Direct Visual and Inertial Odometry for Monocular Mobile Platforms

Jianjun Gui

A thesis submitted for the degree of

Doctor of Philosophy

School of Computer Science and Electronic Engineering

University of Essex

August 2017

Abstract

Nowadays visual and inertial information is readily available from small mobile platforms, such as quadcopters. However, due to the limitation of onboard resource and capability, it is still a challenge to developing localisation and mapping estimation algorithms for small size mobile platforms.

Visual-based techniques for tracking or motion estimation related tasks have been developed abundantly, especially using interest points as features. However, such sparse feature based methods are quickly getting divergence, due to noise, partial occlusion or light condition variation in views. Only in recent years, direct visual based approaches, which densely, semi-densely or statistically use pixel information reveal significant improvement in algorithm robustness and stability.

On the other hand, inertial sensors measure the changes in angular velocity and linear acceleration, which can be further integrated to predict relative velocity, position and orientation for mobile platforms. In practical usage, the accumulated error from inertial sensors is often compensated by cameras, while the loss of agile egomotion from visual sensors can be compensated by inertial-based motion estimation.

Based on the complementary nature of visual and inertial information, in this research, we focus on how to use the direct visual based approaches to providing location information through a monocular camera, while fusing with the inertial information to enhance the robustness and accuracy. The proposed algorithms can be applied to practical datasets which are collected from mobile platforms.

Particularly, direct-based and mutual information based methods are explored in details. Two visual-inertial odometry algorithms are proposed in the framework of multi-state constraint Kalman filter. They are also tested with the real data from a flying robot in complex indoor and outdoor environments. The results show that the direct-based methods have the merits of robustness in image processing and accuracy in the case of moving along straight lines with a slight rotation. Furthermore, the visual and inertial fusion strategies are investigated to build their intrinsic links, then the improvement done by iterative steps in filtering propagation is proposed. As an addition, for experimental implementation, a self-made flying robot for data collection is also developed.

Acknowledgements

First and foremost I would like to express sincere gratitude to my supervisor Professor Dongbing Gu, for accepting me as a PhD student and for his continuous support at every stage of PhD. I believe I am privileged as his excellence in PhD supervision went beyond just expert technical advice. He instilled perseverance in me and showed me with his tireless effort that the hard-work and ambition are the best cure against the many insecurities that many PhD students suffer at some point during our studies.

I would also like to thank the members of my supervisory board, Professor John Qiang Gan and Professor Huosheng Hu. They provided me valuable advice for my research and guided me through a proper pathway throughout the whole PhD milestones. Being surrounded by extraordinarily bright people is the richest source of motivation.

In particular, I would like to thank Mr Robin Dowling and Mr Ian Dukes for their enthusiasm and patience when passing the knowledge on many aspects of systems engineering and servers infrastructure that have been so valuable to me. Also to Dr Ling Chen and Dr Sen Wang, for providing much precious technical inspiration that helped lay the foundations for my research and co-authoring of our publication. Also to Mr Ruihao Li, for always being there for the last three years. His enthusiasm and hard-work have been a source of motivation for the late stage of my PhD. Also thanks to all the undergraduate, MSc and visiting PhD students, Mr Ruijiao Li, Dr Sen Qiu, Dr Wenzhong Zha, Dr Chao Zhang, Dr Minhuan Guo, those at some point were part of Essex Robotics group. They are a source of new and fresh ideas and always willing to lend a hand with my data collection projects.

I also take this opportunity to sincerely acknowledge Professor Dewen Hu and Professor Yadong Liu for their help and guidance at the early stages of my research. Their advice helped me to seize the opportunity pursuing my PhD abroad.

University of Essex exempted me from the tuition fee and Chinese Scholarship Council supported my living expenses for my first three years. It was a great honour for me to have this scholarship. On the personal side, I am most grateful to my parents and aunties' family, for teaching me the values to achieve success in life.

Their integrity, hard-work and love for education have been important pillars of my PhD, adding the motivation, strength and endurance that is needed to accomplish a PhD.

Finally, I would like to express my gratitude to all my friends that have been supportive throughout the PhD and the writing of this thesis. Special thanks to those friends who also are (or were) PhD students once. They are the best counselling service that can offer to a doctoral student. A continuous source of support and well-being recipes: Liangxiao Zhu, Yizhou Tan, Huangchao Yu, Qi Zhu, Zhang Qi, Xiayu Zhu, Yu Zhang, Ruihang Yu, Mao Wang, Liang Zhu and Mingjing Xu.

Thank you very much indeed to all of you!

Contents

Nomenclature

Acronyms

2D	2 Dimensional
3D	3 Dimensional
DoF	Degree of Freedom
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features
FAST	Features from Accelerated Segment Test
BRISK	Binary Robust Invariant Scalable Keypoints
ORB	Oriented FAST and Rotated BRIEF
SSD	Sum of Squared Differences
NCC	Normalized Cross Correlation
ZNCC	Zero mean-Normalized Cross-Correlation
RMSE	Root Mean Square Error
SAD	Sum of Absolute Differences
PDF	Probability Density Function
MI	Mutual Information
RANSAC	RANdom SAmple Consensus
DoG	Difference of Gaussian

LoG	Laplacian of Gaussian
SVD	Singular Value Decomposition
PCA	Principal Component Analysis
SO(3)	Special Orthogonal group
SE(3)	Special Euclidean group
so(3)	Lie algebra of SO(3)
se(3)	Lie algebra of SE(3)
DSO	Direct Sparse Odometry
PTAM	Parallel Tracking and Mapping
SLAM	Simultaneous Localization and Mapping
VO	Visual Odometry
VIO	Visual Inertial Odometry
SfM	Structure from Motion
VS	Visual Servoing
BA	Bundle Adjustment
ICP	Iterative Closest Point
IMRP	Iterative Matching Range Point
IDC	Iterative Dual Correspondence
KF	Kalman Filter
EKF	Extended Kalman Filter
PF	Partical Filter
MSCKF	Multi State Constraint Kalman Filter
EIF	Extended Information Filter
UKF	Unscented Kalman Filter

MAV	Micro Aerial Vehicle
UAV	Unmanned Aerial Vehicle
MEMS	Micro Electro Mechanical Systems
LiDAR	Light Detection and Ranging
AHRS	Attitude Heading Reference System
GNSS	Global Navigation Satellite System
IMU	Inertial Measurement Unit
GPS	Global Position System
CCD	Charge Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
VINS	Visual Inertial Systems
GPU	Graphics Processing Unit
PID	Proportional Integral Derivative
PWM	Pulse Width Modulation

Conventions

$\mathbb{R}$,$\mathbb{C}$	The sets of real numbers and complex numbers
$\mathbb{Z}$,$\mathbb{N}$	The sets of integers and natural numbers
$\mathbb{R}^n$	The space of all n-tuples of real numbers
$\mathcal{W}$	World frame
$\mathcal{C}$	Camera frame
$\mathcal{I}$	Inertial frame
$\mathcal{P}$	Pixel frame
a	Lower case letters indicate scalars
$\boldsymbol{a}$	Bold letters indicate vectors

a_i	The i^{th} element of the vector $\boldsymbol{a}$
a_x,a_y,a_z	Cartesian components of the vector $\boldsymbol{a}$
$\boldsymbol{A}$	Upper case bold letters indicate matrices
a_{ij}	Component of matrix $\boldsymbol{A}$ at row i and column j
$\bar{\boldsymbol{a}}$	The mean or expected value of $\boldsymbol{a}$
$\hat{\boldsymbol{a}}$	An estimation of the random variable $\boldsymbol{a}$
$\boldsymbol{I}$, $\boldsymbol{I}_n$	Identity matrix, identity matrix of order n
$\mathbf{0}$, $\mathbf{0}_n$	Zero vector or matrix, zero matrix of order n
$\boldsymbol{A}^T$, $\boldsymbol{a}^T$	Transpose of matrix $\boldsymbol{A}$ and vetor $\boldsymbol{a}$
$tr(\boldsymbol{A})$	Trace of matrix $\boldsymbol{A}$
$\diagdown(\boldsymbol{a})$	Diagonal matrix with coefficient vector $\boldsymbol{a}$

Vision

$\mathbf{O}$	Center of perspective of the camera
$\mathbf{u} = (u, v)^T$	Image point coordinates in pixel frame
$\mathbf{c} = (u_0, v_0)^T$	Principal point in pixel frame
$\mathbf{x}^{\mathbf{p}}_{\mathcal{W}} = (x^{\mathbf{p}}_{\mathcal{W}}, y^{\mathbf{p}}_{\mathcal{W}}, z^{\mathbf{p}}_{\mathcal{W}})^T$	A generic point $\mathbf{p}$ coordinates in world frame
$\mathbf{x}^{\mathbf{p}}_{\mathcal{C}} = (x^{\mathbf{p}}_{\mathcal{C}}, y^{\mathbf{p}}_{\mathcal{C}}, z^{\mathbf{p}}_{\mathcal{C}})^T$	A generic point $\mathbf{p}$ coordinates in camera frame
$\mathbf{K}$	Intrinsic matrix of a camera
$\mathbf{I}$	The image's photometrical properties
$\mathbf{I}(u, v)$	Image $\mathbf{I}$ at pixel of given coordinates $\mathbf{u}$
$\mathbf{M}$	Projection matrix

Kinematics

$\boldsymbol{q} = (w, x, y, z)^T$	A generic quaternion
$\boldsymbol{q}^*$	The conjugate of the quaternion $\boldsymbol{q}$

$\boldsymbol{e} = (\phi, \theta, \psi)^T$	The eular angles
$\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^T$	Rotation vector
$\lfloor \boldsymbol{\omega} \times \rfloor$	Skew-symmetric matrix of $\boldsymbol{\omega}$
$\boldsymbol{R}$	A generic rotation matrix
$\boldsymbol{t} = (t_x, t_y, t_z)^T$	Translation vector
$\boldsymbol{T}_{\mathcal{CW}} = \begin{bmatrix} \boldsymbol{R}_{\mathcal{CW}} & \boldsymbol{t}_{\mathcal{CW}} \\ \boldsymbol{0} & 1 \end{bmatrix}$	Homogeneous transformation matrix from frame $\mathcal{W}$ to $\mathcal{C}$
$\boldsymbol{\xi} = (\boldsymbol{\omega}, \boldsymbol{v})^T$	Minimal representation of Pose parameters $\boldsymbol{T} = \exp(\boldsymbol{\xi})$

Filtering

$\boldsymbol{x}_k$	State vector of the system at time step k
$\boldsymbol{z}_k$	Measurements vector at time step k
$\boldsymbol{u}_k$	Input vector at time step k
$\boldsymbol{w}_k$	Process noise at time step k
$\boldsymbol{v}_k$	Measurements noise at time step k
$\boldsymbol{f}(\cdot)$	Transition function
$\boldsymbol{h}(\cdot)$	Observation function
$\dot{\boldsymbol{f}}$	Derivative of $\boldsymbol{f}$
$\boldsymbol{F}_k$	Transition matrix at time step k
$\boldsymbol{H}_k$	Observation matrix at time step k
$\boldsymbol{P}_k$	Covariance of the state matrix at time step k
$\boldsymbol{S}_k$	Innovation prediction matrix at time step k
$\boldsymbol{Q}_k$	Process noise covariance matrix at time step k
$\boldsymbol{K}_k$	Kalman gain matrix at time step k
$\boldsymbol{R}_k$	Measurements noise covariance matrix at time step k

Chapter 1

Introduction

1.1 Overview of Research Problem

In recent years, the massive expansion of robots has ushered a new trend in both industrial and academic research. Smart devices, intelligent systems and enormous amounts of data have gradually become indivisible parts of our daily life. Even small mobile platforms nowadays can handle volume information and perform multiple tasks. Micro Aerial Vehicles (MAVs) among them are the prominent representation, which are ideal platforms for a broad range of applications in indoor and outdoor environments.

We have seen various successful commercialised products of autonomous MAVs in outdoor applications such as aerial photography. This experience could provide us a different view towards our world and life. Due to the superior nature in size, mobility and rapid response, such mobile platforms are also popularly adopted in missions including resource monitoring, information gathering, mapping, search and rescue, especially in the dangerous and inaccessible environment for human or ground vehicles.

However, building an intelligent mobile platform is still a challenging task when considering all the size, weight and power constraints as well as minimising human interaction. To achieve the ability of full autonomy, mobile platforms should integrate the capability of perception, motion estimation, planning, control, situational awareness and decision making. Among these, *motion estimation* is the foremost critical component. Actually, for all mobile robots, knowing "*where am I*" is always the prerequisite condition. Here, the "*where*" is not just limited to the location of mobile platform itself but includes the motion state, relative position and translation state to previous states, the world or other objects.

In this thesis, we look into motion estimation methodologies using monocular camera and IMU, both of which are fused in a modified filter. Specifically, two direct Visual and Inertial Odometry (VIO) methods are proposed. They will be applied and tested in practical datasets containing complex outdoor and indoor environments with the goal of achieving higher accuracy in trajectory estimation and more robustness in visual perception. Additionally, as our research is tightly related to practical applications, a quadcopter mobile platform will also be built for data collection and algorithms testing.

1.2 Research Motivation

1.2.1 GPS-denied application

A commonly used method for motion estimation is through using the Global Positioning System (GPS), which consists of three major segments (space, control, and user). In space, the GPS satellites have nearly circular orbits with an altitude of about 20200km above the earth. The present nominal constellation consists of 24 operational satellites deployed in six evenly spaced planes with an inclination of 55° and with four satellites in each plane [1]. The control segment is responsible for monitoring the status of the space segment, which consists of six monitor stations and a master station around the world. The user segment includes antennas and receivers that is commonly used in localisation application. Each satellite continually broadcasts its position, velocity, clock error and health status. By capturing this data from three or more satellites simultaneously, the traditional receiver can estimate the position, velocity and time for the user, while for some advanced receiver may use additional navigation solution to aid in tracking weak satellite signals [2]. An traditional GPS receiver commonly used in mobile platforms can be seen in figure 1.1(h).

In normal civilian applications, only the limited accuracy provided by GPS standard positioning service is available by standalone users. Its positional accuracy is about 10 to 20m at 95% probability level [1]. Precise positioning service is available only to users authorized by the U.S government [2]. On the other hand, although GPS receivers operate passively, the GPS application is limited due to the obstructed signal path between the receiver and a satellite. For many applications of small mobile platforms, the tasks could be performed in the environments such as indoors, forests, underground, underwater, the vicinity of tall buildings, which

will lead to unstable GPS signal receiving and biased position estimation. In past decades, GPS-based navigation is mostly well-developed by the aerospace community [1, 3], motion estimation and autonomous navigation for small mobile platforms in GPS-denied environments have only gained popularity recently [4–7].

1.2.2 Defective robotic perception

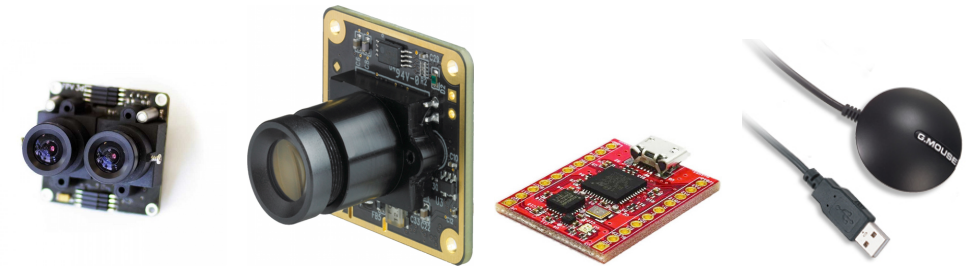
Our human beings and other animals can sense the world through our embodied visual, tactile and auditory modalities, and then interact with the environment effortlessly, although this process includes a lot of environmental noise and incomplete measurements [8]. Similarly, robots can perceive the environment through various sensors, including the types of contact, contact-less, active or passive. *Active sensors* function with emitting some forms of radiation or energy, which then can be reflected by scene structures, and detected by the sensor afterwards. A majority of active sensors are used for measuring the distance to an object. This can be achieved either through measuring the time-of-flight [9] or through triangulation technique in geometry [10]. Such sensors include ultrasonic range finder, infrared range finder, and LiDAR. *Passive sensors* function through purely observing the environment but do not emit any form of radiation or energy, which include cameras, Inertial Measurement Unit (IMU), pressure, radiation and contact sensors. Sensor examples are shown in figure 1.1.

Usually, one robot strives to use multiple sensors to minimise accumulated errors, increase response time or compensate with each other for hidden information which is limited in an individual sensor. However, in mobile platforms, due to the restrictions of low payload capacity, small physical size and complex application environments, large compromises over the number and type of sensors must be carefully considered.

1.2.2.1 Inertial sensing drift

As a proprioceptive sensor in mobile platforms, IMUs are very commonly used to measure the changes in angular velocity and linear acceleration [11–14]. An common IMU example (*myAHRS+* [15]) for small mobile platform is shown in figure 1.1(g). Traditionally, IMUs utilise mechanical gyroscopes, while modern units use fibre optic technologies enabling a higher accuracy [16]. These units, however, weigh several kilogrammes and are expensive, which limits their applications in weight restricted mobile platforms. The recent breakthrough in Micro Electro Mechanical Systems (MEMS) technology has allowed accelerometers to be fabricated on the

(a) Ultrasonic range finder (b) Infrared range finder (c) LiDAR (d) Kinect



(e) Stereo camera (f) Monocular camera (g) IMU (h) GPS reciever

Figure 1.1: Examples of active and passive sensors. 1.1(a) to 1.1(d) in the first row are active sensors, and 1.1(e) to 1.1(h) in the second row are passive sensors.

chip at a fraction of the cost and weight, with only a small compromise in accuracy [17, 18].

Nevertheless, the core issue limiting IMU applications in motion estimation is its essential mechanism of drift. It is easy to compute the velocity and position of mobile platforms through integrating or double integrating the detected accelerations. However, if such calculations are corrupted by noise or bias, the estimation results will be divergent rapidly over time [14, 18]. In order to bound the drift, in practical usage, an IMU is usually combined with other exteroceptive sensors, like cameras.

1.2.2.2 Visual sensing blur

Traditionally, commercial board cameras are physically small, inexpensive and low power consuming. Therefore they are the ideal components in embedded applications for mobile platforms, such as smart phones and MAVs. Within a camera, Charge Coupled Device (CCD) or Complementary Metal Oxide Semiconductor (CMOS) is used as the photosensitive sensor, which works as an optical-electric transducer, receiving light ray then converting it as digit intensity value. This mechanism is equivalent to animal visual processing system between eyes and brain, where cameras can be regarded as an interface between the world and supported

computation system. The appearance of a scene is recorded as an intensity matrix, then further analysed to get motion or location information.

However, for a single camera (monocular) device, like the one in figure 1.1(f), which consists of only one lens and one image sensor, the depth information of a scene or objects not can be detected directly from a single frame. In contrast to some bearing sensors such as time-of-flight cameras, which measure the distance using the time-of-flight principle [19], the depth of a pixel for monocular camera needs to be inferred from inter-frame motions. In some research, the depth estimation might be somewhat mitigated with a stereo (figure 1.1(e)) or structure light rig (figure 1.1(d)), but then the advantages of low power and compactness are lost.

On the other hand, the quality of images is not always satisfying for compacted hardware systems. Noise, occlusion and light condition would change in a series of agile manoeuvres. Low quality in images further leads to a rough estimation result in motion or location, then such algorithm is divergent rapidly. In most algorithms, low quality images are often tagged as outliers and dropped [20–22]. Then irregular gaps will exist in continuous image sequences. In practice, for a standard onboard camera, its sample rate is in $20 \sim 30$Hz; well-manufactured one may achieve higher performance but cost more in price as well [23, 24].

In order to improve the accuracy of estimation, the loss of motion dynamics between image samplings should be compensated by other fast-rate sensors, such as IMUs. Actually, the properties of visual and inertial sensors are complementary in terms of frequency response (see figure 1.3). The MEMS IMU can reach astonishing sampling rate with high accuracies [18]. Even for a common IMU product (figure 1.1(g)), the sampling rate can reach 100Hz [15], which can be adopted to provide more measurements of angular velocity and linear acceleration for a monocular mobile platform.

1.2.3 Limited motion estimation techniques

The combination of camera and IMU can help mobile platforms to learn the world through gathering raw visual and inertial information. However, without efficient and accurate estimation techniques, it is still impossible to achieve reasonable localisation estimates. Only in recent years, motion estimation, localisation and mapping methodologies are rapidly developed, especially based on visual techniques. These methods include Simultaneously Localisation and Mapping (SLAM) [25, 26], Visual Odometry (VO) [27–29], Structure from Motion (SfM) [30, 31], etc. From them,

the motion and pose of mobile platforms can be estimated, and the environmental structure can also be built in various appearance levels.

However, due to the limitation of small physical size, light weight and onboard computational resources, such estimation algorithms not can be applied in most mobile platforms directly. Some issues still need to be carefully considered, including how to trade off efficiency and accuracy in computation, how to use the raw visual information, and how to fuse inertial data with visual data.

1.2.3.1 Visual simultaneously localisation and mapping

In early robotics research, localisation and mapping were tackled independently, due to the fact of mutual dependency for both tasks [25]. A map can only be created when the robot's pose is known, while localisation needs to be confirmed with an accurate map representation. The research of SLAM tackles both tasks simultaneously.

SLAM algorithms based on various sensors have been developed for decades in the robotic community. Traditional SLAM approaches usually track the pose of a robot on the ground plane, and then the 2D maps are created by either representing a slice of the world or projecting 3D landmarks onto the ground [26]. The visual sensors applied in visual SLAM free the movement in height. Thus a full 3D representation of the environment can be built. On the other hand, in SLAM approaches, a loop detection process [32] is required to guarantee the platform is aware it has revisited a known location. The loop closure problem is commonly perceived as a core topic in SLAM, and it is often regarded as an important test when evaluating particular SLAM approaches. As mentioned in previous sections, the inherent noise in sensor measurements makes the estimated pose prone to drift over time during exploration. There will be a significant difference between the estimation and ground truth as time goes by. Once the platform returns to a previously visited location, the key of loop closure detection is to ensure a consistent map representation with the existence of such drift.

SLAM approaches seem quite excellent in providing both localisation and map at the same time. However, it not can ignore the extensive mapping information and the frequently retrieving operation while such algorithm is running. Especially when the map grows largely along time with the existence of accumulated error, the computational burden becomes significant. For resource limited mobile platforms, building and using a global map under strict time constraints is still a challenge with the commercial available sensors and computational power.

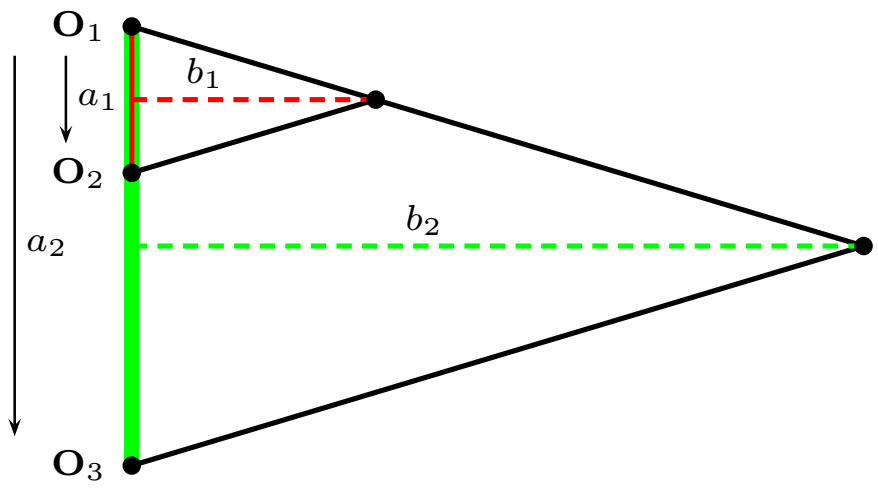


Figure 1.2: Scale ambiguity in monocular motion estimation. Only the proportion of a_1 and b_1 are kept during motion. It is impossible to distinguish whether the camera is moved from $\mathbf{O}_1$ to $\mathbf{O}_2$ or $\mathbf{O}_1$ to $\mathbf{O}_3$, since $\frac{a_1}{b_1} = \frac{a_2}{b_2}$.

1.2.3.2 Visual odometry

Incremental motion estimation with cameras, commonly known as visual odometry, is a fundamental component for almost all mobile platforms. Pairwise sensor measurements, such as consecutive images in monocular case, are usually the only information sources which are used to estimate the motion of the platform. Similar to wheel odometry, which incrementally calculates the movement of a vehicle by integrating the angles of wheel-turns over time, visual odometry incrementally estimates the pose of the vehicle by computing the motion from sequential image frames. Compared to wheel odometry [33], the advantage of VO is that the expected results will not be affected by wheel slip in uneven terrain or other adverse conditions. However, all visual only estimation techniques using monocular camera have the classical issue of scale ambiguity. Given that a space point is co-observed in two distinct camera frames, one can estimate the unknown depth using the triangulation technique. But it is impossible to measure absolute scale based on the monocular visual measurements only. A simple example is shown in figure 1.2. The real scale needs to be confirmed by another reference, since only the proportion of travelling distance and scene depth can be kept.

For another point of view, VO can be regarded as visual SLAM without cumbersome mapping and loop detection. Odometry techniques only aim to guarantee the local consistency of the estimated trajectory, unlike SLAM approaches which seek to get the global consistency of predicted trajectory and map. VO techniques remove

the loop detection for computing efficiency. Most historical data is marginalised or dropped out from lively tracking. In practice, the current state only maintains a few recent poses.

Although visual odometry methods try to enhance computational efficiency by means of sacrificing the integrity of global map, avoiding the revisit of previous map and performing the pose estimation locally, they are still facing the significant scale ambiguity problem. Visual only odometry, estimating the poses from the geometry relationship between frames, is also prone to drift over time. In this thesis, we will use the inertial-based estimation to ease the drift and also provide a scale reference for visual-based odometry.

1.3 Research Focuses

1.3.1 Visual and inertial fusion

Visual and inertial sensors are available for almost all mobile platforms. The properties of simple detection with rich information, make this combination popularly applied in motion estimation approaches. As mentioned above, the inertial sensor can compensate for the motion dynamics between visual samplings. Beyond this, they are also complementary in estimation. On the one hand, the fast drift issue in inertial-based estimation can be bounded by the visual-based method. On the other hand, the rich inertial dynamics can provide more prior information for visual-based estimation.

For the fusion methods of visual and inertial data, there are two categories, i.e. loosely and tightly coupled. Tightly coupled methods can jointly estimate robotic poses with the visual and inertial information [34–37]. The correlation between two sensors will be considered in calculation at every time step. However, due to the resource limitation of mobile platforms, it needs to trade off computational cost with algorithm efficiency. The loosely coupled approaches are more efficient than tightly coupled methods in general [11, 38–40]. They tackle the inertial and visual measurements separately. For example, in a motion estimation algorithm, the inertial measurement may first be processed to get an initial pose guess, then such estimate is used as a priori knowledge for visual-based computation. Visual and inertial parts run independently but correct latest results in turns. In the thesis, we will use the idea of loosely couple methods. Particularly, a filtering framework is used to tackle inertial and visual information in propagation and updating steps

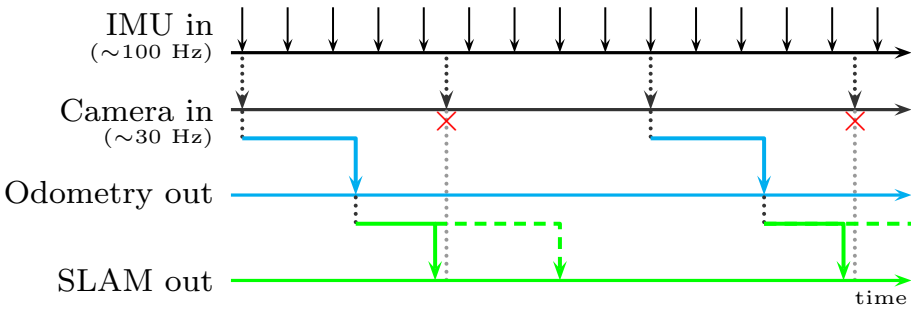


Figure 1.3: Conceptual rate comparison for IMU inputs, camera inputs, Odometry outputs and SLAM outputs. If the pose estimation task of odometry and other tasks of SLAM were done within the camera rate, the SLAM outputs can catch up the inputs of image frames. However, SLAM tasks often require more computational time for loop detection, mapping building or global optimisation. In order to achieve a real-time performance of the whole algorithm, tasks should be carefully tailored or arranged in multi-threads.

respectively. As inspired by iterative techniques from tightly coupled methods, the iterative solution is also studied.

1.3.2 Direct based methods

In sensory information processing, the methods of how to use raw data have a tremendous impact on the efficiency and robustness for algorithms. For inertial information, it is commonly adopted in an inertial-driven dynamic model instantly. While for visual information, methods are various due to the fast developing in computer vision community. Among them, some feature-based methods using SIFT, SURF and ORB features are the most popular [21, 23, 41, 42]. With the helping of sophisticated techniques in feature extraction, expression and matching, visual-based pose estimation can leverage such feature correspondents to build geometry relations between frames. However, it not can be neglected that the procedures of feature processing may consume vast computational resources, especially for large volume image data. Even for some particular cases using binary features, a previous training and loading procedure for feature vocabulary are still needed. Additionally, due to the presence of noise, occlusion and illumination variation in image quality, feature matching results will be largely affected.

Among the research on computer vision, there exists another category of the visual-based method, which directly uses the photometric information to build geometry constraints between frames [43–48]. In such methods, by removing the feature processing procedures, the computational time seems to be saved theoretically.

But tackling every pixel in a whole image range then finding a match is still a massive project. Take an image in VGA format for instance. It contains 640×480 pixels. Even if every pixel is perfectly matched without any outlier and location change, it still needs at least $307,200$ times search and matching steps for every pixel in a pair of images. But in practice, even a slight illumination variation may lead to a significant change in pixel intensity value. Direct-based methods are not processed the same as feature-based extracting and matching for sparse points. They are often using regional pixels. For instance, semi-dense methods [43] usually focus on prominent pixel regions with large gradient, which largely cut down the number of pixels in visual processing while keeping the significant information in an image.

Additionally, among direct-based methods, some methods are using mutual information theory [47, 49] to express the visual data. This idea of representing discrete images as random variables can use the statistic information from the whole image rather than sparse points. Therefore, due to the information for a large area is kept, the more robustness to the variation in full image can be expected. In this thesis, we will develop the visual measurement model using the ideas from direct-based methods. The measurement model will then be fused with the inertial-driven prediction in a filtering framework. Such novel measurement models and the corresponding VIO methods are novel in this research research area.

1.4 Summary

The research of robotic perception and motion estimation are the fundamental of application and further high-level tasks for mobile platforms. In practice, such platforms can sense the world through various sensors, but monocular visual and inertial combination is the most economic and efficient way. But due to the limitation of physical size, payload capacity and onboard computational resources, most of the current estimation methods not can be applied in small mobile platforms, such as quadcopters. Additionally, the essential accumulated error and image quality are easily affected by sensor noise, scene occlusion and illumination variations, which makes the motion estimates prone to divergent over time. In order to get a consistent motion estimation with acceptable performance in the complex indoor and outdoor environments for mobile platforms, we perform our research from the following three aspects: visual and inertial information fusion method in a loosely couple filtering framework, direct-based visual measurement models and practical mobile platform with sufficient sensory capacity and computing resources.

1.5 Contribution Overview

In this thesis, two novel visual-inertial odometry methods are proposed, and one mobile platform is built. The VIO algorithms can estimate the motion from inertial and visual data streams from practical indoor and outdoor environments. The quadcopter can manoeuvre freely and record the visual and inertial data at the same time. Specifically, the contributions can be depicted as follows:

First of all, we review the state-of-the-art research from the perspective of feature-based, direct-based, and visual-inertial fusion methods. The feature-based methods are commonly used in computer vision and robotic communities. They measure the world through visual geometrical constraints, and the visual measurement model is often expressed as the re-projection error of sparse points. The direct-based methods only become popular as the increasing of computing capability in recent years. The way to use the photometric information can make the visual measurement more robust to the environmental variation. There are two branches of visual-inertial fusion methods, i.e. tightly and loosely coupled methods. As the names indicate, they are different by whether using covariance terms between visual and inertial measurements. The loosely coupled filtering based methods has the merits of less computation, while the tightly coupled are more accuracy in results.

Secondly, a novel direct-based VIO algorithm using a multi-state constraint Kalman filter framework is proposed. In this approach, an inertial-driven dynamic model is applied in the propagation step of the filter, and the direct-based visual measurement model is used to update the state. The salient aspect of multi-state constraint Kalman filter is the state augmentation and removal scheme, where the length of the state vector can be modified. In the direct-based measurement model, scattered pixel patches are extracted and matched from image regions with the large gradient. Different from feature-based methods, the patch location and pattern are irregular. Thus more significant visual information can be adopted. Additionally, we analyse the intrinsic links between various estimation methods. Specifically, the fact is presented that adding an iterative process in the filtering framework is equivalent to the optimisation based estimation under the view of likelihood maximisation.

Thirdly, a novel visual measurement model for filtering based method is proposed using the theory of probability and information entropy. The values of mutual information between consecutive images can reflect their similarity. The most significant character of applying MI in visual data is increasing algorithm robustness to the case of noise, occlusion, and illumination variation. We present the details of derivation

process of the MI-based visual measurement model, where the pose is concisely formulated as the minimal expression using the knowledge of Lie algebra. Additionally, following the analysis of the iterative process in a filtering framework, we propose an iterative equation based on the Levenberg-Marquardt solution, where the Hessian matrix of MI with respect to the pose is also presented.

Finally, we compare some commercial products of visual and inertial sensors according to our system requirement, then build a quadcopter as a data collection platform. The major components and practical experience are introduced. Our self-made quadcopter has high compatibility with both software and hardware. On the one hand, the platform can enjoy various supportive resources from the developing community by using an elite Ubuntu system. Additionally, other sensors can connect to the system through universal USB ports. The visual and inertial data stream can be stored lively in memory with a unified time line. The simulations in ROS and Matlab can provide us helpful guidance in practical operations.

Publications

- **J.Gui**, D.Gu, S.Wang, & H.Hu, A Review of Visual Inertial Odometry from Filtering and Optimisation Perspectives, Advanced Robotics, 29(20), 1289-1301.
- **J.Gui**, D.Gu & H.Hu, Pose Estimation Using Visual Entropy, In Automation and Computing (ICAC), 2015 21st International Conference on (pp. 216-221). IEEE.
- **J.Gui**, D.Gu & H.Hu, Robust Direct Visual Inertial Odometry via Entropy-based Relative Pose Estimation, In Mechatronics and Automation (ICMA), 2015 IEEE International Conference on (pp. 887-892). IEEE.
- **J.Gui** & D.Gu, A Direct Visual-Inertial Sensor Fusion Approach in Multi-State Constraint Kalman Filter, In Control Conference (CCC), 2015 34th Chinese (pp. 6105-6110). IEEE.
- R.Li, Q.Liu, **J.Gui**, D.Gu & H.Hu Indoor Relocalization in Challenging Environments With Dual-Stream Convolutional Neural Networks, IEEE Transactions on Automation Science and Engineering
- R.Li, Q.Liu, **J.Gui**, D.Gu & H.Hu Night-time indoor relocalization using depth image with Convolutional Neural Networks, In Automation and Computing (ICAC), 2016 22nd International Conference on (pp. 261-266). IEEE.
- R.Li, Q.Liu, **J.Gui**, D.Gu & H.Hu A novel RGB-D SLAM algorithm based on points and plane-patches, In Automation Science and Engineering (CASE), 2016 IEEE International Conference on, (pp. 1348-1353). IEEE.

1.6 Thesis Outline

The whole thesis has been structured in seven chapters. After a general introduction to the research problems and major contributions in chapter 1, a detailed background review concerning visual-inertial odometry and related work will be given in chapter 2. The technical preliminary regarding visual geometry, rigid body kinematics and computing frameworks of filtering and optimisation are introduced and discussed in chapter 3. In chapter 4, a novel direct visual measurement model fused with inertial information in a multi-state constraint Kalman filter will be presented. Furthermore, the intrinsic link between two major fusion method, i.e. filtering based and optimisation based approaches, will be analysed in an iterative filtering process. Our second VIO algorithm will be presented in chapter 5, where the mutual information based direct visual measurement model is applied. Additionally, an iterative solution will also be proposed in this chapter. As this study is tightly related to practice, in chapter 6 a hardware implementation for quadrotor system with a set of monocular camera and IMU will be illustrated, and the simulation for this platform will also be introduced. Finally, all the research work and future research directions will be concluded in chapter 7.

Chapter 2

Background and Literature Review

In the previous chapter, we introduced the characters of visual-inertial sensing methods and the research problem of motion estimation. In this chapter, we will firstly review the processing of traditional feature-based techniques. Then starting from the topic of structure from motion, the feature-based SLAM approaches will be categorised by their calculation frameworks, i.e. filtering, optimisation and graph-based. The visual odometry methods as a tailored method from SLAM will also be reviewed individually. Some representative works in direct-based visual estimation approaches will be discussed to showcase a different methodology in dealing with visual information. Furthermore, the state-of-the-art research on visual and inertial fusion will also be shown. In our study, the visual-inertial odometry will be proposed by adopting the merit of combining visual with the inertial information in a modified filtering framework and using the direct-based measurement models.

2.1 Feature based Visual SLAM and Odometry

The fundamental SLAM problem has been the hotspot of robotic research for decades. Its basic tasks comprise two problems: localisation and mapping. To solve a localisation problem using only onboard sensors, the referenced map must be available, while to construct a map, the current relative location should be known. By using advanced research, like feature matching techniques and loop closure detection, it seems possible to estimate a reconstructed scene together with series of camera poses from only a sequence of images.

The aim for visual SLAM is to estimate the pose for camera motion while building

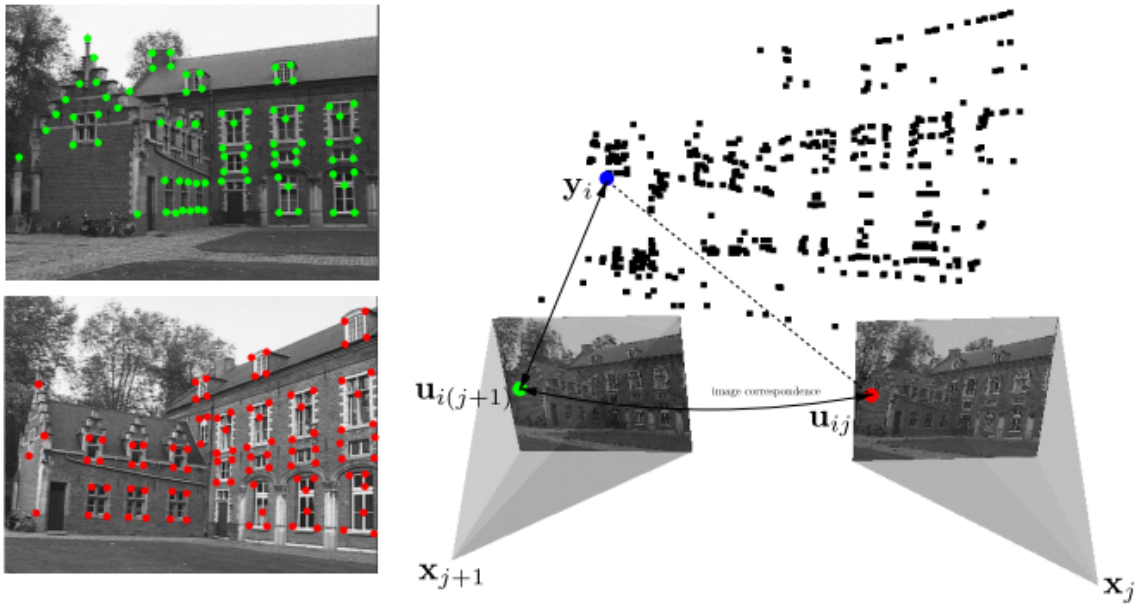


Figure 2.1: A sparse feature-based visual SLAM adapted from the work of [50]. Joint estimation of the camera poses $\mathbf{x}_j$, $\mathbf{x}_{j+1}$ and point cloud structure $\mathbf{y}_i$, is performed using global optimisation, minimising the distance between predicted image points $\mathbf{u}_{ij}$ and observations $\mathbf{u}_{i(j+1)}$ across two frames.

up a map to represent the scene. For an intuitive understanding, here we give an example using sparse features, as shown in figure 2.1. In this visual SLAM case, a scene consisting of N 3D points can be observed by M input images which are taken from a single camera but different locations. In this section, we will start from looking at most used methods in traditional feature-based processing, then sequentially review and discuss Structure from Motion (SfM), visual SLAM and Visual Odometry (VO).

2.1.1 Feature based visual measurements

How to use the visual information efficiently is always a supreme question in the research related to visual sensing. As a sophisticated image processing technique, geometrical features are commonly used in SfM, visual SLAM or VO approaches. These features can represent different types of information, e.g. points of interest [51], straight lines [52], segments [53, 54], ellipses [55] or contours [56, 57]. But they all need to extract such information individually for each new image in data stream. Traditionally, in order to be available in algorithms, a feature must go through the process of detection, description and matching.

2.1.1.1 Feature detection

Usually, without specific indication, geometrical features refer to local ones, where the image is represented by more than one multi-dimensional feature vectors (*feature descriptors*). They are created on local structures and textures from a set of salient regions within the images. Some of them are invariant to a small viewpoint and illumination variation. In the literature, various types of point features have been proposed to detect salient points (*keypoints*, or *interest points*) in images. Some of them permit matching even in the presence of large scale and orientation changes.

Here we only present some representations which are popular used. The most primary detector is *Harris corner detector* [51], which uses the idea that corners can be regarded as the intersection of two edges. Thus a corner is located at the point where the directions of these two edges change. *FAST corner detector* [58] uses a circle of 16 pixels to classify whether a candidate point p is actually a corner. Each pixel in the circle is labelled with integer number from 1 to 16 clockwise as shown in figure 2.2. If a set of nearby pixels in the circle are all brighter than the intensity of candidate pixel over a threshold value or all darker than the intensity of candidate pixel under threshold value, then p is classified as a corner. The main advantage of this corner detector is its computational efficiency regarding time and resources. However, it is not invariant to scale changes and not robust to noise.

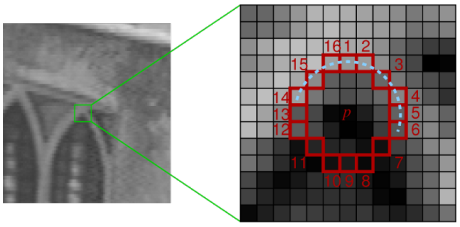


Figure 2.2: FAST corner detection (Taken from Edward Rosten's website).

To achieve scale invariance, *Laplacian of Gaussian* (LoG) [59] represents the scale space of the image by convolving the image with a variable scale Gaussian kernel. The point is obtained by searching for the location and scale extreme of the LoG function. LoG owns the merit of rotational invariance. However, it costs much in computation. To overcome the expensive computation problem, *Scale-Invariant Feature Transform* (SIFT) algorithm [60] approximate to the LoG with *Difference of Gaussian* (DoG), achieving more efficient performance.

Recently, *Oriented FAST and Rotated BRIEF* (ORB) feature detector [61] and *Binary Robust Invariant Scalable Keypoints* (BRISK) [62] become popular due to

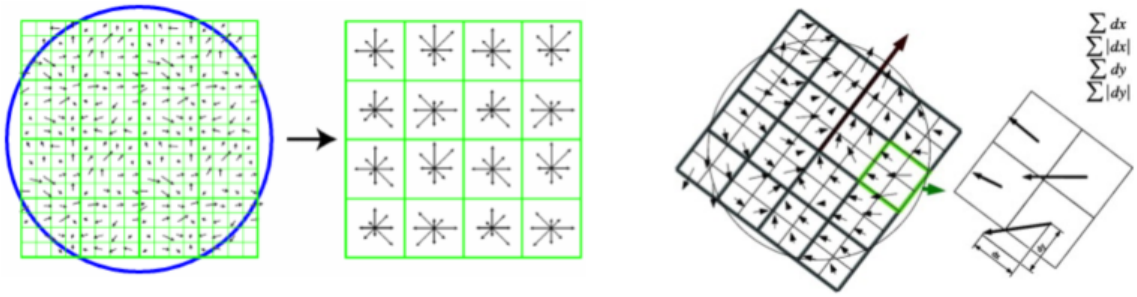


Figure 2.3: SIFT descriptor [60] (left) and SURF descriptor [64] (right).

its robust performance in applications, as shown in the works of [21, 63]. In ORB feature detector, FAST-9 (circular radius of 9) with a scale pyramid of the image is used. FAST features at each level in the pyramid are filtered by using Harris corner at each level. Finally, using the moment of image patch, the orientation of the feature point is calculated to obtain Oriented FAST.

2.1.1.2 Feature description

Once a set of interest regions has been extracted from an image, their location, scale, orientation, and their content information need to be expressed as a descriptor, which is later used for discriminative matching. There are various descriptors in the literature. Here we only present the widely used ones in practice.

SIFT descriptor [60] is a vector of histograms of image gradients. The region around the feature point is divided into a 4×4 grid at a particular scale and orientation. Each cell yields a histogram with eight orientation bins that are arranged in a 128-dimensional vector. This vector is then normalised to a unit length to enhance the invariance to affine changes in illumination. A threshold of 0.2 is applied to reduce the effects of nonlinear illumination, and the vector is again normalised. SIFT feature descriptor is invariant to translation, rotation, uniform scale, orientation, and partially invariant to affine distortion and illumination changes.

SURF descriptor [64] uses Haar wavelet. A neighborhood of size $20s \times 20s$ is taken around the key points, where s is the scale. It is then divided into 4×4 sub regions. For each subregion, horizontal and vertical wavelet responses are taken, and a four element vector is formed. Therefore, the SURF descriptor is a 64-dimensional vector. The main advantage of the SURF descriptor is the processing speed outperforming that of SIFT. SURF is good at handling images with blurring and rotation, but not good at handling viewpoint change and illumination change. To overcome this issue, it can be easily extended to a 128-dimension version without

Figure 2.4: Examples for feature detection and matching. Features are salient points on images. The successful matchings are marked in green, mismatch pairs are in red, and blue marked points not can find their correspondences.

adding much computation complexity.

Binary Descriptors have even lower computational complexity, thus faster than SURF. *Binary Robust Independent Elementary Feature* (BRIEF) [65] is a general purpose feature point descriptor which relies on a relatively small number of intensity difference tests to represent an image patch as a binary string. It provides a shortcut and fastest way to find the binary strings directly without finding descriptors. The main advantage of it is the low requirement on memory. To obtain very good matching results, 128 or 256 bits are normally enough [65]. The standard BRIEF descriptor is a 32-dimensional vector. However, the BRIEF is not rotationally invariant. In ORB [61], the orientation of key point has been applied to BRIEF along the key point direction to enhance the ability of rotational invariance and resistant to noise.

2.1.1.3 Feature matching

Feature matching aims to tell which features of one image correspond to which features of another image. There are two main approaches to finding features and their correspondences. The first is to find features in one image that can be accurately tracked using a local search technique such as correlation [66] or optical flow [67]. The state-of-the-art methods in optical flow are available in Middlebury website [68], where most of the optical flow methods available in the literature can be found, including improved version of pioneer works of [69] and [70]. The second is to independently detect features in the whole image region and then match features based on their local appearance [71]. The former approach is more suitable for image pairs taken from image sequences, while the latter is more appropriate for those with significant motion or appearance variations.

The similarity comparison between two feature vectors is commonly measured by Euclidean distance (L_2 norm) [72], Mahalanobis distance [73], Minkowski distance for non-binary features. But for binary features, Hamming distance is often used [10].

2.1.2 Structure from motion

Feature based Visual SLAM research is closely related to the scientific discipline of structure from motion in the computer vision community. The research on SfM has been developing along with the progress of film cameras. It was often named as *pencigraphy* due to the analogy of pencils of rays projecting into a camera, but later methods adopted the more descriptive term *projective geometry* [10].

Most of the work in computer vision deal with the projection of the 3D world into a 2D plane with the consequent loss of a depth dimension. Specifically, many theoretical techniques deal with purely projective geometry, avoiding the estimation of camera motion or structure in a 3D world. SfM is the combination of both tracked 3D points and camera motion in a consistent coordinate frame.

A critical component of SfM is to estimate the fundamental matrix [10]. By matching sets of eight or five points [74, 75] between two overlapping images, the rotation and translation between two cameras can be found, then a 3D scene structure can be triangulated. This is the basic procedure that drives the majority of monocular SfM techniques. Among the process, if salient feature points are adopted as the visual measurements, series of issues should be carefully studied, which includes accurate feature detection, matching and triangulation, relative pose estimation through the fundamental and essential matrices, camera recovery from the observed structure, and outlier rejection. A comprehensive summation of such work in SfM can be found in work of [10].

Although SfM is the dominating method in structure recovery and motion from images, there are many techniques which parametrize the solution in different ways. As the work in [76, 77], the authors demonstrate that it is possible to recover structure without explicitly computing the motion or intrinsic properties of the cameras observing them.

However, SfM usually runs off-line with its focus on recovering the structure in a potentially high precision. Thus more computation is necessary for a huge number of visual measurements.

2.1.3 SLAM approaches

Visual SLAM [78] can be seen as a particular case of SfM. It refers to the process of constructing or updating a map of an unknown environment while simultaneously keeping track of a location within the map using vision. Adopting the example of feature-based visual SLAM in figure 2.1 again, we can generalise the feature-based visual SLAM formulation as a maximum posterior problem.

The projection of a scene point $\mathbf{y}_i \in \mathbb{R}^3$ into a camera with 6DoF pose $\mathbf{x}_j \in \mathrm{SE}(3)$ results in an image point $\mathbf{u}'_{ij} \in \mathbb{R}^2$ that could be observed in that camera. If a measurement of the predicted point is observed as $\mathbf{u}_{ij}$, the error induced between the predicted and observed point is $\Delta u_{ij} = \mathbf{u}'_{ij} - \mathbf{u}_{ij}$. In probabilistic terms, the PDF over this error is often assumed to be a multi-variate Gaussian distribution with diagonal covariance matrix $\sigma_{ij} \in \mathbb{R}^{3\times3}$,

$$p\left(\mathbf{u}'_{ij}|\mathbf{x}_j, \mathbf{y}_i\right) \propto \exp\left(\frac{1}{2}\Delta u_{ij}^T \sigma_{ij}^{-1} \Delta u_{ij}\right). \tag{2.1}$$

Assuming that observing multiple scene points across different locations is an independent process, then the PDF over all observations can be expressed as

$$p\left(\mathbf{u}'|\mathbf{x}, \mathbf{y}\right) \propto \prod_{i=1}^{N}\prod_{j=1}^{M} p\left(\mathbf{u}'_{ij}|\mathbf{x}_j, \mathbf{y}_i\right), \tag{2.2}$$

where the structure and motion parameters, $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_i, \cdots, \mathbf{x}_M\}$, $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_i, \cdots, \mathbf{y}_N\}$, valid observations $\mathbf{u} = \{\mathbf{u}_{ij}|c_{ij} = 1\}$, and $c_{ij} = 1$ indicates that camera j did observe point i.

Based on the equation (2.2), the unknown structure and motion can be further jointly estimated if enough observations are available. According to Bayes rule $p\left(\mathbf{x}, \mathbf{y}|\mathbf{u}'\right) \propto p\left(\mathbf{u}'|\mathbf{x}, \mathbf{y}\right) p\left(\mathbf{x}, \mathbf{y}\right)$, where $p\left(\mathbf{x}, \mathbf{y}\right)$ is prior, the most likely structure and motion can be estimated by maximising the posterior distribution. By using the monocular visual measurements, particularly local feature measurements as scene landmarks, the maximum posterior can be acquired in various frameworks to achieve better performance in efficiency, accuracy and robustness. These frameworks include filtering based, optimisation based, and graph based methods.

2.1.3.1 Filtering based SLAM

Early attempts of SLAM in robotics research are unreliable because they represent the poses and the scene map as independent states, ignoring their correlations, like

σ_{ij} in equation (2.1). This faulty assumption with inherent noisy sensor measurements will lead rapidly to over-confident state estimates. The robot becomes very certain about a wrong pose estimate, which will lead to fatal inconsistencies sooner or later [79].

The idea of representing all states, the poses as well as all landmark locations by a joint probability distribution is first proposed in the work of [80]. In this work, the authors consider a robot in 2D world, which is equipped with a laser range scanner and other egomotion odometry sensors. The basic EKF is used to fuse the information from multiple sensors. Since then, the EKF formulations have been widely adopted as a standard approach for SLAM in practical applications [81–85].

In the EKF framework, the posterior density is approximated by a Gaussian density with mean $(\hat{\cdot})$ and covariance matrix in the form as

$$\hat{\boldsymbol{x}} = \begin{bmatrix} \hat{\mathbf{X}} \\ \hat{\mathbf{Y}} \end{bmatrix} \quad \mathbf{P} = \begin{bmatrix} \mathbf{P_{XX}} & \mathbf{P_{XY}} \\ \mathbf{P_{YX}} & \mathbf{P_{YY}} \end{bmatrix},$$

where the random state $\boldsymbol{x}$ contains the robot pose vector $\mathbf{X}$ and the mapped landmark positions $\mathbf{Y}$.

However, the essential problems of EKF-SLAM include the linearization for non-linear sensor and motion models in filtering, as well as the quadratic complexity of the algorithm with respect to the number of landmarks on the map. There are an impressive amount of research well studied on such issues and various improved algorithms to achieve constant time complexity [86–90].

In the work of [87], the *Extended Information Filter* (EIF) makes use of the natural quasi-sparsity of the information matrix to achieve a constant time performance. Its sparsity is only affected by the cross-correlations between the robot pose and the set of landmarks, which are generated when performing an uncertain robot motion. Sparse EIF-SLAM approximates the affected terms with the null value and then considers only a constant size subset of the problem to perform the updates. This permits the calculation of filtering loops in constant time. Further improvements to this algorithm, e.g. the work of [91], achieves exactly sparse formulations and hence the null value approximations are no longer necessary.

The drawback of this technique is that the world representation is enclosed inside the information matrix. The geometrically expression of translation requires the operation of matrix inversion, and this process is time-consuming. However, this translation is not an essential step for filtering loops, thus can be performed at a much lower rate in a different thread.

FastSLAM [41] and its improved version *FastSLAM2.0* [92] tackle the complexity problem from a different perspective. The *particle filter* is used in a Rao-Blackwellized form to dividing the SLAM state vector into two differentiated parts, i.e. robot state and landmarks positions. On the one hand, the robot state vector shows a set of particles to approximate the main nonlinearity problems and the PDF. On the other hand, the landmark positions are as modelled as Gaussian distribution to allow variation. With a particular and accurate treatment of the different operations inside the filtering loop, all time steps and measurement steps for the whole SLAM system can be computed in constant time if we just limit the number of particles and simultaneous observations.

However, even for the simplest 2D problem and with very accurate sensor measurements, particle filter based approaches still cannot guarantee completeness with a limited number of particles.

2.1.3.2 Bundle adjustment

Both visual SfM and SLAM research can be abstracted as the problem of given a set of corresponding points among the images, then creating a 3D scene map. Following the definition in equation (2.1), the problem can be generally formulated as:

$$\hat{\mathbf{x}}_j, \hat{\mathbf{y}}_i = \arg \min_{\mathbf{x}_j, \mathbf{y}_i} \sum_{i=1}^{N} \sum_{j=1}^{M} \Delta u_{ij}^T \sigma_{ij}^{-1} \Delta u_{ij}$$

It is actually an *optimisation technique* widely used in the computer vision community for 3D reconstruction, named as *Bundle Adjustment* (BA) [93]. The core of BA aims to minimize the error between re-projections of the three dimensional model and the associated points in the image. Optimisation over the parameters is performed using a nonlinear iterative minimisation scheme, where the initial estimates of the point positions and camera poses are required.

Actually, most visual SLAM methods using recursive calculation have a lot of common aspects to SfM methods using BA. Both of them minimise the sum of squares of re-projection errors, estimate motion and structure in the full 3D space and do not incorporate any additional priors besides the image data.

