# Real-time Visual-Inertial SLAM for Multirotor Platform

Usman Qayyum

A thesis submitted in fulfillment
of the requirements for the degree of
Doctor of Philosophy

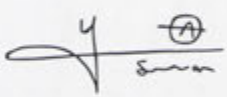
Australian
National
University

**Research School of Engineering**
**The Australian National University**

07, September 2015

# Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the University or other institute of higher learning, except where due acknowledgement has been made in the text.



**Usman Qayyum**

September 07, 2015

# Abstract

Usman Qayyum
The Australian National University

Doctor of Philosophy
September 2015

# Real-time Visual-Inertial SLAM for Multirotor Platforms

This thesis presents the theoretical and practical development of a framework that enables small-scale aerial vehicles to autonomously explore and map unknown environments. A crucial point to achieve such capability is to construct a map, partially or fully, of GPS-denied environments and to determine its location relative to the built map, thus called Simultaneous Localisation and Mapping (SLAM).

The increased demands of all-terrain navigational capability and full 6 Degrees-of-Freedom (DOF) solutions have driven the usage of Inertial Measurement Unit (IMU) and related Inertial Navigation System (INS), which algorithmically computes the position and orientation (or pose) of the robotic platform. Unfortunately, the INS is a highly unstable system, containing three integrator loops coupled with the presence of gravity, causing errors to accumulate cubically as time increases. To constrain these errors effectively, a reliable and constant stream of aiding signals is crucial for the success of inertial applications.

Therefore to utilise the aiding information at higher rates (for inertial system stability), this work proposes a novel visual non-holonomic constraint. These relative pose constraints are estimated from the consecutive monocular image sequence and capable of providing the ego-motion estimates at frame-rate to inertial sensor. The basic concept is the tangential velocity components of INS along the direction of visual-flow motion should be zero, that is similar to the non-holonomic constraint (or non-skidding assumption) in the wheeled robots [1, 2]. We mainly address two key challenges in achieving robust and efficient performance: a) the computational complexity issue in the monocular feature-based SLAM [3] and related scale-ambiguity problem [4] [1] b) the *depth dropout* problem in RGB-D (Kinect or Stereo) sensors [5, 6]. The contributions of this thesis are as follows.

Firstly, the monocular SLAM using the concept of *Visual-Inertial non-holonomic constraint* is investigated and analysed. The monocular SLAM generally estimates the pose of the vehicle and landmarks jointly. The fundamental problem with these

[1]The fusion of visual translation from monocular vision (scale ambiguous measurements) with the inertial sensor (metric measurements).

approaches is that the computational requirements scale quadratically in the number of landmarks. Therefore, this limits the number of visual features to only a few hundred (hence a compromise on the available information to constrain the vehicle's pose) and small-scale environment. We fused the visual non-holonomic constraints (using hundreds of features available from the consecutive images) at high update rates with an inertial sensor for large scale environments. In this way, by using these constraints the on-board IMU/INS can be calibrated and corrected accurately without maintaining a large number of features on the map.

Secondly, a depth dropout problem in RGB-D (Kinect or Stereo) SLAM is addressed and researched. Depth data from the RGB-D or stereo sensors [5, 6] can be easily unavailable or partially available in many outdoor environments due to limited range or sunlight interference. In the case of aerial vehicles, which at times have enough clearance from the environment, the 3D sensors would act virtually as monocular vision. The unavailability of reliable depth data makes the existing RGB-D approaches inapplicable for many outdoor environments. We resolved the depth dropout issue of the RGB-D sensor by providing either full 6DOF (rotation and translation RGB-D constraints) or partial 5DOF visual non-holonomic constraints (rotation and scale-ambiguous translation) to the inertial sensor. A computationally efficient map management method is also developed by utilising a loosely-coupled Visual-Inertial integration framework.

Thirdly, the convexity structure of an Inertial-SLAM was investigated by analysing the number of local minima, which is important to understand the pseudo-optimality of the SLAM system. This sheds light on solving the non-linear SLAM problem without resorting to the exhaustive search techniques in global optimisation (which are typically computationally demanding). We analysed the number of local minima of a two-pose 5DOF Inertial SLAM system by utilising stationary point analysis and hence proved that, when one of the orientation angles is assumed known, there exist a maximum of four local minima in the system.

Lastly, the proposed methods are implemented in an embedded computer on the hexarotor platform and its real-time performance is demonstrated in challenging outdoor environments. A customised, low-cost IMU is developed in-house, which is calibrated statically in the laboratory and dynamically during the flight. The real-time on-board implementation of the proposed framework is performed with modularity and robustness in mind. A demonstration of the developed framework is evaluated with hovering control. For the purpose of demonstration, an attitude and position controller is also developed through an open-source autopilot system. The accuracy and performance are demonstrated against the precise ground truth data (from VICON [2]) and benchmark datasets.

[2]The VICON motion capture system is an infrared marker-tracking system that offers sub-degree and millimetre level accuracy.

# Acknowledgements

I would like to start by thanking my supervisor Jonghyuk Kim for his invaluable supervision, assistance and mentorship throughout my PhD studies. I have learned much from him about how to do academic research. I had a wonderful learning experience and enjoyed all the research discussions with him.

I am grateful for the financial support received during my studies from the following sources:

- Research Supported by ARC DP Project
- ANU-PhD Scholarship, Supplementary Scholarship and Tuition waiver,

My committee panel have all been exemplary. I would like to thank my co-advisors Hongdong Li and Robert Mahony for their feedback and providing me with an opportunity to do lecturing/tutoring at ANU. Big thanks also go to Alex Martin and Erasmo Scipion for their technical help. I had a great time co-supervising the undergraduate students Dafizal Derwai, Josiah Khor, Thavidu Ranatunga, Katrina Kelleher) and graduate students (Thomas Eagen, Veenat Sahu). They all have done great work and contributed to the field of robotics with their enormous effort. I would also like to thank my friends for their good company and support, especially Khurram Aftab and Zubair Khalid. All of them made me feel at home away from home. Finally, I would like to thank my parents, brother and sisters for their unconditional support and encouragement. Special thanks go to my wife for her understanding and love. Without all of them, I would never have been able to finish my thesis. Last but not least my sweet little angel, who has to wait for almost one year to meet with her papa. Love you Saleha Fatima.

*"Knowledge gives life to the soul"*
***Ali**, RA*, 600-661 *CE*

# Publications and Code Contribution

This thesis includes work contained in the following academic papers

- Usman, Q and Kim, J., Global Optimisation for 2D SLAM Problem LNEE Springer Book Chapter, 2014.
- Usman, Q. and Kim, J., Real-time 6DOF Aerial SLAM with 2D-3D Visual Features, Journal for Field Robotics, 2014 (Submitted).
- Usman, Q. and Kim, J., Global optimal and low minima analysis for Visual-Inertial SLAM problem, IEEE Transaction on Aerospace and Electronic Systems 2014 (Submitted).
- Usman, Q and Kim, J., Analysis on the Number of Local Minima for Inertial-SLAM Problem, In Proceeding of IEEE International Conference on Robotics and Biomimetics, 2013.
- Usman, Q and Kim, J., Inertial-Kinect Fusion for Outdoor 3D Navigation, Australasian Conference on Robotics and Automation,2013.
- Usman, Q and Kim, J., Graceful Degradation of IMU-Kinect Fusion for Outdoor Aerial Navigation and Mapping, International Conference on Robotics and Automation 2015, (Submitted).
- Usman, Q. and Kim, J., Seamless Aiding of Inertial-SLAM using Visual Directional Constraints from a Monocular Vision IEEE/RSJ International Conference on Intelligent Robots and Systems (2012)
- Usman, Q. and Kim, J., Global Optimal Solution to SLAM problem for unknown initial estimates,Regular paper (11.0% acceptance rate) ICINCO 2012 (**Nominated for Best Paper Award**)
- Usman, Q and Kim, J., Visual-Odometry Integrated Inertial-SLAM, International Global Navigation Satellite Systems Society, 2011.

- Usman, Q and Kim, J., Accelerometer aided Visual Priors for Robust Monocular SLAM, 12 International Conference Towards Autonomous Robotics, 2011

The open-source code contributions are:

- PID Position Controller [3]
- Autopilot Wrapper Functions for Controller: [4]
- Real-time Point Cloud Generation: [5]

Furthermore, some of our work published (during the PhD period) but not forming part of the thesis.

- Usman, Q and Kim, J., Omni Visual-Laser 3D Scanner for Outdoor Mapping, In Proceeding of IEEE International Conference on Robotics and Biomimetics, 2013.
- Usman, Q, Kim, J., Martin, A. and Shim, D., Omni-VISER: 3D Omni Vision-Laser Scanner, Australasian Conference on Robotics and Automation, 2013.
- Usman, Q, Notch based Face Detection and Facial Feature Localisation: A Real-time Approach to Face Detection and Tracking, Book, 2012. [6]

[3] usmanqayyum.blogspot.com.au/2013/08/pid-position-controller-for-arducopter.html

[4] https://docs.google.com/document/d/1FBzhlDKVSiE9XYZKUo4KbixbVH_-_qFTXDAr4zebAQBY/edit?usp=sharing

[5] https://docs.google.com/open?id=\\0B8gYxW2umHiVLUVPQmxrdVpWRzA

[6] http://www.amazon.com/Notch-Detection-Facial-Feature-Localization/dp/3848429357a

# List of Acronyms

**3D** 2-Dimensional

**3D** 3-Dimensional

**APM** Ardupilot Mega

**ADC** Analog-to-Digital Converter

**BA** Bundle Adjustment

**DCM** Direction Cosine Matrix

**DOF** Degree of Freedom

**EKF** Extended Kalman Filter

**FLANN** Fast Approximate Nearest Neighbour Search Library

**GPS** Global Positioning System

**GNSS** Global Navigation Satellite System

**iSAM** Incremental Smoothing and Mapping

**IMU** Inertial Measurement Unit

**ISP** In-System Programmer

**KF** Kalman Filter

**MAV** Micro Aerial Vehicle

**Mission-Planner** A ground control interface for the autopilot

**MAVlink** Micro Aerial Vehicle Communication Protocol

**MEMS** Micro Electrical Mechanical System

**NLLS** Non-Linear Least Square

**PID** Proportional Integral Derivative

**PTAM** Parallel Tracking and Mapping

**PCB** Printed Circuit Board

**R/C** Remote Control

**RGB-D sensor** Red-Green-Blue-Depth 3D sensor

**RANSAC** RANdom SAmpling Consensus

**ROS** Robot Operating System

**SIFT** Scale Invariant Feature Transform

**SFM** Structure From Motion

**SURF** Speeded Up Robust Feature

**SLAM** Simultaneous Localisation and Mapping

**SVD** Singular Value Decomposition

**SPI** Serial Peripheral Interface

**UART** Universal Asynchronous Receiver Transmitter

**UAV** Unmanned Aerial Vehicle

**vSLAM** Visual Simultaneous Localisation and Mapping

**VO** Visual Odometry

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Autonomous localisation and navigation has drawn significant interest from the mapping and robotics community, as it enables the robotic vehicles to be deployed in a fully autonomous way for a wide variety of robotic missions such as environmental monitoring and exploration [7, 8, 9, 10]. Autonomy is an important property of the robotic vehicle as it reduces the requirement of human attention in the control of the vehicle. One of the key aspects of autonomous vehicles is the ability to localise themselves. For localisation, the knowledge of their position with respect to the environment is necessary. It's not always possible to have prior knowledge about the environment. A crucial element of achieving robust navigation for these autonomous vehicles is the construction of a map of the environment, and the simultaneous determination of the vehicle's location in the build map (SLAM).

The future of reconnaissance in cluttered and constrained environments has made the use of aerial vehicles important in robotics applications [4, 11, 12]. Consequently, the autonomous operation of aerial vehicles has been extensively investigated. Given SLAM based navigational capability, a flying vehicle can perform such tasks as patrolling, search and rescue, package delivery, and environmental or bush fire monitoring in unknown environments [13]. The aerial vehicles offer a huge advantage over traditional ground vehicles in terms of deployment to inaccessible environments; for example, in search and rescue missions the rubble can confine the areas that are

accessible to ground vehicles while aerial exploration is still possible.

In recent years, a palette of approaches for aerial vehicle based SLAM has been introduced; however many implementations still rely on indoor motion capture system [14] or use downward looking cameras to do SLAM in outdoor environments [12] [1] or map based delayed filter approaches [8] [2]. In this thesis, the primary focus is the investigation of precise and efficient autonomous localisation that enables the robotic aerial vehicle to navigate in challenging outdoor environments. We particularly focus on the use of forward-looking vision sensors to provide the cues to aid and calibrate a low-cost inertial sensor for the localisation, mapping and control of the aerial vehicle[3]. In addition, we have designed and developed an in-house inertial sensor (which costs less than $50) and we assembled it along with its payload. All the processing is performed in real-time on-board with a closed-loop controller (partially through an autopilot) enabling the autonomous hovering of an aerial vehicle. Furthermore, we have analysed the highly non-linear structure of the 3D SLAM system and provided an upper bound on the number of local minima along with a global optimal optimisation approach.

In the remaining chapter, we will discuss the existing navigation algorithms for aerial vehicles, followed by the motivation and objective section. Lastly, an overview of the contributions will be presented, followed by the details of the thesis structure.

## 1.1 Autonomous Navigation for Aerial Vehicles

Recently the availability of inexpensive small size platforms has pushed the research thrust into small-scale aerial vehicles. Multirotor vehicles have become an extremely

[1] These approaches rely on a strong assumption that the depth of the scene changes smoothly; however this assumption is not valid for the front-looking camera [15, 16, 17]. Another important point to note for these approaches [12, 18] is that they require an excitation motion at the start of filter convergence.

[2] 3D map generation is intrinsically a delayed process, as resolving the 3D depth requires enough parallax across multiple views, thus reducing the aiding availability.

[3] We provide a vision based seamless approach to aid the low-cost inertial sensor at higher update rates, discussed in Chapter 3.

Figure 1.1: Aerial platforms (a helicopter and a hexacopter) developed in-house during this thesis, the helicopter is equipped with a PC104 and a Stereo Camera and an autopilot system. The hexacopter contains a Kinect and an in-house IMU as a payload with a dual-core Atom board.

popular platform due to their increased agility and manoeuvring capability in contrast to their fixed-wing counterparts (which have the compensating advantage of being able to fly for extended periods of time). They are ideal for confined environments due to their hovering mechanism, where they can observe the environment from multiple vantage points. In recent years, there have been various examples in which multirotor aerial vehicles have been used for critical missions/applications; for example, the inspection of the Fukushima nuclear reactor [19]. Figure 1.1 shows the in-house developed multirotor platforms (a helicopter and a hexacopter system) developed during this thesis. The hexacopter system is mechanically much simpler than the helicopter due to its smaller diameter and fixed pitch of the rotors. Moreover, the hexacopter platform can be enclosed by a safety protection ring and its rotors store less kinetic energy, which makes it a much safer platform than helicopters. In this thesis, we have used the hexacopter platform to demonstrate the developed SLAM framework for autonomous hovering control. Figure 1.2 shows the estimated trajectory and map of the environment build using our SLAM framework on a hexacopter.

In the literature, most of the proposed SLAM algorithms were developed and tested on 2D robotic vehicles with odometry estimates (for motion estimates) and other extero-ceptive aiding sensors. However, precise odometry information is not available in the case of aerial vehicles, as they operate in 3D space. The increased demands

(a) Image from the sequence of outdoor aerial flight data

(b) Estimated trajectory and map of the environment

Figure 1.2: Estimated trajectory and map build using the proposed approach using a hexacopter platform.

of 6DOF solutions have driven the usage of an Inertial Measurement Unit (IMU) [1, 20, 21] and the related Inertial Navigation System (INS) which algorithmically computes the position and orientation of the robotic platform at high data rates. Unfortunately, the INS is a highly unstable system containing three integrator loops coupled with the presence of gravity, causing the errors to accumulate cubically as time increases. To constrain these errors effectively, a reliable and constant stream of an aiding signal is crucial for the success of inertial applications. As an INS error can grow beyond the acceptable bounds before receiving the next aiding information, potentially destabilising the inertial system [1].

Therefore, a multiple sensor fusion approach with inertial sensors is required using various sensors such as laser range finders, cameras, sonar, radar and a Global Positioning System (GPS) [11, 17, 22, 23]. The integrated approaches with inertial sensors for aerial navigation are therefore explained further in the upcoming subsections.

### 1.1.1 GPS Integrated Approaches

Almost 20 years ago, Conway [24] introduced the GPS integrated approach for autonomous control of a small-scale aerial platform. Afterwards, there was extensive

research work on the INS and GNSS [4] integration for dynamic pose estimation of the autonomous vehicle [25, 26, 27, 28]. These approaches are suitable for long-term autonomy. However, the dependency on external signals which can be jammed by intentional interference or limited by urban canyons or indoor environments (GPS-denied environments) is a potential drawback.

### 1.1.2 Motion Capture System based Aiding

To simulate the GPS-like measurements in the indoor environment, various studies [14, 29, 30, 31] have used the motion capture system for aerial vehicle navigation. Motion capture systems such as VICON [29] help the flying vehicle to operate autonomously by performing the task of localisation (with centimetre accuracy). The VICON system consists of multiple infrared cameras and it requires that some markers with reflecting infrared tags are mounted on the vehicle (to estimate the pose of the aerial vehicle). There are various applications in which the VICON system was used for highly precise and fast flight manoeuvres (such as the work of Kumar et al. [29] for aggressive manoeuvres). In recent years, the motion capture system have been used for applications such as cooperatively grasping and transporting as well as swarm formations of small-scale aerial vehicles [14, 30, 31].

The motion capture system has also been used for evaluation purposes (as ground truth data) [32] but their applicability to large-scale areas and outdoor environments is very limited. One important aspect of the pose estimation from the VICON system is that it requires tedious work of multiple-camera calibration, which becomes more laborious as the number of cameras increases [29].

