



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

Doctor of Philosophy

**HUMAN MOTION ESTIMATION
USING WEARABLE RGB-D CAMERA
AND INERTIAL SENSORS**

The Graduate School

of the University of Ulsan

Department of Electrical, Electronic and Computer Engineering

Duc-Cong Dang

**HUMAN MOTION ESTIMATION
USING WEARABLE RGB-D CAMERA
AND INERTIAL SENSORS**

Supervisor: Professor Young-Soo Suh

A Dissertation

**Submitted to
the Graduate School of the University of Ulsan
In partial Fulfillment of the Requirements
for the Degree of**

Doctor of Philosophy

by

Duc-Cong Dang

**Department of Electrical, Electronic and Computer Engineering
University of Ulsan, Korea**

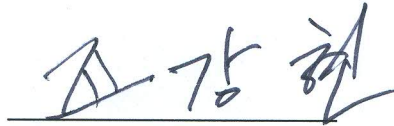
June 2023

**HUMAN MOTION ESTIMATION
USING WEARABLE RGB-D CAMERA
AND INERTIAL SENSORS**

This certifies that the dissertation thesis
of Duc-Cong Dang is approved by:



Committee Chair **Professor Hee-Jun Kang**



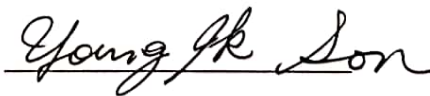
Committee Member **Professor Kang-Hyun Jo**



Committee Member **Professor Young-Joon Chee**



Committee Member **Professor Young-Soo Suh**



Committee Member **Professor Young-Ik Son**

Department of Electrical, Electronic and Computer Engineering
University of Ulsan, Korea

June 2023

Acknowledgements

This work would not have been possible without the combined support of many people.

First of all, I would like to express my sincere appreciation to my supervisor, Professor Young-Soo Suh, for his patient, kindly support throughout my period of research time. His encouragement and enthusiasm guidance are parts of my success.

I would like to thank the committee members, Professor Hee-Jun Kang, Professor Kang-Hyun Jo, Professor Young-Joon Chee and Professor Young-Ik Son for their precious time and helpful feedback to evaluate my work.

My acknowledgment is also sent to my friends and lab mates. They let me know that successful way of man is built with heartfelt friends. They are indispensable thing in my life.

The last but not least thank is for my family. Thanks Dad, Mom and my wife for their unconditional and boundless love. You all are always beside me whenever I am down or happy. Being with you is my most peaceful moment that engraved in my heart.

Abstract

Recently, human motion estimation is an important task and attracting significant attention in sports and medical applications, as it presents both theoretical and practical interest from bio-mechanical, computer vision and robotics perspectives. We focus on developing algorithms to: 1) estimate gait parameters, and 2) analysis human motion during walking activity, with reduced number of sensor count. In this dissertation, gait parameters are estimated as well as human lower-limb motion is reconstructed using a single waist-mounted Intel Realsense Depth Camera D455, which has an integrated 6-DOF Inertial Measurement Unit (IMU).

An inertial navigation algorithm is primarily proposed with only IMU data to obtain spatio-temporal gait parameters, such as: attitude, position, velocity of a human body; and stance phase duration, stride length, and walking distance. However, low-cost inertial sensors come with noise and bias that lead to unavoidable accumulative errors. Therefore, straight-line walking with a constant speed constraints is imposed in the filtering algorithm to improve attitude estimation. Visual odometry (VO) algorithm provides relative pose from image sequence, which plays as a measurement updating role for the filter. Detected stance foot from color and depth image data are used as landmarks to update foot position. Finally, a smoothing algorithm is proposed as a linear optimization problem to minimize estimation errors. Stance foot position is derived and other gait parameters can be calculated from step information.

A deep learning-based segmentation model with custom dataset is proposed to improve foot detection not only in stance phases but also in swing phases. 3-D dual foot trajectories are then calculated from proposed filter results and relative position of dual foot with respect to the camera. However, foot position in between Toe-Off and Mid-Stance phases are missing due to obscurity. To tackle this problem, a Graph Convolutional Network (GCN) based model is proposed to predict pose from previous poses. Complete foot trajectories are finally reconstructed with only a single waist-mounted RGB-D camera.

Through experiments, the proposed system shows promising results and could be applied for human motion estimation in real application with a considerable accuracy.

Contents

Abstract	iii
List of Figures	vi
List of Tables	viii
1 Introduction	1
1.1 Motivation	2
1.2 Thesis Contribution	4
1.3 Thesis Organization	4
2 Human Motion Estimation using Inertial Navigation Algorithms	6
2.1 Standard Inertial Navigation Algorithm using Indirect Kalman Filter	9
2.2 Measurement updates for Indirect Kalman Filter	11
2.3 Optimization-based Smoother for Inertial Navigation Algorithm	14
2.4 Chapter Summary	15
3 Visual Inertial Odometry for Gait analysis Application	16
3.1 System Overview	18
3.1.1 Hardware setup	18
3.1.2 Notation	18
3.2 Visual inertial odometry filtering	19
3.2.1 System equation and state definition	19
3.2.2 Visual odometry updating	23
3.3 Floor and foot detection	26
3.3.1 Floor plane detection	26
3.3.2 Foot detection and ellipse approximation	27

3.4	Measurement equations	28
3.4.1	Measurement equation for markers on the floor	29
3.4.2	Measurement equation for the stance foot	31
3.5	Filtering and smoothing algorithm	34
3.6	Chapter Summary	36
4	Deep Learning-based Human motion estimation	37
4.1	Foot detection framework	38
4.1.1	FCN for Semantic Segmentation	40
4.1.2	Custom Dataset	41
4.1.3	Foot detection model training	42
4.1.4	Foot position estimation	43
4.2	Human lower-body motion prediction framework	44
4.2.1	GCN for human motion prediction	45
4.2.2	Dataset: Lower body pose in walking action	47
4.2.3	Model training	47
4.2.4	Pose prediction	48
4.3	Chapter Summary	48
5	Experiments and Results	50
5.1	Experiments	51
5.2	Results	52
5.2.1	Visual Inertial Odometry results	52
5.2.2	Deep learning-based foot trajectory estimation results	55
5.2.3	Human lowerlimb motion estimation results	59
6	Conclusions and Future works	61
6.1	Conclusions	62
6.2	Future works	63
	Publications	65

List of Figures

3.1	Algorithm structure.	17
3.2	System overview.	18
3.3	Relative camera pose of two consecutive image frames	23
3.4	Stance and swing period	26
3.5	Example of stance foot set $S_{left,l}$	28
3.6	Markers in the starting and final points	29
4.1	An illustration of the progression from coarse to fine inference.	38
4.2	An example of semantic segmentation from Pascal VOC 2010.	39
4.3	An example of visualizing the groundtruth data in our custom dataset.	39
4.4	FCN structure.	40
4.5	FCN strides prediction.	41
4.6	An example of our custom dataset, including RGB, HHA, and groundtruth images using Labelme tool.	42
4.7	Keypoints on a detected foot.	43
4.8	Implemented network architecture with GCN.	47
5.1	Participant with equipped camera system doing the experiments.	51
5.2	Experiment setup, volunteer walks three straight paths from green triangle to red square through the working range of the optical tracker system.	51
5.3	Detected stance foot from point cloud data and masked in RGB image: 1) initial states; 2) first detected right foot; 3) already detected right foot, used as landmark; 4) first detected left foot.	52
5.4	Estimated walking trajectory using integrating internal IMU data only, proposed filter and smoother.	53

5.5	<i>xy</i> -plane estimated walking trajectory and foot position using proposed Kalman filter and smoother.	53
5.6	3-axis estimated walking trajectory and foot position using proposed Kalman filter and smoother (IC: Initial Contact, TO: Toe Off).	54
5.7	Compare estimated foot position with ground truth.	54
5.8	FCN evaluation results: 1) Pixel accuracy; 2: Mean IoU; 3: Frequency weighted average accuracy.	56
5.9	Deep learning-based foot detection results: prediction: 1) predicted classes from network; post-processing: 2) filtered blobs and smooth with depth data; 3) left and right feet separation; 4) 3d point cloud segmentation.	56
5.10	Three axis estimated trajectory of: 1) camera; 2) left and right feet; 3) 3d position of both camera and two foot.	57
5.11	3d position of estimated foot trajectory and ground truth.	58
5.12	3d position of GCN-based predicted left and right foot.	59
5.13	3d position of GCN-based predicted and VIO-based estimated left foot trajectory.	60
5.14	3d position of GCN-based predicted and groundtruth of dual foot trajectory.	60

List of Tables

5.1	Five Subjects Information	52
5.2	Estimated stride length error (Unit: Meter).	55
5.3	Estimated dual foot trajectories error (Unit: Meter).	58

Chapter 1

Introduction

1.1 Motivation

Human motion analysis has grown in popularity in a variety of fields, including clinical gait analysis, rehabilitation, robotics, sports science, animation and entertainment, video surveillance, and smart housing. Any process for acquiring quantitative information including the assessment of bio-mechanical variables such as spatial-temporal gait, kinematic, and kinetic factors is referred to as human motion estimation [1, 2]. The challenge for healthcare providers, sports bio-mechanistic, and researchers is to explore the mechanics of motion and their relationship with various diseases and/or injuries affecting the musculoskeletal system in less-constrained conditions outside of laboratory settings.

Recently, a variety of sensor-based systems are developed to perform gait analysis, particularly wearable sensing technologies, which provide intriguing potential for continuous monitoring of human kinematics and kinetics in free-living circumstances. Sensors can be placed on the floor (force sensors), inside insole of shoes (pressure or sensors), or on the subject body (wearable systems like inertial sensors or EMG electrodes). Floor platform-based and pressure sensors provide an accurate measurement of force pattern and foot pressure distribution to detect step and gait phases [3]. A force sensing resistor or piezoelectric-based in-socket sensor system can be used to detect gait phases from signal responses [4, 5]. EMG is used to measure the muscle electrical activity during walking, and to derive gait phases using the amplitude of EMG signals [6]. There are limitations of space and cost for those non-wearable systems.

Inertial sensors provide a cost effective solution with high sampling rates and require low computing capability to detect steps and estimate gait parameters [7, 8, 9]. Body-worn IMUs have demonstrated excellent capabilities to measure temporal gait parameter. However, accuracy and robustness of inertial sensors-based systems in measuring spatial parameters need to improve due to accumulative errors.

Vision based gait analysis can be separated into direct and indirect approaches. Direct method with optoelectronic systems is a gold standard for movement tracking based on markers placed on certain key points of the body with a sub-millimeter precision [10, 11]. However, those systems are expensive, difficult to set up, and cannot be used outside the clinic or laboratory. Indirect vision based approach extract features of the subjects (gender or human identity [12]) and gait parameters (step length and duration or ankle angles and

distance [13, 14]) from image processing and machine learning technique through a sequence of images. Others use single or multiple fixed depth cameras embedded into living environments to reconstruct 3-D shape and volume during walking which can be used to extract skeleton data and detect symmetrical gait troubles [15, 16].

While third-person view (TPV) systems have shows the potential to detect small changes over long period, these approaches suffer from visual occlusions (e.g., furniture) and the difficulty in detection when the full-body view is unavailable. Moreover, they are restricted to fixed areas. Egocentric first-person view (FPV), in which images are acquired from body-mounted cameras, provides rich and readable information on the propertises of the environment.

There are relatively few previous works aiming to extract spatial gait parameters using FPV cameras. A novel approach was using a walker-mounted depth and/or color camera to estimate 3D pose of lower limbs, mainly in frontal plane [17, 18, 19]. The key limitation of these works is the dependency on a stable platform (i.e., walker) to afford consistent views of the lower limbs and monitor pose over time, which is not generalizable to individuals that do not require a walking aid for ambulations.

The methods using of one or several body-mounted cameras are investigated for 3D full body [20, 21, 22] and upper limb [23, 24] pose estimation. In [20], more than ten cameras were attached to all the persons joints, and structure from motion approach was used to localize the cameras, estimate the joint angles and reconstruct human motion. The main disadvantage of the proposed method is the obtrusive multi-camera setup and intensive computational load. In [21], a 3D body pose model employs camera egomotion to infer body pose from synchronized videos captured by a chest-mounted camera and a Kinect sensor. However, the work were restricted to relatively static activities such as sitting and standing, and failed to examine dynamic scenarios.

Starting from these motivation, this dissertation studies the methods of using a single waist-mounted inertial sensors and RGB-D camera to obtain gait parameters as well as to reconstruct lower body trajectories. More precisely, the spatial gait parameters and forward/inverse kinematics of lower body motion, including step length, foot position, foot trajectory are specially considered in this dissertation.

1.2 Thesis Contribution

The dissertation presents the human motion estimation methods using a single waist-mounted RGB-D camera with integrated inertial sensors. Based on the flexibility and rich information provided by the sensors, human lower body motion can be reconstructed in order to provide a complete solution for gait analysis applications. In summary, the contributions of this research are as follows:

- We address the problem of the existing basic inertial navigation algorithms in human motion estimation and identify the motivation of this study.
- We propose a visual inertial odometry method for estimating human body walking trajectory and global foot position in the stance phases.
- We propose a segmentation algorithm based on deep learning to extend the foot detection algorithm in swing phases.
- We propose a graph convolution based network for predicting human lower body pose from a set of previous pose in order to reconstruct full lower body motion.
- We conduct experiments to evaluate the performance of the proposed method. A custom dataset including color and depth image data is created for foot detection framework.

1.3 Thesis Organization

The rest of this dissertation is organized as follows. In Chapter 2, an inertial navigation algorithm using a standard indirect Kalman filter and a smoother algorithm is introduced. This is fundamental method using to estimate attitude, position and velocity of the human body where the inertial sensor is attached. Chapter 3 proposes a visual inertial odometry algorithm to estimate human body walking trajectory and stance foot position. Relative attitude and relative position estimated from a visual odometry algorithm and detected stance foot position from depth image are used as measurement updates for the filtering and smoothing algorithm. In Chapter 4, the foot detection algorithm is extend to detect feet not only in the stance phases, but also in the swing phases by using deep learning-based segmentation method. Human lower body motion is then fully reconstructed by predicting next pose from a set of previous poses. Chapter 5 illustrates the experimental setup and the

evaluation results of the proposed method. Finally, Chapter 6 presents the conclusions and future directions for the dissertation.

Chapter 2

Human Motion Estimation using Inertial Navigation Algorithms

An inertial navigation algorithm is a crucial part in pedestrian navigation systems. In this chapter, a standard filter-based inertial navigation algorithm is presented to estimate attitude, position, and velocity of the human body from wearable inertial sensor data (including accelerometers and gyroscopes). Estimated errors can be reduced by applying a optimization-based smoothing algorithm.

Two coordinate frames are used in this chapter: the body coordinate frame and the navigation coordinate frame. The body coordinate frame is defined as a frame with three axes coincide with three axes of the IMU. The navigation (or global) coordinate frame is a local level frame with the z -axis coincides with the local gravitational direction (normally pointing upward). The x and y -axis of the navigation coordinate frame can be arbitrarily chosen. The notation p_n (p_b) is used to denote that a vector p is represented in the navigation (body) coordinate frame.

The IMU attitude can be represented using a quaternion $q = [q_0, q_1, q_2, q_3]' \in \mathbf{R}^4$ [25], which represents the rotation relationship between the navigation coordinate frame and the body coordinate frame. The directional cosine matrix corresponding to the quaternion q is defined by

$$C(q) = C_n^b(q) \triangleq \begin{bmatrix} 2q_0^2 + 2q_1^2 - 1 & 2q_1q_2 + 2q_0q_3 & 2q_1q_3 - 2q_0q_2 \\ 2q_1q_2 - 2q_0q_3 & 2q_0^2 + 2q_2^2 - 1 & 2q_2q_3 + 2q_0q_1 \\ 2q_1q_3 + 2q_0q_2 & 2q_2q_3 - 2q_0q_1 & 2q_0^2 + 2q_3^2 - 1 \end{bmatrix} \in SO(3).$$

Let $r_n \in \mathbf{R}^3$ and $v_n \in \mathbf{R}^3$ be the position and velocity of the IMU, expressed in the navigation coordinate frame, respectively. The basic equation for inertial navigation are given as follows [26]:

$$\begin{aligned} \dot{q} &= \frac{1}{2}q \otimes \omega_b = \frac{1}{2}\Omega(\omega_b)q \\ \dot{v}_n &= a_n = C'(q)a_b \\ \dot{r}_n &= v_n, \end{aligned} \tag{2.1}$$

where a_n and a_b are the acceleration made by forces other than gravitational field in the navigation coordinate frame and the body coordinate frame, respectively. $\omega_b = [\omega_x, \omega_y, \omega_z]'$

is the body angular rate. The symbol $\Omega \in \mathbf{R}^{4 \times 4}$ is defined by

$$\Omega(\omega) = \begin{bmatrix} 0 & -\omega' \\ \omega & -[\omega \times] \end{bmatrix} = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix},$$

where $[\omega \times] \in \mathbf{R}^{3 \times 3}$ denotes the skew symmetric matrix of ω .

The IMU used in this research consists of three axes accelerometer and three axes gyroscope. The accelerometer output $y_a \in \mathbf{R}^3$ and the gyroscope output $y_g \in \mathbf{R}^3$ are given by

$$\begin{aligned} y_a &= C(q)\tilde{g} + a_b + b_a + \eta_a \\ y_g &= \omega_b + b_g + \eta_g, \end{aligned} \tag{2.2}$$

where $\tilde{g} = [0, 0, g]'$ $\in \mathbf{R}^3$ denotes the local gravitational acceleration vector. Sensor noises $\eta_a \in \mathbf{R}^3$ and $\eta_g \in \mathbf{R}^3$ are assumed to be zero mean white Gaussian noises, whose covariances are given by

$$\begin{aligned} E \{ \eta_a(t) \eta_a'(s) \} &= r_a I_3 \delta(t - s) \\ E \{ \eta_g(t) \eta_g'(s) \} &= r_g I_3 \delta(t - s), \end{aligned} \tag{2.3}$$

where r_a and r_g are positive scalars.

Accelerometer bias $b_a \in \mathbf{R}^3$ and gyroscope bias $b_g \in \mathbf{R}^3$ are assumed to be nearly constant and can be modeled as follows

$$\begin{aligned} \dot{b}_a &= \eta_{b_a} \\ \dot{b}_g &= \eta_{b_g}, \end{aligned} \tag{2.4}$$

where $\eta_{b_a} \in \mathbf{R}^3$ and $\eta_{b_g} \in \mathbf{R}^3$ are small zero mean white Gaussian noise whose covariance

$$\begin{aligned} E \{ \eta_{b_a}(t) \eta_{b_a}'(s) \} &= Q_{b_a} \delta(t - s) \\ E \{ \eta_{b_g}(t) \eta_{b_g}'(s) \} &= Q_{b_g} \delta(t - s). \end{aligned}$$

2.1 Standard Inertial Navigation Algorithm using Indirect Kalman Filter

In this section, q , r and v are estimated using a standard inertial navigation algorithm using indirect Kalman filter. We implicitly understand that r and v are expressed in the navigation coordinate frame if there is no subscript.