The core difference lies in the demands of algorithms. SLAM is usually perceived as an online method. Representative SLAM applications, such as autonomous navigation, require the pose and map estimation performed in real time. Frames arrive consecutively, and once a new frame arrives, the joint state must be updated in-

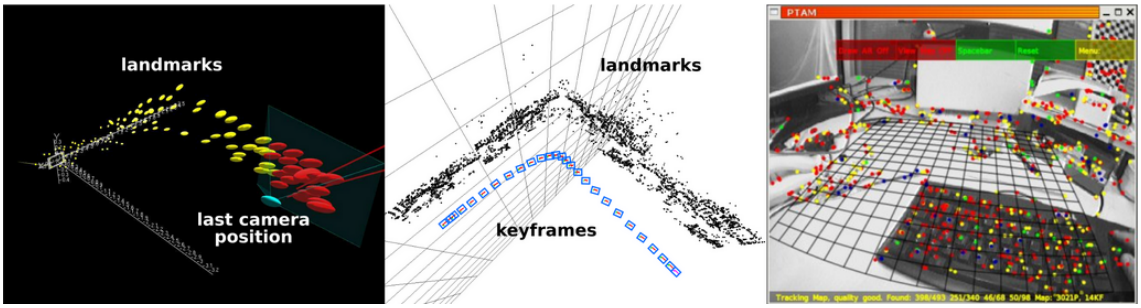


Figure 2.5: Visualized result of MonoSLAM[42] (left), keyframe-based tracking and mapping with PTAM (middle), and demo of PTAM[95] (right)

stantly. In contrast, SfM is usually a *batch based* approach. Firstly, all data is collected from a set of images. Then, a 3D representation is estimated using an extensive offline optimisation. In online SLAM methods such as filtering, the focus is on estimating a probability distribution over the current pose and the map which are statistically valid. On the other hand, batch based approaches solve the problem from random initialization and focus on global accuracy [94].

However, a representative work of *Parallel Tracking And Mapping* (PTAM) [23], using the concept of *keyframes* in BA shows computational advantages without compromising the accuracy. In particular, PTAM splits the simultaneous localisation and mapping task into two separate threads: the tracking thread and the mapping thread.

The *tracking thread* is responsible for the tracking of salient features in the camera image, i.e., it compares the extracted point features with the stored map and thereby attempts to determine the pose of the camera. This is done with the following steps: first, a simple motion model is applied to predict the new pose of the camera. Then the stored map points are projected into the camera frame, and corresponding features are searched. This step is often referred to as data association. Next, the algorithm refines the orientation and position of the camera, so that the total error between the observed point features and the projection of the map points into the current frame is minimised. Thereby *the mapping thread* uses only a set of keyframes to build a 3D point map of the surroundings. The keyframes are selected using some heuristic criteria which are based on a distance measure and the visibility, i.e. overlapping rate with the last keyframe. After adding a new keyframe, a batch optimisation is applied to refine both the map points and the keyframe poses. This attempts to minimise the total error between the re-projected map points and the corresponding observations in the keyframes.

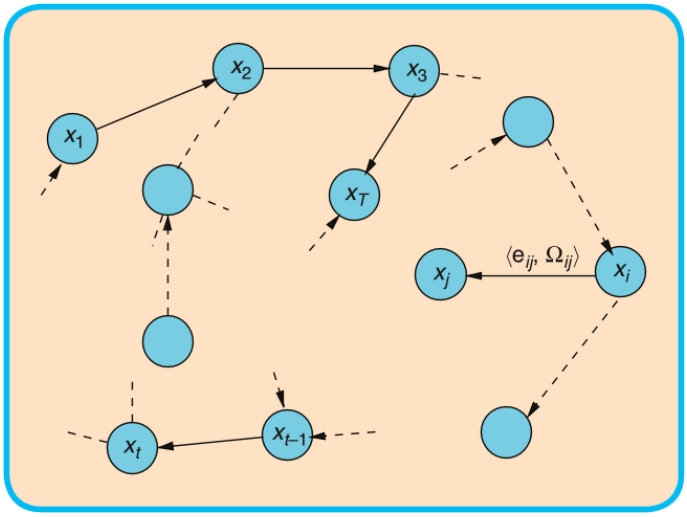


Figure 2.6: A pose-graph representation of a SLAM process from [98]. Every node in the graph corresponds to a robot pose. Nearby poses are connected by edges that model spatial constraints between robot poses arising from measurements.

However, when compared with filtering based SLAM methods, the lack of modelling uncertainties in PTAM must be compensated by the use of a lot of features and keyframes to guarantee the convergence in data association, thus limiting the algorithm only suitable in a small area of operation. Additionally, long time running will still consume large storage and computational resources.

Generally, batch SLAM can become unacceptably slow as the size of the environment grows. There are also some techniques to ease this situation. Incremental update methods *iSAM* [96] have been proposed to speed up the computation. The *sliding window* formulation that only keeps a limited amount of robot states and features is also a popular way to bound the computation complexity. *Conditioning* [95] and *marginalization* [97] are two methods to propagate information from removed states into the current estimate.

2.1.3.3 Graph based SLAM

Only recent years, *graph-based* optimization techniques have emerged and become popular [21, 98–100], an example of pose graph is shown in figure 2.6. In such setting, both robot states and environment features are considered as vertices in the graph, where observations between vertices are considered as the factor nodes in a factor graph formulation [99], or as the edges in a Markov random field formulation [100]. These are generic formulations that allow modelling the observations from multiple sources, such as those in the problems of visual odometry, feature observations,

and loop closures. Although with a large data size, the graphs resulting from all observations are typically sparse, which makes the fast solution with sparse matrix solvers become possible.

A remarkable work in graph based SLAM using features is ORB-SLAM [21] and its improved version [101] for extensive application in stereo and RGBD cameras. An example result can be found in figure 2.8, ORB-SLAM is the traditional feature-based system, and quite similar to PTAM in some way, but attains much more impressive performance in practice. This algorithm implements three parallel threads, namely tracking, mapping, and loop closing to achieve consistent localisation and mapping, while PTAM does not have loop closing thread. There is an automatic map initialization process with a model selection on two paralleling threads where either Homography or Fundamental matrix for motion estimation will be calculated with a RANSAC process. However, PTAM requires a manual operation to finish initialization. The usage of ORB feature detector with binary words searching can improve the robustness and efficiency of feature tracking and matching under the scale and orientation variation. In mapping process, the multi-scale graph maintains a common map for all parts of the algorithm, including local graph for pose estimation, the co-visibility graph for a local map, and essential graph for global BA after loop closure detection.

2.1.3.4 LiDAR based SLAM

After more than a decade of development, *Light Detection and Ranging* (LiDAR) sensors have already become another kind of dominant sensor in the applications of mobile platforms. Different than visual sensing method, LiDARs can detect environmental structure regardless of background illumination. This active nature makes them become an attractive sensor applied in either indoor or outdoor unknown scenes. Since this sensor type is not the focus of our thesis, here we just give a general review of related matching methods, which are important in building point cloud maps in SLAM algorithms.

Whether LiDAR based or visual based, their algorithm structures applied in SLAM problem are very similar. However, due to the different ways of environmental perception, the key of LiDAR is to perform scan and match among multiple points at the front end of the SLAM algorithm. There are two general categories of matching approaches, i.e., probabilistic approach and scan-matching approach [102]. Examples of the former adopt maximum likelihood with posterior estimation [103], Kalman filter [104, 105] or maximum entropy-based energy function [106] to

enhance the robustness with respect to outliers. On the other side, scan-matching approaches are more classical, where *Iterative Closest Point* (ICP) is the most popular used [102]. The core idea of ICP is to solve a point to point least-square problem with the initial guess under closet point rule, then the solution as a relative pose is processed iteratively until satisfying a predefined criterion [107]. Since the basic ICP introduced, its variations have been proposed abundantly [108, 109]. Other than the classical ICP, the approaches of *Iterative Matching Range Point* (IMRP) [110], *Iterative Dual Correspondence* (IDC) [110] and *Polar Scan Matching* (PSM) [111] are also popular adopted in the scan matching.

Apart from SLAM research [112, 113], mobile robots equipped with LiDAR sensors have been also widely explored in the problem of obstacle detection and avoidance [114], 3D reconstruction [115, 116], place/scene classification [117] and as-built floor plans [118]. However, due to the restricts of manufacture size, exploration range and power supplement, the LiDAR sensor is still rarely applied in small mobile platforms such as mobile phones or the quadrotor developed in this thesis.

2.1.3.5 Semantic SLAM

All above SLAM approaches can provide a geometrical map for localization and navigation tasks of mobile platforms, but such map information is not straightforward for human understanding. In recent years, to extend the applications of SLAM and bridge the gap between human and machine interaction, semantic SLAM approaches have been burgeoning. In addition to spatial information about the environment, a semantic SLAM augments the typical contents with the information associated with the entities located in space, i.e., functionalities, properties or connections. This information is available for reasoning in some knowledge base with an associated reasoning engine [119]. Based on traditional SLAM techniques, visual-based semantic SLAM includes three key processes, that is visual detection, object/place recognition and semantic representation [120].

The prior advancements made in traditional SLAM pave the way for developments of semantic SLAM. Their basis of the front-end visual processing is similar, while the research in semantic SLAM more focuses on semantic mapping, specifically, semantic annotation and representation. The first system that can represent the map from both spatial and semantic perspective is presented by the work of [121, 122]. In their system, two hierarchies (a spatial hierarchy and a conceptual hierarchy) are maintained with the interrelation of anchoring [123]. The spatial hierarchy contains raw visual data from a camera, geometric information such as

grid maps as well as the connection of the environmental structure. The conceptual hierarchy represents semantic knowledge and their relations which are modelled by employing standard AI languages. This also enables the mobile platforms with inference ability. However, the conceptual knowledge must be manually annotated into the system, while the environmental entities are not with estimation uncertainties in their representation. Later work [124] adopts a Bayesian classifier to enable probabilistic inference in place and object classification. Researchers [125, 126] also focus on place classification with probabilistic representation but with different sensory methods. Their augmented conceptual map in either laser or visual-based platforms can effectively extend the application in realistic human-robot interaction.

The research of Semantic SLAM can enhance the accuracy of localisation and navigation tasks for mobile platforms in a high-level way. The conceptual information is human-friendly and can be shared easily in a mutual workspace [127]. However, it is obvious that the interpreting steps require more computational time and resources, which does not meet our focus of research as increasing the localisation accuracy as an earlier stage of information processing as possible. Hence the semantically related topics will not be further extended in this thesis.

2.1.4 Odometry approaches

Sparse features are also widely used in Visual Odometry (VO) approaches [128]. When compared to visual SLAM, VO methods remove the loop closure detection in live motion estimation. The VO approaches are usually fast to compute, which is beneficial for mobile platforms with limited onboard computation, such as multi-rotors [6] and smart phones [95]. However, the VO approaches have the disadvantage of larger error accumulation over time than SLAM methods. In order to bound the rapid growing drift, extra constraints from either internal or extra sources are usually adopted.

In some monocular VO works like [129–131], the authors compute and optimise the motion with structure at the same time. In these methods, the constraints as tracking and triangulating structures over multiple frames are used to achieve a consistent pose estimation. However, these additional mapping with extra correction steps are complicated and more computational demanded. While for some works like [132, 133], motion models are used to constraint the motion between consecutive frames. Such model constraints can also be in the form of a known assumed motion model on a plane as presented in the works of [133, 134]. However, the absolute

scale reference to the world still not can be achieved within own algorithms.

In the literature, a number of techniques exist to get a real scale recovery or bound the drift in scale by using the reference form other sensors. The work of [135] use a height estimation over ground to infer the scale, while the work of [136] estimates scale by fusing inertial and monocular visual data in an EKF framework.

Multiple rigidly fixed cameras with overlapping views, time-synchronized to capture the image at the same instant are also commonly used [134, 137, 138]. In most cases, a stereo pair is applied since it is the minimum number of cameras need to resolve scale ambiguity. Such additional camera can significantly increase the visual information available to the VO algorithm, which means that there are more observations of individual features are available. By using stereo vision, the triangulation calculation for the motion estimation can be reduced [129].

As stated in previous section 2.1.3.4, for the laser based methods, the ICP algorithm [107] and its variations [108, 109] are popular used for incremental motion estimation. However, ICP is an iterative optimisation approach without a explicit process of data association. It has poor convergence property when the sensor data is unstable and discontinuous. A work using multi-resolution correlative scan matching [139] reports a surprising robust result even with large scan displacement. In ICP methods, expect for spatial points, corner or line features can also be extracted for data association between scans, as shown in the works of [5, 140]. With such successful data association, scans can be directly matched to obtain the relative transformations.

2.2 Direct Visual SLAM and Odometry

In contrary to local geometric features, there is another branch of utilising image information in the robotic and computer vision communities, i.e. direct-based methods. Such methods extract dense image data from the moving camera, and eliminate the processing of feature detection, description and matching. An institutive comparison is illustrated in figure 2.7. The direct-based methods can achieve better robustness than traditional feature-based methods in the situation of noise pollution, partial occlusion, light condition variation. In some research, the direct-based image information is also called photometric or global features.

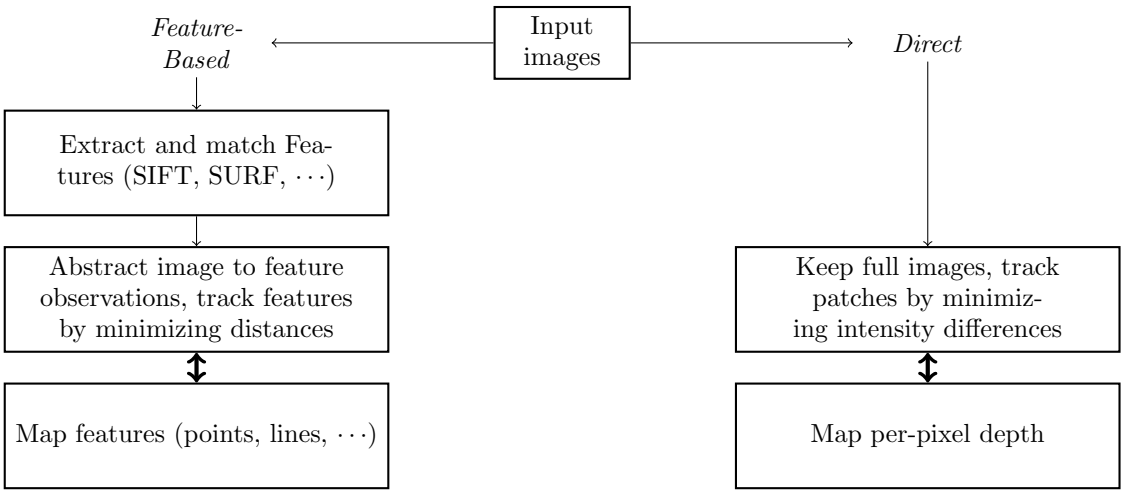


Figure 2.7: Comparison of feature-based and direct-based process .

2.2.1 Photometric measurements

The core of direct-based methods is to build a photometric error formulation including the motion and structure parameters. Here we define a *wrap function* $w(\cdot)$ which takes a pixel $\mathbf{u} \in \Omega \subset \mathbb{R}^2$ in the reference frame in image I_a and transforms it into a pixel in image I_b using pose transform $\boldsymbol{\xi}$ and scene depth d parameters. Given $M \geq 1$ pixel correspondences, the inter frames motion trajectory $\mathcal{T}(\cdot) \in \mathrm{SE}(3)$ and scene structure as depth map $\mathtt{D}(\cdot) : \Omega \to \mathbb{R}^+$, the estimation problem can be formulated as

$$\hat{\mathcal{T}(\boldsymbol{\xi})}, \hat{\mathtt{D}}(\mathbf{u}) = \arg \min_{\boldsymbol{\xi}, d \in \mathbb{R}^+} \sum_{\mathbf{u} \in \Omega, j=1}^{M} \phi \left((\mathrm{I}_b(w(\mathbf{u}_a, d, \boldsymbol{\xi}_{ja}))) - \mathrm{I}_a(\mathbf{u}_a) \right), \tag{2.3}$$

where an image interpolation function $\phi(\cdot)$ enables subpixel intensity values to be computed. Actually, the warp function together with image interpolation constitute a generative model, and they can be expressed as any form that predicts an image measurement from a set of parameters. This flexibility makes the direct-based methods abundant. In practice, the wrap function can be expressed as simple as a projective relationship [44, 141] or as complex as an appearance model [142, 143].

Compared to feature-based methods, the direct-based methods recover the structure or motion directly from intensity and gradient values in the image. The magnitude, direction, or statistic information of intensity can be used in estimation, except for distance discriminant in the feature matching process [28]. And from another point of view, the feature-based methods come with the limitation that only

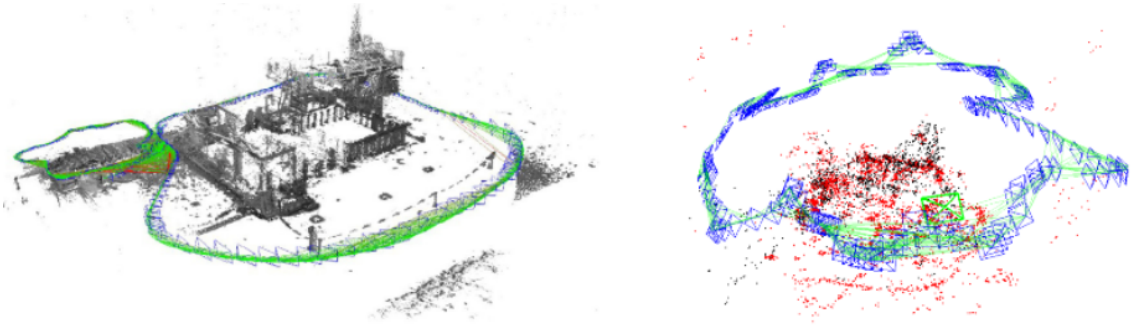


Figure 2.8: Result of LSD-SLAM[43] (left) and result of ORB-SLAM [21] (right).

the scene with a particular feature pattern could be expressed and used. However, with the successful rotation and scale invariant character of some feature detectors and descriptors, some classical methods in visual related SLAM or odometry methods like PTAM [95], ORB-SLAM [101] have been developed and commonly accepted. But to use such algorithms to provide a real-time onboard guidance for mobile platforms, it still not can neglect the computational burden when mapping feature points into a high-dimension description and conducting outlier estimation. In the research of direct-based methods, the advantage of removing tedious feature-based processing is naturally available, so that considering how to use the direct pixel information becomes the primary focus.

The most prominent character of direct-based methods is that all pixel information in one image could be exploited, even from environments where have a little texture, few key points or tremendous impact of camera defocus and blur. Only recently, some direct-based visual SLAM approaches have been proposed. In REMODE [142], DTAM [143], the whole dense depth maps are built, depending on the computational ability of state-of-the-art GPU. But they are tough to launch on small size mobile platforms. Thus, researchers in work of [141] consider to cut down dense region for reducing the computational complexity and propose a semi-dense depth mapping method. The method of SVO [44] combines direct information with key points repetitively to enhance the accuracy of tracking results.

Among these above-mentioned methods, LSD-SLAM [43] is a representative work of direct-based SLAM, which is entirely different from PTAM or ORB-SLAM. An example result can be found in figure 2.8. Rather than relying on image features, this method is based on image pixels directly, particularly the salient edges in images. The tracking is performed by image alignment using a coarse-to-fine algorithm with a robust Huber loss. Depth estimation is just like many other SLAM systems, using

an inverse depth parametrization with a bundle of relatively small baseline image pairs. Map optimisation is also executed in commonly used graph optimisation, with existing keyframe poses expressed just as that of ORB-SLAM. However, LSD-SLAM recovers only a semi-dense map since it only estimates depth at pixels solely in salient boundaries, which makes it to be the first direct visual SLAM system that can run in real-time on a CPU. In practice, processing every pixel over all image sequences is a computationally consuming task. This is also the reason why many dense visual SLAM systems, like DTAM [143] require a GPU to attain real-time performance.

Direct sparse odometry (DSO) [48] is also a recent work using a direct and sparse formulation for Visual Odometry. It combines a fully direct probabilistic model with a photometric error. Joint optimisation is performed over inverse depth and camera motion parameters. Moreover, by omitting the smoothness prior used in other direct methods and sampling pixels evenly throughout the images, this method can achieve real time performance. DSO also does not depend on feature detectors or descriptors. Thus it can naturally sample pixels from across all image regions that have intensity gradient, including edges or smooth intensity variations on the white walls. Furthermore, this method integrates a full photometric calibration, accounting for exposure time and nonlinear response functions, making it more accurate and robust in using image intensity.

There are also some works based on image intensity in different mathematical space but avoid directly using the intensity value. In the works of [144] and [145], the authors consider using the full image by reducing the dimensionality of image data. They apply an eigenspace decomposition to the images. Their target controller is then performed directly in the eigenspace, which requires both an offline computation of such eigenspace using a Principal Component Analysis (PCA) and the projection of each acquired image on this subspace. Differently, the work in [146] proposes to regulate directly the Sum of Squared Differences (SSD) between the current and reference images. However, this approach is quite sensitive to illumination variations although using a more sophisticated illumination model in some particular cases is possible. The authors in [147] consider the pixel intensities with a kernel-based method, but this approach is very limited in the case of appearance variations, and has bad performance in a 6 DoF pose estimation.

Figure 2.9: Illustration of multi iterations using MI to find the best patch correspondence from the work of [150].

2.2.2 Mutual information

Direct-based methods using photometric error have already shown their advantage of robustness in visual processing in localisation and mapping applications [43, 48, 143]. However, the fact remains that pixel intensities are quite sensitive to environmental variation [146]. There is another branch in direct methods that avoiding the calculation of pixel intensity itself but exploiting the maximum statistic value for image pairs, i.e. *Mutual Information* (MI) [148]. Based on the information theory of Shannon in [149], the MI-based methods are built from the joint entropy of image pairs to measure their mutual dependency. In particular, this type of direct methods does not directly use the individual or modelled photometric error to build the cost function. Instead, the images are regarded as random variables and the distribution of pixel information will be considered in an alignment processing.

Similar to the definition of equation (2.3), assuming a wrap function can model a transformation $\boldsymbol{\xi}$ from patch $\mathbf{x} \in \Omega$ (the size is flexible, which can be as large as the image or as small as only several pixels) in reference image I_a to another image I_b. The best alignment is to find the most-like corresponding patch in image I_b. If mutual information is used to measure the similarity between two patches, the problem can be rewritten generally as

$$\hat{\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}} \sum_{\mathbf{x}\in\Omega} \mathtt{MI}\left(\left(\mathrm{I}_b(w(\mathbf{x},\xi))\right) - \mathrm{I}_a(\mathbf{x})\right). \tag{2.4}$$

An example of such best patch matching can be found in figure 2.9. More details about the knowledge of MI will be introduced in chapter 5.

There are some works in the literature using MI as the core of visual tracking and visual servoing problems. The work of [150] builds up an MI cost function used for

augmented reality applications. They refine the computation of the Hessian matrix and condition the optimisation to deal with the quasi-concave shape of MI cost function, enhancing the robustness and accuracy of the MI-based tracker. However, the algorithm is limited to planar object tracking. Later, the improvement work of [47] expands the tracking into 3D space using object models. A synthetic view of the target, together with a buffer of known depth information from a model generator, the calculation process strives to find the best matching iteratively within a searching range of position and angle. However, the major drawback of this method is that it is not real time. It takes approximate four seconds for the processing of each image. It should be noticed that such MI-based methods rely on a known target view or model. However, in practice, acquiring environment information beforehand is not an easy and effective way in most visual tracking and pose estimation tasks. In a limited working environment, the work of [49] build a control law based on MI for visual servoing. The inherited merit shows huge accuracy and robustness improvement in a positioning task.

From another point of view, MI is a classic similarity measure commonly adopted as multi-modal registration techniques in medical imaging [151, 152] and remote sensing [148]. The images in such disciplines are often taken from different sources. Take the earth map for instance, traditional feature-based methods not can distinguish whether satellite view and normal colour map are the same places, but MI-based methods can find the best match although the map modality is different [153].

However, we notice that the function (2.4) does not rely on any parameters related to scene structure, if the wrap function is just modelled as simple similarity transformation, like affine. Without extra scale reference, the MI-based methods also inherit the scale drift problem from visual-based estimation. In the thesis, we will hand over the scale recovery to an inertial-driven estimation process. Additionally, although MI has been widely used for multimodal medical image registration [154] and in tracking [155], visual servoing [49], to the best of our knowledge, none have been considered to be applied in visual odometry mission and fused with an inertial sensor.

2.3 Visual Inertial Fusion

As shown in previous sections, visual-based methods, particularly in the monocular case, inherently suffer from the lack of scale reference, thus causing drift over time.

Instead of applying other additional visual setting like stereo vision [46], we can think of a variety of sensors to retrieve the scale. But most of the sensors have drawbacks including the aspects of range restriction, physical size or power supplement, thus limiting their applications in light weighted mobile platforms. Apart from a motion model as used in the works of [156–158], the IMU is always an ideal choice when used as a complementary measurement method for the monocular camera. Actually, the research on SLAM and odometry approaches based on visual and inertial sensors never stopped. The focus is on different detailed aspects regarding visual and inertial fusion, including inter sensor calibration [159, 160], scale recovery [161], fusion algorithm framework [11, 36] and techniques for efficiency performance [36, 37, 40].

The early work of [162] demonstrates accurate motion and structure estimation by combining IMU measurements and camera observations of features. The works in [14, 163] fuse IMU and camera data in an EKF framework for motion estimation and they also use an additional EKF process to estimate the positions of landmarks. Along with the improvements of onboard computing power, the realisation of visual-inertial navigation becomes possible [13]. However, although all of these algorithms predict motion and structure, they do not incorporate calibration between two sensors.

For visual-inertial calibration techniques, there are some representative works in the literature. The work of [159] uses a constrained nonlinear optimisation algorithm to solve the fixed rotation between a camera and an IMU. By comparing camera measurements of the relative angles to several external markers with integrated gyroscope outputs, such optimisation algorithm determines the rotation which best aligns the sensor frames. While in the work of [160], the authors describe a visual-inertial calibration procedure in which the relative orientation and the relative translation of the sensors are estimated separately. Firstly, the relative orientation is found by rotating the camera-IMU platform while the camera captures images of a vertical planar target. The relative translations are then determined by spinning the camera and the IMU on a turntable, making the measured horizontal acceleration of IMU is zero. A drawback of the method is that separate calibration of orientation and translation ignores any correlations that exists between the parameters, and hence do not adequately account for further error propagation.

From the perspective for visual and inertial fusion, there are two main categories. To achieve higher precision, *tightly coupled* methods are more favourable in the literature, which jointly estimate robotic pose and camera information, thus

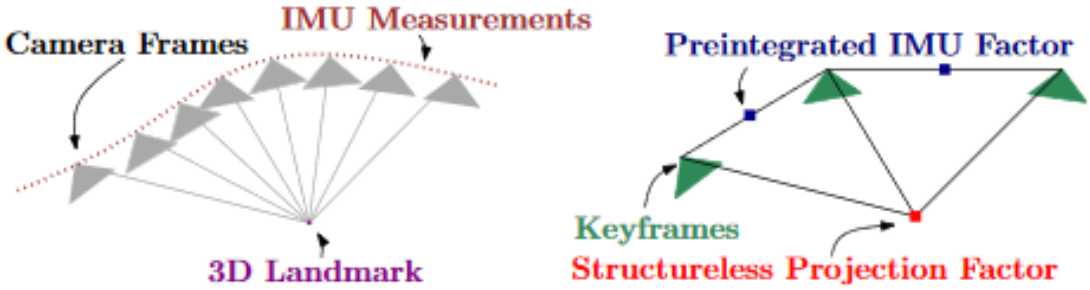


Figure 2.10: Visual and inertial measurements and its factor graph from the work of [37], in which several IMU measurements are summarized in a single preintegrated IMU factor, together with a structureless vision factor constrained by keyframes observing the same landmark.

involving calibration between two sensors. Like the work of [164], the authors use all information from the two sensors under the tightly coupled structure to statistically optimise the 6 DoF IMU pose, the inter sensor calibration, the visual scale factor and the gravity vector in the vision frame. Although it not can ignore the fact that tightly coupled methods essentially own the disadvantage of more computational complexity, strategies like dynamic marginalising out oldest state and keeping a moving window [165] or select keyframes among consecutive images can still bound algorithm in an acceptable level [22, 141]. On contrary, the *loosely coupled* methods are more computationally efficient. They process inertial data and image measurements separately. For instance, in the works of [166, 167], the authors firstly process the consecutive images for computing relative motion estimate then fuse with the inertial measurement. Alternatively, like the work of [168], the inertial measurement could be used to conduct rotational estimation, then combined with the image-based algorithm. Or in the work of [12], the authors view the visual framework as a black box, which could be replaced by any other pose estimation from different sensors then fusing with inertial data. The strategy of separating these two measurement sources leads to a reduction in computational cost. It is more suitable for mobile platforms with limited onboard resources.

Among various fusion methods, optimal values are acquired from either iterative minimization or EKF formulations. Although they can be converted to each other theoretically [169], different state expressions and resolving methods still make them have dramatically different performance in practical computation. Previously-mentioned methods, like [22, 44, 95, 143], essentially transform the problems into nonlinear least square problems, solved by classical algorithms like iterative Lucas-

Kanade [170] or just basic Gauss-Newton method. The need of multiple iterations during minimization inevitably results in increasing computational cost. However, this is not to imply that filtering based outperform optimisation methods all-around. In the typical EKF-based SLAM method [164], the authors keep the features in state vector, thus causing its runtime reaches an unacceptable high status as $O(n^2)$ for n features. To address this problem, the MSCKF algorithm was proposed as an alternative filter based visual-inertial odometry method [13]. In contrast to traditional filtering based SLAM methods, the MSCKF can actively augment and remove the state vector, and uses the feature measurements to impose constraints on these poses. This strategy makes the computational complexity dramatically reduce to be linear with the number of features as $O(n)$. In this thesis, we will use the MSCKF framework for localisation and mapping, but improve the visual measurement models from direct-based methods.

2.4 Summary

Although methods in feature-based visual SLAM or odometry problems have tended to be mature in recent years, the demand of algorithms with high accuracy, robustness and less complexity is still in tension. Recent development in direct-based visual measurements seems open another window for related research communities. No matter dense, semi-dense or mutual information based methods, pixel intensity being exploited directly or statistically in motion estimation has enhanced the robustness for the image processing in SLAM or odometry approaches. Furthermore, with the help of abundant fusion strategies in algorithm frameworks, there seems a clue to compensate for the scale ambiguity for monocular visual measurements, i.e., combined with inertial-based estimation. While on the other hand, the novel techniques adopted in computation (like sparse techniques, actively adjusting the length of state or separating tasks into multiple threads) can reduce the computational burden, further paving the way for such algorithms applied in small size mobile platforms. In our work, we aim at developing the odometry methods by adopting direct-based visual measurements and inertial estimates. The target VIO approaches are expected with acceptable performance in accuracy and efficiency with significant advantages in robustness when applied in complex indoor and outdoor environment.

Chapter 3

Technical Preliminaries

This chapter introduces the spatial relationship and mathematical notations of a visual-inertial system, building up the theoretical basis for later chapters. Starting from the basic perspective model of a monocular camera, the geometrical and mathematical transformation between different reference frames will be illustrated to show how a 3D point in the world can link to the images taken from different views. In the kinematics section, three expressions of rotation will be introduced, i.e., rotation matrix, Euler angle and quaternion. Together with the transition expression, the minimal representation of pose for a visual-inertial system will also be described. Furthermore, the two basic computational frameworks which are popularly used in most estimation algorithms are presented from simple linear to nonlinear cases.

3.1 Visual Geometry

To gather the visual data from the world, a digital camera needs to receive light rays and convert them into images through photosensitive sensors. A single image is often divided into an enormous number of elements as *pixels* in the form of a rectangular matrix, which contains the information of colour or luminosity as integer values. The pixel sampling is a diffusion process gathering the rays from direct light sources or reflection of various surfaces in a short time interval. This is actually the *Lambertian reflectance model* [171], building up the bridge between the position of objects in the scene and their projection on the image plane in Euclidean geometry. In this section, we will recall the visual geometry and their mathematical expressions.

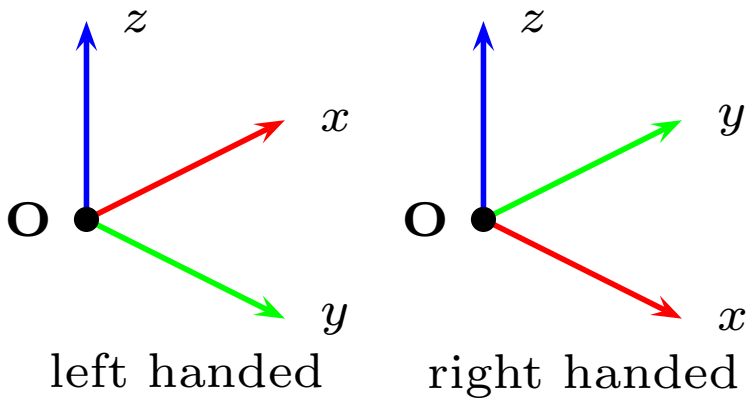


Figure 3.1: Left and right handed Cartesian coordinate frames.

3.1.1 Reference frames for monocular camera

Our world is usually to be modelled as 3D Euclidean $\mathbb{E}^3$ space using triplets of real numbers as the expression. The origin of such coordinates is the joining point from three perpendiculars directed lines as x, y and z-axis. Although the origin location and axis orientation can be defined arbitrarily, the directions are often following the convention of sensory methods.

Traditionally, the 3D coordinate frames can be defined following the rule of *left-handed* or *right-handed* as shown in figure 3.1, where the thumb, fore finger and middle finger defines the x, y and z-axes respectively. These two possible frames not can be transformed to each other through only rigid movements, thus here in this thesis, we only follow the right-handed rule to define a coordinate.

In a visual geometry model, all the points in 3D space will be projected onto the image plane which is firmly attached with a camera. There are three reference frames (world frame $\mathcal{W}$, camera frame $\mathcal{C}$ and pixel frame $\mathcal{P}$) need to be taken into account to express a 3D point $\mathbf{p}$ in the scene and its projected point $\mathbf{u}$ in image plane, as shown in figure 3.2.

The *world frame* $\mathcal{W}$ or *global frame*, is a reference frame external to the camera used for expressing the global position, thus it should be an unique, static and absolute frame. For convenience, we align the z-axis pointing upward with the vertical direction, the other two are defined following the right handed rule. If a particular direction on this horizontal plane needs to be defined, the x-axis will be aligned with the rightward (see figure 3.2 or 3.9). The 3D coordinates of a point $\mathbf{p}$ are $\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} = (x^{\mathbf{p}}, y^{\mathbf{p}}, z^{\mathbf{p}})^T$ in this frame.

The origin of the *camera frame* $\mathcal{C}$ is located on the centre of camera's perspective

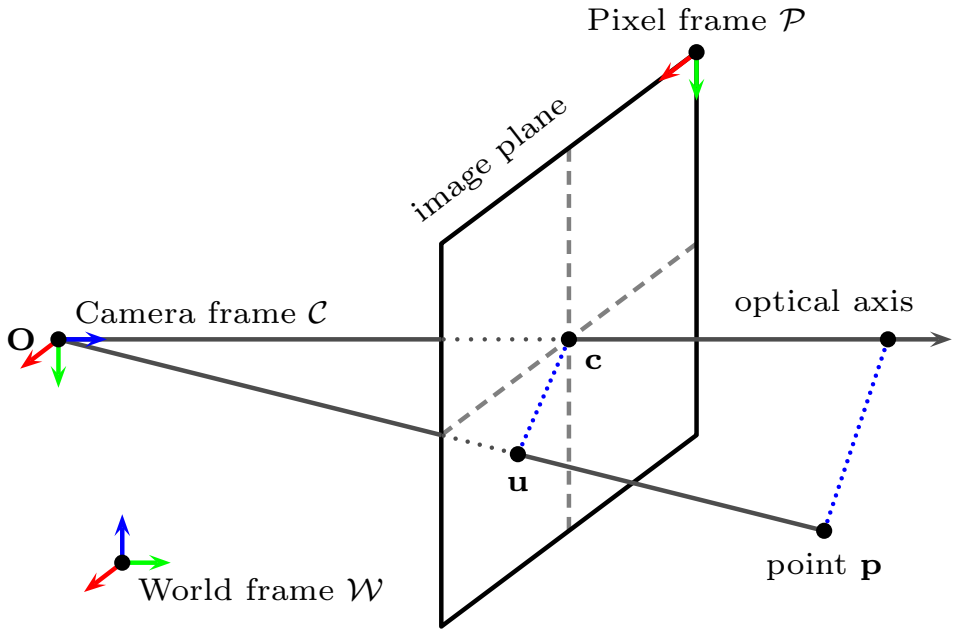


Figure 3.2: Reference frames for monocular camera. x, y, z-axis in red, green and blue receptively of world and camera frames, while u, v-axis is in red and green of pixel frame.

projection, thus often named as *camera centre* **O**. This origin is essential to express the position of a camera. The z-axis for camera is usually defined pointing through the image centre thus making the xy plane parallel to the image plane. Further followed the right handed rule, the x-axis is defined as horizontal while y-axis is vertical for convenience as shown in figure 3.2. In the camera frame, the 3D coordinates of point **p** is noted as $\mathbf{x}_{\mathcal{C}}^{\mathbf{P}} = (x_{\mathcal{C}}^{\mathbf{P}}, y_{\mathcal{C}}^{\mathbf{P}}, z_{\mathcal{C}}^{\mathbf{P}})^T$.

The *pixel frame* $\mathcal{P}$ is to represent the image plane with the origin located on the upper-left corner. Any point in the plane is measured in pixel units and expressed as integer values $\mathbf{u} = (u, v)^T$. The horizontal left to right coordinate u and the vertical top to bottom coordinate v also indicate the pixel location in an image matrix.

As shown in figure 3.2, starting from camera centre, the optical axis goes through image plane perpendicularly, then intersect with the image plane in *principal point* **c**, which is located at the centre of image plane $(u_0, v_0)^T$. The principal point is often referred as the origin of *image frame*, which is also an expression of the image plane but in pixel units. Here, we only adopt pixel frame and make a unity expression for image plane in pixel cells. Moreover, the distance between the camera centre and the principal point is measured by the *focal length* f, which also represents the

depth of any projected point in an image plane. It must be noted that the pixel frame is a 2D expression for image plane, while the position of any spatial point is expressed in 3D Euclidean space. In order to further explore the location and motion of visual-based problems, the relationship between different frames should be built up.

3.1.2 Perspective projection

The geometrical operation named *projection* is viewed as an injective process which maps points from the 3D space into points of the 2D plane ($\mathbb{R}^3 \rightarrow \mathbb{R}^2$). There exist different models in the computer vision community used for building such projective relations according to the type of camera. For a monocular case, the *pinhole model* is applied widely in the perspective projection. Within the projection process, the intrinsic and extrinsic parameters are critical factors for this mathematical model.

The *intrinsic* parameters consist of the focal length, image centre as principal point and other lens parameters, taking care of the inner relationship from normalized image plane to pixel frame. Following figure 3.2, by similar triangulation, the projection of point $\mathbf{p}$ in camera frame $\mathbf{x}_{\mathfrak{C}}^{\mathbf{p}} = (x_{\mathfrak{C}}^{\mathbf{p}}, y_{\mathfrak{C}}^{\mathbf{p}}, z_{\mathfrak{C}}^{\mathbf{p}})^T$ onto the image plane can be written as

$$x = \frac{x_{\mathfrak{C}}^{\mathbf{p}} \cdot f}{z_{\mathfrak{C}}^{\mathbf{p}}}, \quad \text{and} \quad y = \frac{y_{\mathfrak{C}}^{\mathbf{p}} \cdot f}{z_{\mathfrak{C}}^{\mathbf{p}}}, \tag{3.1}$$

where f denotes the focal length. By using the homogeneous expression, equation (3.1) can be converted into a linear matrix form as

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} x_{\mathfrak{C}}^{\mathbf{p}} \cdot f \\ y_{\mathfrak{C}}^{\mathbf{p}} \cdot f \\ z_{\mathfrak{C}}^{\mathbf{p}} \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{\mathfrak{C}}^{\mathbf{p}} \\ y_{\mathfrak{C}}^{\mathbf{p}} \\ z_{\mathfrak{C}}^{\mathbf{p}} \\ 1 \end{bmatrix}, \tag{3.2}$$

where $\lambda = 1/z_{\mathfrak{C}}^{\mathbf{p}}$ is the scale factor. Since the origin of pixel frame is located as the top-left corner of the image, in order to express the projected point in pixel frame, the transformation from metric to pixel distance is required as the unit of $mm/pixel$:

$$s_x = \frac{x}{u - u_0}, \quad s_y = \frac{y}{v - v_0}, \tag{3.3}$$

where u_0, v_0 are the coordinate of principal points in pixel frame. Following the

equation of (3.3), we can obtain:

$$u = \frac{1}{s_x}x + u_0, \quad v = \frac{1}{s_y}x + v_0. \tag{3.4}$$

Combining with equation (3.4), the linear matrix expression of equation (3.2) can be further written as

$$\begin{aligned} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \lambda \begin{bmatrix} \frac{1}{s_x} & s_\theta & u_0 \\ 0 & \frac{1}{s_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{\mathcal{C}}^{\mathbf{p}} \\ y_{\mathcal{C}}^{\mathbf{p}} \\ z_{\mathcal{C}}^{\mathbf{p}} \\ 1 \end{bmatrix} \\ &= \lambda \begin{bmatrix} \frac{f}{s_x} & s_\theta & u_0 \\ 0 & \frac{f}{s_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_{\mathcal{C}}^{\mathbf{p}} \\ y_{\mathcal{C}}^{\mathbf{p}} \\ z_{\mathcal{C}}^{\mathbf{p}} \\ 1 \end{bmatrix}, \end{aligned} \tag{3.5}$$

where s_θ is the skew of pixel closing to zero. By abusively using the point expression in non-homogeneous form, the 3×3 *intrinsic matrix* $\mathbf{K}$ can be shown in the projection process:

$$\mathbf{u} = \lambda \mathbf{K} \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix}_{3\times 4} \mathbf{x}_{\mathcal{C}}^{\mathbf{p}}. \tag{3.6}$$

The *extrinsic* parameters express the location and orientation of a camera in the 3D scene, indicating the spatial relationship between world frame and camera frame. As shown in figure 3.3, every camera defines the same 3D point in their coordinate frame. In order to get a unified expression of the scene structure and camera motion trajectory, a common frame is necessary. By using a fixed world frame, every observation expressed in local camera frame can be further aligned to a consistent reference. Actually, this method is commonly adopted in many visual-based systems, where the first pose of the camera is initialized as the basic fixed frame, then the incremental motion is computed relative to this initial given position and orientation. In this process, rotation and translation are the key factors in finding the twisted angle and travelled distance.

In general, the motion of a monocular camera system can be defined by a homogeneous matrix $\mathbf{T} \in \mathrm{SE}(3)$, which belongs to the special Euclidean group as

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \tag{3.7}$$

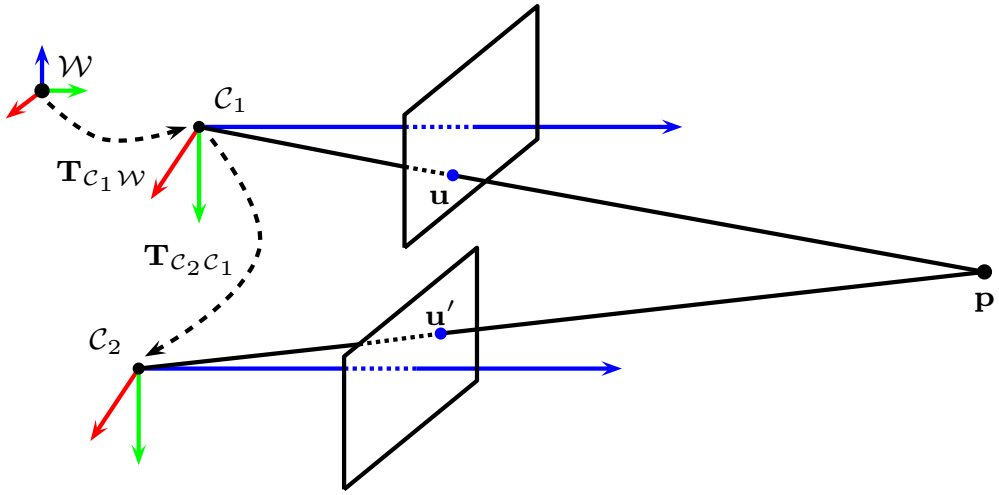


Figure 3.3: Transformations between the world frame and camera frames.

where $\mathbf{R} \in \mathrm{SO}(3)$ is a rotation matrix of the special orthogonal group and $\mathbf{t} \in \mathbb{R}^3$ denotes the translation vector. The expression of $\mathbf{T}_{\mathcal{C}\mathcal{W}}$ means the transformation from camera frame $\mathcal{W}$ to world frame $\mathcal{C}$, which is also named as *extrinsic matrix* for the camera.

$$\mathbf{x}_{\mathcal{C}}^{\mathbf{p}} = \mathbf{T}_{\mathcal{C}\mathcal{W}}\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} \tag{3.8}$$

Therefore, the full projection equation maps a 3D point in the world to a 2D image point as a pixel, which can be obtained by jointly applying the intrinsic and extrinsic matrix as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \lambda \begin{bmatrix} \frac{f}{s_x} & s_\theta & u_0 \\ 0 & \frac{f}{s_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix}_{3\times 4} \begin{bmatrix} x_{\mathcal{W}}^{\mathbf{p}} \\ y_{\mathcal{W}}^{\mathbf{p}} \\ z_{\mathcal{W}}^{\mathbf{p}} \\ 1 \end{bmatrix} \tag{3.9}$$

We can write equation (3.9) into a compact form as

$$\mathbf{u} = \lambda \mathbf{M} \mathbf{x}_{\mathcal{W}}^{\mathbf{p}} \tag{3.10}$$

where $\mathbf{M}$ is a *projection matrix* with the dimension of 3×4 combining all intrinsic and extrinsic parameters together. Usually, the intrinsic matrix can be obtained beforehand by exploiting calibration techniques. More details about camera calibration can be found in computer vision tutorials [10, 172] and resource of OpenCV.

3.1.3 Back projection

The process of a back projection maps a 2D point in the image plane to 3D space ($\mathbb{R}^2 \rightarrow \mathbb{R}^3$). It is the reverse process of forward projection. But for a single image case, the projection is not an invertible process with the lost dimension in scene depth. It is easy to derive the inversion of intrinsic parameters from equation (3.2) to (3.5) which maps the pixel from pixel frame to camera frame:

$$\begin{aligned} &\begin{bmatrix} x_{\mathcal{C}} \\ y_{\mathcal{C}} \\ z_{\mathcal{C}} \end{bmatrix} = \frac{1}{\lambda} \begin{bmatrix} \frac{s_x}{f} & -s_x s_y s_\theta & s_x s_y s_\theta v_0 - s_x u_0 \\ 0 & \frac{s_y}{f} & -s_y v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, \\ &i.e. \quad \mathbf{x}_{\mathcal{C}}^{\mathbf{p}} = \frac{1}{\lambda} \mathbf{K}^{-1} \mathbf{u}. \end{aligned} \tag{3.11}$$

After camera calibration, all elements in the intrinsic matrix can be obtained, but the scale parameter is still unknown. Actually, if we rewrite above equation (3.11) as follows,

$$\begin{aligned} \lambda x_{\mathcal{C}}^{\mathbf{p}} &= \frac{s_x}{f} u - \frac{s_x s_y s_\theta}{f} v + \frac{s_x s_y s_\theta v_0 - s_x u_0}{f} \\ \lambda y_{\mathcal{C}}^{\mathbf{p}} &= \frac{s_y}{f} v - \frac{s_y v_0}{f} \\ \lambda z_{\mathcal{C}}^{\mathbf{p}} &= 1 \end{aligned} . \tag{3.12}$$

This can be actually described as the line function, starting from camera centre $\mathbf{O}$, then passing through the pixel point $\mathbf{u}$ with a scale parameter λ, which is tightly related to the point coordinate in z-axis, as shown in figure 3.4. Usually, when a single image from a monocular camera is taken, the *depth* observation is not preserved. In order to recover the scene depth, additional information from other sensors or more images from a different angle of view towards the same scene are required.

3.1.4 Geometric relations between frames

In above section, the forward and back projection processes showcase the mapping between 2D and 3D by using a single point from a monocular camera case. It is clear that the lack of depth value in back projection will lead to multiple corresponding spatial points and further confuse visual-based tracking or mapping tasks. In order to recover relative motion of a monocular based mobile platform in a consistent

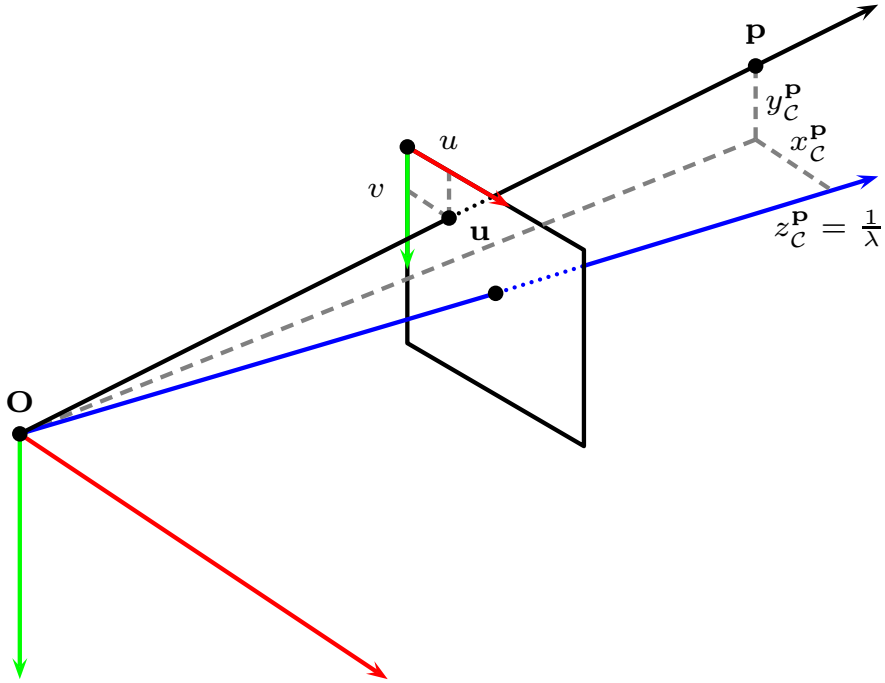


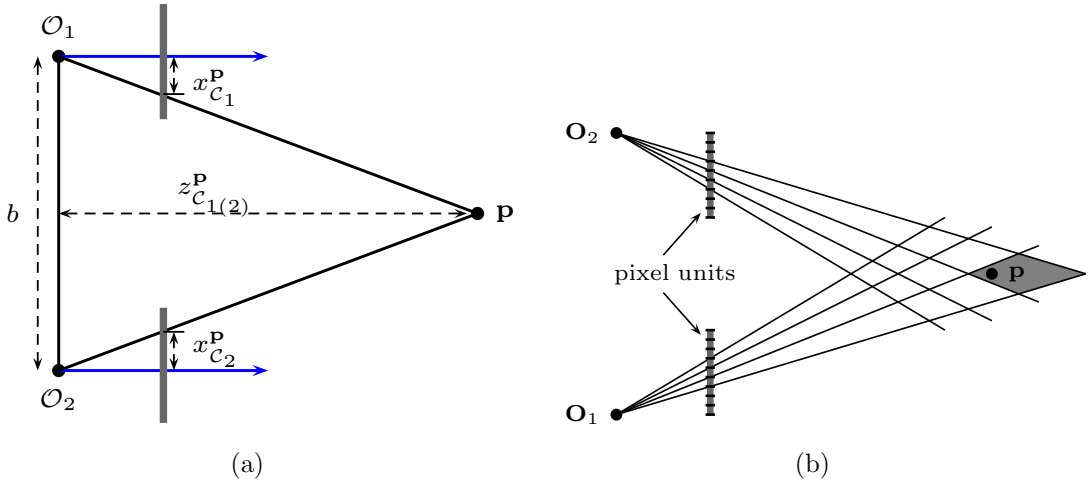
Figure 3.4: The scale parameter s of a certain point $\mathbf{p}$ in space is required in a 2D to 3D back projection. All points along the ray starting from optical centre $\mathbf{O}$ then passing the pixel point $\mathbf{u}$ satisfy equation (3.12).

trajectory, the inter camera relations between different view and scene need to be carefully studied. In this section, the frame relations from geometrical and analytical views will be studied, including the simple triangulation, epipolar geometry and homography matrix.

3.1.4.1 Triangulation

Without loss of generality, the spatial relationship of two images from a two single camera view can be built from the horizontal binocular disparity. As shown in figure 3.5(a), a pair of images with known *baseline* b (distance between two camera centre $\mathbf{O}_1$ and $\mathbf{O}_2$) are used to find the correspondence point sets.

This basic model is a simplified case for stereo cameras. However, due to a fixed physical limitation of the stereo case, the baseline of two cameras is small, and the two optical axis are approximate to parallel, as shown in figure 3.5(a). In stereo cameras, the majority information from the first image will appear in the second image, except for extreme occlusion cases. Thus for an ideal case, the left and right images are perfectly aligned, the depth of spatial can be estimated from the baseline



(a) (b)

Figure 3.5: (a) Depth estimation from triangulation of stereo camera case, (b) Depth ambiguity in the estimation of a pair of images.

b and horizontal disparity along x-axis of the correspondences $d(u, u')$:

$$z^{\mathbf{p}}_{\mathcal{C}_1} = z^{\mathbf{p}}_{\mathcal{C}_2} = \frac{b \cdot f}{d(u, u')} = \frac{b \cdot f}{x^{\mathbf{p}}_{\mathcal{C}_1} - x^{\mathbf{p}}_{\mathcal{C}_2}}, \tag{3.13}$$

where the focal length f, the spatial point $\mathbf{p}$ and its projection points u, u' are defined the same as above section.

Similar to stereo settings, the triangulation of monocular camera case consists of a pair of close images but without a fixed known baseline and angle of views. This definitely increases the difficulty in motion and structure estimation in monocular case. If relying on visual information totally, sufficient correspondence points are required to solve the triangulation equations, thus bringing in more computational burden and more accumulated errors in calculation. However, the relative position between different camera views can also be acquired from other prior knowledge, such as the estimates of an inertial-based motion sensor.

On the other aspect, the depth estimates are dramatically affected by the spatial distance between the points and the baseline of image pairs. As can be seen in figure 3.5(b), all the points contained in the marked area, which includes a wide range of depth values, can be referred to the same pair of pixel units. Moreover, image blurs which mostly happen when using mobile platforms can cause even worse results. Thus, more techniques for accurate correspondence matching, such as outlier rejections, are required for later motion estimation.

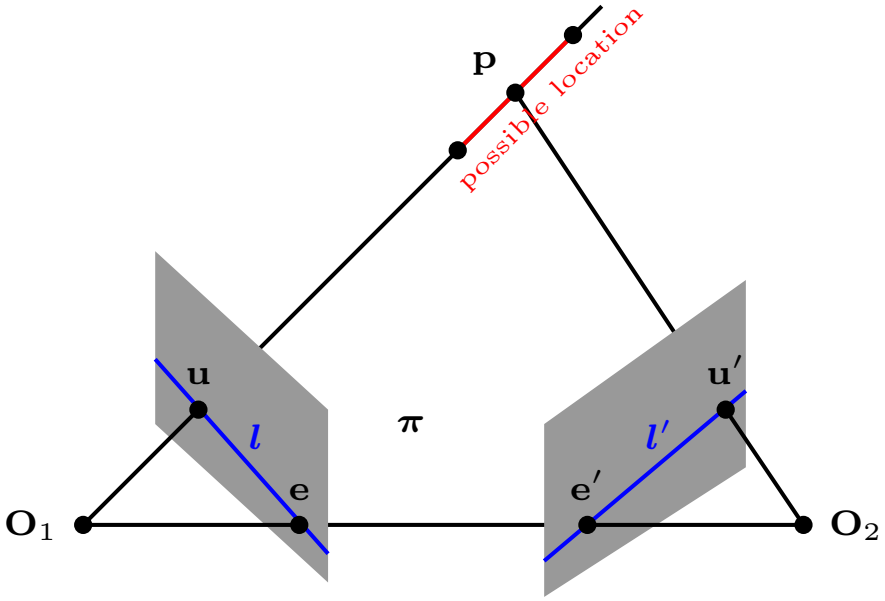


Figure 3.6: Point correspondence in epipolar geometry.

