Vacuum Oven for die-attachemnt and high-temperature bake-out are available from NPOS Technologies, Inc.

Pfeiffer Vacuum, Inc.

Location: Nashua, NH, USA

Phone: 800-248-8254

Email: contact@pfeiffer-vacuum.com

Pfeiffer Vacuum, Inc. specializes in vacuum pumps and related products. Purchase and regular maintenance of the pumps are available.

PI (Physik Instrumente) L.P.

Location: Irvine, CA, USA

Phone: 949-679-9191

Email: info@pi-usa.us

Piezoelectric stacks for piezoelectric actuation of devices are purchased from PI (Physik Instrumente) L.P.

Polhemus

Location: Colchester, VT, USA

Phone: 800-357-4777

Email: sales@polhemus.com

Polhemus specializes in magnetic motion tracking technology. Characterization of the motion of foot requires PATRIOT 6DOF motion tracker by Polhemus.

Sigma-Aldrich Corp.

Location: St. Louis, MO, USA

Phone: 800-521-8956

Email: cssorders@sial.com

Potassium Hydroxide for Poly-Si hard mask removal is available from Sigma-Aldrich Corp.

Spectrum Micromechanical, Inc.

Location: La Jolla, CA, USA

Phone: 858-395-2264

Email: jeff@spmmi.com

Wafer dicing service is available from Spectrum Micromechanical, Inc.

TDI International, Inc.

Location: Tucson, AZ, USA

Phone: 520-799-8000

Email: sales@tdiinternational.com

TDI International, Inc. provides precision hand tool and equipment, such as tweezers, diamond scribes, dry boxes, and glove boxes.

Ted Pella, Inc.

Location: Redding, CA, USA

Phone: 530-243-2200

Email: sales@tedpella.com

Ted Pella, Inc. sells instruments and supplies in a variety of types of microscopy: Transmission Electron Microscopy (TEM), and Scanning Electron Microscopy (SEM), and Atomic Force Microscopy (AFM), etc. Some relevant products include conductive glues and epoxies, AFM tips, and sample holders.