Let \hat{q} , \hat{r} , \hat{v} , \hat{b}_g and \hat{b}_a be the estimated values of q , r , v , b_g and b_a using (2.1), where ω_b and a_b are replaced by $y_g - \hat{b}_g$ and $y_a - C(\hat{q})\tilde{g} - \hat{b}_a$. From (2.2) and (2.1), we have

$$\begin{aligned}\dot{\hat{q}} &= \frac{1}{2}\Omega(y_g - \hat{b}_g)\hat{q} \\ \dot{\hat{v}} &= C'(\hat{q})(y_a - \hat{b}_a) - \tilde{g} \\ \dot{\hat{r}} &= \hat{v}.\end{aligned}\tag{2.5}$$

The inertial navigation algorithm in this section uses an indirect Kalman filter to estimate the errors in quaternion, position and velocity instead of directly estimate quaternion, position and velocity. Let q_e , r_e , v_e , $b_{g,e}$, $b_{a,e}$ denote the small error in \hat{q} , \hat{r} , \hat{v} , \hat{b}_g , \hat{b}_a [27]:

$$\begin{aligned}q &= \hat{q} \otimes q_e \\ r_e &= r - \hat{r} \\ v_e &= v - \hat{v} \\ b_{g,e} &= b_g - \hat{b}_g \\ b_{a,e} &= b_a - \hat{b}_a,\end{aligned}\tag{2.6}$$

where \otimes is the quaternion multiplication operation, and q_e can be approximated by

$$q_e = f_{quat}(\bar{q}_e) \approx \begin{bmatrix} 1 \\ \bar{q}_e \end{bmatrix} \in \begin{bmatrix} \mathbf{R} \\ \mathbf{R}^3 \end{bmatrix}.$$

From [27] and [28], we have

$$\begin{aligned}\dot{\bar{q}}_e &= \frac{1}{2}(\omega_b - (y_g - \hat{b}_g)) + \frac{1}{2}\bar{q}_e \times (\omega_b - (y_g - \hat{b}_g)) - y_g \times \bar{q}_e \\ &\approx -y_g \times \bar{q}_e - \frac{1}{2}(b_{g,e} + \eta_g).\end{aligned}\tag{2.7}$$

Representing the first equation of (2.6) with the rotation matrix, we obtain

$$C(q) = C(q_e)C(\hat{q}) \approx (I - 2[\bar{q}_e \times])C(\hat{q}). \quad (2.8)$$

The state $x \in \mathbf{R}^{15 \times 1}$ of an indirect Kalman filter is defined by

$$x(t) = \begin{bmatrix} \bar{q}_e \\ r_e \\ v_e \\ b_{g,e} \\ b_{a,e} \end{bmatrix}.$$

The dynamic equation of $x(t)$ is given by

$$\dot{x}(t) = Ax(t) + \zeta, \quad (2.9)$$

where

$$A \triangleq \left[\begin{array}{ccc|cc} [-y_g \times] & 0_{3 \times 3} & 0_{3 \times 3} & -0.5I_3 & 0_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} & I_3 & 0_{3 \times 3} & 0_{3 \times 3} \\ -2C'(\hat{q})[y_a \times] & 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & -C'(\hat{q}) \\ \hline 0_{6 \times 3} & 0_{6 \times 3} & 0_{6 \times 3} & 0_{6 \times 3} & 0_{6 \times 3} \end{array} \right] \in \mathbf{R}^{15 \times 15}, \quad \zeta = \begin{bmatrix} -0.5\eta_g \\ 0_{3 \times 1} \\ -C'(\hat{q})\eta_a \\ \eta_{b_g} \\ \eta_{b_a} \end{bmatrix} \in \mathbf{R}^{15 \times 1}.$$

The noise ζ is process noise for compensation of slowly time-varying gyroscope and accelerometer bias. Assuming the noises in (2.9) are uncorrelated, zero mean white Gaussian noise, we obtain

$$\mathbb{E} \{ \zeta \zeta' \} = \mathbb{E} \left\{ \begin{bmatrix} \begin{bmatrix} -0.5\eta_g(t) \\ 0_{3 \times 1} \\ -C'(\hat{q})\eta_a(t) \\ \eta_{b_g}(t) \\ \eta_{b_a}(t) \end{bmatrix} \\ \begin{bmatrix} -0.5\eta_g(s) \\ 0_{3 \times 1} \\ -C'(\hat{q})\eta_a(s) \\ \eta_{b_g}(s) \\ \eta_{b_a}(s) \end{bmatrix} \end{bmatrix}' \right\} = Q\delta(t-s),$$

where $Q = \text{diag} \{ 0.25R_g, 0_{3 \times 3}, C'(\hat{q})R_aC(\hat{q}), Q_{b_g}, Q_{b_a} \}$.

Let T denotes the sampling period of an IMU sensor data. For a continuous time signal $y(t)$, the discrete value is denoted by $y_k = y(kT)$. The system (2.9) is discretized as follows

$$x_{k+1} = F_k x_k + \zeta_k, \quad (2.10)$$

where

$$F_k = \exp(AT)$$

$$Q_k = E\{\zeta_k \zeta_k'\} = \int_{kT}^{(k+1)T} \exp(At) Q \exp(At)' dt.$$

Note that F_k and Q_k should be computed at every step since A is time-varying. In this chapter, the following approximation is used [28]:

$$F_k \approx I_{15} + A_k T + 0.5 A_k^2 T^2$$

$$Q_k \approx QT + 0.5(A_k Q + Q A_k'),$$

where $A_k = A(kT)$.

Based on discrete model (2.10), a standard project ahead algorithm is used [29]:

$$\hat{x}_{k+1}^- = F_k \hat{x}_k$$

$$P_{k+1}^- = F_k P_k F_k' + Q_k, \quad (2.11)$$

where \hat{x}_k^- and \hat{x}_k are a priori state estimate before a measurement update and a state estimate after a measurement update, respectively. Estimation error covariance P_k^- and P_k are defined by

$$P_k^- = E\{(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)'\}$$

$$P_k = E\{(x_k - \hat{x}_k)(x_k - \hat{x}_k)'\}.$$

2.2 Measurement updates for Indirect Kalman Filter

When the object is being transformed, including moving and rotating, only the state projection (2.11) is used in the filter without any measurement update. The integrating process

might cause accumulate error if the error and bias are not compensated.

A so-called zero-velocity update (ZUPT) technique uses the fact that the velocity must be zero whereas the object is not moving in a specific time interval to reduce the error drift in the integration part of the filter, including position, velocity, and attitude estimations. This technique is usually used with foot-mounted inertial navigation systems, where the foot is stationary during stance phases. However, with the sensor unit attached on other body parts rather than on the feet, it is relatively difficult since there is no periodical instance of zero-velocity during walking.

A more appropriate approach for the attitude estimation is to apply gravity measurement update (GMU) and the zero-angular-rate update (ZARU) at detected events. The GMU model employs the acceleration measurement to compensate roll and pitch under the condition that no significant linear acceleration is present. Hence, the measured acceleration is considered to be due to gravitational acceleration and bias only. Whereas, the ZARU model relies on the condition that there is no rotational motion at the detected ZARU events, and the gyroscope measurement is due to sensor bias only.

In this section, we propose a two step measurement update for indirect Kalman filter, which consists of a gravity measurement update (GMU) and a zero-angular-rate update (ZARU). From (2.2) and (2.8), we obtain

$$y_{a,k} - C(\hat{q}_k)\tilde{g} - \hat{b}_{a,k} \approx 2 [C(\hat{q}_k)\tilde{g} \times] \bar{q}_{e,k} + b_{a,e,k} + a_{b,k} + \eta_{a,k} \quad (2.12)$$

$$y_{g,k} - \hat{b}_{g,k} = b_{g,e,k} + \omega_{b,k} + \eta_{g,k}. \quad (2.13)$$

GMU is utilized under the condition that no significant linear acceleration is present. An acceleration-moving-variance detector or an acceleration-magnitude detector can be used to determine when the IMU is stationary. A discrete time index k belongs to a non-moving interval if exist a moving window around k that satisfy the following conditions:

$$\|y_{a,i} - y_{a,i-1}\| \leq \delta_a, \quad \forall k - \frac{N_a}{2} \leq i \leq k + \frac{N_a}{2} \quad (2.14)$$

where δ_a is threshold value and N_a is specified window length for detecting the non-moving intervals.

When (2.14) is satisfied, the term $a_{b,k}$ in (2.12) can be ignored. Prior estimation is

calculated by (2.11), and $y_{a,k} - C(\hat{q}_k)\tilde{g} - \hat{b}_{a,k}$ is used as measurement update term for the filter as follows

$$\begin{aligned} K_{a,k+1} &= P_{a,k+1}^- H'_{a,k+1} \left(H_{a,k+1} P_{a,k+1}^- H'_{a,k+1} + R_a \right)^{-1} \\ \hat{x}_{k+1} &= \hat{x}_{k+1}^- + K_{a,k+1} (z_{a,k+1} - H_{a,k+1} \hat{x}_{k+1}^-) \\ P_{a,k+1} &= (I_{15} - K_{a,k+1} H_{a,k+1}) P_{a,k+1}^- (I_{15} - K_{a,k+1} H_{a,k+1})', \end{aligned}$$

where

$$\begin{aligned} H_{a,k+1} &= \begin{bmatrix} 2 [C(\hat{q}_{k+1}^-)\tilde{g}\times] & 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & I_3 \end{bmatrix} \\ z_{a,k+1} &= y_{a,k+1} - C(\hat{q}_{k+1}^-)\tilde{g} - \hat{b}_{a,k+1}^- \end{aligned} \quad (2.15)$$

ZARU is performed when no remarkable rotation is present. This can be determined by using a magnitude detector as follows:

$$\|y_{g,i}\| \leq \delta_g, \quad \forall k - \frac{N_g}{2} \leq i \leq k + \frac{N_g}{2} \quad (2.16)$$

where δ_g is threshold value and N_g is specified window length.

When (2.16) is satisfied, the term $\omega_{b,k}$ in (2.13) can be ignored. $y_{g,k} - \hat{b}_{g,k}$ is used as measurement update term for the filter as follows

$$\begin{aligned} K_{g,k+1} &= P_{g,k+1}^- H'_{g,k+1} \left(H_{g,k+1} P_{g,k+1}^- H'_{g,k+1} + R_g \right)^{-1} \\ \hat{x}_{k+1} &= \hat{x}_{k+1}^- + K_{g,k+1} (z_{g,k+1} - H_{g,k+1} \hat{x}_{k+1}^-) \\ P_{g,k+1} &= (I_{15} - K_{g,k+1} H_{g,k+1}) P_{g,k+1}^- (I_{15} - K_{g,k+1} H_{g,k+1})', \end{aligned}$$

where

$$\begin{aligned} H_{g,k+1} &= \begin{bmatrix} 0_{3\times 3} & 0_{3\times 3} & 0_{3\times 3} & I_3 & 0_{3\times 3} \end{bmatrix} \\ z_{g,k+1} &= y_{g,k+1} - \hat{b}_{g,k+1}^- \end{aligned} \quad (2.17)$$

A combination of above-mentioned detectors (2.14) and (2.16) can be used to detect the zero-velocity intervals, which appear in the before walking and after walking period, or in foot-mounted sensor cases.

2.3 Optimization-based Smoother for Inertial Navigation Algorithm

To reduce the estimation errors of the Kalman filter, a smoothing algorithm in [30] is applied in which the estimation problem is expressed as a quadratic optimization. Let Z_m be the set of discrete-time indices when the measurement is available. The measurement equation is given by

$$z_k = H_k x_k + \eta_k, \quad k \in Z_m,$$

where η_k is the measurement noise whose covariance matrix is R_k .

Let $\hat{q}_{SM,k}$, $\hat{r}_{SM,k}$ and $\hat{v}_{SM,k}$ be the smoothed estimations of the quaternion, position, and velocity, respectively. The estimation errors in $\hat{q}_{SM,k}$, $\hat{r}_{SM,k}$ and $\hat{v}_{SM,k}$ are defined as follows

$$x_{SM,k} = \begin{bmatrix} \bar{q}_{SM,k} \\ \bar{r}_{SM,k} \\ \bar{v}_{SM,k} \end{bmatrix} = \begin{bmatrix} [0_{3 \times 1} \quad I_3] (\hat{q}_k^* \otimes q_k) \\ r_k - \hat{r}_k \\ v_k - \hat{v}_k \end{bmatrix} \in \mathbf{R}^9 \quad (2.18)$$

$$\bar{b}_{g,SM} = b_g - \hat{b}_{g,N}$$

$$\bar{b}_{a,SM} = b_a - \hat{b}_{a,N},$$

where q^* denotes the quaternion conjugate of a quaternion q .

Let an optimization variable \tilde{X} is defined by

$$\tilde{X} = \begin{bmatrix} x_{SM,1} \\ x_{SM,2} \\ \vdots \\ x_{SM,N} \\ \bar{b}_{g,SM} \\ \bar{b}_{a,SM} \end{bmatrix} \in \mathbf{R}^{(9N+6) \times 1}.$$

We can formulate the smoothing problem as the following optimization problem:

$$\begin{aligned}
 J(\tilde{X}) = & \frac{1}{2} \sum_{k=1}^{N-1} (x_{SM,k+1} - F_k x_{SM,k})' Q_k^{-1} (x_{SM,k+1} - F_k x_{SM,k}) \\
 & + \frac{1}{2} \sum_{k \in Z_m} (z_k - H_k x_{SM,k})' R_k^{-1} (z_k - H_k x_{SM,k}) \\
 & + \frac{1}{2} (x_{SM,1} - x_{init})' P_{x_{init}}^{-1} (x_{SM,1} - x_{init}) \\
 & + \frac{1}{2} (\hat{b}_{g,N} + \bar{b}_{g,SM} - b_{g,init})' P_{b_{g,init}}^{-1} (\hat{b}_{g,N} + \bar{b}_{g,SM} - b_{g,init}) \\
 & + \frac{1}{2} (\hat{b}_{a,N} + \bar{b}_{a,SM} - b_{a,init})' P_{b_{a,init}}^{-1} (\hat{b}_{a,N} + \bar{b}_{a,SM} - b_{a,init}),
 \end{aligned} \tag{2.19}$$

where $x_{init} \in \mathbf{R}^3$, $b_{g,init} \in \mathbf{R}^3$, $b_{a,init} \in \mathbf{R}^3$ are the initial information.

We can easily see that (2.19) is a quadratic function of \tilde{X} , the matrix form of the optimization is given by

$$J(\tilde{X}) = \frac{1}{2} \tilde{X}' M_1 \tilde{X} + M_2 \tilde{X} + M_3, \tag{2.20}$$

where $M_1 \in \mathbf{R}^{(9N+6) \times (9N+6)}$, $M_2 \in \mathbf{R}^{(9N+6) \times 1}$, and $M_3 \in \mathbf{R}$ can be computed from (2.19). Since (2.20) is a quadratic function of \tilde{X} , it can be computed efficiently using the quadratic optimization method in [31]. Minimizing (2.20) will provide a set of estimation errors. From these values, \hat{q} , \hat{r} and \hat{v} can be updated using (2.6).

2.4 Chapter Summary

An inertial navigation algorithm using the indirect Kalman filter is simple but still provides a good quality of estimation for motion tracking. The Kalman filter's recursive structure allows its real-time execution without storing observations or past estimates. The limitation is that Kalman filter is only applied for white Gaussian noise processes and for linear models.

The smoothing-based inertial navigation algorithm uses a different approach for estimating the motion. In the optimization-based smoothing algorithm, all the data are used together to estimate each state at each sampling time. The advantage of this method is its high accuracy estimation comparing to other estimators. However, since all the data are used, it requires a high computational cost that is not feasible for real-time applications.

Chapter 3

Visual Inertial Odometry for Gait analysis Application

Visual odometry (VO) is widely used in autonomous navigation systems, humanoid robots or aerial vehicles to track and reconstruct the motion of the camera in real-time using sequential images [32, 33, 34]. However, the currently research on applying visual inertial odometry to gait analysis is surprisingly limited. In this chapter, we propose a visual-inertial approach to detect feet and estimate walking trajectory for gait parameter estimation, where a body-installed RGB-D camera are used for data acquisition. Relative attitude and relative position estimated from a visual odometry algorithm and detected stance foot position from depth image are used as measurement updates for a filtering and smoothing algorithm to compute two feet’s trajectories. The proposed algorithm in this chapter is designed based on the structure given in Fig. 3.1, where stance foot position estimation is the main objective.

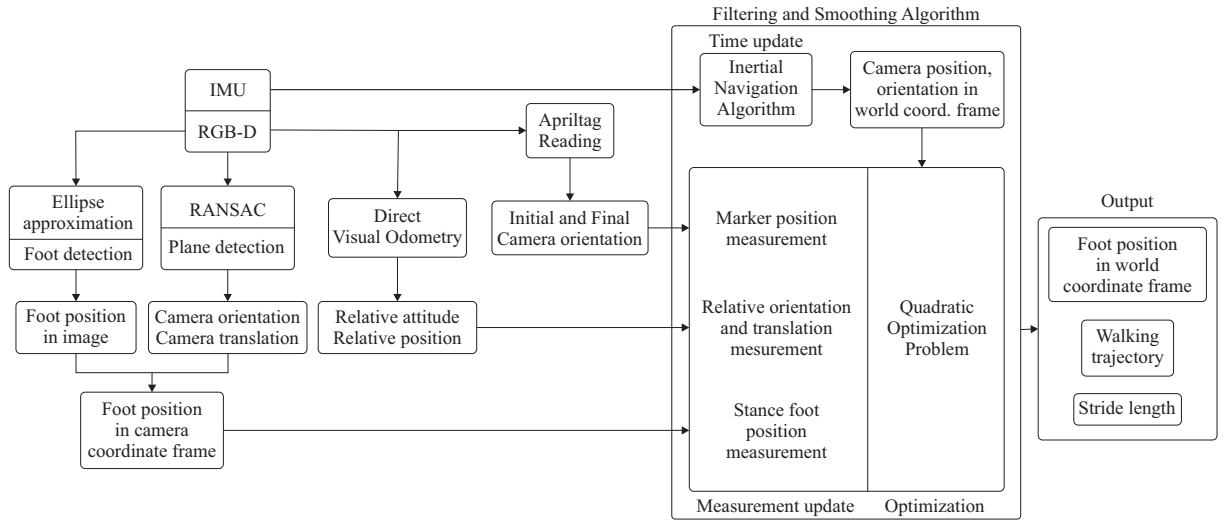


FIGURE 3.1: Algorithm structure.

The remainder of this chapter is organized as follows. Section 3.1 presents our system setup for data acquisition, the definition of the coordinate systems, and some notations. Section 3.2 describes the system equations and state definition for an inertial system along with extended states for the visual odometry information. Measurement equation from the visual odometry algorithm is derived for updating relative position and relative orientation. Section 3.3 presents the floor plane parameter estimation and foot tracking algorithm. Measurement equations for the markers at the initial and final step, and the measurement equation for the stance foot which is used as a landmark, are derived in Section 3.4. The filtering algorithm and the formulation of the smoothing algorithm are summarized in Section 3.5.

3.1 System Overview

3.1.1 Hardware setup

The system setup is shown in Fig. 3.2. Images are acquired using an Intel Realsense D455 RGB-D camera with a resolution of 640×480 pixels and a frame rate of 30Hz. A 6-DOF internal IMU of the camera is configured to provide 200Hz accelerometer and 200Hz gyroscope data. Two infrared markers are mounted on top of both feet to provide ground truth data from an optical tracker system. The intrinsic parameters of the camera are pre-calibrated using a chessboard with Camera Calibration Toolbox of MATLAB. Images and inertial data timestamps are synchronized.

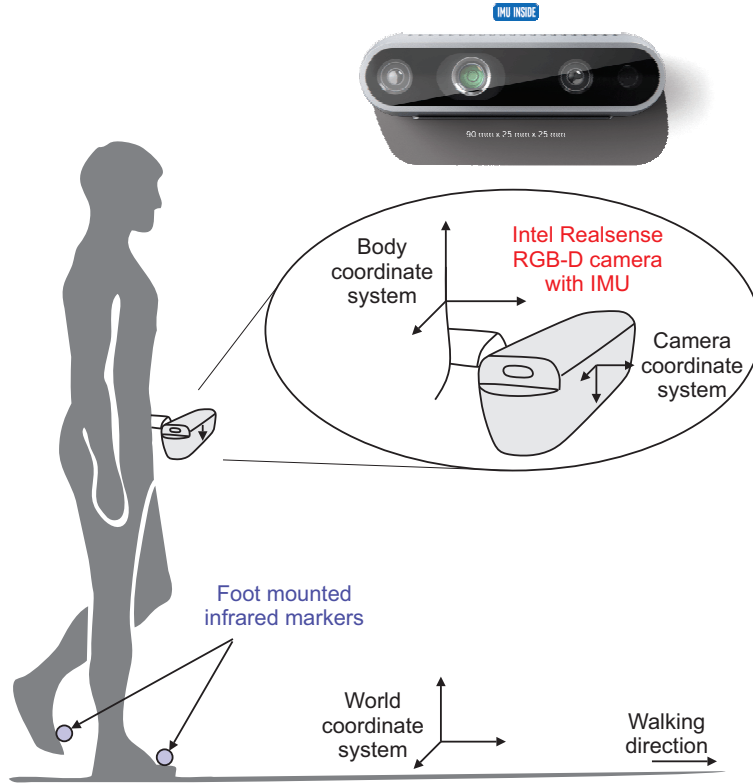


FIGURE 3.2: System overview.

3.1.2 Notation

Recall some notations from Chapter 2, for a vector $a \in \mathbf{R}^3$, $[a \times] \in \mathbf{R}^{3 \times 3}$ denotes the corresponding skew-symmetric matrix. For a quaternion $q \in \mathbf{R}^4$, $C(q) \in SO(3)$ denotes the corresponding rotation matrix.

$q_{identity} = \begin{bmatrix} 1 & 0_{1 \times 3} \end{bmatrix}'$ denotes the identity quaternion. Let \hat{q} denote the estimated value of a quaternion q . Assuming the estimation error is small, we use the following three vector error model $\bar{q}_e \in \mathbf{R}^3$ [35]:

$$q = \hat{q} \otimes f_{quat}(\bar{q}_e), \quad (3.1)$$

where \otimes denotes the quaternion multiplication, and

$$f_{quat}(\bar{q}_e) = \begin{bmatrix} 1 \\ \bar{q}_e \end{bmatrix} \in \begin{bmatrix} \mathbf{R} \\ \mathbf{R}^3 \end{bmatrix}.$$

Representing (3.1) with the rotation matrix, we obtain

$$C(q) \approx (I - 2[\bar{q}_e \times])C(\hat{q}).$$

Let $f_{vector} : \mathbf{R}^4 \rightarrow \mathbf{R}^3$ be a function extracting the vector part of a quaternion. For a matrix $A \in \mathbf{R}^{n \times m}$, $A(r_1 : r_2, c_1 : c_2) \in \mathbf{R}^{(r_2 - r_1 + 1) \times (c_2 - c_1 + 1)}$ denotes a submatrix of A consisting of rows $\{r_1, \dots, r_2\}$ and columns $\{c_1, \dots, c_2\}$.