3.1.4.2 Epipolar geometry

To study the geometrical relationship of two views, the epipolar geometry is an essential theory applied in correspondence search in most visual-based tracking tasks. As shown in figure 3.6, the image points $\mathbf{u}$, $\mathbf{u}'$, space point $\mathbf{p}$, and camera centres $\mathbf{O}$ are coplanar. Denote this plane as $\boldsymbol{\pi}$. From the theory of back projection from above section, it is clear that the rays back projected from $\mathbf{u}$ and $\mathbf{u}'$ should intersect at $\mathbf{p}$, and these two rays are also coplanar in plane $\boldsymbol{\pi}$. The relationship can be expressed as a unique 3×3 homogeneous matrix, which is called *fundamental matrix* $\mathbf{F}$, and the image points satisfy the relationship as follows,

$$\mathbf{u}'^{T}\mathbf{F}\mathbf{u} = \mathbf{0}, \tag{3.14}$$

where $\mathbf{F}$ has seven degrees of freedom with the rank of two, and does not include the scale parameter. Actually, fundamental matrix maps point to line between two views. As shown in figure 3.6, a point in the first image $\mathbf{u}$ defines a line in the second $\mathbf{l}' = \mathbf{F}\mathbf{u}$, which is the *epipolar line* of $\mathbf{u}$. If $\mathbf{l}$ and $\mathbf{l}'$ are the corresponding epipolar lines, then any point $\mathbf{u}$ on $\mathbf{l}$ is projected on the same line $\mathbf{l}'$. The *epipole* $\mathbf{e}$ defines the intersection of the line joining the optical centres with the image plane. The epipole is also the projection of the optical centre of the other camera view. On the other hand, from equation (3.11), we know that a normalized back projection

$(f = 1, s_\theta = 0)$ maps the points in pixel frame to a normalized image frame as

$$\hat{\mathbf{u}} = \mathbf{K}^{-1}\mathbf{u}, \quad \hat{\mathbf{u}}' = \mathbf{K}^{-1}\mathbf{u}'. \tag{3.15}$$

The *essential matrix* $\mathbf{E}$ is a special case of the fundamental matrix, and they are related by the intrinsic matrix:

$$\mathbf{E} = \mathbf{K}'^{T}\mathbf{F}\mathbf{K}, \tag{3.16}$$

which actually defines the geometric relationship between normalized pixel points as $\hat{\mathbf{u}}'^{T}\mathbf{E}\hat{\mathbf{u}} = \mathbf{0}$. The essential matrix $\mathbf{E}$ is in the dimension of 3×3 having five degrees of freedom, i.e. three for rotation and two for the direction of translation. However, the translation is defined only up-to-scale, i.e. a scale parameter makes the estimated translation proportion to the real one.

For visual-based pose recovery tasks, assume $\mathbf{R}$ and $\mathbf{t}$ are relative rotation and translation between two views, $\mathbf{O}_1$ and $\mathbf{O}_2$. The projection matrices $\mathbf{M}$ and $\mathbf{M}'$, which map the pixel points in each image plane to the same 3D point in the first camera frame $\mathcal{C}_1$, can be written as

$$\mathbf{M} = \mathbf{K}[\mathbf{I} \quad \mathbf{0}]_{3\times4} \quad \text{and} \quad \mathbf{M}' = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]_{3\times4}. \tag{3.17}$$

Thus combining above with the equations of (3.10), (3.12) and (3.15), the essential matrix can also be written as

$$\mathbf{E} = [\mathbf{t}\times]\mathbf{R} = \mathbf{R}[\mathbf{R}^{T}\mathbf{t}\times], \tag{3.18}$$

where the $\lfloor\cdot\times\rfloor$ is the skew symmetric operation.

If the intrinsic parameters are known, the essential or fundamental matrices are solved via a least squares solution. For the simplest case, where no additional information other than the correspondence point sets are known, a minimum of eight correspondences is required to determine the essential or fundamental matrix uniquely, as the matrix is determined up to scale. More details about such method can be referred to the tutorial of [10]. The theory of epipolar geometry is used later in the thesis as geometric constraints to remove outliers.

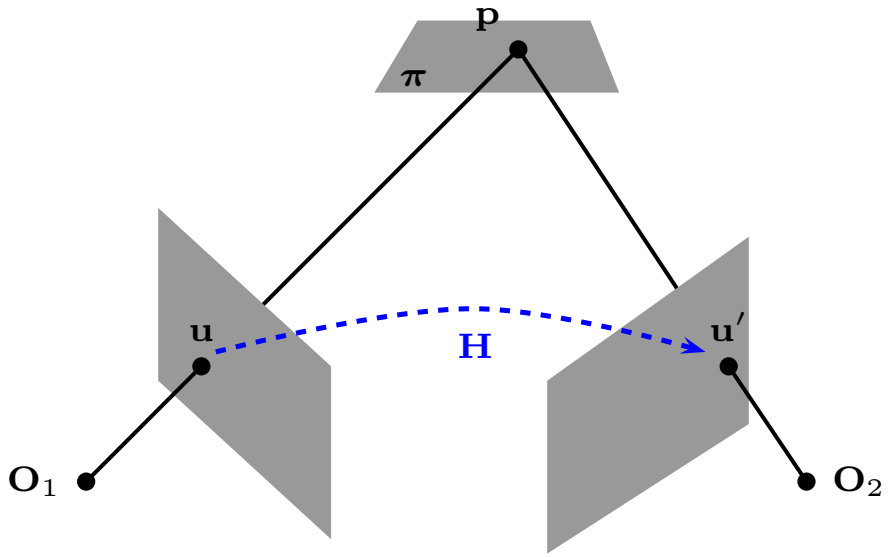


Figure 3.7: Illustration of Homography.

3.1.4.3 Homography matrix

Similar to the fundamental and essential matrices, there exists *homography matrix* **H** that defines a linear transformation relation between any two images, which only holds exactly when the imaged scene is planar or almost planar, as shown in figure 3.7.

The direct mapping correspondences $\mathbf{u}$ and $\mathbf{u}'$ by homography is written as

$$\mathbf{u}' \simeq \mathbf{H}\mathbf{u}. \tag{3.19}$$

If $\mathbf{n}$ is a unit vector normal to the plane $\boldsymbol{\pi}$ in camera frame $\mathcal{C}_1$, d is the perpendicular distance of the plane from first camera centre and every point in the plane satisfies $d = \mathbf{x}^{\mathbf{P}}_{\mathcal{C}_1} \cdot \mathbf{n}$ and $\frac{\mathbf{n}^T \mathbf{x}^{\mathbf{P}}_{\mathcal{C}_1}}{d} = 1$, then the 3×3 homography matrix $\mathbf{H}$ can be expressed as

$$\mathbf{H} = \mathbf{K}(\mathbf{R} - \frac{\mathbf{t}\mathbf{n}^T}{d})\mathbf{K}^{-1} \tag{3.20}$$

where $\mathbf{R}$ and $\mathbf{t}$ are the rotation and translation between two views with defined $\mathbf{K}$ as camera intrinsic matrix.

If rewrite the homography matrix in equation (3.19) with a scale parameter λ, we can have the equation as

$$\lambda \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{3.21}$$

This expression can generate two linear equations with a unit vector constraints to omit the scale factor,

$$\begin{aligned} u'(h_{31}u + h_{32}v + h_{33}) - h_{11}u - h_{12}v - h_{13} = 0 \\ v'(h_{31}u + h_{32}v + h_{33}) - h_{21}u - h_{22}v - h_{23} = 0 \\ h_{11}^2 + h_{12}^2 + h_{13}^2 + h_{21}^2 + h_{22}^2 + h_{23}^2 + h_{31}^2 + h_{32}^2 + h_{33}^2 = 1 \end{aligned} \tag{3.22}$$

As we can see from above equations, the planar homography matrix has 8 DoF. If n points generate $2n$ linear equations, the solution for $\mathbf{H}$ needs at least four correspondences. Actually, it can be regarded as finding the solution for least square problem as $\mathbf{AH} = \mathbf{0}$, where a solution can be found through Singular Value Decomposition (SVD) method.

The homography matrix is commonly applied to the spatial point sets lying in a common plane. It can be regarded as an additional constraint for pose estimation problems especially when the fundamental matrix based estimation fails. Additionally, it is useful in building a planar mosaic map.

3.1.5 Subpixel technique

In the projection model of the above sections, the value of a pixel is often defined as a positive number in an integer position, which is exactly matched with an image matrix. From another point of view, the image can also be expressed as a function reflecting the photometric properties of an image:

$$\mathbf{I} : \mathbb{R}^2 \to \mathbb{R}_+; (u, v) \to \mathbf{I}(u, v). \tag{3.23}$$

In practice, the projective calculation often results in a non-integer value, meaning the position of projected point does not exactly fit the image matrix unit. The most simple approach to get the intensity of this projected point is to adopt the value from the closest neighbouring pixels. However, it will bring in accumulated error in alignment when the calculation goes on. In order to get a more accurate intensity value, the *image interpolation* in subpixel is required. It can be shown in figure 3.8, the resulting projected point at (u', v') will adopt the value of pixel nearby, because it is located within the region of (u_1, v_1). But the intensity value of this neighbour pixel not can full represent that of the resulting position. For a more reasonable value, the information from other nearby pixels also should be considered.

Among all interpolation techniques, the bilinear approach is most commonly

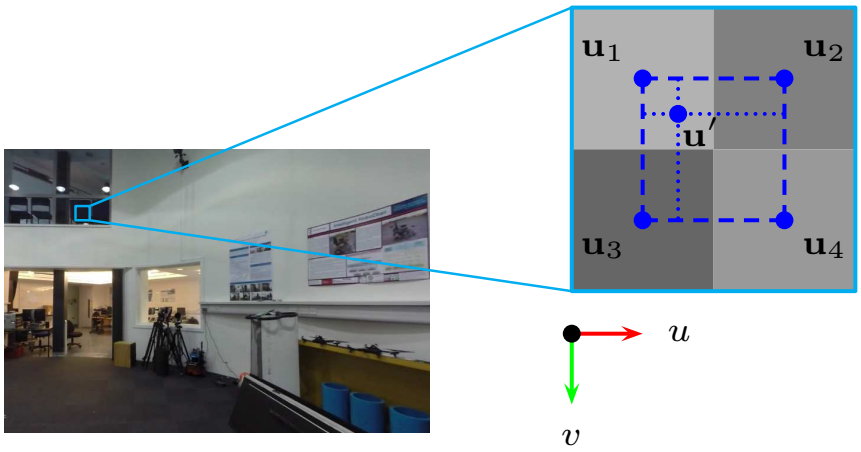


Figure 3.8: Interpolation for intensity estimation in subpixel region. The estimated point is located at the position $\mathbf{u}'$ with four close neighbours $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4$.

used (see figure 3.8), which computes the intensity of a resulting pixel $\mathbf{u}' = (u', v')$ at a non-integer position by combing its four nearest neighbours $\mathbf{u}_1 = (u_1, v_1), \mathbf{u}_2 = (u_2, v_2), \mathbf{u}_3 = (u_3, v_3), \mathbf{u}_4 = (u_4, v_4)$. If the position of $\mathbf{u}'$ is expressed with respect to the positions of four neighbours as

$$\mathbf{u}' = \mathbf{u}_1 + \alpha_u(\mathbf{u}_2 - \mathbf{u}_1) + \alpha_v(\mathbf{u}_3 - \mathbf{u}_1), \tag{3.24}$$

then the intensity is computed as a linear form first along u then v axis:

$$\begin{aligned} \mathbf{I}(u', v') = \alpha_v \left((1-\alpha_u)\mathbf{I}(u_1, v_1) + \alpha_u \mathbf{I}(u_2, v_2)\right) \\ +(1-\alpha_v)\left((1-\alpha_u)\mathbf{I}(u_3, v_3) + \alpha_u \mathbf{I}(u_4, v_4)\right). \end{aligned} \tag{3.25}$$

By applying the interpolation approach, the estimated intensity value of projected point with higher accuracy can be achieved by gathering several neighbouring pixels instead of just one. This technique is important for almost all the methods which highly depend on photometric information.

3.2 Kinematics for Mobile Platforms

In this section, the pose expressions and frame notations for a visual and inertial combined system will be introduced, which builds up the basic motion model and spatial relationship for later chapters. Here we will go over from basic pose expressions to minimal expression with their properties. Notably, different expressions

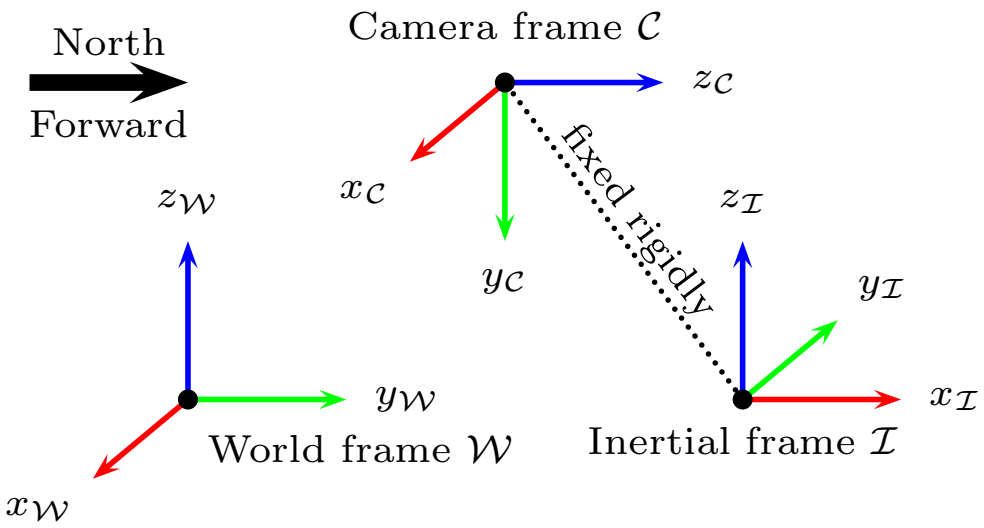


Figure 3.9: Frames definition for visual and inertial combined system.

for the rigid body rotation and transformation will be illustrated as a reference for algorithms development.

3.2.1 Frame conventions

Following the frame definition in section 3.1.1, we can further add an *inertial frame* $\mathcal{I}$ to the visual system, which is rigidly attached to an IMU sensor and regarded as a local motion reference, as shown in figure 3.9.

The orientation of inertial frame follows the convention that x-axis pointing to the forward of IMU sensor, the y-axis pointing to leftwards and the z-axis upwards.

3.2.2 Pose expression

When describing the state of a mobile platform, its position and orientation should be given in a local or global frame. Particularly, a couple of rotation and transition parameters $\{\boldsymbol{R}, \boldsymbol{t}\}$[1], need to be expressed between local attached frames like camera frame $\mathcal{C}$ or inertial frame $\mathcal{I}$ and global unify frames like initial camera frame $\mathcal{C}_0$ and world frame $\mathcal{W}$. In the previous section of 3.1.4, we have discussed a basic pose transition process as projection. In this section, we will recall the details in pose expressions to build the basis for the visual-inertial odometry.

[1]The pose notations here for the whole visual-inertial system is slightly different from those in visual projection section in the aspect of font format. Here is in bold italic type for distinguishing different application backgrounds.

In general, the full transformation of a rigid body includes a 3×1 translation vector $\boldsymbol{t}$ and a 3×3 rotation matrix $\boldsymbol{R}$. Between two different frames, the transformation of a spatial point $\mathbf{p}$ can be expressed as

$$\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} = \boldsymbol{R}\mathbf{x}_{\mathcal{C}}^{\mathbf{p}} + \mathbf{t}, \tag{3.26}$$

To reverse the transformation, from frame $\mathcal{W}$ to $\mathcal{C}$, the expression becomes

$$\mathbf{x}_{\mathcal{C}}^{\mathbf{p}} = \boldsymbol{R}^T\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} - \boldsymbol{R}^T\mathbf{t}. \tag{3.27}$$

It is easy to rewrite above equations (3.26) and (3.27) as homogeneous form for clear expression and calculation. The transformation from frame $\mathcal{C}$ to $\mathcal{W}$ can be rewritten as

$$\begin{bmatrix} \mathbf{x}_{\mathcal{W}}^{\mathbf{p}} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathcal{C}}^{\mathbf{p}} \\ 1 \end{bmatrix}, \tag{3.28}$$

and the reverse transformation becomes

$$\begin{bmatrix} \mathbf{x}_{\mathcal{C}}^{\mathbf{p}} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^T & -\boldsymbol{R}^T\boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathcal{W}}^{\mathbf{p}} \\ 1 \end{bmatrix}. \tag{3.29}$$

In compact form, this transformation can be expressed as $\boldsymbol{T}_{\mathcal{C}\mathcal{W}}$ and the reverse $\boldsymbol{T}_{\mathcal{W}\mathcal{C}}$ as

$$\boldsymbol{T}_{\mathcal{C}\mathcal{W}} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \quad \text{and} \quad \boldsymbol{T}_{\mathcal{W}\mathcal{C}} = \boldsymbol{T}_{\mathcal{C}\mathcal{W}}^{-1} = \begin{bmatrix} \boldsymbol{R}^T & -\boldsymbol{R}^T\boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathbb{R}^{4\times4}. \tag{3.30}$$

Thus the spatial pose transformation of any mobile platform can be written as a unified form with notation superscripts from right to left as indicating the transformation direction:

$$\mathbf{x}_{\mathcal{C}}^{\mathbf{p}} = \boldsymbol{T}_{\mathcal{C}\mathcal{W}}\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} \quad , \quad \mathbf{x}_{\mathcal{W}}^{\mathbf{p}} = \boldsymbol{T}_{\mathcal{W}\mathcal{C}}\mathbf{x}_{\mathcal{C}}^{\mathbf{p}}. \tag{3.31}$$

Although the rotation matrix seems very natural to describe rotations in spatial motion, the redundancy in matrix limits its efficiency in calculation. The rotation matrix contains nine elements in the matrix, but any rotation in 3D can be specified with just three parameters as minimal rotation representation.

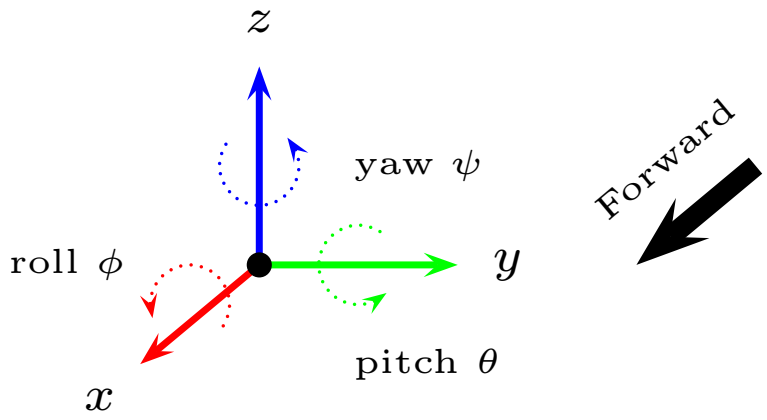


Figure 3.10: The Euler angles and rotation representation.

Euler angle	axis	axis positive
yaw (ψ)	x	vertical upwards
pitch (θ)	y	transversal leftward
roll (ϕ)	z	longitudinal forward

Table 3.1: Use Euler angles to describe the motion of mobile platforms.

3.2.3 Euler angle

Euler angles are the three rotated angles representing the orientation in 3D with three consecutive rotations around three different axes. There exists several conventions[1] on the order of these rotations. Take the most popular *zyx* convention as example (see figure 3.10), the first rotation is performed around z-axis as *yaw angle*, followed by the second rotation around y-axis as *pitch angle* and ended with the third one around x-axis as *roll angle*. In practice, with the x-axis pointing to the forward direction, it usually defines the y-axes pointing to the right to maintain the positive of pitch angle when aircraft raising. Thus making the z-axis points to downward, which is opposite to the defined z-axis of inertial frame (figure 3.9) applied in this thesis. Here, we follow the directions of forward, leftward and upward to represent the x, y, z-axes respectively of the inertial sensor and also the motion of mobile platform, which can be summarised as table 3.1.

From the defined Euler angles $\boldsymbol{e} = [\phi\ \theta\ \psi]^T$ corresponding to the roll, pitch and yaw angles of a reference frame, the rotation matrix $\boldsymbol{R}$ can be obtained by performing the product of three rotation matrices corresponding to the three elementary

[1]For more information, please refer to Euler angles in wikipedia.

rotations as

$$\mathbf{R}(\mathbf{e}) = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\theta\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\theta\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\theta\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

Although the vector of Euler angles describes a minimal representation for rotations, the essential drawback of presenting discontinuities makes it applied in limited ranges as $\phi \in [-\pi, \pi]$, $\theta \in [-\pi/2, \pi/2]$, and $\psi \in [0, 2\pi]$. Furthermore, for a target pose, the process of rotation can follow different axis orders in Euler expression. This will make the rotation solution not unique without knowing the rotational order, which limits the Euler angle expressions applied in practical computation to some extent.

3.2.4 Quaternion

The other way to express rotation is using the quaternion. Different from the definition of Euler angles, which needs to rotate around three different axes, the quaternion defines the rotation using only one axis $\boldsymbol{u}$ and the angle θ around it. It can be written as

$$\boldsymbol{q} = [\cos\theta, u_x \sin\theta, u_y \sin\theta, u_z \sin\theta]^T, \tag{3.32}$$

where $[u_x, u_y, u_z]^T = \boldsymbol{u}$ is a unit vector $||\boldsymbol{u}|| = 1$ and θ in radian is a real scalar. Furthermore, applying temporal differential of equation (3.32), we can achieve an equation as

$$2\boldsymbol{q}^* \otimes \dot{\boldsymbol{q}} = \boldsymbol{\omega}, \tag{3.33}$$

where $\boldsymbol{\omega} = [0, \omega_x, \omega_y, \omega_z]^T$ is the angular velocity along the rotation axis $\boldsymbol{u}$, $\boldsymbol{q}^*$ is the conjugated quaternion, and the multiplication $\otimes$ adopt the rules in quaternion algebra. With the normalization constraint as $\mathbf{q} \otimes \mathbf{q}^* = 1$, above equation (3.33) can be re-written as

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \boldsymbol{\omega} \tag{3.34}$$

If we express the angular velocity in the form as skew symmetric matrix $\lfloor \boldsymbol{\omega} \times \rfloor$, the derivation of quaternion can be expressed in a linear form as

$$\dot{\boldsymbol{q}} = \frac{1}{2}\lfloor \boldsymbol{\omega} \times \rfloor \boldsymbol{q}, \tag{3.35}$$

where

$$\lfloor \boldsymbol{\omega} \times \rfloor = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}.$$

It is useful in deriving the inertial-based dynamic model in visual and inertial odometry algorithms.

In mathematics, the quaternions are a number system that extends the complex numbers, which can be expressed as

$$\boldsymbol{q} = a + b\boldsymbol{i} + c\boldsymbol{j} + d\boldsymbol{k}, \tag{3.36}$$

where $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$ are the imaginary unit along three perpendicular direction with the definition as $\boldsymbol{i}^2 = -1$, $\boldsymbol{j}^2 = -1$, $\boldsymbol{k}^2 = -1$, $\boldsymbol{ij} = -\boldsymbol{ji} = \boldsymbol{k}$, $\boldsymbol{jk} = -\boldsymbol{kj} = \boldsymbol{i}$ and $\boldsymbol{ki} = -\boldsymbol{ik} = \boldsymbol{j}$.

The quaternion can be also expressed in the form of a vector as

$$\boldsymbol{q} = [a, b, c, d]^T \in \mathbb{R}^4, \tag{3.37}$$

and the conjugation of such quaternion is defined by $\boldsymbol{q}^* = [a, -b, -c, -d]^T$. The norm of quaternion is defined as $||\boldsymbol{q}|| = \sqrt{\boldsymbol{q} \otimes \boldsymbol{q}^*}$. If $||\boldsymbol{q}|| = 1$, the quaternion is said to be normalized.

In the problem of pose estimation, a basic rotation can be abstracted as a vector $\boldsymbol{v} = 0 + v_x\boldsymbol{i} + v_y\boldsymbol{j} + v_z\boldsymbol{k}$ in spatial frame turning into another vector $\boldsymbol{v}'$ by performing a operation as

$$\boldsymbol{v}' = \boldsymbol{q} \otimes \boldsymbol{v} \otimes \boldsymbol{q}^*. \tag{3.38}$$

If we adopt the quaternion expressed in equation (3.36), then expand the equation (3.38) linear to $\boldsymbol{v}$. It will lead to a 3×3 matrix, which is the same rotation matrix as defined in early section. Here we note is as

$$\mathrm{C}_{\boldsymbol{q}} = \begin{bmatrix} a^2 + b^2 - c^2 - d^2 & 2 \cdot (bc - ad) & 2 \cdot (bd + ac) \\ 2 \cdot (bc + ad) & a^2 - b^2 + c^2 - d^2 & 2 \cdot (cd - ab) \\ 2 \cdot (bd - ac) & 2 \cdot (cd + ab) & a^2 - b^2 - c^2 + d^2 \end{bmatrix}.$$

This is the equation links quaternion and rotation matrix in practical algorithm computation. The reverse rotation is actually the conjugate quaternion as $\boldsymbol{q}^*$, thus $\mathrm{C}_{\boldsymbol{q}^*}$ is equivalent to $\mathrm{C}_{\boldsymbol{q}}^T$. It should be noticed that the negative quaternion repre-

sents exactly the same rotation as the original one, i.e. $C_{-\boldsymbol{q}} = C_{\boldsymbol{q}}$. This ambiguity in expression also indicates that the quaternion is *not a minimal representation* for rotations. The results should be further filtrated to get a unique result.

Although there are three ways to express rotation here, i.e. rational matrix, Euler angle and the quaternion, they perform different properties in practical computation. The rotation matrix is the linear method to perform the rotation to the points and vectors in spatial frames. It is straightforward and explicit. The Euler angle owns merit of easy visualization and understanding, especially in the expression of assumption small change in angles. On the other hand, the quaternion has the ability of continuous and continuously derivable, which is useful in deriving the system formulations and storing orientation information in the later section.

For more practical equations of mutual transformation between different rotation expressions, please refer to appendix.

3.2.5 Lie group

Lie group is popularly applied in the robotic research as it brings in practical and efficient characters in optimisation and filtering calculation. In Euclidean space, if a small incremental value $\boldsymbol{\delta}$ is added into the current estimated $\boldsymbol{x}_k$, it is simply expressed as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{\delta}. \tag{3.39}$$

However, if a minimal rotation represented by a vector $\boldsymbol{w}$ such as Euler angles, the small updating value $\boldsymbol{\delta}$ not can be added by directly using $\boldsymbol{w} + \boldsymbol{\delta}$. While in the research of mobile platforms, it is unavoidable to update rotation information during incremental calculation. Fortunately, the expressions in Lie group break the limitation in the computation for the group of rotation SO(3), making the error form of a state vector can be written in a unified linear matrix for further algorithm derivation and analysis. This method has been adopted in chapter 4.

In the expression of special Euclidean group SE(3), the rotation matrix $\boldsymbol{R}$ and a translation vector $\boldsymbol{t}$ are adopted, as shown in previous section 3.2.2. If a rotation in 3D space is parametrized using as a vector $\boldsymbol{w} = [w_x, w_y, w_z]^T$ and translated expressed as $\boldsymbol{v} = [v_x, v_y, v_z]^T$. The vector $\boldsymbol{\xi} = [\boldsymbol{w}, \boldsymbol{v}]^T$ can minimally express the pose with the exponential map. Among them, the rotation angle is defined as the magnitude $\boldsymbol{\theta} = ||\boldsymbol{w}||$ with rotation axis as $\boldsymbol{u} = \frac{\boldsymbol{w}}{\boldsymbol{\theta}} = [u_x, u_y, u_z]^T$. The transformation $\boldsymbol{R} = \exp(\lfloor \boldsymbol{w} \times \rfloor)$ rotates a point $\mathbf{x}$ the angle $\boldsymbol{\theta}$ around the axis $\boldsymbol{u}$.

It has been proved that the minimal expression $\boldsymbol{\xi}$ in the tangent space se(3) can be fully mapped into the elements in SE(3) [94]. Here, we only state the properties of commutative and additivity for later usage. For more properties of Lie groups, please refer to mathematical books or practical tutorials. With the exponent and its reverse logarithm operation, the consecutive poses are addable through

$$\boldsymbol{\xi}_{\mathcal{W}\mathcal{C}_2} = \boldsymbol{\xi}_{\mathcal{W}\mathcal{C}_1} \odot \boldsymbol{\xi}_{\mathcal{C}_1\mathcal{C}_2} = \log\left(\exp\left(\boldsymbol{\xi}_{\mathcal{W}\mathcal{C}_1}\right) \cdot \exp\left(\boldsymbol{\xi}_{\mathcal{C}_1\mathcal{C}_2}\right)\right). \tag{3.40}$$

For a small value for update in consecutive steps, it can be expressed simply as

$$\boldsymbol{\xi}_{k+1} = \delta\boldsymbol{\xi} \odot \boldsymbol{\xi}_k. \tag{3.41}$$

3.3 Optimisation and Filtering based Frameworks

After introducing and defining visual geometry and kinematics for mobile platforms, in this section the optimisation and filtering based frameworks for motion estimation will be discussed from the perspectives of optimisation and general filtering. This will pave the way for further algorithm developing and analysis in following chapters.

3.3.1 Optimisation based framework

The goal of an optimisation problem is to achieve an *optimised* estimate after iterative calculation, which starts from approximate initializations and ends with a convergent result of a cost function. As a quick example, we can find the optimal value of a single real variable x determined by a cost function as $f(x) = x^2 + b$ at the point $x = 0$, which minimises the cost function to the constant $b \in \mathbb{R}$. Optimisation problems can be categorised into linear or nonlinear types, according to the complexity of an optimised function. In above case, the x^2 term makes the problem nonlinear. Generally, linear problem can be solved in a single optimisation step through getting the solutions from a set of equations, while for nonlinear cases, the optimal solution is often not directly observable and an additional linearisation step at current optimal point is required. Only by getting the estimates from a linearised point is it possible to perform a forward step in optimisation and determine whether the cost function is decreasing. The localisation and motion estimation problems of visual-inertial odometry is a highly nonlinear optimisation, thus numerous techniques concerning finding the most accurate solutions efficiently are widely studied. Here, we start from the basic least square problem towards a goal of building a

motion estimation cost function under the concept of optimisation.

3.3.2 Least square minimisation

Given a set of measurements $\boldsymbol{z}$ as point locations in image plane as shown in figure 3.3, an observation function $\boldsymbol{h}(\cdot)$ that maps a set of state parameters including the spatial locations of corresponding points, as $\mathbf{x}_{\mathcal{W}}^{\mathbf{p}}$, and camera positions as $\boldsymbol{\xi}$, which can be written as

$$\boldsymbol{z} = \boldsymbol{h}(\boldsymbol{\xi}, \mathbf{x}_{\mathcal{W}}^{\mathbf{p}}) + \mathbf{v} \tag{3.42}$$

The goal seeks to recover an estimation set of $\hat{\boldsymbol{\xi}}$ with known $\mathbf{x}_{\mathcal{W}}^{\mathbf{p}}$. The noise $\mathbf{v}$ not can be observed directly but modelled as Gaussian distribution.

A cost function $\boldsymbol{E}$ that minimises the output to get the best estimates can be formalised as

$$\boldsymbol{\xi}^* = \arg\min \boldsymbol{E}\left(\boldsymbol{\xi}, \mathbf{x}_{\mathcal{W}}^{\mathbf{p}}\right), \tag{3.43}$$

where $\boldsymbol{\xi}^*$ indicates the local minimum of the optimisation. There are various methods to express the cost function $\boldsymbol{E}$, but the most applicable is the sum of squares of the residual error as

$$\boldsymbol{E} = \sum ||\boldsymbol{z} - \hat{\boldsymbol{z}}||^2 = \sum ||\boldsymbol{\epsilon}||^2, \tag{3.44}$$

where $\boldsymbol{\epsilon}$ consists of the difference between the actual observation $\boldsymbol{z}$ and the estimated $\hat{\boldsymbol{z}}$ with given parameters $\mathbf{x}_{\mathcal{W}}^{\mathbf{p}}$.

If the measurement function $\boldsymbol{h}(\cdot)$ is linear without constraints, the solution can be directly given via a method like SVD: for a linear system written as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ where $\boldsymbol{A} \in \mathbb{R}^{m \times n} (m \geq n)$, there exists a factorisation of system matrix $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*$ and its inverse matrix $\boldsymbol{A}^{-1} = \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^*$ calculating the solution directly as $\mathbf{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$. The matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ is a non-negative diagonal matrix containing the singular values of the matrix $\boldsymbol{A}$.

3.3.3 Nonlinear least square minimisation

However, the function $\boldsymbol{h}(\cdot)$ is often nonlinear, such as defining the forward and backward projection process in above section. In order to achieve a satisfactory solution, a local assumption for local linearity should be performed. Then the estimated results proceed to a process of iterative refinement.

Following from above with partially known parameter $\mathbf{x}_{\mathcal{W}}^{\mathbf{p}}$, when given an initial estimate of the parameters $\boldsymbol{\xi}_0$, we can assume that the function $\boldsymbol{h}(\cdot)$ with a small

perturbation $\boldsymbol{\delta}$ can be approximated by a Taylor series at $\hat{\boldsymbol{\xi}}_0$ as

$$\boldsymbol{h}(\hat{\boldsymbol{\xi}}_0 + \boldsymbol{\delta}, \mathbf{x}_{\mathcal{W}}^{\mathrm{p}}) = \boldsymbol{h}(\hat{\boldsymbol{\xi}}_0, \mathbf{x}_{\mathcal{W}}^{\mathrm{p}}) + \boldsymbol{\delta}\boldsymbol{h}'(\hat{\boldsymbol{\xi}}_0, \mathbf{x}_{\mathcal{W}}^{\mathrm{p}}) + \boldsymbol{\delta}\frac{\boldsymbol{h}''(\hat{\boldsymbol{\xi}}_0, \mathbf{x}_{\mathcal{W}}^{\mathrm{p}})}{2} + \cdots \tag{3.45}$$

Usually, except for the first term, the remaining terms of equation (3.45) are high order terms with little additional benefits if assuming the function is locally smooth. To save the redundancy in computation, the function can be approximated as

$$\boldsymbol{h}(\hat{\boldsymbol{\xi}}_0 + \boldsymbol{\delta}, \mathbf{x}_{\mathcal{W}}^{\mathrm{p}}) = \boldsymbol{h}(\hat{\boldsymbol{\xi}}_0, \mathbf{x}_{\mathcal{W}}^{\mathrm{p}}) + \boldsymbol{J}\boldsymbol{\delta}, \tag{3.46}$$

where $\boldsymbol{J}$ is the Jacobian matrix defined as

$$\boldsymbol{J} = \dot{\boldsymbol{h}} = \frac{\partial \boldsymbol{h}}{\partial \hat{\boldsymbol{\xi}}} \tag{3.47}$$

From this approximation, the next estimation $\boldsymbol{\xi}_{i+1}$ can be achieved through

$$\hat{\boldsymbol{\xi}}_{i+1} = \hat{\boldsymbol{\xi}}_i \oplus \boldsymbol{\delta}_i, \tag{3.48}$$

which is subject to

$$\boldsymbol{E}(\hat{\boldsymbol{\xi}}_{i+1}) < \boldsymbol{E}(\hat{\boldsymbol{\xi}}_i) \tag{3.49}$$

for acceptance of the update. The final solution will be achieved by finding each incremental value making the cost function become a minimum within the searching area. It should be noticed that here for simplifying expression, the additive symbol $\oplus$ means general add operation and it can be applied in a rotation as stated in previous section 3.2.5.

By using Gauss-Newton method, the optimal solutions for cost function (3.43) can be iteratively computed by calculating the incremental value from

$$\boldsymbol{J}^T\boldsymbol{J}\boldsymbol{\delta} = -\boldsymbol{J}^T\boldsymbol{\epsilon} \tag{3.50}$$

3.3.4 Levenberg-Marquardt solution

In the above process, the nonlinear function has been transferred to a linear one through the local use of the Taylor series of equation (3.45). The solution from the most generic Gauss-Newton method is considered completely linear in the appropriate region. This means that convergence will be quadratic near the solution, but if given a poor initialization or the function has many local minima, the optimisation

will often fail to converge appropriately. The step length, i.e. the incremental value $\boldsymbol{\delta}$, in each iteration getting from equation (3.50), will tend to oscillate around the solution.

By applying a damping factor β into equation (3.50), the increment $\boldsymbol{\delta}$ becomes adjustable

$$(\boldsymbol{J}^T\boldsymbol{J} + \beta\mathbf{I})\boldsymbol{\delta} = -\mathbf{J}^T\boldsymbol{\epsilon}, \tag{3.51}$$

where $\mathbf{I}$ is the identity matrix. If the reduction speed of cost function $\boldsymbol{E}(\cdot)$ is rapid, a smaller value will be adopted, making this method closer to generic Gauss-Newton method. But if an iteration gives insufficient reduction in the residual, β can be increased, giving a step closer to the gradient descent direction as $(\beta\mathbf{I})\boldsymbol{\delta} = -\boldsymbol{J}^T\boldsymbol{\epsilon}$. In the iterative calculation, if either the incremental value $\boldsymbol{\delta}$ or the reduction of sum of squares from the latest estimate $\boldsymbol{\xi}_{i+1}$ falls below a predefined threshold, the iteration stops and the last estimate is considered to be the solution.

However, if the value of damping factor β is large, inverting the matrix of $(\boldsymbol{J}^T\boldsymbol{J} + \beta\mathbf{I})$ becomes not suitable in iteration. The Levenberg-Marquardt solution scales each component of the gradient according to the curvature, thus making larger movement along the directions where the gradient is smaller. This avoids slow convergence in the direction of a small gradient. Therefore, the final solution replaces the identity matrix $\mathbf{I}$ with the diagonal matrix $\diagdown(\cdot)$ consisting of the diagonal elements of $\boldsymbol{J}^T\boldsymbol{J}$, expressed as

$$(\boldsymbol{J}^T\boldsymbol{J} + \beta\diagdown(\boldsymbol{J}^T\boldsymbol{J}))\boldsymbol{\delta} = -\boldsymbol{J}^T\boldsymbol{\epsilon}. \tag{3.52}$$

In practice, by carefully adjusting the damping factor β, an algorithm can be generated that converges rapidly in linear regions and improves the convergence in nonlinear regions. One possible strategy can be performed as the flowchart in figure 3.11 during iterations.

3.3.5 Motion estimation while mapping

Following the definition of above sections, we can further explicitly illustrate the optimal process in the visual-based motion estimation here. In this case, the measurement function $\boldsymbol{h}(\cdot)$ is defined as the forward and backward process. Thus iterative nonlinear optimization is formulated to find the camera pose changes and point coordinates by minimising a re-projection error of observed regions in images.

$$\boldsymbol{\xi}, \mathbf{x}_{\mathcal{W}}^{\mathbf{p}} = \arg\min \boldsymbol{E}\left(\hat{\boldsymbol{\xi}}, \hat{\mathbf{x}}_{\mathcal{W}}^{\mathbf{p}}\right). \tag{3.53}$$

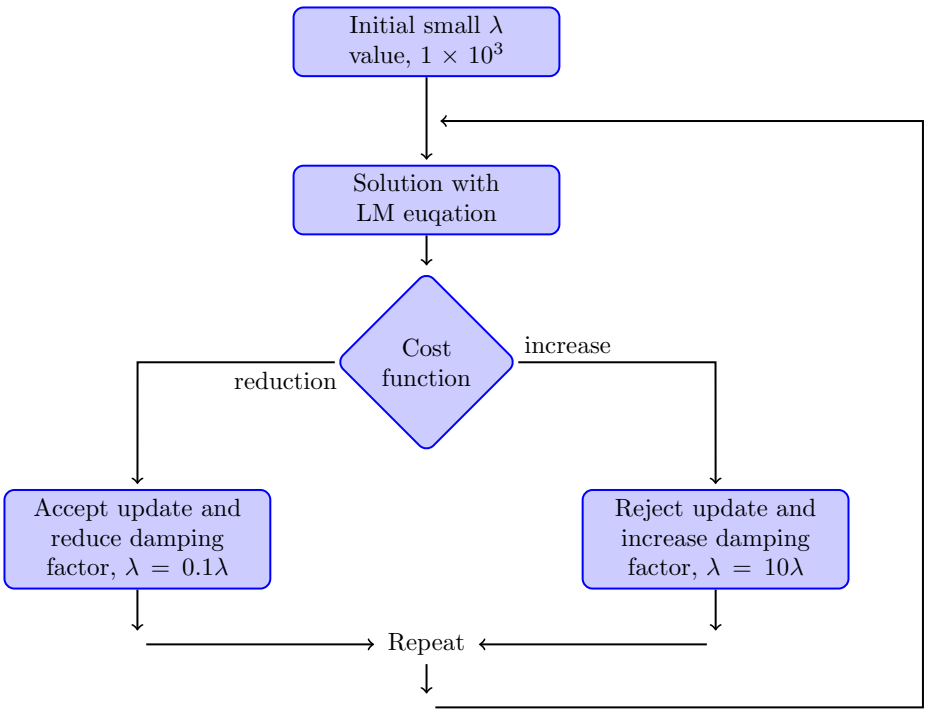


Figure 3.11: An example strategy for damping factor in Levenberg-Marquardt solution.

Different from the problem description in section 3.3.2, the reference map is unknown and should be simultaneously estimated in iterative process,

$$\boldsymbol{\epsilon} = \boldsymbol{z} - \hat{\boldsymbol{z}}(\hat{\boldsymbol{\xi}}, \hat{\mathbf{x}}_{\mathcal{W}}^{\mathbf{p}}). \tag{3.54}$$

The cost function $\boldsymbol{E}(\hat{\boldsymbol{\xi}}, \hat{\mathbf{x}}_{\mathcal{W}}^{\mathbf{p}})$ can be written analytically as the sum of all squared errors with weighting parameters

$$\boldsymbol{E}\left(\hat{\boldsymbol{\xi}}, \hat{\mathbf{x}}_{\mathcal{W}}^{\mathbf{p}}\right) = \sum_{i=1}^{n}\sum_{j=1}^{m} \boldsymbol{\epsilon}_{i,j}^{\mathbf{T}} w_{i,j} \boldsymbol{\epsilon}_{i,j}, \tag{3.55}$$

where j from 1 to m is the index of points within a frame, and i is the number of frames indexing a set with size n. Thus, in different iterative periods, when given the pose tracking estimated $\hat{\boldsymbol{\xi}}$, the optimisation problem for mapping becomes

$$\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} = \arg\min \boldsymbol{E}\left(\hat{\boldsymbol{\xi}}, \mathbf{x}_{\mathcal{W}}^{\mathbf{p}}\right), \tag{3.56}$$

and given the mapping results $\hat{\mathbf{x}}_W^{\mathbf{p}}$, the optimisation problem for pose estimation is

$$\boldsymbol{\xi} = \arg\min \boldsymbol{E}\left(\boldsymbol{\xi}, \hat{\mathbf{x}}_W^{\mathbf{p}}\right).$$

Further applying Levenberg-Marquardt method, the problem becomes to find an increment $\boldsymbol{\delta}$ in each iterative step, then update the variables. By adding the weighting matrix $\boldsymbol{W}$, the solution to $\boldsymbol{\delta}$ is found by

$$\left(\boldsymbol{J}^T\boldsymbol{W}\boldsymbol{J} + \beta\mathbf{I}\right)\boldsymbol{\delta} = -\boldsymbol{J}^T\boldsymbol{W}\boldsymbol{\epsilon}.$$

The above process is the most common case used in feature-based visual measurement to jointly estimate the camera pose and scene structure, which will be used for comparison with our methods in later chapters.

3.3.6 Filtering based framework

The filtering based methods are further defined following the concept of a general Markov system. The belief state can be updated from time step k to time step $k+1$ using a system model and a measurement model. The former one describes the distribution of the current state from given previous state, expressed as $p(\boldsymbol{x}_k+1|\boldsymbol{x}_k)$, while the latter model describes the probability of making a particular measurement from a given system state, written as $p(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1})$. If we begin with a probability over state at time step k, which is conditioned on all of the measurements up to this time, then the state at time $k+1$ can be expressed as

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}) = \int p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}, \boldsymbol{x}_k)p(\boldsymbol{x}_k)d\boldsymbol{x}_k. \tag{3.57}$$

By applying Bayes' rule to the first probability in the above equation, we can get

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}) \propto \int p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}, \boldsymbol{x}_k)p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k)d\boldsymbol{x}_k. \tag{3.58}$$

Given the current state, the current measurement $\boldsymbol{y}_{k+1}$ is assumed to be independent of the previous state or previous measurements, so that $p(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1}, \boldsymbol{x}_k) = p(\boldsymbol{y}_{k+1}|\boldsymbol{x}_{k+1})$ and equation (3.58) becomes

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1}) \propto \int p(\boldsymbol{x}_{k+1}|\boldsymbol{y}_{k+1})p(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)p(\boldsymbol{x}_k)d\boldsymbol{x}_k. \tag{3.59}$$

The last probability term in above equation is the state at time step k, which indicates a recursive process for updating at each time step.

3.3.7 Kalman filter

In Kalman filtering problem, all uncertainties are considered as Gaussian distribution. The additional constraint of linear evolution and measurement equations leads to a finite dimension functional formulation of the whole recursive process, which is closed and provably optimal.

Consider the linear Gaussian system as

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{F}\boldsymbol{x}_k + \boldsymbol{G}\boldsymbol{\omega}_{k+1} \\ \boldsymbol{y}_{k+1} &= \boldsymbol{H}\boldsymbol{x}_{k+1} + \boldsymbol{\upsilon}_{k+1} \end{aligned} \tag{3.60}$$

with

$$\begin{aligned} p(\mathbf{x}_0) &= \mathcal{N}(\boldsymbol{x}_0 - \bar{\boldsymbol{x}}_0, \boldsymbol{P}_0) \\ p(\boldsymbol{\omega}_{k+1}) &= \mathcal{N}(\boldsymbol{\omega}_{k+1} - \mathbf{0}, \boldsymbol{Q}) \\ p(\boldsymbol{\upsilon}_{k+1}) &= \mathcal{N}(\boldsymbol{\upsilon}_{k+1} - \mathbf{0}, \boldsymbol{R}), \end{aligned}$$

where

$$\mathcal{N}(\boldsymbol{x} - \bar{\boldsymbol{x}}, \boldsymbol{P}) = \frac{1}{\sqrt{(2\boldsymbol{\pi})|\boldsymbol{P}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \bar{\boldsymbol{x}})^T \boldsymbol{P}^{-1}(\boldsymbol{x} - \bar{\boldsymbol{x}})\right) \tag{3.61}$$

is the PDF of an n-dimensional Gaussian variable $\boldsymbol{x}$ with the mean value as $\bar{\boldsymbol{x}}$ and covariance matrix $\boldsymbol{P}$, where the noise are independent to state variables.

The Kalmen filter is the set of equations to predict and update the mean and covariances matrix of system states.

3.3.7.1 Prediction step of KF

From the definitions of the mean $\bar{\boldsymbol{x}}$ and the covariances matrix $\boldsymbol{P}$ of a multi-dimensional variable $\boldsymbol{x}$,

$$\begin{aligned} \bar{\boldsymbol{x}} &= \mathtt{E}(\boldsymbol{x}), \\ \bar{\boldsymbol{P}} &= \mathtt{E}\left((\boldsymbol{x} - \bar{\boldsymbol{x}})(\boldsymbol{x} - \bar{\boldsymbol{x}})^T\right), \end{aligned} \tag{3.62}$$

and from the linear properties if the expectation operation $\mathtt{E}(\cdot)$ and the zero variance of independent variables is defined as $\mathtt{E}\left((\boldsymbol{x} - \bar{\boldsymbol{x}})(\boldsymbol{x}' - \bar{\boldsymbol{x}'})\right) = \mathbf{0}$, we have the

equations

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1|k} &= \boldsymbol{F}\hat{\boldsymbol{x}}_{k|k},\\ \hat{\boldsymbol{P}}_{k+1|k} &= \boldsymbol{F}\boldsymbol{P}_{k|k}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{Q}\boldsymbol{G}^T,\end{aligned} \tag{3.63}$$

where

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1|k} &= \mathtt{E}(\boldsymbol{x}_{k+1}|\boldsymbol{y}_0^k),\\ \hat{\boldsymbol{P}}_{k+1|k} &= \mathtt{E}\left((\boldsymbol{x}_{k+1} - \hat{\boldsymbol{x}}_{k+1|k})(\boldsymbol{x}_{k+1} - \hat{\boldsymbol{x}}_{k+1|k})^T\right).\end{aligned} \tag{3.64}$$

3.3.7.2 Update step of KF

If define the innovation as $\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}$, its covariance matrix can be derived as

$$\boldsymbol{S} = \boldsymbol{H}\boldsymbol{P}\boldsymbol{H}^T + \mathbf{R}, \tag{3.65}$$

which is obviously a zero mean Gaussian as $\mathcal{N}(\boldsymbol{z}-\mathbf{0}, \boldsymbol{S})$. The *Kalman gain* is defined as

$$\boldsymbol{K} = \boldsymbol{P}\boldsymbol{H}^T\boldsymbol{S}^{-1}. \tag{3.66}$$

Then the Kalman update equations can be written as

$$\begin{aligned}\hat{\boldsymbol{x}}_{k+1|k+1} &= \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{K}(\boldsymbol{y}_{k+1} - \boldsymbol{H}\hat{\boldsymbol{x}}_{k+1|k}),\\ \hat{\boldsymbol{P}}_{k+1|k+1} &= \hat{\boldsymbol{P}}_{k+1|k} - \boldsymbol{K}\boldsymbol{S}\boldsymbol{K}^T.\end{aligned} \tag{3.67}$$

3.3.8 Extended Kalman filter

If the linearity assumption is removed, then the local linearization around the most recent estimates are used to construct the Extended Kalman filter as sub-optimal. Consider the nonlinear Gaussian system as

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{\omega}_{k+1}),\\ \boldsymbol{y}_{k+1} &= \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{\upsilon}_{k+1},\end{aligned} \tag{3.68}$$

with

$$\begin{aligned}p(\mathbf{x}_0) &= \mathcal{N}(\boldsymbol{x}_0 - \bar{\boldsymbol{x}}_0, \boldsymbol{P}_0),\\ p(\boldsymbol{\omega}_{k+1}) &= \mathcal{N}(\boldsymbol{\omega}_{k+1} - \bar{\boldsymbol{\omega}}_{k+1}, \boldsymbol{Q}),\\ p(\boldsymbol{\upsilon}_{k+1}) &= \mathcal{N}(\boldsymbol{\upsilon}_{k+1} - \mathbf{0}, \boldsymbol{R}).\end{aligned}$$

It is different from the linear case here, the means of perturbations $\bar{\boldsymbol{\omega}}_{k+1} \neq \mathbf{0}$ is taken into account. At each time step k, the equations (3.68) are linearised around the most recent estimates, which is similar to the process (3.45) in optimisation. Then the prediction and update processes constitute the Kalman filter.

3.3.8.1 Prediction step of EKF

The evolution equation is linearised around the latest estimate and the known input with respect to the system state and the perturbation via Jacobian matrices,

$$\boldsymbol{F}_{\boldsymbol{x}} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}^T}\right|_{\hat{\boldsymbol{x}},\bar{\boldsymbol{\omega}}} \quad \text{and} \quad \boldsymbol{F}_{\boldsymbol{\omega}} = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\omega}^T}\right|_{\hat{\boldsymbol{x}},\bar{\boldsymbol{\omega}}}. \tag{3.69}$$

Thus the EKF prediction equations can be written as

$$\begin{aligned} \hat{\boldsymbol{x}}_{k+1|k} &= \boldsymbol{f}(\boldsymbol{x}_{k|k}, \boldsymbol{\omega}_{k+1}), \\ \hat{\boldsymbol{P}}_{k+1|k} &= \boldsymbol{F}_{\boldsymbol{x}_{k|k}} \boldsymbol{P}_{k|k} \boldsymbol{F}_{\boldsymbol{x}_{k|k}}^T + \boldsymbol{F}_{\boldsymbol{\omega}_{k+1}} \boldsymbol{Q}_{k+1} \boldsymbol{F}_{\boldsymbol{\omega}_{k+1}}^T. \end{aligned} \tag{3.70}$$

3.3.8.2 Update step of EKF

The measurement equation is linearised around the latest estimate with respect to the system state via the Jacobian matrix, written as

$$\boldsymbol{H}_{\boldsymbol{x}} = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}^T}\right|_{\hat{\boldsymbol{x}}}. \tag{3.71}$$

If the innovation is defined as $\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{h}(\hat{\boldsymbol{x}})$ with the covariance matrix written the same form as equation (3.65), which also follows the distribution as Gaussian $\mathcal{N}(\boldsymbol{z}, \boldsymbol{S})$, then the update equations can be written as

$$\begin{aligned} \hat{\boldsymbol{x}}_{k+1|k+1} &= \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{K}_{k+1} \left(\boldsymbol{y}_{k+1} - \boldsymbol{h}(\hat{\boldsymbol{x}}_{k+1|k})\right), \\ \hat{\boldsymbol{P}}_{k+1|k+1} &= \hat{\boldsymbol{P}}_{k+1|k} - \boldsymbol{K}_{k+1} \boldsymbol{S}_{k+1} \boldsymbol{K}_{k+1}^T. \end{aligned} \tag{3.72}$$

with the Kalman gain as

$$\boldsymbol{K}_{k+1} = \hat{\boldsymbol{P}}_{k+1|k} \boldsymbol{H}_{\boldsymbol{x}_{k+1|k}}^T \left(\boldsymbol{H}_{\boldsymbol{x}_{k+1|k}} \hat{\boldsymbol{P}}_{k+1|k} \boldsymbol{H}_{\boldsymbol{x}_{k+1|k}}^T + \boldsymbol{R}_{k+1}\right)^{-1}. \tag{3.73}$$

3.4 Summary

In this chapter, we firstly recall the geometrical relations in visual measurement and pose expression for visual and inertial systems. Starting from the basic forward projection process, the geometry constraints between frames and correspondences are discussed. Particularly, the cause of scale ambiguity is illustrated in the process of back projection. For more accurate estimation of pixel intensity, the subpixel interpolation technique is also introduced. Furthermore, starting from basic rotation matrix, two popular expressions for rotation are also introduced and compared. The minimal representation for the pose as a compact six element vector and its properties are presented at last. Moreover, based on the visual based motion estimation process and the definition of spatial transformation, the two most fundamental frameworks are presented from linear to nonlinear cases. All the technical preliminary discussed in this chapter forms the fundamental knowledge for a further exploration of visual-inertial algorithms in the following chapters.

Chapter 4

Direct Visual-Inertial Fusion in Multi-State Constraint Kalman Filter

4.1 Overview

In our research, *Visual Inertial Odometry* (VIO) technique is used to estimate the change of a mobile platform in position and orientation over time by using the measurements from onboard cameras and IMU. The process of achieving pose estimates is the strategy to use and manage both information resources. A filtering framework consists of a prediction step and an updating step. For a filtering based VIO approach, the inertial sensor can provide acceleration and rotational velocity measurements in three axes, which serve as the data to drive a dynamic model or the prior distribution for a 3D rigid body motion. This is motion prediction in propagation step. A camera can provide the angular and ranging measurements between spatial points and the mobile platform, which serve as the measurement model or likelihood distribution. This model will update the prediction results.

Traditional filtering methods only keep a certain number of variables in the state vector while the system is running. The multi-state constraint Kalman filter can alter the length of state vector while running. In our research, we only maintain recent inertial states, new poses of keyframes and successive frames in the current state. Our analysis of intrinsic links between basic filtering and optimisation method is performed through the derivation of an iterative filtering method, which provides an underlying insight for algorithm developing.

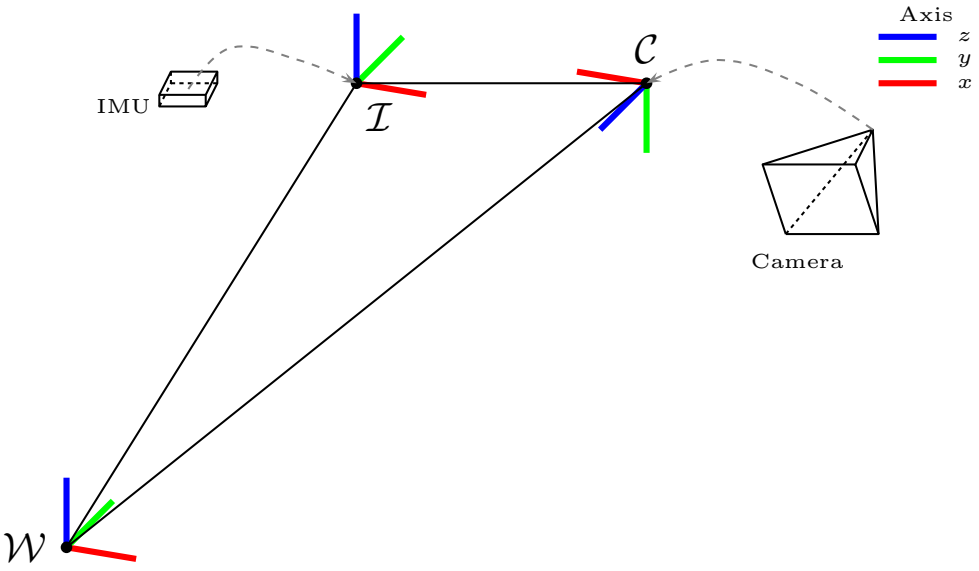


Figure 4.1: Coordinate frames for visual-inertial system. Each frame can be transformed from other frame by a rotation $\bar{\boldsymbol{q}}$ and a translation $\boldsymbol{p}$.