### 1.1.3 Vision and Laser Based Integrated Approaches

Cameras and laser scanners are two well-known extero-ceptive sensors which have been integrated with inertial sensors to provide autonomous navigational capability

[4]Global Navigation Satellite System (GNSS) is the satellite navigation system with global coverage (such as GPS, GLONASS, Galileo and Beidou positing systems).

to robotic vehicles [7, 33, 34, 35]. These sensors relieve the dependence on the external infrastructure for navigation and control (such as VICON cameras or GPS satellites).

#### 1.1.3.1 Laser based Aiding

An alternative approach to the VICON and GPS system for localisation are laser-scanner based SLAM algorithms [34]. The miniature form-factor of 2D laser sensors enables its use for flying vehicles while 3D scanners (like Velodyne [36]) are still bulky [37]. Recently with $2D$ the Hokuyo laser scanner, in the work of [34, 37] in an unstructured indoor environment, had performed 2D SLAM on an inertial-stabilised aerial platform. While laser scanners are popular with the robotics community, they can only measure range values along a plane, making them more appropriate for structural environments.

#### 1.1.3.2 Computer Vision based Aiding

The visual sensing modality, in particular, has drawn great attention due to its passiveness, light-weight and rich information content, coupled with the advances in embedded computing technology. Over the last decade, a lot of attention has been focused on camera based SLAM solutions [35, 38]. The integration of inertial sensor data with visual information curtails various limitations [5] in vision-only systems, hence making both the sensors complimentary to each other. IMU observations provide high-frequency information corrupted with low-frequency noise, helping to predict the high dynamics of the platform (or camera). The visual sensor compliments this data by providing low-frequency information with high-frequency noise for drift correction. Traditionally, vision-inertial integrated approaches can be categorised into artificial beacon/markers [39, 40, 41] and natural feature based approaches [42, 43, 44].

In the first category, artificial markers [40, 41] are used for the autonomous navigation of the aerial vehicle. The earlier work of Kim et al. [41] used the range and bearing

[5] The limitations are fast feature tracking, handling the low parallax of features for depth estimation (monocular camera) and low output data-rates for pose estimation.

information of artificial markers for the Inertial SLAM (where the problem of re-localisation was made simpler by knowing the relative 3D position of the markers hence minimising the inertial error). In another work [40], the 5DOF pose estimation of the aerial vehicle was performed using two concentric circles. The approaches in this category required a-priori knowledge about the environment or needed to intervene with the environment (hence making their use limited).

In the second category, the environment is completely unknown and natural geometric features are used for autonomous navigation. The recent work in monocular integrated based approach is from Engel et al. [43]. They used the popularly used SLAM algorithm (PTAM [42] in which the camera information was fused with the IMU data for multirotor pose estimation along with altimeter data for metric map estimation. They demonstrated that their approach was robust to the temporary loss of visual tracking (due to the incorporation of inertial data) and did not require any external sensors or markers. However it required external processing of the video information [6] and therefore it was not a self-contained system. Another notable work on the monocular integrated approach was from Weiss et al. [44], using PTAM to autonomously control and navigate the vehicle. They performed on-board estimations of vehicle state and used those estimates to control the aerial multirotor platform. An Extended Kalman Filter (EKF) was used to estimate the pose of the vehicle along with the scale factor to determine the metric information from the monocular camera.

Various authors have used stereo cameras for the natural feature based integrated approach [6, 45, 46, 47, 48]. In the work of Kelly et al. [48], they used the stereo imagery to fuse the pose estimates with inertial measurements for a small-scale helicopter using EKF. The processing was done off-board on the acquired flight data so the vehicle is not fully autonomous. Recently the Kinect sensor has offered real-time 3D sensing capability, which is crucial for high-speed, manoeuvrable vehicles operating in 3D environments. The real-time accessibility of three-dimensional point clouds from the Kinect has been used for navigation, path planning and collision avoidance [5, 32]. The work of Huang et al. [49] used the Kinect sensor to enable 3D the flight

[6]The video was transmitted to a ground based computer for processing and the final control commands were sent back to the aerial vehicle

of a quad-rotor vehicle using only on-board sensor data. An EKF filter fusing visual and IMU measurements was used to determine the real-time position and velocity estimates, which were used by a PID position controller to maintain the stable flight.

## 1.2 Research Challenges

The existing navigation algorithms [7] that have been a huge success in ground based vehicles are not easily transferable to the aerial vehicles due to the unstable nature of an aerial platform working in a 6DOF environment. As with ground-based vehicles, the zero velocity command means that the aerial vehicle will decelerate until it comes to a complete stop. However the aerial vehicle cannot smoothly stop and, depending upon the uncertainty in the state estimate, it may end up oscillating instead (requiring continuous counteracting movements commands to hover rather than stopping completely, hence making the navigation algorithms more complex).

It is vital to consider the operational and theoretical challenges to come up with a robust and efficient 3D SLAM solution. The situational/theoretical constraints are mentioned in this section, considering a real-time SLAM algorithm that can be used for autonomous control of small-scale aerial vehicles for both indoor and outdoor environments.

### 1.2.1 Environmental Constraints: GPS-denied and Unstructured Environment

As stated earlier, GPS based navigation has been widely used to aid inertial measurements in various civilian/military applications. However, when the aerial vehicles have to be operated in a dense and close proximity environment (urban canyons), the utility of conventional GPS is limited [20] (and unavailable in indoor environments). In addition to the GPS-denied scenario, the environment of interest is mostly unstructured (such as an outdoor forest environment). Hence, it is highly desirable to

have a navigational approach that can aid the inertial sensor (to minimise the inertial drift). However, the GPS/laser integrated approaches discussed earlier fall short of satisfactory performance in a cluttered and unstructured environment.

These environmental constraints inspire us to look into the *Visual-Inertial integrated approaches*, which are the cornerstone of autonomous navigation. However, there are still various unresolved challenges, ranging from software (Section 1.2.3) and hardware (Section 1.2.2) to pure theoretical problems (Section 1.2.4).

### 1.2.2 Low-Cost Aerial Vehicle and Payload System

One of the challenges is to design and build the hardware structure of a small-scale aerial vehicle with off-the-shelf hobbyist grade components. Recently, low-cost aerial platforms such as AR. Drone [50]) have become readily available, but the problem with them is that they are incapable of carrying a payload system for on-board processing to achieve fully autonomous navigation. In addition to the requirement of a customised aerial platform, another important factor is to build a low-cost and lightweight payload system. Generally, ground vehicles don't have payload limitations; however, a lightweight payload is an important consideration for the aerial platform to remain airborne for a sustained period of time. The low cost of the payload system can be achieved by using a MEMS based inertial sensor along with an easily accessible low-cost vision sensor (such as Microsoft's Kinect sensor). Hence, our motivation is to build a low-cost in-house multirotor platform with a lightweight payload system.

### 1.2.3 Computational Efficient On-board Approach

SLAM algorithms have presented many computational problems in the recent literature on autonomous control of robotic vehicles, especially aerial vehicles [7]. Aerial platforms, due to their highly dynamic behaviour, require real-time algorithms for autonomous stable flight. Various studies [29, 43, 48] has offloaded the computation to

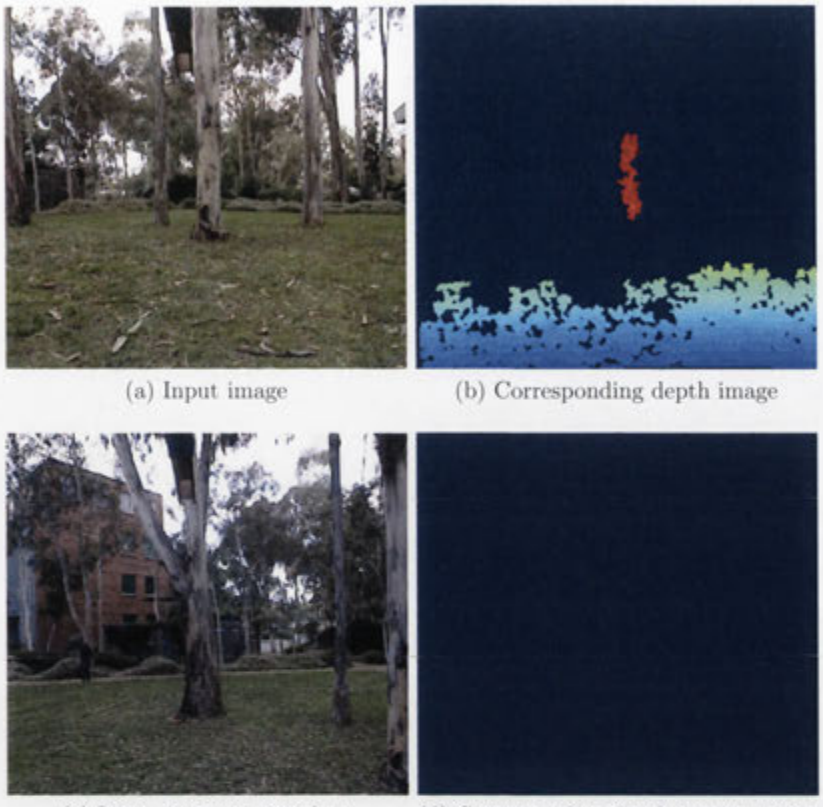
powerful ground stations by transmitting the sensor data through a wireless medium ( doing off-board processing) or using an external tracking system (VICON) for control purposes, as discussed earlier (not fully autonomous approaches). A data or remote-control communication link requires a stable connection (line-of-sight communication) for the continuous control of the aerial vehicle. However, the communication is subject to distortion and it's sometimes impossible to fit the environment with a camera tracking system. We are motivated by this constraint to implement real-time on-board computation for the fully autonomous behaviour of the aerial vehicle using a Visual-Inertial computationally efficient approach.

### 1.2.4 Seamless Integration of Visual Inertial System

By fusing vision and inertial sensors with an on-line generated map, Visual-Inertial SLAM has enabled full 6-DOF navigation without relying on external navigational aids [4, 40]. Although successful, Visual Inertial-SLAM still faces several challenges in achieving stable and robust performance, which largely stem from the limited availability and reliability of visual aiding [38].

The monocular camera has been extensively studied for navigational and mapping purposes by many researchers [42, 44]. One of the challenges with the monocular based approaches is that the environment scale cannot be determined in metric form because the monocular camera observes the 2D projection of the physical world. Therefore, sensor fusion approaches have to either work with scale ambiguous information [44, 51], estimate the scale alongside the vehicle's pose/map information [4] or use a-priori calibration objects in the environment [40, 41]. If the inertial sensor is of low quality, as in the case of this work, the estimated scale can diverge quite quickly, potentially destabilising the system [52]. The challenges faced in this thesis for monocular based Inertial aiding are:

- *Inconsistency*: When a feature is first observed in the vision sensor, its unknown depth should be properly resolved to aid the inertial system[4, 44]. 3D map



(a) Input image

(b) Corresponding depth image

(c) Input image at time k+1

(d) Corresponding depth image at time k+1, demonstrating depth dropout

Figure 1.3: Flight data of an aerial vehicle showing a visual/depth images from the RGB-D sensor. The aerial vehicle is flying high above the ground and far from the objects resulting in the unavailability of depth data (d).

generation is intrinsically a delayed process as resolving the 3D depth requires enough parallax across multiple views, thus reducing the aiding availability. Existing monocular based Inertial SLAM approaches [4, 44] have computational benefits, but the rate of visual aiding becomes highly environment dependent. Coupled with the low-grade sensors, the INS error can grow beyond the acceptable bounds before receiving the next update from the visual sensor, potentially destabilising the inertial system.

- *Non-optimal usage*: Incremental visual motion (also called visual odometry)

[21, 23] [7], can provide highly valuable vehicle motion information to inertial systems but has not been fully considered due to the scale ambiguity problem. That is, the translation output from the visual odometry has an arbitrary scale while the inertial output is represented in a metric space, thus hampering direct integration.

The projective nature and the lack of scale of the environment make it very challenging to avoid scale drift [54] over longer trajectories. In contrast, 3D sensing approaches (stereo vision or Kinect) avoid the arbitrary scale assumption or drift problems. In recent years, vision based 3D sensing devices (such as Kinect or stereo camera) have been very successful for indoor applications but there are still many challenges for them to operate in 3D outdoor environments.

- *Partial or no depth information*: Depth data from these sensors is unavailable or only partially available for many outdoor environments due to limited range or sunlight interference. In the case of aerial vehicles, which at times have enough clearance from the environment, the 3D sensing sensor would act virtually as a monocular camera, which causes a depth drop-out problem, thus further degrading the performance. The unavailability of depth data makes the existing Stereo/RGB-D approaches inapplicable for outdoor environments [4, 5, 32]. Figure 1.3 show visual/depth images from the Kinect sensor, showing the aerial vehicle flying high above the ground and far from the objects (Figure 1.3 (d)), depicting the unavailability of the depth information (depth drop out case).

Hence, the motivation is to look into the robust visual approaches that can use monocular/stereo cues to seamlessly aid the inertial sensor in an optimal way.

[7] The two frequent terms used in this thesis are SLAM and Visual Odometry (VO). SLAM based approaches [17, 35] provide drift free tracking relative to a persistent map with the capability of place recognition. In contrast to SLAM, VO approaches can track hundreds of visual features in consecutive images without considering a persistent or global map [35, 47, 53].

### 1.2.5 Optimality Analysis of the 3D SLAM

In the context of the 3D SLAM problem which is a highly complex non-linear estimation problem, it is vital to understand the hidden theoretical structure (in regard to the optimality of the solution). Therefore, understanding the convex structure of the 3D SLAM problem is of much interest to robotics community. SLAM is a high dimensional estimation problem, which means that it should be expected to have a lot of local minima. Earlier work has shown that if the initial guess provided to optimisation based SLAM is bad, then the algorithm generally results in a divergence of the solution [55, 56].

The maximum number of local minima remains a vital question when devising the algorithms to find a global optimal solution. This issue motivates us to analyse the number of local minima in the 3D SLAM problem and formulate a methodology to solve the SLAM problem optimally.

## 1.3 Contributions

In this thesis, we will present a real-time on-board Visual-Inertial estimation framework for the localisation, mapping and control of the multirotor platform in a $3D$ GPS-denied environment. The core of this thesis considers the seamless integration of visual and inertial sensors to obtain the precise motion estimates using monocular and stereo cues. We have demonstrated a real-time developed framework in an outdoor environment and validated the accuracy against precise ground truth data (VICON). Figure 1.4 shows the hardware developed in-house (multirotor platform, IMU, 3D vision-laser scanner) during the course of this thesis. Furthermore, we have also investigated and formed a theoretical foundation for 3D SLAM by analysing the structure and formulating an optimal methodology to solve the SLAM problem. Figure 1.5 shows the overall framework of our contribution along with the associated chapter. We summarise the key contribution of our work as follows:


(b) Inertial Measurement Unit


(c) 3D Laser-Vision system

Figure 1.4: In-house developed hardware for SLAM applications. Please note that for the 3D Laser-Vision system, the interested readers are refereed to our earlier work [57].

- Seamless visual aiding using a concept of non-holonomic constraints. The vision node for non-holonomic constraints is developed which can provide partial 5DOF (rotation and scale-ambiguous translation) vehicle pose measurements. The visual translation in 5DOF mode is treated as a *directional constraint* of the motion rather than estimating the depth from inertial output. Consequently, the vision output can be seamlessly fused with the low-cost inertial system without involving the delay to resolve the feature depth [11] or overloading the estimator with the arbitrary scale factor estimation [8]. With these constraints, we address the depth dropout problem by proposing a novel Visual-Inertial fu-

[8] A thorough comparison for Visual-Inertial seamless integration and the depth dropout case will be provided in Chapter 4.

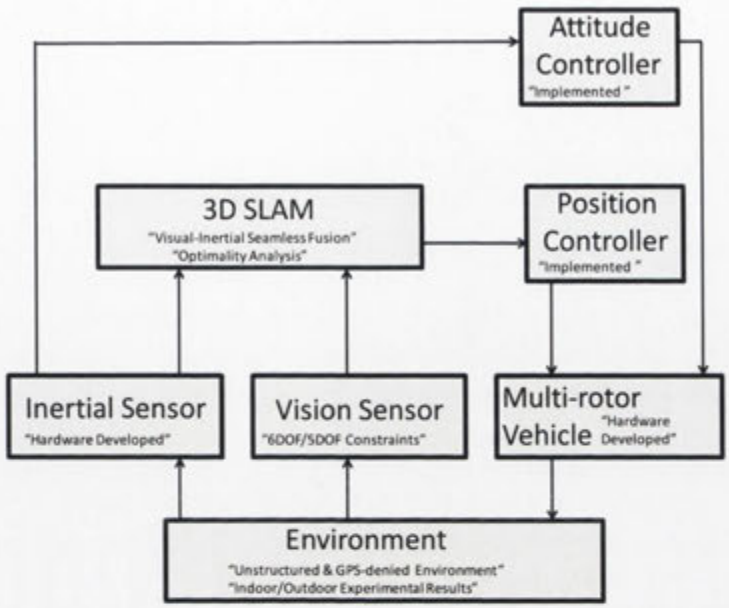



Figure 1.5: Overall framework of our system in which real-time on-board Visual-Inertial estimation framework for the localisation, mapping and control of the multi-rotor platform is developed.

sion framework (a graceful degradation mechanism that uses the 2D sensing mode in case of unavailability of 3D information). We have also used these constraints to resolve the computation complexity problem in monocular SLAM.

- A computational efficient approach using a loose coupling framework [38] where vision based measurements (6DOF and 5DOF constraints) are treated separately from the main filter. The real-time on-board implementation of the Visual-Inertial framework is developed using a direct filtering approach. The architecture of our implementation was developed with modularity and robustness in mind. A demonstration of the developed framework is evaluated on an in-house multirotor platform with hovering control. For demonstration purposes, an attitude and position PID controller were developed through an open-source autopilot system.