3.2 Visual inertial odometry filtering

In this section, the state variables are defined to derive dynamic equations for a Kalman-based filter using IMU data. These states are extended to include visual odometry information. The visual odometry algorithm provides relative position and attitude of consecutive image frames as the measurement updating for the filter.

3.2.1 System equation and state definition

Let $q \in \mathbf{R}^4$, $r \in \mathbf{R}^3$ and $v \in \mathbf{R}^3$ be quaternion, position and velocity of IMU. The quaternion q represents the rotation from the world coordinate system to the body coordinate system. Let $y_a \in \mathbf{R}^3$ and $y_g \in \mathbf{R}^3$ be the accelerometer and gyroscope outputs:

$$\begin{aligned} y_a &= C(q)\tilde{g} + a_b + b_a + \eta_a \\ y_g &= \omega_b + b_g + \eta_g, \end{aligned} \quad (3.2)$$

where $\tilde{g} \in \mathbf{R}^3$ is the local gravitation vector, $a_b \in \mathbf{R}^3$ is the external acceleration and $\omega_b \in \mathbf{R}^3$ is the angular velocity. $b_a \in \mathbf{R}^3$ and $b_g \in \mathbf{R}^3$ are the accelerometer and gyroscope bias. $\eta_a \in \mathbf{R}^3$ and $\eta_g \in \mathbf{R}^3$ are white Gaussian sensor noises, whose covariances are given by $r_a I_3$ and $r_g I_3$.

Usually, the sampling period T_{camera} of a camera is larger than the sampling period T_{imu} of an IMU. In this work, we assume that the sampling period of a camera is an integer multiple of the sampling period of an IMU: that is, $T_{camera} = M_{ratio} T_{imu}$ for a positive integer M_{ratio} .

There are two discrete time indices k (with the sampling period T_{camera}) and i (with the sampling period T_{imu}). The function $f_{discrete}(k)$ relates two discrete indices:

$$i = f_{discrete}(k) = (k - 1)M_{ratio} + 1.$$

In the attitude and position filtering and smoothing, estimation error terms are usually estimated instead of direct estimation of attitude and position [27]. The estimation error terms $\bar{q}_{e,i} \in \mathbf{R}^3$, $r_{e,i} \in \mathbf{R}^3$ and $v_{e,i} \in \mathbf{R}^3$ are defined by:

$$\begin{aligned} q_i &= \hat{q}_i \otimes f_{quat}(\bar{q}_{e,i}) \\ r_i &= \hat{r}_i + r_{e,i} \\ v_i &= \hat{v}_i + v_{e,i}, \end{aligned} \tag{3.3}$$

where $\hat{q}_i \in \mathbf{R}^4$, $\hat{r}_i \in \mathbf{R}^3$ and $\hat{v}_i \in \mathbf{R}^3$ represent estimated values of q_i , r_i and v_i , respectively.

The dynamic equation of estimation error terms is given by [35]

$$\begin{bmatrix} \bar{q}_{e,i+1} \\ r_{e,i+1} \\ v_{e,i+1} \\ b_{g,e,i+1} \\ b_{a,e,i+1} \end{bmatrix} = F_i \begin{bmatrix} \bar{q}_{e,i} \\ r_{e,i} \\ v_{e,i} \\ b_{g,e,i} \\ b_{a,e,i} \end{bmatrix} + \zeta_i, \tag{3.4}$$

where $b_{g,e,i} \in \mathbf{R}^3$ and $b_{a,e,i} \in \mathbf{R}^3$ are error terms of sensor biases b_g and b_a . The covariance of noise ζ_i is given by [35]

$$Q_d = E\{\zeta_i \zeta_i'\} = \text{Diag}(0.25r_g I_3, 0_{3 \times 3}, r_a I_3, Q_{b_g}, Q_{b_a}).$$

F_i is computed by

$$F_i = \begin{bmatrix} F_{11,i} & F_{12,i} \\ 0_{6 \times 9} & I_6 \end{bmatrix} = \exp \left(\begin{bmatrix} A_{11,i} & A_{12,i} \\ 0_{6 \times 9} & 0_{6 \times 6} \end{bmatrix} T_{imu} \right), \quad (3.5)$$

where

$$A_{11,i} = \begin{bmatrix} [-y_{g,i} \times] & 0_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} & I_3 \\ -2C'(\hat{q}_i)[y_{a,i} \times] & 0_{3 \times 3} & 0_{3 \times 3} \end{bmatrix} \in \mathbf{R}^{9 \times 9}$$

$$A_{12} = \begin{bmatrix} -0.5I_3 & 0_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & -C'(\hat{q}_i) \end{bmatrix} \in \mathbf{R}^{9 \times 6}.$$

Note that (3.4) is a discrete-time system equation with the sampling period T_{imu} . Since measurement information from the camera is available with the sampling T_{camera} , (3.4) is transformed to a discrete-time system equation with the sampling period T_{camera} .

Let $U_k \in \mathbf{R}^9$ (notice that the image discrete index k is used instead of IMU discrete index i) be an error state vector defined by

$$U_k = \begin{bmatrix} \bar{q}_{e,fdiscrete(k)} \\ r_{e,fdiscrete(k)} \\ v_{e,fdiscrete(k)} \end{bmatrix} \in \mathbf{R}^{9 \times 1}. \quad (3.6)$$

Let $V_i \in \mathbf{R}^{15}$ be a state vector with bias terms defined by

$$V_k = \begin{bmatrix} U_k \\ b_{g,e,fdiscrete(k)} \\ b_{a,e,fdiscrete(k)} \end{bmatrix} \in \mathbf{R}^{15 \times 1}. \quad (3.7)$$

where $b_{g,e} \in \mathbf{R}^3$ and $b_{a,e} \in \mathbf{R}^3$ are error terms of sensor biases b_g and b_a .

By repeating (3.4) M_{ratio} times, the following equation is given by

$$V_{k+1} = \bar{F}_k V_k + \bar{\zeta}_k, \quad (3.8)$$

where

$$\begin{aligned}\bar{F}_k &= \prod_{i=f_{discrete}(k)}^{f_{discrete}(k)+M_{ratio}-1} F_i, \\ Q_{\bar{\zeta}} &= \mathbb{E}\{\bar{\zeta}_k \bar{\zeta}_k'\} = \sum_{j=1}^{M_{ratio}} G_{k+j-1} Q_d G_{k+j-1}', \\ \bar{G}_k &= I_{15}, \quad \bar{G}_{k+j} = G_{k+j-1} \bar{F}_{f_{discrete}(k+1)-j}, \\ &1 \leq j \leq M_{ratio} - 1.\end{aligned}$$

From the structure of F_i (see (3.5)), \bar{F}_k and $\bar{\zeta}_k$ are partitioned as follows for later use:

$$\begin{aligned}\bar{F}_k &= \begin{bmatrix} \bar{F}_{11,k} & \bar{F}_{12,k} \\ 0_{6 \times 9} & I_6 \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^{9 \times 9} & \mathbf{R}^{9 \times 6} \\ \mathbf{R}^{6 \times 9} & \mathbf{R}^{6 \times 6} \end{bmatrix}, \\ \bar{\zeta}_k &= \begin{bmatrix} \bar{\zeta}_{k,1} \\ \bar{\zeta}_{k,2} \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^9 \\ \mathbf{R}^6 \end{bmatrix}, \\ Q_{\bar{\zeta}_k} &= \begin{bmatrix} Q_{\bar{\zeta}_k,11} & Q_{\bar{\zeta}_k,12} \\ Q_{\bar{\zeta}_k,21} & Q_{\bar{\zeta}_k,22} \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^{9 \times 9} & \mathbf{R}^{9 \times 6} \\ \mathbf{R}^{6 \times 9} & \mathbf{R}^{6 \times 6} \end{bmatrix}.\end{aligned}\tag{3.9}$$

The visual odometry information at k -th discrete time is obtained using $(k-1)$ -th and k -th images; thus the information imposes constraints on U_{k-1} and U_k . To cope with this fact, we construct an extended state X_k containing both U_{k-1} and U_k [36].

Let the extended state X_k be defined by

$$X_k = \begin{bmatrix} U_k \\ U_{k-1} \\ b_{g,e,k} \\ b_{a,e,k} \end{bmatrix} \in \mathbf{R}^{24 \times 1}.$$

From (3.8), we have the following system equation for the Kalman filter.

$$X_{k+1} = \begin{bmatrix} \bar{F}_{11,k} & 0_{9 \times 9} & \bar{F}_{12,k} \\ I_9 & 0_{9 \times 9} & 0_{9 \times 6} \\ 0_{6 \times 9} & 0_{6 \times 9} & I_6 \end{bmatrix} X_k + \bar{\eta}_k,\tag{3.10}$$

where

$$\bar{\eta}_k = \begin{bmatrix} \bar{\zeta}_{k,1} \\ 0_{9 \times 1} \\ \bar{\zeta}_{k,2} \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^9 \\ \mathbf{R}^9 \\ \mathbf{R}^6 \end{bmatrix}.$$

Let $Q_{\bar{\eta}_k}$ be the covariance of $\bar{\eta}$, which can be computed from $Q_{\bar{\zeta}}$.

3.2.2 Visual odometry updating

Visual odometry estimates the motion of a camera using sequential images [37]. Visual odometry can be divided into two methods: indirect and direct methods. Indirect methods extract and track point features in the environment. Direct methods use raw image data by comparing pixel intensities of consecutive images. In this work, the direct method in [38] is used since this method is known to be robust in low-texture environments. Note that the camera in this paper is looking downward to the floor, which generally gives low-texture images.

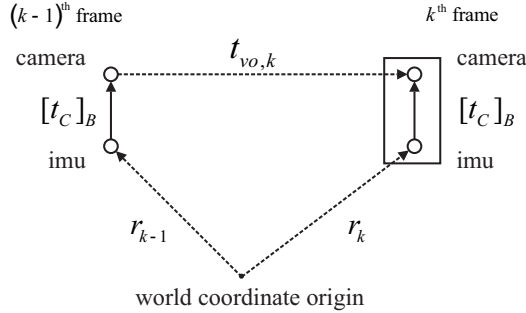


FIGURE 3.3: Relative camera pose of two consecutive image frames

Let r_k and q_k be the position and attitude of the IMU represented in the world coordinate corresponding to k -th image frame (see Fig. 3.3). Let $[t_C]_B$ and q_B^C be the position and orientation of the camera in IMU (body) coordinate. Since $[t_C]_B$ and q_B^C are known constants, the position and attitude of the camera in world coordinate can be calculated as $r_k + C(q_k)'[t_C]_B$ and $q_k \otimes q_B^C$ [35].

The visual odometry algorithm provides the relative attitude $q_{vo,k}$ and relative position $t_{vo,k}$ of the camera with its previous frame, (represented in the camera coordinate at the k -th discrete time) that are $(k-1)$ -th and k -th images.

The relative attitude $q_{vo,k}$ and relative position $t_{vo,k}$ satisfies the following equations:

$$q_k \otimes q_B^C = q_{k-1} \otimes q_B^C \otimes q_{vo,k}. \quad (3.11)$$

$$t_{vo,k} = C_B^C C(q_k) ((r_{k-1} + C(q_{k-1})'[t_C]_B) - (r_k + C(q_k)'[t_C]_B)). \quad (3.12)$$

Let $\hat{q}_{vo} \in \mathbf{R}^4$ and $\hat{t}_{vo} \in \mathbf{R}^3$ be visual odometry estimate values and $q_{vo,e} \in \mathbf{R}^4$ and $t_{vo,e} \in \mathbf{R}^3$ be estimation error terms defined by

$$\begin{aligned} q_{vo,k} &= \hat{q}_{vo,k} \otimes f_{quat}(\bar{q}_{vo,e,k}) \\ t_{vo,k} &= \hat{t}_{vo,k} + t_{vo,e,k}. \end{aligned} \quad (3.13)$$

Inserting (3.3) and (3.13) into (3.11), we have

$$\begin{aligned} q_{identity} &= (q_B^C)^* \otimes q_k^* \otimes q_{k-1} \otimes q_B^C \otimes q_{vo,k} \\ &= (q_B^C)^* \otimes \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes q_B^C \otimes \hat{q}_{vo,k} \\ &\quad - (q_B^C)^* \otimes f_{quat}(\bar{q}_{e,k}) \otimes \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes q_B^C \otimes \hat{q}_{vo,k} \\ &\quad + (q_B^C)^* \otimes \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes f_{quat}(\bar{q}_{e,k-1}) \otimes q_B^C \otimes \hat{q}_{vo,k} \\ &\quad + (q_B^C)^* \otimes \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes q_B^C \otimes \hat{q}_{vo,k} \otimes f_{quat}(\bar{q}_{vo,e,k}). \end{aligned} \quad (3.14)$$

In deriving the above equation, second-order error terms are ignored.

Let $z_{vo,q,k} \in \mathbf{R}^3$ (the vector part of a quaternion) be defined by

$$z_{vo,q,k} = f_{vector} \left((q_B^C)^* \otimes \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes q_B^C \otimes \hat{q}_{vo,k} \right). \quad (3.15)$$

Let $a \in \mathbf{R}^4$ and $b \in \mathbf{R}^4$ be quaternions and $\bar{c} \in \mathbf{R}^3$ be a vector. It is straightforward to verify that

$$a \otimes \begin{bmatrix} 0 \\ \bar{c} \end{bmatrix} \otimes b = \begin{bmatrix} \star \\ L(a, b)\bar{c} \end{bmatrix}, \quad (3.16)$$

where ‘ \star ’ term is irrelevant and

$$L(a, b) = -(\bar{b}\bar{a}') + (a_0 b_0)I + b_0[\bar{a} \times] - a_0[\bar{b} \times] - [\bar{b} \times][\bar{a} \times].$$

Combining (3.14), (3.15) and (3.16), we obtain

$$\begin{aligned}
 z_{vo,q,k} &= L((q_B^C)^*, \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes q_B^C \otimes \hat{q}_{vo,k}) \bar{q}_{e,k} \\
 &- L((q_B^C)^* \otimes \hat{q}_k^* \otimes \hat{q}_{k-1}, q_B^C \otimes \hat{q}_{vo,k}) \bar{q}_{e,k-1} \\
 &- L((q_B^C)^* \otimes \hat{q}_k^* \otimes \hat{q}_{k-1} \otimes q_B^C \otimes \hat{q}_{vo,k}, q_{identity}) \bar{q}_{vo,e,k}.
 \end{aligned} \tag{3.17}$$

Let $z_{vo,t,k}$ be defined by

$$z_{vo,t,k} = \hat{t}_{vo,k} - \hat{t}_k, \tag{3.18}$$

where

$$\hat{t}_k = C_B^C C(\hat{q}_k)(\hat{r}_{k-1} - \hat{r}_k) + C_B^C (C(\hat{q}_k) C(\hat{q}_{k-1})' - I) [t_C]_B.$$

Combining (3.14) and (3.18), we obtain

$$\begin{aligned}
 z_{vo,t,k} &\approx C_B^C C(q_k)(r_{e,k-1} - r_{e,k}) \\
 &- 2C_B^C [\bar{q}_{e,k} \times] C(\hat{q}_k)(\hat{r}_{k-1} - \hat{r}_k) \\
 &- 2C_B^C [\bar{q}_{e,k} \times] C(\hat{q}_k) C(\hat{q}_{k-1})' [t_C]_B \\
 &- 2C_B^C C(\hat{q}_k) C(\hat{q}_{k-1})' [\bar{q}_{e,k-1} \times]' [t_C]_B - t_{vo,e,k} \\
 &= C_B^C C(q_k)(r_{e,k-1} - r_{e,k}) \\
 &+ 2C_B^C [(C(\hat{q}_k)(\hat{r}_{k-1} - \hat{r}_k)) \times] \bar{q}_{e,k} \\
 &+ 2C_B^C [(C(\hat{q}_k) C(\hat{q}_{k-1})' [t_C]_B) \times] \bar{q}_{e,k} \\
 &- 2C_B^C C(\hat{q}_k) C(\hat{q}_{k-1})' [[t_C]_B \times] \bar{q}_{e,k-1} - t_{vo,e,k}.
 \end{aligned} \tag{3.19}$$

Let $z_{vo,k}$ be defined by

$$z_{vo,k} = \begin{bmatrix} z_{vo,q,k} \\ z_{vo,t,k} \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^3 \\ \mathbf{R}^3 \end{bmatrix}. \tag{3.20}$$

The measurement equation from the visual odometry algorithm is given by

$$z_{vo,k} = H_{vo,k} X_k + \eta_{vo,k}, \tag{3.21}$$

where $H_{vo,k} \in \mathbf{R}^{6 \times 24}$ can be derived from (3.17) and (3.19). Let $R_{vo,k} \in \mathbf{R}^{6 \times 6}$ denote the covariance of $\eta_{vo,k}$ which is given by

$$R_{vo,k} = J_{vo,k} \begin{bmatrix} r_{vo,q} I_3 & 0_{3 \times 3} \\ 0_{3 \times 3} & r_{vo,t} I_3 \end{bmatrix},$$

where $J_{vo,k} \in \mathbf{R}$ is the sum of squares of brightness residual of each pixel [38].

3.3 Floor and foot detection

As in Fig. 3.4, walking cycle consists of stance period (when a foot is on the floor) and swing period (when a foot is swinging). A stance period is detected (Section 3.3.2) and is used as a landmark (Section 3.4.2) since a foot does not move during stance period even if the other body parts are moving.

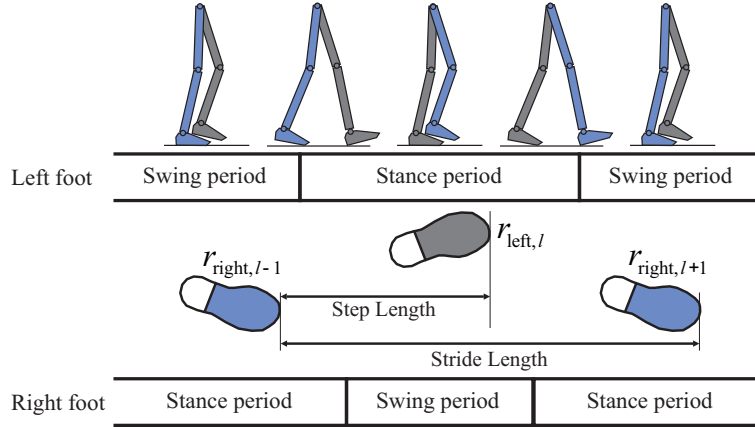


FIGURE 3.4: Stance and swing period

Let $r_{left,l} \in \mathbf{R}^3$ be the position (expressed in the world coordinate system) of the l -th left stance period foot. Similarly, $r_{right,l} \in \mathbf{R}^3$ is defined for the right foot. The final goal of this chapter is to estimate foot positions $r_{left,l}$ and $r_{right,l}$ and the walking stride length is computed from the foot positions (see Fig. 3.4).

In this section, the stance period foot is detected from depth images. The foot could have been detected from RGB images. However, it is not easy to detect foot if the shoe and floor have similar color or lighting condition is poor. Since depth image does not depend on color and lighting conditions, more robust foot detection is possible.

3.3.1 Floor plane detection

From depth image, floor plane is detected. The floor plane helps to detect a stance period foot.

Let $[p(u, v)]_C \in \mathbf{R}^3$ be a point on the floor represented in the camera coordinate system corresponding to (u, v) pixel, which is obtained from the depth camera. The floor plane

equation is given by

$$[n_{floor}]_C [p]_C + [h_{floor}]_C = 0, \quad [h_{floor}]_C > 0, \quad (3.22)$$

where $[n_{floor}]_C \in \mathbf{R}^3$ is the unit normal vector of the floor plane and $[h_{floor}]_C \in \mathbf{R} > 0$ is the distance to the plane from the camera.

Plane parameters are estimated using RANSAC algorithm [39].

3.3.2 Foot detection and ellipse approximation

A stance period foot instep is slightly above the floor plane. Based on this observation, foot candidate points are selected.

Firstly, the height matrix $F_{height}(u, v)$ (the height from the floor) is constructed as follows:

$$F_{height}(u, v) = [\hat{h}_{floor}]_C - \hat{n}'_{floor} [p(u, v)]_C. \quad (3.23)$$

Then the foot candidate points are selected using the following condition:

$$h_{foot,min} \leq F_{height}(u, v) \leq h_{foot,max}. \quad (3.24)$$

For the pixels (u, v) satisfying (3.24), connected (8-direction connectivity) pixels are grouped [40]. If the number of points in a connected group is less than $N_{foot,min}$, then the group is discarded. If the number of remaining groups is greater than 2, only the two largest groups are selected as possible foot candidates.

For each candidate group of points, the foot edge is detected. Once foot edge is detected, foot is approximated by an ellipse with known major and minor axes length assuming that foot size is known.