Different from the common feature-based error construction, we propose to use the photometric error of small patches as visual measurements. These patches are scattered at significant edges through the whole image. The results show that our method performs smoothly and outperforms feature-based methods in accuracy when travelling along a straight line with a slight tolerance of altering the field of view. The estimated trajectory for our collected data is capable to depict the motion of our controlled movement under extreme illumination condition. However, to ensure the direct-based method can be performed without failures, the overlapping proportion of consecutive images should be within the maximum allowance. Additionally, the building of small-scale scene structure also reveals the mapping potential of our proposed method.

4.2 Inertial-driven Propagation Model

In this section, the inertial-driven propagation model together with the system state will be introduced and derived.

4.2.1 IMU dynamic model

An IMU state vector of 3D rigid body at any time instant can be defined by a 16×1 vector,

$$\boldsymbol{x}_{\mathcal{I}} = \begin{bmatrix} {}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}^T & {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{I}}^T & {}^{\mathcal{W}}\boldsymbol{v}_{\mathcal{I}}^T & \boldsymbol{b}_g^T & \boldsymbol{b}_a^T \end{bmatrix}^T,$$

where ${}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}$ is the unit quaternion describing the rotation from world frame $\mathcal{W}$ to IMU frame $\mathcal{I}$, ${}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{I}}$ and ${}^{\mathcal{W}}\boldsymbol{v}_{\mathcal{I}}$ are the position and velocity with respect to $\mathcal{W}$, $\boldsymbol{b}_g$ and $\boldsymbol{b}_a$ are 3×1 vectors that describe the biases affecting the gyroscope and accelerometer measurements, respectively. The spatial relations between frames are shown in figure 4.1.

Assuming that the inertial measurements contain noises with zero-mean Gaussian models, denoted as $\boldsymbol{n}_g$ and $\boldsymbol{n}_a$, the real angular velocity $\boldsymbol{\omega}$ and the real acceleration $\boldsymbol{a}$ are related with gyroscope and accelerometer measurements can be written in the following form:

$$\begin{aligned} \boldsymbol{\omega}_m &= \boldsymbol{\omega} + \boldsymbol{b}_g + \boldsymbol{n}_g, \\ \boldsymbol{a}_m &= \boldsymbol{a} + \boldsymbol{b}_a + \boldsymbol{n}_a. \end{aligned} \tag{4.1}$$

The data driven dynamic model is a combination of 3D rigid body dynamics and the above IMU measurements, which can be represented by the following equation group:

$$\begin{aligned} {}^{\mathcal{I}}_{\mathcal{W}}\dot{\bar{\boldsymbol{q}}} &= \frac{1}{2}\Omega\left(\boldsymbol{\omega}\right) {}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}, & {}^{\mathcal{W}}\dot{\boldsymbol{v}}_{\mathcal{I}} &= \mathrm{C}^T_{{}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}}\boldsymbol{a} - \boldsymbol{g}, \\ {}^{\mathcal{W}}\dot{\boldsymbol{p}}_{\mathcal{I}} &= {}^{\mathcal{W}}\boldsymbol{v}_{\mathcal{I}}, & \dot{\boldsymbol{b}}_g &= \boldsymbol{n}_{wg}, & \dot{\boldsymbol{b}}_a &= \boldsymbol{n}_{wa}, \end{aligned} \tag{4.2}$$

where $\mathrm{C}_{{}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}}$ denotes a rotational matrix described by ${}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}$, $\boldsymbol{g}$ is the gravity vector in world frame $\mathcal{W}$, $\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T$ is the angular velocity expressed in IMU frame $\mathcal{I}$, and $\Omega\left(\boldsymbol{\omega}\right) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\times\rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix}$ is the quaternion kinematic matrix with $\lfloor\boldsymbol{\omega}\times\rfloor$ representing the skew-symmetric matrix. The IMU biases are modelled as random walk process, driven by the Gaussian noise, $\boldsymbol{n}_{wg}$ and $\boldsymbol{n}_{wa}$.

Let unit quaternion be $\bar{\boldsymbol{q}} := \left(q_0, \boldsymbol{q}^T\right)^T$ and its corresponding rotational matrix be $\mathrm{C}_{\bar{\boldsymbol{q}}}$. Two orientation representations can be linked via the equation below:

$$\mathrm{C}_{\bar{\boldsymbol{q}}} = \left(2q_0^2 - 1\right)\mathbf{I}_3 - 2q_0\lfloor\boldsymbol{q}\times\rfloor + 2\boldsymbol{q}\boldsymbol{q}^T.$$

Apart from the current IMU state $\boldsymbol{x}_{\mathcal{I}}$ mentioned above, the camera poses ${}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}} =$

$\begin{bmatrix} {}^{\mathcal{W}}_{\mathcal{C}}\bar{\boldsymbol{q}}^T & {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}}^T \end{bmatrix}^T$ are also included in the state vector. We assume there are N camera poses in current state vector. Thus a full state vector of our system can be written as

$$\boldsymbol{x} = \begin{bmatrix} {}^{\mathcal{I}}_{\mathcal{W}}\bar{\boldsymbol{q}}^T & {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{I}}^T & {}^{\mathcal{W}}\boldsymbol{v}_{\mathcal{I}}^T & \boldsymbol{b}_g^T & \boldsymbol{b}_a^T & {}^{\mathcal{W}}_{\mathcal{C}_1}\bar{\boldsymbol{q}}^T & {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_1}^T & \cdots & {}^{\mathcal{W}}_{\mathcal{C}_N}\bar{\boldsymbol{q}}^T & {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_N}^T \end{bmatrix}^T. \tag{4.3}$$

By applying the expectation operator in above equations, we obtain the prediction results using the IMU data driven dynamic model with $i \in [1, N]$:

$$\begin{aligned} {}^{\mathcal{I}}_{\mathcal{W}}\dot{\hat{\bar{\boldsymbol{q}}}} &= \frac{1}{2}\Omega\left(\boldsymbol{\omega}_m - \hat{\boldsymbol{b}}_g\right) {}^{\mathcal{I}}_{\mathcal{W}}\hat{\bar{\boldsymbol{q}}}, \\ {}^{\mathcal{W}}\dot{\hat{\boldsymbol{v}}}_{\mathcal{I}} &= \mathrm{C}^T_{{}^{\mathcal{I}}_{\mathcal{W}}\hat{\bar{\boldsymbol{q}}}}\left(\boldsymbol{a}_m - \hat{\boldsymbol{b}}_a\right) - \boldsymbol{g}, \\ {}^{\mathcal{W}}\dot{\hat{\boldsymbol{p}}}_{\mathcal{I}} &= {}^{\mathcal{W}}\hat{\boldsymbol{v}}_{\mathcal{I}}, \qquad {}^{\mathcal{W}}\dot{\hat{\boldsymbol{p}}}_{\mathcal{C}_i} = \boldsymbol{0}, \\ \dot{\hat{\boldsymbol{b}}}_g &= \boldsymbol{0}, \qquad \dot{\hat{\boldsymbol{b}}}_a = \boldsymbol{0}, \\ {}^{\mathcal{W}}_{\mathcal{C}_i}\dot{\hat{\bar{\boldsymbol{q}}}} &= \boldsymbol{0}. \end{aligned} \tag{4.4}$$

It should be noticed that in inertial only propagation, the visual term is static, i.e.

$${}^{\mathcal{W}}_{\mathcal{C}_i}\dot{\bar{\boldsymbol{q}}} = \boldsymbol{0}, \quad {}^{\mathcal{W}}\dot{\boldsymbol{p}}_{\mathcal{C}_i} = \boldsymbol{0}, \quad i \in [1, N]. \tag{4.5}$$

4.2.2 Error state representation

For the position, velocity, and bias state variables, the arithmetic difference can be applied. That is the error in the estimate $\hat{\boldsymbol{x}}$ of a quantity $\boldsymbol{x}$ is defined as $\tilde{\boldsymbol{x}} = \boldsymbol{x} - \hat{\boldsymbol{x}}$. But the error quaternion should be defined under the assumption as local minimal increment. If $\hat{\bar{\boldsymbol{q}}}$ is the estimated value of quaternion $\bar{\boldsymbol{q}}$, then the orientation error is described by the error quaternion $\delta\bar{\boldsymbol{q}}$, which is defined by the relation $\bar{\boldsymbol{q}} = \delta\bar{\boldsymbol{q}} \otimes \hat{\bar{\boldsymbol{q}}} \Rightarrow \delta\bar{\boldsymbol{q}} = \bar{\boldsymbol{q}} \otimes \hat{\bar{\boldsymbol{q}}}^{-1}$. In this expression, the symbol $\otimes$ denotes the quaternion multiplication as introduced in section 3.2.4.

Intuitively, the quaternion $\delta\bar{\boldsymbol{q}}$ describes a small rotation that causes the true and estimated attitude to coincide. Since attitude corresponds to 3 DoF, $\delta\theta$ with three elements can be a minimal representation to describe the attitude errors. With the minimal approximation, the error quaternion $\delta\bar{\boldsymbol{q}}$ can be written as

$$\delta\bar{\boldsymbol{q}} = \begin{bmatrix} \frac{1}{2}\delta\theta \\ \sqrt{1 - \frac{1}{4}\delta\theta^T\delta\theta} \end{bmatrix} \approx \begin{bmatrix} \frac{1}{2}\delta\theta \\ 1 \end{bmatrix}. \tag{4.6}$$

Thus, the error state vector with the length of $15 + 6N$ can be expressed as

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \delta\theta_{\mathcal{W}}^{\mathcal{I}\,T} & {}^{\mathcal{W}}\tilde{\boldsymbol{p}}_{\mathcal{I}}^{T} & {}^{\mathcal{W}}\tilde{\boldsymbol{v}}_{\mathcal{I}}^{T} & \tilde{\boldsymbol{b}}_{g}^{T} & \tilde{\boldsymbol{b}}_{a}^{T} & \delta\theta_{\mathcal{C}_1}^{\mathcal{W}T} & {}^{\mathcal{W}}\tilde{\boldsymbol{p}}_{\mathcal{C}_1}^{T} & \cdots & \delta\theta_{\mathcal{C}_N}^{\mathcal{W}\,T} & {}^{\mathcal{W}}\tilde{\boldsymbol{p}}_{\mathcal{C}_N}^{T} \end{bmatrix}^{T}, \quad (4.7)$$

or simply denoted with step index k as

$$\tilde{\boldsymbol{x}}_k = \begin{bmatrix} \tilde{\boldsymbol{x}}_{\mathcal{I}_k}^{T} & \tilde{\boldsymbol{\chi}}_{\mathcal{C}_1}^{T} & \cdots & \tilde{\boldsymbol{\chi}}_{\mathcal{C}_N}^{T} \end{bmatrix}^{T}. \quad (4.8)$$

The differential equations for the continuous time error state are

$$\begin{aligned} \delta\dot{\theta}_{\mathcal{W}}^{\mathcal{I}} &= -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor\delta\theta_{\mathcal{W}}^{\mathcal{I}} - \tilde{\boldsymbol{b}}_g - \boldsymbol{n}_g, \\ {}^{\mathcal{W}}\dot{\tilde{\boldsymbol{v}}}_{\mathcal{I}} &= -\mathrm{C}_{{}^{\mathcal{W}}_{\mathcal{I}}\hat{\boldsymbol{q}}}^{T}\left(\lfloor\hat{\boldsymbol{a}}\times\rfloor + \tilde{\boldsymbol{b}}_a + \boldsymbol{n}_a\right), \\ {}^{\mathcal{W}}\dot{\tilde{\boldsymbol{p}}}_{\mathcal{I}} &= {}^{\mathcal{W}}\tilde{\boldsymbol{v}}_{\mathcal{I}}, \\ \dot{\tilde{\boldsymbol{b}}}_a &= \boldsymbol{n}_{wa}, \qquad \dot{\tilde{\boldsymbol{b}}}_g = \boldsymbol{n}_{wg}, \\ {}^{\mathcal{W}}_{\mathcal{C}_i}\dot{\tilde{\boldsymbol{q}}} &= \boldsymbol{0}, \qquad {}^{\mathcal{W}}\dot{\tilde{\boldsymbol{p}}}_{\mathcal{C}_i} = \boldsymbol{0}, \end{aligned} \quad (4.9)$$

with $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}_m - \hat{\boldsymbol{b}}_g$, $\hat{\boldsymbol{a}} = \boldsymbol{a}_m - \hat{\boldsymbol{b}}_a$ and $i \in [1, N]$.

By stacking the differential equations for error state, the linearised continuous-time error state equation can be formed as

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{F}_c\tilde{\boldsymbol{x}} + \boldsymbol{G}_c\boldsymbol{n}, \quad (4.10)$$

with the noise vector $\boldsymbol{n} = \begin{bmatrix} \boldsymbol{n}_a^T, \boldsymbol{n}_{wa}^T, \boldsymbol{n}_g^T, \boldsymbol{n}_{wg}^T \end{bmatrix}^T$. And the covariance matrix of $\boldsymbol{n}$ depends on the noise characteristics of IMU, $\boldsymbol{Q} = \mathrm{diag}\left(\boldsymbol{\sigma}_{\boldsymbol{n}_a}^2, \boldsymbol{\sigma}_{\boldsymbol{n}_{wa}}^2, \boldsymbol{\sigma}_{\boldsymbol{n}_g}^2, \boldsymbol{\sigma}_{\boldsymbol{n}_{wg}}^2\right)$.

4.2.3 State propagation

The covariance matrix is defined as

$$\boldsymbol{P}_{k|k} = \begin{bmatrix} \boldsymbol{P}_{\mathcal{I}\mathcal{I}_{k|k}} & \boldsymbol{P}_{\mathcal{I}\mathcal{C}_{k|k}} \\ \boldsymbol{P}_{\mathcal{I}\mathcal{C}_{k|k}} & \boldsymbol{P}_{\mathcal{C}\mathcal{C}_{k|k}} \end{bmatrix} \quad (4.11)$$

where $\boldsymbol{P}_{\mathcal{I}\mathcal{I}_{k|k}}$ is the covariance matrix of the IMU state, $\boldsymbol{P}_{\mathcal{C}\mathcal{C}_{k|k}}$ is the $6N \times 6N$ covariance matrix of the camera pose estimates, and $\boldsymbol{P}_{\mathcal{I}\mathcal{C}_{k|k}}$ is the correlation between errors in IMU state and camera pose estimates. With this notation, the covariance matrix can be propagated by

$$\boldsymbol{P}_{k+1|k} = \boldsymbol{F}_d\boldsymbol{P}_{k|k}\boldsymbol{F}_d^T + \boldsymbol{Q}_d, \quad (4.12)$$

where the state-transition matrix which can be calculated by assuming $\boldsymbol{F}_c$ and $\boldsymbol{G}_c$ to be constant over the integration time interval between two consecutive steps,

$$\boldsymbol{F}_d = \exp(\boldsymbol{F}_c \Delta t) = \boldsymbol{I} + \boldsymbol{F}_c \Delta t + \frac{1}{2}\boldsymbol{F}_c^2 \Delta t^2 + \cdots \tag{4.13}$$

and the discrete-time covariance matrix $\boldsymbol{Q}_d$ can also be derived through numerical integration:

$$\boldsymbol{Q}_d = \int_{\Delta t} \boldsymbol{F}_d(\tau)\, \boldsymbol{G}_c \boldsymbol{Q}_c \boldsymbol{G}_c^T \boldsymbol{F}_d(\tau)^T \, d\tau. \tag{4.14}$$

In practice, the inertial measurements for state propagation are obtained from IMU in discrete form, thus we assume the signals from gyroscopes and accelerometers are sampled with time interval Δt, and the state estimate is propagated using numerical integration like Runge-Kutta methods. Thus, the mean and covariance propagation process using the inertial feeding can be summarised as follows,

(1) When IMU data, $\boldsymbol{\omega}_m$ and $\boldsymbol{a}_m$, in a certain sample frequency, is available to the filter, the state vector is propagated by using numerical integration on equation (4.4).

(2) Calculate $\boldsymbol{F}_d$ and $\boldsymbol{Q}_d$ according to (4.13) and (4.14) respectively.

(3) The propagated state covariance matrix is computed from (4.12).

4.3 Direct Visual Measurement Model

Due to the biases and noises in IMU data, the prediction results from propagation step become worse and worse over time. However, the measurements from visual sensors are able to provide critical information to bound the increased errors. To do so, in a filtering framework, key information extracted from images should be cast into the measurement equations.

4.3.1 Keyframe selection

In the image processing, we adopt the concept of keyframe to maintain the scene information for a period of movement. There is the only keyframe pose is kept in a normal state vector when the algorithm is running. The key camera pose is located at the first one in state vector of camera poses, defined as ${}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_1}$ in 4.3. A keyframe maintains a probability depth map, which provides depth reference for consecutive

images in direct visual tracking. This depth map is continuously propagated from frame to frame by new feeding until a new keyframe is decided.

When a new image arrives, it will be decided whether it is suitable to be a keyframe. If the camera has moved too far away from the existing map, a replacement of current tracking reference becomes more important than other estimation tasks. Here, we use a criterion that measures the distance between the oldest camera pose state (keyframe pose) ${}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_1}$ and the latest estimate ${}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_N}$. It can be expressed as

$$\text{distance}\left({}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_1}, {}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_N}\right) := \left[\mathrm{C}_{\mathcal{C}_1\bar{\boldsymbol{q}}} \cdot \mathrm{C}_{\mathcal{C}_1\bar{\boldsymbol{q}}} \quad {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_1}^T - {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_N}^T\right]^T. \tag{4.15}$$

If such distance spans a certain threshold, the new frame will be adopted as a keyframe. Additionally, taking into account the impact of significant rotation in small translation, the matching rate of patch correspondences with a threshold also should be set as a selection factor.

Once a new frame is chosen to be a keyframe after the camera pose has been jointly estimated by past frames, the new depth map will be formed by projecting all co-visible patches from previous frames, and then the latest frame will be treated as a new reference. All the old frames with their pose before this new keyframe will be marginalised out from the state vector (see section 4.4.2). In this situation, the pose of new keyframe is the only camera pose that is available in the current state vector.

If the new tracked image is not sufficient to be a keyframe, the new estimated pose will be augmented and maintained in state vector (see section 4.4.1) and their depth estimates will also temporally stored. The non-keyframes will be used to refine the map of current keyframe through a process of patch correspondence searching and triangulation geometry building, which has been introduced as the process in previous section 3.1.4.1.

The full process of keyframe selection can be illustrated in flowchart 4.2.

4.3.2 Corresponding patches

In our research, we consider extracting dense small patches in the image region with a large gradient. An intuitive example can be found in figure 4.13(b), where the edges of walls and objects are often with large image gradient. The dense patches can cover representative and significant regions in one image while removing large redundancy areas. When compared to feature-based methods, this patches in the area with large gradient are more flexibly deployed, which does not need to take

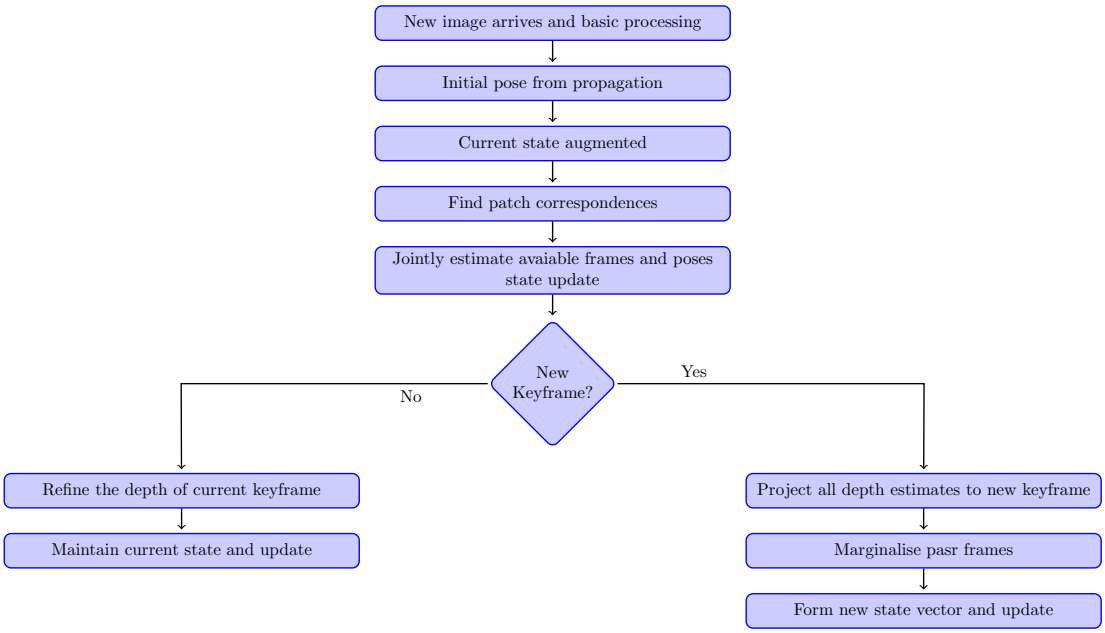


Figure 4.2: Flowchart for keyframe selection.

account the shape of constraints of particular feature types, especially for irregular object edges.

In the searching process, we firstly find an estimated corresponding patch in new image. For simplicity in expression, here we use the centre pixel expression $\mathbf{u}$ to represent a $3n \times 3n$ patch $[\mathbf{u}]_{3n\times 3n}$, $(n \in \mathbb{Z}_+)$, showing the spatial relationship between a current image $\mathbf{I}_i$ as frame $\mathcal{C}_i$ and reference keyframe $\mathbf{I}_k$ as frame $\mathcal{C}_\mathtt{k}$. As is illustrated in figure 4.3, an image patch $[\mathbf{u}]_{3n\times 3n}$ in image domain $\Phi \subset \mathbb{R}^2$ contains the photometric information of intensity, $\mathbf{I}(\cdot) : \Phi \to \mathbb{R}^{3n\times 3n}$. The function is nonlinear and unrelated to $\mathbf{u}$, but we can extract the intensity value directly at location $\mathbf{u}$ in the image matrix. Thus a corresponding patch $\mathbf{u}'$ in another frame can be found through a projective model, which is similar to the point projection in section 3.1.2.

The patch $\mathbf{u}$ in keyframe $\mathbf{I}_k$ can be projected into 3D space under camera frame $\mathcal{C}_\mathtt{k}$ using back projective equation (3.11) as

$$\mathbf{x}^{\mathbf{p}}_{\mathcal{C}_\mathtt{k}} = \mathtt{D}_\mathtt{k}(\mathbf{u})\,\mathbf{K}^{-1}\mathbf{u} \tag{4.16}$$

where we define the depth function $\mathtt{D}(\mathbf{u})$ getting scale reference at image point $\mathbf{u}$ from the latest estimated depth map. $\mathbf{K}$ representing the intrinsic parameters is known from camera calibration beforehand, which is defined the same as in equation (3.6).

Then, through spatial transformation, this point can be moved to world frame $\mathcal{W}$ by

$$\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} = \mathrm{C}_{\mathcal{C}_k \bar{\boldsymbol{q}}} \cdot \mathbf{x}_{\mathcal{C}_k}^{\mathbf{p}} + {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_k}. \tag{4.17}$$

Finally, the estimated corresponding $\hat{\mathbf{u}}'$ in image $\mathbf{I}_i$ is computed by a forward projection (refer to equation (3.10)) as

$$\hat{\mathbf{u}}' = \lambda \mathbf{K} \mathrm{C}_{\mathcal{W} \bar{\boldsymbol{q}}}^{\mathcal{C}_i} \left(\mathbf{x}_{\mathcal{W}}^{\mathbf{p}} - {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_i} \right), \tag{4.18}$$

where λ is defined as the reciprocal of $z_{\mathcal{C}_k}^{\mathbf{p}}$, making the spatial point fallen into image plane $\mathbf{I}_i$. By combining equation (4.16) to (4.18), we can define a wrap function $w(\cdot)$ that maps a patch $\mathbf{u}$ in keyframe to an estimated patch as $\hat{\mathbf{u}}'$ in current frame as

$$\begin{aligned} \hat{\mathbf{u}}' &= w \left({}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_i}, {}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_k}, \mathtt{D}_k(\mathbf{u}), \mathbf{u} \right) \\ &= \lambda \mathbf{K} \mathrm{C}_{\mathcal{W} \bar{\boldsymbol{q}}}^{\mathcal{C}_i} \left(\mathrm{C}_{\mathcal{C}_k \bar{\boldsymbol{q}}} \mathtt{D}_k \left(\mathbf{u} \right) \mathbf{K}^{-1} \mathbf{u} + {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_k} - {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_i} \right). \end{aligned} \tag{4.19}$$

On the other aspect, another corresponding patch $\mathbf{u}'$ as direct measurement should be found to build the error function. Under the assumption that the same image region makes a little movement between current and keyframe frame, this can be achieved easily by controlling threshold in distance criterion (4.15) when selecting a keyframe. By applying the epipolar constraints introduced in section 3.1.4.2, the candidate patch of the highest similarity in intensity and gradient can be found along the epipolar line. From above projective process, we can already get a radius as search range on the epipolar line, thus avoiding simple infinite exhaustive search and saving the computational time. To the worst situation, if a corresponding patch is not found, the range can be extended further to explore other possible candidates nearby. This little technique increases the ability of relaxation for alignment. Once all available pairs of corresponding patches between two frames have been found, the filtering update can be ignited.

4.3.3 State update

Different from the case only using features as the visual measurements, where the measurement model leverages the re-projection error to build the measurement model. In our method, we define the visual measurement as the photometric information at certain pixel patch including the $3n \times 3n$ intensity vector and its gradient. They can be noted as $\hat{\boldsymbol{z}} = \mathbf{I}\left(\hat{\mathbf{u}}'\right)$ and $\boldsymbol{z} = \mathbf{I}\left(\mathbf{u}'\right)$, which are extracted from the

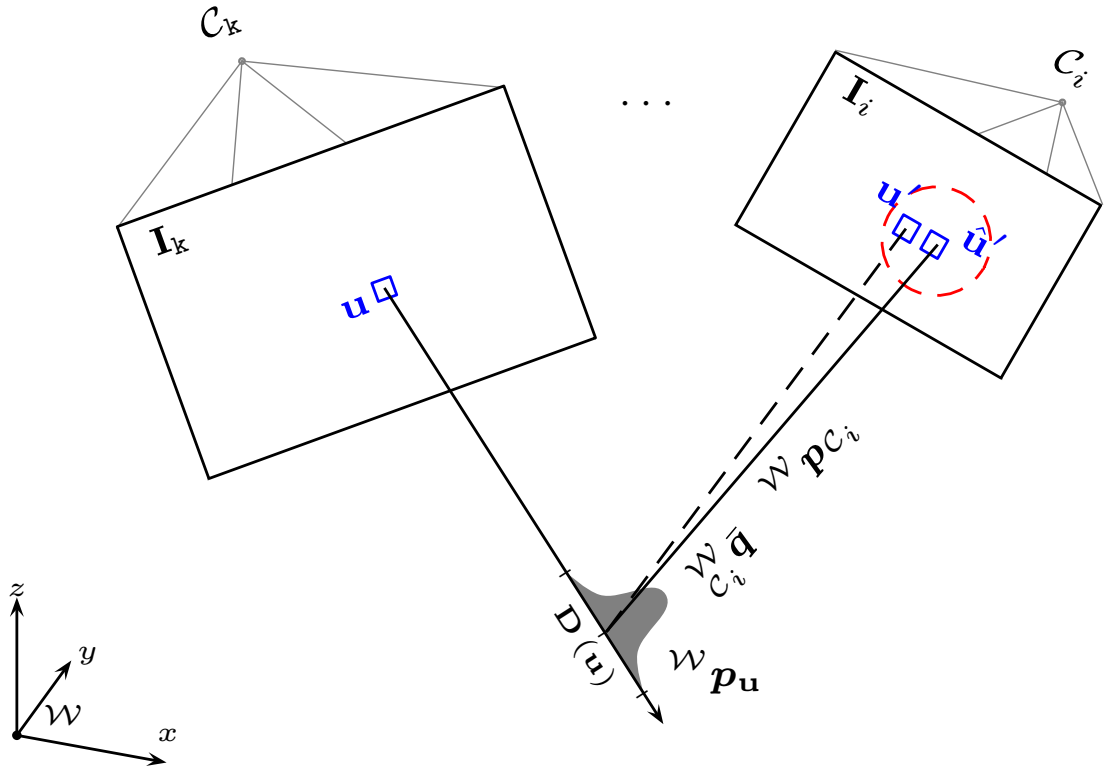


Figure 4.3: A patch is projected from reference keyframe to current frame.

location of projective prediction and direct search respectively.

Based on these measurements, we can build the residual from photometric error as

$$\begin{aligned}\tilde{\boldsymbol{z}} = \boldsymbol{z} - \hat{\boldsymbol{z}} &= \sum_{\mathbf{u}\in\Phi'} \mathrm{I}\left(\mathbf{u}'\right) - \mathrm{I}\left(\hat{\mathbf{u}}'\right) \\ &= \sum_{\mathbf{u}\in\Phi'} \mathrm{I}\left(\mathbf{u}'\right) - \mathrm{I}\left(w\left({}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_i}, {}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_k}, \mathtt{D}_k(\mathbf{u}), \mathbf{u}\right)\right),\end{aligned} \tag{4.20}$$

where Φ' is the sub regions with large gradient of the whole image region Φ. This is a nonlinear measurement function, which should be linearised to find Jacobian matrix in the filtering update process (This can refer to previous section 3.3.8 of filtering basis). The error measurement function can be approximated as

$$\tilde{\boldsymbol{z}} = \boldsymbol{H}\tilde{\boldsymbol{x}} + \boldsymbol{n}_p \tag{4.21}$$

where the processing noise $\boldsymbol{n}_p$ includes a zero-mean Gaussian white noise with covariance σ_p and all the other constant term. Then the covariance matrix can be expressed as $\boldsymbol{R} = \sigma_p^2 \mathbf{I}_N$ with current N camera poses in state vector. The notation

$\boldsymbol{H}$ defines the measurement matrix, which includes the Jacobian with respect to the current state, which can be given the form by

$$\boldsymbol{H} = \begin{bmatrix} \mathbf{0} & \cdots & \underbrace{\boldsymbol{J}_i^{\boldsymbol{q}} \quad \boldsymbol{J}_i^{\boldsymbol{p}}}_{\text{Jaccobian w.r.t. } {}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_i}} & \cdots \end{bmatrix}, \quad i \in [1, N] \tag{4.22}$$

Follow the chain rule of a differentiation process to get every component along the chain as

$$\begin{aligned} \boldsymbol{J}_i^{\boldsymbol{q}} &= \frac{\partial \mathbf{I}}{\partial \mathbf{u}'} \frac{\partial w}{\partial {}^{\mathcal{W}}_{\mathcal{C}_i}\bar{\boldsymbol{q}}}, \\ \boldsymbol{J}_i^{\boldsymbol{p}} &= \frac{\partial \mathbf{I}}{\partial \mathbf{u}'} \frac{\partial w}{\partial {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_i}}, \end{aligned} \tag{4.23}$$

where the partial differential term $\frac{\partial \mathbf{I}}{\partial \mathbf{u}'}$ represents the gradient of a patch, which can be acquired directly through basic image processing. Other terms can be applied by differentiating equation (4.19) and written as tidy forms as

$$\begin{aligned} \frac{\partial w}{\partial {}^{\mathcal{W}}_{\mathcal{C}_i}\bar{\boldsymbol{q}}} &= -\lambda \mathbf{K} \lfloor \left(\mathrm{C}_{{}^{\mathcal{W}}_{\mathcal{C}_k}\bar{\boldsymbol{q}}} \mathrm{D}_{\mathbf{k}} \left(\mathbf{u} \right) \mathbf{K}^{-1} \mathbf{u} + {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_k} - {}^{\mathcal{W}}\hat{\boldsymbol{p}}_{\mathcal{C}_i} \right) \times \rfloor, \\ \frac{\partial w}{\partial {}^{\mathcal{W}}\boldsymbol{p}_{\mathcal{C}_i}} &= -\lambda \mathbf{K} \mathrm{C}_{{}^{\mathcal{C}_i}_{\mathcal{W}}\hat{\boldsymbol{q}}}. \end{aligned} \tag{4.24}$$

By stacking all above calculations of the corresponding patches between all past frames whose poses are still available in the current state vector, we can obtain the measurement matrix $\boldsymbol{H}$, and it is ready for updating the prediction results from the inertial propagation step. Then the Kalman gain is calculated as

$$\boldsymbol{K} = \boldsymbol{P}_{k+1|k} \boldsymbol{H}^T \left(\boldsymbol{H} \boldsymbol{P}_{k+1|k} \boldsymbol{H}^T + \boldsymbol{R} \right)^{-1}. \tag{4.25}$$

The final correction is $\tilde{\boldsymbol{x}}_{k+1} = \boldsymbol{K} \cdot \tilde{\boldsymbol{z}}$. After the correction, we can get the updated state estimate $\boldsymbol{x}_{k+1}$. Lastly, the error state covariance is updated as

$$\boldsymbol{P}_{k+1|k+1} = \left(\boldsymbol{I} - \boldsymbol{K} \boldsymbol{H} \right) \boldsymbol{P}_{k+1|k}. \tag{4.26}$$

The full update process can be summarised as follows:

(1) Follow the process of keyframe selection in section 4.3.1.

(2) When getting into the step of state update, calculate the measurement matrix

$\boldsymbol{H}$ from (4.23) to (4.24).

(3) Compute the Kalman gain as equation (4.25).

(4) Update the normal state by adding the correction and update the error state covariance as equation (4.26).

4.4 State Augmentation and Removal

Over all the process of inertial propagation and visual update, we notice that if the estimated camera poses join in or move out from the current state, the length of the state vector will be altered. In this section, we will present the technical details of augmentation and removal of the state vector and covariance matrix. This is also the core for multi-state constraint filtering method.

4.4.1 State augmentation

The pose of new recording image can be obtained by transforming the latest pose estimation from inertial driven propagation. Thus we have

$$\begin{aligned} {}^{\mathcal{C}}_{\mathcal{W}}\hat{\bar{\boldsymbol{q}}} &= {}^{\mathcal{C}}_{\mathcal{I}}\bar{\boldsymbol{q}} \otimes {}^{\mathcal{I}}_{\mathcal{W}}\hat{\bar{\boldsymbol{q}}} \\ {}^{\mathcal{W}}\hat{\boldsymbol{p}}_{\mathcal{C}} &= {}^{\mathcal{W}}\hat{\boldsymbol{p}}_{\mathcal{I}} + \mathrm{C}^{T}_{{}^{\mathcal{I}}_{\mathcal{W}}\hat{\bar{\boldsymbol{q}}}}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{C}}, \end{aligned} \tag{4.27}$$

where ${}^{\mathcal{C}}_{\mathcal{I}}\bar{\boldsymbol{q}}$ is the quaternion indicating the rotation between the inertial frame $\mathcal{I}$ and the camera frame $\mathcal{C}$, and ${}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{C}}$ is the relative position of the origin of camera in inertial frame, both of which are known beforehand by setting up hardware parameters.

The estimated camera pose is appended at the end of state vector of (4.7) and the covariance matrix is augmented accordingly as

$$\begin{bmatrix} \boldsymbol{I}_{15+6N} \\ \mathbf{A} \end{bmatrix} \boldsymbol{P}_{15+6N} \begin{bmatrix} \boldsymbol{I}_{15+6N} \\ \mathbf{A} \end{bmatrix}^{T} \rightarrow \boldsymbol{P}_{15+6(N+1)}, \tag{4.28}$$

Where the augmented matrix $\boldsymbol{A}$ is the Jacobian related to current camera pose and the inertial pose in state vector, which can be derived from equation (4.27):

$$\boldsymbol{A} = \begin{bmatrix} \mathrm{C}_{{}^{\mathcal{C}}_{\mathcal{I}}\bar{\boldsymbol{q}}} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} & \mathbf{0}_{3\times6N} \\ \underbrace{\lfloor \mathrm{C}^{T}_{{}^{\mathcal{I}}_{\mathcal{W}}\hat{\bar{\boldsymbol{q}}}}\,{}^{\mathcal{I}}\boldsymbol{p}_{\mathcal{C}}\times \rfloor}_{\text{inertial rotation block}} & \underbrace{\boldsymbol{I}_{3\times3}}_{\text{inertial position block}} & \underbrace{\mathbf{0}_{3\times3}}_{\text{velocity block}} & \underbrace{\mathbf{0}_{3\times6}}_{\text{bias block}} & \underbrace{\mathbf{0}_{3\times6N}}_{\text{visual pose block}} \end{bmatrix}. \tag{4.29}$$

4.4.2 State removal

If all the state variables and patch correspondences during operation are maintained, the computational complexity becomes larger and larger with an increase of distance. In our method, we keep the poses of one keyframe and several consecutive frames in a live state vector. Once a new keyframe is found, the past camera poses will be marginalised out.

At the moment when a new keyframe has been chosen and its pose has been updated, the current state vector includes three parts of variables, i.e. inertial state variables, old visual state variables, and pose of the newly chosen keyframe. Here for an explicit explanation, we note them as states $\boldsymbol{x}_{\mathcal{I}}, \boldsymbol{x}_{\mathcal{C}}, \boldsymbol{x}_{\mathcal{K}}$ respectively. According to the measurement model of (4.21), we can regard the equation as a least squares problem. To marginalise out the old states from the state of new keyframe, we can rewrite the visual block of the whole measurement model as

$$\begin{bmatrix} \boldsymbol{H}_{\mathcal{C}\mathcal{C}} & \boldsymbol{H}_{\mathcal{C}\mathcal{K}} \\ \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T & \boldsymbol{H}_{\mathcal{K}\mathcal{K}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{\mathcal{C}} \\ \boldsymbol{x}_{\mathcal{K}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_{\mathcal{C}} \\ \boldsymbol{b}_{\mathcal{K}} \end{bmatrix}. \tag{4.30}$$

By applying the the Schur complement, the above equation (4.30) can be changed into

$$\begin{bmatrix} \boldsymbol{H}_{\mathcal{C}\mathcal{C}} & \boldsymbol{H}_{\mathcal{C}\mathcal{K}} \\ \boldsymbol{0} & \boldsymbol{H}_{\mathcal{K}\mathcal{K}} - \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T \boldsymbol{H}_{\mathcal{C}\mathcal{C}}^{-1} \boldsymbol{H}_{\mathcal{C}\mathcal{K}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{\mathcal{C}} \\ \boldsymbol{x}_{\mathcal{K}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_{\mathcal{C}} \\ \boldsymbol{b}_{\mathcal{K}} - \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T \boldsymbol{H}_{\mathcal{C}\mathcal{C}}^{-1} \boldsymbol{b}_{\mathcal{C}} \end{bmatrix}. \tag{4.31}$$

Thus, the marginalised visual block becomes

$$\left(\boldsymbol{H}_{\mathcal{K}\mathcal{K}} - \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T \boldsymbol{H}_{\mathcal{C}\mathcal{C}}^{-1} \boldsymbol{H}_{\mathcal{C}\mathcal{K}}\right) \boldsymbol{x}_{\mathcal{K}} = \boldsymbol{b}_{\mathcal{K}} - \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T \boldsymbol{H}_{\mathcal{C}\mathcal{C}}^{-1} \boldsymbol{b}_{\mathcal{C}}. \tag{4.32}$$

Marginalising out the state $\boldsymbol{x}_{\mathcal{C}}$ will induce dependencies between other states that are dependent on $\boldsymbol{x}_{\mathcal{C}}$ like $\boldsymbol{H}_{\mathcal{C}\mathcal{C}}$ and $\boldsymbol{H}_{\mathcal{C}\mathcal{K}}$. However, in our system, the inertial block is defined as independent to visual block. Therefore, we can easily stack the matrices and form the new $\boldsymbol{H}$ for the updated visual model:

$$\begin{bmatrix} \boldsymbol{H}_{\mathcal{I}\mathcal{I}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}_{\mathcal{K}\mathcal{K}} - \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T \boldsymbol{H}_{\mathcal{C}\mathcal{C}}^{-1} \boldsymbol{H}_{\mathcal{C}\mathcal{K}} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_{\mathcal{I}} \\ \boldsymbol{x}_{\mathcal{K}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_{\mathcal{I}} \\ \boldsymbol{b}_{\mathcal{K}} - \boldsymbol{H}_{\mathcal{C}\mathcal{K}}^T \boldsymbol{H}_{\mathcal{C}\mathcal{C}}^{-1} \boldsymbol{b}_{\mathcal{C}} \end{bmatrix}. \tag{4.33}$$

Additionally, since the covariance matrix is in diagonal form, the new matrix can be achieved by simply permuting and pruning operation.

4.4.3 Outlier rejection

During the process of finding patch correspondences in section 4.3.2, it is inevitable to import gross outliers into visual measurements. Such mismatching pixel patches can dramatically lead to divergence when calculating a solution. Therefore, we need to remove as many outliers as possible.

There are various outlier rejection methods available in the literature. Among them, RANdom SAmple Consensus (RANSAC) is the most basic and appealing one in practical use. The core idea of RANSAC method is estimating the parameters of a function with valid data and outliers but ensuring that the outliers do not affect the result to a great extent. In the method, two prerequisites are needed: a model of the potential function and criterion to classify outliers.

In the case of determining the depth and pose parameters in our visual model, the projection process as equation (4.19) is adopted. Here in outlier rejection, we use the residual error the same as in measurement model (4.20), where the photometric error between the detected patches and the projected corresponding estimates allow us to build a formula for distinguishing the outliers,

$$\begin{aligned} ||\tilde{\boldsymbol{z}}|| &< \delta \subset \text{inlier} \\ ||\tilde{\boldsymbol{z}}|| &> \delta \subset \text{outlier}. \end{aligned} \tag{4.34}$$

With known camera intrinsic, estimates of camera poses and probability of depth value, the outlier rejection process iteratively chooses a random set of eight correspondences and checks the consensus with the criterion as (4.34). Only the inliers will be adopted in measurement model calculation and further used to update the depth as mentioned in section 4.3.1.

4.5 Analysis

4.5.1 Links between various estimation methods

As stated in the background review chapter, there are two basic approaches widely used in pose estimation problems, i.e. filtering based and optimisation based. Our visual-inertial odometry method is a filtering based one. However, we notice that both of filtering based and optimisation based approaches can be formed under the Bayesian inference. When the succession of approximation linearisation is available, their link can be made explicitly via the iterated EKF. In our publication of [39],

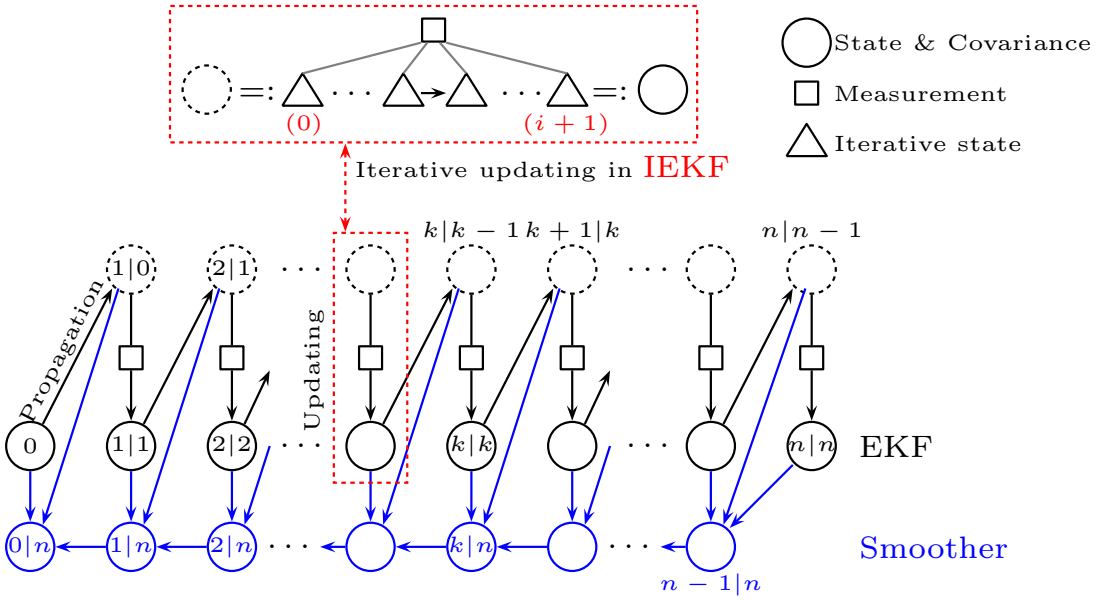


Figure 4.4: Links of filtering and optimisation methods.

we illustrate the process and relationship of EKF, IEKF and smoother as figure 4.4. When the approximation linearisation is just a single step, the smoother based approaches which include a forward and a backward pass are equivalent to the optimisation based approaches which are solved via the Cholesky decomposition of information matrix of least square problems. Here, to break the gap and avoid separately understanding between both methods, we give an insightful analysis of the links between filtering and optimisation based methods.

The core of filtering based approaches is the Kalman filter and the core of optimisation based approaches is the Gauss-Newton method. The link between them is the Iterated EKF (IEKF) [173]. An EKF has two steps: prediction and update. Let the result of prediction step is $\hat{\boldsymbol{x}}_k \sim \mathcal{N}(\bar{\boldsymbol{x}}_k, \boldsymbol{P}_k)$ at current time k. The difference between EKF and IEKF is that there is an iterative loop in the update step of IEKF while only a single loop is executed in the update step of EKF. It is the iterative loop of IEKF which can drive the error caused by model linearization as close as possible to the counterpart in optimisation based approaches.

In the following, we will show the equivalence between the iterative loop in update step of IEKF and the Gauss-Newton method of optimisation approaches from the point view of likelihood maximisation.

At time k, the IEKF has $\boldsymbol{x}_k, \hat{\boldsymbol{x}}_{k|k-1}, \boldsymbol{z}_k$ as the current state, the current state

estimate, and the measurement, respectively. The measurement model is the same as equation (3.68) and $\hat{\boldsymbol{x}}_{k|k-1} \sim \mathcal{N}(\bar{\boldsymbol{x}}_k, \boldsymbol{P}_{k|k-1})$.

Define an error vector as in quadratic cost function with a free variable $\boldsymbol{\mu}$.

$$\boldsymbol{E}(\boldsymbol{\mu}) = \boldsymbol{S} \begin{bmatrix} \boldsymbol{z}_k - h(\boldsymbol{\mu}) \\ \hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{\mu} \end{bmatrix},$$

where $\boldsymbol{S}^T\boldsymbol{S} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{P}_{k|k-1} \end{bmatrix}^{-1}$.

The maximum likelihood optimisation problem is

$$\boldsymbol{\mu} = \arg\max_{\boldsymbol{\mu}} \exp\left(-\frac{1}{2}\boldsymbol{E}(\boldsymbol{\mu})^T\boldsymbol{E}(\boldsymbol{\mu})\right)$$

or

$$\boldsymbol{\mu} = \arg\min_{\boldsymbol{\mu}} \left(\frac{1}{2}\boldsymbol{E}(\boldsymbol{\mu})^T\boldsymbol{E}(\boldsymbol{\mu})\right).$$

Given the initial value $\boldsymbol{\mu}^{(0)} = \hat{\boldsymbol{x}}_{k|k-1}$, the Gauss-Newton method gives

$$\boldsymbol{\mu}^{(i+1)} = \boldsymbol{\mu}^{(i)} - \left(\left(\nabla\boldsymbol{E}(\boldsymbol{\mu}^{(i)})\right)^T \nabla\boldsymbol{E}(\boldsymbol{\mu}^{(i)})\right)^{-1} \left(\nabla\boldsymbol{E}(\boldsymbol{\mu}^{(i)})\right)^T \boldsymbol{E}(\boldsymbol{\mu}^{(i)})$$

with

$$\nabla\boldsymbol{E}(\boldsymbol{\mu}^{(i)}) = -\boldsymbol{S}\begin{bmatrix} \boldsymbol{H}^{(i)} \\ \mathbf{I} \end{bmatrix}$$

and $\boldsymbol{H}^{(i)} = \nabla h(\boldsymbol{\mu}^{(i)})$. Using above gradient, the Gauss-Newton method becomes

$$\begin{aligned} \boldsymbol{\mu}^{(i+1)} &= \left(\boldsymbol{H}_{(i)}^T\boldsymbol{R}^{-1}\boldsymbol{H}^{(i)} + \boldsymbol{P}_{k|k-1}^{(i)}\right)^{-1} \left(\boldsymbol{H}_{(i)}^T\boldsymbol{R}^{-1}\left(\boldsymbol{z}_k - h(\boldsymbol{\mu}^{(i)}) + \boldsymbol{H}^{(i)}\boldsymbol{\mu}^{(i)}\right) + \boldsymbol{P}_{k|k-1}^{(i)}\hat{\boldsymbol{x}}_{k|k-1}\right) \\ &= \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k^{(i)}\left(\boldsymbol{z}_k - h(\boldsymbol{\mu}^{(i)}) - \boldsymbol{H}^{(i)}\left(\hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{\mu}^{(i)}\right)\right) \end{aligned} \tag{4.35}$$

with the gain

$$\boldsymbol{K}_k^{(i)} = \boldsymbol{P}_{k|k-1}^{(i)}\boldsymbol{H}_{(i)}^T\left(\boldsymbol{H}^{(i)}\boldsymbol{P}_{k|k-1}^{(i)}\boldsymbol{H}_{(i)}^T + \boldsymbol{R}\right)^{-1}. \tag{4.36}$$

And the covariance is

$$\boldsymbol{P}_{k|k-1}^{(i+1)} = \mathtt{E}\left[(\boldsymbol{\mu}^{(i+1)} - \boldsymbol{\mu}^{(i)})^T(\boldsymbol{\mu}^{(i+1)} - \boldsymbol{\mu}^{(i)})\right] = \left(\mathbf{I} - \boldsymbol{K}_k^{(i)}\boldsymbol{H}^{(i)}\right)\boldsymbol{P}_{k|k-1}^{(i)}. \tag{4.37}$$

After the loop in i, it can be seen that the results from the update step are $\hat{\boldsymbol{x}}_{k|k} = \boldsymbol{\mu}^{(i+1)}$ and $\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1}^{(i+1)}$.

In summary, the above iterative loop of Gauss-Newton method is viewed as the update step of IEKF, that is

(1) Initialization: $i = 0$, $\boldsymbol{\mu}^{(0)} = \hat{\boldsymbol{x}}_{k|k-1}$ and $\boldsymbol{P}^{(0)}_{k|k-1} = \boldsymbol{P}_{k|k-1}$,

(2) Loop calculation from equation (4.35) to (4.37),

(3) Final updating: $\hat{\boldsymbol{x}}_{k|k} = \boldsymbol{\mu}^{(i+1)}$ and $\boldsymbol{P}_{k|k} = \boldsymbol{P}^{(i+1)}_{k|k-1}$.

When only one iterative loop is executed, the above is the update step of EKF. Their relationship can be viewed clearly in figure 4.4.

4.5.2 State observability and parameter identifiability

Observability is a fundamental property which reflects the possibility of estimating states by input/output data. If states are distinguishable, they can be determined from the outputs and the known measurements. If states are indistinguishable, they are called unobservable and their corresponding variance will grow without bound.

Observability of a linear system is a global property that can be determined either from the rank of the observability matrix or from the rank of Gramian matrix. However, observability of a nonlinear system is determined locally to a given state [174]. Given only the measurements from IMU and camera, the question whether these states and parameters can be recovered is determined by the analysis of observability and identifiability.

The VIO problem is to recover the motion trajectory of a mobile platform in the global frame with only the measurements from inertial and visual sensors. However, this issue not can be solved in such a straightforward way. Apart from the complexity of filtering or optimisation based approaches, some parameters play a crucial role in the succession of state estimation. These parameters include

- Camera intrinsic parameters: focal length, principal points, lens distortion;
- IMU parameters: acceleration and gyroscope biases;
- Spatial parameters: the transform between IMU and camera;
- Temporal parameter: the time delay between IMU and camera measurements.

They are treated as time-invariant variables except for the IMU biases, called the system parameters. On contrary, the motion trajectory of a mobile platform is represented by time-variant variables as pose states.

Pose estimation is the core task for VIO or other SLAM problems. Based on the observability rank condition, some publications in robotics shown that the platform pose in the global frame is *unobservable* and the rotation around gravity vector (yaw) is *unobservable*. However, the analysis of observability rank condition shows that 6 DOF camera-IMU transformation, along with IMU biases, gravity vector, and the metric scene structure are all *observable*. The intuitive interpretation is that the visual camera is a bearing only sensor and the IMU is only a double integrator, which is not able to provide the pose and yaw information in the global frame.

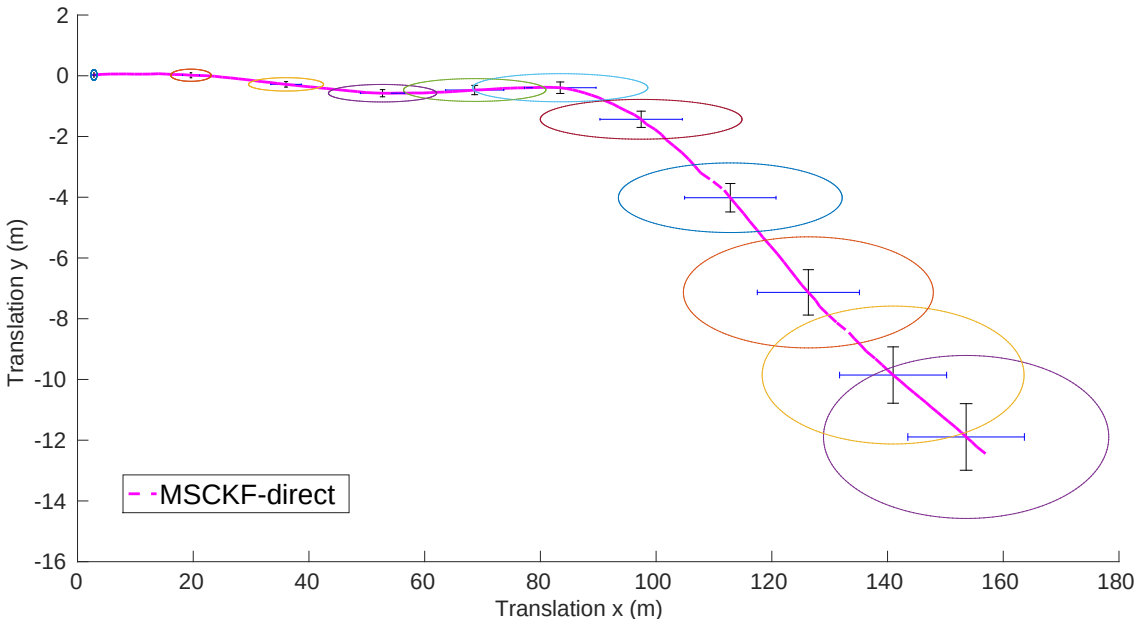
As for the other parameters, in most of the visual related localisation techniques, the intrinsic camera parameters are known in advance. But they can also be calibrated online, such as the methods introduced in work of [175]. The IMU biases vary with time and are modelled as time variant states in most cases. The spatial parameters between IMU and camera are 6 DOF transformation, which must be known precisely to stabilise the estimation results.