- The hardware development of the low-cost IMU using MEMS sensors (with gyroscope, accelerometer, and magnetometer). Furthermore the static and dy-

namic calibration of the in-house IMU is performed with the vision sensor for accelerometer and gyro biases.

- Finally, we analysed the theoretical foundation of the 3D SLAM problem by investigating the structure of the problem (number of local minima). The analysis conducted in this work helps to understand the highly non-convex nature of 3D SLAM and provides an upper bound for the number of local minima in the SLAM problem. Furthermore, we have provided the theory behind a practical approach for finding a globally optimal solution to the 3DOF SLAM problem.

## 1.4 Thesis Structure

This thesis comprises eight chapters including this introduction. We have provided a general overview of the navigational approaches for aerial vehicles. However, the relevant literature review for any particular topic will be found where that topic is presented.

**Chapter 2**, provides the statistical background for the estimation approaches. We have presented the basic theory behind the Bayesian estimation, Kalman and extended Kalman filter. We have also provided the details of the inertial navigation algorithms, and the inertial aiding architecture is discussed.

**Chapter 3**, formulates the theory behind the non-holonomic visual constraints for monocular inertial SLAM. The approach is based on a novel two-stage integration of visual odometry and feature-based SLAM. The proposed approach is evaluated using simulation and a benchmark dataset available on-line. The results show that the visual non-holonomic constraints relieve the burden on SLAM, aiding loop and thus opening up the possibility of reduced map size while maintaining the inertial stability and performance.

**Chapter 4**, shows that we developed a modular approach and followed a loosely coupled architecture for sensor fusion. Firstly, the work of visual constraints from the monocular camera is extended to the RGB-D sensor. We provide the related work

and depth drop-out problem in the 3D sensing sensor. Then we discuss the details of the developed approach to estimate the 6DOF and 5DOF motion constraints from the vision sensor. These extracted motion constraints are used in the fusion framework of vision and inertial sensors in a probabilistic framework. The inertial sensor drives the process model of EKF and visual 6DOF/5DOF constraints are used as a measurement model to minimise the inertial drift.

**Chapter 5**, gives the details of hardware development and real-time on-board implementation. Firstly, we describe the in-house development of a multicopter platform and inertial measurement unit. We also present the avionics and payload system for autonomous control. Afterwards, we describe the real-time Visual-Inertial implementation with a position and attitude controller for autopilot system.

**Chapter 6**, experimental results are presented. We describe the experimental set-up with the static sensor calibration of the inertial sensor and its relative calibration with the vision sensor. The first set of results provides validation for the developed approach by comparing it with the centimetre/0.5° ground truth data (VICON). The second set of results was generated in an outdoor environment and a complete analysis of the gyro/accelerometer biases is provided.

**Chapter 7**, details the conducted analysis on the 3D SLAM problem using stationary point analysis where the SLAM is formulated as a non-linear least square problem. The mathematical derivation is also provided with the proof on the maximum number of local minima. A simulation study is carried out to provide the analytical results. Afterwards, the theoretical limit on the number of local minima is exploited to solve the SLAM problem optimally. The results are provided in the simulation and real dataset to prove the global convergence.

Finally **Chapter 8**, provides the conclusion of this thesis by summarising the contribution along with future directions.

# Chapter 2

# Background

## 2.1 Introduction

This chapter presents the background for the Bayesian estimation theory and inertial navigation system. The chapter first provides a brief overview of the Bayesian theory, which forms the basis of sensor fusion in this thesis. The chapter also discusses the inertial navigation system and aiding architectures which are used to estimate the 6DOF motion of the autonomous vehicle.

Firstly, the chapter provides the foundation for the discrete Gaussian based Bayesian filtering. Two versions of this filter, linear and non-linear Kalman filters, are also discussed.

Secondly, the details of the IMU which is used as a dead-reckoning system for the autonomous platform are provided. The high-date rates of the inertial system play a key part in estimating the 6DOF pose of the highly dynamic vehicle (such as aerial vehicles). In addition the different coordinate systems used in this thesis for measuring the sensor measurements are discussed. The theory of inertial differential equations is also presented to estimate the position/velocity/attitude of the inertial platform.

Finally, the inertial aiding mechanism with different architecture (in the context of estimation theory) is provided. The concepts of loosely and tightly coupled modes

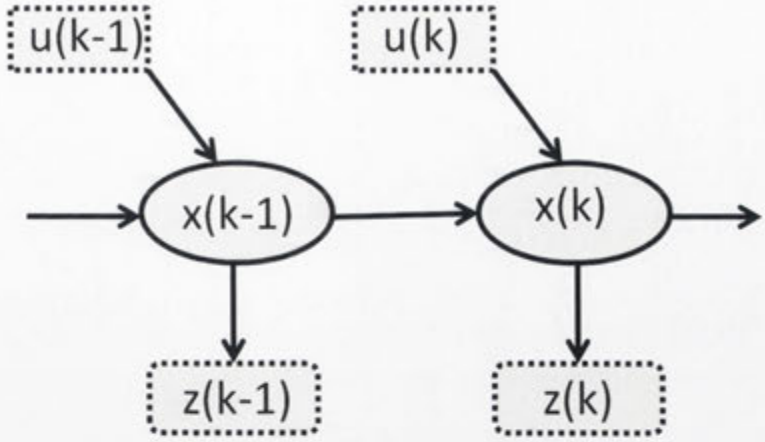



Figure 2.1: The Hidden Markov Model (HMM) where the true state is assumed to be an unobserved Markov process, and the measurements are the observed states.

are also presented along with their advantages and disadvantages.

## 2.2 Bayesian Filtering

Many problems in robotics require the estimation of the state of the dynamic system that evolves over time using a sequence of noisy measurements. Bayesian estimation is a probabilistic technique used to estimate the unknown probability density function in an iterative fashion (over time) using the process and measurement model [7].

The process model, is the state transition model and describes the evolution of the state of the system. On the other hand, the measurement model (observation model) explains the relationship between state and observations. In Bayesian estimation the assumption is that both of these models are probabilistic in nature. In contrast to classical estimation literature, the Bayesian philosophy is to exploit the prior information with noisy measurements in order to estimate the state of the system.

The state vector $\mathrm{x}(k)$ to be evolved over time $k$ with its probabilistic model $\mathrm{P}(\mathrm{x}(k))$ is assumed to be an unobserved Markov process [7]. The observations $\mathrm{z}(k)$ are the observed states in the hidden Markov model as shown in Figure 2.1. The set of all of

these observations are defined as

$$\mathrm{Z}^k = \{\mathrm{z}(1), \mathrm{z}(2), \ldots, \mathrm{z}(k-1), \mathrm{z}(k)\} = \left\{\mathrm{Z}^{k-1}, \mathrm{z}(k)\right\}. \tag{2.1}$$

The process model is considered as a Markov process, in which the state $\mathrm{x}(k)$ depends only on the previous state $\mathrm{x}(k-1)$ given the input control $\mathrm{u}(k)$ as

$$\mathrm{P}(\mathrm{x}(k) \mid \mathrm{x}(k-1), \mathrm{x}(k-2), \ldots, \mathrm{x}(0), \mathrm{u}(k)) \;=\; \mathrm{P}(\mathrm{x}(k) \mid \mathrm{x}(k-1), \mathrm{u}(k)). \tag{2.2}$$

Here the Markov assumption makes the probability of the current state conditionally independent of the earlier state. The $\mathrm{x}(0)$ is the (initial) prior state with its respective probability distribution $\mathrm{P}(\mathrm{x}(0))$. It can be also observed from the above form that the process model is independent of the observation $\mathrm{z}(k)$. The observation model defined as: given the current state $\mathrm{x}(k)$, what's the probability of making an observation $\mathrm{z}(k)$

$$\mathrm{P}(\mathrm{z}(k) \mid \mathrm{x}(k), \mathrm{x}(k-1), \ldots, \mathrm{x}(0)) \;=\; \mathrm{P}(\mathrm{z}(k) \mid \mathrm{x}(k)) \tag{2.3}$$

Similarly, the measurement at the time step k is conditionally independent of all the other states given the current state due to Markov assumption.

The Bayes theorem [7] defines a relationship between the prior and observation density to estimate the posterior density. The posterior density of the state $\mathrm{x}(k)$ is the conditional probability that is assigned after the relevant observation $\mathrm{z}(k)$ has taken place.

$$\mathrm{P}(\mathrm{x}(k) \mid \mathrm{Z}^k) = \frac{\mathrm{P}(\mathrm{z}(k) \mid \mathrm{x}(k))\mathrm{P}(\mathrm{x}(k) \mid \mathrm{Z}^{k-1})}{\mathrm{P}(\mathrm{z}(k) \mid \mathrm{Z}^{k-1})} \tag{2.4}$$

Where the denominator is a normalisation factor (a constant value with respect to x). In practice, the numerator can be calculated first and then simply normalised (since its integral must be unitary) defined as

$$\eta = \frac{1}{\mathrm{P}(\mathrm{z}(k) \mid \mathrm{Z}^{k-1})} \qquad (2.5)$$
$$\frac{1}{\int \mathrm{P}(\mathrm{z}(k) \mid \mathrm{x}(k)\mathrm{P}(\mathrm{x}(k) \mid \mathrm{Z}^{k-1}))dx(k-1)}$$

Substituting Equation 2.5 into Equation 2.4 leads to Equation 2.6 (also known as the update step). Upon arrival of a new observation at time $k$ it calculates the posterior density from the prior density at time $k-1$. Thus the equation 2.4 simplifies to

$$\underbrace{\mathrm{P}(\mathrm{x}(k) \mid \mathrm{Z}^{k})}_{Posterior} = \eta \cdot \underbrace{\mathrm{P}(\mathrm{z}(k) \mid \mathrm{x}(k))}_{Likelihood} \underbrace{\mathrm{P}(\mathrm{x}(k) \mid \mathrm{Z}^{k-1})}_{Prior}. \qquad (2.6)$$

This is the core of the Bayesian estimation and leads to two major steps of measurement likelihood (update) and prior density (prediction). The prediction $\mathrm{P}(\mathrm{x}(k)|\mathrm{Z}^{k-1})$ of the current state $\mathrm{x}(k)$ can be defined in terms of the marginalisation of the joint probability of the current and previous state using the total probability law [7]:

$$P(\mathrm{x}(k) \mid \mathrm{Z}^{k-1}) = \int \mathrm{P}(\mathrm{x}(k), \mathrm{x}(k-1) \mid \mathrm{Z}^{k-1})dx(k-1) \qquad (2.7)$$

In addition to that, by using the chain rule the joint probability can be represented as:

$$P(\mathrm{x}(k) \mid \mathrm{Z}^{k-1}) = \int \mathrm{P}(\mathrm{x}(k) \mid \mathrm{x}(k-1), \mathrm{u}(k))\mathrm{P}(\mathrm{x}(k-1) \mid \mathrm{Z}^{k-1})dx(k-1) \qquad (2.8)$$

Finally we have

$$\underbrace{\mathrm{P}(\mathrm{x}(k) \mid \mathrm{Z}^k)}_{Posterior} = \eta \cdot \underbrace{\mathrm{P}(\mathrm{z}(k) \mid \mathrm{x}(k))}_{Likelihood} \times \underbrace{\int \mathrm{P}(\mathrm{x}(k) \mid \mathrm{x}(k-1), \mathrm{u}(k)) \mathrm{P}(\mathrm{x}(k-1) \mid \mathrm{Z}^{k-1}) dx(k-1)}_{Prediction} \tag{2.9}$$

Bayes filtering in this form is exact and can be used on any kind of system for which the Markov assumption holds. Unfortunately, in the above equations there are some integrations over the state space (Equation 2.9). In many cases the state space is high dimensional, and the Bayes filtering cannot be directly implemented. For instance, in the SLAM problem the dimension of the system state contains the pose and map information, which is in the order of hundreds (or thousands). A straightforward evaluation of Equation 2.4 would require an integration over the entire state space. For this reason, approximated techniques are needed where the probability functions are being represented as Gaussian (such as Kalman filters).

### 2.2.1 Kalman Filter

The Kalman filter is named after Rudolph Kalman, whose seminal paper in 1960 has provided a recursive solution to the discrete-time linear filtering problem [58, 59]. It is the optimal minimum mean square error estimator to the linear systems based upon the observation and state transition model.

Consider a linear process represented in state-space form by the equation

$$\mathrm{x}(k) = \mathrm{F}(k)\mathrm{x}(k-1) + \mathrm{B}(k)\mathrm{u}(k) + \mathrm{G}(k)\mathrm{w}(k) \tag{2.10}$$

where $\mathrm{F}(k)$ is a state transition model relating the state vector $\mathrm{x}(k)$ at time $k-1$ to $k$, $\mathrm{B}(k)$ is a control-input model mapping the control inputs $\mathrm{u}(k)$ to the states, and $\mathrm{G}(k)$ is a noise model mapping the process noise vector $\mathrm{w}(k)$ to the states.

The process noise $\mathrm{w}(k)$ is assumed Gaussian with zero mean and strength

$Q(k)$:

$$\begin{aligned} E[w(k)] &= 0 \\ E\left[w(k)w^T(j)\right] &= Q(k)\delta(j) \end{aligned} \tag{2.11}$$

where $\delta(j)$ is the Dirac delta function [58].

The linear observation model is assumed to be represented by

$$z(k) = H(k)x(k) + D(k)v(k) \tag{2.12}$$

where $H(k)$ is a matrix mapping the state vector to observation space and $D(k)$ is a matrix mapping the observation noise $v(k)$ to observation space. The process noise $v(k)$ is assumed Gaussian with zero mean and strength $R(k)$:

$$\begin{aligned} E[v(k)] &= 0 \\ E\left[v(k)v^T(j)\right] &= R(k)\delta(j) \end{aligned} \tag{2.13}$$

Further, it is also assumed that the process and observation noises are uncorrelated

$$E\left[v(k)w^T(j)\right] = 0 \qquad \forall\, k, j \tag{2.14}$$

Following from the earlier definition of the Kalman filter [1].

$$\hat{\mathbf{x}}(k) = F(k)\hat{x}(k) + B(k)u(k) + K(k)[z(k) - H(k)\hat{x}(k)] \tag{2.15}$$

where $\hat{x}(k)$ is the estimate of the state vector $x(k)$, and $K(k)$ is the Kalman gain.

The filter innovation is defined as $[z(k) - H(k)\hat{x}(k)]$, which provides the difference between the observation $z(k)$ and the predicted observation $H(k)\hat{x}(k)$. This innovation term is generally represented by the symbol $\nu(k)$.

[1] For the Kalman filter derivation details, interested readers are encouraged to explore the work of [60].

The measure of uncertainty in the system is given by the covariance update equation, and is given by the solution of the Riccati equation [60]:

$$\mathrm{P}(k) = \mathrm{F}(k)\mathrm{P}(k) + \mathrm{P}(k)\mathrm{F}^T(k) + \mathrm{G}(k)\mathrm{Q}(k)\mathrm{G}^T(k) - \mathrm{K}(k)\mathrm{H}(k)\mathrm{P}(k) \tag{2.16}$$

where $\mathrm{P}(k)$ is the covariance matrix. The covariance matrix P provides a representation of the uncertainty in the state estimate due to the fact that both the sensor observations and the process model predictions are corrupted by noise. The Kalman filter minimises this uncertainty in a minimum mean squared error sense.

The Kalman gain $\mathrm{K}(k)$ is given by

$$\mathrm{K}(k) = \mathrm{P}(k)\mathrm{H}^T(k)\mathrm{R}^{-1}(k) \tag{2.17}$$

Equations 2.15 and 2.16 are differential equations and in general are solved by integrating forward from some initial conditions

$$\hat{\mathrm{x}}(0) = \hat{\mathrm{x}}_0, \qquad \hat{\mathrm{P}}(0) = \hat{\mathrm{P}}_0 \tag{2.18}$$

where $\hat{\mathrm{x}}_0$ is the initial value of the state vector and $\hat{\mathrm{P}}_0$ is the initial value for the covariance matrix (generally the initial covariance is made large enough to account for the uncertainty in the initial state estimate [7]).

### 2.2.2 Extended Kalman Filtering

The extended Kalman filter is the non-linear version of the Kalman filter in the estimation theory [58]. Many common applications cannot be adequately represented using linear models due to their non-linear nature. In those cases non-linear models must be used. However, the Kalman filter is explicitly derived for the linear state transition and measurement models. The adapted technique of Taylor series expansion (from calculus) become the working solution for Kalman filter to be used for the

non-linear systems hence called the extended Kalman filter[2].

Consider a non-linear process represented in state space form by the equation

$$\mathrm{x}(k) = \mathrm{f}(\mathrm{x}(k-1), \mathrm{u}(k)) + \mathrm{G}(k)\mathrm{w}(k) \tag{2.19}$$

where $\mathrm{f}(\cdot)$ is a non-linear function mapping the control inputs $\mathrm{u}(k)$ and state $\mathrm{x}(k-1)$ to the state $\mathrm{x}(k)$.

The observation equation is also a non-linear function given by

$$\mathbf{z}(k) = \mathrm{h}(\mathrm{x}(k)) + \mathrm{D}(k)\mathrm{v}(k) \tag{2.20}$$

where $\mathrm{h}(\cdot)$ is a non-linear function mapping the state to observation space. The matrix $\mathrm{D}(k), \mathrm{G}(k)$, and the noise sources $\mathrm{v}(k), \mathrm{w}(k)$ retain their definitions from the earlier Section 2.2.1.