Using a simple tracking algorithm, left and right feet are separated and stance period foot is detected. If a left stance period foot is detected at the k -th image, the position of a foot represented in the camera coordinate system is denoted by $y_{left,k} \in \mathbf{R}^3$. Similarly $y_{right,k}$ is defined for the right foot. The detected stance feet belonging to the same stance period are grouped. For the l -th stance period step, let $S_{left,l}$ be a set of discrete indices with $y_{left,k}$ is

available. An example is given in Fig. 3.5, where $S_{left,l}$ is given by

$$S_{left,l} = \{k, k + 1, k + 2, k + 3\}. \quad (3.25)$$

Note that $y_{left,k}$ and $y_{left,k+1}$ are different since the camera is moving as the body is moving. However, if $y_{left,i}$ and $y_{left,i+1}$ are transformed to the world coordinate system, they should be the same since a stance period foot is not moving. Similarly, a set $S_{right,l}$ is defined for the right foot. Let $s_{index}(k)$ be a function representing the step number given image with discrete index k . For the example in (3.25), s_{index} is given by

$$l = s_{index}(k) = s_{index}(k + 1) = s_{index}(k + 2) = s_{index}(k + 3).$$

Let S_{foot} be the union of all $S_{left,l}$ and $S_{right,l}$: that is, $k \in S_{foot}$ means that at least one stance foot is observed in the k -th image.

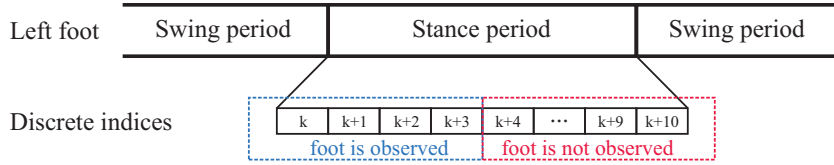


FIGURE 3.5: Example of stance foot set $S_{left,l}$

Let $[y_{foot}]_C \in \mathbf{R}^3$ be the stance foot (either left or right foot) position represented in the camera coordinate frame. The measured foot position is denoted by \hat{y}_{foot} and the following measurement model is used:

$$y_{foot} = \hat{y}_{foot} + \eta_{foot}, \quad (3.26)$$

where η_{foot} is a zero mean white Gaussian measurement noise with covariance $r_{foot}I_3 \in \mathbf{R}^{3 \times 3}$.

3.4 Measurement equations

In this section, in addition to measurement equation (3.21) from visual odometry algorithm, we propose two additional measurement updating as follows. April tags at known positions provide measurement updates in initial and final stances. Detected stance feet in Section 3.3 are used as landmarks to reduce estimated foot position errors.

3.4.1 Measurement equation for markers on the floor

Two A3 size papers containing 5×7 Apriltag markers [41] are placed both in the starting and final walking points as in Fig. 3.6. Two papers are placed along the world coordinate y axis and the distance between the left upper Apriltags of two papers is denoted by L , which is measured with a tape measure.

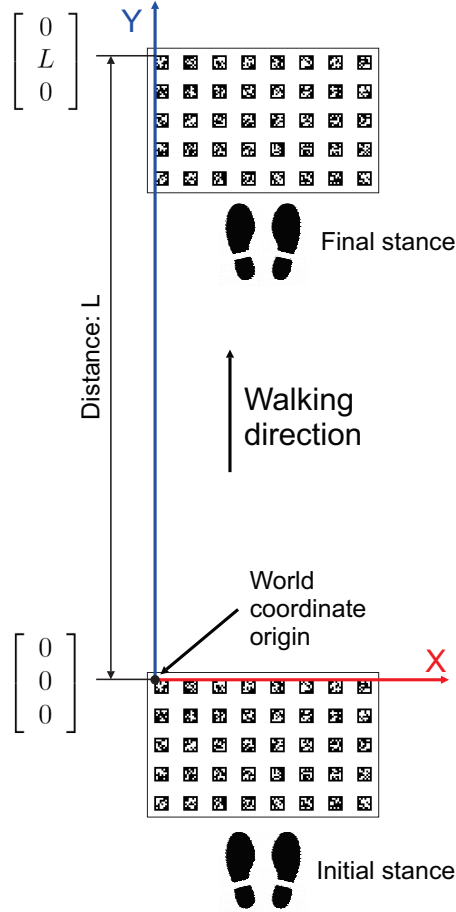


FIGURE 3.6: Markers in the starting and final points

Let $[r_{tag,j}]_C \in \mathbf{R}^3$ be four corners of Apriltags and $r_{img,j} \in \mathbf{R}^2$ be the corresponding measured image coordinates. Then the two points are related as follows

$$r_{img,j} = f_{img}([r_{tag,j}]_C) + \eta_{img,j}, \quad (3.27)$$

where

$$[r_{tag,j}]_C = C'(q_C^B)(C(q)([r_{tag,j}]_W - r) - [P_c]_B). \quad (3.28)$$

Function $f_{img} : \mathbf{R}^3 \rightarrow \mathbf{R}^2$ is defined by

$$f_{img} \left(\begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} \right) = \begin{bmatrix} \frac{x_j}{z_j} \\ \frac{y_j}{z_j} \end{bmatrix}.$$

An image measurement noise $\eta_{img,j} \in \mathbf{R}^2$ is assumed to be independent Gaussian noise whose covariance is $\epsilon_1 I_2 \in \mathbf{R}^{2 \times 2}$. Note that $[r_{tag,j}]_W$ is known since the tag location on the A3 paper is known.

Attitude and position of the camera can be estimated using the homography estimation method in [42]. Let $\hat{q}_{tag,k} \in \mathbf{R}^4$ and $\hat{r}_{tag,k} \in \mathbf{R}^3$ denote estimated attitude and position and let $\bar{q}_{tag,e,k} \in \mathbf{R}^3$ and $r_{tag,e,k} \in \mathbf{R}^3$ denote the corresponding estimation errors, where the following is satisfied

$$\begin{aligned} q_k &= \hat{q}_{tag,k} \otimes f_{quat}(\bar{q}_{tag,k,e}) \\ r_k &= \hat{r}_{tag,k} + r_{tag,k,e}. \end{aligned} \quad (3.29)$$

The estimation error covariance $R_{tag,k} \in \mathbf{R}^{6 \times 6}$ is given by

$$\begin{aligned} R_{tag,k} &= \left\{ \begin{bmatrix} \bar{q}_{tag,k,e} \\ r_{tag,k,e} \end{bmatrix} \begin{bmatrix} \bar{q}_{tag,k,e} \\ r_{tag,k,e} \end{bmatrix}' \right\} \\ &= \epsilon_1 \left(\sum_{j=1}^{N_{tag,k}} \bar{H}'_{tag,j,k} \bar{H}_{tag,j,k} \right)^{-}, \end{aligned} \quad (3.30)$$

where $N_{tag,k}$ is the number of observed tag corners in the k -th image.

$\bar{H}_{tag,j,k} \in \mathbf{R}^{2 \times 6}$ is given by

$$\begin{aligned} \bar{H}_{tag,j,k} &= \left. \frac{\partial f_{img}}{\partial r_{tag,j}} \frac{\partial r_{tag,j}}{\partial X_k} \right|_{r_{tag,j} = \hat{r}_{tag,j}} \\ &= \frac{1}{z_j^2} \begin{bmatrix} \hat{z}_j & 0 & -\hat{x}_j \\ 0 & \hat{z}_j & -\hat{y}_j \end{bmatrix} J_{tag,j}, \end{aligned}$$

where

$$[\hat{P}_{tag,j}]_C = C'(q_C^B)(C(\hat{q})([r_{tag,j}]_W - \hat{r}) - [t_C]_B) = \begin{bmatrix} \hat{x}_j \\ \hat{y}_j \\ \hat{z}_j \end{bmatrix} \quad (3.31)$$

$$J_{tag,j} = [2C'(q_C^B)[(C(\hat{q})([r_{tag,j}]_W - \hat{r})) \times], C'(q_C^B)C(\hat{q})] \in \mathbf{R}^{3 \times 6}. \quad (3.32)$$

$\hat{q}_{tag,1}$ and $\hat{r}_{tag,1}$ are used as initial attitude and position. After that only $\hat{r}_{tag,k}$ is used in the following measurement equation:

$$z_{tag,k} = \hat{r}_{tag,k} - \hat{r}_k = H_{tag,k} X_k + \eta_{tag,k} \quad (3.33)$$

where

$$H_{tag,k} = \begin{bmatrix} 0_{3 \times 3} & I_3 & 0_{3 \times 18} \end{bmatrix} \in \mathbf{R}^{3 \times 24}. \quad (3.34)$$

Recall that there are two A3 papers as in Fig. 3.6 which are in the starting and final positions. When tags are observed in the starting point, $R_{tag,k}(4:6,4:6)$ is used as a measurement noise covariance. When tags are observed in the final point, the measurement noise covariance is increased to compensate uncertainties in L . Thus, the following measurement covariance is used:

$$R_{tag,k}(4:6,4:6) + \begin{bmatrix} \epsilon_2 I_2 & 0_{2 \times 1} \\ 0_{1 \times 2} & 0 \end{bmatrix}.$$

3.4.2 Measurement equation for the stance foot

The stance foot detected in Section 3.3 is used as a landmark. The detected stance foot position is given by

$$\begin{aligned} [y_{foot}]_C &= (C_C^B)' \left(C(q) \left(\begin{bmatrix} \bar{r}_{foot} \\ 0 \end{bmatrix} - r \right) - [t_C]_B \right) \\ &= f_{foot}(q, r, \bar{r}_{foot}), \end{aligned} \quad (3.35)$$

where $\bar{r}_{foot} \in \mathbf{R}^2$ is xy world coordinates for the stance foot position. The z axis value is 0 since a stance foot is on the ground.

As in SLAM algorithms [43], a new stance foot position in the world coordinate frame is estimated whenever a new stance foot is observed.

Let $\hat{r}_{foot} \in \mathbf{R}^2$ be the estimated foot position, which can be computed from (3.35) once $[\hat{y}_{foot}]_C$ is given. Let $\bar{r}_{foot,e,m} \in \mathbf{R}^2$ be the m -th stance foot position estimation error:

$$\bar{r}_{foot,e,m} = \bar{r}_{foot,m} - \hat{r}_{foot,m}.$$

The state X_k is extended to include $\bar{r}_{foot,e,m}$ term. Let the extended state $\bar{X}_{m-1,k}$ with $m-1$ observed stance feet be defined by

$$\bar{X}_{m-1,k} = \begin{bmatrix} X_k \\ \bar{r}_{foot,e,1,k} \\ \vdots \\ \bar{r}_{foot,e,m-1,k} \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^{24} \\ \mathbf{R}^2 \\ \vdots \\ \mathbf{R}^2 \end{bmatrix}. \quad (3.36)$$

Let the estimation error covariance of $\hat{X}_{m-1,k}$ be denoted by $\bar{P}_{m-1,k}$:

$$\bar{P}_{m-1,k} = \begin{bmatrix} P_{X,m-1,k} & P_{X,foot,m-1,k} \\ P_{foot,X,m-1,k} & P_{foot,m-1,k} \end{bmatrix} \in \begin{bmatrix} \mathbf{R}^{24 \times 24} & \mathbf{R}^{24 \times 2(m-1)} \\ \mathbf{R}^{2(m-1) \times 24} & \mathbf{R}^{2(m-1) \times 2(m-1)} \end{bmatrix}. \quad (3.37)$$

When m -th stance foot is newly detected, the estimated state is extended as follows:

$$\hat{X}_{m,k} = \begin{bmatrix} \hat{X}_{m-1,k} \\ \hat{r}_{foot,e,m,k} \end{bmatrix}. \quad (3.38)$$

The state estimation error covariance $\bar{P}_{m,k}$ is given by

$$\bar{P}_{m,k} = \begin{bmatrix} \bar{P}_{m-1,k} & P'_{*,m,k} \\ P_{*,m,k} & P_{foot,m,k} \end{bmatrix}, \quad (3.39)$$

where

$$P_{foot,m,k} = G_X P_{X,m-1,k} G_X' + r_{foot} G_y G_y' \quad (3.40)$$

$$P_{*,m,k} = G_X \begin{bmatrix} P_{X,m-1,k} & P_{X,foot,m-1,k} \end{bmatrix} \quad (3.41)$$

and $G_X = \frac{\partial \bar{r}_{foot}}{\partial X}$ and $G_y = \frac{\partial \bar{r}_{foot}}{\partial v_{foot}}$ are computed from (3.44):

$$G_X = E_{12} \begin{bmatrix} 2C(\hat{q})'([t_C]_B + C_C^B \hat{y}_{foot}) \times & I_3 & 0_{3 \times 18} \end{bmatrix} \in \mathbf{R}^{2 \times 24} \quad (3.42)$$

$$G_y = E_{12} C(\hat{q})' C_C^B \in \mathbf{R}^{2 \times 3} \quad (3.43)$$

$$E_{12} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

G_X is computed from the following equation (ignoring second-order error terms):

$$\begin{aligned}
 \bar{r}_{foot} &= E_{12}(\hat{r} + r_e + C(\hat{q})'(I + 2[q_e \times])([t_C]_B \\
 &\quad + C_C^B(\hat{y}_{foot} + v_{foot}))) \\
 &= E_{12}(\hat{r}_{foot} - 2C(\hat{q})'([t_C]_B + C_C^B\hat{y}_{foot}) \times) q_e \\
 &\quad + r_e + C(\hat{q})'C_C^B v_{foot}.
 \end{aligned} \tag{3.44}$$

If the m -th detected foot $\hat{y}_{foot,m,k}$ has been observed before, it is used in the Kalman filter as a measurement. Let $z_{foot} \in \mathbf{R}^3$ be defined by

$$z_{foot,m,k} = \hat{y}_{foot,m,k} - f_{foot}(\hat{q}_k, \hat{r}_k, \hat{\bar{r}}_{foot,m,k}). \tag{3.45}$$

From (3.35), we have

$$\begin{aligned}
 z_{foot,m,k} &= f_{foot}(q_k, r_k, \bar{r}_{foot,m,k}) - \eta_{foot,k} \\
 &\quad - f_{foot}(\hat{q}_k, \hat{r}_k, \hat{\bar{r}}_{foot,m,k}) \\
 &= 2(C_C^B)'[(C(\hat{q})(E'_{12}\hat{\bar{r}}_{foot,m,k} - \hat{r})) \times] \bar{q}_{e,k} \\
 &\quad + (C_C^B)'C(\hat{q})(E'_{12}\bar{r}_{foot,e,m,k} - r_{e,k}) - \eta_{foot,k}.
 \end{aligned} \tag{3.46}$$

Inserting (3.46) into (3.45), we obtain

$$z_{foot,m,k} = \bar{H}_{foot,m,k} \bar{X}_{m,k} - \eta_{foot,k}, \tag{3.47}$$

where

$$\bar{H}_{foot,m,k} = \begin{bmatrix} H_{foot1,m,k} & 0_{3 \times 18+2(m-1)} & H_{foot2,m,k} \end{bmatrix} \tag{3.48}$$

$$\begin{aligned}
 H_{foot1,m,k} &= [2(C_C^B)'[(C(\hat{q})(\hat{r}_{foot,m,k} - \hat{r})) \times], \\
 &\quad -(C_C^B)'C(\hat{q}), \quad 0_{3 \times 3}] \in \mathbf{R}^{3 \times 9}.
 \end{aligned}$$

$$H_{foot2,m,k} = (C_C^B)'C(\hat{q})E'_{12} \in \mathbf{R}^{3 \times 2}.$$

When a state is extended, H_{tag} in Section 3.4.1 should also be extended to be compatible with the extended dimension by filling zeros.

3.5 Filtering and smoothing algorithm

In this section, the results in Section 3.2 and 3.4 are combined. A Kalman filter is used to estimate q , r , v and r_{foot} , where \hat{q}_{filter} , \hat{r}_{filter} , \hat{v}_{filter} and $\hat{r}_{filter,foot}$ are estimated values. Then based on the estimated value, a smoother algorithm is applied to obtain more accurate estimation, where $\hat{q}_{smoother}$, $\hat{r}_{smoother}$, $\hat{v}_{smoother}$ and $\hat{r}_{smoother,foot}$ are estimated values.

The filtering algorithm is fairly standard and is summarized in Algorithm 1.

```

initialization;
while  $k \leq N_{img}$  do
    time update (3.10);
    measurement update using visual odometry (3.21);
    if foot is detected then
        if first observed then
            extend state (3.38);
        else
            measurement update using the foot (3.47);
        end
    end
end

```

Algorithm 1: Filtering algorithm

The smoothing problem is formulated as a linear optimization problem, where small errors in the filter estimated values are computed. For example, the true position $r_k \in R^3$ can be modeled as

$$r_k = \hat{r}_{filter,k} + r_{e,k}. \quad (3.49)$$

In the smoothing algorithm, $r_{e,k}$ is estimated, where $\hat{r}_{e,k}$ is the estimated value. Then the position is given by

$$\hat{r}_{smoother,k} = \hat{r}_{filter,k} + \hat{r}_{e,k}. \quad (3.50)$$

Let the state of the smoother $X_{smoother}$ be defined by

$$X_{smoother} = \begin{bmatrix} U_1 \\ \vdots \\ U_{N_{image}} \\ b_{g,e} \\ b_{a,e} \\ r_{foot,e,1} \\ \vdots \\ r_{foot,e,M_{foot}} \end{bmatrix} \in R^{9N_{image}+6+2M_{foot}},$$

where M_{foot} is the total number of detected stance periods (both left and right feet).

The smoothing algorithm can be formulated as a linear quadratic optimization problem:

$$\begin{aligned} J(X) &= \sum_{k=1}^{N_{image}-1} \text{quad}(\bar{\zeta}_{k,1}, Q_{\bar{\zeta}_{k,1}}) \\ &+ \sum_{k=2}^{N_{image}} \text{quad}(z_{vo,k} - H_{vo,k}(1:6,1:18) \begin{bmatrix} U_k \\ U_{k-1} \end{bmatrix}, R_{vo,k}) \\ &+ \sum_{k \in S_{tag}} \text{quad}(z_{tag,k} - H_{tag,k}(1:3,1:9)U_k, R_{tag,k}(1:3,1:3)) \\ &+ \sum_{k \in S_{foot}} \text{quad}(z_{foot,s_{index}(k),k} - H_{foot1,s_{index}(k),k}X_k \\ &\quad - H_{foot2,s_{index}(k),k} r_{foot,e,s_{index}(k),k}, R_{foot,k}) \\ &+ \text{quad}(\hat{b} + b_{g,e} - b_{g,init}, P_{b_{g,init}}) \\ &+ \text{quad}(\hat{b} + b_{a,e} - b_{a,init}, P_{b_{a,init}}) \\ &+ \text{quad}(U_{init} - U_1, P_{U,init}) \\ &+ \text{quad}(U_{final} - U_{N_{image}}, P_{U,final}), \end{aligned}$$

where quad is defined by

$$\text{quad}(a, B) = \frac{1}{2} a' B a.$$

The initial and final constraints U_{init} and U_{final} are given by

$$U_{init} = \begin{bmatrix} f_{vector}(\hat{q}_1 \otimes \hat{q}_{tag,1}) \\ \hat{r}_{tag,1} - \hat{r}_1 \\ \mathbf{0}_{3 \times 1} - \hat{v}_1 \end{bmatrix},$$

$$U_{final} = \begin{bmatrix} f_{vector}(\hat{q}_{f_{discrete}(N_{image})} \otimes \hat{q}_{tag,N_{image}}) \\ \hat{r}_{tag,N_{image}} - \hat{r}_{f_{discrete}(N_{image})} \\ \mathbf{0}_{3 \times 1} - \hat{v}_{f_{discrete}(N_{image})} \end{bmatrix}.$$

From (3.8), $\bar{\zeta}_{k,1}$ in the quadratic term $\text{quad}(\bar{\zeta}_{k,1}, Q_{\bar{\zeta}_{k,1}})$ is given by

$$\bar{\zeta}_{k,1} = U_{k+1} - \bar{F}_{11,k} U_k - \bar{F}_{12,k} \begin{bmatrix} b_{g,e} \\ b_{a,e} \end{bmatrix}.$$

Once the optimal X is computed, $\hat{q}_{smoother}$, $\hat{r}_{smoother}$, $\hat{v}_{smoother}$ and $\hat{r}_{smoother,foot}$ can be computed with (3.50).

3.6 Chapter Summary

This chapter has proposed a filter and smoother algorithm to fuse IMU data with RGB-D camera data to estimate stance foot positions and reconstruct human body walking trajectory. Hence, gait parameters such as stride length, step time, walking speed can be directly derived from these information.

With the IMU-camera system attached to user waist, walking range is unrestricted. Therefore, longer walking range can be achieved in comparison with pressure mat or optical tracker system. Visual odometry algorithm provides incremental attitude and position of the camera alongside with position of the detected stance foot as a landmark in the measurement update equations of the filter. However, the proposed foot detection algorithm is restricted to only stance foot due to the height from the floor constraints.

In the next chapter, the deep learning-based model is applied in foot detection algorithm in order to extend the problem to foot walking trajectory estimation. From estimated waist and two feet walking trajectory, human lower body shape and pose during walking motion are reconstructed using another deep learning-based model.

Chapter 4

Deep Learning-based Human motion estimation

This chapter introduces an implementation of a semantic segmentation model to extend the foot detection algorithm (Section 3.3) in order to reconstruct two feet trajectories in walking motion. It is clear that with image data from downward-looking waist-mounted camera, foot is missing sometimes due to visual self-occlusion. To fully reconstruct human lower-body motion, we proposed a graph convolutional network to predict human pose from a set of previous poses.