4.6 Experiments and Results

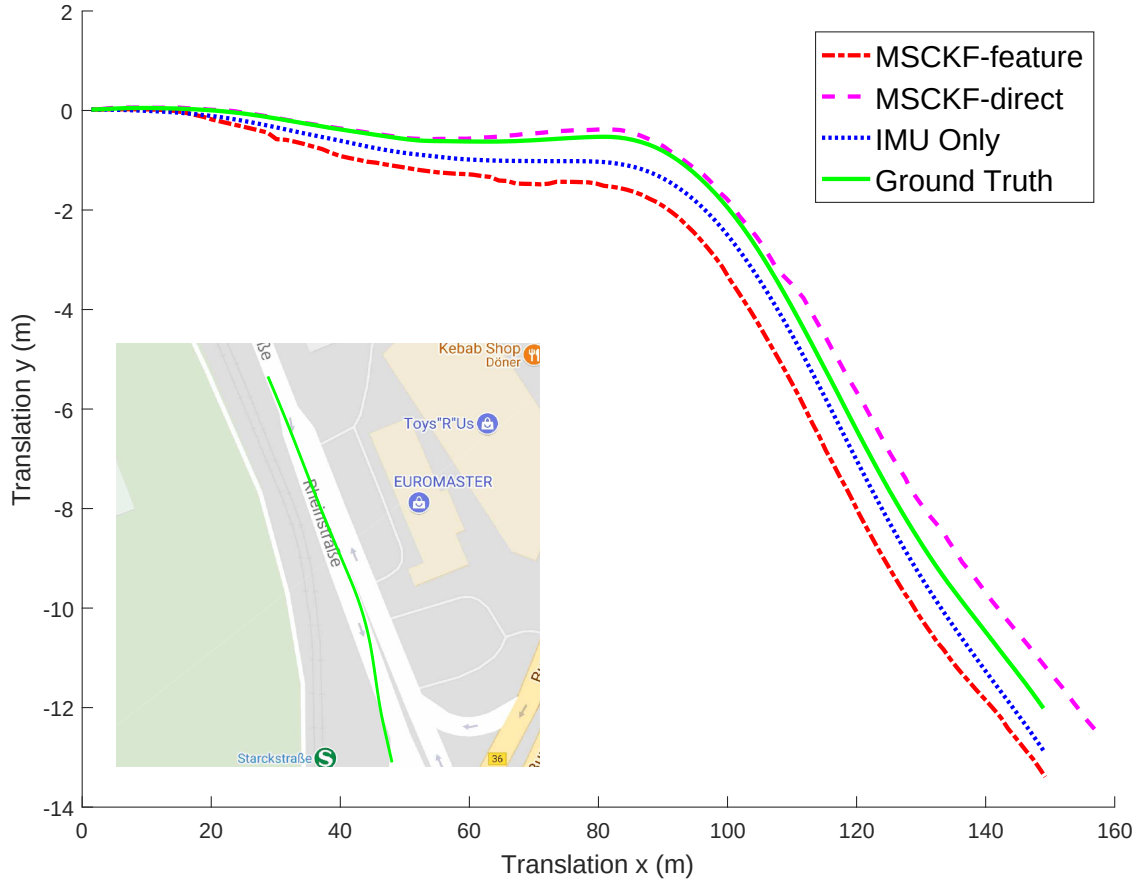
In this section, we first compare our direct-based method with the feature-based MSCKF in public dataset of KITTI [176]. This dataset was built on an automobile equipped with diverse sensors including camera, high precision IMU and laser scanner, which is suitable for research on mobile robots and autonomous driving. The data is well calibrated and synchronised, and the ground truth benchmarks are also provided. Thus it is beneficial for algorithm testing, analysis and comparison. Furthermore, we apply our algorithm to the data we collected from our self-made mobile platform (described in chapter 6), which includes only data from a monocular camera and a consumer-grade IMU. The experiments on our own recorded scenes are targeted at analysing the process and merits of our algorithm in indoor and outdoor applications on the mobile platform.

4.6.1 Translational and rotational accuracy

Our direct method is compared with a feature-based MSCKF method extracting SURF features in scenes and tracking using KLT tracking approach [177]. The case 0051 and 0095 of city category and the case 0036 of residential category in the KITTI dataset are chosen because they contain few moving objects. There are more stable features or image regions with large gradient are available.



(a) Statistic analysis of multiple trials using MSCKF-direct method. The error in both x and y directions are accumulated over the mean trajectory, while the 0.95 confidence ellipse are expanded as algorithm goes on.



(b) The estimated mean trajectories of multiple trails and the IMU-only/GPS recorded trajectories.

Figure 4.5: Estimated trajectories of the case 0051 of city scenes.

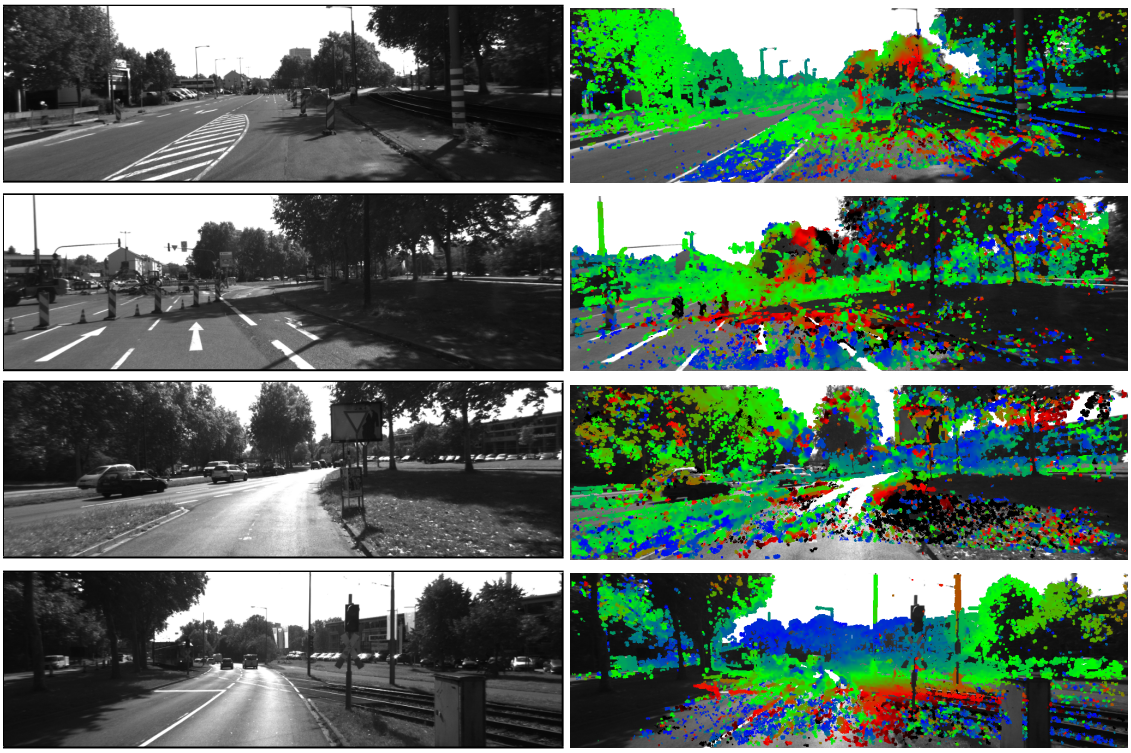
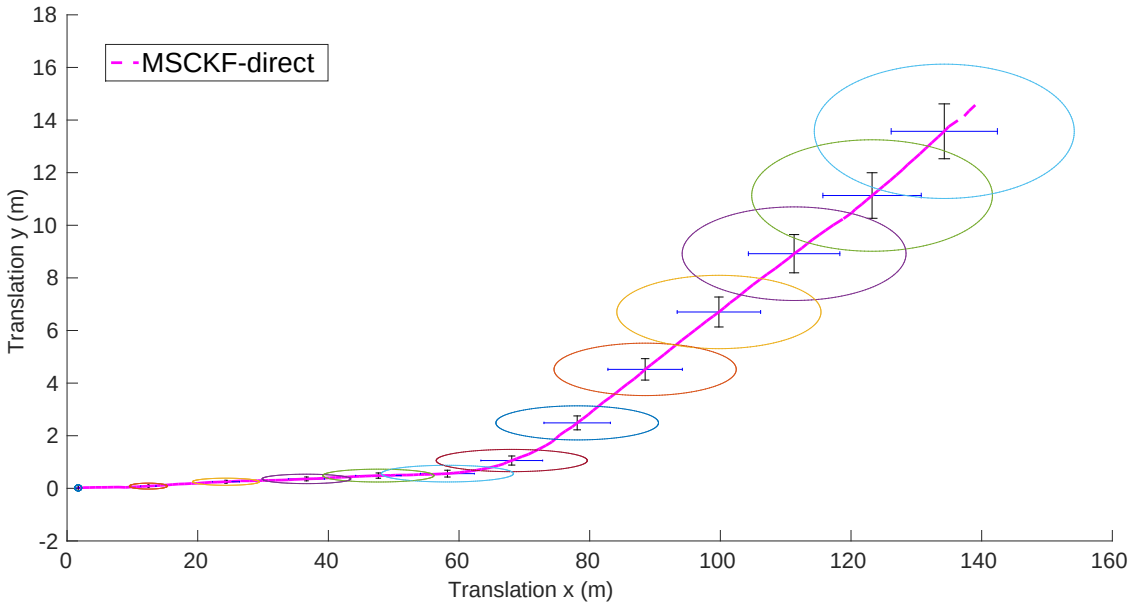
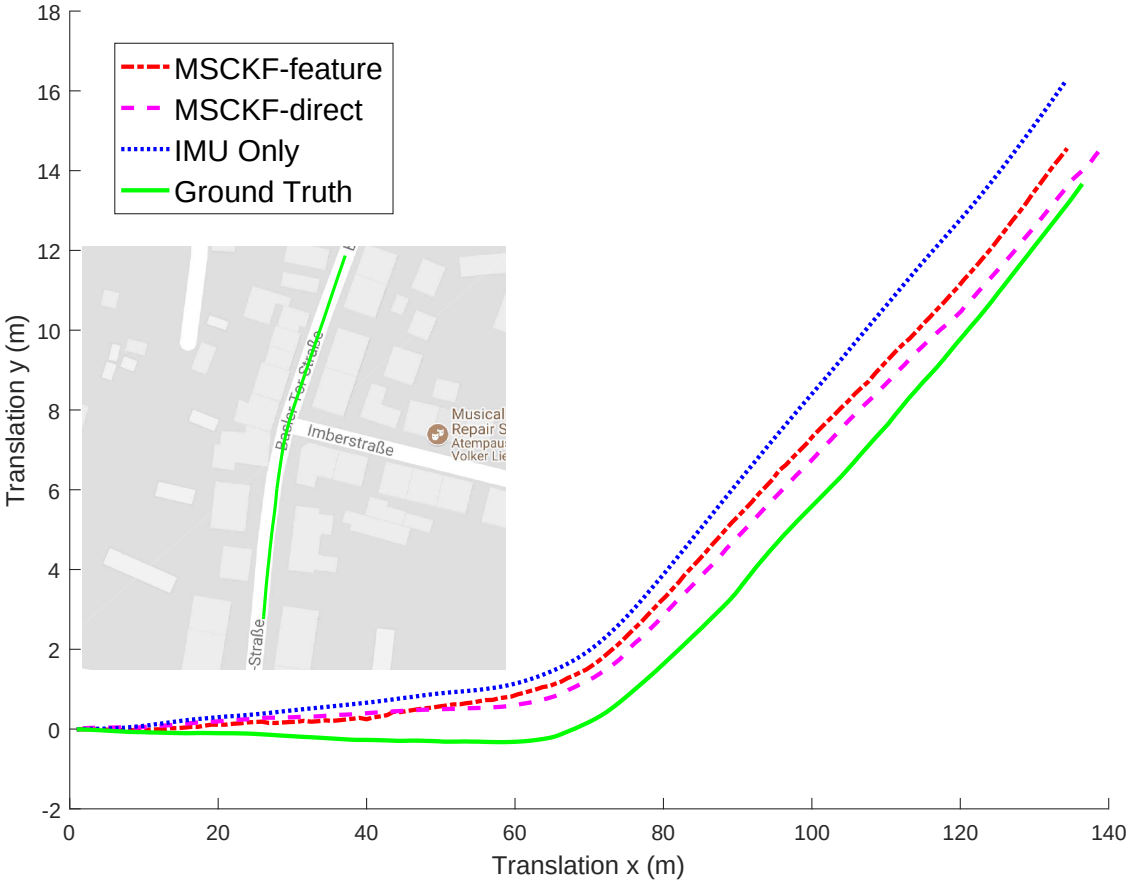


Figure 4.6: Some scene pictures and corresponding keyframes captured from the case 0051 during algorithm running.

Figure 4.5 and 4.7 illustrate multiple trials and depict the statistics information of the case 0051 and 0095 respectively. Some scene pictures and corresponding keyframes in figure 4.6 and 4.8 show scattered significant regions adopted by the direct-based method. The plotted mean trajectories, error and 0.95 confidence ellipses in figure 4.5(a) and 4.7(a) are the statistical results from ten trials in each case. As we can see from such figures, the bias and error are accumulated along with the running distance. Additionally, the estimated mean trajectories in figure 4.5(b) and 4.7(b) show more details than the one recorded from GPS signals, such as the moment for rotations. In both cases, the estimated mean trajectories are divergent from the ground truth as time goes on for both direct-based and feature-based methods, which indicates the unavoidable trend of the essential bias and error accumulated in odometry problem. Although we meet a lot of failure cases when performing the experiments, in these chosen cases, the IMU-only estimation has an acceptable performance, where its estimated trajectory is even better than using the feature-based MSCKF in the case 0051. This is the merit of high-precision devices with low processing noise and using a proper calibration algorithm. Both of the cases are performed using MSCKF as the fusion approach. However, we can see



(a) Statistic analysis of multiple trials using MSCKF-direct method. The error in both x and y directions are accumulated over the mean trajectory, while the 0.95 confidence ellipse are expanded as algorithm goes on.



(b) The estimated mean trajectories of multiple trails and the IMU-only/GPS recorded trajectories.

Figure 4.7: Estimated trajectories for the case 0095 of city scenes.

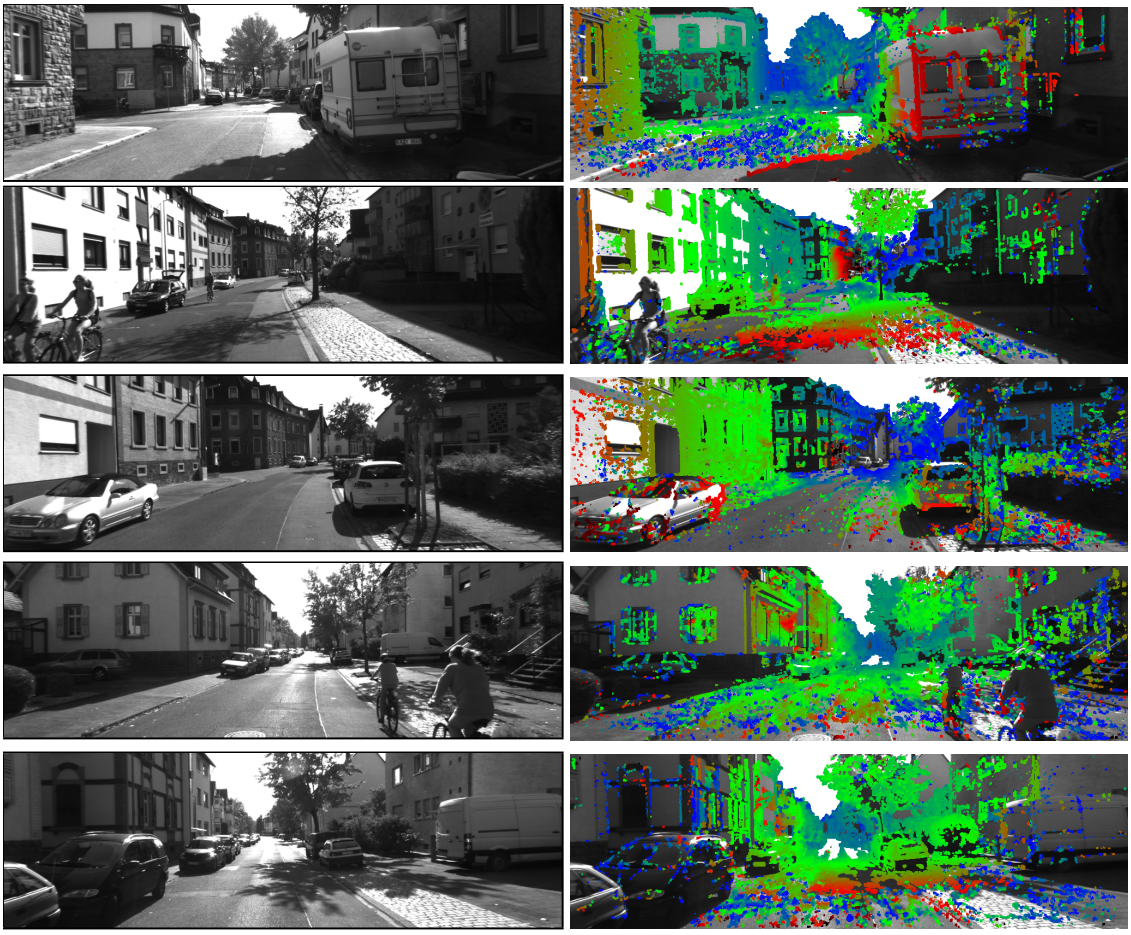
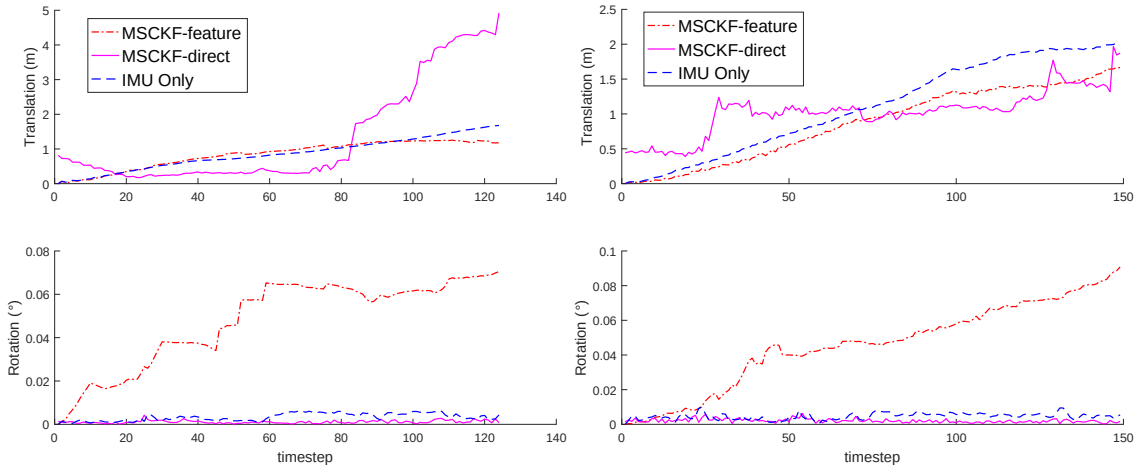


Figure 4.8: Some scene pictures and corresponding keyframes captured from the case 0095 during algorithm running.

that the estimated trajectories of our direct-based method in both cases are closer to the ground truth generally than the feature-based. To analyse the accuracy further, we illustrate the *Root Mean Squared Error* (RMSE) for both cases.

Figure 4.9 shows the RMSE of IMU-only pose estimation in blue dashed line, feature-based MSCKF in red dot-dashed line and our direct-based MSCKF in solid magenta line. In both translation RMSE figures, since the feature-based method [177] using the ground truth to initialize the algorithm, our direct-based method seems more unreliable at the beginning. However, it outperforms the feature-based method after several travel distances, i.e. at 20 timesteps of the case 0051 and at 75 timesteps of the case 0095. The translation errors of the direct-based method tend to more stable than the feature-based method except for the situation of outliers interference and the mismatch happened after 82 timesteps of translation RMSE in the case 0051. This situation also causes a huge error and a suddenly large



(a) RMSE of the case 0051

(b) RMSE of the case 0095

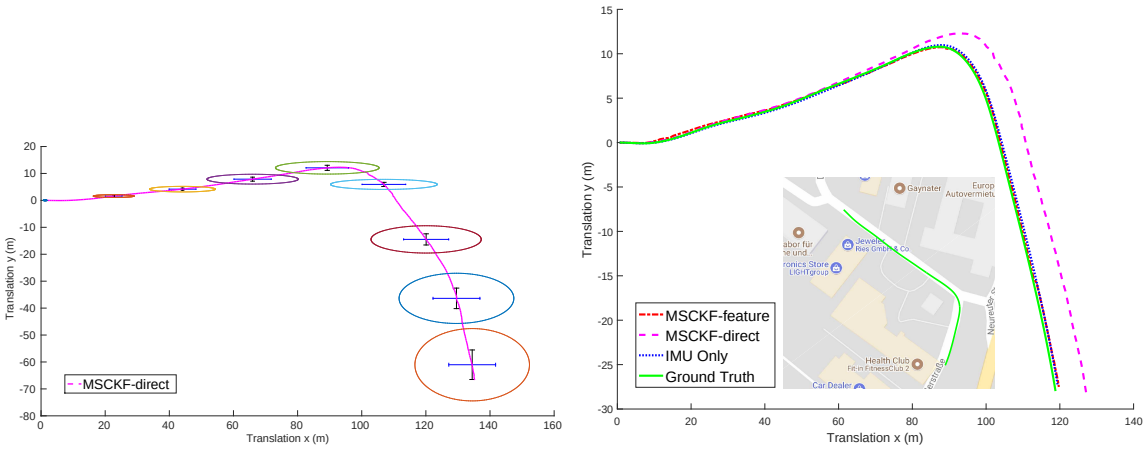
Figure 4.9: Comparison of RMSE of the cases 0051 and 0095. Poses have already been estimated by IMU-only, feature-based and direct-based MSCKF methods.

	KITTI 0095	KITTI 0051 (before polluted at timestep 82)
IMU-only	1.1079	0.5980
Feature-based MSCKF (initialised with ground truth)	0.8651	0.6538
Direct-based MSCKF	**1.0262**	**0.3752**

Table 4.1: Mean RMSE of the translation cases of 0051 and 0095. Direct-based method keeps less or average value in both cases.

divergent in trajectory estimation. However, the estimated trajectory from high-quality inertial data always shows a stable incremental error along the travel. Table 4.1 summarizes the mean RMSE of the three estimates.

As for the rotational error, the direct-based method seems to outperform the feature-based method to a large extent at first glance. However, we notice that it actually contains few sharp and rapid rotational scenario in both chosen cases. Thus we further perform experiments in a case with more turning to compare both methods, one of the cases (0036) in the KITTI dataset is shown on figure 4.10. Unfortunately, through multiple trials and cases, our method does not show significant advantages in sharp and rapid rotational situations. The good performance in our method only happens when more scenes are covered between consecutive image frames, which means that the travelling is near a straight line and the field of view not can be altered too much. The reason for this stems from the essential mechanism of the direct-based method, where the photometric error heavily relies



(a) Statistic results of multiple trials. (b) The estimated and recorded trajectories.

Figure 4.10: Sharp and rapid rotation situation in the residential case 0036 of the KITTI dataset. There exists huge error after the right turn for the direct-based method.

on the pixels positions. Large movements in consecutive frames no matter rotation or not will lead to a large transition of similar pixel regions, thus further making the corresponding patch jump outside from any setting threshold of the searching boundary.

To quantify the overlapping boundary of consecutive images for the direct-based method, we perform more trails in pure rotation situation in the case 0036 and our outdoor experiments (figure 4.17). The right turn scenes and the algorithm propagation can be clearly revealed from figure 4.11, 4.12 and 4.17. As the turning goes on, there firstly exist large new regions in the right part of the whole images, then these regions are soon measured and matched in the algorithm by new incoming images. Other than increasing the rotation speed or image rates to enlarge the novel regions, we manually choose and form the sequences with the proportion of new regions in the whole image as 1/2, 1/3 and 1/4 while keeping the image rate as 10Hz to ensure sufficient calculation with fewer outliers. The results show that the maximum proportion for a new region is 1/3 in our direct-based method, i.e., minimal overlapping for consecutive images is 2/3. When the overlapping rate is under 2/3, tracking failure will mostly happen in operation. The case 0036 illustrated here (figure 4.11) is with 3/4 overlapping proportion and the outdoor case (figure 4.17) is with 2/3 overlapping proportion. Both of them can have stable estimated results. However, if we increase the image rates or motion speed for both cases, in order to catch up the motion and have a true/near real-time estimation, more consecutive

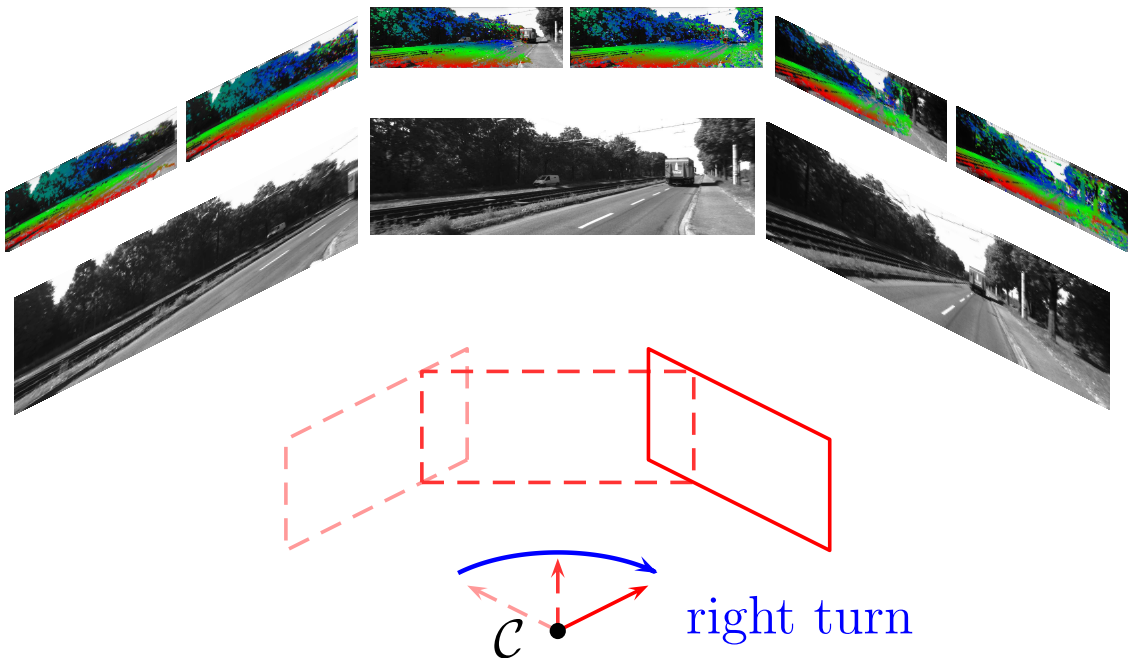


Figure 4.11: The right turn situation in the case 0036. The overlapping proportion of consecutive images is set to 3/4.

images will be discarded. In this situation, the less overlapping proportion leads to rough estimation or even failure in the direct-based method. Therefore, when applying the direct-based method to grantee stable results in the case of large rotation, the motion speed and overlapping proportion in consecutive images need to be carefully considered.

4.6.2 Availability of visual information

In addition to using public dataset, we also collect the data from our lab and an outdoor farm scene by using our self-made mobile platform. The merit of our direct-based method will be further analysed based on the collected data. Since the physical restriction of our mobile platform, only the sensory source as consumer-grade IMU and monocular camera are available on our platform. The essential noise will definitely increase the accumulated error. However, it does not shadow the advantages of the direct-based method.

Figure 4.13 and 4.14 are the scenes snapped from our dataset, where the subfigure 4.13(a) illustrates the feature matching using SURF [64]. As is shown, there are abundant feature points available in the lab scene including some points which we do not even notice. However, most of the good feature points are only limited

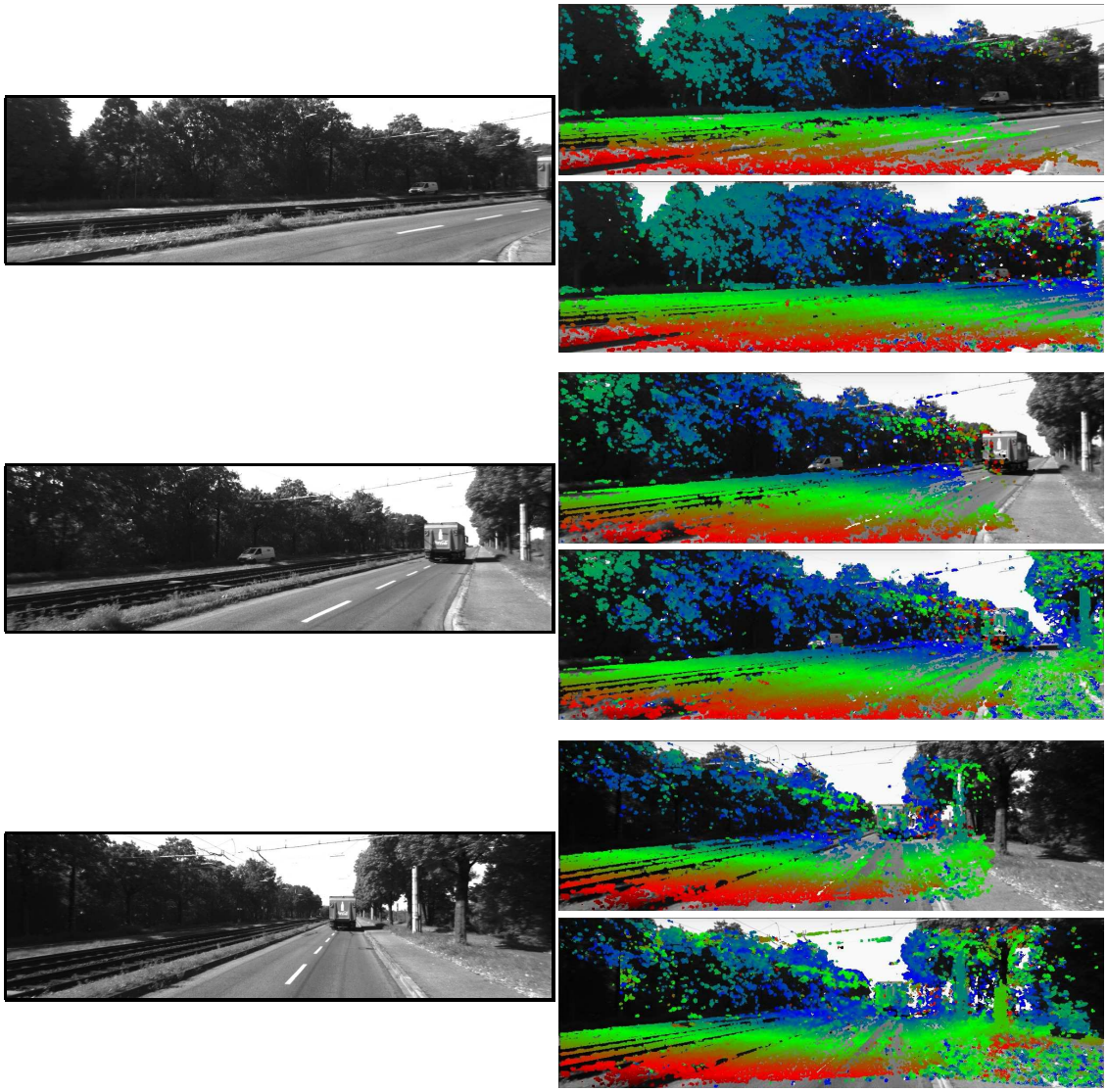
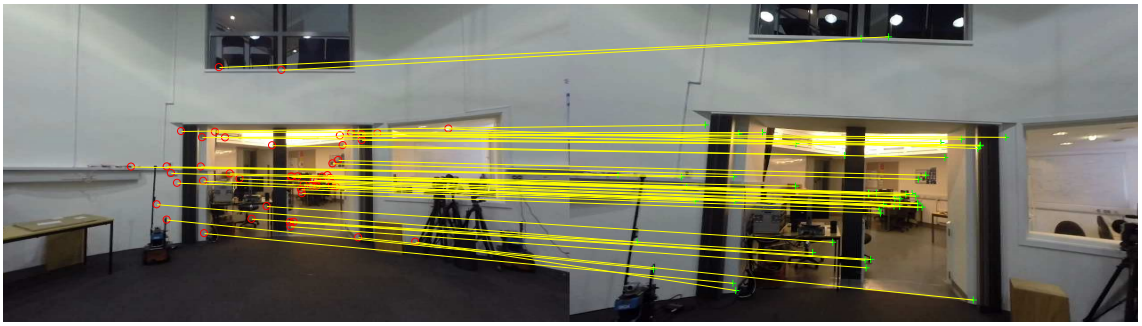


Figure 4.12: Algorithm propagation during turning.

in the region of a distant desk, as shown in figure 4.13(a). The large pillars in front of the scene and their significant edges are ignored. As we can see from the third column of figure 4.13(b), all the prominent edges are adopted to build up depth estimates, and the full visual formation is spanned over the whole image in our direct-based method. This direct-based solution does provide more information than feature-based methods in the aspects of spatial constraints.

This situation becomes apparent when the estimation is applied in an outdoor environment. As shown in figure 4.14(a), the matched features are only available in the woods while large earth scene which occupies a large percentage of the whole image is ignored. From the right column of sub-figure 4.14(b), we can find the



(a) Feature matching in lab scene.

(b) Direct patches region in lab scene.

Figure 4.13: Available visual information in the indoor environment. (a) is the matched feature points, and (b), from left to right, includes the original scenes, significant edges and marked patches in our direct-based method.

texture on the floor is also useful in visual processing, where the significant edge information can be directly extracted. Moreover, even if we can acquire feature matches on the floor through lowering feature searching parameters, it will still cause a multiple corresponding issue in the matching process. But in our method, through adjusting the searching radius with geometry constraints as described in previous section 4.3.2, we still can find the corresponding patches rapidly.

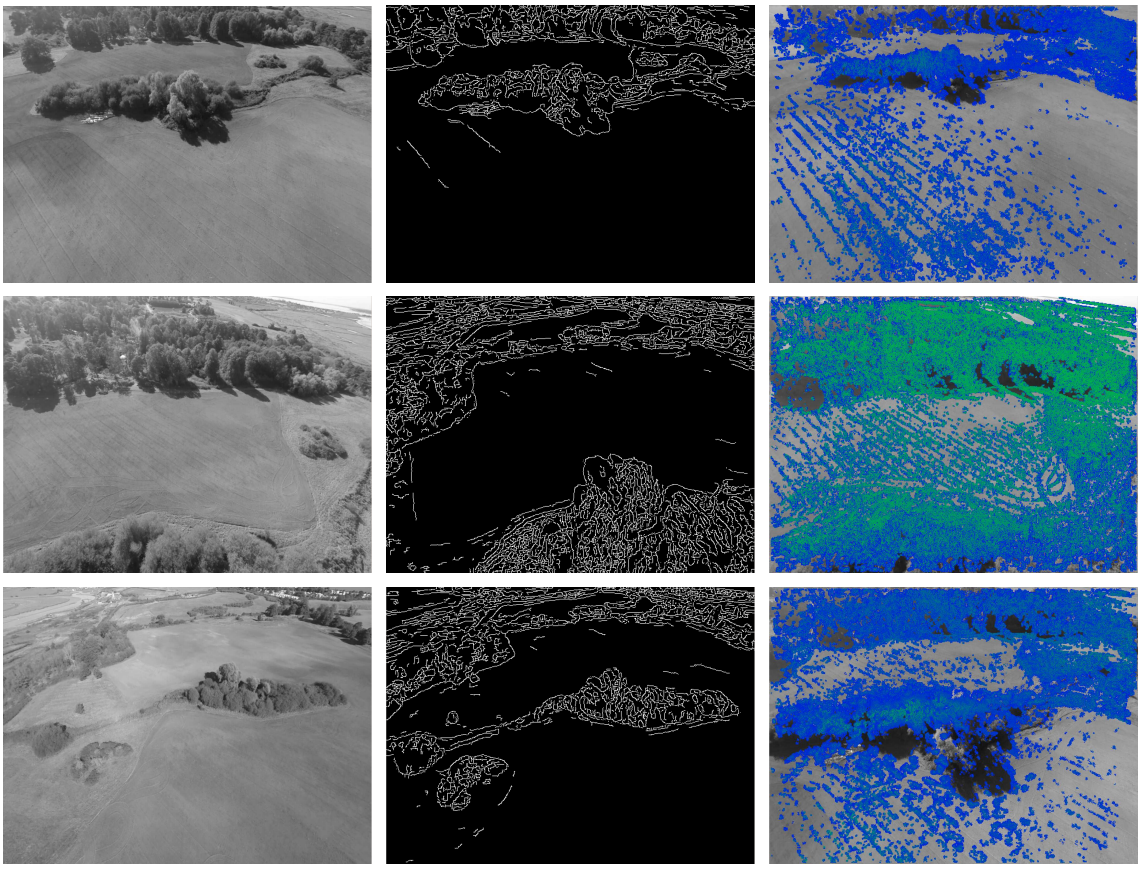
4.6.3 Trajectory estimation

One of the goals of VIO is to estimate the trajectory and let the mobile platform know where it is. In the beginning, we try to estimate the trajectory merely from the inertial data, but no matter what kinds of filtering methods we being used, the estimates will blow up in a few seconds. The online IMU integration with a consumer-grade bias will definitely lead to divergent results. An intuitive example can be found in figure 4.15. Although the raw reading can respond to small movement, the estimates of velocity and position through first and second order integration will be incremental even when the movement stops. While in our direct-based fusion method, the inertial estimates are bounded by visual measurement. More precisely, it is limited by the pose of image frames in the current state. Only the increments between frames will be adopted for pose initial estimation of a new frame. This solution largely reduces the accumulated error in inertial calculation.

Figure 4.16 show the pose estimates in our lab scene, where the self-made quadrotor is controlled travelling around circles at a different altitude. The estimated pose and trajectory can clearly reflect the movement and altitude turning points. In this indoor case, we turn off the lights of the ceiling to check the availability of the direct-based method. As we can see from the frames captured from algorithm running, even under little illumination condition, the direct-based matching can track the significant edges in the scenes. However, it will be failed when applying for feature-based approaches.

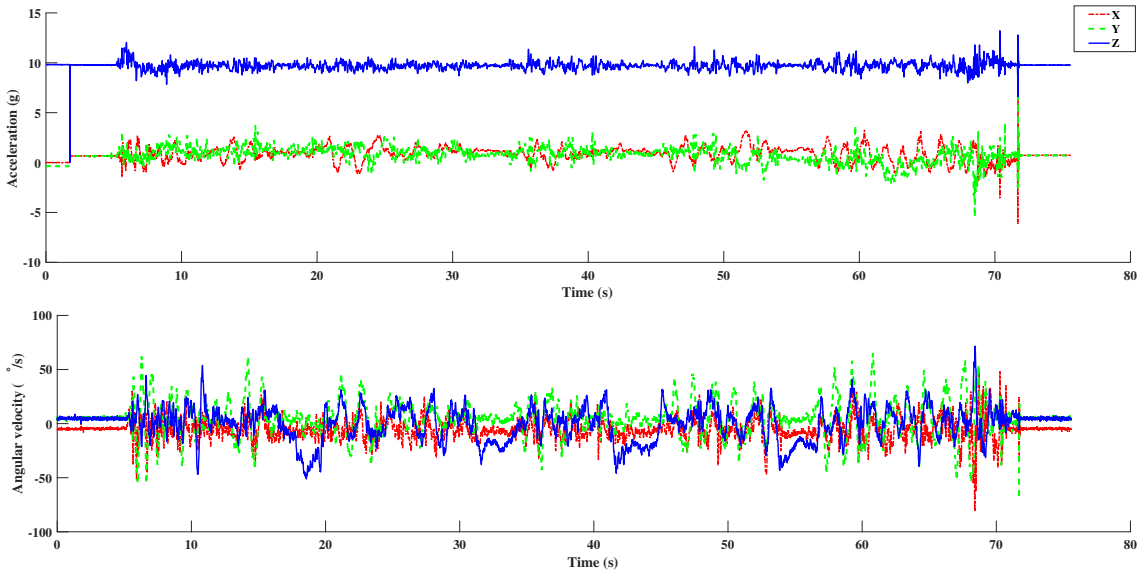
For outdoor experiments, due to the limited capacity of the electric circuit, the GPS sensor is just for controlling but recording, therefore no reliable recorded trajectory can be shown. In figure 4.17, we illustrate the estimated trajectory on a google map where the experiment took place. The estimated trajectory implies that the mobile platform starts from the green point and ends at the blue one. We use a remote panel to control the manoeuvres. The initial and final landing spot was measured by a metre ruler. while the true final position expressed with respect to

(a) Feature matching in outdoor scene.

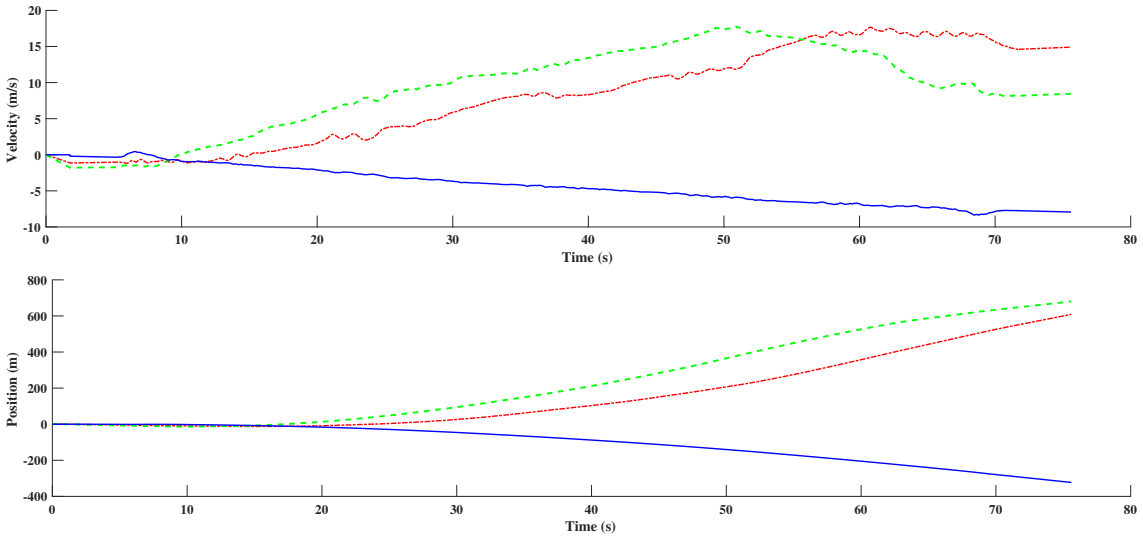


(b) Direct patches region in outdoor scene.

Figure 4.14: Available visual information in outdoor environment. (a) is the matched feature points, and (b), from left to right, includes the original scenes, significant edges and marked patches in our direct-based method.



(a) Raw reading of comsumer-grade IMU. The noise is together with valid signals.



(b) Velocity and position estimation from raw IMU reading. The estimation error is accumulated rapidly as time goes on.

Figure 4.15: Raw reading and further velocity and position estimation of a comsumer-grade IMU.

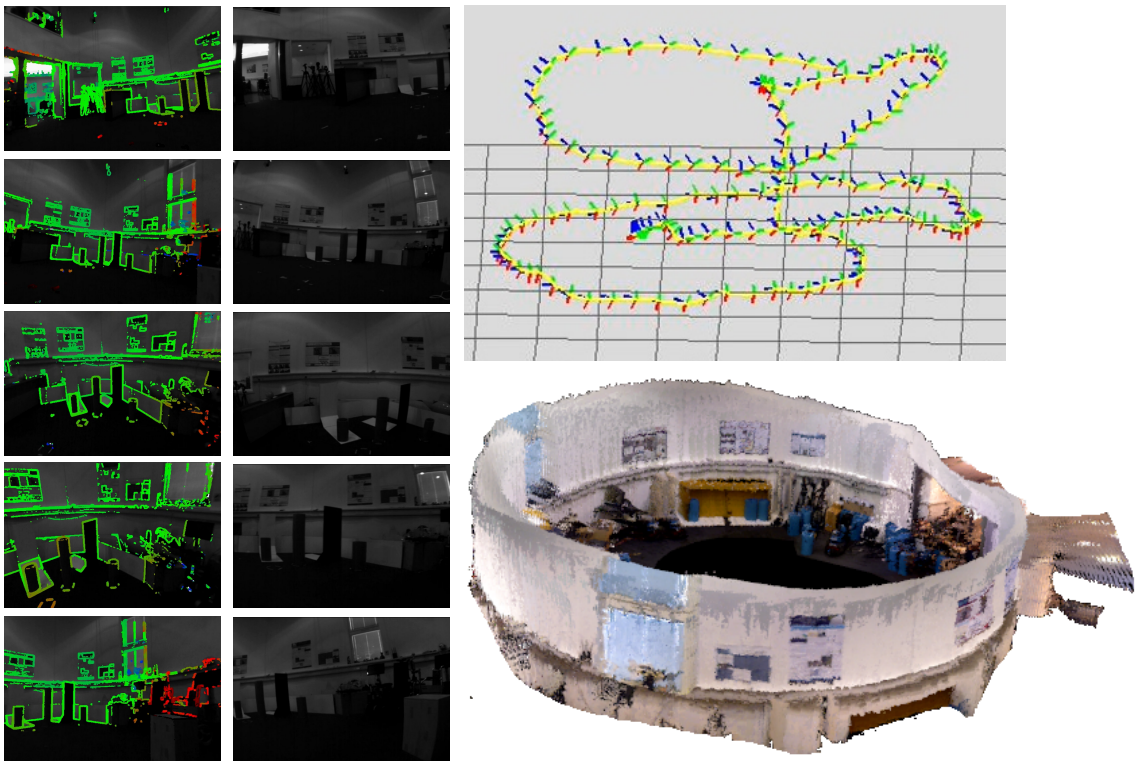


Figure 4.16: Trajectory estimation for the indoor lab scenes. Our direct-based method is evaluated under a low illumination condition, while this estimation will be failed when applying feature-based approaches.

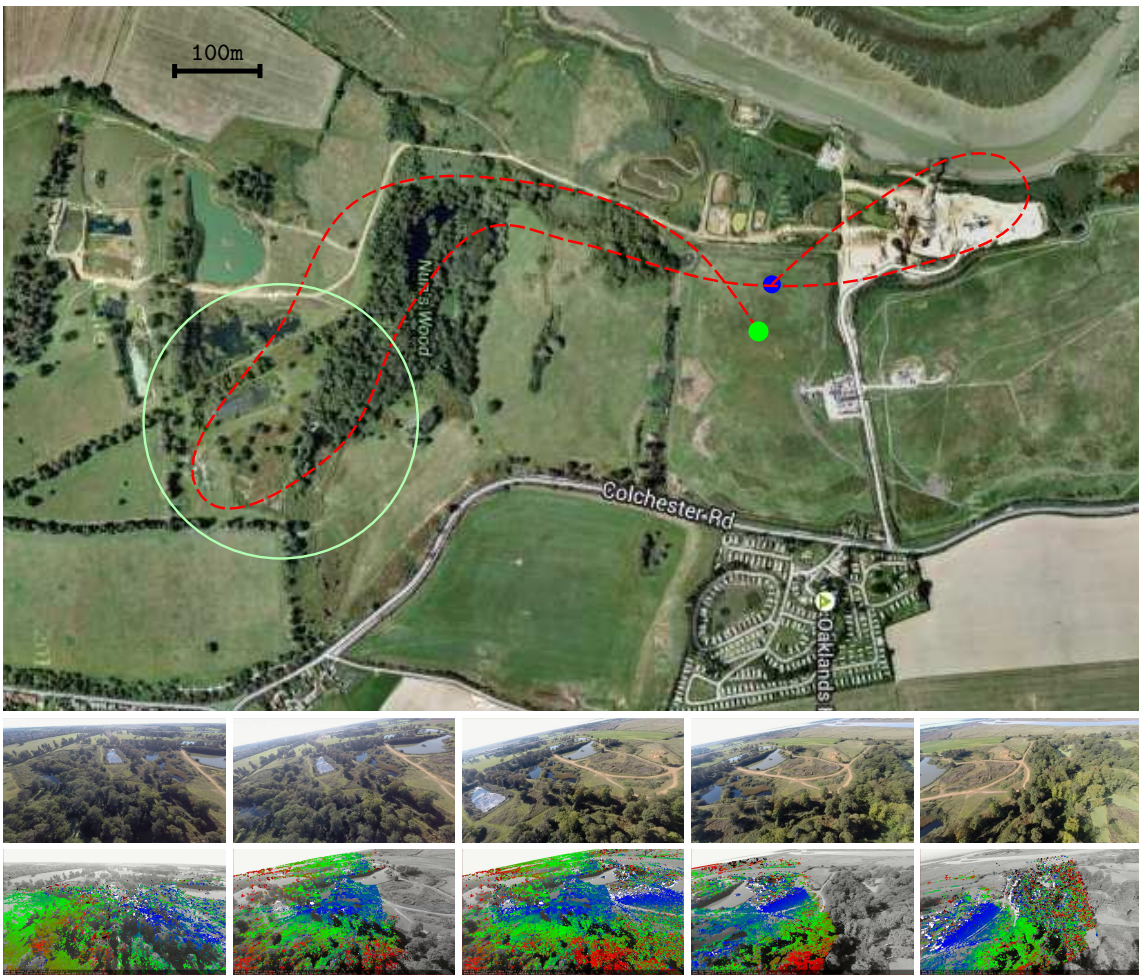


Figure 4.17: Trajectory estimation for the outdoor scenes. The data is collected from a self-made mobile platform. The selected frames in the bottom rows depict the algorithm propagation with the overlapping proportion as 2/3. This is the boundary for our direct-based method.

the original pose is approximately $(-3, -2, -0.5)^T$ m, the estimated final position is $(-8.1352, -9.8206, -3.7473)^T$m. Thus the final position error is approximate 10m in 1.5km travelled distance. This is remarkable that the algorithm does not detect the loop closure or depend on prior information about this area. Additionally, in a rotational situation, we downsample the video stream to 2/3 overlapping regions between consecutive frames. This is the boundary for producing an effective estimation. The algorithm propagation between frame also can be seen in the bottom rows of figure 4.17.

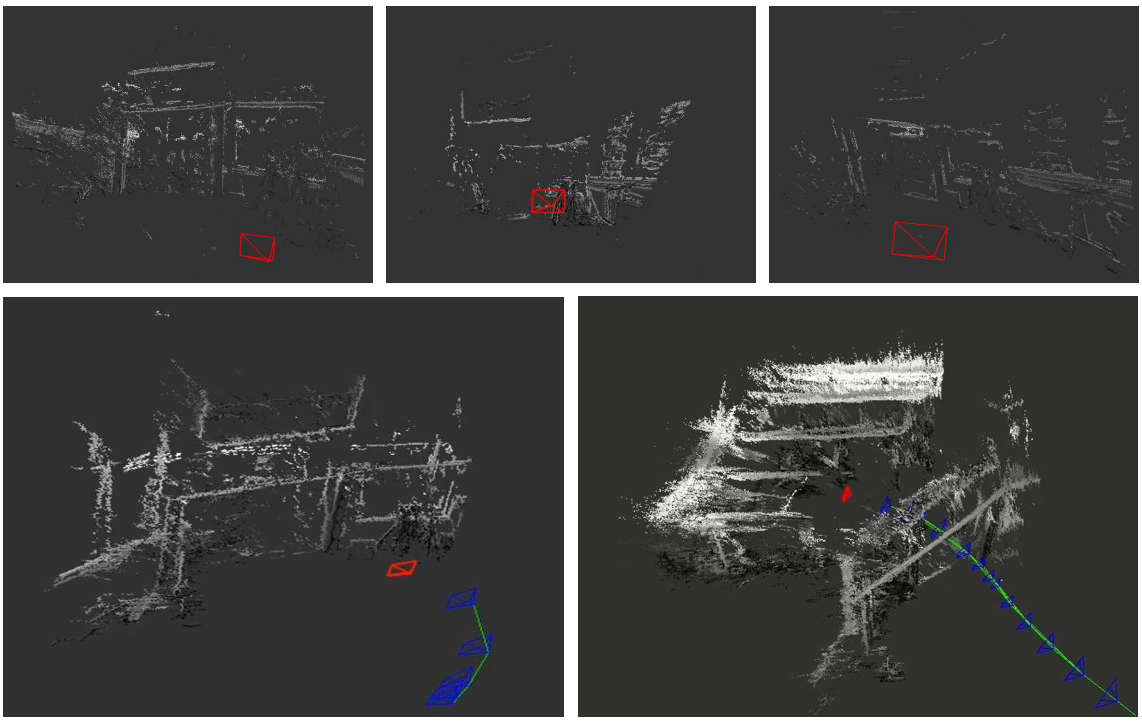


Figure 4.18: Keyframe patches and the merged maps. The top row is the depth estimation from certain keyframes and the bottom row includes two merged maps.

4.6.4 Potentials in mapping

Although the primary task in VIO is not to form a consistent map, the direct-based methods naturally involve more depth estimates in 3D, which builds the basis for a mapping task. The top row of figure 4.18 shows some examples of the depth estimates from keyframes. These patches located at significant edges can construct the basic structure of the scene if we stack the depth estimate from certain keyframes. Ideally, the full structure of the scene can be depicted from all keyframes. However, to achieve a consistent map, more tasks regarding globally consistency or fusion of multiple estimates should be taken into account. As we can see from the bottom row of figure 4.18, we merge several depth estimates from consecutive keyframes in our indoor trials. A blur scene structure can be built but with large overlaps. If this redundancy not can be further removed, the full depth estimates will eventually become chaos. In our direct-based method, to guarantee the algorithm is running efficiently, the states of frames will be altered lively without reuse the marginalised ones. Therefore, we not can build a consistent map while the algorithm is operating, but with further developing, it still owns the potentials.

4.7 Summary

In this section, we present the method of direct-based visual-inertial odometry. Starting from the IMU dynamic model, we illustrate the normal system state and its error form. Then the inertial-driven propagation model is built. The measurement model is constructed on the basic projective model but using photometric information, where corresponding patches instead of features are taken into account in building up the spatial relationship while the algorithm is running. The role of keyframe is important in our approach. It needs to be regarded as a reference for consecutive frames with ongoing refinement. The system states are altered while a new frame is imported. The argumentation and removal of camera pose states are the core for an MSCKF.

The analysis of IEKF bridges the gap and illustrates the intrinsic links between filtering based and optimisation based method, thus providing potential guidance in further developing both methods. The observability analysis clarifies the character of global unobservable and the definition of some time-invariant parameters in our method.

Through the comparison in the experiments, it has been illustrated that our direct-based method outperforms the feature-based one while travelling along straight lines with a slight tolerance of rotation. The direct-based method can make the full use of visual information in a single image, which allows the effective estimation under extreme illumination condition. However, we should carefully control the moving speed of mobile platforms to ensure that the overlapping proportion is within the 2/3 boundary between consecutive images. As long as within this boundary, the trajectories of collected data from our self-made platform can be estimated. Additionally, the stacking depth estimates from keyframes can potentially produce the structure of scene if being further developed.

Chapter 5

Direct Visual-Inertial Odometry based on Mutual Information

5.1 Overview

Mutual information (MI) based measurement method has high robustness in visual-based tasks, especially in the situation of illumination variation, noise interference and partial occlusion cases. In this chapter, we will start from fundamental basis of information theory to the MI expression for digital images, where the reason why MI rather than Joint entropy is suitable for measuring the similarity of two random variables will be discussed. The novel MI-based measurement model, which can replace that of our MSCKF-direct methods will be presented in details. Further the derivation of visual measurement model, a promising iterative process is proposed, which is suitable to build an iterative filtering framework as analysed in the previous chapter. The experimental results show that the MI-based measurement method outperforms other methods in the case of illumination variation, occlusion and noise pollution cases. However, since our motion estimates are based on a known scene model, the original estimation error from the model will lead to a worse error accumulation afterwards. This limits our method only available for short term travelling along straight lines with only slight tolerance of rotation. Additionally, a broad scene image of our experimental environment is built based on MI-based image mosaicing techniques for showcasing its potential mapping ability.

5.2 Probability and Information Entropy

In this section, we will provide some elements of the information theory which are used in our mutual information based visual measurement. Then, the sequential digital images will be regarded as discrete variables, and the derivation process of mutual information on sequential images will be given.

5.2.1 Random variables and their probability

In probability and statistics, a *random variable* is a variable which is unknown but can be assigned to a possible numerical value from a random phenomenon, such as a primary physical experiment outcome. Through multiple trials, we can get the clue of the most likely values of the outcomes, which is given as the form of *probability distribution*.