The main EKF equations for the estimation of the non-linear system [58] are given as

$$\hat{\mathrm{x}} = \mathrm{f}(\hat{\mathrm{x}}(k), \mathrm{u}(k)) + \mathrm{K}(k)\left[\mathrm{z}(k) - \mathrm{h}(\hat{\mathrm{x}}(k))\right] \tag{2.21}$$

with covariance and gain equations as

$$\begin{aligned} \mathrm{P}(k) &= \mathrm{F}'(k)\mathrm{P}(k) + \mathrm{P}(k)\mathrm{F}'^{T}(k) + \mathrm{G}(k)\mathrm{Q}(k)\mathrm{G}^{T}(k) - \mathrm{K}(k)\mathrm{H}'(k)\mathrm{P}(k) \\ \mathrm{K}(k) &= \mathrm{P}(k)\mathrm{H}'^{T}(k)\mathrm{R}^{-1}(k) \end{aligned} \tag{2.22}$$

The above equations are functionally similar to the linear Kalman filter, however the linear state transition $\mathrm{F}(k)$ and $\mathrm{H}(k)$ are replaced by $\mathrm{F}'(k)$ and $\mathrm{H}'(k)$ respectively. The model $\mathrm{F}'(k)$ and $\mathrm{H}'(k)$ are calculated as linearisation about the state estimate of the functions $\mathrm{f}(\cdot)$ and $\mathrm{h}(\cdot)$ as

[2]Other approaches like the Monte-Carlo methods (such as Particle filters) are also employed for non-linear estimation however they are more computationally expensive for any moderate or greater size state-space [61].

$$\mathrm{F}' = \left.\frac{\partial \mathrm{f}(\mathrm{x}(k-1), \mathrm{u}(k))}{\partial \mathrm{x}}\right|_{\mathrm{x}=\hat{\mathrm{x}}(k)} \tag{2.23}$$

$$\mathrm{H}' = \left.\frac{\partial \mathrm{h}(\mathrm{x}(k), k)}{\partial \mathrm{x}}\right|_{\mathrm{x}=\hat{\mathrm{x}}(k)} \tag{2.24}$$

The Jacobians $\mathrm{F}'(k)$ and $\mathrm{H}'(k)$ are the result of a Taylor series expansion by neglecting the higher order terms. Therefore EKF provides the approximation of the true state of the non-linear system. This approximation in the state estimator is not valid for systems where the sample time is larger than the rate of change of non-linear functions (called linearisation errors). To avoid the linearisation error in the literature, smoothing based approaches are generally employed [62]. However the smoothing based methods adds a level of computational overhead not suitable for real-time implementation [44].

## 2.3 Inertial Navigation System

Inertial navigation determines the pose of the robotic vehicle at high data-rates. Therefore, it is considered to be a vital part of airborne navigation [3] as a dead-reckoning system [64]. The inertial system works on a principle of inertia [4] and measures the exerted force and rotation rate of an inertial unit.

The sensor module in the inertial system, which measures the acceleration generated by an external force is called an accelerometer [65]. It measures the total acceleration encountered, that is, both the acceleration due to gravity and all other due to external forces (hence the free-fall accelerometer has no detectable input to measure). To compensate the component of acceleration due to gravity, the attitude of the accelerometer needs to be known with respect to the local vertical. Once the

[3] Inertial sensors are also very popular in various land and underwater applications [8, 9, 11, 63].

[4] Inertia is defined as the tendency of an object to remain in constant motion until disturbed by an external force or a torque.

effect of gravity is compensated, the mathematical integration of acceleration yields the velocity and position information.

The inertial sensor which is used to measure the angular rate is known as a gyroscope (or gyro), which is based on the principal of angular momentum [64]. The mathematical integration of the angular rate yields the attitude information. Another inertial sensor which is used alongside with gyros (to estimate the heading information) is the magnetometer. The magnetometer works on the principal of measuring the direction and magnitude of the Earth's magnetic field. To estimate the heading information, the magnetometer must be level to the earth's surface and it should not be affected by magnetic disturbance of the nearby devices[5].

### 2.3.1 Inertial Measurement Unit

The IMU is an electronic instrument and its structure is a combination of power supplies, micro-controller and the inertial sensors. IMUs have many different designs but they generally fall into two major categories, strapdown or traditional gimbaled system [65].

In the gimbaled system, the inertial sensors are mounted on a gimbaled table which is kept level and mechanically isolated from the rotational motion of the platform. This category of IMU can provide more accurate estimates and is used in high-precision applications [65]. However they are expensive to build and much heavier than its counterparts.

The second category is the strapdown IMU, which has recently gained a lot of attention due to MEMS based sensors (which offer lower cost and reduced size [6]). On these type of systems the inertial sensors are fixed with respect to the body frame, therefore the mechanical complexity is much simpler than the earlier systems. The major drawback of strapdown IMUs (as compare to traditional gimbaled system) is that they require more computational resources and are less precise.

[5]Heading is the angle measured clockwise from a true North direction.

[6]There are some high-cost strapdown versions as well, which uses fibre-optic or ring-laser gyroscope [66] to provide more accurate attitude estimates.

In this thesis, we focused on the low-cost strapdown tri-axial IMU, which is comprised of three orthogonally arranged accelerometers, gyroscopes and magnetometers. The computational controller with the mathematical integration algorithms (on the IMU data) forms the INS (which is used to estimate the position, velocity and attitude information). The MEMS based inertial sensor typically is not compensated for instrumental errors (such as bias and scale factors) [7] [8]. These instrumental errors cause the inertial drift (the error accumulation during the mathematical integration of position, velocity and attitude information). Chapter 6 describes the static/dynamic calibration method to obtain the bias and scale factor information in order to minimise the impact of these error. In addition to these error, some other high level errors such as a random noise [9] and misalignment errors are also the cause of the unbounded drift in the low-cost IMU sensors.

### 2.3.2 Coordinate Systems for Navigation

The transformation methods are required to convert and express different sensor measurements in a common reference frame. Depending upon the application and their context, different coordinate systems can be formulated for sensor measurements or navigation equations. Generally the transformation involves rotation and translation, where translation is used to represent different origin coordinates and rotation to represent different axis direction. The knowledge of the following coordinate systems (essential for this thesis) is explained as

- Inertial Coordinate Frame: An inertial frame is a non-accelerating reference frame in which the Newton's first law of motion is valid. It is defined as a reference frame with arbitrary origin and orientation. The inertial sensor measures the force and rotation rate with respect to this frame of reference.

[7] Bias is simply an offset in the sensor output that varies randomly from time to time.

[8] The scale factor errors are proportional to the output measurements and hence the effect of the error is worse when the input acceleration/gyro values are large (it is defined as a ratio between the measured output and change in sense input) [67].

[9] These error causes due to temperature variation and non-linear characteristics of the sensors.

- Earth-Centered Earth-Fixed Frame: This frame rotates with the earth and its origin is fixed to the centre of the earth. The x-axis is along the Greenwich meridian and z-axis is along the rotational axis of the earth, where y-axis follows the right-hand rule. The vehicle's position in this frame can be represented in Cartesian or geodetic (latitude, longitude and height) coordinate frame.

- Earth-Fixed Local-Tangent Frame: This frame has a origin defined on a local reference point while its x-axis points to north, y-axis points to east and z-axis points down. A frame with the mentioned arrangement is called north-east-down (NED) system. It does not rotate with the earth and is often suited to the small area applications (where the curvature of the earth can be ignored). This is the most popular reference frame used for low-cost IMUs, where the application is not sensitive to the Earth's rotation.

- Body Frame: This frame is rigidly attached to the navigation platform with its origin coinciding with the platform's centre of gravity and moving/rotating with the platform. The body frame consist of a set of three orthogonal axes where the x-axis points forward with respect to the platform, the z-axis point down and the y-axis is defined by the right hand rule [10].

- Sensor Frame: In general, the observations from the different sensors are represented in their own respective frame. The origin is at the location of the sensor and the axes are typically defined according to the type of the sensor. In this thesis, the IMU and vision sensor are being used and each of them has its own sensor frame to represent the measurements. The calibration routine to estimate the inter-transformation between different coordinate frames will be presented in the Chapter 6 (in order to consider different observations in a common reference frame).

[10]Note that the mentioned definition is not unique, but the dominant approach typically deployed in various platforms.

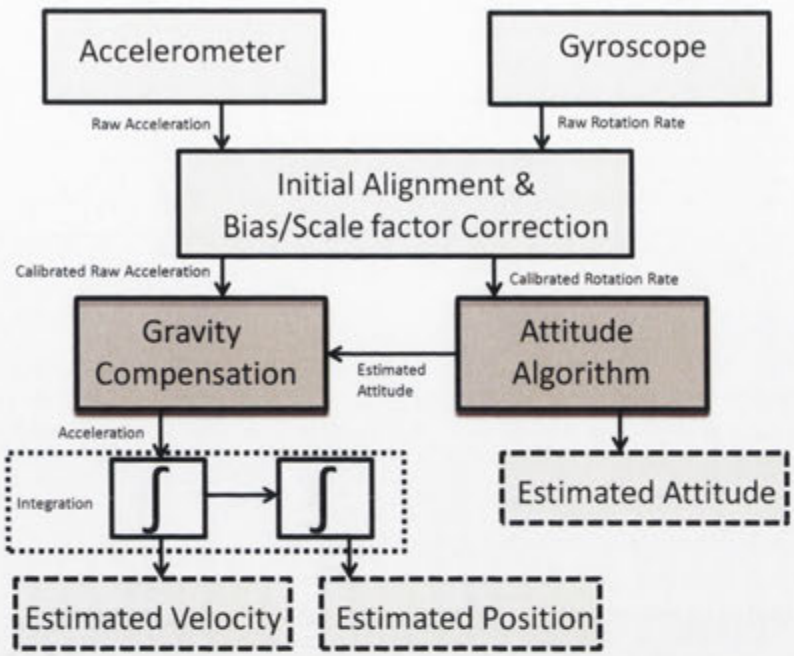



Figure 2.2: The position, velocity and attitude computation from raw accelerometer and gyro measurements.

### 2.3.3 Inertial Navigation Equations

The inertial navigation equations refers to the set of differential equations which are used to relate the inertial frame measurements (such as velocity, attitude and position) to the corresponding navigation measurements in the desired common reference frame. The set of equations may vary depending upon the selection of reference frame, type of the application and category of IMU (Gimbaled or strap-down). In addition, the complexity of the equations varies depending upon the coriolis force and effects of coning/sculling [41]. In this thesis, we are using a low-cost strap-down IMU so it is acceptable to use the simplified navigation equation while ignoring the coriolis/coning effect [11].

[11]The effect of coning/sculling is insignificant unless the platform experience vibration with frequency around the sampling rate of the inertial sensor. As the application area covers a small area so the effect of curvature of the earth is considered to be negligible.

#### 2.3.3.1 Position/Velocity Algorithm

The accelerations measured from an IMU are in a body frame and it is necessary to transform them into the navigation frame as discussed earlier. This is achieved by using the rotation matrix (generally denoted by $C$) as shown below

$$\boldsymbol{f}^n(k) = \boldsymbol{C}_b^n(k)\boldsymbol{f}^b(k) \tag{2.25}$$

Where, the $\boldsymbol{C}_b^n$ is the rotation matrix from body frame to navigation frame (detail to estimate the rotation matrix can be found in the Section 2.3.4) and $\boldsymbol{f}^b$ is the acceleration measured from the accelerometer in the body frame at time $k$. This transformation is required to evaluate the acceleration vector in the desired navigation frame, given the acceleration vector in the body fixed frame. However as the acceleration also contains the gravity vector, so the acceleration in the navigation frame $\boldsymbol{f}^n$ is compensated as

$$\dot{\boldsymbol{v}}^n(k) = \boldsymbol{C}_b^n(k)\boldsymbol{f}^b(k) + g \tag{2.26}$$

Where $g$ is the gravity vector defined as $g = [0, 0, 9.8m/s^2]^T$ [12]. The $\dot{\boldsymbol{v}}^n(k)$ is the rate of change of velocity, which is integrated to lead to velocity as

$$\boldsymbol{v}^n(k) = \boldsymbol{v}^n(k-1) \; + \; \int_{k-1}^{k} \dot{\boldsymbol{v}}^n(\tau) \; d\tau \tag{2.27}$$

Finally the position is computed as

$$\boldsymbol{p}^n(k) = \boldsymbol{p}^n(k-1) \; + \; \int_{k-1}^{k} \boldsymbol{v}^n(\tau) d\tau \tag{2.28}$$

[12]The gravity value varies from place to place and the gravitational model for the sensor cannot be modelled exactly hence approximated value is used here. The axes convention used for IMU is x-axis pointing forward, y-axis pointing left and z-axis facing down

### 2.3.4 Attitude Equations

When the platform is rotating with respect to the navigation frame, the gyro information is used continuously to update the rotation matrix $\boldsymbol{C}_b^n$. The main approaches used to propagate the attitude of the platform are Euler, direction cosine matrix and quaternion algorithms [41].

The selection of the attitude update approach is based upon the application and processing requirement, however regardless of the type of the approach implemented, the analytical form provided by these algorithm yield similar results. Among all the approaches, the Euler approach is conceptually easy to understand, however it has the problem of gimbal lock [13]. The gimbal lock problem can be avoided by either using quaternion or DCM approach or making sure that the system remains inside the bounds of the Euler angle (in case of aerial vehicle, if the application does not involve the flipping (upside down) operation, then the system can avoid the gimbal lock situation). In this section we will discuss the attitude algorithms (Euler and quaternion [14]) relevant to this thesis for attitude propagation.

#### 2.3.4.1 Euler Approach

The attitude of the platform in the 3D Euclidean space is represented by Euler angles, which is the mathematical similar form of the gimbal system. The rotation of an inertial frame can be understood as a combination of three successive elemental rotations described by the Euler angles (such as roll $\phi$, pitch $\theta$ and yaw $\psi$). The Euler angles used to represent the rotations from navigation to body frames are placed in a rotation matrix $\boldsymbol{C}_n^b$. This rotation matrix can be decomposed as the product of three basic rotations using ZYX-convention (that is, the navigation frame is rotated and matched with the body frame in a sequence of yaw, pitch and then roll [69]) as

[13] The degeneration of the system from 3D to 2D space when one degree of freedom is lost due to the parallel alignment of two rotational axes[63].

[14] More description about the direction cosine matrix can be found in our earlier work [68].

$$C_n^b(k) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \boldsymbol{C}_\phi & \boldsymbol{S}_\phi \\ 0 & -\boldsymbol{S}_\phi & \boldsymbol{C}_\phi \end{bmatrix} \begin{bmatrix} \boldsymbol{C}_\theta & 0 & -\boldsymbol{S}_\theta \\ 0 & 1 & 0 \\ \boldsymbol{S}_\theta & 0 & \boldsymbol{C}_\theta \end{bmatrix} \begin{bmatrix} \boldsymbol{C}_\psi & \boldsymbol{S}_\psi & 0 \\ -\boldsymbol{S}_\psi & \boldsymbol{C}_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.29}$$

Where $\boldsymbol{S}_{(\cdot)}$ and $\boldsymbol{C}_{(\cdot)}$ represents the $\boldsymbol{sin}(\cdot)$ and $\boldsymbol{cos}(\cdot)$ trigonometric functions respectively.

The transformation matrix from body to navigation $\boldsymbol{C}_b^n$ can be obtained by taking the inverse of $\boldsymbol{C}_n^b$ or its transpose (as the rotation matrix is orthogonal)

$$\boldsymbol{C}_b^n(k) = (\boldsymbol{C}_n^b(k))^{-1} = (\boldsymbol{C}_n^b(k))^T = \begin{bmatrix} \boldsymbol{C}_\theta\boldsymbol{C}_\psi & -\boldsymbol{C}_\phi\boldsymbol{S}_\psi + \boldsymbol{S}_\phi\boldsymbol{S}_\theta\boldsymbol{C}_\psi & \boldsymbol{S}_\phi\boldsymbol{S}_\psi + \boldsymbol{C}_\phi\boldsymbol{S}_\theta\boldsymbol{C}_\psi \\ \boldsymbol{C}_\theta\boldsymbol{S}_\psi & \boldsymbol{C}_\phi\boldsymbol{C}_\psi + \boldsymbol{S}_\phi\boldsymbol{S}_\theta\boldsymbol{C}_\psi & \boldsymbol{S}_\theta\boldsymbol{C}_\psi + \boldsymbol{C}_\phi\boldsymbol{S}_\theta\boldsymbol{S}_\psi \\ \boldsymbol{S}_\theta & \boldsymbol{S}_\psi\boldsymbol{C}_\theta & \boldsymbol{C}_\phi\boldsymbol{C}_\theta \end{bmatrix} \tag{2.30}$$

The angular measurements from the gyro are used to update the attitude transformation matrix dynamically which cause the Euler angles to change as well. Based on the sequence of rotation from navigation to body frame, Euler rates can be obtained from the transformation of the output angular rate from the gyro measurements. Where the roll rate ($\dot{\phi}$) is the angular rate along the x-axis of the body frame of gyro ($\boldsymbol{w}_x^b$). However the pitch rate ($\dot{\theta}$) is transformed according to the roll angle and the yaw rate ($\dot{\psi}$) according to roll and pitch angle. The summation of these three Euler rates will result in the following angular rates when expressed in the body frame:

$$\begin{bmatrix} \boldsymbol{w}_x^b \\ \boldsymbol{w}_y^b \\ \boldsymbol{w}_z^b \end{bmatrix} = \tag{2.31}$$

$$\begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \boldsymbol{C}_\phi & \boldsymbol{S}_\phi \\ 0 & -\boldsymbol{S}_\phi & \boldsymbol{C}_\phi \end{bmatrix} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \boldsymbol{C}_\phi & \boldsymbol{S}_\phi \\ 0 & -\boldsymbol{S}_\phi & \boldsymbol{C}_\phi \end{bmatrix} \begin{bmatrix} \boldsymbol{C}_\theta & 0 & -\boldsymbol{S}_\theta \\ 0 & 1 & 0 \\ \boldsymbol{S}_\theta & 0 & \boldsymbol{C}_\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix}$$