4.1 Foot detection framework

Few studies interest in detecting 3D foot position from outside fixed camera in application of Augmented Reality, i.e. Virtual Try-On of Footwear [44], Virtual Mirror [45] or Magic Mirror [46]. Similar task is hand detection in human-computer interaction (HCI), gesture/sign language recognition and VR/AR [47, 48]. However, detecting feet in images from wearable downward looking camera is not a popular study [49].

Semantic Segmentation is the task of assigning each pixel a class label given an input image. Therefore, instead of a bounding box for each detected object, the segmentation framework gives a mask-like with labeled class for each object (see Fig. 4.1). It can be considered as image classification at the pixel level. Because the label is predicted for every pixel in the image, the task is also commonly referred as *dense prediction*.

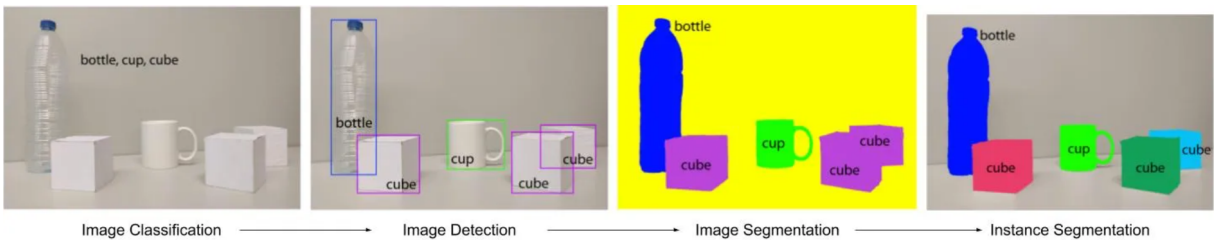


FIGURE 4.1: An illustration of the progression from coarse to fine inference.

Related works on semantic segmentation are widely used in indoor/outdoor scene understanding or biomedical imaging purposes, where the context of the environment is crucial. Fig. 4.2 shows an example of groundtruth labels from Pascal VOC 2010 dataset [50]. Each class is represented by a specific color that enable user to understand and evaluate a scene context.

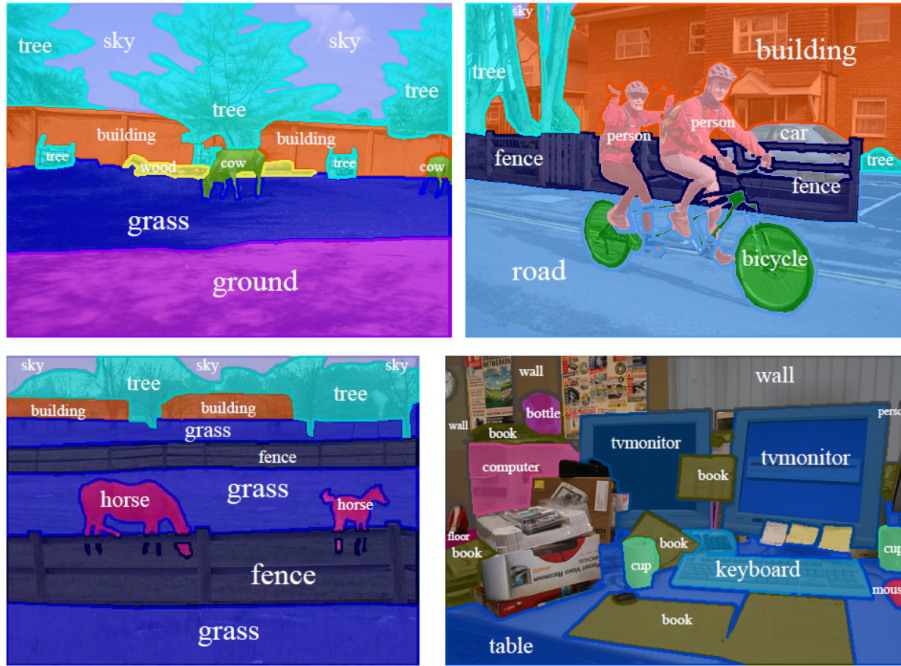


FIGURE 4.2: An example of semantic segmentation from Pascal VOC 2010.

In our study, the image only contains ground, feet, legs, and some objects placed on the ground. Hence, in most cases, there are only 3 classes are needed to be labeled: ground, foot and leg. Fig. 4.3 illustrates an example of a labeled data with defined classes in our created custom dataset.

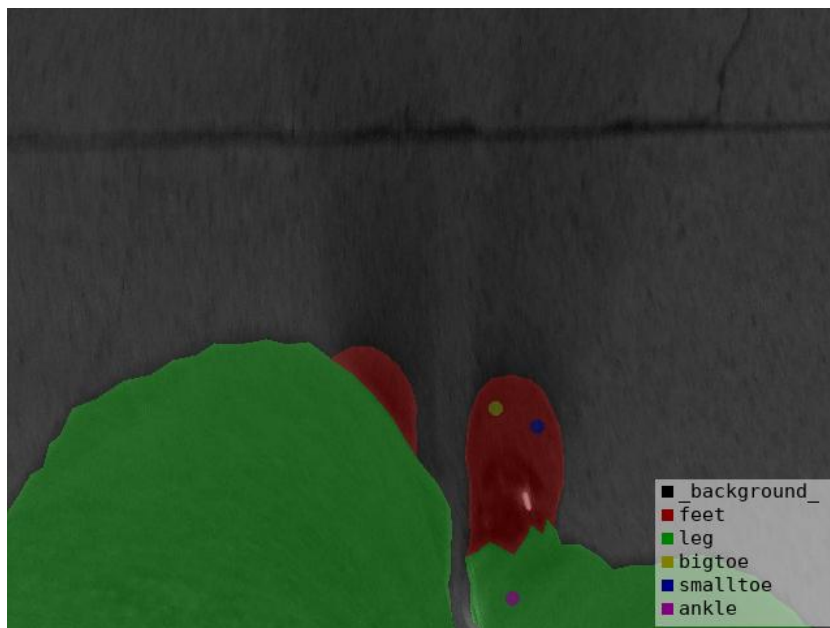


FIGURE 4.3: An example of visualizing the groundtruth data in our custom dataset.

Before Deep Neural Networks, quiet a few algorithms have been designed to solve this non-trivial task, such as Watershed algorithm, Image thresholding, K-means clustering, Conditional Random Fields, etc. In spite of many traditional image processing techniques the deep learning methods has been a game changer in the field of computer vision. One of the very early deep convolutional neural networks used for semantic segmentation is Fully Convolutional network (FCN). A variety of more advanced FCN-based approaches have been proposed to address this issue, including SegNet, U-Net, PSPNet, and DeepLab. In this study, FCN [51] is used to evaluate the semantic segmentation task.

4.1.1 FCN for Semantic Segmentation

A fully convolutional network (FCN) is the first work that re-architects and fine-tunes classification networks to direct, dense prediction of semantic segmentation. Both learning and inference are performed whole-image-at-a-time by dense feedforward computation and back-propagation (see Fig. 4.4).

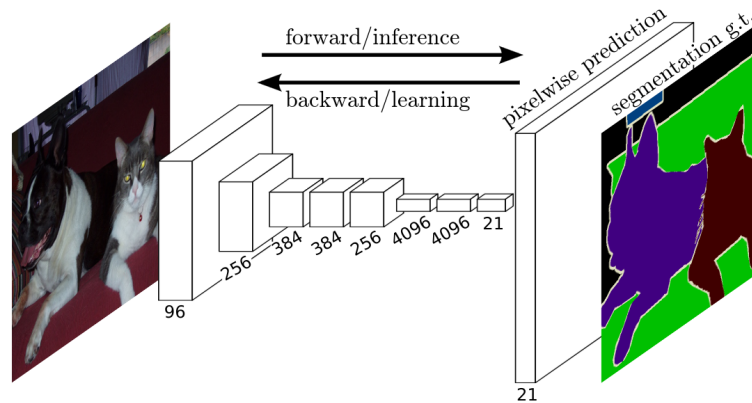


FIGURE 4.4: FCN structure.

In classification networks, conventionally, an input image is downsized and goes through convolution layers and fully connected layers, and output a predicted label. If the fully connected layers turn into 1×1 convolutional layers and the image is not downsized, the output will not be a single label. Instead, the output is a feature map with smaller resolution than the input image (due to max pooling). If the output is upsampled to the original resolution, then the pixelwise label map is obtained. Thus, by simply applying these tricks on any state-of-the-art deep CNNs for image classification, we can acquire corresponding FCNs for semantic segmentation.

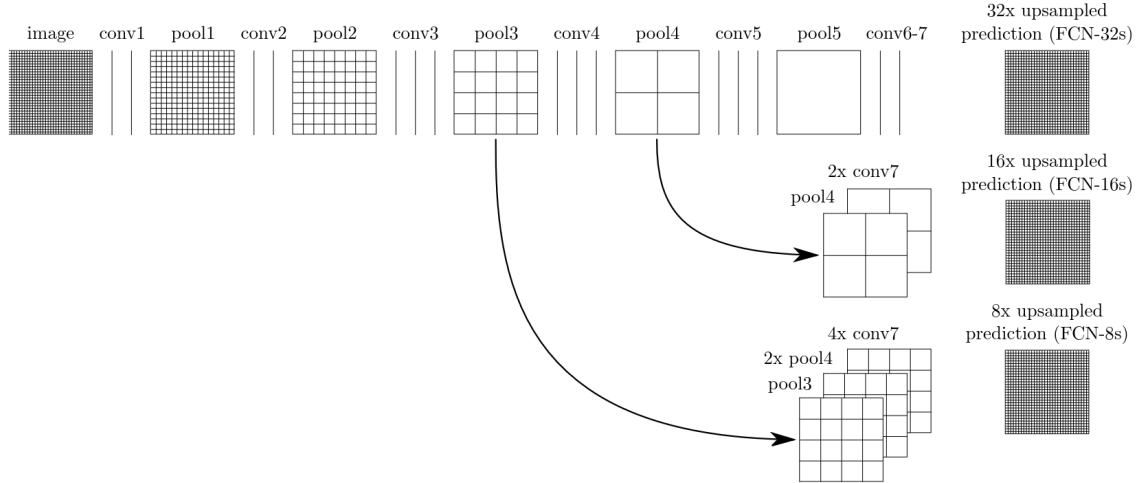


FIGURE 4.5: FCN strides prediction.

While fully convolutional classifiers can be fine-tuned to segmentation with even high score on the standard metrics, their output is unsatisfactorily coarse. The 32 pixel stride at final prediction layer limits the scale of the detail in the output. The authors addressed this problem by adding a links that combine the final prediction layer with lower layers with finer strides (see Fig. 4.5). Combining fine layers and coarse layers lets the model make predictions that respect global structure.

4.1.2 Custom Dataset

Sufficiently large datasets are challenging to collect. Some example benchmarks for semantic segmentation task are Cityscapes, PASCAL VOC, ADE20k. Those dataset include a huge number of scenes and classes. Dataset of egocentric hand segmentation are also available as EgoHands, Ego2Hands, Mo2Cap2, GTEA. However, these dataset are also not appropriate for foot segmentation task. To our knowledge, there is no existing dataset of downward looking camera for foot detection.

To create dataset for our task, we define a custom dataset classes based on VOC dataset. In addition to three RGB channels of input images, we include three more channels for HHA images, which is encoded from depth data with three channels at each pixel, including horizontal disparity, height above ground, and the angle that the pixel's local surface normal makes with the inferred gravity direction. The HHA representation encodes properties of geocentric pose that emphasize complementary discontinuities in the image (depth, surface

normal and height) because of which it works better than using raw depth images for learning feature representations with convolutional neural networks [52].

Color image and depth information are extracted from Intel Realsense RGB-D camera with a size of 640×480 . From our set of walking experiments, approximately 2500 images (6 channels) data are extracted. Among them, 250 images data is annotated in Pascal VOC format by drawing polygon regions around the legs and feet, using Labelme tool [53]. Fig. 4.6 shows the grid of some image data and labels generated from the tool.

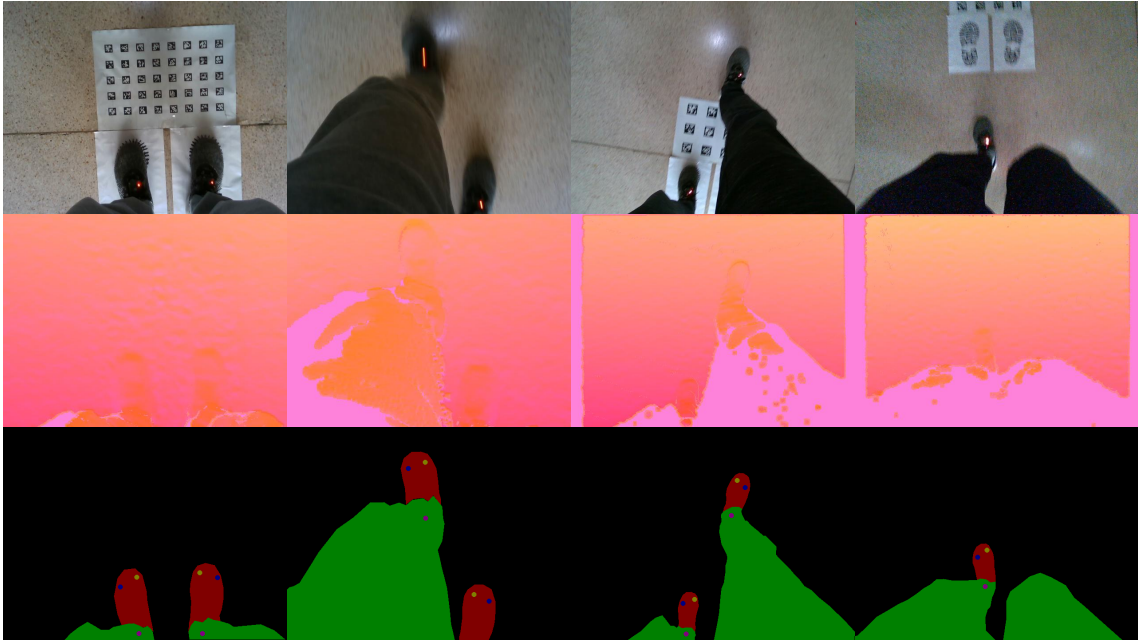


FIGURE 4.6: An example of our custom dataset, including RGB, HHA, and groundtruth images using Labelme tool.

Train and validate dataset are create from images data with label, test dataset is created from images extracted from other experiments.

4.1.3 Foot detection model training

To implement foot segmentation, the original FCN-8s and our modified FCN-8s was trained with our custom dataset as the following setup:

- RGB: Only RGB image as three-channel input
- HHA: Only HHA image as three-channel input
- RGBD: RGB and HHA images are fused with modified network structure

The modified FCN-8s net is jointly trained late fusion model that sums RGB and HHA predictions.

Cross entropy loss is typically used in semantic segmentation. However, class imbalance is one problem for one-stage detector using cross entropy loss. In our case, the background class is the dominant in most image frame. Focal loss [54] is a more effective alternative to deal with class imbalance. Stochastic Gradient Descent (SGD) is used as the optimizer during training process.

4.1.4 Foot position estimation

In this section, visible legs and feet are detected from RGB color and HHA images. Output of the foot detection network is an image with each pixel is assigned to its predicted label. This result is then post-processed to separate left and right legs and feet, and calculate foot positions in the image coordinate frame.

Before calculating foot position, the predicted image is fused with depth data to remove outliers smooth the predicted edges of feet and legs. We apply a simple threshold constraint on height information to correctly redefine the edges of each detected foot.

Let I_i be the i^{th} frame in a sequence of images with the length N retrieved from the waist-mounted camera in an experiment. Let the detected f -foot ($f = \{left, right\}$) in the I_i represented by $P_{f,i} = [p_{f,i}^t, p_{f,i}^m, p_{f,i}^b] \in \mathbf{R}^{3 \times 3}$, where $p_{f,i}^j(u, v) \in \mathbf{R}^3$ be a point on the foot surface represented in the camera coordinate frame corresponding to (u, v) pixel in the image (Fig. 4.7), and $j \in \{t, m, b\}$ which stands for $\{top, mid$ and $bot\}$.

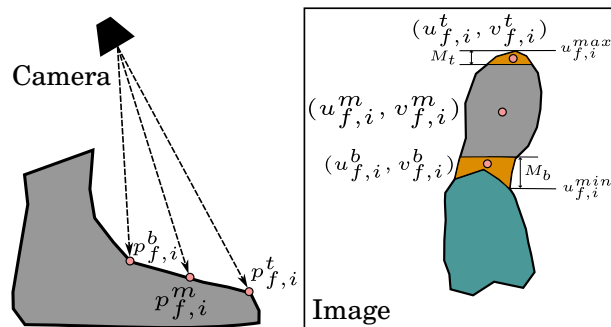


FIGURE 4.7: Keypoints on a detected foot.

Let S_f be set of (u, v) pixel that belongs to the f -foot. Three keypoints $\{top, mid$ and $bot\}$ are calculated as follows

$$\begin{aligned}(u^t, v^t) &= \text{mean}\{(u, v) | (u, v) \in S_f \ \& \ u^{max} - M_t < u < u^{max}\} \\ (u^b, v^b) &= \text{mean}\{(u, v) | (u, v) \in S_f \ \& \ u^{min} < u < u^{min} + M_b\} \\ (u^m, v^m) &= \text{mean}\{(u, v) | (u, v) \in S_f\},\end{aligned}\tag{4.1}$$

where $u^{max} = \max(u | (u, v) \in S_f)$, $u^{min} = \min(u | (u, v) \in S_f)$ and M_t, M_b is arbitrarily chosen as the scalar margin pixel threshold.

$P_{f,i}$ is then calculated from corresponding point cloud depth data. Hence, foot positions in the world coordinate frame are estimated from $P_{f,i}$ and camera position and orientation.

4.2 Human lower-body motion prediction framework

Human lower body motion prediction is a specific subtask of human motion prediction that focuses on forecasting the future movements of the hips, legs and feet of a human in a given environment. This task is important in applications such as sports training, physical therapy, and human-robot interactions.

The spatial and temporal structure of the human pose is an important aspect of human motion prediction. Spatial structure refers to the positions and orientations of the different body parts of a human, while temporal structure refers to the changes in these positions and orientations over time.

Various methods have been proposed to model the spatial and temporal structure of human pose for prediction. One popular approach is to use a hierarchical representation of the body, such as a kinematic chain or a tree structure. These representations can capture the relationships between different body parts and enable the modeling of the joint angles and positions. Another approach is to use a graph-based representation, where each node represents a body part and the edges represent the connections between them. Graph-based methods can capture more complex relationships between body parts and enable the modeling of non-linear dependencies.

Recurrent neural networks (RNNs) can capture the temporal dependencies in human motion trajectories and generate accurate predictions [55, 56, 57], while convolutional neural networks (CNNs) can extract spatial features from the input data and predict future poses

based on current poses and environmental factors such as obstacle positions [58, 59]. Recent research has also explored the use of graph neural networks (GNNs) to model the spatial and temporal structure of human pose. GNNs can learn the underlying graph structure and incorporate temporal information to predict future poses. One popular type of GNN for human pose prediction is the graph convolutional neural network (GCN). GCNs can be used to learn spatial and temporal features of the body pose by propagating information across the graph structure. GCNs have been used for both full body [60] and lower body pose prediction [61], and have shown promising results in terms of accuracy. Another type of GNN that has been used for human pose prediction is the graph attention network (GAT), which uses attention mechanisms to selectively weight the contributions of different nodes to the output. GATs have been shown to outperform traditional GCNs in some tasks and can better capture the complex relationships between different body parts [62, 63]. In this study, a GCN framework based on the work of [60] is used for predicting human lower body poses in walking activities, hence, to fully reconstruct trajectories of the dual foot.

4.2.1 GCN for human motion prediction

Problem formalization

We assume to be given the 3D coordinates or angles of the body consists of V joints and T frames. The history motion sequence is denoted by the tensor $\mathbf{X}_{1:T} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T] \in \mathbf{R}^{3V \times T}$, where $\mathbf{x}_k = [(x_k^1)', (x_k^2)', \dots, (x_k^V)']' \in \mathbf{R}^{3V \times 1}$ represents the pose at specific frame k . Let $\mathbf{x}^v = [(x_1^v)', (x_2^v)', \dots, (x_T^v)']' \in \mathbf{R}^{3T \times 1}$ be the motion of v^{th} joint (3D coordinates or angles). Our goal is to predict the future K poses $\mathbf{X}_{T+1:T+K} = [\mathbf{x}_{T+1}, \mathbf{x}_{T+2}, \dots, \mathbf{x}_{T+K}] \in \mathbf{R}^{3V \times K}$.