Let us consider a *discrete* random variable X is defined as the resulting number of dice game. The possible value x are fallen in the set of $\Omega = \{1, 2, 3, 4, 5, 6\}$, then the probability distribution $p(x) = P(X = x)$ means the proportion of times in all experiment trails, when the variable X is equivalent to the value x. As we can expect, each value in the set Ω should have equal probability to occur at any random trial, and the sum of probabilities over all elements in this set should be identical:

$$\sum_{x \in \Omega} p(x) = 1. \tag{5.1}$$

Similarly, the pixel intensities of a digital image in a grey colour map can also be viewed as a random variable, and its value falls into the set of $\Omega = \{0, \cdots, 255\}$ whenever the camera exposure happens.

When considering two random variables X and Y, the concept of *joint probability* should be adopted to express the chance of a couple of values as (x, y) occur at the same time,

$$p(x, y) = P(X = x \cap Y = y). \tag{5.2}$$

The properties of two random variables inherit the one from the single case. The probability is not defined for any value couple outside from a particular set as Ω_X, Ω_Y, and the sum of probabilities is also identical:

$$\sum_{x \in \Omega_X} \sum_{y \in \Omega_Y} p(x, y) = 1. \tag{5.3}$$

Actually, the joint probability measures the link between two random variables. It can be depicted as the joint probability of (x, y) is the probability of Y equals to y when X is known as x or vice versa, which is called as *conditional probability* and noted as,

$$p(x, y) = P(Y = y|X = x)p(x) = P(X = x|Y = y)p(y). \tag{5.4}$$

However, if the happening of $X = x$ is irrelevant to Y, i.e. these two random variables are independent, then the condition expressions are not necessary in above equations because of $P(Y = y|X = x) = P(Y = y) = p(y)$ or $P(X = x|Y = y) = P(X = x) = p(x)$. In this situation, the joint probability can be simply written as

$$p(x, y) = p(x)p(y). \tag{5.5}$$

Although the concept of joint probability can express the links between two different random variables in a way, they still not can give a quantitative comparison between these two variables. Especially for the case when sequential images are regarded as discrete random variables, and their similarity should be clear defined.

5.2.2 Information entropy and mutual information

From the perspective of information theory, Shannon uses the quantity of a variable's information [149], *entropy* $\mathtt{H}(\cdot)$, to measure its variability. For an intuitive understanding, if a variable contains more possibilities as valuable information, then it will be more variable. But if a variable is constant, then there is no information included. The entropy of a discrete random variable X is written as

$$\mathtt{H}(X) = -\sum_{x \in \Omega} p(x) \log_\alpha p(x), \tag{5.6}$$

with the definition of $0 \cdot \infty = 0$. The α represents the logarithm basis, which defines the entropy in a different unit. For instance, when α is set as 2, the entropy of a signal is united in *bit*. The change of different logarithm basis is a matter of applying a scale factor to an entropy value. In this thesis, we will keep using the traditional $\alpha = 2$ as logarithm basis and omit this parameter in the following expressions.

Here, we can understand the meaning of information entropy through an example intuitively. The red and green curves in figure 5.1 illustrate the values for probability, where the probability of a random variable is fallen into $[0, 1]$. The red curve shows that a small probability achieves high value, which can be interpreted as a small

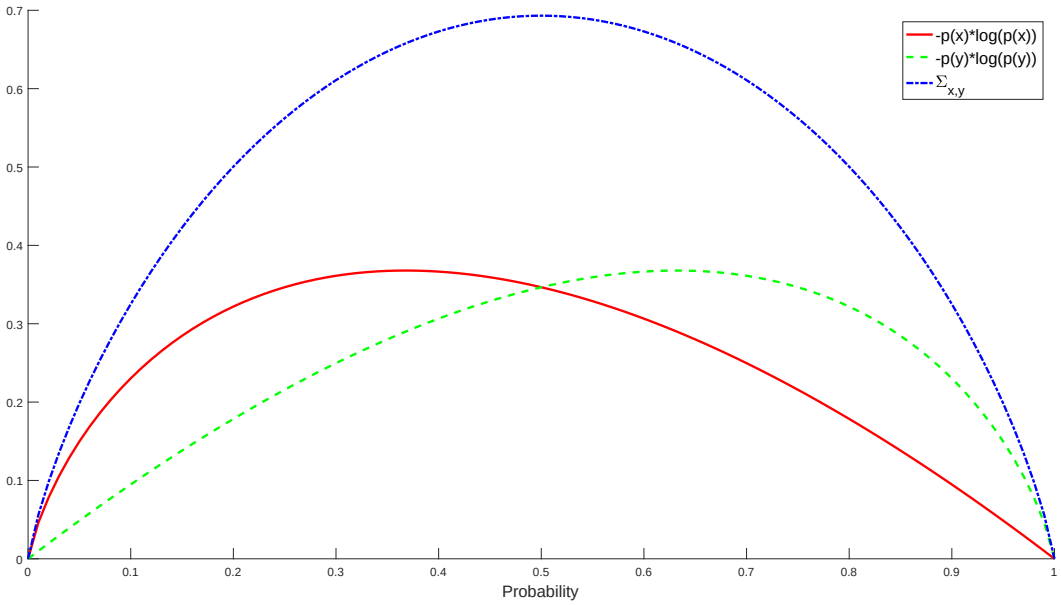


Figure 5.1: Uncertainty added by a value with respect to its probability (a) and entropy of a binary variable with respect to the probability of its first value x_1 (b).

possibility reflects a large uncertainty thus a large amount of valuable information is contained. On contrary, when the variable is near constant, its probability is high up to one, and the uncertainty becomes low in value.

Furthermore, we consider a simple case of one variable with only two possible value, which is restrained by equation (5.3). Then follow equation (5.6), the entropy value of every possible pair can be given as the blue line in figure 5.1. It shows that the maximal value occurs at the middle point, where both of the possible value of X owns the probability of 0.5. This is a uniform distribution with the meaning of every possible value happens in equal chance. When the variable is constant, which means $p(x_1) = 1$ or $p(x_2) = 1$, the null value of entropy will occur.

Similar to the definition of information entropy, the *joint entropy* describe the variability of the couple of random variables (X, Y) based on joint probability, which can be expressed as

$$\mathtt{H}(X, Y) = -\sum_{x \in \Omega_X} \sum_{y \in \Omega_Y} p(x, y) \log_\alpha p(x, y). \tag{5.7}$$

The joint entropy seems to provide a measurement about the correlation between two variables. However, following the example in figure 5.1, we can see that, if the probability of X and Y are equal, $p(x, y) = p(x) = p(y)$, then the joint entropy becomes the same as individual single entropy, i.e. $\mathtt{H}(X, Y) = \mathtt{H}(X) = \mathtt{H}(Y)$. But when

one of the variables is constant, its entropy becomes zero. X and Y are irrelevant, so joint entropy becomes the individual entropy $\mathtt{H}(X,Y) = \mathtt{H}(X)$ or $\mathtt{H}(X,Y) = \mathtt{H}(Y)$. These two situations above not can be distinguished if only using joint entropy to measure their similarity.

If taking the mutual dependency between variable entropies into account, the *conditional entropy* is defined, which describes the system uncertainty when parts of the information are known. The definition of the system of variables (X,Y) with known X is given by

$$\mathtt{H}(Y|X) = \mathtt{H}(X,Y) - \mathtt{H}(X). \tag{5.8}$$

However this expression still not can be used to measure the similarity of both variables. When two variables are completely correlated with respect to each other, the conditional entropy is zero because of $\mathtt{H}(X,Y) = \mathtt{H}(X) = \mathtt{H}(Y)$. And if the two variables are independent, the conditional entropy becomes individual one as $\mathtt{H}(Y|X) = \mathtt{H}(Y)$. There exists ambiguity if X is constant and $\mathtt{H}(Y) = 0$.

To overcome the ambiguity above, a solution must be considered to remove the dependency of conditional entropy for individual one, which can be formulated as the *mutual information* (MI),

$$\mathtt{MI}(X,Y) = \mathtt{H}(Y) - \mathtt{H}(Y|X) = \mathtt{H}(X) + \mathtt{H}(Y) - \mathtt{H}(X,Y). \tag{5.9}$$

This expression defines the difference of variability between the variable Y and the variable Y with known X as a condition. The more dependent this two variables are, the larger value of MI will be. If these two variable are tightly correlated to each other, i.e. $\mathtt{H}(X) = \mathtt{H}(Y)$, then $\mathtt{H}(X,Y) = \mathtt{H}(X) = \mathtt{H}(Y)$ and $\mathtt{MI}(X,Y) = \mathtt{H}(X) = \mathtt{H}(Y)$ reach the maximum. And if one of the variables is constant, say X is constant, then $\mathtt{H}(Y|X) = \mathtt{H}(Y)$ and $\mathtt{MI}(X,Y)$ becomes zero. Therefore, the concept of MI can be used to measure the amount of information that is shared by the two variables.

5.2.3 Digital images as random variables

From another point of view, a digital image as a matrix can be regarded as a discrete random variable about image intensities. The pixel units record intensity values from a certain set, which is usually $\Omega = [0, 255] \subset \mathbb{N}$ in gray-level images. The probability $p(i)$ of value $i \in [0, 255]$ is the proportion of such value in one image I, which is estimated using the image histogram and normalised by the total number

of pixels,

$$p_{\mathrm{I}}(i) = \frac{1}{N} \sum_{x} \delta(i - \mathrm{I}(x)), \tag{5.10}$$

where x is the location of pixel unit in total number of N and $\delta(\cdot)$ defines the Kronecker delta as

$$\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases}. \tag{5.11}$$

The computation of image histogram is similar to a polling process. At the time of $\mathrm{I}(x) = i$, the i^{th} histogram bin value is incremented.

By importing the concept of entropy and applying the probability of the pixel values into equation (5.6), we can get

$$\mathtt{H}(\mathrm{I}) = -\sum_{i \in \Omega} p_{\mathrm{I}}(i) \log p_{\mathrm{I}}(i). \tag{5.12}$$

This expression measures the dispersion of image histogram, which reflecting the variability of one image.

Furthermore, for the case of two images, the joint probability of a pair of pixels (i, j) from two images reflects the proportion of their occurrence times at corresponding locations. For two images having the same size of total N pixel units, this probability can be expressed as

$$p(i, j) = \frac{1}{N} \sum_{x} \delta(i - \mathrm{I}(x)) \delta(j - \mathrm{I}_k(x)) \tag{5.13}$$

Substituting above equation into joint entropy (equation (5.7)) and yielding,

$$\mathtt{H}(\mathrm{I}, \mathrm{I}_k) = -\sum_{i,j \in \Omega} p_{\mathrm{II}_k}(i, j) \log p_{\mathrm{II}_k}(i, j). \tag{5.14}$$

Following the definition of MI (equation (5.9)), we can derive the MI expression of

a pair of images as

$$\begin{aligned}\mathtt{MI}(\mathrm{I}, \mathrm{I}_k) &= \mathtt{H}(\mathrm{I}) + \mathtt{H}(\mathrm{I}_k) - \mathtt{H}(\mathrm{I}, \mathrm{I}_k) \\ &= -\sum_i p_{\mathrm{I}}(i) \log p_{\mathrm{I}}(i) - \sum_j p_{\mathrm{I}_k}(j) \log p_{\mathrm{I}_k}(j) + \sum_{i,j} p_{\mathrm{II}_k}(i,j) \log p_{\mathrm{II}_k}(i,j) \\ &= \sum_{i,j} -p_{\mathrm{I}}(i) \log p_{\mathrm{I}}(i) - p_{\mathrm{II}_k}(i,j) \log p_{\mathrm{I}_k}(j) + p_{\mathrm{II}_k}(i,j) \log p_{\mathrm{II}_k}(i,j) \\ &= \sum_{i,j} p_{\mathrm{II}_k}(i,j) \log \left(\frac{p_{\mathrm{II}_k}(i,j)}{p_{\mathrm{I}}(i) p_{\mathrm{I}_k}(j)} \right).\end{aligned} \tag{5.15}$$

5.3 MI-based Measurement Model for Filtering

In this section, we will present the MI-based visual measurement model in detail. We assume that a scene model with a certain amount of spatial points in world frame $\mathcal{W}$ is known. Like our previous method, a semi-dense depth estimation for a local scene can be acquired by saving the keyframes information. Here, in the MI-based visual model, such keyframe and their scene estimation will be reused to estimate a relative pose from keyframe to current frame. The framework as filtering will be adopted again to fuse the inertial information. Therefore, the goal here is simplified to build the MI-based visual measurement model and give its necessary derivatives with respect to camera poses.

Firstly, we recall the state update formulation of our visual model as photometric information in section 4.3.3. The measurement model targets to build a residual formulation concerning an increment to current state:

$$\tilde{\boldsymbol{z}} = \boldsymbol{z} - \hat{\boldsymbol{z}} = \boldsymbol{H}\tilde{\boldsymbol{x}} + \boldsymbol{n}_p, \tag{5.16}$$

where the $\tilde{\boldsymbol{x}}$ is the error state containing the camera state ${}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_i}$ and $\boldsymbol{n}_p$ is the Gaussian noise during processing. The processing noise $\boldsymbol{n}_p$ includes a zero-mean Gaussian white noise with covariance σ_p and all the other constant term. Then the covariance matrix can be expressed as $\boldsymbol{R} = \sigma_p^2 \mathbf{I}_N$ with current N camera poses in state vector. The notation $\boldsymbol{H}$ defines the observation matrix, which includes the

Jacobian with respect to the current state, which can be given as

$$\boldsymbol{H} = \begin{bmatrix} \mathbf{0} & \cdots & \underbrace{\boldsymbol{J}_i^{\boldsymbol{q}} \quad \boldsymbol{J}_i^{\boldsymbol{p}}}_{\text{Jaccobian w.r.t.}\,{}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_i}} & \cdots \end{bmatrix}, \quad i \in [1, N]. \tag{5.17}$$

Different from the previous chapter, here we use the mutual information between nearest keyframe and image taken at pose ${}^{\mathcal{W}}\hat{\boldsymbol{\chi}}_{\mathcal{C}_i}$ as the observation, which can be expressed as

$$\hat{\boldsymbol{z}} = \mathtt{MI}(\mathrm{I}_k, \mathrm{I}_{kc_i}({}^{\mathcal{W}}\hat{\boldsymbol{\chi}}_{\mathcal{C}_i})). \tag{5.18}$$

It should be noticed that the camera pose ${}^{\mathcal{W}}\boldsymbol{\chi}_{\mathcal{C}_i}$ actually can be updated by the incremental translation $\boldsymbol{\upsilon}$ and rotation $\boldsymbol{\omega}$, as defined in section 3.2.5. Thus the pose can be updated using the exponential map as the form of $\boldsymbol{\chi}_{k+1} = \exp(\boldsymbol{\xi})\boldsymbol{\chi}_k$. In the following derivation, we only use the minimal representation of pose $\boldsymbol{\xi}$ and its increment $\delta\boldsymbol{\xi}$ for a concise notation.

Whenever a new image arrives, an initial estimation of its pose will be given from the inertial propagation step. The other observed pose is acquired through finding a relative pose from nearest keyframe to current image. This process is measured by maximising the MI between the current image and a virtual image taken at estimated pose from scene model. This step can be regarded as an optimisation process to finding the maxima. Following the definition in section 3.3.3, it can be expressed as

$$\delta\boldsymbol{\xi}^* = \arg\max \mathtt{MI}(\mathrm{I}_{c_i}, \mathrm{I}_{kc_i}(\delta\hat{\boldsymbol{\xi}})). \tag{5.19}$$

For the requirement of efficiency, we can use a simple solution from Gauss-Newton method, where only the Jacobian of MI function, $\boldsymbol{J}$, is required during iterative calculation. Once the relative pose is given, the current camera pose as observation can be achieved by adding ($\oplus$) the pose from the referred keyframe.

The mutual information essentially measures the similarity between two images. In our measurement model, we reuse this fact that similar image pairs should have the same MI value to the third one. Here, the third image is the nearest keyframe, which is already known. Thus, the residual of measurement can be expressed as

$$\tilde{\boldsymbol{z}} = h(\delta\hat{\boldsymbol{\xi}}) = \sum_{i=1}^{N} \mathtt{MI}(\mathrm{I}_k, \mathrm{I}_{c_i}) - \mathtt{MI}(\mathrm{I}_k \mathrm{I}_{kc_i}(\delta\hat{\boldsymbol{\xi}})). \tag{5.20}$$

To form the observation matrix $\boldsymbol{H}$, the Jacobian of MI is required again. Thus, the core problem here of building an MI-based measurement model is to get the Jacobian of MI.

5.3.1 MI Jacobian

Applying the derivation chain rule to MI definition equation (5.15), we can get the resulting mutual information derivative as,

$$\boldsymbol{J} = \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \log \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}} p_{\mathrm{I}_k}} + \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} - \frac{\partial p_{\mathrm{I}}}{\partial \delta \boldsymbol{\xi}} \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}}, \tag{5.21}$$

where the probabilities and joint probability which are actually depending on i, j are simply denoted as $p_{\mathrm{I}}, p_{\mathrm{I}_k}$ and p_{II_k}. With the fact that the summation of the probability is constant to one, $\sum p = 1$, its derivative is always equals to zero.

Thus, the last term of above equation (5.21) can be decomposed and simplified as

$$\begin{aligned} \sum_{i,j} \frac{\partial p_{\mathrm{I}}}{\partial \delta \boldsymbol{\xi}} \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}} &= \sum_{i} \sum_{j} \frac{\partial p_{\mathrm{I}}}{\partial \delta \boldsymbol{\xi}} \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}} \\ &= \sum_{i} \frac{\partial p_{\mathrm{I}}}{\partial \delta \boldsymbol{\xi}} \frac{1}{p_{\mathrm{I}}} \sum_{j} p_{\mathrm{II}_k} \\ &= \sum_{i} \frac{\partial p_{\mathrm{I}}}{\partial \delta \boldsymbol{\xi}} = 0. \end{aligned} \tag{5.22}$$

Then the MI Jacobian becomes

$$\begin{aligned} \boldsymbol{J} &= \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \left(1 + \log \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}} p_{\mathrm{I}_k}} \right) \\ &= \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \left(1 + \log \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}} - \log(p_{\mathrm{I}_k}) \right) \\ &= \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \left(1 + \log \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}} \right) - \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \log(p_{\mathrm{I}_k}) \\ &= \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \left(1 + \log \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}} \right) - \sum_{j} \frac{\partial p_{\mathrm{I}_k}}{\partial \delta \boldsymbol{\xi}} \log(p_{\mathrm{I}_k}). \end{aligned} \tag{5.23}$$

Since the derivative of the probability is zero in the last term above, the final

expression of the Jacobian of mutual information can be written as

$$\boldsymbol{J} = \sum_{i,j} \frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} \left(1 + \log \frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}}}\right). \tag{5.24}$$

5.3.2 Joint probability

As we can see from the final Jacobian equation (5.24), the derivation process of joint probability p_{II_k} needs to be further expanded for computation. Here, we recall the original definition of joint probability of equation (5.13) and apply a wrap function $w(\cdot)$ into it. The form of wrap function is similar to the one used in section 4.3.2. Thus, the joint probability expression becomes

$$p_{\mathrm{II}_k}(i, j, \boldsymbol{\xi} \odot \delta\boldsymbol{\xi}) = \frac{1}{N} \sum_{\mathbf{u}} \phi\left(i - \mathrm{I}\left(w\left(\mathbf{u}, \boldsymbol{\xi} \odot \delta\boldsymbol{\xi}\right)\right)\right) \phi\left(j - \mathrm{I}_k\left(\mathbf{u}\right)\right), \tag{5.25}$$

where $\mathbf{u}$ is the pixel location in the reference frame, N is the total number of pixels taken into account and $\phi(\cdot)$ is a B-spline function used to smooth the MI function. The general definition of ϕ_n with order n can be expressed recursively as

$$\phi_n(\varepsilon) = (\phi_{n-1} * \phi_1)(\varepsilon) \qquad \text{with} \quad \phi_1 = \begin{cases} 1 & \text{if} \quad \varepsilon \in [0.5, 0.5] \\ 0 & \text{otherwise} \end{cases}, \tag{5.26}$$

where the notation $*$ denotes the convolution operation.

By applying the derivative to the joint probability of equation (5.25) we can get the following expression:

$$\frac{\partial p_{\mathrm{II}_k}}{\partial \delta \boldsymbol{\xi}} = \frac{1}{N} \sum_{\mathbf{u}} \frac{\partial \phi\left(i - \mathrm{I}\left(w\left(\mathbf{u}, \boldsymbol{\xi} \odot \delta\boldsymbol{\xi}\right)\right)\right)}{\partial \delta \boldsymbol{\xi}} \phi\left(j - \mathrm{I}_k\left(\mathbf{u}\right)\right). \tag{5.27}$$

The B-spline function derivative can be expressed by applying the chain rule, thus yielding,

$$\frac{\partial \phi\left(i - \mathrm{I}\left(w\left(\mathbf{u}, \boldsymbol{\xi} \odot \delta\boldsymbol{\xi}\right)\right)\right)}{\partial \delta \boldsymbol{\xi}} = \frac{\partial \phi}{\partial \varepsilon} \frac{\partial \mathrm{I}\left(w\left(\mathbf{u}, \boldsymbol{\xi} \odot \delta\boldsymbol{\xi}\right)\right)}{\partial \delta \boldsymbol{\xi}}, \tag{5.28}$$

where the derivative of B-spline function is computed by using difference of lower order B-splines:

$$\frac{\partial \phi_n(\varepsilon)}{\partial \varepsilon} = \phi_{n+1}\left(\varepsilon + \frac{1}{2}\right) - \phi_{n+1}\left(\varepsilon - \frac{1}{2}\right). \tag{5.29}$$

The second term of equation (5.28) is the partial derivative of the current image intensity with respect to the updated pose. It can be further decomposed by

$$\begin{aligned}\frac{\partial \mathrm{I}\left(w\left(\mathbf{u}, \boldsymbol{\xi} \odot \delta\boldsymbol{\xi}\right)\right)}{\partial \delta\boldsymbol{\xi}} &= \nabla \mathrm{I} \frac{\partial w(w(\mathbf{u}, \delta\boldsymbol{\xi}), \boldsymbol{\xi})}{\partial \delta\boldsymbol{\xi}} \\ &= \nabla \mathrm{I} \frac{\partial w(\mathbf{u}', \boldsymbol{\xi})}{\partial \mathbf{u}'} \frac{\partial w(\mathbf{u}, \delta\boldsymbol{\xi})}{\partial \delta\boldsymbol{\xi}} \\ &= \nabla \mathrm{I}(\mathbf{u}') \frac{\partial w(\mathbf{u}, \delta\boldsymbol{\xi})}{\partial \delta\boldsymbol{\xi}}\end{aligned} \tag{5.30}$$

where the $\nabla \mathrm{I} = (\nabla_u \mathrm{I}, \nabla_v \mathrm{I})$ is the gradient of image I with respect to the directions of axes at one certain pixel point and the second term represents the link between the wrap function and the pose update, which is related to a projection process.

5.3.3 Wrap function

In previous section 3.1.2, we formulate the forward process that maps a spatial point from camera frame to an image plane, which can be rewritten from equation (3.5) with standard camera intrinsic parameters, as

$$u = \frac{x_{\mathrm{c}}^{\mathbf{p}}}{z_{\mathrm{c}}^{\mathbf{p}}} \quad \text{and} \quad v = \frac{y_{\mathrm{c}}^{\mathbf{p}}}{z_{\mathrm{c}}^{\mathbf{p}}}. \tag{5.31}$$

The update pose incremental is tightly related to the variation of pixels, thus we differentiate this expression and get

$$\begin{aligned}\dot{u} &= \frac{\dot{x}_{\mathrm{c}}^{\mathbf{p}}}{z_{\mathrm{c}}^{\mathbf{p}}} - \frac{x_{\mathrm{c}}^{\mathbf{p}} \dot{z}_{\mathrm{c}}^{\mathbf{p}}}{\left(z_{\mathrm{c}}^{\mathbf{p}}\right)^2} = \frac{(\dot{x}_{\mathrm{c}}^{\mathbf{p}} - u\dot{z}_{\mathrm{c}}^{\mathbf{p}})}{z_{\mathrm{c}}^{\mathbf{p}}} \\ \dot{v} &= \frac{\dot{y}_{\mathrm{c}}^{\mathbf{p}}}{z_{\mathrm{c}}^{\mathbf{p}}} - \frac{y_{\mathrm{c}}^{\mathbf{p}} \dot{z}_{\mathrm{c}}^{\mathbf{p}}}{\left(z_{\mathrm{c}}^{\mathbf{p}}\right)^2} = \frac{(\dot{y}_{\mathrm{c}}^{\mathbf{p}} - v\dot{z}_{\mathrm{c}}^{\mathbf{p}})}{z_{\mathrm{c}}^{\mathbf{p}}}\end{aligned}. \tag{5.32}$$

As we can see from above equations, the derivative of pxiel position is related to the derivative of the spatial point. However, in world frame, these points are static, we can only regard a relative motion of the camera as $\boldsymbol{\xi} = (\boldsymbol{\nu}, \boldsymbol{\omega})$ to express this differential

$$\dot{\mathbf{x}}_{\mathrm{c}}^{\mathbf{p}} = -\boldsymbol{\nu} - \lfloor \boldsymbol{\omega} \times \rfloor \mathbf{x}_{\mathrm{c}}^{\mathbf{p}}, \tag{5.33}$$

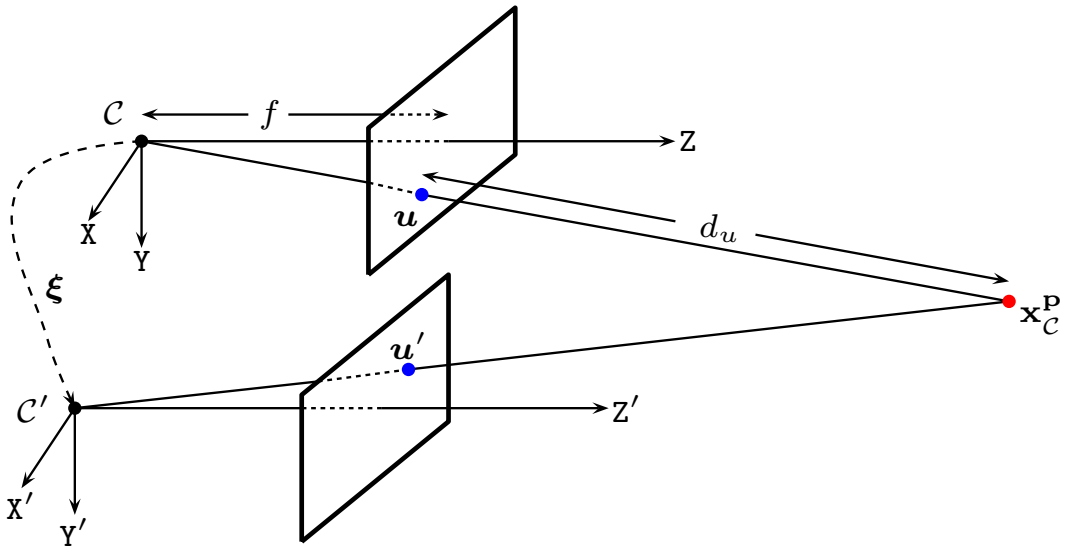


Figure 5.2: Camera movement and spatial projection of a pixel.

which can be further expanded as

$$\begin{aligned} \dot{x}_{\mathcal{C}}^{\mathbf{p}} &= -\nu_x - \omega_y z_{\mathcal{C}}^{\mathbf{p}} - \omega_z y_{\mathcal{C}}^{\mathbf{p}} \\ \dot{y}_{\mathcal{C}}^{\mathbf{p}} &= -\nu_y - \omega_z x_{\mathcal{C}}^{\mathbf{p}} - \omega_x z_{\mathcal{C}}^{\mathbf{p}} \\ \dot{z}_{\mathcal{C}}^{\mathbf{p}} &= -\nu_z - \omega_x y_{\mathcal{C}}^{\mathbf{p}} - \omega_y x_{\mathcal{C}}^{\mathbf{p}} \end{aligned} \tag{5.34}$$

Substituting equation (5.34) into equation (5.32), we get

$$\begin{aligned} \dot{u} &= -\frac{\nu_x}{z_{\mathcal{C}}^{\mathbf{p}}} + \frac{u\nu_z}{z_{\mathcal{C}}^{\mathbf{p}}} + uv\omega_x - (1+u^2)\omega_y + v\omega_z \\ \dot{v} &= -\frac{\nu_y}{z_{\mathcal{C}}^{\mathbf{p}}} + \frac{v\nu_z}{z_{\mathcal{C}}^{\mathbf{p}}} + (1+v^2)\omega_x - uv\omega_y - u\omega_z \end{aligned} \tag{5.35}$$

Rewriting above equations in matrix form, we can get the derivative of this projective wrap with respect to the pose update as

$$\frac{\partial w(\mathbf{u}, \delta\boldsymbol{\xi})}{\partial \delta\boldsymbol{\xi}} = \begin{bmatrix} -\lambda & 0 & \lambda u & uv & -(1+u^2) & v \\ 0 & -\lambda & \lambda v & 1+v^2 & -uv & -u \end{bmatrix}, \tag{5.36}$$

where $\lambda = \frac{1}{z_{\mathcal{C}}^{\mathbf{p}}}$ reflects the depth value of a point $\mathbf{u}$ in the image plane. The depth value is acquired from the scene model and can be updated while the system is running. This process is the same as stated in previous section 4.3.2.

Till here, we get all details in the computation for the derivative of a mutual

information function. The main expression is (5.24) with derivative components as equation (5.27), (5.28), (5.30) and (5.36). This measurement model can replace the one in previous chapter 4 and further fuse with inertial information in the framework of multi-state constraint Kalman filter. A full process can be written as

1. Inertial-driven propagation

   (1) Apply IMU data inputs as instant $\boldsymbol{\omega}_m$ and $\boldsymbol{a}_m$ into equation group (4.4).

   (2) Calculate $\boldsymbol{F}_d$ and $\boldsymbol{Q}_d$ according to (4.13) and (4.14) respectively.

   (3) Get propagated state covariance matrix $\boldsymbol{P}_{k+1|k}$ from (4.12).

   (4) Continue propagation until new image arrives or update triggers.

2. MI-based visual measurement model for update

   (1) Whenever new image arrives, initialise the camera pose from inertial-driven propagation and augment current state.

   (2) Find the best relative pose through maximize the MI from nearest keyframe to current image (equation 5.19) and get an observed camera pose for current image.

   (3) Calculate the individual Jacobian of MI for each pose estimation as equation (5.24), then stack all $\boldsymbol{J}$ to get the full measurement matrix $\boldsymbol{H}$ as (5.17).

   (4) Compute the Kalman gain as equation (4.25).

   (5) Update the normal state by adding the correction, and update the error state covariance as equation (4.26).

   (6) If the length of current camera state is over N, remove the oldest camera pose and perform update calculation again.

5.4 Analysis

5.4.1 A higher order iterative process for MI-based filtering

From the derivation of IEKF in previous chapter 4.5.1, we have already seen the inner connections between an iterative loop of Gauss-Newton method and the update step in the filtering process. When the estimate of a minimal update is calculated recursively in the update process, the filtering based method is actually equivalent to the optimisation based approach. Here, we derive an iterative process based on

the Levenberg-Marquardt solution, which can be potentially applied into the filtering update to increase the accuracy if being developed further.

Since the mutual information can define the similarity between two images in a known scene, a camera at optimum pose can capture a most similar image to the reference one. We also can directly estimate the full pose of the camera from an optimisation problem as

$$\boldsymbol{\xi}^* = \arg\max_{\boldsymbol{\xi}} \mathtt{MI}\left(\mathrm{I}_k, \mathrm{I}(\boldsymbol{\xi})\right). \tag{5.37}$$

It is a nonlinear optimisation, which can be solved through Levenberg-Marquardt solution as stated in previous section 3.3.4. Here we directly give the incremental update as

$$\begin{aligned} \delta\boldsymbol{\xi} &= -(\boldsymbol{J}^T\boldsymbol{J} + \beta\diagdown(\boldsymbol{J}^T\boldsymbol{J}))^{-1}\boldsymbol{J}^T\boldsymbol{\epsilon} \\ &= -\left(\mathrm{H}_J + \beta\diagdown(\mathrm{H}_J)\right)^{-1}\boldsymbol{J}^T\boldsymbol{\epsilon}, \end{aligned} \tag{5.38}$$

where $\diagdown(\cdot)$ represents the diagonal matrix, β is the damping factor, $\boldsymbol{\epsilon}$ is the residual of measurement but being changed to MI value here and $\boldsymbol{J}$ is the Jaccobian of MI. In order to perform this update, the Hessian matrix $\mathrm{H}_J = \boldsymbol{J}^T\boldsymbol{J}$ needs to be analysed further.

By applying the chain rule again in the Jacobian of MI (equation (5.24)), we can have the following expression:

$$\mathrm{H}_J = \sum_{i,j}\left(\frac{\partial p_{\mathrm{II}_k}}{\partial\boldsymbol{\delta\xi}}\right)^T\frac{\partial p_{\mathrm{II}_k}}{\partial\boldsymbol{\delta\xi}}\left(\frac{1}{p_{\mathrm{II}_k}} - \frac{1}{p_{\mathrm{I}_k}}\right) + \frac{\partial^2 p_{\mathrm{II}_k}}{\partial\delta\boldsymbol{\xi}^2}\left(1 + \log\frac{p_{\mathrm{II}_k}}{p_{\mathrm{I}_k}}\right). \tag{5.39}$$

The second-order derivative of the joint probability and its computation are similar to those of first order one,

$$\frac{\partial^2 p_{\mathrm{II}_k}}{\partial\delta\boldsymbol{\xi}^2} = \frac{1}{N}\sum_{\mathbf{u}}\frac{\partial^2\phi(i - \mathrm{I}(w(\mathbf{u}, \boldsymbol{\xi}\odot\delta\boldsymbol{\xi})))}{\partial\delta\boldsymbol{\xi}^2}\phi(j - \mathrm{I}_k(\mathbf{u})). \tag{5.40}$$

The derivative of the B-spline function is given by

$$\begin{aligned} \frac{\partial^2\phi(i - \mathrm{I}(w(\mathbf{u}, \boldsymbol{\xi}\odot\delta\boldsymbol{\xi})))}{\partial\delta\boldsymbol{\xi}^2} &= \frac{\partial}{\partial\delta\boldsymbol{\xi}}\left(-\frac{\partial\phi}{\partial\varepsilon}\frac{\partial\mathrm{I}(w(\mathbf{u}, \boldsymbol{\xi}\odot\delta\boldsymbol{\xi}))}{\partial\delta\boldsymbol{\xi}}\right) \\ &= \frac{\partial^2\phi}{\partial\varepsilon^2}\left(\frac{\partial\mathrm{I}}{\partial\delta\boldsymbol{\xi}}\right)^T\frac{\partial\mathrm{I}}{\partial\delta\boldsymbol{\xi}} - \frac{\partial\phi}{\partial\varepsilon}\frac{\partial^2\mathrm{I}}{\partial\delta\boldsymbol{\xi}^2}. \end{aligned} \tag{5.41}$$

Among above equation,

$$\begin{aligned}\frac{\partial^2 \mathrm{I}}{\partial \delta \boldsymbol{\xi}^2} &= \frac{\partial}{\partial \delta \boldsymbol{\xi}} \left(\nabla \mathrm{I}(w(\mathbf{u}, \boldsymbol{\xi})) \frac{\partial w(\mathbf{u}, \delta \boldsymbol{\xi})}{\partial \delta \boldsymbol{\xi}} \right) \\ &= \left(\frac{\partial w}{\partial \delta \boldsymbol{\xi}} \right)^T \nabla^2 \mathrm{I}\left(\mathrm{u}'\right) \frac{\partial w}{\partial \delta \boldsymbol{\xi}} + \nabla_u \mathtt{I}(\mathrm{u}') \frac{\partial^2 w_u}{\partial \delta \boldsymbol{\xi}^2} + \nabla_v \mathtt{I}(\mathrm{u}') \frac{\partial^2 w_v}{\partial \delta \boldsymbol{\xi}^2}\end{aligned} \tag{5.42}$$

where the $\nabla^2 \mathrm{I}(\mathbf{u}')$ is the Hessian of $\mathrm{I}(\mathbf{u}')$ with respect to u, v axes and $\frac{\partial^2 w}{\partial \delta \boldsymbol{\xi}^2}$ is the Hessian of wrap function $w(\cdot)$. Further with the derivative of projective wrap in section 5.3.3. The Hessian of the wrap in u and v direction can be expressed as

$$\frac{\partial^2 w_u}{\partial \delta \boldsymbol{\xi}^2} = \begin{bmatrix} 0 & 0 & -\lambda^2 & 0 & 2\lambda u & -\lambda v \\ 0 & 0 & 0 & -\lambda & 0 & -\lambda u \\ -\lambda^2 & 0 & 2\lambda^2 u & \lambda v & -2\lambda u^2 & 2\lambda uv \\ 0 & -\lambda & \lambda v & -u & -2uv & v^2 - u^2 \\ 2\lambda u & 0 & -2\lambda u^2 & -2uv & 2(1+u^2)u & -(1+2u^2)v \\ -\lambda v & -\lambda u & 2\lambda uv & v^2 - u^2 & -(1+2u^2)v & (2v^2+1)u \end{bmatrix}, \tag{5.43}$$

and

$$\frac{\partial^2 w_u}{\partial \delta \boldsymbol{\xi}^2} = \begin{bmatrix} 0 & 0 & 0 & \lambda & \lambda v & 0 \\ 0 & 0 & -\lambda^2 & 0 & \lambda u & -2\lambda v \\ 0 & -\lambda^2 & 2\lambda^2 v & -\lambda u & -2\lambda uv & 2\lambda v^2 \\ \lambda & 0 & -\lambda u & -v & u^2 - v^2 & -2uv \\ \lambda v & \lambda u & -2\lambda uv & u^2 - v^2 & (1+2u^2)v & -2uv^2 \\ 0 & -2\lambda v & 2\lambda v^2 & -uv & -uv^2 & 2(1+v^2)v \end{bmatrix}. \tag{5.44}$$

As we can see from above equations, the Hessian matrices of projective wrap function depend on the position of the pixel and depth estimation. However, during the iterative optimisation, the point and depth value will be altered, thus bringing in huge computational burden. To improve the convergence and efficiency of the optimisation, we can use the fact that the consecutive images only exist when translation is insignificant. Therefore, the initial pose estimation from the inertial-driven dynamic model can be adopted as the optimum pose at the beginning. Then the Jacobian and Hessian matrices are kept constant during the iteration thus saving the computational resources. When the algorithm becomes converging, both of the matrices will be updated again.

5.4.2 Smoothing the MI funtion

From the definition and analysis of the entropy in the previous section we can see that, if the probability of a random variable is small, the value of entropy becomes large. This usually happens when an insignificant image translation occurs. For an extreme example, when a region of interest is located near the boundary of an image, even a small translation of this region will import new pixel intensity into the original distribution. Then several bins of histograms will be increased from null to a small value as well as decreasing other original bins, thus making such insignificance become valuable information under the concept of entropy. However, these new pixel intensities are not the actual camera transformation but noise. Thus they need to be removed and the MI function needs to be smoothed.

A most simple smoothing method is to apply a Gaussian filter template in the input images before further comparison. This filter model can refine the intensity of a pixel using its neighbours. In the above example of noise interference, their intensity values will be smoothed without affecting the shape of original entropy. However, the operation will also weaken the large area of valuable information in other image regions.

In our method, as shown in the derivation process, the B-spline function is adopted. The merit of using B-splines interpolation comes from its simple expression in the calculation, and its derivative is clearly defined. The B-spline solution is actually an approximation to Gaussian function [148, 178]. The higher the order used, the smoother the curve can be achieved. However, we not can be neglected the fact that, the complexity and computational burden will increase when the high order of B-spline is chosen.

5.5 Experiments and Results

5.5.1 Advantages in MI-based similarity function

As stated previously, the MI-based method enjoys the merits of better robustness in the situation of noise pollution, illumination variation and partial occlusion. Here, we compare the MI-based similarity measurement method with other two representative error measures, i.e. Sum of the Squared Difference (SSD) and Zero mean Normalized Cross Correlation (ZNCC).

The SSD based on the Euclidean norm is the most classical error measurement method popularly applied in alignment problems, such as KLT [53], PTAM [95].

Here, we use it to measure the distance of pixel intensities directly, which can be expressed as

$$\texttt{SSD}(\mathrm{I}, \mathrm{I}_k) = \sum ||\mathrm{I}(\mathbf{u}) - \mathrm{I}_k(\mathbf{u}')||, \tag{5.45}$$

where $\mathbf{u}$ and $\mathbf{u}'$ are the corresponding pixels in two images and $\mathrm{I}(\cdot)$ extract the photometric information as intensity value from the image matrix.

The ZNCC makes use of the correlation coefficients to quantify the similarity of two images, which can be expressed as

$$\texttt{ZNCC}(\mathrm{I}, \mathrm{I}_k) = \frac{\sum \left(\mathrm{I}(\mathbf{u}) - \bar{\mathrm{I}}\right) \left(\mathrm{I}_k(\mathbf{u}') - \bar{\mathrm{I}}_k\right)}{\sigma_{\mathrm{I}} \sigma_{\mathrm{I}_k}}, \tag{5.46}$$

where $\bar{\mathrm{I}}$ and $\bar{\mathrm{I}}_k$ are the average intensities of the images I and I_k respectively, while σ_{I} and σ_{I_k} are the standard deviations of these two images.

A patch in the size of 200×200 is extracted from a particular pose facing to an image of our lab scene, as shown in figure 5.3. Then the translation of ± 50 along u and v directions are performed from the original chosen position. At every position, the three error measurement functions are calculated, then saved and plotted as a 3D mesh to illustrate their performance in various image conditions.

As a basis for comparison, we firstly evaluate the considered patch of the same as the reference one, as shown in figure 5.4. The results reflect the variation of the error measurement for camera translation. All of them showing the Optima at the centre point of $(u = 0, v = 0)$, which fits the desired outcome. The shapes of these error measurement are smoothing without local minima or maxima, which indicates their ability to measure the difference while the translation happens. However, in the raw lab indoor scene, the minima point in SSD and ZNCC measurement methods seem not as significant as the optimum point in the MI-based method.

5.5.1.1 Illumination variations

The light condition can be altered when a mobile platform travels in a scene. Even for the same environment, different seasons of one year or different time of one day for outdoor, and different rooms with various lights for indoor will lead to different light conditions. As mentioned before, the light condition has an enormous impact on the visual-based system. To evaluate the performance of MI-based method in such situation, we set two illumination cases in our lab scene. In case one, we turn off one of the four lights at the ceiling while turning off two in case two. Sample patches in these two cases are shown in figure 5.5. With the same searching and

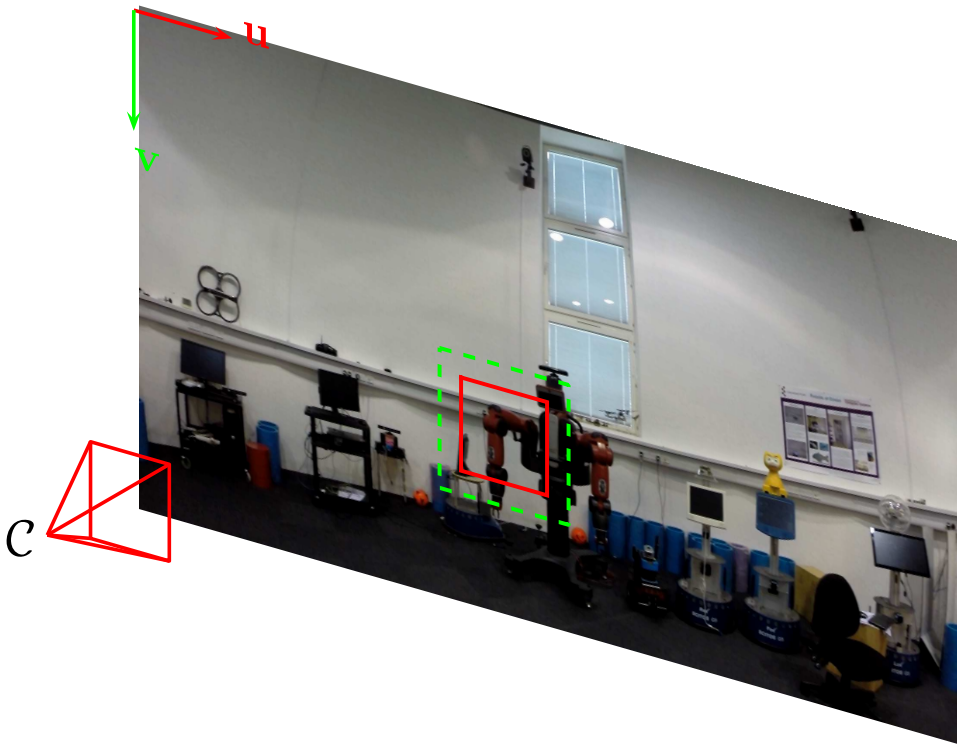


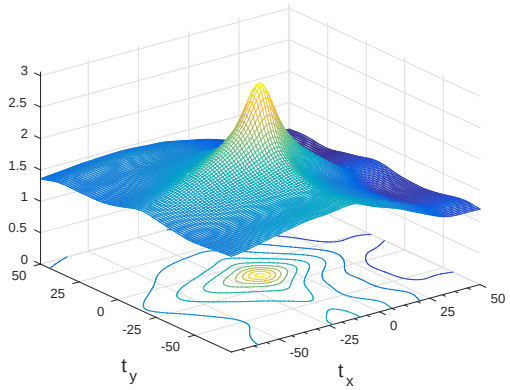
Figure 5.3: Patches are selected in our lab scene. The translations with ±50 pixels are performed along u and v, the green dashed line marks the boundary.

computing method, we get the resulting values as the meshes of three measurement methods for two cases. As we can see in figure 5.6, it shows no minimum in SSD method and the variation in ZNCC becomes week with a near flat region when the scene is darker. However, the MI-based measurements can maintain the shape for a significant optimum point, even when the light condition becomes worse.

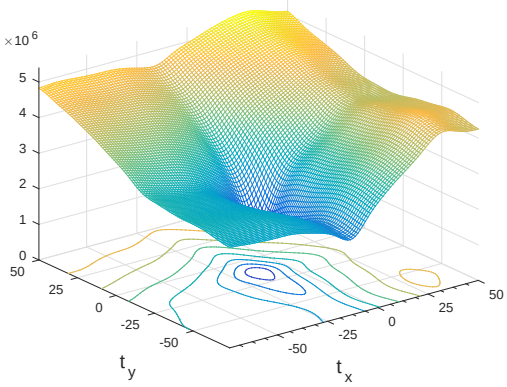
5.5.1.2 Occlusions

It is ideal to have a static scene while operating the visual-based system, but this is always hard to guarantee. Random obstacles may occur at any moment even when a mobile platform travels along a known path. The size of obstacles has direct impacts on the quality of sequential images, thus causing problems on the calculation of visual measurements. To evaluate the variation of such measurement functions concerning occlusion situation, we set two cases to block the scene. As the sample patches are shown in figure 5.7, three circular areas in different grey levels block the robot in the scene and the case two is worse than case one. Results on figure 5.8 show that three of the measurement functions are affected by the blocks, and such

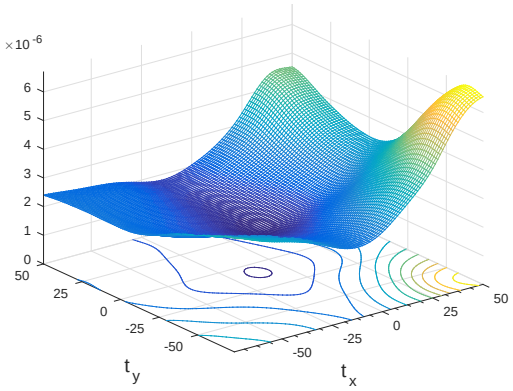
(a) Ref. patch



(b) MI



(c) SSD



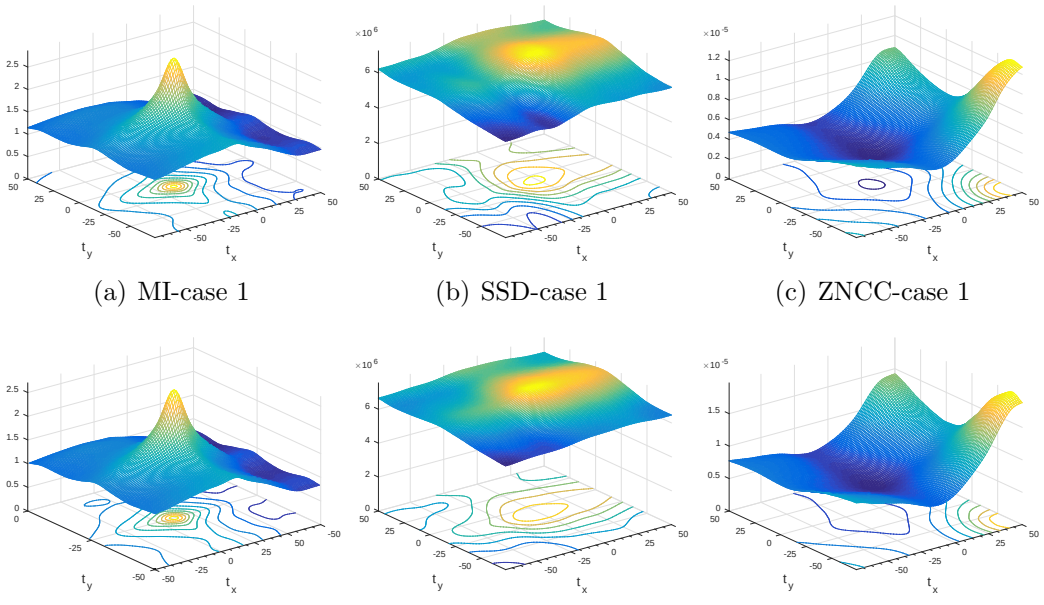
(d) ZNCC

Figure 5.4: Evaluation for MI, SSD and ZNCC based measurement functions when translations performed in raw patches.

situation gets worse when the covered areas become larger. The SSD changes into a slope gradually while the pit or peak in ZNCC and MI become slow with a large flat area. However, from the contour curves, we still can find out that, the MI in peak area is more prominent than the pit in ZNCC.

(a) Illumination case 1 (b) Illumination case 2

Figure 5.5: Sample patches in illumination variation cases.



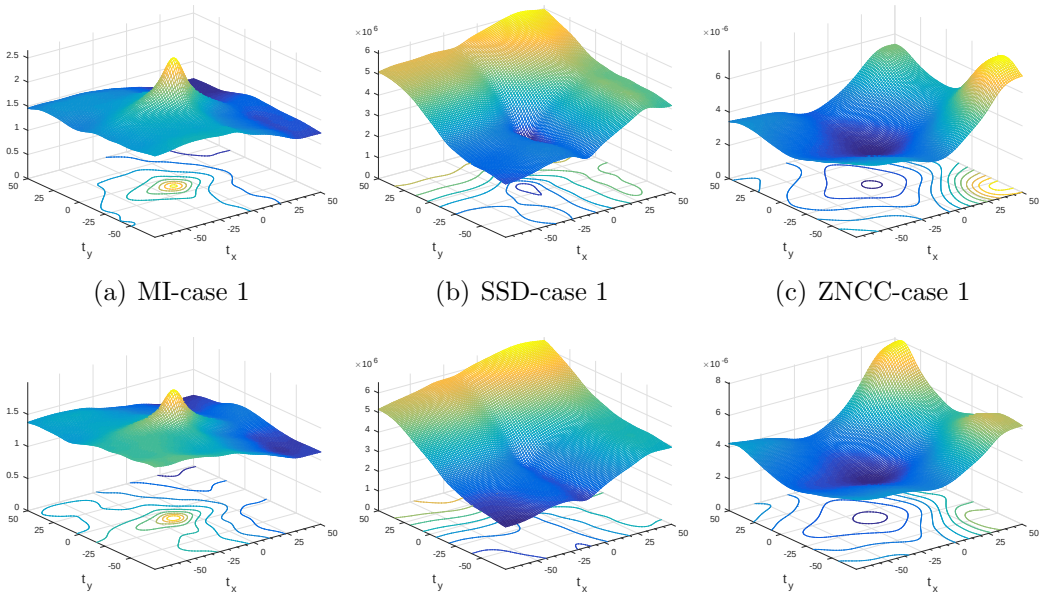
(a) MI-case 1 (b) SSD-case 1 (c) ZNCC-case 1

(d) MI-case 2 (e) SSD-case 2 (f) ZNCC-case 2

Figure 5.6: Evaluation for MI, SSD and ZNCC based measurement functions when translations performed in illumination variation cases.

(a) Occlusion case 1 (b) Occlusion case 2

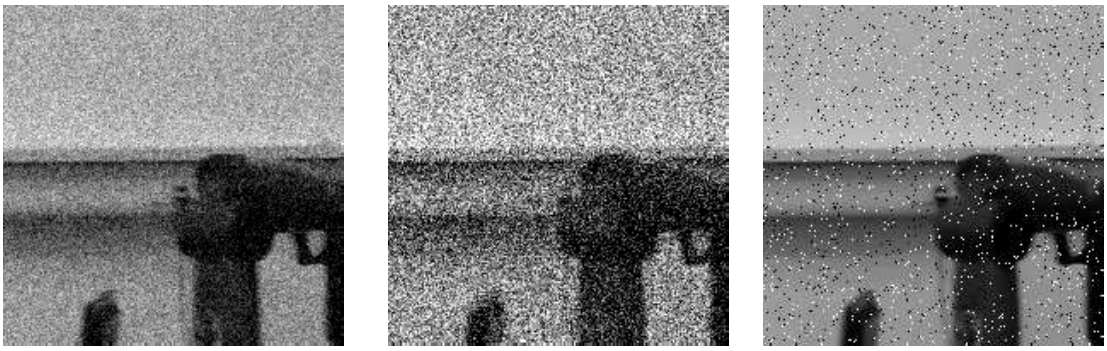
Figure 5.7: Sample patches in occlusion environment.



(a) MI-case 1 (b) SSD-case 1 (c) ZNCC-case 1

(d) MI-case 2 (e) SSD-case 2 (f) ZNCC-case 2

Figure 5.8: Evaluation for MI, SSD and ZNCC based measurement functions when translations performed in partial occlusion cases.



(a) Gaussian noise (0, 0.01) (b) Gaussian noise (0, 0.05) (c) Salt & pepper noise

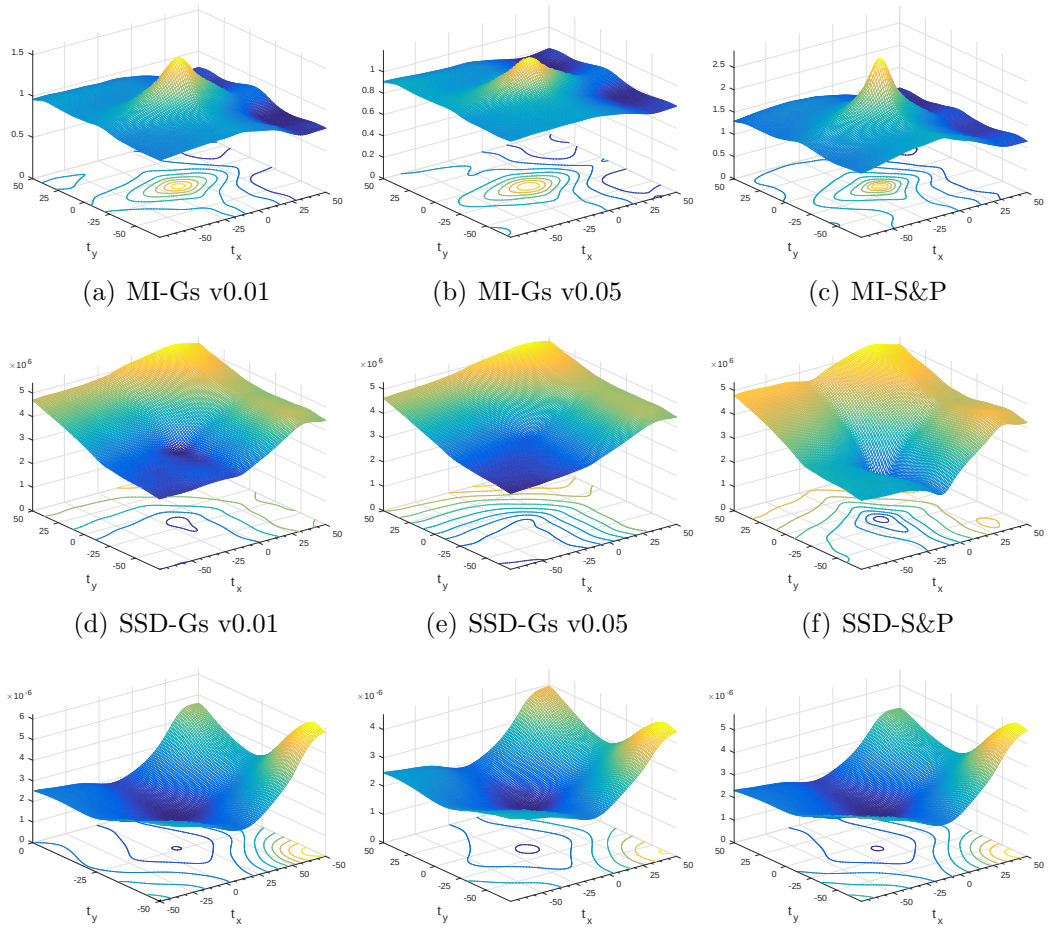
Figure 5.9: Sample patches with different noise.