The inverse of the Equation 2.32 reveals the rate of change of angle estimated from the gyro measurements as

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \boldsymbol{S}_{\phi}\boldsymbol{S}_{\theta}\boldsymbol{C}_{\theta}{}^{-1} & \boldsymbol{C}_{\phi}\boldsymbol{S}_{\theta}\boldsymbol{C}_{\theta}{}^{-1} \\ 0 & \boldsymbol{C}_{\phi} & -\boldsymbol{S}_{\phi} \\ 0 & \boldsymbol{S}_{\phi}\boldsymbol{C}_{\theta}{}^{-1} & \boldsymbol{C}_{\phi}\boldsymbol{C}_{\theta}{}^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{w}_x^b \\ \boldsymbol{w}_y^b \\ \boldsymbol{w}_z^b \end{bmatrix} \tag{2.32}$$

$$= \boldsymbol{E}_b^n(k)\boldsymbol{w}^b(k)$$

The integration of the angular rates yields the Euler angles as

$$\begin{bmatrix} \phi(k) \\ \theta(k) \\ \psi(k) \end{bmatrix} = \begin{bmatrix} \phi(k-1) \\ \theta(k-1) \\ \psi(k-1) \end{bmatrix} + \int_{k-1}^{k} \begin{bmatrix} \dot{\phi}(\tau) \\ \dot{\theta}(\tau) \\ \dot{\psi}(\tau) \end{bmatrix} d\tau \tag{2.33}$$

#### 2.3.4.2 Quaternion Approach

This approach provides a convenient mathematical notation for representing the attitude of the platform [69]. The quaternion approach can be utilised to propagate the attitude from one frame to the other frame by using a single rotation about a unit vector $[\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3]$ with a magnitude $\boldsymbol{q}_0$. The quaternion vector is represented by these four components as

$$\boldsymbol{q}(k) = [\boldsymbol{q}_0 \quad \boldsymbol{q}_1 \quad \boldsymbol{q}_2 \quad \boldsymbol{q}_3] \tag{2.34}$$

For the attitude propagation, the half angle increment approach is used for the Equation 2.34, as done in the method presented by [70]. The method provides the relationship between the input angular information and the attitude quaternion as

$$\dot{\boldsymbol{q}}(k) = \frac{1}{2}[\boldsymbol{q}\otimes]\,\grave{w}^b \tag{2.35}$$

Where the symbol $\otimes$ represents the product between two quaternions, the notation $[\boldsymbol{q}\otimes]$ is defined as

$$[q\otimes] = \begin{bmatrix} q_0 & -q_1 & -q_2 & -q_3 \\ q_1 & q_0 & -q_3 & q_2 \\ q_2 & q_3 & q_0 & -q_1 \\ q_3 & -q_2 & q_1 & q_0 \end{bmatrix} \tag{2.36}$$

The quaternion from the gyro angular information is formed as

$$\dot{w^b} = \begin{bmatrix} 0 \\ w_x^b \\ w_y^b \\ w_z^b \end{bmatrix} \tag{2.37}$$

For the transformation of acceleration (Equation 2.25) from the body to navigation frame, the quaternion is converted into the rotation matrix representation $C_b^n$ as

$$C_b^n = \begin{bmatrix} {q_0}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2 & -2(q_0q_3 - q_1q_2) & 2(q_0q_2 + q_1q_3) \\ 2(q_0q_3 + q_1q_2) & {q_0}^2 - {q_1}^2 + {q_2}^2 - {q_3}^2 & -2(q_0q_1 - q_2q_3) \\ -2(q_0q_2 - q_1q_3) & 2(q_0q_1 + q_2q_3) & {q_0}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2 \end{bmatrix} \tag{2.38}$$

## 2.4 Architectures of Inertial Aided Navigation

Earlier the concepts of probabilistic filtering and the inertial navigation system were introduced. However in this section, we will discuss how the aided inertial sensor problem is formulated and solved using the filtering concept. In the aided navigation approach, the aiding source can be considered as providing the navigational data (position, velocity or attitude) or a raw sensor information. For example, in the vision sensor, the information is the relative range, bearing and elevation. Furthermore the inertial component can be in the state transition model as either IMU (raw data) or INS (navigation data). The aiding source and inertial component along with the state vector in the filter decides the architecture of the aided navigation. In literature, the

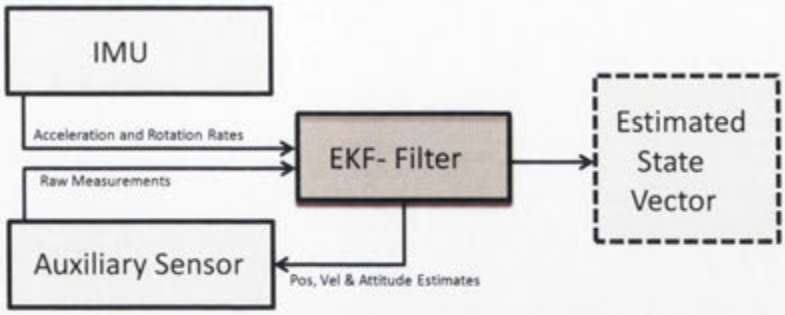



Figure 2.3: The tightly coupled architecture estimating position, velocity and attitude states of EKF filter.

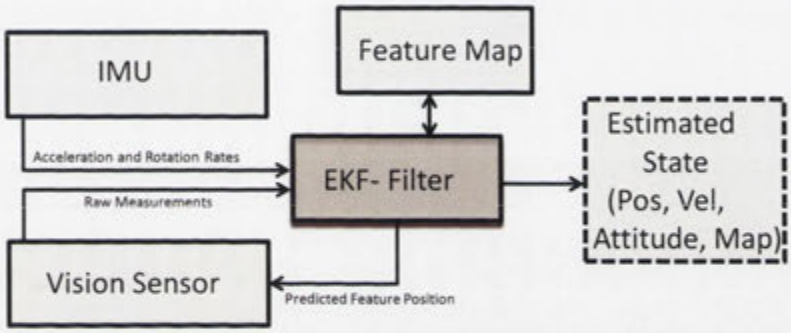



Figure 2.4: The tightly coupled architecture estimating position, velocity, attitude and map states of EKF filter using vision sensor.

aided navigation system can be categorised into loosely coupled or tightly coupled modes.

- Tightly Coupled Mode: The tightly coupled mode is also known as the centralised approach, in which all the processing of the estimated quantities is undertaken in the integrated statistical filter [38, 71, 72]. The mode treats all the information required to estimate a solution in a single EKF filter and requires the raw (or intermediate) data of inertial and aiding sensors. In this mode the feedback is also provided to the aiding source to have a tighter configuration, which in turn provides the overall system integrity. This mode offers the advantage to the system designer of allowing work with the algorithms and operation of both aiding and inertial sensors. The inertial and aiding sensors

both provide the raw data, where inertial data usually incorporates the kinematics data. Figure 2.3 shows a tightly coupled mode in which the auxiliary aiding sensor provides the raw measurement along with IMU data (providing accelerometer and gyro data) to estimate the state of the filter. The statistical filter (in our case EKF filter) estimates the states of the platform and uses these estimates to aid the auxiliary sensor in obtaining new measurements.

In relevance to this thesis, the vision aided inertial example is shown in Figure 2.4 (Our earlier work [17]). The integrated EKF uses the raw measurements from the vision sensor (image geometric features) to correct the drift in the IMU system. The IMU provides the acceleration and rotation rates to perform the prediction of position, velocity and attitude information through the INS navigation equations. The predicted platform states are used to predict the new feature measurements from the vision sensor [15]. The feature map is maintained within the filter along with the platform states (position, velocity and attitude) to perform the SLAM estimation [16]. As It is clear that the number of features increases the dimension of the system states, thereby increasing the overall complexity of the system [17].

The tightly coupled approach provides more robustness than its counterpart. However, it is more difficult to implement and computationally heavy. In addition, the model and implementation has to be changed significantly if a new type of aiding sensor is introduced (so it lacks the software modularity).

- Loosely Coupled Mode

  The loosely coupled mode is also known as the decentralised approach. In it, the inertial and auxiliary aiding sensor operate independently and provide separate navigation solutions to a statistical filter for state estimation [38, 71]. Figure

[15] The calibration between the inertial and vision sensor need to be known before sensor integration, the calibration will discussed in Chapter 6.

[16] This joint estimation of platform state information and environmental feature map is called the tightly coupled EKF-SLAM.

[17] In tightly coupled EKF-SLAM literature its well known computational complexity problem of cost $O(n^2)$ [35]. Our proposed work to handle the computational complexity problem of EKF-SLAM is presented in Chapter 3.

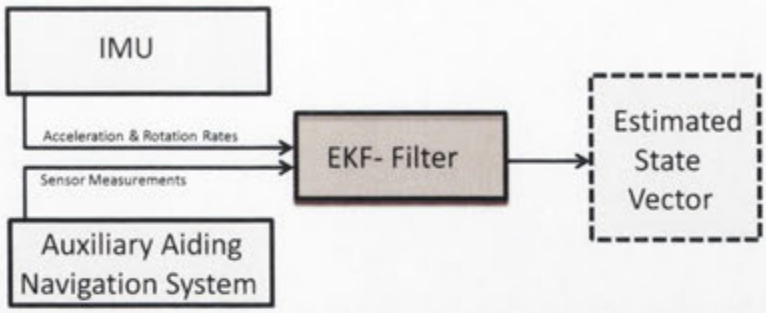



Figure 2.5: The direct implementation of a loosely coupled mode in which position, velocity and attitude states of EKF filter are estimated.

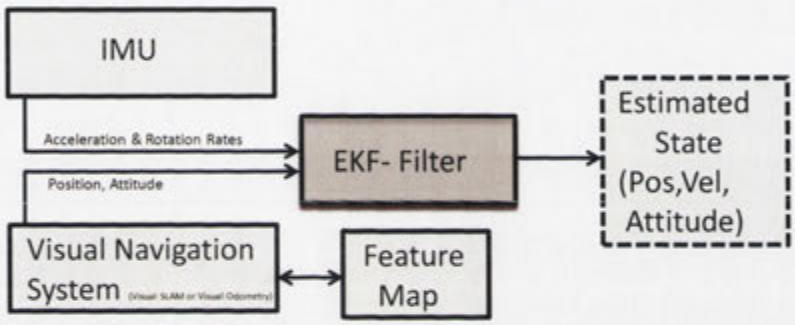



Figure 2.6: The loosely coupled architecture of vision and inertial sensor aiding to estimate the position, velocity and attitude states of the EKF filter.

2.5 shows the loosely coupled system in which the EKF uses acceleration and rotation rate in a prediction model through the INS equation (to estimate the states of the platform). At each update step, the EKF uses the navigation solution (such as position, velocity or attitude states) from the auxiliary aiding system to minimise the inertial error. The main advantage the loosely coupled mode offers is the redundancy and simplicity of the system.

In relevance to this thesis, the vision aided example is shown in Figure 2.6 (Our earlier work [73]). The integrated EKF uses the navigation solution of the vision sensor (camera position and attitude) to correct the IMU system. The IMU predicted platform states (position, velocity and attitude) are corrected

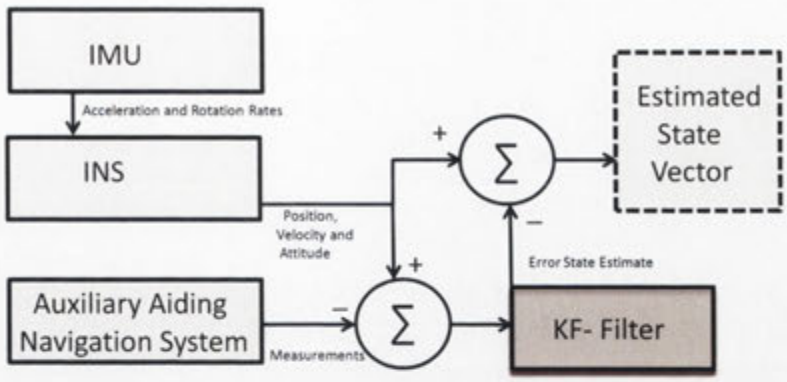



Figure 2.7: The loosely coupled indirect feed-forward implementation where position, velocity and attitude states of Kalman filter are being estimated.

with the vision based navigation solution [18].

The loosely coupled mode offers the advantage of software modularity and efficient computation. This mode is not as robust in performance as its counterpart, but it is generally a preferred architecture in literature [74, 75, 76]. The major reason being the ease of selection of aiding sensor (so that the algorithm's designer doesn't have to deal with the intricacy of using the internal data of the aiding sensor).

Earlier, the tightly and loosely coupled modes in which the filter directly estimates the state of interest were discussed. This type of implementation is called direct filtering. In it, the non-linear filter is driven by the inertial sensor (where the evaluation of the prediction equation is performed for each sample), which results in computational overhead. To overcome the computational complexity, the INS equations have to be applied external to the filter whereas the filter estimates the error in these states instead of actual states ( known as indirect filtering). Figure 2.7 shows a loosely coupled example of an indirect filtering in which an observed error is provided to the filter (feed-forward method). As the indirect filter estimates the error in the inertial navigation system, the model results in a linear error model (leading to Kalman

[18]The term vision based navigation (such as Visual-SLAM and Visual Odometry) will be discussed in more detail in Chapter 3.

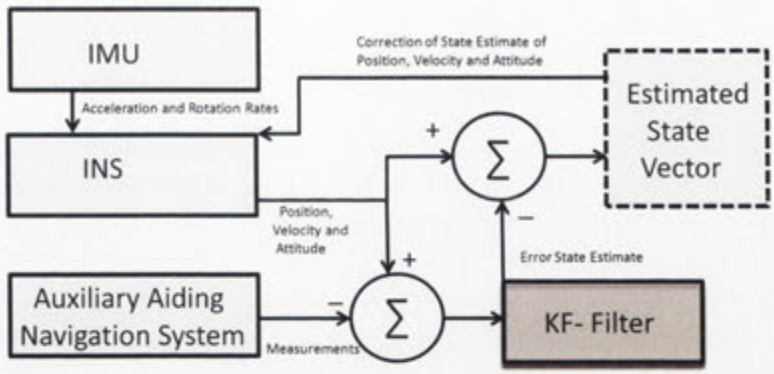



Figure 2.8: The loosely coupled indirect feedback implementation where position, velocity and attitude states of Kalman filter are being estimated.

filter based estimation). The indirect filtering offers the advantage of decoupling the main loop of inertial sensor from the main filter (therefore the prediction stage can run at the same rate as the INS sampling time). However the unbounded error in the INS can cause a huge correction term in the filter, therefore destabilising the filter. To eradicate that problem in the indirect filtering a feedback method is generally implemented (where the estimated error is fed back to the INS system to minimise the inertial drift). The Figure 2.8 shows a loosely coupled indirect feedback filtering approach. Although the indirect approach provides computational efficiency, the direct implementation is generally preferred due to the ease of representation to understand the navigation system [70].

## 2.5 Summary

This chapter provides the Bayesian filter theory as a probabilistic technique to estimate the unknown probability density function using the process and measurement model. The Kalman filter as an optimal estimator for the linear system is discussed along with its extension to the non-linear systems (EKF).

After discussing the statistical filtering, the details of inertial navigation system and design categories of the IMU (gimbaled and strapdown) are provided. This is followed

by the presentation of the different coordinate frames. In addition to inertial sensors, we detail the differential equations known as inertial navigation equations, which are used to estimate the position/velocity and attitude information.

Finally, the integration architecture is provided where the theory of statistical filtering and inertial sensor is utilised to formulate a filter to estimate the states of interest. Each architecture is followed by an example of inertial sensor and visual information aiding. Lastly, the implementation strategies of tightly/loosely coupled modes are presented, and the advantages and disadvantages are discussed.

# Chapter 3

# Monocular Vision aided Inertial Navigation

## 3.1 Introduction

Precise localisation with map estimates of an unknown environment is an essential part of autonomous navigation. Inertial sensors are promising systems to obtain continuous 6DOF information and can track the vehicle's position by integrating the rotational rate and linear acceleration. The pure INS process is inherently unstable due to the integration of noise and bias errors and therefore requires constant assistance (as discussed earlier in Chapter 2). In recent years, the vision sensor has proven to be a complimentary aiding sensor and has appealed to many researchers for Visual-Inertial SLAM [9, 15, 38, 52, 77]. The availability of low-cost inertial and vision sensors and a light-weight and accurate mapping system could be achieved for many robotic tasks such as land/aerial explorations.

Although successful Visual-Inertial approaches exist with monocular [11, 44], stereo [48] sensors, they still face several challenges regarding the availability of reliable and constant aiding information [52] (to correct the inertial system which is intrinsically unstable). The limitations of visual approaches [11, 78] to aid the inertial system

include the handling of the low parallax of features for depth estimation (monocular camera) and the low output data-rates (computational complexity) of the processing algorithm (stereo approaches). The inability of the visual approaches to provide timely updates induce instability in the inertial sensor, hence resulting in the growth of attitude and sensor errors.

In using monocular visual information to aid INS, it is well known that the translational motion from the monocular sensor has a scale-ambiguity problem, providing only the directional information of the motion and not the magnitude [79]. The depth can be resolved by triangulation (if a certain parallax between the two camera views exists). It is generally a delayed process [47] to resolve the feature depth and, therefore, prohibits its fusion with INS unless the depth is resolved at a high frequency. The existing monocular Visual-Inertial SLAM methodologies have been relying on feature based maps [15, 44], which require an accurate depth-resolution process to correct the inertial units properly, and the aiding rate is highly dependent on the map density. In the case of stereo/RGB-D cameras, the possibility of implicit availability of metric scale eliminates some of the challenges encountered when relying only on a monocular camera. However, these sensors act like a monocular when the features exist away from the detection ranges of these cameras (i.e. stereo/RGB-D) or sunlight illumination (i.e. RGB-D).