DCT-based Temporal encoding

To encode the temporal nature of human motion, we can either directly use the trajectories \mathbf{x}^v as input and output of for the motion prediction, or represent a trajectory using the Discrete Cosine Transform (DCT), which can provide more compact representation by discarding the high frequencies. Given a trajectory \mathbf{x}^v , the corresponding l^{th} DCT coefficient can be computed as

$$C_l^v = \sqrt{\frac{2}{T}} \sum_{n=1}^T x_n^v \frac{1}{\sqrt{1 + \delta_{l1}}} \cos\left(\frac{\pi}{2T}(2n-1)(l-1)\right), \quad (4.2)$$

where δ_{ij} denotes the *Kronecker* delta function with $\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0. & \text{if } i \neq j \end{cases}$

In practice, $l \in 1, 2, \dots, T$, but one can ignore the higher values, to remove the high motion frequencies. Equation (4.2) allows us to model the temporal information for each joint using DCT coefficients. Given such coefficients, the original pose representation (3D coordinates or angle) can be obtained via the Inverse Discrete Cosine Transform (IDCT) as

$$x_n^v = \sqrt{\frac{2}{T}} \sum_{l=1}^T C_l^v \frac{1}{\sqrt{1 + \delta_{l1}}} \cos\left(\frac{\pi}{2T}(2n-1)(l-1)\right), \quad (4.3)$$

where $n \in 1, 2, \dots, T$.

To make use of the DCT representation, instead of treating motion prediction as the problem of learning a mapping from $\mathbf{X}_{1:T}$ to $\mathbf{X}_{T+1:T+K}$, we reformulate it as a mapping between observed and future DCT coefficients.

Graph Convolution Layer

To encode the spatial structure of human pose, we make use of GCNs by assuming that human body is modeled as a fully-connected graph with V nodes. The strength of the edges in this graph can be represented by a weighted adjacency matrix $\mathbf{A} \in \mathbf{R}^{3V \times 3V}$. A graph convolution layer p then takes as input a tensor $\mathbf{H}^{(p)} \in \mathbf{R}^{3V \times F_p}$, with F_p is the number of features output by the previous layer. For the first layer, the network takes as input the $3V \times L$ matrix of DCT coefficients. Given this information and a set of trainable weights $\mathbf{W}^{(p)} \in \mathbf{R}^{F_p \times F_{p+1}}$, a graph convolutional layer outputs a matrix of the form

$$\mathbf{H}^{(p+1)} = \sigma(\mathbf{A}^{(p)} \mathbf{H}^{(p)} \mathbf{W}^{(p)}), \quad (4.4)$$

where $\mathbf{A}^{(p)}$ is the trainable weighted adjacency matrix for p^{th} layer and σ is the activation function, such as ReLU, PReLU or tanh.

Network Structure

We used the similar structure from [60] as in Fig. 4.8. It consists of 12 residual blocks, each of which consisting 2 graph convolutional layers and two additional graph convolutional layers, one at the beginning and one at the end, to encode the temporal information and decode the features to the residual DCT coefficients, respectively. Each layer p relies on a learnable weight matrix $\mathbf{W}^{(p)} \in \mathbf{R}^{256 \times 256}$, and a learnable weighted adjacency matrix $\mathbf{A}^{(p)}$.

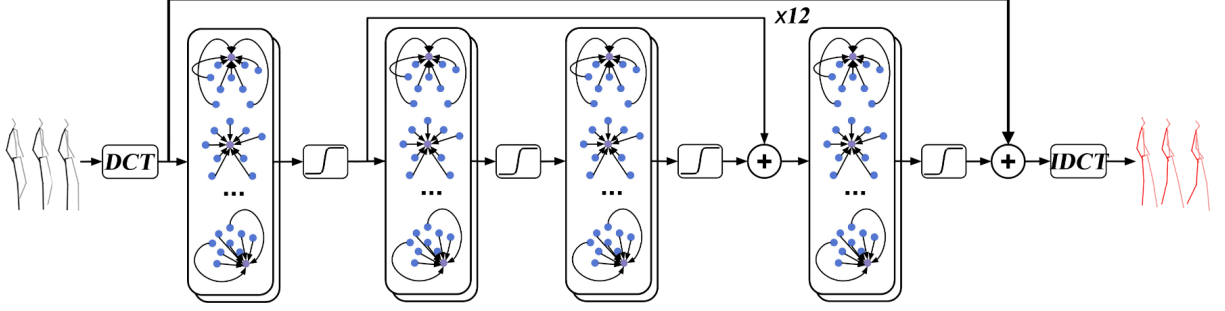


FIGURE 4.8: Implemented network architecture with GCN.

4.2.2 Dataset: Lower body pose in walking action

From the Archive of Motion Capture as Surface Shapes (AMASS [64]) dataset, we use the SMPL [65] parameterization to derive a representation of human pose based on a shape vector, which defines the human skeleton, and its joints rotation angles. Overall, AMASS consists of 40 human-subjects that perform the action of walking. Each human pose is represented by 52 joints, including 22 body joints and 30 hand joints. Here we consider for predicting lower body joints only and model as 7-joints human pose (root, 2-hips, 2-knees, 2-ankles). These sequences are downsampled to 30fps.

4.2.3 Model training

The model is trained to minimize the losses that measure error with respect to (w.r.t) ground truth in terms of Mean Per Joint Position Error (MPJPE) and Mean Angle Error (MAE). The loss based on MPJPE is defined as follows

$$L_{MPJPE} = \frac{1}{V(T+K)} \sum_{k=1}^{T+K} \sum_{v=1}^V \|\hat{p}_k^v - p_k^v\|_2, \quad (4.5)$$

where $\hat{p}_k^v \in \mathbf{R}^3$ denotes the predicted 3D coordinates of the joint v in the frame k , and $p_k^v \in \mathbf{R}^3$ is the corresponding groundtruth. The loss based on MAE is given by

$$L_{MAE} = \frac{1}{V(T+K)} \sum_{k=1}^{T+K} \sum_{v=1}^V |\hat{x}_k^v - x_k^v|, \quad (4.6)$$

where $\hat{x}_k^v \in \mathbf{R}^3$ denotes the predicted joint angles in exponential map representation of the joint v in the frame k , and $x_k^v \in \mathbf{R}^3$ is its groundtruth.

4.2.4 Pose prediction

In this section, dual foot trajectories are fully reconstructed by predicting future poses from a set of previous poses and the measurements are the estimated dual foot positions and the camera position and orientation. The pose forecasting is the feed-forward process, thus, a numerical optimizer is added to correct the predicted pose in order to align with measurements. In this study, the human pose representation is chosen as joint angles. The corresponding joint positions are calculated from forward kinematic (i.e. joint regressor in SMPL model).

Assume a walking sequence contains total N frames with the poses in first T frames are known. Let $\bar{\mathbf{x}}_i = \hat{\mathbf{x}}_{i+K-1} \in \mathbf{R}^{21 \times 1}$ ($i > T$) be the last pose in the set of K poses $\hat{\mathbf{X}}_{i:i+K-1}$ predicted from the last T poses $\mathbf{X}_{i-T:i-1}$ at the i^{th} frame

$$\hat{\mathbf{X}}_{i:i+K-1} = \text{net}(\mathbf{X}_{i-T:i-1}).$$

The optimal estimated pose \mathbf{x}_i^* is defined as the pose that minimizes the following optimization problem

$$J(\mathbf{x}) = w^{init}|\bar{\mathbf{x}}_i - \mathbf{x}| + w^{root}(|\tilde{x}_i^r - x^r| + \|\tilde{p}_i^r - p^r\|_2) + w^{foot}\|\tilde{p}_i^f - p^f\|_2, \quad (4.7)$$

where the terms $\tilde{x}_i^r, \tilde{p}_i^r$ refer to the measurements of root joint r at frame i (angle and position, respectively). Similarly, \tilde{p}_i^f is the measurement of foot joint f (position of left or right feet) at frame i . The weights w^{init}, w^{root} and w^{foot} are arbitrarily chosen. Besides, the constraints of the shank length and thigh length can be added into the optimization as follows

$$J(\mathbf{x}) = w^{init}|\bar{\mathbf{x}}_i - \mathbf{x}| + w^{root}(|\tilde{x}_i^r - x^r| + \|\tilde{p}_i^r - p^r\|_2) + w^{foot}\|\tilde{p}_i^f - p^f\|_2 \\ + w^{shank}(\|p^k - p^a\|_2 - L_{shank}) + w^{thigh}(\|p^k - p^h\|_2 - L_{thigh}), \quad (4.8)$$

where p^k, p^a, p^h are position of knee, ankle and hip joint, respectively. L_{shank} and L_{thigh} are the constrained shank and thigh length.

4.3 Chapter Summary

This chapter has presented two deep learning-based frameworks to improve foot positions estimation and reconstruct human lower-body walking trajectory.

A FCN-based semantic segmentation model is trained with our custom dataset for legs and feet detection. The network is modified to use both color and depth information in the framework. The segmented output is then post-processed to separate left and right foot, and compute each foot 3-D position with respect to camera and world coordinate frame. The foot trajectories are derived from these information. However, foot positions are missing during the interval between a foot Toe Off and Mid Swing events due to self occlusion.

A lower-body human pose prediction framework is used to deal with foot missing problem and reconstruct the whole foot trajectories in walking action. The prediction framework is based on a GCN to model the spatial and temporal structure of human pose. We use the DCT representation to encode the temporal nature, and the graph-based model of human body to encode the spatial structure of human motion. The framework is trained with existing AMASS dataset with selected walking sequences and extracted lower-body with 7 joints. The optimal pose is corrected by apply some constraints to the predicted pose in a optimization problem.

In the next chapter, the experiments are conducted to verify the algorithm accuracy.

Chapter 5

Experiments and Results

5.1 Experiments

Experiments are performed to verify the proposed algorithm. Camera frame is attached to user's waist by velcro tapes and buckles clips, and two reflective markers (for the motion capture system) are mounted on top of both feet (see Fig. 5.1). A motion capture system is set up with 6 cameras from Optitrack, creating a tracking space of about 5m long in Fig. 5.2.

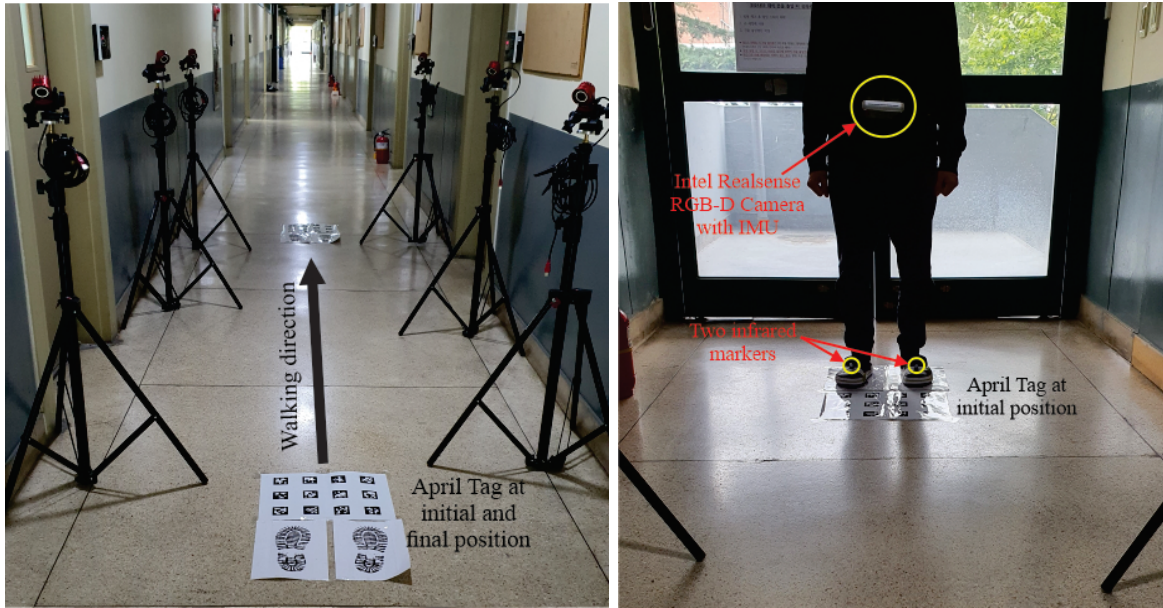


FIGURE 5.1: Participant with equipped camera system doing the experiments.

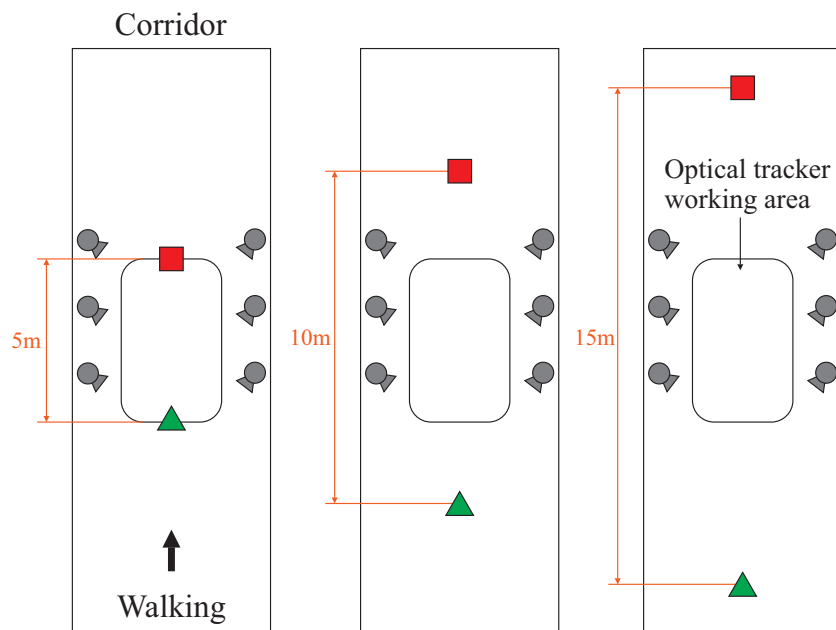


FIGURE 5.2: Experiment setup, volunteer walks three straight paths from green triangle to red square through the working range of the optical tracker system.

TABLE 5.1: Five Subjects Information

Volunteer	Age	Weight(kg)	Height(cm)
Range	26-35	55-75	161-185
Mean	30.2	62.2	170.2
Standard deviation	3.27	7.60	6.05

Five healthy persons are recruited to perform the data acquisition. Subjects information is given in Table 5.1.

In the experiment, volunteers are asked to walk in a straight line inside a long corridor. Each person walks in separated paths with walking distance of 5, 10, and 15 meters, five times each. At the starting and ending point of each walking path, two identical printed Apriltag of A3 size are placed with the same orientation. All walking paths are designed so that the tracking range is at the middle of each path as in Fig. 5.2.

5.2 Results

5.2.1 Visual Inertial Odometry results

Fig. 5.3 shows detected stance foot from point cloud depth data using foot detection algorithm in Section 3.3, where foot candidate points are selected using a simple threshold constraint of the height from the floor. Candidate points are grouped to defined foot edge. An ellipse approximation algorithm is used to fit the detected foot edge. Foot position in the camera coordinate is then calculated from ellipse position in the RGB images.

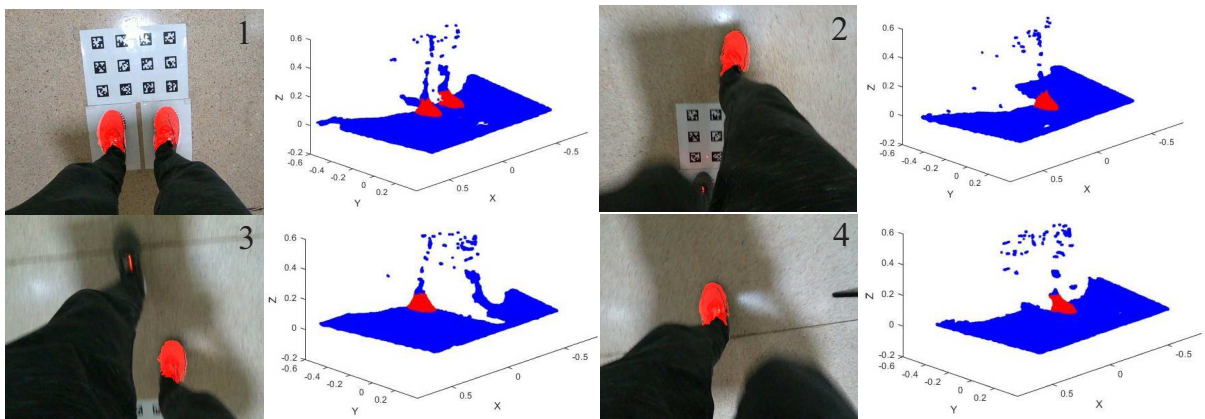


FIGURE 5.3: Detected stance foot from point cloud data and masked in RGB image: 1) initial states; 2) first detected right foot; 3) already detected right foot, used as landmark; 4) first detected left foot.

Fig. 5.4 shows a 16.5-meter-walking example of an estimated walking trajectory integrating the waist-mounted internal IMU data only without any measurement updating. It is clear that just integrating IMU data method leads to divergence.

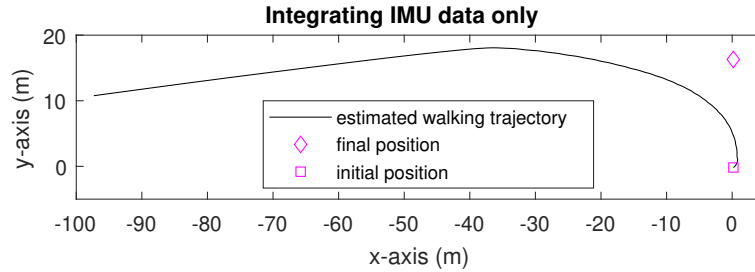


FIGURE 5.4: Estimated walking trajectory using integrating internal IMU data only, proposed filter and smoother.

From the proposed filter in Algorithm 1 and the smoothing algorithm, estimated walking trajectories are given in the Fig. 5.5 and Fig. 5.6. Left and right foot positions in world coordinate system are computed. The relative position, relative attitude from visual odometry and foot position as landmarks are used in the measurement updating equation. The estimated walking trajectory is improved and corrected in the smoothing algorithm. Note that the initial foot position is slightly negative since it is aligned with the origin of the calibrated optical tracker system.

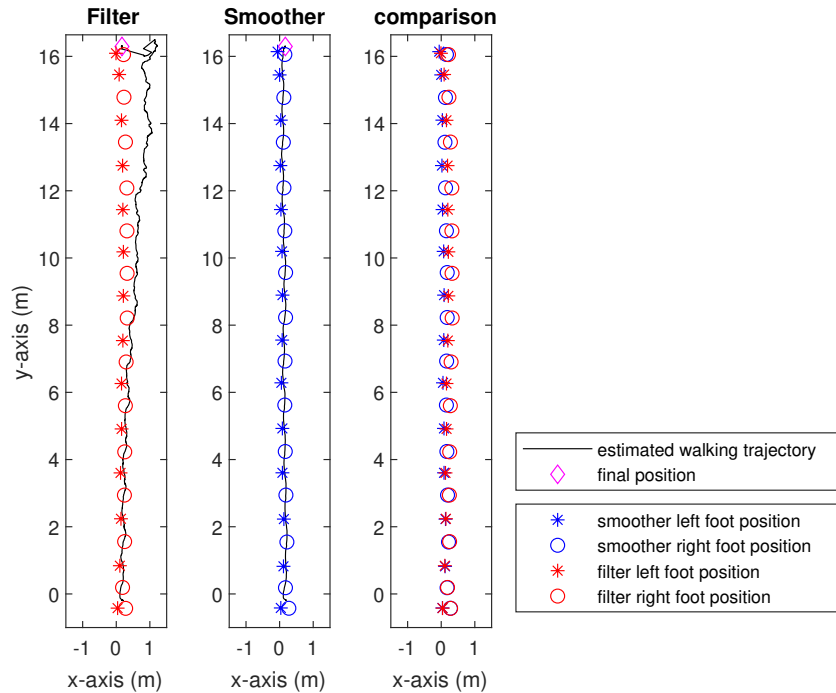


FIGURE 5.5: xy -plane estimated walking trajectory and foot position using proposed Kalman filter and smoother.

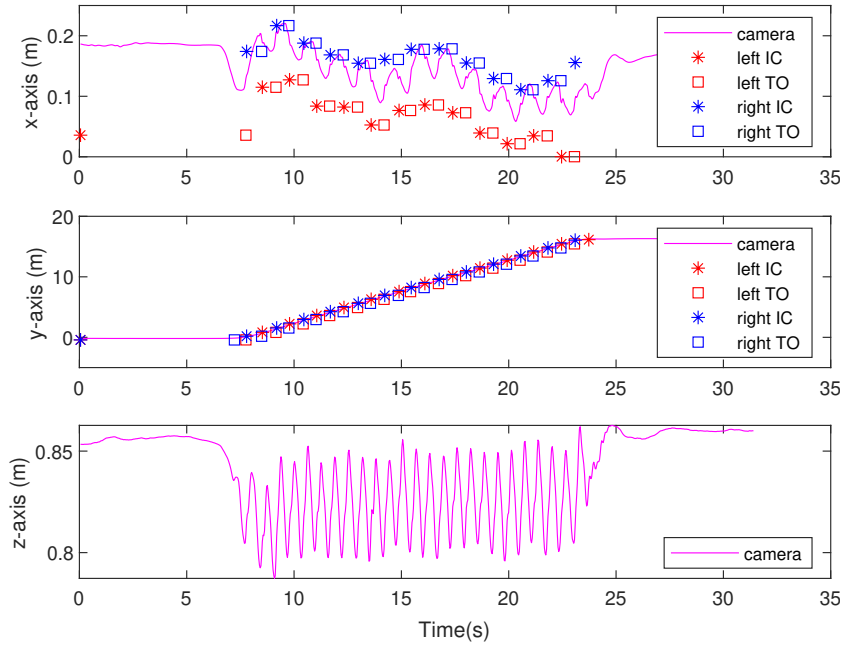


FIGURE 5.6: 3-axis estimated walking trajectory and foot position using proposed Kalman filter and smoother (IC: Initial Contact, TO: Toe Off).