5.5.1.3 Noise

Image noise commonly exists in any visual-based processing. It can be led in by the quality of physical sensors or the error of software computing. When the noise is accumulated, the estimation results tend to be divergent. This situation is apparent in IMU only motion estimation, which has been shown in the previous chapter. For the visual case, to evaluate the variation of visual measurement for the image noise, we import three types of noise, including Gaussian noise with zero mean but different variance and salt & pepper noise. The sample patches are shown in figure 5.9. As we can see from the resulting figures 5.10, in general, the MI and ZNCC are robust to all noise types with identical minima or maxima, but SSD has been largely impacted by Gaussian noise. Particularly, the Gaussian noise with large variance brings in more uncertainty thus leading a less pronounced result for measurement function. The situation of salt & pepper noise can be regarded as small scattered point blocks. Thus the shapes of measurement functions are similar to those with occlusion. In practice, the Gaussian noise will not be as bad as the one with Gaussian parameters as (0, 0.05) in figure 5.9(b). Therefore, MI is the preferred measurement function for any mild noise pollution case.

5.5.2 Robustness compared with feature-based matching

To access the ability of MI-based method in finding correspondences when compared to the feature-based method, we use two consecutive images from an outdoor driving scene. The five most corresponding patch pairs from the first and second image are firstly found based on the best matching feature points of the up-left corner of interest patches, which are marked as red and blue, respectively, in figure 5.12.



(a) MI-Gs v0.01 (b) MI-Gs v0.05 (c) MI-S&P

(d) SSD-Gs v0.01 (e) SSD-Gs v0.05 (f) SSD-S&P

(g) ZNCC-Gs v0.01 (h) ZNCC-Gs v0.05 (i) ZNCC-S&P

Figure 5.10: Evaluation for MI, SSD and ZNCC based measurement functions when translations performed in different noise type.

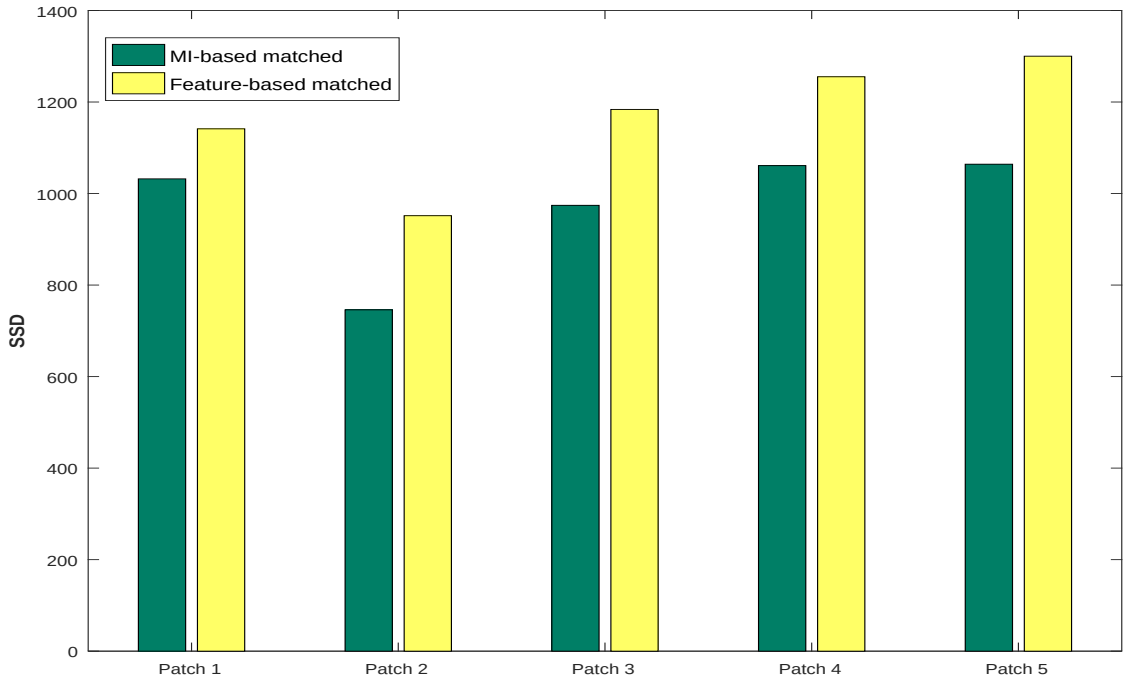


Figure 5.11: SSD comparison for correspondence given by feature-based and MI-based methods.

The patches from the first image are then used as references for the best similarity searching with a maximum MI value. As we can see from the last row of figure 5.12, the peaks occur while performing full image searching. The best corresponding patches are marked as green. If we limit the searching range in ± 20 pixels based on the matched corner feature points as prior (marked as yellow), the local MI values can be given as the third row of figure 5.12. It shows that all corresponding patches can be found through maximum MI globally or locally and the maximum is significant for most patches. However, there exists less significant situation if the searching is performed in a textureless scene like the second patch. And the ambiguity peaks in the fifth patch will also lead to a failure matching for maximum MI if no further validation process is performed.

We use SSD to compare the corresponding patches from feature-based and MI-based methods. The result is shown in figure 5.11. The patch correspondences given from MI are more similar to the reference patches from the first images. It indicates that the application of MI-based method outperforms feature-based in finding corresponding patches in the outdoor environment. But it also should be noticed that more validation steps are necessary if the MI measurements are applied in a textureless scene or in the cases with multiple maximum situation.

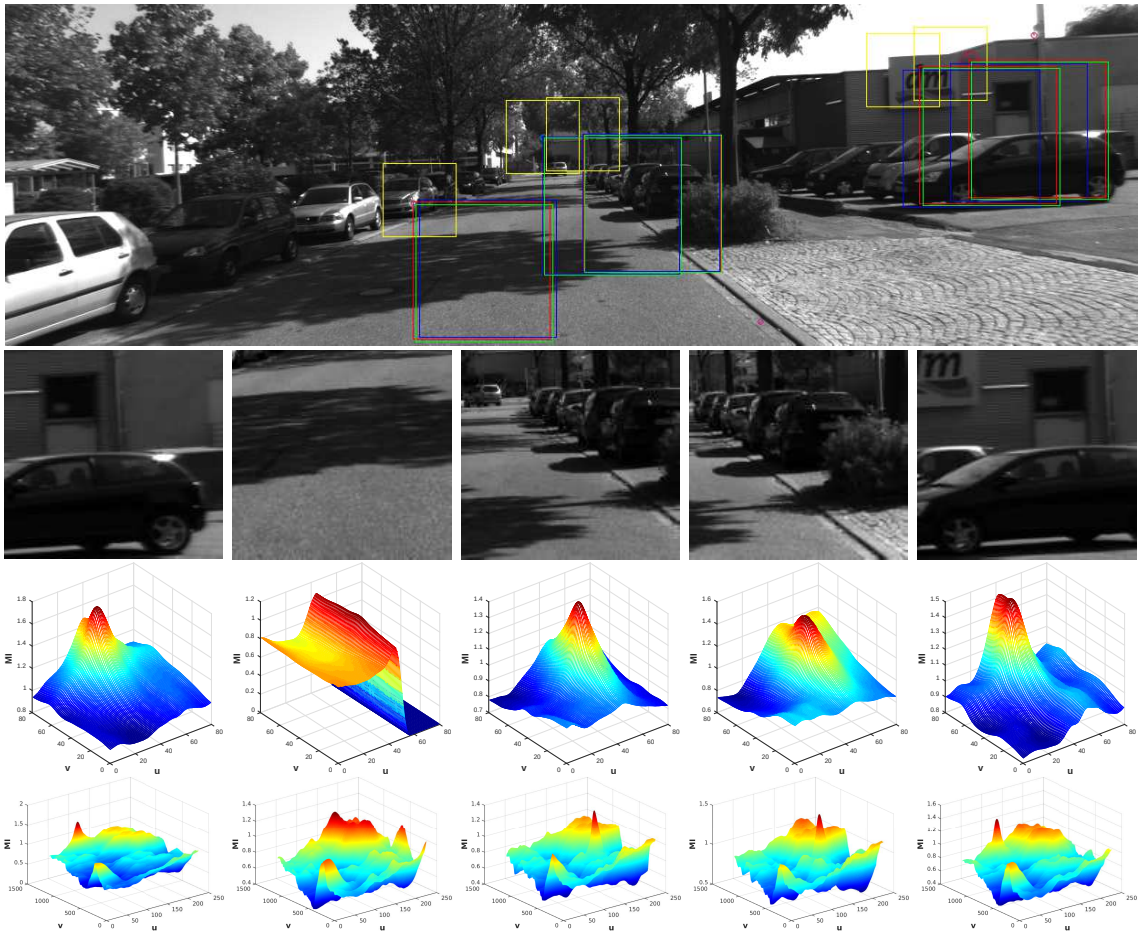
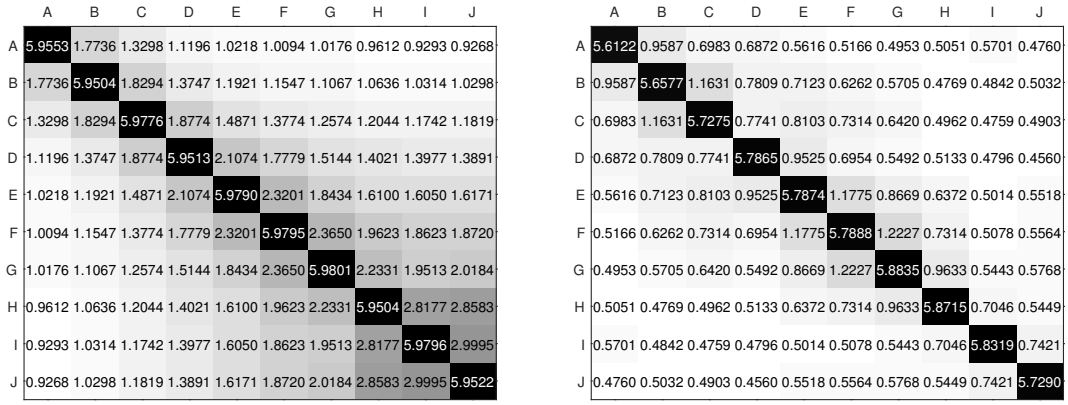


Figure 5.12: Five patches are found through maximum MI value in global and local image regions. In the scene picture, the red square indicates the patch from the first of consecutive image pairs, the blue marks corresponding patches given from feature-based matching, the green ones are the correspondence from maximum MI and the yellow bound local searching range. The reference patches and their MI values while searching locally or globally are shown in the second, third and fourth rows, respectively.



(*a*) (*b*)

Figure 5.13: Mutual information for two different trials. (a) 10 frames are extracted from original sequence in every two frames; (b) 10 frames are extracted in every ten frames. The value is the mutual information between the corresponding frames from A to J.

Furthermore, to give a quantitative understanding of MI, we form two image sequences, each including ten frames (noted as A to J) from the above outdoor environments. These image sequences are down-sampled from the original sequence at 1/2 and 1/10 of the original rate. In figure 5.13, we can obviously see that the values of MI are around 2 between every other two frames (the adjacent diagonal starting from B in figure 5.13(a)), while the values are around 1 between every ten frames (the adjacent diagonal starting from F in figure 5.13(a) and B in Figure 5.13(b)). Significantly, the values drop from average 5.8 to 2 dramatically between adjacent frames but slightly between distant frames. Therefore, if MI-based measurements are applied, the threshold for distinguishing similarity in consecutive frames should be easily found.

5.5.3 Trajectory estimation

Following the experiments from the previous chapter using the public dataset of KITTI, we choose the case 0009 with a straight line followed by a near perpendicular turning, and the case 0113 with turning from slow to sharp, to evaluate the performance of MI-based visual measurement model. As we can see the statistical estimation results of case 0009 from figure 5.14 and 5.16(a), when travel along a straight line, the MI-based method can tightly follow the trajectory given by the method in chapter 4. But when a rotation is performed, massive error accumulation occurs after a delay. Furthermore for the case 0113, if turning is performed

gradually at the beginning, the accumulated error becomes significant for our previous MSCKF-direct method, as the results indicated in chapter 4. However, the estimation from the MI-based method can follow the trend until a sharp turning is performed. The RMSE curves for both cases are illustrated in figure 5.17.

The results seem not satisfying but can be predicted, because our MI-based method is based on an estimated scene depth in scattered edge points from the MSCKF-direct method. There already exists accumulated error in depth estimation, and it will become worse when further adopted as the reference for pose estimation. However, the MI-based pose estimation method shows an excellent tracking behaviour in the straight trajectory and the one with a slight tolerance of rotation.

5.5.4 Potentials in mapping

Different from the concept of mapping in previous chapters, which indicates sparse point cloud with depth value in world frame, the map here is a mosaicing image stitched by a series of adjacent pictures. Similar to the way using maximum MI values to find best matches, the consistent picture overlays the most similar parts of consecutive images together through planar transformation. Although there are various image mosaicing methods available in literature, such as feature-based [10, 179, 180] or direct [181, 182] methods, the core of them is similar including:

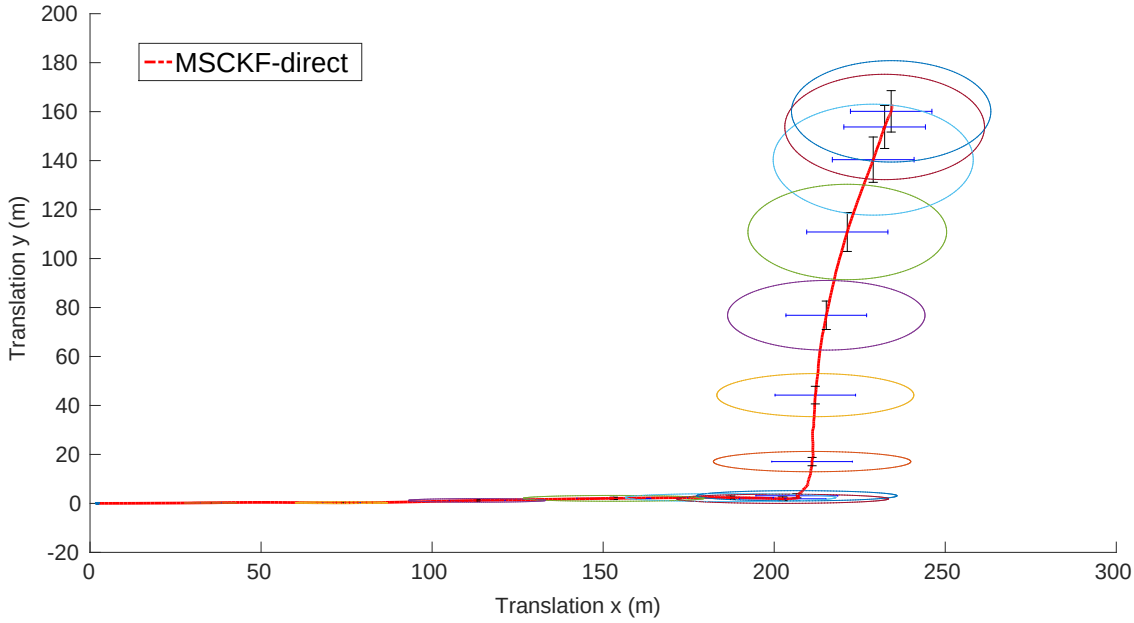
(1) Select the sequential images of a scene with at least a quarter (by empirical trials) of the overlapping area.

(2) Find the corresponding patches and compute the homography matrix as illustrated in section 3.1.4.3.

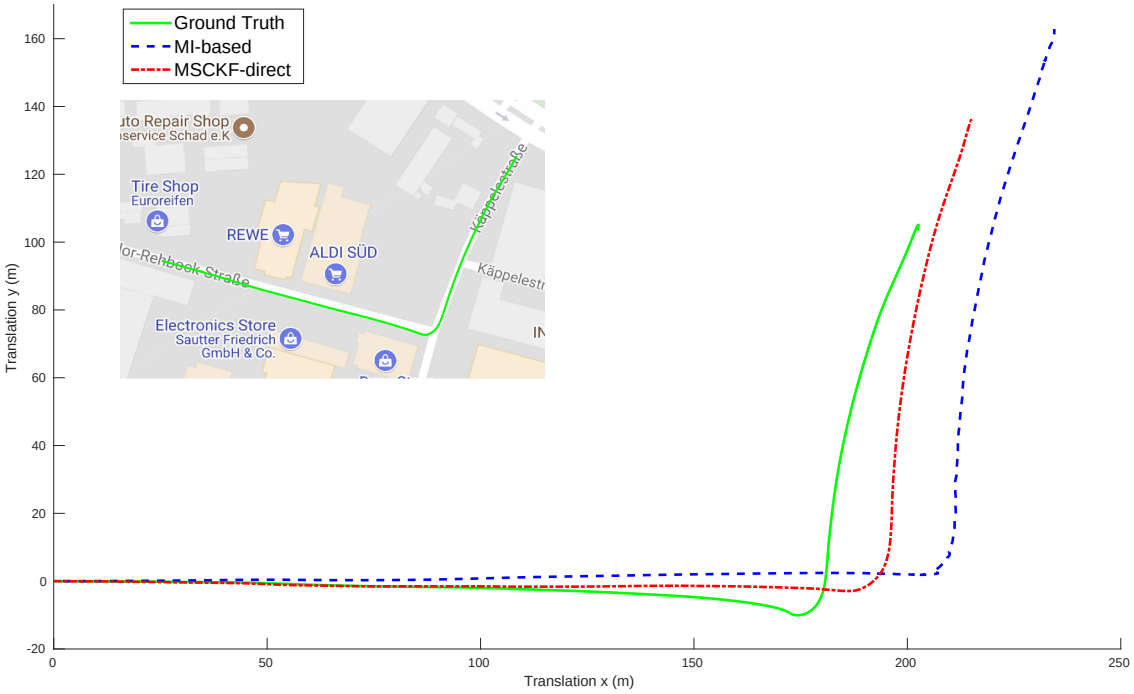
(3) Perform the transformation to the current image and overlay it with previous one.

(4) Repeat the process until all images are stitched.

As mapping is not the primary task for an odometry task, here we just illustrate the potentials of using MI-based method for building a consistent scene image. A resulting mosaicing image created from five images of our lab is shown in figure 5.18, where the robots, cable case along the wall and the window can be depicted clearly. But there exists ambiguity in similar texture regions like the window, which leads to blurring fusion in the image. This also provide us an evidence that MI-based method should be carefully processed when meeting repetitive texture scenarios.



(a) Statistic analysis of multiple trials using the MSCKF-direct method. The error in both x and y directions are accumulated over the mean trajectory, while the 0.95 confidence ellipse are expanded as algorithm goes on.



(b) The estimated mean trajectories of multiple trails and the IMU-only/GPS recorded trajectories.

Figure 5.14: Estimated trajectories for the case 0009 in the KITTI dataset.

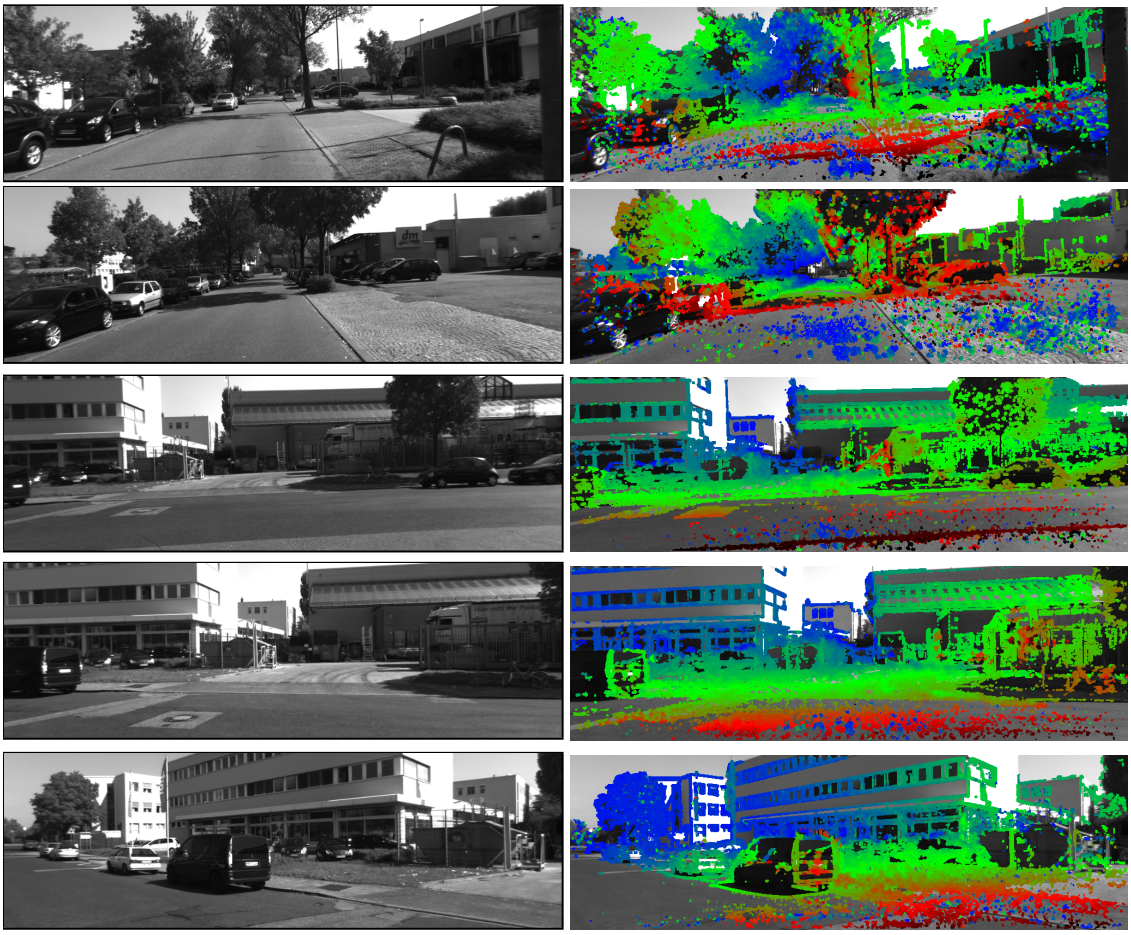
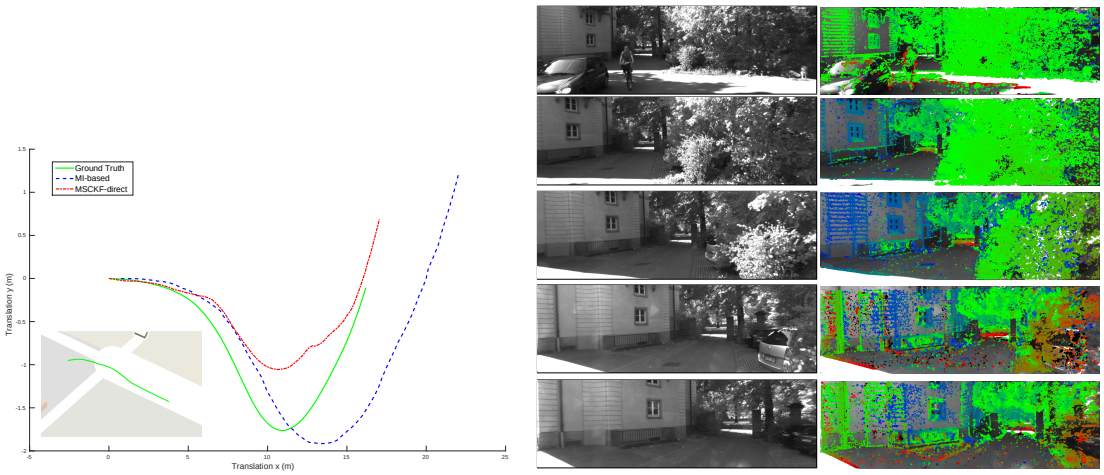


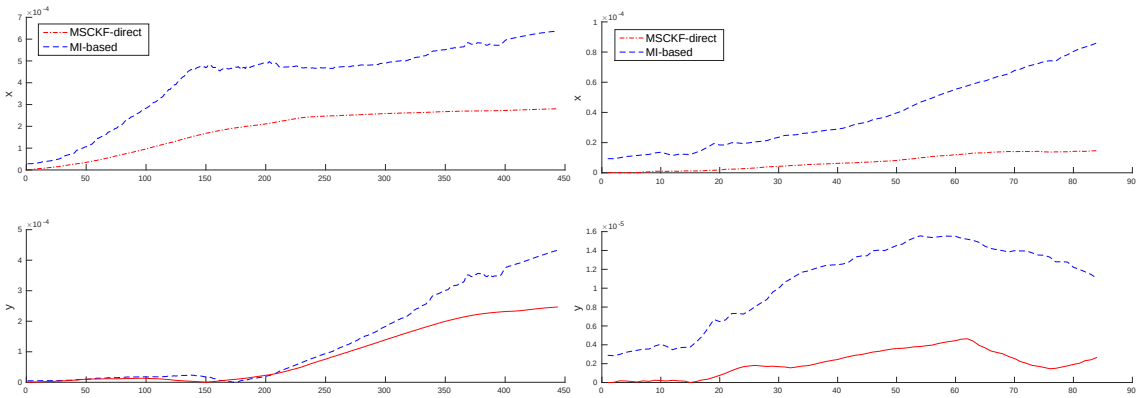
Figure 5.15: Some scene pictures and corresponding keyframes captured from the case 0009 during algorithm running.



(a) Estimated trajectory.

(b) Selected keyframes.

Figure 5.16: Trials for the case 0113 in the KITTI dataset.



(a) RMSE of 009 (b) RMSE of 113

Figure 5.17: RMSE of the case 0009 and 0113 of the KITTI dataset.

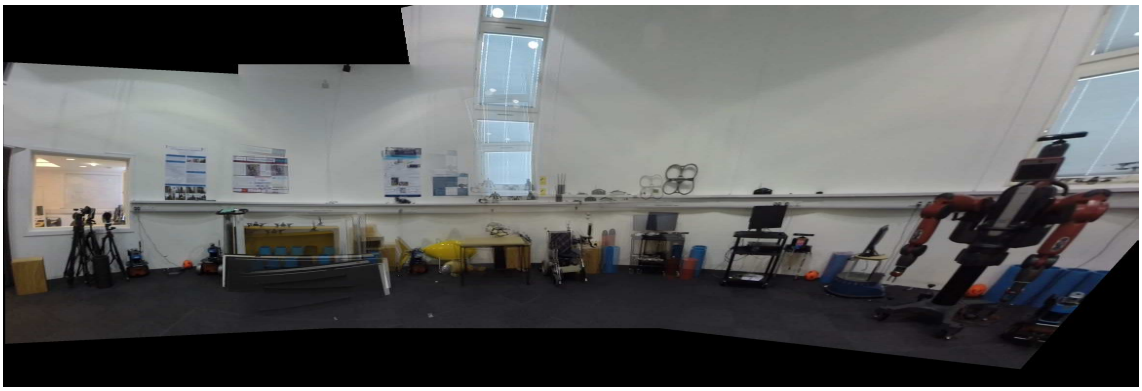


Figure 5.18: MI-based image mosaicing map for our lab scene.

5.6 Summary

In this chapter, we present the MI theory from the core concept of information theory to the detail process of its partial derivative, where the ability of MI outperforming joint entropy for image comparison is discussed. This theoretical basis provides a basic guide to building an MI-based measurement model, which can be used in the filtering framework. Furthermore, from the detailed derivation of the Jacobian matrix of MI measurement function, the Hessian matrix of MI is also presented to fit for an iterative process in the filtering framework. This is a promising method if being further developed according to the IEKF analysed in the previous chapter. The iterative process can be used to enhance the accuracy of estimation results.

For the experiments, we strive to showcase the advantages and limitation of MI-based visual measurements. Firstly, two representative error measures are used to be compared with the MI-based method in the case of illumination variation, occlusion and noise interference in different levels. Results show that the MI-based method outperforms other measures in all these situations and maintains a significant convergent region for translation, but the MI-based method still can be impacted if large occlusion or noise pollutes the images. When compared to the feature-based method in patches searching, the MI-based method shows the advantage in finding the best-fit patches in sequential images, but when applied in the case with repetitive and less texture scene, it will show a less significant result. There also exist multiple peaks when a patch is searching within the whole image. Thus the additional process for evaluating the best fit is still necessary.

When applied to the trajectory estimation in the VIO framework, the MI-based method needs the depth reference beforehand to find the best pose observation, thus leading to an increasing accumulated error in estimation. The situation is getting worse when applied in the rotational case. However, we still can see a good trajectory following behaviour when the travelling is along straight lines and with a slight tolerance of rotation.

Additionally, a mosaic image of our lab scene is built based on the MI-based image mosaicing technique. Although it is not the focus of our research, this experiment showcases the ability of MI-based measurement in mapping. And the situation of regional mismatching reminds us to avoid less or repetitive texture scene when applying the MI-based methods.

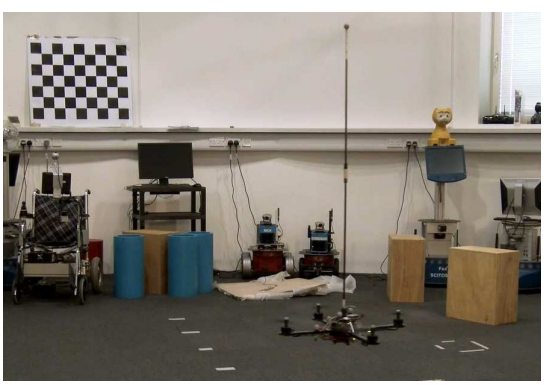
Chapter 6

Mobile Platform for Data Collection

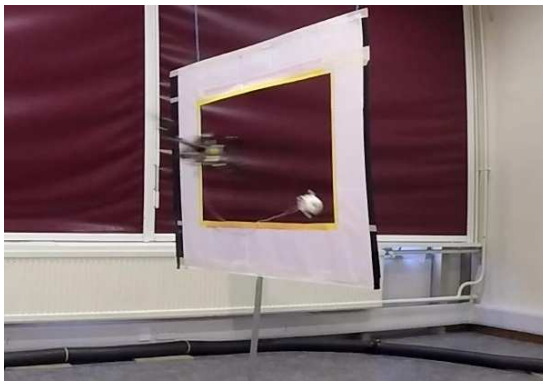
6.1 Overview

Mobile platforms with the powerful sensing and computing capacity have triggered an overwhelming interest in not only the industrial market but also academic research. Similar to a general concept of mobile robots, a mobile platform commonly refers to a physically small device that can enjoy enough freedom in movement. Quadcopters are a representation among them. Different from traditional flying robots, such as the fixed wing aircraft or helicopters, a quadcopter has four rotors, which are placed away from the centre of platform symmetrically. This arrangement makes each propeller can be controlled by individual motor. A simple motion variation needs to coordinate the outputs from four motors. Even for a simple hovering behaviour, the complex tasks of balancing the different outputs of each motor or rapidly adjusting disturbance dynamically need to be taken into account. Quadcopter flying platforms are inherently unstable, thus making them heavily rely on electronics and sensor systems. Fortunately, its structure arrangement makes the minimisation of a quadcopter platform becomes possible. There exists large empty space for placing electronic systems at the middle, where the gravity centre of the whole platform can be maintained.

In our lab, we have already conducted some excellent control tests of quadcopter. Examples can be seen in figure 6.1, which includes the ability to fly through a window with a freely hanging payloads and the case of autonomously balancing an inverted pendulum on the top of quadcopter. However, due to the limits of weight capacity



(a) Balance an inverted pendulum on the top of quadcopter.



(b) Fly through a gap with a freely hanging payload.

Figure 6.1: Examples of quadcopter control demonstrated by our robotics lab.

and onboard computation resources, such tasks still need to rely on external devices, such as the external infrared motion tracking camera system (VICON) in our lab. Although there are various excellent SLAM algorithms which are claimed to have a real-time performance when running in desktop computer, it is still challenging to let such localisation or navigation algorithms run autonomously onboard. To lessen the processing burden, some techniques have already been applied in practice, including separating major calculation tasks to a ground-station or carefully setting the scene structure and motion planning information in memory beforehand.

However, we not can achieve fully onboard autonomous for quadcopter just through any single technique in hardware or software. It is a systematic project that should combine both hard electronic devices and soft programs. The software part is mainly focusing on the problem regarding control and communication. At this stage, we focus only on building an integrated quadcopter system equipped with the basic visual and inertial sensor set. The expected quadcopter can perform manoeuvres through a remote control panel and store the visual-inertial data while flying. Rather than using commercial products, we build the quadcopter by individual components. All parts, including the hard frame of the quadcopter, electronic speed controller, flying control board, a power supplying system, high-level computing board and necessary sensors are carefully chosen according to the limitation of payload and the requirements of data collection.

In this chapter, we will introduce the major parts of our quadcopter and related development environment, which include the selection between the different monocular camera and inertial sensors, the properties of onboard systems and potential

development tools in ROS and Matlab. Finally, the full integrated quadcopter will be presented.

6.2 Major Components

A basic quadcopter system should include a body frame with four arms, four motors, two pairs of propellers, four electronic speed controllers, a flight control board, a remote controller and a power supply. If more functions are expected, extra components like high-level computing board or various sensors should also be included. For our goal of building a quadcopter with the ability of recording visual and inertial data, the separate visual and inertial sensors with a suitable computing board are required.

Although there are a plenty of tutorials teaching hobbyists how to build a quadcopter step by step, none of them give a full list that meets our requirement. In this section, we present some principal components adopted in our quadcopter, including visual and inertial sensors for data recording, flight control board and high-level computing board. Other parts can be found according to some general developing guides on the Internet[1], but some empirical developing procedures can be concluded firstly as follows:

1. Before buying any components, one can follow:

   (1) Propose a target system with particular functions.

   (2) Build a full requirement list regarding all parts with necessary accessories.

   (3) Select every component available on the current market with a detail specification.

   (4) Recheck the compatibility of all parts including the interface in software and hardware.

   (5) Buy all the components.

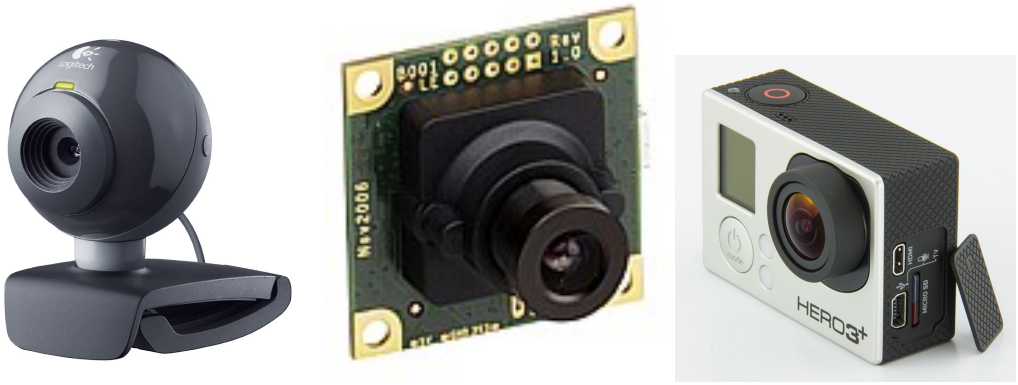
2. Testing should be performed from the separate to the complete sets, and before any power-on testing, one must consider:

   (1) Whether the cable connection is correct as the requirement.

   (2) What is expected to occur if power is on.

   (3) Are the operation procedures clear after power on.

[1]For more details, please refer to myfisrtdrone.com or mydronelab.com etc.



(a) Logitech (b) IDS UI-1221LE-C-HQ (c) Gopro Hero3$^+$

Figure 6.2: Cameras

(4) What to do if an unexpected situation happens.

(5) When to exit the system if we have already seen what we expected.

6.2.1 Monocular camera

There are overwhelming amounts of camera types available on the market. Their quality and price vary according to the different application background and performance specifications. For our platform, since the physical limitation by the quadcopter, the ones only with properties of small size and light weight are taken into account. Additionally, it should be accessible for developing with high compatibility for both software and hardware. We find three representative types here, i.e. web camera, camera board, commercial motion capture camera, as shown in Figure 6.2.

The web cameras are widely used in the network applications and can be connected by the USB port. To lessen the memory burden and communication latency, the image resolution is usually limited up to HD (1280×720) with the frame rate of 30Hz. However, due to highly integrated of the electronic components, the field of view of web camera is often limited by the fixed focal length and the embedded image sensor. In a web camera, image stream can be acquired by performing consecutively recording. But the capture method of rolling shutter[1] limits the motion of such camera. Figure 6.3 shows the impacts of rolling shutter when taking a picture of a fan. For application in motion platforms which are usually under highly flexible

[1]Rolling shutter is a method of image capture in which a still picture is captured not by taking a snapshot of the entire scene at the single instant time but by scanning across the scene rapidly, either vertically or horizontally. In other words, not all parts of the image of the scene are recorded at the same instant.

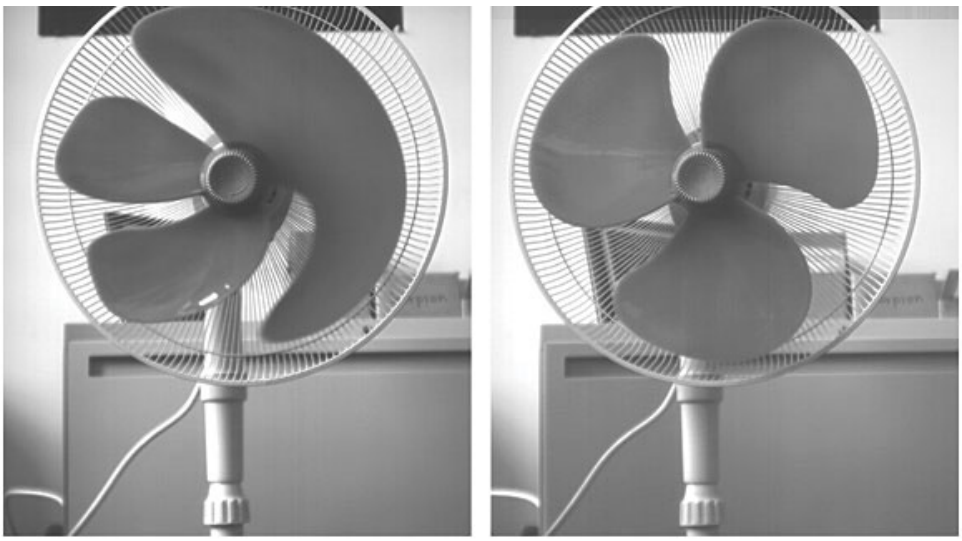


Figure 6.3: A fan is captured by the rolling and global shutter.

movement, this effect can happen easily.

The motion capture camera is popularly used in recording sports and life footages. There are various commercial products in the market, while the Gopro is the most well-known one. Take the *Gopro hero 3 plus silver edition* as an example, its recording mode varies from 848×480 with 60 fps to 1920×1080 with 60 fps, while the medium setting at 1280×720 with 120 fps is the highest resolution with the fastest capture rate. However, due to this product being highly integrated, it is hard to read and use the recording images live. Although we have tried to get a live image stream through a set of radio transmission devices, as shown in figure 6.4, the huge image noise, significant latency and limitation of radio bandwidth still make the received images impossible for further processing. Additionally, to enlarge the field of view, the fish eye lens are usually adopted for motion capture cameras, thus leading to the distortion effect in resulting images. Therefore, it is not the best choice of our mobile platform.

The camera boards are very suitable for developing due to their small size and various supporting source. Here, we use the product of IDS UI-1221LE-C-HQ, the resolution is 752×480 with $1/3''$ CMOS color and its frame rate is up to 87 fps. The image stream can be delivered through the USB connection with low latency. The key property of this camera type is using the global shutter mode to capture an entire image at the same instant time, thus avoiding the case of motion distortion as shown in figure 6.3. Additionally, this camera board is of 36×36 mm and only

Figure 6.4: Live image stream transmission by a set of radio devices for Gopro Hero3$^+$.

16 g. The lens is not fixed and can be chosen according to different application environments. In practice, we prefer the lens of 6mm focal length, which is with 43° angle of view and about 10 metre clear observation distance. The camera board is installed facing to the positive flying direction of the quadcopter, as shown in figure 6.5, and the coordination frame is defined as section 3.1.1.

6.2.2 Inertial measurement unit

IMU consists of two different parts, i.e. linear accelerometer and rate gyroscope. As the name suggests, they can sensitively detect the variation and measure the instant linear acceleration and angular velocity directly. Traditionally, IMUs are built on large mechanical systems with complex moving parts, such as aviation inertial navigation system. When the fibre optic technologies are applied, accuracy has been increased to a large extent. However, these types of IMU are usually expensive and weigh several kilogrammes, which is impossible to install on a small mobile platform. Fortunately, the development of MEMS makes an IMU miniaturise in size and decrease in price. As a passive sensor, IMU does not need to rely on

Figure 6.5: The camera board is installed facing to the front of quadcopter.

external support for other devices, and no antennas or receivers are required.

Theoretically, by knowing the acceleration and angular velocity of an object, its pose variation can be calculated by integrating the raw sensor readings. However, although IMU can detect every small variation in movement, we still get the error accumulation of pose estimation while the double integration is performed over time. This has been shown in section 4.6.3. Most MEMS IMUs belong to consumer-grade products. The inner gyroscopes and accelerometer have common error sources such as the temperature variation and the running variation as random walks. The former can be compensated by calibration, while the latter should be estimated by a further analysis, where the Allan variance method [17] is commonly used.

In our mobile platform, we adopt a consumer-grade product named as myAHRS+ from ITHROBOT. The IMU includes 16-bit 3-axis accelerometer with the measurement limits as $\pm 16g$ and 16-bit 3-axis gyroscope with the range as ± 2000 dps. Its maximum rate is 100 Hz. The way of power supplement and data connection are the same as the camera board, through a USB interface.

The placement of IMU should be considered carefully, because of its sensitivity to variation. The mechanical vibration will bring in more noise into data. Additionally, to avoid the magnetic interference, the maximum distance should be kept from all the motors and high current components like batteries and electronic speed controllers. Overall the consideration, the centre of quadrotor is the best place for an IMU.

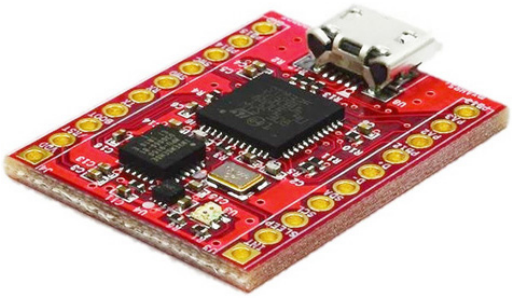


Figure 6.6: An IMU product: myAHRS+ from ITHROBOT.

6.2.3 Flight control board

Flight control board is the core part of a quadcopter. Every manoeuvre from taking off to landing should be processed through this part. It acts as a pilot in a manned vehicle, receiving the commands from various high-level sources while controlling the vehicle towards a target state by a series of motions as roll, pitch and yaw.

A primary flight control board mainly consists of a computing unit and several sensors. For our quadcopter, we adopt the modern autopilot hardware, Pixhawk, in MAV development community. It is an open-hardware released in the year of 2014, providing high availability with a low cost. The embedded real-time operating system on an ARM-cortex-M4 allows different sensor modules working efficiently together. This board also includes the double sets of MEMS IMUs, which can compensate with each other in highly agile manoeuvres. It should be noticed that we do not extract the inertial information from this control board since it needs to keep in a high regular running frequency, and any irregular operation will bring in extra unstable factors.

In our quadcopter, the Pixhawk module directly receives the commands from a radio remote control panel. These commands combine the variations as roll, pitch and yaw. Then based on the current state and the tuned parameters, the flight control board interprets the commands out as the form of Pulse Width Modulation (PWM), which can directly be used to drive the modules of electronic speed controller or motors. Before all these processes, the tuned parameters, particularly including the inner control method based on PID, should be obtained beforehand. This can follow an automatic tuning process as AUTOTUNE mode[1]. A tuning result can be found in figure 6.8.

[1]For more details, please refer to ardupilot.org.

Figure 6.7: Flight control board: Pixhawk.

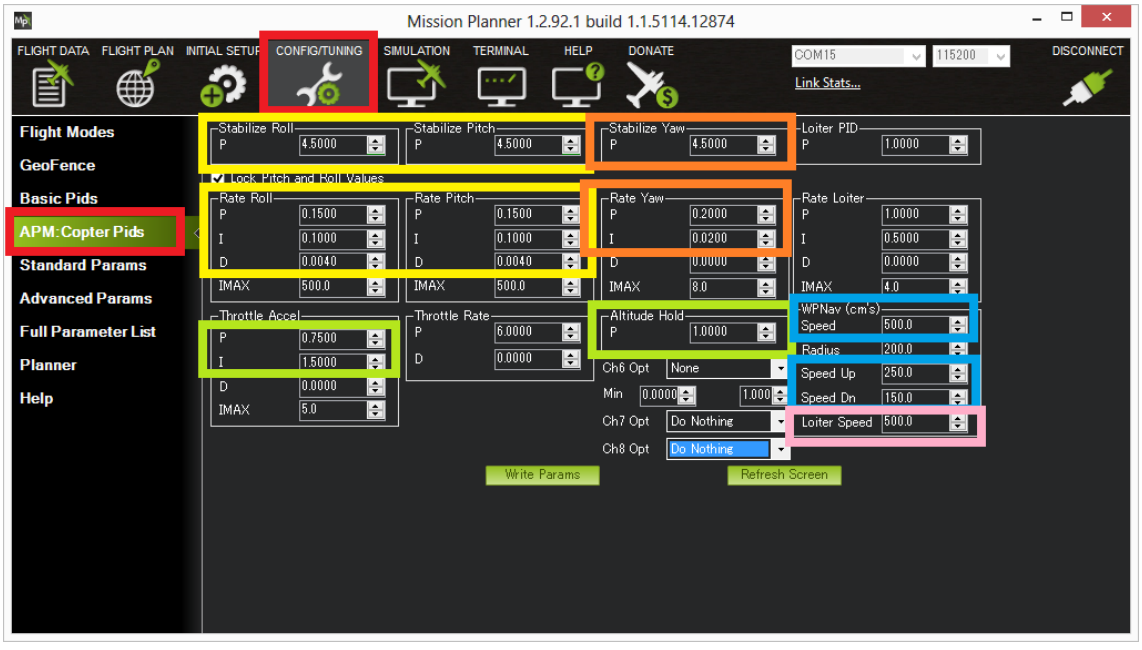


Figure 6.8: PID parameters through a process of automatic tuning.

CPUs	Samsung Exynos5422 Cortex^{TM}-A15 2.0Ghz quad core Cortex^{TM}-A7 quad core
Graphics card	Mali-T628 MP6
RAM	2Gbyte LPDDR3 at 933MHz PoP stacked
Flash storage	eMMC5.0 HS400
USB ports	3.0 Host ×1, 3.0 OTG ×1, 2.0 Host ×4
Other ports	HDMI 1.4a and DisplayPort1.1

Table 6.1: Specification of ODROID-XU3

6.2.4 High-level computing board

Other than the flight control board, the high-level computing board mainly takes care of the tasks which do not strictly rely on a regular high frequency. This board can mostly share the computing burden of the flight control board and guarantee the efficiency of the whole quadcopter system. It can be regarded as a master in high-level, monitoring the supplemental sensor feeding, making a decision, and forming the commands to flight control board if necessary. All the tasks such as localisation and navigation can usually be processed within this part.

After browsing various related commercial products in the robotic community, we trade off significant factors regarding computing power, energy efficiency and expandability, then choose the ODROID-XU3 as our high-level computing board, as shown in figure 6.9. An custom Ubuntu system can be installed on this board without compromising the booting and transfer speed. This is important because most development modules in the robotic community are available under the Ubuntu system, such as ROS. It will provide more possibility for the system developing. The specification of ODROID-XU3 can be found on table 6.1.

According to our target on this stage, the mission on this board is to record the data stream from visual and inertial sensors simultaneously. The monocular camera and IMU we selected in the previous sections are all connected to this board, which they get the power supply from and transfer the information to. Additionally, in order to switch on and off the recording task at controllable opportunities, we adopt the associated WIFI module. The onboard system thus can communicate with the desktop operation system wirelessly. It should be noticed that, to make the full use of the onboard resource, only the initial operation is performed through WIFI, the rest recording and shutting down tasks are processed all automatically on the high-level computing board. Some pictures of debugging the system can be found in figure 6.10.

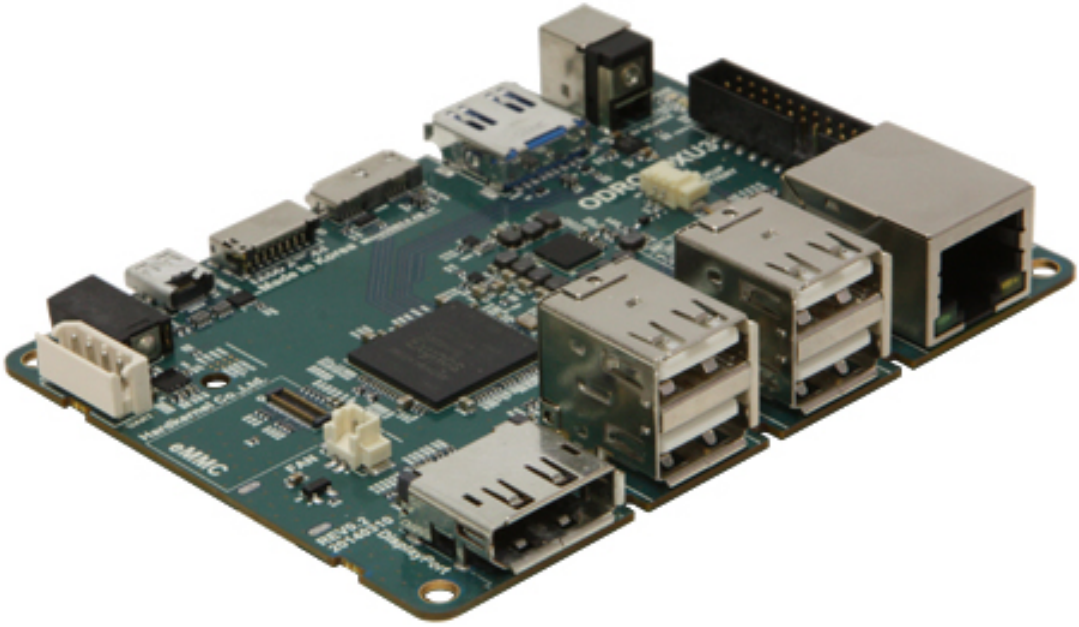


Figure 6.9: High-level computing board: ODROID-XU3.

(a) System under debugging (b) Compatibility with IDS Camera

Figure 6.10: High-level computing board under debugging.

6.3 Simulation

It is not an easy task to become a good operator for a quadcopter. It is necessary to get familiar with the behaviour of the quadcopter, thus reducing the risk of failure operation in practice. We conduct two different simulations for our quadcopter, where ROS and Matlab platforms are adopted. Both methods can build the simulated environments and display the flying dynamics, thus providing us more clues to design and plan the flying behaviour while performing data collection.

6.3.1 ROS-based method

Robot Operating System (ROS) is the most preferred robotic development environment in the robotic community. Its modular framework and a unified message processing strategy make the development of a required robot system become easy. Different components of a system can be developed and maintained separately without breaking the links between other parts. Various inputs from the same sensor type with different specifications can also be recorded and processed as a unified message format, thus making distributed computing becomes possible. Additionally, the large development community provides abundant supporting tools, which can be adopted as modules for a customised robotic system only with slight modification. For our platform, we would like to simulate the process that commands are sent from a remote controller, then instant responding can be found in the behaviour of quadcopter. Based on the package of *hector_quadrotor*[1], we only change the structural parameters in related files, which include the weight and size for major parts. The remote controller can also be set as an individual node, which simulates the operations of a real remote controller.

Figure 6.11 shows a virtual environment where we should control the quadcopter fly stably without any conflict. The node and topic diagram (figure 6.12) shows the remote controller as */joy* in a circle, and the quadcopter system is contained in the node of */gazebo*. As we can see from the right part of this diagram, the outputs show various simulated topics, including the visual information as a front camera and the inertial information from IMU. Although these data not can be used to drive an algorithm due to the existence of large impractical setting, the process of controlling the quadcopter with an instant response through a remote controller can still be successfully simulated.

[1]For more details, please refer to ros.org.

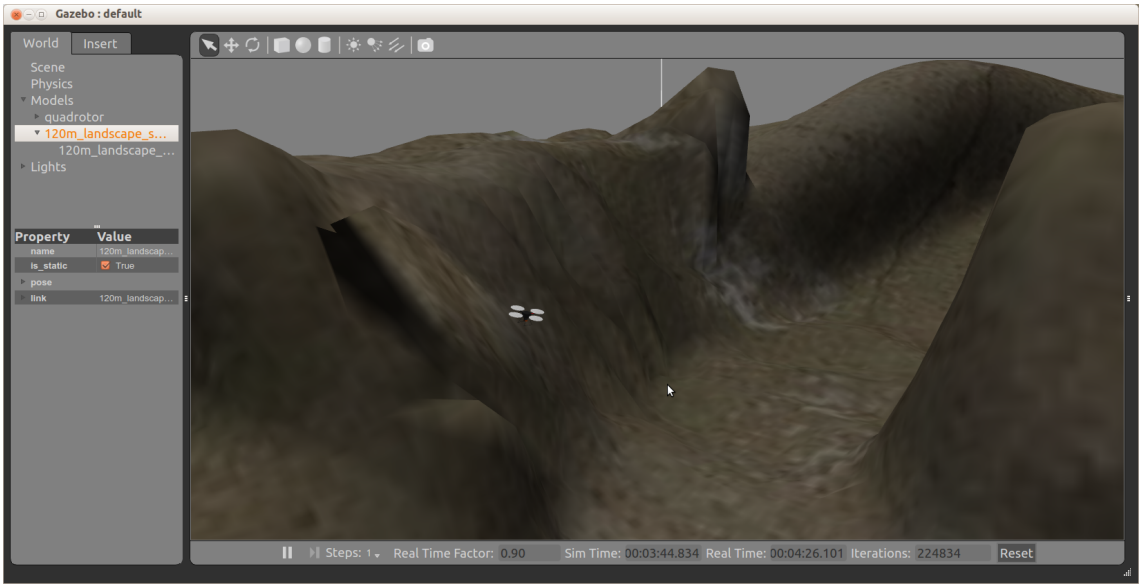


Figure 6.11: Quadcopter simulation in Gazebo

6.3.2 Matlab-based method

Matlab is widely used in the numerical computing and analysis. It also provides abundant additional toolboxes to perform a graphical multi-domain simulation. By applying our system model and parameters to the quadcopter tool[1] in Simulink, the output signal can be displayed live, and the input signals can be graphically designed as shown in figure 6.13. This is helpful to understand the instant response behaviour as different operations in a remote controller.

However, all the experiments in simulation only provide us with a guidance to operating a real quadcopter in the indoor and outdoor environment. The actual visual and inertial datasets while flying are the goals of our quadcopter.

6.4 Quadcopter System

The core system of our integrated quadcopter can be found in figure 6.15(a), which can be controlled to perform any manoeuvre as roll, pitch and yaw with a remote controller. A hovering behaviour over the ground proves such basic quadcopter system can work stably with suitable inner parameters. The full quadcopter system with visual and inertial data collecting parts can be found in figure 6.15(b). The high-level computing board is mounted in a case at the top centre of the quadcopter, while the battery is moved to the bottom centre. The monocular camera is installed

[1]For more details, please refer to mathworks.com.