There were several efforts made in integrating visual flow (or monocular visual odometry) constraint to the inertial systems. One of bottleneck challenges of monocular vision has been the scale ambiguity in its translational motion due to its projective nature in sensing, preventing it from being fused with other metric navigational sensors such as IMU. That is, a monocular vision system can only provide translational motion up to scale. To relieve this problem, additional information could be used, such as known camera height from the ground [9], use of pressure sensors [18] or feeding the initial scale information by hand [12] in land/aerial vehicle navigation. In [21], navigation is performed in topological scale space rather than in a metric space using monocular vision, whereas rotational information from the inertial sensor is used for vision based pose estimation. Earlier works [1, 2] have exploited the

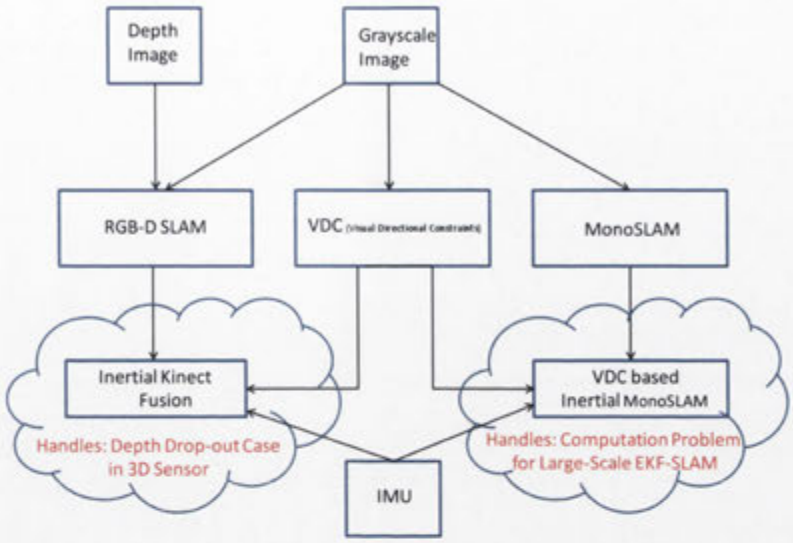



Figure 3.1: Visual non-holonomic constraints for Inertial Monocular and RGB-D SLAM

non-holonomic constraints for land vehicles in inertial aiding. However, in all-terrain or aerial applications the non-holonomic conditions are violated or unavailable.

Therefore to utilise the aiding of information at higher rates (for inertial system stability), this work proposes a novel visual non-holonomic constraint [1]. The basic concept is that the tangential velocity components of INS along the direction of visual-flow motion should be zero (this is similar to the non-skidding constraint in wheeled robots [1, 2] [2].

The vision based non-holonomic constraints from a monocular camera are useful to the monocular-SLAM to solve the computational problem for EKF monocular SLAM [80, 81] . The monocular SLAM generally estimates the position of the vehicle and landmarks jointly. The fundamental problem with these approaches is that the com-

[1]These visual non-holonomic constraints are called visual directional constraints hereafter in this thesis. These constraints are estimated from consecutive image sequence and are capable of providing the ego-motion estimates at frame-rate.

[2]Earlier works have exploited the non-holonomic constraints for land vehicles in inertial aiding. However in all-terrain or aerial applications the non-holonomic conditions are violated or unavailable whereas our proposed visual directional constraint could be utilised in a similar context.

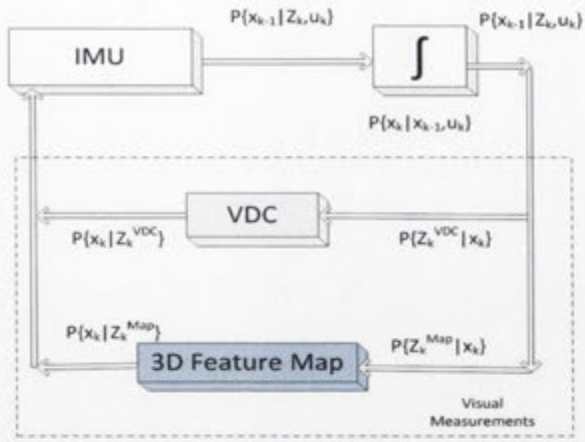



Figure 3.2: Multiple loop aiding architecture with a VDC aiding inner loop to stabilise inertial system and an outer SLAM loop with a periodic feature updates.

putational requirements scale quadratically in the number of landmarks [81]. Therefore, this limits the number of visual features to only a few hundreds and hence creates a compromise on the available information to constrain the vehicle's pose. This work estimates the visual directional constraints at frame rate (using hundreds of features from the consecutive images –visual odometry) in conjunction to few inverse depth features (monocular SLAM). Figure 3.1 provides the use of the visual non-holonomic constraints fusion with monocular SLAM constraints.

In this chapter, we will be discussing a monocular inertial SLAM approach, improving the stability and robustness of the inertial system. The approach is based on a novel two-stage integration of visual constraints and feature based SLAM. The unknown scale problem is resolved by firstly fusing the visual direction constraint in a seamless way and then a sparse feature map is used to further constrain the errors along the longitudinal direction. The proposed approach is evaluated with a SLAM benchmark dataset available on-line and with simulations. The result shows that inertial errors can be constrained effectively using VDC, which relieves the burden of SLAM, aiding loop and thus opening up the possibility of reduced map size while maintaining the inertial stability and performance. Figure 3.2 shows the multi-loop aiding architecture where a VDC based inner loop enables constant updates to stabilise the inertial

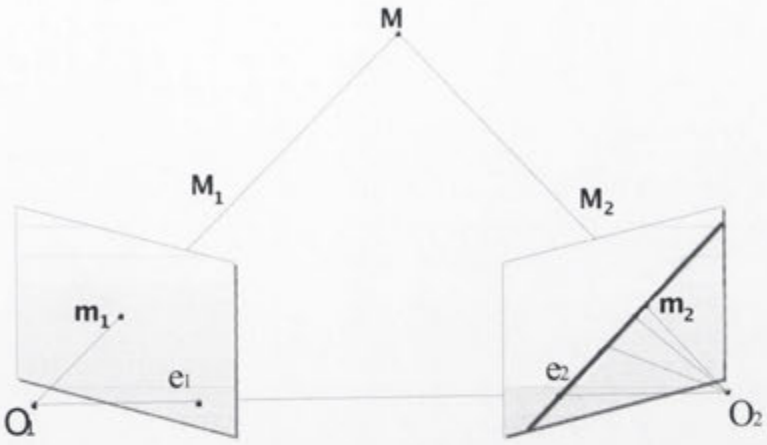



Figure 3.3: Perspective projection of 3D feature for two camera views defining an epipolar geometry.

system while the outer SLAM loop provide aperiodic feature updates. In our work we have focused on the challenging indoor/outdoor environments with a front-facing camera, [15] where depth variations vary frequently, making the overall problem more challenging.

## 3.2 Monocular Pose Estimation

In computer vision, the corresponding pixel coordinates of the features in the scene (from two or more cameras) are used to estimate the relative pose information. For relative pose estimation techniques, it is generally sufficient to use a single camera which could move or a single static camera which looks at a scene involving moving objects. A camera creates a 2D representation of a 3D scene by way of imaging a projection of the light emitted or reflected from objects in the scene. The simplest of cameras can be described as a pinhole camera. Such a camera is described as having a focal plane at a fixed distance $\Gamma$ (the focal length) in front of an image plane. A pinhole, referred to as the optical centre $O$, is made such that rays of light arriving at the camera from the scene form an inverted image of the scene on the image plane. Thus, for each point on the object, there is a corresponding point made on the image

plane. This manner of projection from the 3D scene to the 2D image plane is referred to as perspective projection.

Let $M$ be a 3D point $(M_x, M_y, M_z)^T$), its projection in camera 1 is $m$ $(m_x, m_y)^T$), the mapping from the coordinates of a 3D point $M$ to the 2D image coordinates of the point's projection onto the image plane, according to the pinhole camera model is given by

$$\begin{bmatrix} m_x \\ m_y \end{bmatrix} = \frac{\Gamma}{M_z} \begin{bmatrix} M_x \\ M_y \end{bmatrix} \tag{3.1}$$

where ($M_x$, $M_y$, $M_z$) are the 3D coordinates of $M$ relative to a camera centered coordinate system, ($m_x$, $m_y$) are the resulting image coordinates for which we assume $\Gamma > 0$.

To derive the camera matrix this expression is rewritten in terms of homogeneous coordinates. Instead of the 2D vector ($m_x$,$m_y$) we consider the projective element (a 3D vector) $\hat{m} = (m_x, m_y, 1)^T$ and $\hat{M} = (M_x, M_y, M_z, 1)^T$ . Instead of equality we consider equality up to scaling by a non-zero number (denoted as $\simeq$) . First, we write the homogeneous image coordinates as expressions in the usual 3D coordinates.

$$\hat{m} \simeq \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{\Gamma} & 0 \\ & & & \end{bmatrix} \hat{M}$$

Which can be simplified by replacing intrinsic parameters matrix with $k$ as

$$\hat{m} \simeq k\hat{M} \tag{3.2}$$

However for preforming an analysis on multiple images of the same scene that have been taken from different viewpoints, it is necessary to ensure that the scene's origin is constant across all images. In order to achieve this, it is necessary to know the extrinsic camera parameters for a set of two images. These consist of the translation of the camera and the rotation of the relative to the origin of the world coordinate system. As rotation and translation can be expressed simply in linear algebra, it is convenient to rewrite the relationship as:

$$M_{cam} = R^T(M_{world} - t) \tag{3.3}$$

Where $M_{cam}$ is the 3D camera coordinate and $M_{world}$ is the world coordinate. Note that $t \in \mathbb{R}^3$ and $R \in SO(3)$.

So far, we have described how a camera is able to map points from a 3D world coordinate system, to a 2D image coordinate system - taking into account how the camera lies in relation to the scene (the extrinsic parameters). Now we will consider how this relates to a series of two images of the same scene - where the images are taken by the same camera that has undergone a translation and rotation around a scene. The geometry of how these images relate to one another, and how the 3D coordinates of a scene are represented by two 2D images is known as the epipolar geometry of a scene. Considering a two calibrated cameras viewing a rigid scene, it is possible to estimate the relative rotation and translation, only using the five image feature point correspondence. Consider two cameras, one at the origin and the other has a relative rotation $R$ and translation $t$ as shown in the Figure 3.3, with camera centres $O_1$ and $O_2$. Then the vectors between $M$ and the two camera centres are ($O_1$ - $M$) and ($O_2$ - $M$). Because all vectors lie in the same plane, the cross product between t and ($O_2$ - $M$) (the normal of the plane) is perpendicular to ($O_1$ - $M$) as

$$(O_1 - M)^T \ (t \times (O_2 - M)) = 0 \tag{3.4}$$

By using the knowledge that $m_1$ is parallel to ($O_1$ - $M$) and R$m_2$ is parallel to ($O_2$ - $M$), it is possible to write the above Equation 3.2 as using the earlier projection Equation 3.2

$$m_1{}^T \ (t \times Rm_2) \ = \ 0 \tag{3.5}$$

By introducing the antisymmetric matrix T

$$\bar{T} \ = \ \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix}$$

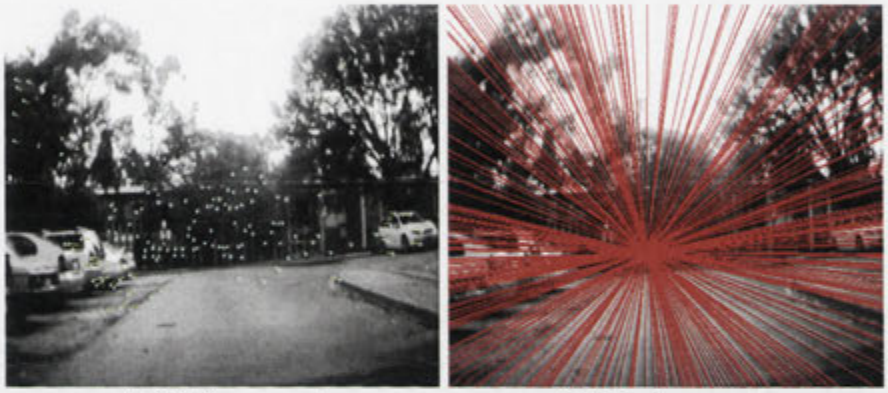
for the cross product yields $(t \times s) = \bar{T}\ s$, Plugging this in the above equation yields

$$m_1{}^T \ (\bar{T}\ R\ m_2) \ = \ 0 \tag{3.6}$$

Where the $\bar{T}R$ are the essential matrix constraint [79]. The rotation matrix has 3DOF and translation can only have recovered up to an unknown scale factor. Setting the length of translation vector to some arbitrary unit gives two DOF for the $\bar{T}$, overall resulting in 5DOF. Each correspondence point among the pair of 5 feature points yield one constraint. For 5 unknowns there are five points corresponding points are required. The epipolar constraint is simply a co-planarity constraint where the point $m_1, m_2$ is the 2D image points of the 3D point M in the scene from two different views. The points $e_1$and $e_2$ are referred to as the epipoles of their respective images, and can be described as the image of the projection centre ($O$) of one camera in the other. Epipoles are therefore intersections of the line $O_1$ from $O_2$ (which can be referred to as the baseline) with the respective image planes.

(a) Input image

(b) Feature matching between consective images



(c) Outlier removal

(d) Epipolar geometry

Figure 3.4: Relative pose estimation using 5-point RANSAC based outlier removal.

To estimate the pose from the monocular camera image, firstly the Harris features [53] and SURF descriptors [32] are extracted. The feature points from the current image are matched with those of the previous image using a 5-point algorithm with RANSAC processing [79]. The inliers are then used to extract delta rotation and translation over two consecutive images. Figure 3.4 shows a relative pose estimation using 5-point RANSAC based algorithm. The vision node outputs rotational rate and translational velocity (up to scale) in the camera coordinates (the calibration between inertial and camera is performed using [52], which helps to covert all the camera coordinate measurements into the inertial/body coordinates). Without loss of generality, we assume the camera and body coordinates are aligned each other.

To fuse these with the inertial sensor outputs, which operate in a metric space, both inertial and visual velocities are converted into unit velocities.

**Algorithm 1** Relative pose estimation using 5 point RANSAC algorithm

inputs: point correspondence
**while** iterate till max iteration are not reached **do**
Randomly select 5 points from the correspondence set
Estimate the essential matrix using [79]
For estimated essential matrix estimate the projection error for all the other corresponding points to select supported inliers
**if** Supported Inliers are > max-inliers **then**
$max - inliers = supportedinliers$
Store all the inliers
**end if**
**end while**
Estimate essential matrix from all the stored supported inliers with selection of correct solution using [79]
**return** $R, \bar{T}$

## 3.3 General Motion

The general motion of a vehicle in a 3-dimensional space can be described by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{3.7}$$

where the state vector $\mathbf{x} = [\mathbf{P}^T, \mathbf{V}^T, \phi, \theta, \psi]^T$, and the measurement $\mathbf{u} = [\mathbf{f}_b^T, \omega_b^T]^T$.

The kinematic equation can be written

$$\dot{\mathbf{P}}^n = \mathbf{V}^n \tag{3.8}$$

$$\dot{\mathbf{V}}^n = \mathbf{C}_b^n \mathbf{f}^b + g \tag{3.9}$$

$$\dot{\phi} = \frac{\omega_y \sin\psi + \omega_z \cos\psi}{\cos\theta} \tag{3.10}$$

$$\dot{\theta} = \omega_y \cos\psi - \omega_z \cos\psi \tag{3.11}$$

$$\dot{\psi} = \omega_x + (\omega_y \sin\psi + \omega_z \sin\psi)\tan\theta \tag{3.12}$$

## 3.4 Motion of a Vehicle with Visual Constraint

There have been three approaches to deal with this unknown scale:

- The pose can directly be used for navigation but in the unknown scale-space as in many pure vision-based approaches such as Visual-SLAM or Rat-SLAM [82].
- The unknown scale value can be recovered by observing any a prior infrastructural information such as visual markers or known vehicle height [83].
- The unknown scale can be estimated using other aiding sensors such as pressure sensor [18] or inertial sensors which integrate the sensor measurements to provide the scale information [4, 44]. The work of [4] used visual inertial measurements in an EKF filter to estimate the scale in addition to the pose information of the platform. There approach depends upon the external SLAM (PTAM) approach which assumes smooth depth variation [15] and requires initial excitation motion for filter convergence.
- Explicit scale resolution by treating the visual translation (with unknown scale) as a directional constraint of the motion (this section and our earlier work [17]).

The third and fourth option seems most relevant to our system. In the third option the initial visual scale needs to be known, [3] Whereas our work [17] enables the direct fusion of inertial and vision output (without any a-prior knowledge about the scale).

[3] Two major issues are 1) assumption of smooth depth variation and 2) filter convergence for depth estimation.

As discussed earlier that due to perspective projection of the 3D into 2D image plane, the recovery of the absolute depth is not possible. Hence the translation estimated from the monocular vision is normalised or unit vector (only providing directional information with unit magnitude). To fuse the visual (non-metric) and inertial (metric) information, in this section we have devised an approach which expresses both the translation of inertial and visual as unit vector for fusion without explicitly estimating the scale of visual magnitude. In this work we utilised the *directional constraint of the motion* for our loosely coupled framework to integrate the monocular pose measurements with inertial information.