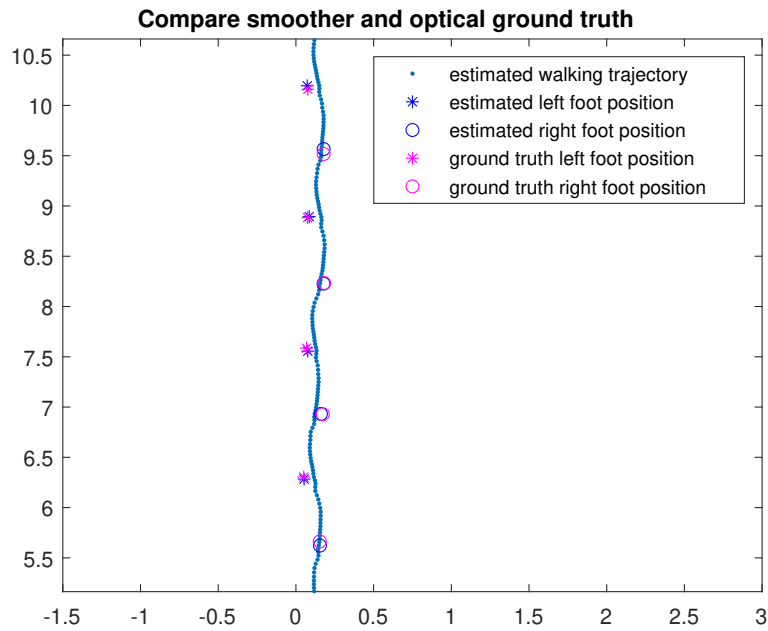


FIGURE 5.7: Compare estimated foot position with ground truth.

To evaluate the accuracy, foot positions are compared with ground truth from optical tracker system in Fig. 5.7. In each trial, three strides in 5-meter working space of optical tracking system are selected to provide ground truth data. In total, there are 5 (person) x 3 (path) x 5 (trial) x 3 (stride) x 2 (left,right) = 450 strides data. Since the difference in walking paths might cause different estimated results, the number of selected strides for calculating stride length is the same for the whole experiment, and average stride length error

over the walking paths for each volunteer is computed and given in Table 5.2. The overall root mean square errors (RMSE) is about 3.8 centimeters.

TABLE 5.2: Estimated stride length error (Unit: Meter).

ID	Left			Right		
	Max. error	RMSE	Std.	Max. error	RMSE	Std.
1	0.043	0.033	0.017	0.039	0.033	0.014
2	0.049	0.027	0.019	0.037	0.026	0.012
3	0.075	0.067	0.031	0.056	0.047	0.029
4	0.050	0.030	0.025	0.073	0.059	0.024
5	0.057	0.036	0.024	0.046	0.020	0.017

5.2.2 Deep learning-based foot trajectory estimation results

To evaluate deep learning-based foot detection algorithm, some evaluation metrics is used as follows:

Pixel Accuracy Pixel accuracy is the most basic metric which can be used to validate the results. Accuracy is obtained by taking the ratio of correctly classified pixels with respect to total pixels. The main disadvantage of using such a technique is the result might look good if one class overpowers the other. Say for example the background class covers 90% of the input image, we can get an accuracy of 90% by just classifying every pixel as background.

Intersection over Union (IoU) IoU is defined as the ratio of intersection of ground truth and predicted segmentation outputs over their union. If we are calculating for multiple classes, IoU of each class is calculated and their mean is taken. It is a better metric compared to pixel accuracy as if every pixel is given as background in a 2 class input the IOU value is $(90/100+0/100)/2$ i.e 45% IOU which gives a better representation as compared to 90% accuracy.

Frequency weighted IoU This is an extension over mean IOU which we discussed and is used to combat class imbalance. If one class dominates most part of the images in a dataset like for example background, it needs to be weighed down compared to other classes. Thus instead of taking the mean of all the class results, a weighted mean is taken based on the frequency of the class region in the dataset.

Fig. 5.8 shows the evaluation results of our foot detection network, while Fig. 5.9 shows an example of prediction output from the network.

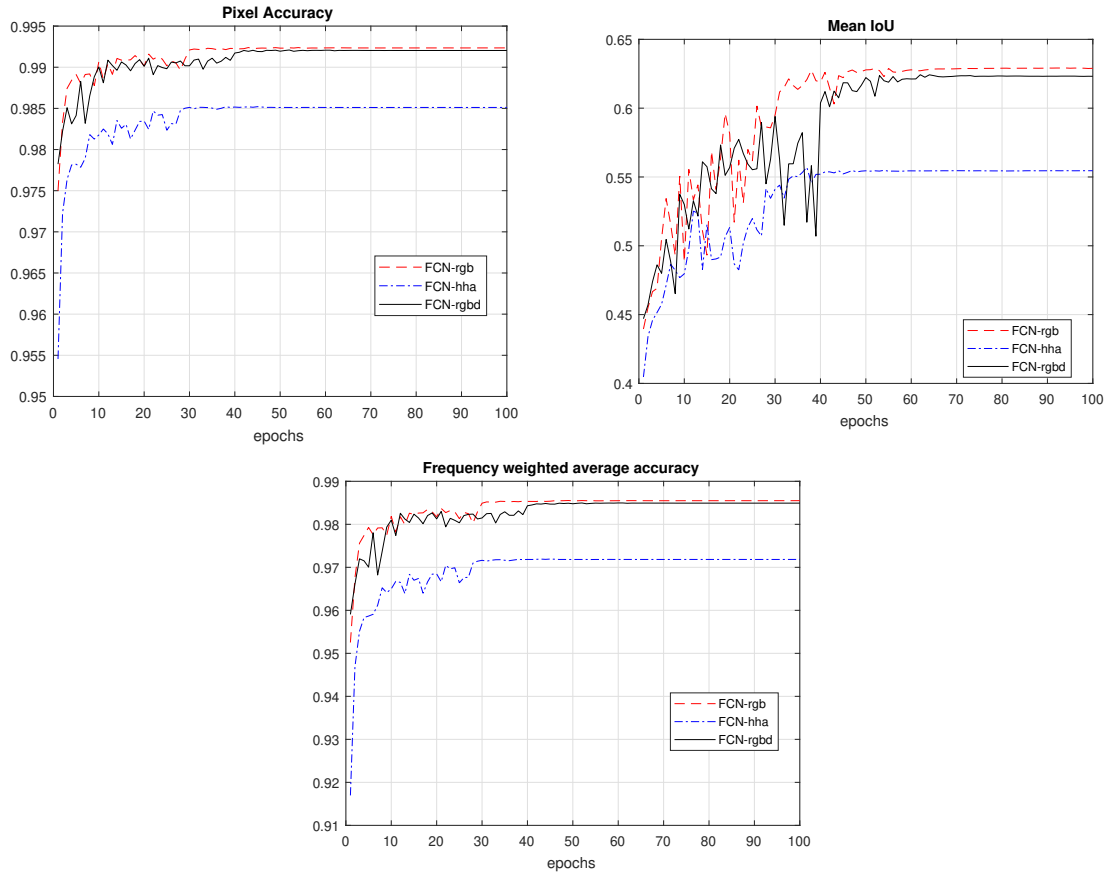


FIGURE 5.8: FCN evaluation results: 1) Pixel accuracy; 2: Mean IoU; 3: Frequency weighted average accuracy.

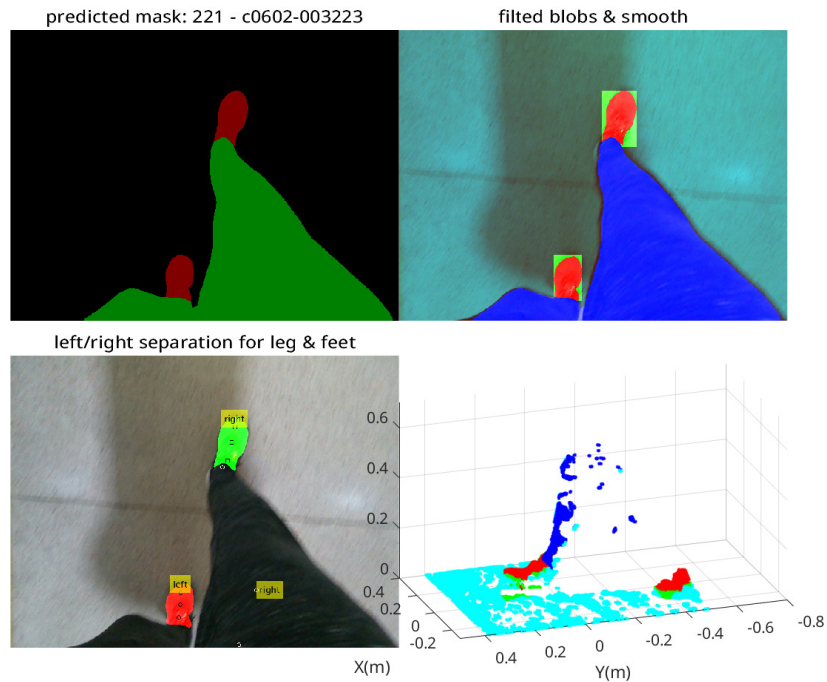


FIGURE 5.9: Deep learning-based foot detection results: prediction: 1) predicted classes from network; post-processing: 2) filtered blobs and smooth with depth data; 3) left and right feet separation; 4) 3d point cloud segmentation.

The post-processing step provides us the filtered blobs and smooth the blobs with constraints of depth information, and robust separation of left and right feet, even in the case that only one feet is visible from the image.

Once foot position in camera coordinate frame is derived, its position expressed in world coordinate frame can be calculated with the camera position and orientation. Fig. 5.10 presents three-axis estimated trajectories of both the camera and two foot. We can see that two foot trajectories still have missing data, due to the occlusion of the feet between Toe Off and Midswing.

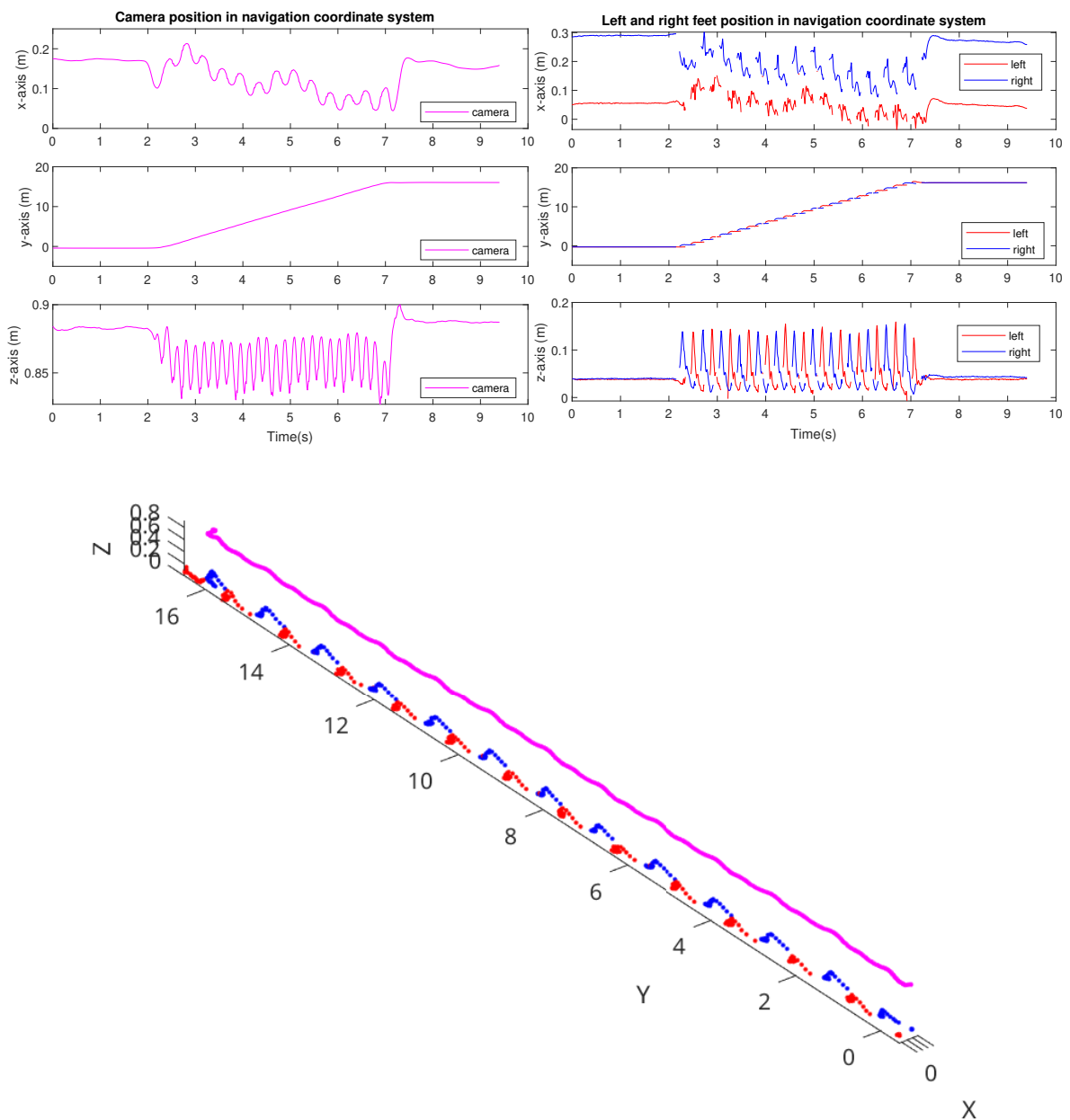


FIGURE 5.10: Three axis estimated trajectory of: 1) camera; 2) left and right feet; 3) 3d position of both camera and two foot.

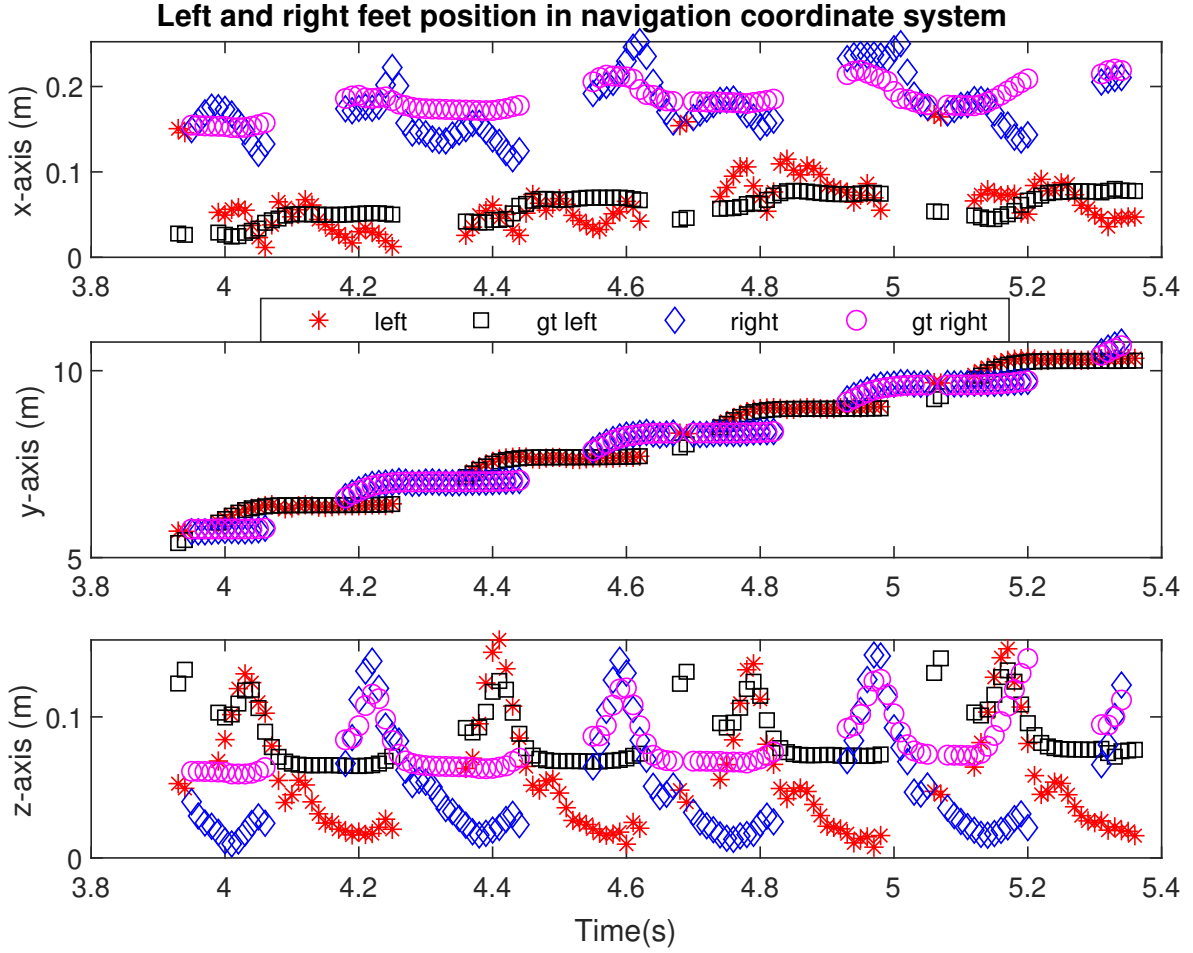


FIGURE 5.11: 3d position of estimated foot trajectory and ground truth.

The estimated dual foot trajectories are compared with the groundtruth in Fig. 5.11. Since the sampling rate of two systems are different, the groundtruth data is downsampled to 30Hz and both data are filtered to remove missing data. To quantitatively evaluate the quality of an estimated trajectory, we use the method in [66], where the estimated trajectory is aligned towards the groundtruth, and compute the distance errors. The norm of mean of absolute trajectory error (NMATE) within 5 meters of working space is given in Table 5.3.

TABLE 5.3: Estimated dual foot trajectories error (Unit: Meter).

ID	Left		Right	
	NMATE	Std.	NMATE	Std.
1	0.069	0.070	0.060	0.041
2	0.055	0.050	0.051	0.034
3	0.049	0.034	0.048	0.034
4	0.076	0.030	0.074	0.028
5	0.046	0.047	0.043	0.049

5.2.3 Human lowerlimb motion estimation results

GCN-based model is implemented using Pytorch, and we used ADAM [67] optimizer in the training process. The training rate is set to 0.0005 with a 0.96 decay every two epochs. The batch size is set to 32. Our model is trained for 50 epochs on a NVIDIA Geforce RTX3070 8GB. We use L-BFGS [68] optimizer with the learning rate of 2, maximum iteration is 25, tolerance change is $1e-6$ in the measurement optimization (4.7), which run faster than ADAM but still takes about 0.3 seconds on each frame. We predict 10 future poses from the past 10 poses with DCT encoding of the depth 20. The dual foot trajectories are reconstructed in Fig. 5.12.

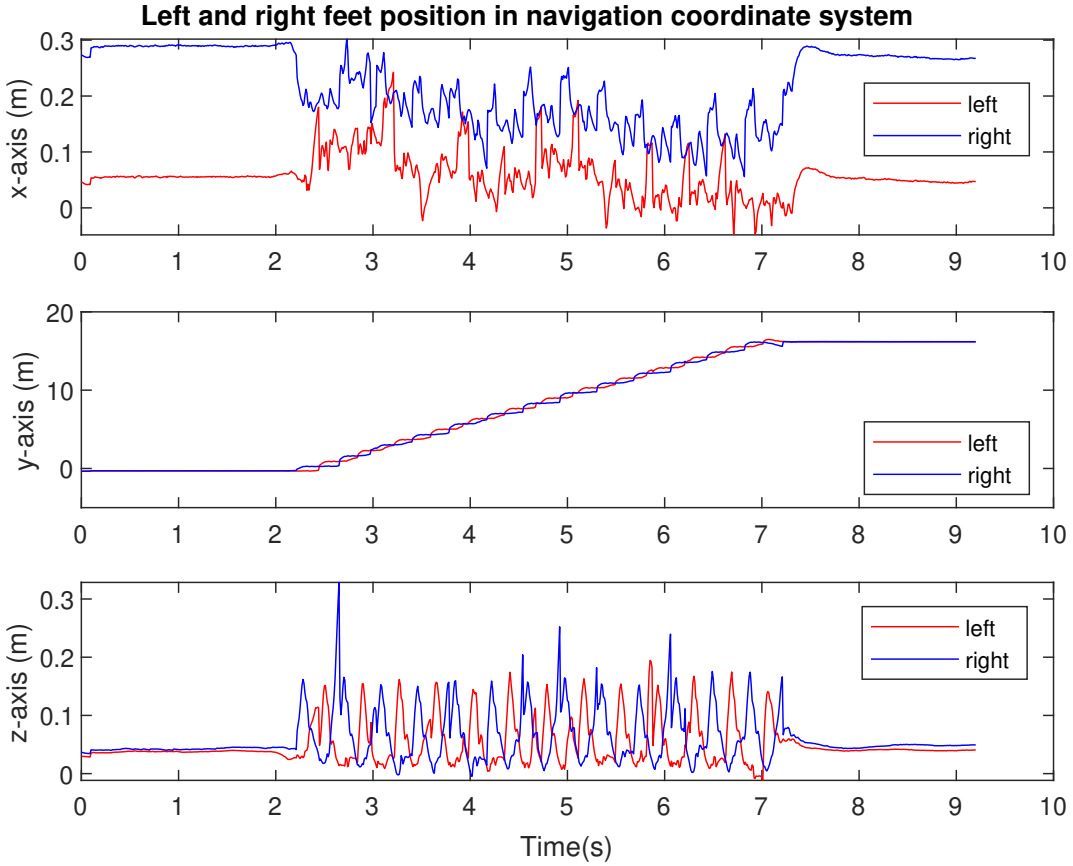


FIGURE 5.12: 3d position of GCN-based predicted left and right foot.