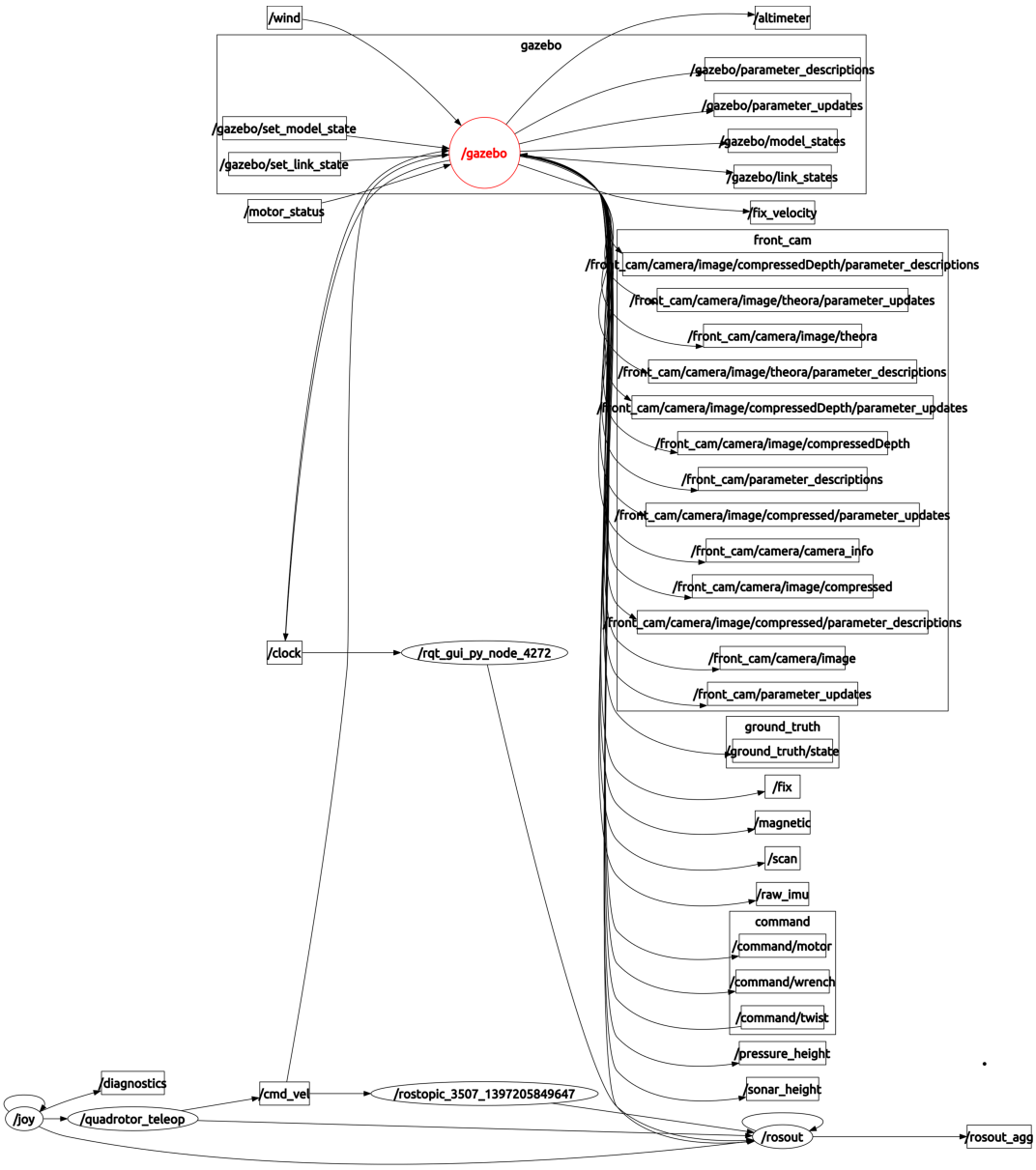
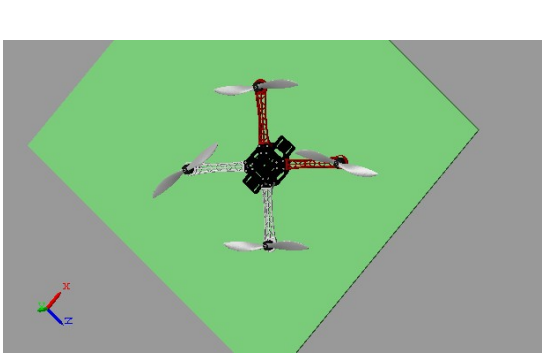
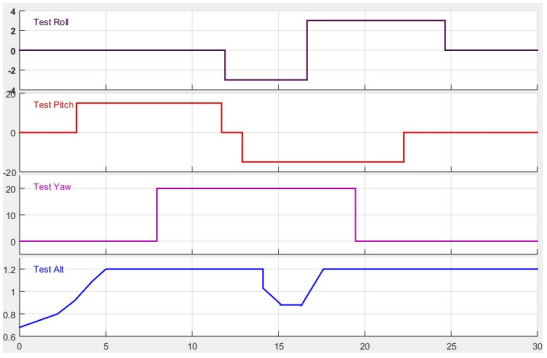


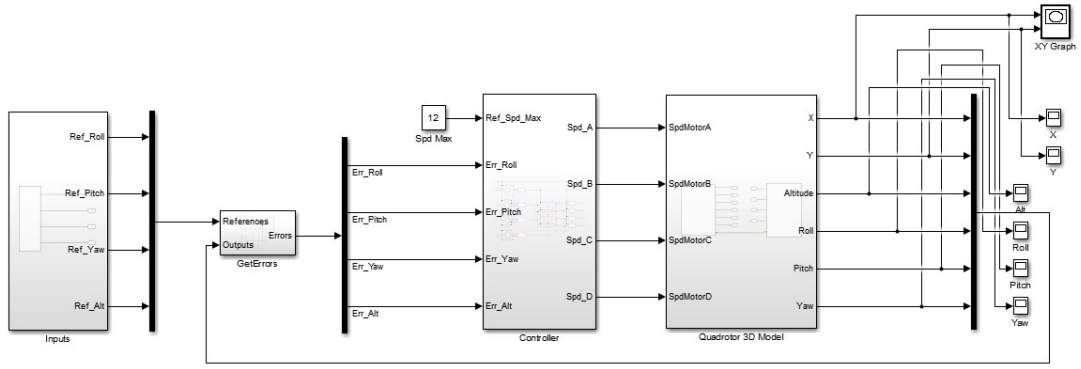
Figure 6.12: Nodes and topics diagram of a quadcopter system in ROS. The quadcopter system is contained in the node of */gazebo*.



(a) Quadcopter graphical model in Matlab.



(b) Design input commands ghraphically.



(c) Simulation modules in Simulink.

Figure 6.13: Simulink modules and input signals design for quadcopter in Matlab.

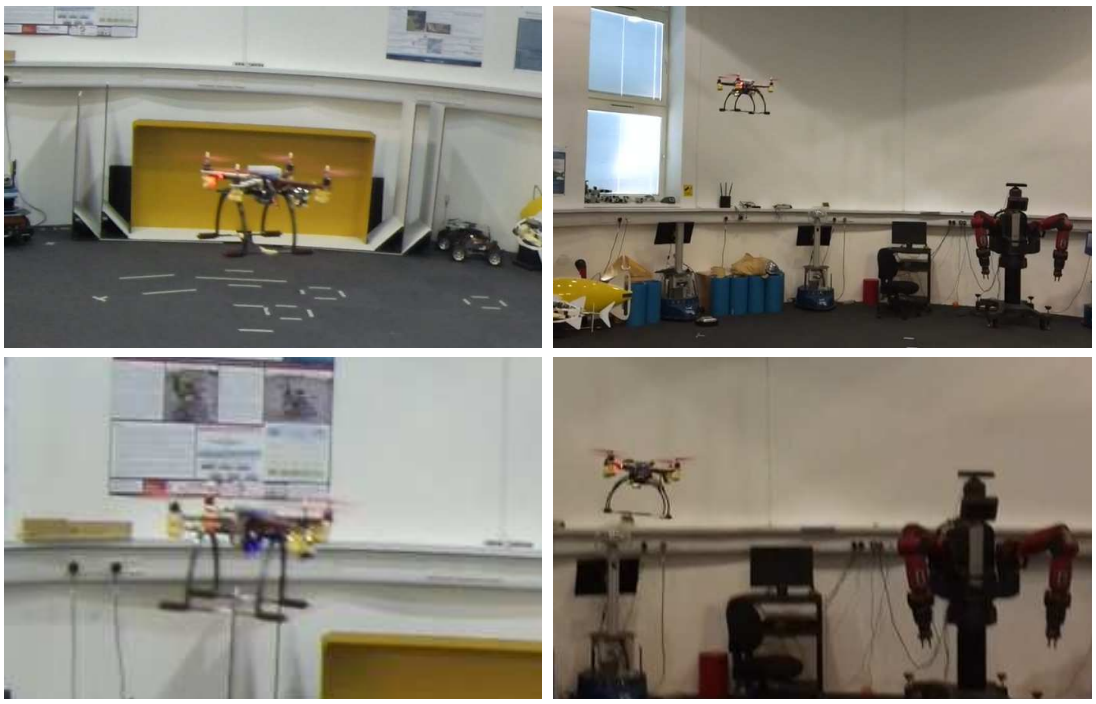
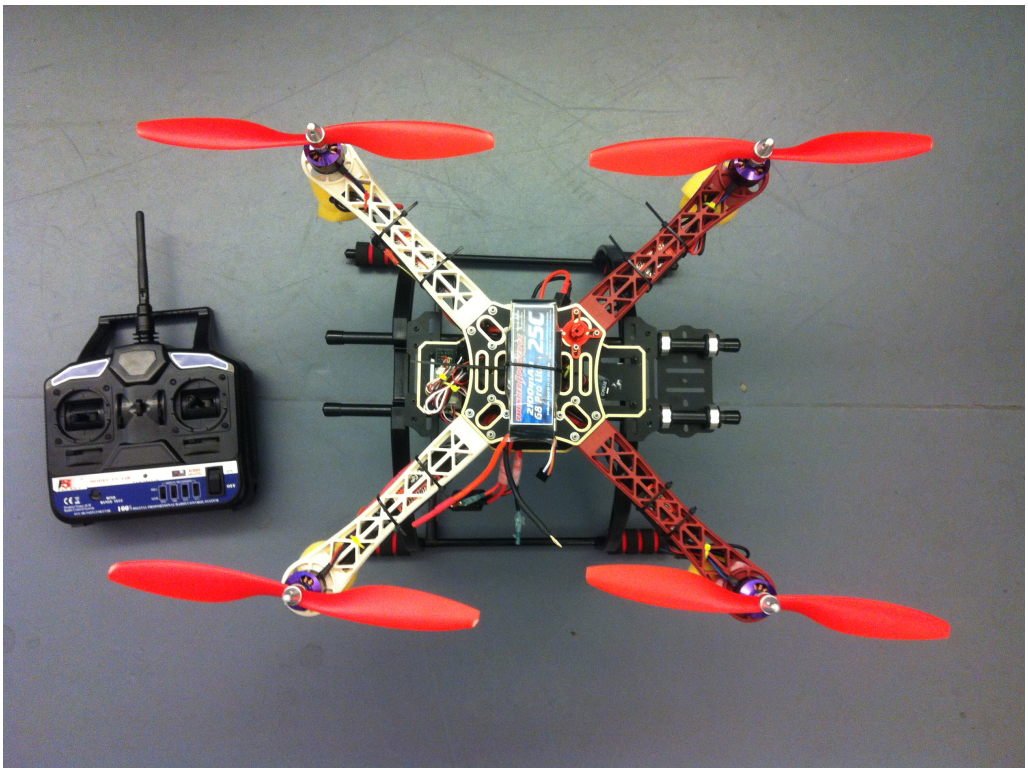


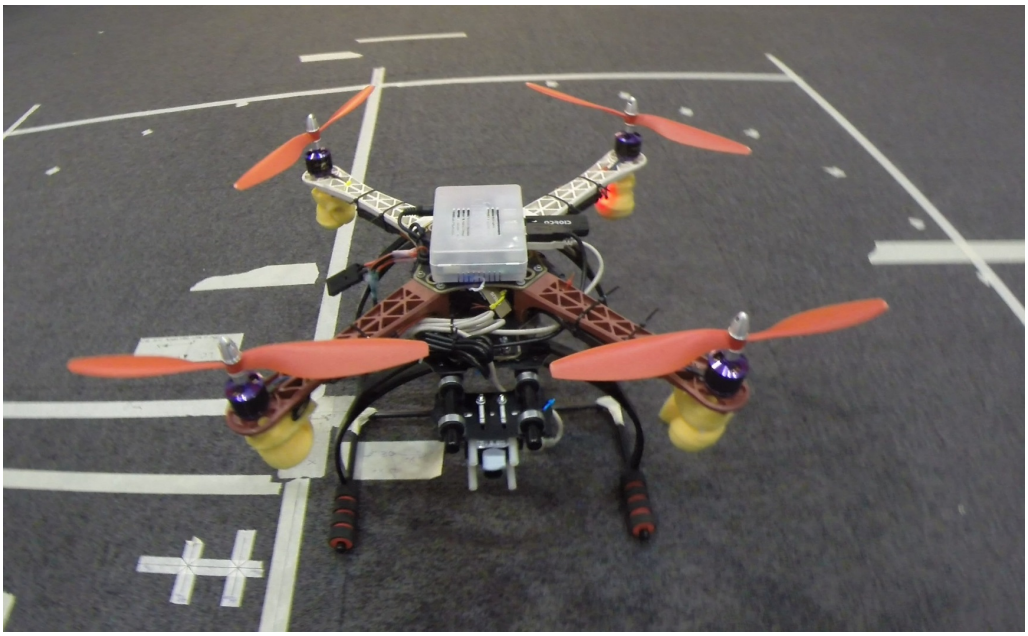
Figure 6.14: Visual and inertial data streams are recorded under different illumination conditions.

facing to the front. All payloads are carefully adjusted to maintain the gravity centre at the middle of the quadcopter.

The visual and inertial data is stored live in the memory of the high-level computing board. We use the tool of *rosbag* to save the data, which can guarantee both of the data streams tracked in the same timeline. Figure 6.14 shows the case of our quadcopter collecting the data under different illumination conditions. Such data have been used to evaluate the robustness of MI-based method in previous section 5.5, while the data bags of outdoor environment have been used to estimate the trajectory of our MSCKF-direct method in section 4.6.3.



(a) Basic quadcopter system without extra high-level parts



(b) Full quadrotor system with visual-inertial data collecting system

Figure 6.15: Our resulting integrated quadcopter system.

6.5 Summary

The goal of this chapter is to build a data collecting mobile platform for the testing of visual-inertial odometry algorithms. Specifically, this mobile platform is a quadrotor, developed by using commercial-off-the-shelf components to reduce development time and to increase reliability. Even most of the components are well supported in the robotic community, we still need to adjust both the hardware and software interface to get a fully compatible system.

The key sensors in our platform include a monocular camera and an inertial measurement unit, which can provide time-stamped data stream by using the *rosbag* tool in ROS. As the major hardware parts for our platform, the flying control board and high-level computing board are also presented. All the tasks tightly concerning the motor control are implemented through the flying control board, while the data receiving and storing are implemented by the high-level computing board.

To reduce the failure rate when performing practical operation, we use two simulation methods to observe the dynamics and instant response of our quadcopter system when sending different signals by a remote controller. Results show that the quadcopter can be controlled stably hovering and flying in the outdoor and indoor environments. Several datasets have been applied in the algorithm testing for previous chapters.

However, our quadcopter still has the drawback of weighty and significant power consuming. To perform the fully autonomous tasks as localisation and navigation, the high-level computing board needs to be developed further.

Chapter 7

Conclusions and Future Work

In this thesis, we present the contributions to the direct visual-inertial odometry for monocular mobile platforms in the indoor and outdoor environments. Starting from an introduction (chapter 1) of visual-inertial perception and motion estimation problems, a background review of the state-of-the-art visual SLAM and odometry methods of both feature-based and direct-based are presented in chapter 2. Given necessary technical preliminaries regarding the visual geometry, motion kinematics and computing frameworks in chapter 3, our first contribution as a direct visual-inertial odometry method is presented in chapter 4, where a multi-state constraint Kalmen filter is used to fuse inertial data and direct visual information. Another novel measurement model based on mutual information theory is presented in chapter 5, which is also a direct-based approach to enhancing the robustness in image processing. In chapter 6, we introduce the experience and principal components to build a quadcopter. This mobile platform can manoeuvre freely under a remote controller in the indoor and outdoor environments while recording the dynamic data from a set of IMU and monocular camera simultaneously. In the following, we will conclude the contributions made in this thesis.

7.1 Contribution Summary

Visual inertial odometry is a motion estimation technique that provides the pose increments for mobile platforms depending on consecutive images with the aid of acceleration and angular measurements. The estimated locations and attitudes are the key for performing any further tasks for mobile platforms. Take the quadcopter as an example, such micro aerial vehicles are commonly found on the occasion of military or civil applications, such as the resource exploring, mapping, search and

rescue, especially in human inaccessible environments. However, without knowing the instant pose information, such platforms not can manoeuvre autonomously.

In contrast to other methods that can also acquire the motion information for mobile platforms, such as SLAM, the VIO method does not make the focus on mapping or loop closure detection, thus saving the valued computing resources for other core tasks. The compensative nature of visual and inertial sensors can help each other to largely bound the rapid rising tendency of accumulated errors. Specifically, the fast drift issue from the inertial-driven model can be limited by visual-based estimates, while the rich inertial dynamics can provide a sufficient prior information between visual sampling intervals.

7.1.1 Insightful review

In *the chapter of background and literature review*, three main categories of methods related to our thesis are reviewed, i.e. feature-based, direct-based and visual-inertial fusion. We start from the most common feature-based image processing techniques. The motion and structure recovery methods, including SLAM, SfM and VO are presented. Such methods are the most representative in the robotic and computer vision communities, which model the environment as spatial feature points and measure the disparities as re-projection errors.

The direct visual measurement methods only become popular in recent years, which eliminate the extracting and matching procedures of feature-based methods. Some notable works in the literature include LSD-SLAM, DTAM, DSO and MI-based methods. Although some of these works are computational demanding when using dense pixels to form photometric errors, the advantages of making full use of the visual information and enhancing the robustness of algorithms are also prominent.

For the fusion of visual and inertial information, there are two major frameworks in the literature, i.e. loosely and tightly coupled methods. The former takes the filtering as representation and processes inertial and visual measurements separately. In contrast, the tightly coupled methods jointly estimate robotic pose and visual measurements in an iterative minimization process. Its accuracy can be increased by trading off the computational efficiency, while the filtering methods have the advantage of computational efficiency.

7.1.2 MSCKF-direct VIO method

Our first VIO method is proposed in *the chapter of direct visual-inertial fusion in multi-state constraint Kalman filter* (chapter 4). In this method, we apply an inertial-driven dynamic model in the propagation step and develop a measurement model that directly uses photometric information from pixel-centred patches in the update step. This fusion approach leverages the benefits of state augmentation and removal in the MSCKF, which keeps only recent camera poses in a state vector. Therefore, a limited computational complexity can be kept even for a long time running with volume pixel patches.

The novel visual measurement model is based on direct pixel information from small patches. These patches are scattered at salient edges across the whole image. Different from feature-based methods, their locations and patterns are irregular, and all the prominent image area will be taken into account. Therefore, higher utilisation of an image can be achieved. Especially for the image containing large repetitive texture. However, this direct visual measurement model is vulnerable when the illumination variation or a significant rotation appears since the assumption of a small translation with a large common field of view have been broken.

Additionally, the intrinsic links between various estimation methods are analysed and illustrated. It shows the fact that adding an iterative process in the filtering framework is equivalent to the optimisation based estimation under the view of likelihood maximisation.

7.1.3 MI-based VIO method

Our second VIO method is proposed in *the chapter of direct visual-inertial odometry based on mutual information* (chapter 5). We firstly adopt the theories of probability and information entropy to build the MI formulation for digital images. When regarding consecutive images as discrete random variables, their similarity can be valued through the MI measures, The value of MI can distinguish more extreme situations than joint entropy. A simple case is that two tightly correlated variables lead to the same value of joint entropy, but if one variable becomes constant, the value of joint entropy remains the same while the value of MI will change to zero.

Further following the visual and inertial fusion framework in the MSCKF, a novel MI-based visual measurement model is developed. The derivation process of this model is presented in details, where the pose is concisely formulated as a minimal expression using the knowledge of Lie algebra. Finally, we find that the Jacobian of

MI is related to the derivatives of joint probability and projective wrap function for the pose. The MI-based VIO method exists the drawback that a prior scene model is required, and the estimation error will be amplified based on such model. However, the robustness is still significant for the MI-based measurement when applied in the cases of illumination variation, partial occlusion and noise pollution.

Additionally, following the analysis of the iterative process in a filtering framework, we also propose an iterative equation based on the Levenberg-Marquardt solution. In this process, the Hessian matrix of MI function to the camera pose is presented. If continuously applying the inertial-driven pose as the initial optimum point for linearisation, the computation burden of iterative steps can be expected to reduce.

7.1.4 Data collection platform

As a data collection mobile platform, a quadcopter is built up. The practical experience and major components are introduced in *the chapter of mobile platform for data collection* (chapter 6). Different from commercialised MAV platforms, our quadcopter owns a great compatibility in both software and hardware. On the one hand, there is volume supporting resources for flying control board and high-level computing board, which makes the development of software becomes efficiency. Only the software interfaces and hardware related parameters need to be further modified. In the high-level computing board, an elite version of Ubuntu system can be installed for applying various robotic resource like ROS packages.

On the other hand, additional sensors including a specified monocular camera and an IMU can run with an excellent performance in the platform. The data stream can also be stored lively in onboard memory. The specification of the high-level control board indicates that it can process the estimation algorithm in real-time if being further developed. It is worth to notice that, in our quadcopter, if the motors and propellers are carefully selected, the power can be increased further, thus making more sensors or batteries as payloads available.

Additionally, although there exists inconsistency between simulated and real system, the available simulation can provide us a helpful guidance in practical operations. The simulations performed in ROS and Matlab illustrate the system from a different view point. The modular structure in ROS showcases the connections between various components while the numerical illustration in Matlab the relationship between the manual commands and the instant response for our quadcopter.

7.2 Future Works

We notice that in the situation of rotation or fast motion, our direct visual-inertial odometry methods are easy to accumulate significant error over time and the algorithm will be divergent eventually. Although it is unavoidable for the odometry problem, we still can explore the ways to bound the error or slow down this trend. Here, we get some ideas from the perspectives of software techniques and hardware complements.

On the one hand, we could explore to maintain some past keyframes and key poses instead of just removing them by marginalisation. Such critical information can provide revisit references to the current estimates. It is similar to the loop closure steps in SLAM techniques, but the frequency and function of such key information are different. Firstly, the number of such key frames or pose can be altered actively according to the motion status. More details will be recorded if a large change field of view is observed even for a small movement. Secondly, the historical references are only used to bound the accumulation error, and their poses will not be globally updated as those did in SLAM methods.

Furthermore, direct-based methods bring in more available visual information into an estimation algorithm, thus increasing the computational burden at the same time. It is always an issue to trade off accuracy and efficiency in practical robotic applications. In the earlier exploration, we find that the direct-based methods and feature-based methods are not a black or white problem. Although the sparse features not can express the image as productive as that of direct-based methods, the FAST techniques in feature processing still have the advantage in efficiency. There exists no substantial gap to leverage the merits of both approaches. We can use the feature-based methods to get an initial estimation result and treat it as an approximated solution. Then, this initial guess is used in further direct-based optimisation. Or both of the projective and photometric errors are jointly used to build a measurement or cost function. This combination can reduce the number of direct pixels in the calculation with the promising of improving the efficiency.

From the perspective of algorithm computation, the strategy of multi-threads can be applied. We can try to distribute the visual related tasks to GPU and maintain a fixed calculation rate in the main computing unit to guarantee a stable output in practical usage.

Also, when applying the MI-based measurement in motion estimation, we should precondition a scene model, which will provide the depth reference for the later

comparison. There are various methods which can build this initial model. While in practical applications, such scene model is not always accessible instantly. It takes time to rebuild a new scene structure when the scene is altered. To break the limitation of such prerequisite and enhance the intelligence of mobile platforms, we can try to build a semantic library for general objects, such as computers and chairs in an office scene. Different from static points and pixels in any feature-based or direct-based methods, semantic objects are meaningful. The common sense of defined objects can also be adopted, like the size of a desktop monitor in an office can be used to evaluate the real scale of motion. Such objects can be loaded in memory beforehand with a functional information, then extracted lively as references for the motion estimation and further navigation tasks. Furthermore, the scene expression and recognition can be combined with the modern research of deep learning. The reference objects can be learnt through a training and learning process.

Concerning mobile platforms, our current quadcopter owns the merits of high compatibility and high payload capacity by combining multiple commercial components together. However, it still has the issues in power distribution and consumption. The redundancy of integrated hardware components adds unnecessary weights on the platform. To achieve a minimised system with enough sensory and computing capability, some electronic circuit components should be redesigned accordingly. The redundant modules in the electronic system such as extra communication ports can be removed, and the cables can be fixed in circuits. Also, the fabric material of the body frame can be used to lighten the weight without comprising a required strength for flying.

Current research for VIO methods just provides the most basic estimation result to get the motion of mobile platforms. To achieve a target platform as an individual system, which has the ability of fully autonomous, more tasks like navigation, mission planning should be carefully studied accordingly.

Appendix

Converting rotation representations

We give the conversions between the three rotation representations presented in this thesis: the rotation matrix, the Euler angles and the quaternion.

Euler angles to rotation matrix: Given the Euler angles as $\mathbf{e} = (\phi \quad \theta \quad \psi)^T$ corresponding to the *roll*, *pitch* and *yaw* orientations in the ZYX convention, the corresponding rotation matrix is given by

$$\mathbf{R} = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\theta\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\theta\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\theta\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}.$$

Quaternion to rotation matrix: Given the quaternion as $\mathbf{q} = (a \quad b \quad c \quad d)^T$, the rotation matrix is accordingly given by

$$\mathbf{R}(\mathbf{q}) = \begin{bmatrix} a^2 + b^2 - c^2 - d^2 & 2 \cdot (bc - ad) & 2 \cdot (bd + ac) \\ 2 \cdot (bc + ad) & a^2 - b^2 + c^2 - d^2 & 2 \cdot (cd - ab) \\ 2 \cdot (bd - ac) & 2 \cdot (cd + ab) & a^2 - b^2 - c^2 + d^2 \end{bmatrix}.$$

Rotation matrix to Euler angles: Given the rotation matrix as $\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$, the three Euler angles $\mathbf{e} = (\phi \quad \theta \quad \psi)^T$ corresponding to the same rotation are given by

$$\theta = \arcsin(-r_{31}), \quad \phi = \arctan(\frac{r_{32}}{r_{33}}), \quad \text{and} \quad \psi = \arctan(\frac{r_{21}}{r_{11}}).$$

There is another solution in different quadrant if defining $\theta = \pi - \arcsin(-r_{31})$.

Euler angles to quaternion: This conversion is best achieved by using quaternion algebra. We first define the three quaternions corresponding to the three elementary Euler rotations:

$$\mathbf{q}_\phi = \begin{bmatrix} \cos(\phi/2) \\ \sin(\phi/2) \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{q}_\theta = \begin{bmatrix} \cos(\theta/2) \\ 0 \\ \sin(\theta/2) \\ 0 \end{bmatrix}, \quad \text{and} \quad \mathbf{q}_\psi = \begin{bmatrix} \cos(\psi/2) \\ 0 \\ 0 \\ \sin(\psi/2) \end{bmatrix}.$$

The composed rotation is obtained by multiplying them up as

$$\mathbf{q} = \mathbf{q}_\psi \mathbf{q}_\theta \mathbf{q}_\phi = \begin{bmatrix} \cos(\psi/2)\cos(\theta/2)\cos(\phi/2) + \sin(\psi/2)\sin(\theta/2)\sin(\phi/2) \\ \cos(\psi/2)\cos(\theta/2)\sin(\phi/2) - \sin(\psi/2)\sin(\theta/2)\cos(\phi/2) \\ \cos(\psi/2)\sin(\theta/2)\cos(\phi/2) + \sin(\psi/2)\cos(\theta/2)\sin(\phi/2) \\ -\cos(\psi/2)\sin(\theta/2)\sin(\phi/2) + \sin(\psi/2)\cos(\theta/2)\cos(\phi/2) \end{bmatrix}.$$

Quaternion to Euler angles: Given the quaternion as $\mathbf{q} = (a \quad b \quad c \quad d)^T$, the Euler angles $\mathbf{e} = (\phi \quad \theta \quad \psi)^T$ corresponding to the same rotation are given by

$$\begin{aligned} \phi &= \arctan(\tfrac{2cd+2ab}{a^2-b^2-c^2+d^2}), \\ \theta &= \arcsin(-2bd+2ac), \\ \psi &= \arctan(\tfrac{2bc+2ad}{a^2+b^2-c^2-d^2}). \end{aligned}$$

We should notice that there exists multiple solutions in different quadrant.

References

[1] Bernhard Hofmann-Wellenhof, Herbert Lichtenegger, and James Collins. *Global positioning system: theory and practice*. Springer Science & Business Media, 2012. 2, 3

[2] Jay Farrell. *Aided navigation: GPS with high rate sensors*. McGraw-Hill, Inc., 2008. 2

[3] Bernhard Hofmann-Wellenhof, Herbert Lichtenegger, and Elmar Wasle. *GNSS-global navigation satellite systems: GPS, GLONASS, Galileo, and more*. Springer Science & Business Media, 2007. 3

[4] Adam Bry, Abraham Bachrach, and Nicholas Roy. State estimation for aggressive flight in GPS-denied environments using onboard sensing. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1–8. IEEE, 2012. 3

[5] Abraham Bachrach, Samuel Prentice, Ruijie He, and Nicholas Roy. RANGE-robust autonomous navigation in GPS-denied environments. *Journal of Field Robotics*, 28(5):644–666, 2011. 28

[6] Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments. *Journal of Field Robotics*, 28(6):854–874, 2011. 27

[7] Davide Scaramuzza, Michael C Achtelik, Lefteris Doitsidis, Fraundorfer Friedrich, Elias Kosmatopoulos, Agostino Martinelli, Markus W Achtelik, Margarita Chli, Savvas Chatzichristofis, Laurent Kneip, et al. Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in GPS-denied environments. *IEEE Robotics & Automation Magazine*, 21(3):26–40, 2014. 3

[8] John S Brlow. Inertial navigation as a basis for animal navigation. *Journal of Theoretical Biology*, 6(1):76–117, 1964. 3

[9] Sergi Foix, Guillem Alenya, and Carme Torras. Lock-in time-of-flight (tof) cameras: A survey. *IEEE Sensors Journal*, 11(9):1917–1926, 2011. 3

[10] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 3, 19, 42, 47, 127

[11] Mingyang Li and Anastasios I Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711, 2013. 3, 8, 34

[12] Stephan Weiss and Roland Siegwart. Real-time metric state estimation for modular vision-inertial systems. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4531–4537. IEEE, 2011. 35

[13] Anastasios I Mourikis and Stergios I Roumeliotis. A multi-state constraint Kalman filter for vision-aided inertial navigation. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3565–3572. IEEE, 2007. 34, 36

[14] Dennis Strelow and Sanjiv Singh. Motion estimation from image and inertial measurements. *The International Journal of Robotics Research*, 23(12):1157–1195, 2004. 3, 4, 34

[15] Hardkernel.com. Mems imu myahrs+. http://www.hardkernel.com/main/products/prdt_info.php?g_code=G141464363369. Accessed May 10, 2017. 3, 5

[16] Averil Burton Chatfield Chatfield. *Fundamentals of high accuracy inertial navigation*, volume 174. Aiaa, 1997. 3

[17] Oliver J Woodman. An introduction to inertial navigation. Technical report, University of Cambridge, Computer Laboratory, 2007. 4, 138

[18] Volker Kempe. *Inertial MEMS: principles and practice*. Cambridge University Press, 2011. 4, 5

[19] Wikipedia.org. Time-of-flight camera. https://en.wikipedia.org/wiki/Time-of-flight_camera. Accessed May 1, 2017. 5

[20] Denis Chekhlov, Mark Pupilli, Walterio Mayol-Cuevas, and Andrew Calway. Real-time and robust monocular SLAM using predictive multi-resolution descriptors. In *Advances in visual computing*, pages 276–285. Springer, 2006. 5

[21] Raul Mur-Artal, JMM Montiel, and Juan D Tardós. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 9, 17, 24, 25, 30

[22] Stefan Leutenegger, Paul Timothy Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. Keyframe-based visual-inertial SLAM using nonlinear optimization. In *Robotics: Science and Systems*, 2013. 5, 35

[23] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. *2007 6th IEEE ACM Int. Symp. Mix. Augment. Real.*, pages 1–10, November 2007. 5, 9, 23

[24] Ids-imaging.com. Ids industrial camera. https://en.ids-imaging.com/store/products/cameras.html. Accessed January 10, 2017. 5

[25] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *Robotics & Automation Magazine, IEEE*, 13(2):99–110, 2006. 5, 6

[26] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (SLAM): part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006. 5, 6

[27] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–652. IEEE, 2004. 5

[28] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *Robotics & Automation Magazine, IEEE*, 18(4):80–92, 2011. 29

[29] Friedrich Fraundorfer and Davide Scaramuzza. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robotics & Automation Magazine*, 19(2):78–90, 2012. 5

[30] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. 5

[31] Agostino Martinelli. Visual-inertial structure from motion: observability and resolvability. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4235–4242. IEEE, 2013. 5

[32] Adrien Angeli, David Filliat, Stéphane Doncieux, and Jean-Arcady Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics*, 24(5):1027–1037, 2008. 6

[33] Khaled S Ali, C Anthony Vanelli, Jeffrey J Biesiadecki, Mark W Maimone, Yang Cheng, A Miguel San Martin, and James W Alexander. Attitude and position estimation on the Mars exploration rovers. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 1, pages 20–27. IEEE, 2005. 7

[34] Stefan Leutenegger and Paul Furgale. Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization. *Robot. Sci.*, 2013. 8

[35] Anton Kasyanov, Francis Engelmann, Jörg Stückler, and Bastian Leibe. Keyframe-Based Visual-Inertial Online SLAM with Relocalization. *arXiv preprint arXiv:1702.02175*, 2017.

[36] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3):314–334, 2015. 34

[37] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017. 8, 34, 35

[38] Laurent Kneip, Stephan Weiss, and Roland Siegwart. Deterministic initialization of metric state estimation filters for loosely-coupled monocular vision-inertial systems. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2235–2241. IEEE, 2011. 8

[39] Jianjun Gui, Dongbing Gu, Sen Wang, and Huosheng Hu. A review of visual inertial odometry from filtering and optimisation perspectives. *Advanced Robotics*, 29(20):1289–1301, 2015. 80

[40] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. Georgia Institute of Technology, 2015. 8, 34

[41] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *AAAI/IAAI*, pages 593–598, 2002. 9, 22

[42] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. MonoSLAM: Real-time single camera SLAM. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, 2007. 9, 23

[43] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014. 9, 10, 30, 32

[44] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast Semi-Direct Monocular Visual Odometry. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, 2014. 29, 30, 35

[45] Michal Irani and P Anandan. About direct methods. In *Vision Algorithms: Theory and Practice*, pages 267–277. Springer, 2000.

[46] Vladyslav Usenko, Jakob Engel, Jörg Stückler, and Daniel Cremers. Direct visual-inertial odometry with stereo cameras. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1885–1892. IEEE, 2016. 34

[47] Guillaume Caron, Amaury Dame, and Eric Marchand. Direct model based visual tracking and pose estimation using mutual information. *Image and Vision Computing*, 32(1):54–63, 2014. 10, 33

[48] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. In *arXiv:1607.02565*, July 2016. 9, 31, 32

[49] Amaury Dame and Eric Marchand. Mutual information-based visual servoing. *Robotics, IEEE Transactions on*, 27(5):958–969, 2011. 10, 33

[50] Marc Pollefeys, Reinhard Koch, Maarten Vergauwen, and Luc Van Gool. Hand-held acquisition of 3D models with a video camera. In *3-D Digital*

Imaging and Modeling, 1999. Proceedings. Second International Conference on, pages 14–23. IEEE, 1999. 15

[51] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988. 15, 16

[52] Rachid Deriche and Olivier Faugeras. Tracking line segments. *Image and vision computing*, 8(4):261–270, 1990. 15

[53] Gregory D Hager and Peter N Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE transactions on pattern analysis and machine intelligence*, 20(10):1025–1039, 1998. 15, 116

[54] Samia Boukir, Patrick Bouthemy, François Chaumette, and Didier Juvin. A local method for contour matching and its parallel implementation. *Machine Vision and Applications*, 10(5-6):321–330, 1998. 15

[55] Markus Vincze. Robust tracking of ellipses at frame rate. *Pattern Recognition*, 34(2):487–498, 2001. 15

[56] MO Berger. How to track efficiently piecewise curved contours with a view to reconstructing 3D objects. In *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, pages 32–36. IEEE, 1994. 15

[57] Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on image processing*, 10(2):266–277, 2001. 15

[58] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):105–119, 2010. 16

[59] Tony Lindeberg. Feature detection with automatic scale selection. *International journal of computer vision*, 30(2):79–116, 1998. 16

[60] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 16, 17

[61] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*, pages 2564–2571. IEEE, 2011. 16, 18

[62] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. BRISK: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011. 16

[63] Daniel Jaymin Mankowitz and Subramanian Ramamoorthy. BRISK-based visual feature extraction for resource constrained robots. In *Robot Soccer World Cup*, pages 195–206. Springer, 2013. 17

[64] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417, 2006. 17, 91

[65] Michael Calonder, Vincent Lepetit, Mustafa Ozuysal, Tomasz Trzcinski, Christoph Strecha, and Pascal Fua. BRIEF: Computing a local binary descriptor very fast. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1281–1298, 2012. 18

[66] Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999. 18

[67] John L Barron, David J Fleet, Steven S Beauchemin, and TA Burkitt. Performance of optical flow techniques. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 236–242. IEEE, 1992. 18

[68] Middlebury.edu. Middlebury website for latest optical flow methods. `http://vision.middlebury.edu/flow/`. Accessed May 1, 2017. 18

[69] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981. 18

[70] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981. 18

[71] Stephen Se, David Lowe, and Jim Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 2051–2058. IEEE, 2001. 18

[72] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and image processing*, 14(3):227–248, 1980. 19

[73] Roy De Maesschalck, Delphine Jouan-Rimbaud, and Désiré L Massart. The mahalanobis distance. *Chemometrics and intelligent laboratory systems*, 50(1):1–18, 2000. 19

[74] Richard I Hartley. In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593, 1997. 19

[75] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004. 19

[76] Vadim Indelman, Andrew Melim, and Frank Dellaert. Incremental light bundle adjustment for robotics navigation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1952–1959. IEEE, 2013. 19

[77] Hongdong Li. Multi-view structure computation without explicitly estimating motion. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2777–2784. IEEE, 2010. 19

[78] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, 43(1):55–81, 2015. 20

[79] Andrew J Davison and David W Murray. Mobile robot localisation using active vision. In *European Conference on Computer Vision*, pages 809–825. Springer, 1998. 21

[80] P Cheeseman, R Smith, and M Self. A stochastic map for uncertain spatial relationships. In *4th International Symposium on Robotic Research*, pages 467–474, 1987. 21

[81] John J Leonard and Hugh F Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *Intelligent Robots and Systems' 91.'Intelligence for Mechanical Systems, Proceedings IROS'91. IEEE/RSJ International Workshop on*, pages 1442–1447. IEEE, 1991. 21

[82] Stephane Betge-Brezetz, Patrick Hebert, Raja Chatila, and Michel Devy. Uncertain map making in natural environments. In *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, volume 2, pages 1048–1053. IEEE, 1996.

[83] Jose A Castellanos and Juan D Tardos. *Mobile robot localization and map building: A multisensor fusion approach*. Springer Science & Business Media, 2012.

[84] Paul Newman. On the structure and solution of the simultaneous localisation and map building problem. *Doctoral diss., University of Sydney*, 41, 1999.

[85] Jorge Artieda, José M Sebastian, Pascual Campoy, Juan F Correa, Iván F Mondragón, Carol Martínez, and Miguel Olivares. Visual 3-D SLAM from UAVs. *Journal of Intelligent and Robotic Systems*, 55(4):299–321, 2009. 21

[86] Simon J Julier and Jeffrey K Uhlmann. A counter example to the theory of simultaneous localization and map building. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4238–4243. IEEE, 2001. 21

[87] Sebastian Thrun, Daphne Koller, Zoubin Ghahramani, Hugh Durrant-Whyte, and Andrew Y Ng. Simultaneous mapping and localization with sparse extended information filters: Theory and initial results. In *Algorithmic Foundations of Robotics V*, pages 363–380. Springer, 2004. 21

[88] Mark A Paskin. Thin junction tree filters for simultaneous localization and mapping. In *in Int. Joint Conf. on Artificial Intelligence*. Citeseer, 2003.

[89] Michael Montemerlo and Sebastian Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 2, pages 1985–1991. IEEE, 2003.

[90] Giorgio Grisettiyz, Cyrill Stachniss, and Wolfram Burgard. Improving grid-based SLAM with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2432–2437. IEEE, 2005. 21

[91] Ryan M Eustice, Hanumant Singh, and John J Leonard. Exactly sparse delayed-state filters. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2417–2424. IEEE, 2005. 21

[92] Michael Montemerlo and Sebastian Thrun. FastSLAM 2.0. *FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics*, pages 63–90, 2007. 22

[93] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment - a modern synthesis. In *Vision algorithms: theory and practice*, pages 298–372. Springer, 2000. 22

[94] Hauke Strasdat. *Local accuracy and global consistency for efficient visual SLAM*. PhD thesis, Citeseer, 2012. 23, 57

[95] Georg Klein and David Murray. Parallel tracking and mapping for small AR workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007. 23, 24, 27, 30, 35, 116

[96] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008. 24

[97] Gabe Sibley, Larry Matthies, and Gaurav Sukhatme. Sliding window filter with application to planetary landing. *Journal of Field Robotics*, 27(5):587–608, 2010. 24

[98] Giorgio Grisetti, Rainer Kummerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010. 24

[99] Frank Dellaert and Michael Kaess. Square root SAM: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203, 2006. 24

[100] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011. 24

[101] Raul Mur-Artal and Juan D Tardos. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *arXiv preprint arXiv:1610.06475*, 2016. 25, 30

[102] Zheyuan Lu, Zhencheng Hu, and Keiichi Uchimura. SLAM estimation in dynamic outdoor environments: A review. In *International Conference on Intelligent Robotics and Applications*, pages 255–267. Springer, 2009. 25, 26

[103] Rod Frehlich and Robert Sharman. Maximum likelihood estimates of vortex parameters from simulated coherent Doppler lidar data. *Journal of Atmospheric and Oceanic Technology*, 22(2):117–130, 2005. 25

[104] Pierre Gauthier, Philippe Courtier, and Patrick Moll. Assimilation of simulated wind lidar data with a Kalman filter. *Monthly weather review*, 121(6):1803–1820, 1993. 25

[105] Michael Bosse, Paul Newman, John Leonard, and Seth Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *The International Journal of Robotics Research*, 23(12):1113–1139, 2004. 25

[106] Will Maddern, Alastair Harrison, and Paul Newman. Lost in translation (and rotation): Rapid extrinsic calibration for 2D and 3D LIDARs. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3096–3102. IEEE, 2012. 25

[107] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 26, 28

[108] Paul J Besl and Neil D McKay. Method for registration of 3-D shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*, volume 1611, pages 586–607. International Society for Optics and Photonics, 1992. 26, 28

[109] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001. 26, 28

[110] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2D range scans. *Journal of Intelligent and Robotic systems*, 18(3):249–275, 1997. 26

[111] Albert Diosi and Lindsay Kleeman. Laser scan matching in polar coordinates with application to SLAM. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3317–3322. IEEE, 2005. 26

[112] David M Cole and Paul M Newman. Using laser range data for 3d slam in outdoor environments. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1556–1563. IEEE, 2006. 26

[113] Sebastian Thrun and John J Leonard. Simultaneous localization and mapping. In *Springer handbook of robotics*, pages 871–889. Springer, 2008. 26

[114] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, MN Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, et al. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008. 26

[115] Paul Newman, David Cole, and Kin Ho. Outdoor slam using visual appearance and laser ranging. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1180–1187. IEEE, 2006. 26

[116] Dorit Borrmann, Jan Elseberg, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg. Globally consistent 3d mapping with scan matching. *Robotics and Autonomous Systems*, 56(2):130–142, 2008. 26

[117] Paul Newman, Gabe Sibley, Mike Smith, Mark Cummins, Alastair Harrison, Chris Mei, Ingmar Posner, Robbie Shade, Derik Schroeter, Liz Murphy, et al. Navigating, recognizing and describing urban spaces with vision and lasers. *The International Journal of Robotics Research*, 28(11-12):1406–1433, 2009. 26

[118] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor. Real-time loop closure in 2d lidar slam. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1271–1278. IEEE, 2016. 26

[119] Andreas Nüchter and Joachim Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11):915–926, 2008. 26

[120] Qiang Liu, Ruihao Li, Huosheng Hu, and Dongbing Gu. Extracting semantic information from visual data: A survey. *Robotics*, 5(1):8, 2016. 26

[121] Cipriano Galindo, Alessandro Saffiotti, Silvia Coradeschi, Pär Buschka, Juan-Antonio Fernandez-Madrigal, and Javier González. Multi-hierarchical semantic maps for mobile robotics. In *Intelligent Robots and Systems, 2005.(IROS*

2005). 2005 IEEE/RSJ International Conference on, pages 2278–2283. IEEE, 2005. 26

[122] Cipriano Galindo, Juan-Antonio Fernández-Madrigal, Javier González, and Alessandro Saffiotti. Robot task planning using semantic maps. *Robotics and autonomous systems*, 56(11):955–966, 2008. 26

[123] Silvia Coradeschi and Alessandro Saffiotti. An introduction to the anchoring problem. *Robotics and Autonomous Systems*, 43(2-3):85–96, 2003. 26

[124] Shrihari Vasudevan and Roland Siegwart. Bayesian space conceptualization and place classification for semantic maps in mobile robotics. *Robotics and Autonomous Systems*, 56(6):522–537, 2008. 27

[125] Óscar Martínez Mozos. *Semantic labeling of places with mobile robots*, volume 61. Springer, 2010. 27

[126] Andrzej Pronobis. *Semantic mapping with mobile robots*. PhD thesis, KTH Royal Institute of Technology, 2011. 27

[127] Frank Dellaert, David Bruemmer, and A Common Cognitive Workspace. Semantic slam for collaborative cognitive workspaces. In *AAAI Fall Symposium Series*, 2004. 27

[128] Steven Lovegrove, Andrew J Davison, and Javier Ibanez-Guzmán. Accurate visual odometry from a rear parking camera. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 788–793. IEEE, 2011. 27

[129] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, 2006. 27, 28

[130] Marc Pollefeys, Luc Van Gool, Maarten Vergauwen, Frank Verbiest, Kurt Cornelis, Jan Tops, and Reinhard Koch. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232, 2004.

[131] Marc Pollefeys, David Nistér, J-M Frahm, Amir Akbarzadeh, Philippos Mordohai, Brian Clipp, Chris Engels, David Gallup, S-J Kim, Paul Merrell, et al. Detailed real-time urban 3D reconstruction from video. *International Journal of Computer Vision*, 78(2):143–167, 2008. 27

[132] Davide Scaramuzza, Friedrich Fraundorfer, and Roland Siegwart. Real-time monocular visual odometry for on-road vehicles with 1-point RANSAC. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4293–4299. IEEE, 2009. 27

[133] Davide Scaramuzza. Performance evaluation of 1-point-RANSAC visual odometry. *Journal of Field Robotics*, 28(5):792–811, 2011. 27

[134] Henning Lategahn and Christoph Stiller. City GPS using stereo vision. In *Vehicular Electronics and Safety (ICVES), 2012 IEEE International Conference on*, pages 1–6. IEEE, 2012. 27, 28

[135] Bernd Manfred Kitt, Joern Rehder, Andrew D Chambers, Miriam Schonbein, Henning Lategahn, and Sanjiv Singh. Monocular visual odometry using a planar road model to solve scale ambiguity. 2011. 28

[136] Sebastian Hilsenbeck, Andreas Möller, Robert Huitl, Georg Schroth, Matthias Kranz, and Eckehard Steinbach. Scale-preserving long-term visual odometry for indoor navigation. In *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pages 1–10. IEEE, 2012. 28

[137] Damien Eynard, Pascal Vasseur, Cédric Demonceaux, and Vincent Frémont. UAV altitude estimation by mixed stereoscopic vision. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 646–651. IEEE, 2010. 28

[138] Joern Rehder, Kamal Gupta, Stephen Nuske, and Sanjiv Singh. Global pose estimation with limited GPS and long range visual odometry. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 627–633. IEEE, 2012. 28

[139] Edwin B Olson. Real-time correlative scan matching. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4387–4393. IEEE, 2009. 28

[140] Gian Diego Tipaldi and Kai O Arras. Flirt-interest regions for 2D range data. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3616–3622. IEEE, 2010. 28

[141] Jakob Engel, Jurgen Sturm, and Daniel Cremers. Semi-dense visual odometry for a monocular camera. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1449–1456. IEEE, 2013. 29, 30, 35

[142] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. REMODE: Probabilistic, Monocular Dense Reconstruction in Real Time. In *Proc. IEEE Int. Conf. on Robotics and Automation*, 2014. 29, 30

[143] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. DTAM: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011. 29, 30, 31, 32, 35

[144] Koichiro Deguchi. A direct interpretation of dynamic images with camera and object motions for vision guided robot control. *International Journal of Computer Vision*, 37(1):7–20, 2000. 31

[145] Shree K Nayar, Sameer A Nene, and Hiroshi Murase. Subspace methods for robot vision. *IEEE Transactions on Robotics and Automation*, 12(5):750–758, 1996. 31

[146] Christophe Collewet and Eric Marchand. Photometric visual servoing. *IEEE Transactions on Robotics*, 27(4):828–834, 2011. 31, 32

[147] Vinutha Kallem, Maneesh Dewan, John P Swensen, Gregory D Hager, and Noah J Cowan. Kernel-based visual servoing. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 1975–1980. IEEE, 2007. 31

[148] Paul Viola and William M Wells. Alignment by maximization of mutual information. In *Computer Vision, 1995. Proceedings., Fifth International Conference on*, pages 16–23. IEEE, 1995. 32, 33, 116

[149] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001. 32, 103

[150] Amaury Dame and Eric Marchand. Accurate real-time tracking using mutual information. In *Mixed and Augmented Reality (ISMAR), 2010 9th IEEE International Symposium on*, pages 47–56. IEEE, 2010. 32

[151] Josien PW Pluim, JB Antoine Maintz, and Max A Viergever. Mutual-information-based registration of medical images: a survey. *IEEE transactions on medical imaging*, 22(8):986–1004, 2003. 33

[152] David Mattes, David R Haynor, Hubert Vesselle, Thomas K Lewellyn, and William Eubank. Nonrigid multimodality image registration. In *Medical Imaging 2001*, pages 1609–1620. International Society for Optics and Photonics, 2001. 33

[153] Baofeng Guo, Steve R Gunn, RI Damper, and JDB Nelson. Band selection for hyperspectral image classification using mutual information. *IEEE Geoscience and Remote Sensing Letters*, 3(4):522–526, 2006. 33

[154] Paul Viola and William M Wells III. Alignment by maximization of mutual information. *International journal of computer vision*, 24(2):137–154, 1997. 33

[155] Nicholas Dowson and Richard Bowden. A unifying framework for mutual information methods for use in non-linear optimisation. In *Computer Vision–ECCV 2006*, pages 365–378. Springer, 2006. 33

[156] Stefano Soatto, Ruggero Frezza, and Pietro Perona. Motion estimation via dynamic vision. *Automatic Control, IEEE Transactions on*, 41(3):393–413, 1996. 34

[157] Richard J Prazenica, Adam Watkins, Andrew J Kurdila, Qi Fa Ke, and Takeo Kanade. Vision-based Kalman filtering for aircraft state estimation and structure from motion. In *AIAA Guidance, Navigation, and Control Conference*, volume 5, page 3, 2005.

[158] Stefano Soatto and Pietro Perona. Recursive 3D visual motion estimation using subspace constraints. *International Journal of Computer Vision*, 22(3):235–259, 1997. 34

[159] Peter Lang and Axel Pinz. Calibration of hybrid vision/inertial tracking systems. In *Proceedings of the 2nd InerVis: Workshop on Integration of Vision and Inertial Sensors*. Stanford University, 2005. 34

[160] Jorge Lobo and Jorge Dias. Relative pose calibration between visual and inertial sensors. *The International Journal of Robotics Research*, 26(6):561–575, 2007. 34

[161] Gabriel Nützi, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Fusion of IMU and vision for absolute scale estimation in monocular SLAM. *Journal of intelligent & robotic systems*, 61(1):287–299, 2011. 34

[162] Henrik Rehbinder and Bijoy K Ghosh. Pose estimation using line-based dynamic vision and inertial sensors. *Automatic Control, IEEE Transactions on*, 48(2):186–199, 2003. 34

[163] Peter Gemeiner, Peter Einramhof, and Markus Vincze. Simultaneous motion and structure estimation by fusion of inertial and vision data. *The International Journal of Robotics Research*, 26(6):591–605, 2007. 34

[164] Jonathan Kelly and Gaurav S Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The International Journal of Robotics Research*, 30(1):56–79, 2011. 35, 36

[165] Mingyang Li and Anastasios I Mourikis. Improving the accuracy of EKF-based visual-inertial odometry. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 828–835. IEEE, 2012. 35

[166] Stephan Weiss, Markus W Achtelik, Simon Lynen, Michael C Achtelik, Laurent Kneip, Margarita Chli, and Roland Siegwart. Monocular Vision for Long-term Micro Aerial Vehicle State Estimation: A Compendium. *Journal of Field Robotics*, 30(5):803–831, 2013. 35

[167] Shaojie Shen, Yash Mulgaonkar, Nathan Michael, and Vijay Kumar. Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft mav. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4974–4981. IEEE, 2014. 35

[168] Roland Brockers, Sara Susca, David Zhu, and Larry Matthies. Fully self-contained vision-aided navigation and landing of a micro air vehicle independent from external sensor inputs. In *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics, 2012. 35

[169] Sen Wang, Ling Chen, Dongbing Gu, and Huosheng Hu. An optimization based moving horizon estimation with application to localization of autonomous underwater vehicles. *Robotics and Autonomous Systems*, 2014. 35

[170] Simon Baker and Iain Matthews. Lucas-Kanade 20 years on: A unifying framework. *International journal of computer vision*, 56(3):221–255, 2004. 36

[171] Johann Heinrich Lambert. *Photometria sive de mensura et gradibus luminis, colorum et umbrae*. Klett, 1760. 37

[172] Yi Ma. *An invitation to 3-D vision: from images to geometric models*, volume 26. Springer, 2004. 42

[173] Bradley M Bell and Frederick W Cathey. The iterated Kalman filter update as a Gauss-Newton method. *IEEE Transactions on Automatic Control*, 38(2):294–297, 1993. 81

[174] Robert Hermann and Arthur J Krener. Nonlinear controllability and observability. *IEEE Transactions on automatic control*, 22(5):728–740, 1977. 83

[175] Javier Civera, Diana R Bueno, Andrew J Davison, and JMM Montiel. Camera self-calibration for sequential bayesian structure from motion. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 403–408. IEEE, 2009. 84

[176] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 84

[177] Lee E Clement, Valentin Peretroukhin, Jacob Lambert, and Jonathan Kelly. The battle for filter supremacy: a comparative study of the multi-state constraint Kalman filter and the sliding window filter. In *Computer and Robot Vision (CRV), 2015 12th Conference on*, pages 23–30. IEEE, 2015. 84, 88

[178] Frederik Maes, Andre Collignon, Dirk Vandermeulen, Guy Marchal, and Paul Suetens. Multimodality image registration by maximization of mutual information. *IEEE transactions on medical imaging*, 16(2):187–198, 1997. 116

[179] Imad Zoghlami, Olivier Faugeras, and Rachid Deriche. Using geometric corners to build a 2D mosaic from a set of images. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 420–425. IEEE, 1997. 127

[180] David Capel and Andrew Zisserman. Automated mosaicing with super-resolution zoom. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 885–891. IEEE, 1998. 127

[181] Heung-Yeung Shum and Richard Szeliski. Construction and refinement of panoramic mosaics with global and local alignment. In *Computer Vision, 1998. Sixth International Conference on*, pages 953–956. IEEE, 1998. 127

[182] Richard Szeliski and Heung-Yeung Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 251–258. ACM Press/Addison-Wesley Publishing Co., 1997. 127