The unit velocity in the body frame $\mathbf{V}^b = [V_x, V_y, V_z]$ is related to the velocity is the navigation frame by

$$\mathbf{V}^b = [\mathbf{R}_b^n]^T \mathbf{V}^n \tag{3.13}$$

where $\mathbf{R}_b^n$ is re-written here for clarity

$$\mathbf{R}_b^n = \begin{bmatrix} C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi \\ C_\theta S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \tag{3.14}$$

Thus the velocity equation becomes

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} C_\theta C_\psi V_N + C_\theta S_\psi V_E + (S_\phi S_\psi - S_\theta V_D \\ (-C_\phi S_\psi + S_\phi S_\theta C_\psi) V_N + (C_\phi C_\psi + S_\phi S_\theta C_\psi) V_E + S_\phi C_\theta V_D \\ (S_\phi S_\psi + C_\phi S_\theta C_\psi) V_N + (-S_\theta C_\psi + C_\phi S_\theta S_\psi) V_E + C_\phi C_\theta V_D \end{bmatrix}. \tag{3.15}$$

The unit velocity vector, thus becomes

$$\mathbf{U}^b = \frac{\mathbf{V}^b}{||\mathbf{V}^b||}. \tag{3.16}$$

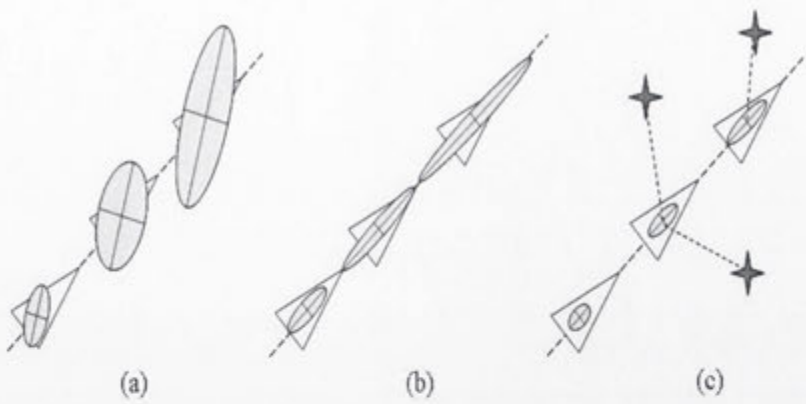



Figure 3.5: Effects of visual directional constraints aiding on position covariance: a) unaided inertial system, b) VDC aided inertial system and c) VDC and map aided inertial system.

Unlike the ground non-holonomic constraint, the visual motion can constraint the motion to general $2D$ surface. For example, if the vehicle moves only in forward direction in the body frame, and assuming the camera frame is aligned to the body frame, the lateral unit velocities becomes zero which are related to the vehicle state

$$\begin{bmatrix} U_y \\ U_z \end{bmatrix} = \frac{1}{\sqrt{V_y^2 + V_z^2}} \begin{bmatrix} C_\theta S_\psi V_N + (C_\phi C_\psi + S_\phi S_\theta C_\psi) V_E + (S_\theta C_\psi + C_\phi S_\theta S_\psi) V_D \\ S_\theta V_N + S_\psi C_\theta V_E + C_\phi C_\theta V_D \end{bmatrix} \quad (3.17)$$

It can be observed that

- This case is analogues to the non-holonomic constraint, restricting the lateral motions to zero (or skidding is not allowed).
- The vision measures the forward velocity up to scale, the forward velocity $V_x$ is not directly recoverable. It is somewhat similar that a mobile robot experiences wheel slippages, thus unable to measure the forward velocity.
- These effects on the vehicle uncertainty ellipsoid is illustrated in Figure 3.5, in which the lateral uncertainties are constrained, whilst the longitudinal direction diverging. Using other methods, such as mapped landmarks, the longitudinal error can be estimated. Figure 3.5 shows that the VDC does not rely on

3D depth estimation or scale information of the environment and thus can be seamlessly integrated into inertial system. As the constraint provides only the directional information without the magnitude, the error will eventually grow across the vehicle's longitudinal direction. This error can be further observed by maintaining the sparse 3D map in the inertial loop but at lower update rates.

If the vehicle moves to positive vertical direction

$$\begin{bmatrix} U_x \\ U_y \end{bmatrix} = \frac{1}{\sqrt{V_x^2 + V_y^2}} \begin{bmatrix} C_\theta C_\psi V_N - (C_\phi S_\psi + S_\phi S_\theta C_\psi) V_E + (S_\phi S_\psi + C_\phi S_\theta C_\psi) V_D \\ C_\theta S_\psi V_N + (C_\phi C_\psi + S_\phi S_\theta C_\psi) V_E + (S_\theta C_\psi + C_\phi S_\theta S_\psi) V_D \end{bmatrix} \quad (3.18)$$

- It is clear that the visual directional measurement can constrained the horizontal motion $(V_x, V_y) = 0$. item Thus it is possible to direct the vehicle in general to suppress the uncertainties normal to the direction of motion. For example, the reduce the uncertainty in horizontal direction under the visual measurement, the robot can undergo any vertical motions.
- This is important in that the vehicle can be aided in high data rate, without estimating the unknown scale factor (or resolving the depth of features).

If the vehicle motion is constrained to the ground, the so called non-holonomic constraint could be applied [64]. For example the unit $y$ and $z$ velocity components are set to zero ($\widetilde{v}_y = 0, \widetilde{v}_z = 0$). In the general case, such as hovering flying vehicles, this type of constraints cannot be applied. However the vision system can still provide such motion constraints: the lateral and normal velocities are constrained by the visual velocity.

## 3.5 Visual Constraint Aiding

The incremental rotation matrix $\mathbf{R}_{t-1}^{t}$ is directly measurable from the monocular vision. The incremental Euler angles can be computed as from Equation 3.4,

$$\Delta\phi = \arctan 2(R_{31}, R_{32}) \tag{3.19}$$

$$\Delta\theta = \arccos(R_{33}) \tag{3.20}$$

$$\Delta\psi = \arctan 2(R_{13}, R_{23}) \tag{3.21}$$

The directional information puts a bound on the motion of camera and helps in limiting the inertial prediction errors of camera motion by periodic updates.

### 3.5.1 Accelerometer and Magnetometer Aiding

IMU as an inclinometer provides independent measurements of the accelerometers to obtain roll and pitch angles, referred as gravity normal [53]. As the accelerations have zero-mean in the long run, the absolute information about the direction of gravity provides the roll $\phi$ and pitch $\theta$ angles

The gravity vector in navigation frame $\mathbf{g} = [0, 0, -g]^T$ where $g = 9.8m/s^2$ is measured by accelerometers in the body frame

$$\mathbf{f}^b = [\mathbf{R}_b^n]^T \mathbf{g}^n \tag{3.22}$$

In component form,

$$\begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} C_\theta C_\psi & C_\theta S_\psi & -S_\theta \\ -C_\phi S_\psi + S_\phi S_\theta C_\psi & C_\phi C_\psi + S_\phi S_\theta C_\psi & S_\phi C_\theta \\ S_\phi S_\psi + C_\phi S_\theta C_\psi & -S_\theta C_\psi + C_\phi S_\theta S_\psi & C_\phi C_\theta \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} = \begin{bmatrix} g\sin\theta \\ -g\sin\phi\cos\theta \\ g\cos\phi\cos\theta \end{bmatrix} \tag{3.23}$$

From these equations, the roll angle $\phi$ can be recovered by dividing $y-$component by $z-$component

$$\phi = \arctan\left(-f_y/f_z\right) \tag{3.24}$$

For the pitch angle $\theta$, first eliminate the $\phi$ by squaring and summing $y$ and $z$ components gives $f_y^2 + f_z^2 = g^2\cos^2\theta$. Dividing the $x-$component by this yields

$$\theta = \arctan\left(f_x/\sqrt{f_y^2 + f_z^2}\right) \tag{3.25}$$

The magnetometer measures the Earth magnetic field $\mathbf{o}^b = [o_x, o_y, o_z]^T$ in the body coordinate system. The heading angle $\psi$ can be found from this measurement first by transforming the measurement into a Local-Level vector $\mathbf{G}^{LL} = [G_x, G_y, G_z]^T$,

$$\mathbf{G}^{LL} = \mathbf{R}_b^{LL}\mathbf{o}^b \tag{3.26}$$

where $\mathbf{R}_b^{LL}$ is a transformation from the body to Local-Level frame which can be obtained by setting $\psi = 0$ in $\mathbf{R}_b^n$ (thus no change in the heading during the transformation)

$$\mathbf{R}_b^{LL}(\phi, \theta, \psi = 0) = \begin{bmatrix} C_\theta & S_\phi S_\theta & C_\phi S_\theta \\ 0 & C_\phi & -S_\theta \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \tag{3.27}$$

Thus, Equation 3.26 becomes

$$\begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} = \begin{bmatrix} o_x\cos\theta + o_y\sin\phi\sin\theta + o_z\cos\phi\sin\theta \\ o_y\cos\phi\sin\theta - o_z\sin\phi \\ -o_x\sin\theta + o_y\sin\phi\cos\theta + o_z\cos\phi\cos\theta \end{bmatrix}. \tag{3.28}$$

The $z-$component of the magnetic field in the local-level frame is due to the magnetic

inclination which becomes 90° (or straight down) at the North magnetic pole. The heading with respect to the true North can be found from the horizontal magnetic components with a declination angle $\psi_D$

$$\psi = \arctan\left(G_y/G_x\right) - \psi_D \tag{3.29}$$

where the declination angle in Australia/Canberra region is approximately 5.4° East (The magnetic pole deviates towards East from the true North.).

From Equations 3.24, 3.25 and 3.29, the estimated rotation angles constraints are estimated (the usage of these constraints is described in Section 3.7).

## 3.6 3D map integration

Monocular camera provides only bearing information so multiple observation are required to localise a feature in 3D space. Initialisation of features without any delay is particularly challenging due to the requirement of large baseline (or enough parallax) but with limited field-of -view and short observation time. To initialise the features smoothly in all depths without any delay, we integrate the Inverse Depth Parameterisation (IDP) method [35] with the visual directional constraints.

Direct integration of inertial sensor with Inverse Depth Parameterisation (IDP) faces with a challenge of filter update time, which scales quadratically with the number of features whereas by using the visual directional constraints as a prior aiding step to IDP integration suggests that, we can afford to maintain sparse number of features in our framework. The representation of the features in the SLAM filter contains the feature's IDP encoding [35] scheme as:

$$\bar{\mathbf{m}} = \begin{bmatrix} \mathbf{p}_m^n \\ \varphi^n \\ \beta^n \\ \alpha^n \end{bmatrix} \tag{3.30}$$

where $\mathbf{p}_m^n$ is the position of platform, at the time when feature $\bar{m}$ is firstly observed, $\varphi^n$ is the azimuth and $\beta^n$ elevation of optical ray. $\alpha^n$ represents the inverse distance of the camera and measured feature.

The observed point on the image sensor defines a directional ray in the camera reference frame. The directional ray can be calculated from the information present in the state vector and inverse depth features as:

$$\mathbf{h}^c = \left[ \mathbf{R}_b^c \, \mathbf{R}_n^b \, \left( \, (\mathbf{p}_m^n + \tfrac{1}{\alpha^n} \eta(\varphi^n, \beta^n) \,) - \mathbf{p}^n \, \right) \right] \tag{3.31}$$

where $\mathbf{h}^c \in (\mathbf{h}_x^c, \mathbf{h}_y^c, \mathbf{h}_z^c)$ is directional ray and the translation offset between camera and body frame is assumed to be negligible/zero. The unit vector pointing from the camera's optical centre to feature $\mathbf{m}$ is defined as follows:

$$\eta(\varphi^n, \beta^n) = \begin{bmatrix} \mathbf{cos}(\varphi^n) \, \mathbf{cos}(\beta^n) \\ \mathbf{sin}(\varphi^n) \, \mathbf{cos}(\beta^n) \\ \mathbf{sin}(\beta^n) \end{bmatrix} \tag{3.32}$$

The observation for inverse depth in relation to state vector is defined as:

$$\begin{aligned} p\left(\mathbf{x}_k \mid \mathbf{z}^{MAP}\right) \Leftrightarrow \mathbf{z}^{MAP} &= h^{Map}(\mathbf{x}, \mathbf{m}) + v_k^{MAP} \\ &= \begin{bmatrix} tan^{-1}(\frac{\mathbf{h}_y^c}{\mathbf{h}_x^c}) \\ tan^{-1}(\frac{\mathbf{h}_z^c}{\sqrt{(\mathbf{h}_x^c)^2 + (\mathbf{h}_y^c)^2}}) \end{bmatrix} + v_k^{MAP} \end{aligned} \tag{3.33}$$

Where $\mathbf{v}_k^{MAP}$ is a zero-mean measurement noise with covariance $\mathbf{R}^{MAP}$. The observation vector for compatible undistorted feature matches $\mathbf{u}, \mathbf{v}$ are estimated as:

$$\tilde{\mathbf{z}}^{\mathbf{MAP}} = \begin{bmatrix} \tan^{-1}\left(\frac{\mathbf{u}-\mathbf{u_x}}{\mathbf{f_x}}\right) \\ \tan^{-1}\left(\frac{\mathbf{v}-\mathbf{v_y}}{\mathbf{f_y}}\right) \end{bmatrix} \tag{3.34}$$

Where $\mathbf{u_x}, \mathbf{v_y}, \mathbf{f_x}, \mathbf{f_y}$ are intrinsic parameters of a camera.

The work of [35] is modified to predict features position based upon inertial aiding. Data association is based upon feature matching with outlier removal [84] using in-

ertial information. A single random measurement which supports maximum number of hypothesis is selected by RANSAC to update the inertial-EKF filter by estimating the innovation vector from the predicted feature positions. The innovation vector for 3D map features for EKF filter update is as follows:

$$\nu^{\mathrm{MAP}} = \left[ \tilde{\mathbf{z}}^{\mathrm{MAP}} - \mathbf{h}^{\mathrm{MAP}}(\tilde{\mathbf{x}}_{\mathbf{k}}, \mathbf{m}) \right] \tag{3.35}$$

## 3.7 Filter Updates for 6DOF and 5DOF Constraints

The state covariance is propagated using the Jacobian of the state transition model and process noise matrix by,

$$\mathbf{P}(k|k-1) = \nabla \mathbf{f}_{\mathbf{x}}(k)\mathbf{P}(k-1|k-1)\nabla \mathbf{f}_{\mathbf{x}}{}^{T} + \nabla \mathbf{f}_{\mathbf{w}}(k)\mathbf{Q}(k)\nabla \mathbf{f}_{\mathbf{w}}{}^{T}(k),$$

When an observation occurs, the state vector and its covariance are updated according to

$$\begin{aligned} \tilde{\mathbf{x}}(k|k) &= \tilde{\mathbf{x}}(k|k-1) + \mathbf{W}(k)\nu(k) \\ \mathbf{P}(k|k) &= [\mathbf{I} - \mathbf{W}(k)\nabla \mathbf{h}_{\mathbf{x}}(k)]\mathbf{P}(k|k-1) \\ &\quad \times [\mathbf{I} - \mathbf{W}(k)\nabla \mathbf{h}_{\mathbf{x}}(k)]^{T} + \mathbf{W}(k)\mathbf{R}(k)\mathbf{W}^{T}(k), \end{aligned}$$

where the innovation vector, Kalman gain, and innovation covariance are computed as,

$$\begin{aligned} \nu(k) &= \mathbf{z}(k) - \mathbf{h}(\tilde{\mathbf{x}}(k|k-1)) \\ \mathbf{W}(k) &= \mathbf{P}(k|k-1)\nabla \mathbf{h}_{\mathbf{x}}{}^{T}(k)\mathbf{S}^{-1}(k) \\ \mathbf{S}(k) &= \nabla \mathbf{h}_{\mathbf{x}}(k)\mathbf{P}(k|k-1)\nabla \mathbf{h}_{\mathbf{x}}{}^{T}(k) + \mathbf{R}. \end{aligned}$$

$\nabla \mathbf{h}_{\mathbf{x}}(k)$ is the Jacobian of the non-linear state transition function $\mathbf{h}(\cdot)$ with respect to the predicted state $\widetilde{\mathbf{x}}(k|k-1)$.

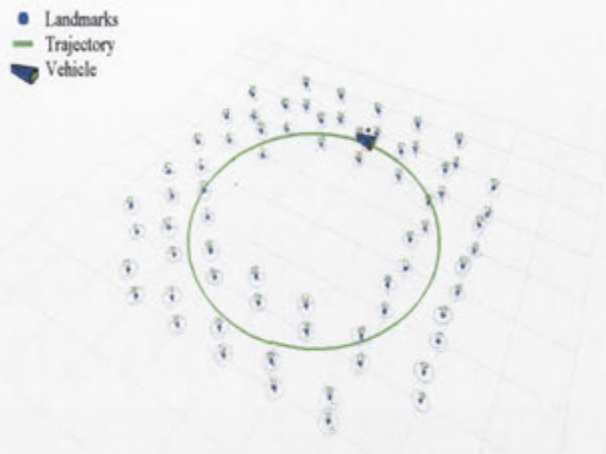
The multi-loop aiding architecture is used for EKF filter update as shown in Figure 3.2. The VDC based inner loop (Section 3.4) enables constant updates to stabilise the inertial system while the outer SLAM loop (Section 3.6) provide aperiodic feature updates. If the inverse depth features are visible then 3D map constraints are used for EKF-update step (otherwise monocular VDC constraints are utilised). We have bounded the absolute attitude error of flying vehicle by using the gravity normal (Section 3.5) [4].