The predicted left feet trajectory comparing with estimated one from VIO algorithm is shown in Fig. 5.13, and with the ground truth is given in Fig. 5.14. We can easily see that the results are not very good because of the misalignment in x -axis and z -axis trajectories. This may be caused by the lack of foot joints in 7-joint lower-body model that leads to foot joint position is not coincide with ankle joint position. However, the y -axis trajectory shows the good prediction along the walking direction.

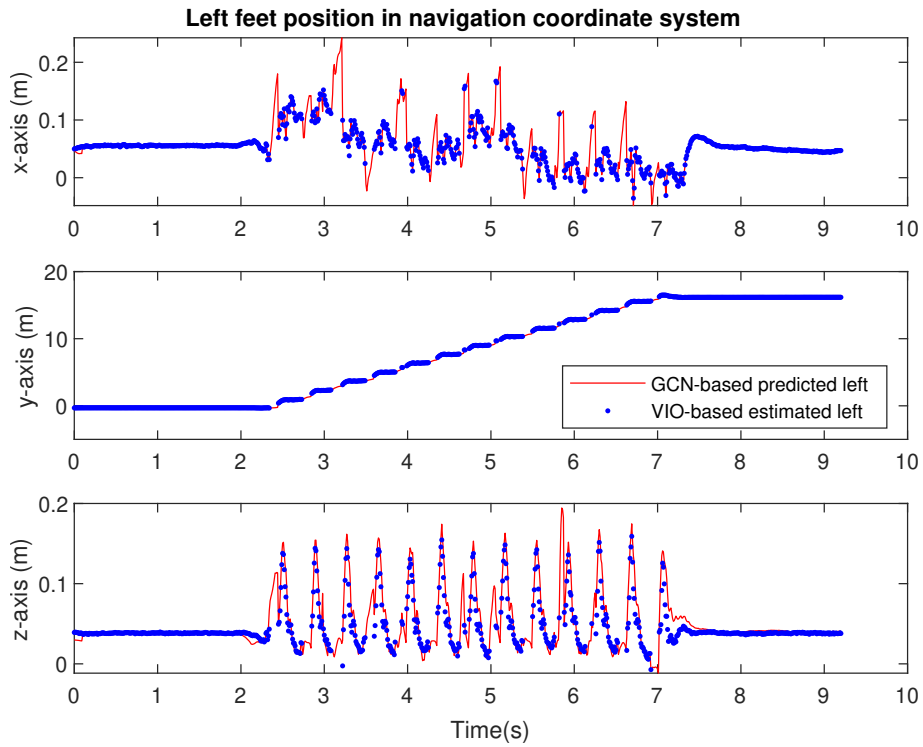


FIGURE 5.13: 3d position of GCN-based predicted and VIO-based estimated left foot trajectory.

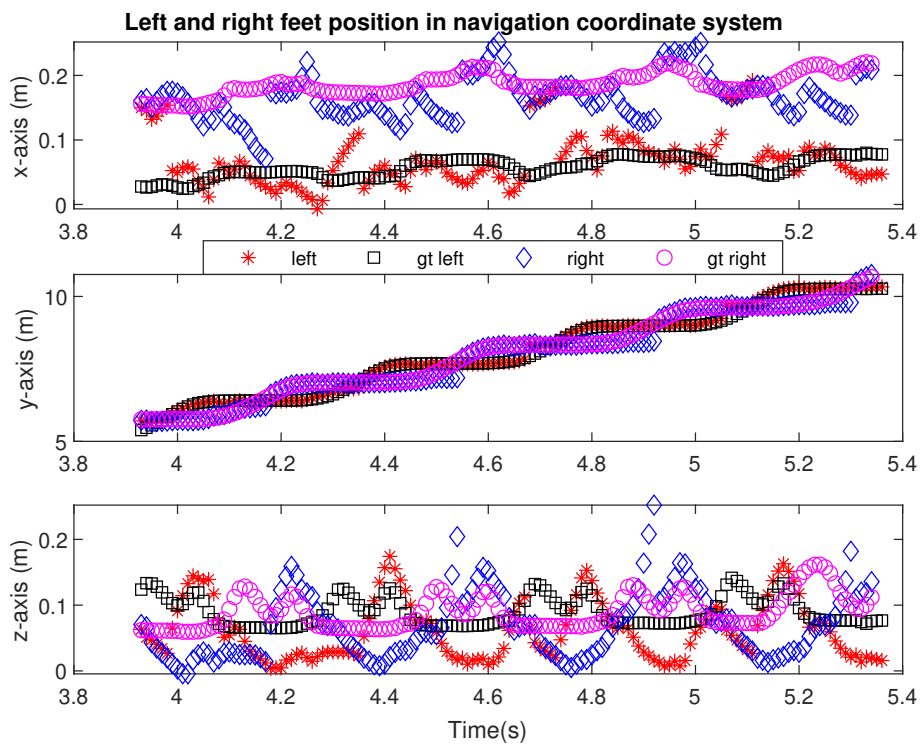


FIGURE 5.14: 3d position of GCN-based predicted and groundtruth of dual foot trajectory.

Chapter 6

Conclusions and Future works

6.1 Conclusions

A human motion estimation system used for tracking lower-body motion during walking activity with a single waist-mounted RGB-D camera and integrated inertial sensors has been the main topic of this dissertation. Firstly, with the achievement of applying standard indirect Kalman filter for waist-mounted inertial sensor in position and attitude estimation during walking, we are motivated to develop a novel navigation algorithm based on visual inertial odometry (VIO). The VIO algorithm is able to estimate relative pose, including position and attitude, of the camera from image sequence. These information are used as a measurement for the proposed filter. Moreover, from color and depth image data, the visible foot in stance phases can be detected using a threshold-based image processing framework. Due to the fact that position of the foot are unchanged during stance phases, they are used as landmarks for updating the measurements. A smoothing algorithm is proposed as linear optimization problem to reduce the estimation errors. The body walking trajectory and detected foot positions can be used to derive gait parameters such as walking stride length, step time, speed, etc. Secondly, a deep learning-based framework is proposed to extend foot detection algorithm as a semantic segmentation problem. Visible foot on the image can be detected, thus, 3-D dual foot trajectories during walking can be derived from the proposed filter results and the relative position of dual foot with respect to the camera. Thirdly, a Graph Convolutional Network (GCN) based model is proposed to predict pose from previous poses to tackle the problem of missing feet on the images. Complete foot trajectories are finally reconstructed with only a single waist-mounted RGB-D camera with integrated IMU.

Chapter 2 introduces the standard position and attitude estimation algorithm based on indirect Kalman filter for wearable inertial sensor data during walking. The method uses the inertial sensor data including accelerometer and gyroscope outputs to estimate attitude and position by double integrating method in time update process. To reduce the accumulation errors, we apply two measurement updating models: Gravity measurement update (GMU) and Zero-angular-rate update (ZARU). Due to the limitation of the algorithms for low-cost inertial sensors-based in a long walking distance or period, we propose a constrained optimization-based smoothing algorithm to improve the accuracy.

Chapter 3 presents the VIO filtering algorithm, where the filter state is extended to combine consecutive image sequences, that enable us to use the estimated relative position

and attitude from visual odometry algorithm as an measurement update in the filter. Color and depth image data are combined to detect the floor plane, measure the height information, and detect the candidate group of points that belongs to the stance feet. The foot is then detected by a simple ellipse approximation algorithm. Left and right feet are easily separated and their position in the camera and world coordinate frame can be computed from depth image data. We propose the measurement from Apriltag, which is put on the floor at the starting and ending location, to provide initial and final value for constraints of foot position and camera orientation. The truth that position of a stance feet is unchanged during stance phases enable us to use stance foot as the landmarks for updating the camera position in the filter. An smoothing-based algorithm is also used to improve the accuracy.

In chapter 4, we extend the foot detection framework using a deep learning-based method. A Fully Convolutional Network (FCN) is modified to be able to trained with both color and depth image from our created custom data set of first person view foot and legs images. The output of the network is an image whose each pixel indicates its class. Predicted image is then post-processed to calculate foot position in camera and world coordinate frames. A Graph Convolutional Network (GCN) is trained with selected walking data from AMASS dataset to propose a model for predicting human lower body pose from previous poses. The problem of missing feet on the image, which led to missing data on estimated foot trajectory, is now solved.

Chapter 5 shows the experimental setup and the results of estimated body trajectory, stance foot position, and the whole foot trajectory. The proposed method gives small errors in comparison with groundtruth from optical tracker system. Besides, the human poses can be reconstructed and visualized with an acceptable errors and processing time. Therefore, we conclude that the proposed methods are highly suitable for human motion estimation applications using a single wearable camera and inertial sensors.

6.2 Future works

The custom dataset is created with small-scale, related and fixed-size images. That can cause difficulty in improving foot detection results. The similarity of floor color and lighting condition can also effect the estimated relative attitude and position of the visual odometry algorithm. Besides, the proposed methods are used in an offline manner.

In future research, we will investigate how to extend the dataset scale and diversity, and automatically obtain the data label for foot detection framework. The visual inertial odometry algorithm can be integrated in SLAM framework for multi-floor indoor or outdoor navigation applications. The quality and processing time of the lower body pose prediction are needed to improve to work in real-time applications. Camera and root joint position and orientation need to be carefully calibrated. Other deep learning models can be considered in the future research.

Publications

- [1] Duc-Cong Dang and Young-Soo Suh, “Improved Single Inertial-Sensor-Based Attitude Estimation during Walking Using Velocity-Aided Observation,” *Sensors*, vol. 21, no. 10, pp. 3428, 2021.
- [2] Duc-Cong Dang and Young-Soo Suh, “Visual inertial odometry-based gait analysis using waist-attached RGB-D camera and inertial sensors,” *IEEE Sensors Journal*, vol. 23, no. 3, pp. 2539–2549, 2023.

References

- [1] J. K. Aggarwal and Q. Cai. Human motion analysis: a review. In *Proceedings IEEE Nonrigid and Articulated Motion Workshop*, pages 90–102, 1997.
- [2] Duane V Knudson and D Knudson. *Fundamentals of biomechanics*, volume 183. Springer, 2007.
- [3] S. Frenkel-Toledo, N. Giladi, C. Peretz, T. Herman, L. Gruendlinger, and JM. Hausdorff. Effect of gait speed on gait rhythmicity in parkinson’s disease: variability of stride time and swing time respond differently. *J Neuroeng Rehabil*, 2(23), 2005.
- [4] Farahiyah Jasni, Nur Azah Hamzaid, Tawfik Yahya Al-Nusairi, Nur Hidayah Mohd Yusof, Hanie Nadia Shasmin, and Siew-Cheok Ng. Feasibility of a gait phase identification tool for transfemoral amputees using piezoelectric- based in-socket sensory system. *IEEE Sensors Journal*, 19(15):6437–6444, 2019.
- [5] Ji Su Park, Chang Min Lee, Sang-Mo Koo, and Choong Hyun Kim. Gait phase detection using force sensing resistors. *IEEE Sensors Journal*, 20(12):6516–6523, 2020.
- [6] D. H. Sutherland. The evolution of clinical gait analysis: Part ii kinematics. *Gait Posture*, 16(2):159–179, 2002.
- [7] A. Mannini and AM. Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors (Basel)*, 10(2):1154–1175, 2010.
- [8] Rafael Caldas, Marion Mundt, Wolfgang Potthast, Fernando Buarque de Lima Neto, and Bernd Markert. A systematic review of gait analysis methods based on inertial sensors and adaptive algorithms. *Gait Posture*, 57:204–210, 2017.
- [9] Petraglia Federica, Scarcella Luca, Pedrazzi Giuseppe, Brancato Luigi, Puers Robert, Costantino Cosimo, and Brancato Luigi. Inertial sensors versus standard systems in gait

- analysis: a systematic review and meta-analysis. *European Journal Of Physical And Rehabilitation Medicine*, 55(2):268–280, 2019.
- [10] MP Kadaba, HK Ramakrishnan, and ME Wootten. Measurement of lower extremity kinematics during level walking. *J Orthop Res.*, 8(3):383–392, 1990.
- [11] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2001.
- [12] Jiwen Lu, Gang Wang, and Pierre Moulin. Human identity and gender recognition from gait sequences with arbitrary walking directions. *IEEE Transactions on Information Forensics and Security*, 9(1):51–61, 2014.
- [13] X. Gu, F. Deligianni, B. Lo, W. Chen, and G.Z. Yang. Markerless gait analysis based on a single rgb camera. In *2018 IEEE 15th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pages 42–45, Nevada, USA, 2018.
- [14] AP Rocha, HMP Choupina, MDC Vilas-Boas, JM Fernandes, and JPS Cunha. System for automatic gait analysis based on a single rgb-d camera. *PLoS One*, 13(8), 2018.
- [15] A. Dubois and F. Charpillet. A gait analysis method based on a depth camera for fall prevention. In *Annu Int Conf IEEE Eng Med Biol Soc*, pages 4515–4518, 2014.
- [16] Edouard Auvinet, Jean Meunier, and Franck Multon. Multiple depth cameras calibration and body volume reconstruction for gait analysis. In *2012 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA)*, pages 478–483, 2012.
- [17] Samantha Ng, Adel Fakhri, Adam Fourney, Pascal Poupart, and John Zelek. Towards a mobility diagnostic tool: Tracking rollator users’ leg pose with a monocular vision system. In *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1220–1225. IEEE, 2009.
- [18] Solenne Page, Maria M Martins, Ludovic Saint-Bauzel, Cristina P Santos, and Viviane Pasqui. Fast embedded feet pose estimation based on a depth camera for smart walker. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4224–4229. IEEE, 2015.

- [19] Richard Zhi-Ling Hu, Adam Hartfiel, James Tung, Adel Fakh, Jesse Hoey, and Pascal Poupart. 3d pose tracking of walker users' lower limb with a structured-light camera on a moving platform. In *CVPR 2011 WORKSHOPS*, pages 29–36. IEEE, 2011.
- [20] Takaaki Shiratori, Hyun Soo Park, Leonid Sigal, Yaser Sheikh, and Jessica K Hodgins. Motion capture from body-mounted cameras. In *ACM SIGGRAPH 2011 papers*, pages 1–10, 2011.
- [21] Hao Jiang and Kristen Grauman. Seeing invisible poses: Estimating 3d body pose from egocentric video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3501–3509. IEEE, 2017.
- [22] Weipeng Xu, Avishek Chatterjee, Michael Zollhoefer, Helge Rhodin, Pascal Fua, Hans-Peter Seidel, and Christian Theobalt. Mo 2 cap 2: Real-time mobile 3d motion capture with a cap-mounted fisheye camera. *IEEE transactions on visualization and computer graphics*, 25(5):2093–2101, 2019.
- [23] Grégory Rogez, James S Supancic, and Deva Ramanan. First-person pose recognition using egocentric workspaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4325–4333, 2015.
- [24] Franziska Mueller, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt. Real-time hand tracking under occlusion from an egocentric rgb-d sensor. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1154–1163, 2017.
- [25] Jack B Kuipers et al. *Quaternions and rotation sequences*, volume 66. Princeton university press Princeton, 1999.
- [26] David Titterton, John L Weston, and John Weston. *Strapdown inertial navigation technology*, volume 17. IET, 2004.
- [27] Glenn Creamer. Spacecraft attitude determination using gyros and quaternion measurements. *The Journal of the Astronautical Sciences*, 44(3):357–371, 1996.
- [28] Young Soo Suh. Orientation estimation using a quaternion-based indirect kalman filter with adaptive estimation of external acceleration. *IEEE Transactions on Instrumentation and Measurement*, 59(12):3296–3305, 2010.

- [29] Robertg Brown and Patrickyc Hwang. *Introduction to random signals and applied Kalman filtering*. New York, John Wiley & Sons, Inc., 1992. 512, 1992.
- [30] Young Soo Suh. Inertial sensor-based smoother for gait analysis. *Sensors*, 14(12):24338–24357, 2014.
- [31] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [32] Changdi Li, Lei Yu, and Shumin Fei. Real-time 3d motion tracking and reconstruction system using camera and imu sensors. *IEEE Sensors Journal*, 19(15):6460–6466, 2019.
- [33] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, 2004.
- [34] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad. An overview to visual odometry and visual slam: Applications to mobile robotics. *Intell Ind Syst*, (1):289–311, 2015.
- [35] F. Landis Markley and J. L. Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control*. Springer, New York, 2014.
- [36] Anastasios I. Mourikis and Stergios I. Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3565–3572, 2007.
- [37] Guoquan Huang. Visual-inertial navigation: A concise review. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9572–9582, 2019.
- [38] Christian Kerl, Jrgen Sturm, and Daniel Cremers. Robust odometry estimation for rgb-d cameras. In *2013 IEEE International Conference on Robotics and Automation*, pages 3748–3754, 2013.
- [39] Lina Yang, Yuchen Li, Xichun Li, Zuqiang Meng, and Huiwu Luo. Efficient plane extraction using normal estimation and ransac from 3d point cloud. *Computer Standards & Interfaces*, 82:103608, 2022.
- [40] I.W. Selesnick and C.S. Burrus. Generalized digital butterworth filter design. *IEEE Transactions on Signal Processing*, 46(6):1688–1694, 1998.

- [41] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *2011 IEEE International Conference on Robotics and Automation*, pages 3400–3407, 2011.
- [42] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.
- [43] Joan Sola. Simultaneous localization and mapping with the extended kalman filter, 2014. Last accessed 2021-10-29.
- [44] Yu-I Yang, Chih-Kai Yang, Xin-Lan Liao, and Chih-Hsing Chu. Virtual try-on of footwear in augmented reality using rgb-d cameras. In *2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1072–1076, 2015.
- [45] P. Eisert, P. Fechteler, and J. Rurainsky. 3-d tracking of shoes for virtual mirror applications. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6, 2008.
- [46] David Murphy. Vykings launches 'magic mirror' virtual footwear try-on solution, 2023.
- [47] Cristina Manresa, Javier Varona, Ramon Mas, and Francisco J Perales. Hand tracking and gesture recognition for human-computer interaction. *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, 5(3):96–104, 2005.
- [48] Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, et al. Megatrack: monochrome egocentric articulated hand-tracking for virtual reality. *ACM Transactions on Graphics (ToG)*, 39(4):87–1, 2020.
- [49] Mina Nouredanesh, Aaron W Li, Alan Godfrey, Jesse Hoey, and James Tung. Chasing feet in the wild: a proposed egocentric motion-aware gait assessment tool. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018.
- [50] M. Everingham. The pascal visual object classes challenge 2010 (voc2010) results, 2010.
- [51] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015.

- [52] Saurabh Gupta, Ross Girshick, Pablo Arbeliz, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation, 2014.
- [53] Kentaro Wada. Image polygonal annotation with python, 2016.
- [54] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollr. Focal loss for dense object detection, 2018.
- [55] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks, 2017.
- [56] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics, 2015.
- [57] Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs, 2016.
- [58] Judith Butepage, Michael Black, Danica Kragic, and Hedvig Kjellstrom. Deep representation learning for human motion prediction and classification, 2017.
- [59] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics, 2018.
- [60] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction, 2020.
- [61] Yi-Lian Chen, I-Ju Yang, Li-Chen Fu, Jin-Shin Lai, Huey-Wen Liang, and Lu Lu. Imu-based estimation of lower limb motion trajectory with graph convolution network. *IEEE Sensors Journal*, 21(21):24549–24557, 2021.
- [62] Wei Mao, Miaomiao Liu, and Mathieu Salzmann. History repeats itself: Human motion prediction via motion attention, 2020.
- [63] Junfa Liu, Juan Rojas, Yihui Li, Zhijun Liang, Yisheng Guan, Ning Xi, and Haifei Zhu. A graph attention spatio-temporal convolutional network for 3d human pose estimation in video. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3374–3380, 2021.
- [64] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. Amass: Archive of motion capture as surface shapes, 2019.

- [65] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015.
- [66] Zichao Zhang and Davide Scaramuzza. A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7244–7251. IEEE, 2018.
- [67] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
- [68] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.