## 3.8 Simulations Evaluation

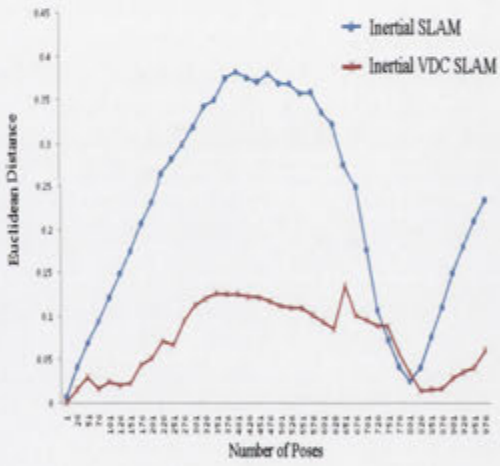
The proposed approach is evaluated using a simulation scheme in which a camera is mounted in a forward-looking way so that it can observe the features at far distance. The vehicle trajectory is generated using the inertial process model with a linear acceleration of [0.004, 0, 0] $m/sec^2$ and an angular rate of [0, 0, 0.45] deg/sec during turning with standard deviations of respective noise (to simulate noisy measurements) as [0.003, 0.003, 0.003] $m/sec^2$ and [0.03, 0.03, 0.03] deg/sec. The noise is added to mimic a camera of $640 \times 480$ pixels is assumed with fixed radial distortion parameters.

A total of 73 features are simulated and the inverse depth method was used to initialise and map as shown in Figure 3.6a. Initially the prior depth of feature is unknown and thus all the features are considered to be close to infinity and the inverse depth parameter was set to 0.1 as an initial approximation with standard deviation of 0.5 including infinity at its range. The VDC constraints are simulated using these features by estimating the essential matrix. The comparison of the proposed approach is evaluated against the work of [85] and simulated ground truth. Figure 3.6b compares the positional errors of conventional inertial-SLAM approach [85] with proposed approach.The difference in vehicle position from the ground truth position is taken for

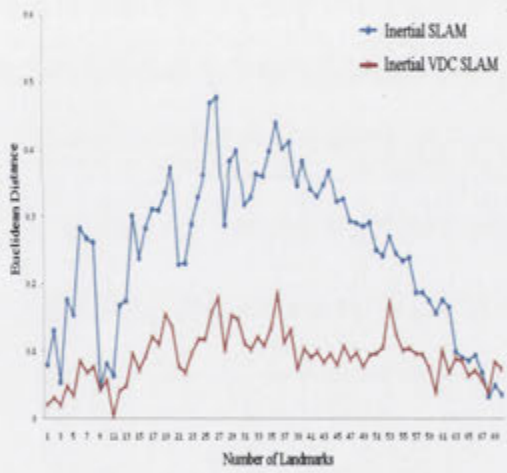
[4]These constraints are only added when static or near static state of flying platform is detected [73]




(a) Simulated Environment




(b) Vehicle/Camera Euclidean Position Error from GT




(c) Landmark Position Error from GT

Figure 3.6: Simulation Environment, with Vehicle Pose and detected Landmarks.

complete simulation time. The positional error increases as we explore the new environment whereas the drop in error starts occurring when the features are re-observed again (closing the loop by manually simulating the loop closure event). It can be observed that the decrease in error is rather gradual than abrupt due the inverse depth parameterisation. This is due to the lower weighting in depth compared to the bearing for features at far distance. The errors of the estimated position of features from the ground truth is shown in Figure 3.6c.

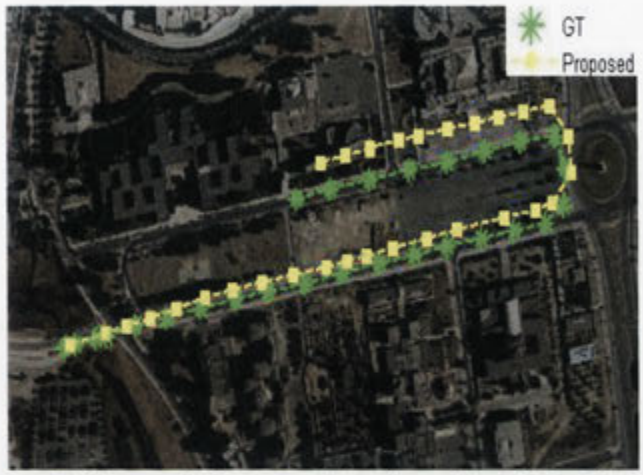
## 3.9 Public Dataset evaluation

Publicly available dataset [86] has been chosen to prove the validity of proposed approach. We have used two different sequence to evaluate the performance of proposed procedure against the ground truth of 1.0 cm, 0.5° pose accuracy. During sequence evaluation, we have reduced the resolution of monocular imagery to $640 \times 480$ for real time performance.

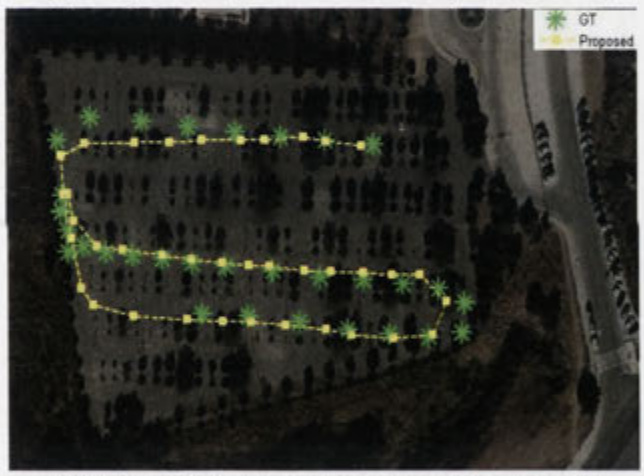
The proposed approach is tested on 2102 images of Campus-0L (4.53 minutes) in which vehicle has travelled trajectory of $1001m$ with ground truth available for complete sequence with 2 sharp turns along the trajectory. Figure 3.7 illustrates the vehicle localisation results against the ground truth on Campus-0L datasets. Attitude comparison is also shown in Figure 3.8. Where mean attitude error is [0.274° , 0.805°, 0.678°]. Due to road like scenarios, it can be easily observed that roll/pitch standard deviations are in small magnitude along the way whereas yaw angle has depicted two sharp turns.

Parking-0L sequence is also tested for 1022 images (3.16 minutes) for 324m trajectory, in which ground truth is not available at smaller portion as shown in the Figure 3.7b where ground truth position's are missing.

Overall the proposed approach shows resilience to high speed and sharp turns due to inertial dynamic model with stable integration to visual information.




(a) Campus-0L Sequence: Mean Error (24m E, 15m N)




(b) Parking-0L Sequence: Mean Error (11m E, 7m N)

Figure 3.7: Estimated vehicle trajectory for SLAM benchmark dataset of 1001m and 324m length respectively, with mean positional error between estimated and ground truth trajectories.

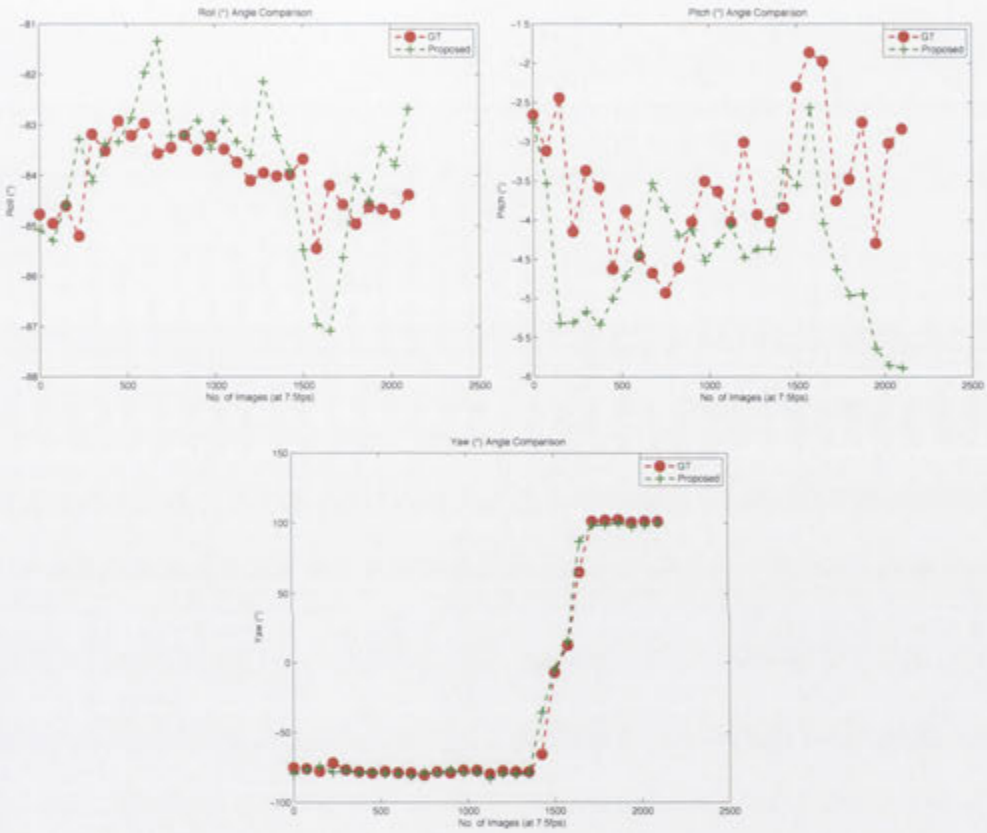



Figure 3.8: Comparison of the estimated attitude with ground-truth, mean attitude error (0.274° , 0.805°, 0.678°).

## 3.10 Discussions and Summary

In this chapter, we proposed a monocular inertial SLAM approach, improving the stability and robustness of the inertial system. The approach is based on a novel two-stage integration of visual constraints and feature-based SLAM. The unknown scale problem was resolved by fusing the visual direction constraint in a seamless way and then a sparse feature map was used to further constrain the errors along the longitudinal direction. The proposed approach was evaluated with a SLAM benchmark dataset available on-line and with simulations. The result shows that inertial errors can be constrained effectively using VDC, which relieves the burden of SLAM aiding loop thus opening up the possibility of reduced map size while maintaining the inertial stability and performance.

# Chapter 4

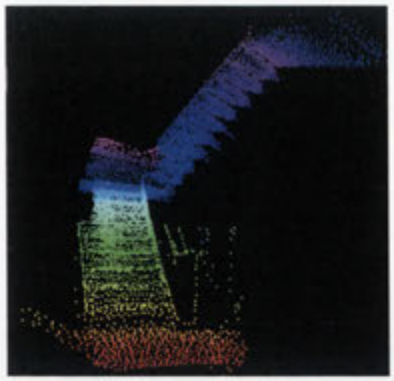
# RGB-D aided Visual-Inertial SLAM

## 4.1 Introduction

RGB-D sensors have revolutionised robotic mapping and navigation research. These sensors are capable of providing RGB and depth data which has various application in 3D reconstruction, gaming, indoor localisation and mobile robotics. Its depth sensor consists of a projector that emits a known pattern, which is detected by an infrared camera (structure light based depth estimation [87]). Although quite successful, their usage is significantly hampered due to the unavailability of depth data. The problem of insufficient or no depth data from the sensor is called here *a depth dropout problem*, hence making the RGB-D sensor act as a monocular camera. The degeneration of the RGB-D sensor into a monocular camera is either due to the features being beyond the range or because of intensive sunlight interference.

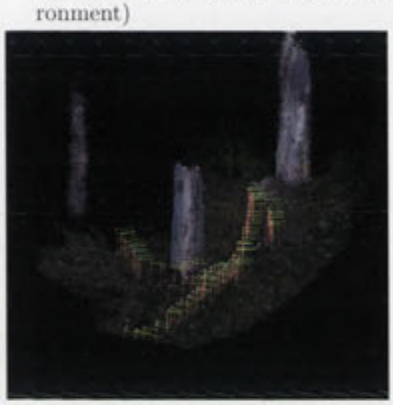
In the RGB-D depth dropout case, the vision system can still provide 5DOF pose information: the rotation and translation with an unknown scale $(R, \bar{T})$. The monocular vision in conjunction with RGB-D/Stereo visual cues provides a broader observability of features (the linear/angular information available at frame-rates) [47, 88].

(a) Input image from staircase sequence



(b) 3D map generated while walking through the staircases (indoor environment)

(c) Input image from outdoor (forest) sequence



(d) 3D map generated

Figure 4.1: Proposed Approach able to tackle both indoor and outdoor environments.

Visual directional constraints from a monocular camera provide a systematic integration with the colour-depth sensors when the depth information is not available (Our earlier work [73]).

This work addresses the depth dropout problem by proposing a novel mechanism that can switch to 2D sensing mode (monocular cues) in case of the unavailability of 3D information. The contributions of this work are twofold. Firstly, a vision node is designed which can provide either full 6DOF (rotation and translation) or partial 5DOF (rotation and scale-ambiguous translation) vehicle pose measurements to the

inertial sensor in an Extended Kalman Filter (EKF) [1].

Secondly the visual translation from monocular vision (5DOF non-metric measurements) is fused with the inertial sensor (metric measurements) without explicitly resolving the ambiguous scale. Therefore, this fusion mechanism ensures the inertial sensor to be aided without delay or under small parallax motion. We have tested our proposed approach on a low-cost inertial sensor [2], Kinect sensor and a custom built hexacopter platform. In addition, we have dynamically calibrated the accelerometer/gyro biases of the low-cost inertial sensor using the proposed pose estimator. In our work we adopted the concept of gravity normal [53] at the hovering state of the aerial vehicle to bound the overall attitude error for stable autonomous flight [3]. Our approach is applicable to active/passive RGB-D sensors (stereo and Kinect both providing depth and colour data). Here we opted to use the Kinect sensor (active RGB-D) due to its computational efficiency in getting depth data for on-board real-time processing and it is more challenging to deal with outdoor environments (as it is known to fail regularly under sunlight interference).

In this chapter, the theoretical and practical development of an approach that can handle the 3D to 2D degeneration of the RGB-D sensor using Visual-Inertial monocular constraints is being discussed. The depth-drop out problem in the RGB-D sensors (Kinect/stereo) which has never been addressed elsewhere is used in this work to calibrate the low-cost IMU seamlessly at higher rates, which is crucial for the success of inertial based SLAM. In our work, we have focused on the challenging indoor/outdoor environments with a front-facing camera, [15] where depth variations vary frequently, hence making the overall problem more challenging. The Figure 4.1 shows the output of our developed system capable of working in indoor/outdoor environments using monocular and depth based cues.

[1] Google's RGB-D sensor (Tango) works in indoor and outdoor environments by using the Visual-Inertial odometry; however, the approach has not been made public knowledge yet [89].

[2] customised in-house developed IMU costing 40USD.

[3] During static or near-static states of the inertial platform, the absolute roll and pitch angles can be estimated using the direction of gravity[73].

## 4.2 Related Work

Since there is a large amount of literature on SLAM, this section is devoted to RGB-D/stereo SLAM and its inertial-integrated variants. For monocular approaches, please refer to the previous chapter for monocular inertial SLAM.

Previous work on RGB-D based SLAM has used full depth-colour for aerial vehicles [49] or depth-only information [90] for hand-held scenarios. The work of [49] uses RGB-D information by detecting features from the image and their corresponding depths are used to build 3D feature points. The feature points are then matched with the key-frame features to estimate the camera pose. The work is based on a multi-threaded approach and graph optimisation is performed for the global consistency of the map as an off-line process. They have demonstrated their approach in the indoor environment on an aerial vehicle. The work of [90] relies on depth-only information and hence used the Iterative Closest Point (ICP) approach to exploit the structural properties of the environment. Recent work [87], which emphasises real-time performance, considers the visual keypoints to initialise the ICP for the depth odometry pipeline and maintains a constant size feature map. Another study has been dedicated to using depth information in the monocular SLAM framework [91]. Depth information is used to solve the drawbacks of the monocular Parallel Tracking and Mapping (PTAM) algorithm; for instance, automatic bootstrapping and 3D feature initialisation. Recently there is another work [92] in which 3D visual odometry is integrated with the ICP-based SLAM approach. The underlying methodology for the existing stereo SLAM [47, 88, 93] also relies on the fusion of colour information and geometry information (similar to the RGB-D approaches).

The aforementioned state-of-the-art techniques in RGB-D/stereo rely heavily on the availability of the depth data [94] and hence are deemed to fail in depth dropout cases. The depth dropout issue was recently considered by [95], in which they treated this problem as an off-line optimisation problem in an indoor environment. The approach is based upon heuristically combining monocular constraints and RGB-D constraints with local mapping problems in an offline way. In the offline setting the scale of

the monocular camera can be recovered easily; however for online processing, the scale is not easily recoverable. Our work focuses on the online real-time probabilistic approach by utilising visual directional constraints in conjunction with the inertial sensor.

Considering the work in the Visual-Inertial domain, the two well known paradigms are loosely and tightly coupled architectures [4]. In a tightly coupled paradigm, the work of [76, 85, 96] fused visual and inertial information using a single EKF and maintained the cross-correlations between the poses and visual features in the filter. However, as mentioned before, the computational complexity of these approaches is high, hence a compromise has to be made on the total number of features (visual constraints) that can be maintained in the filter.

An alternative option is a loose coupling approach in which the visual and inertial information is treated as a separate entity and visual constraints are used to update/aid the inertial sensor [38, 53]. A stream of work in which the inertial (gyro based rotation) information is aided to the depth based pose estimator includes [97, 98]. In [97] gyro information is used as an initialisation for the pose estimator in ICP [99]. The work of [100] used an IMU and vision integration system where gyroscopes were used to estimate the rotation of the cameras, and the main target of the fusion was to interpret the visual measurements. In another series of work [4, 12, 44], the vision sensor was treated as a global positioning sensor for inertial aiding [5]. Their system was designed using an indirect Kalman filter that is based on the errors in the estimated measurements instead of the direct measurements from camera and IMU systems. The work estimates the scale of the monocular camera motion estimate in the filter with the assumption of a downward looking camera (the scale of the scene changes smoothly). Earlier, our work [52] and recently the work of [15] highlighted the issue of forward-looking camera where the assumption of smooth variation in the scale is not valid, hence making the filter diverge. Our work is based upon the loosely coupled

[4]The generic architecture details were discussed earlier in Chapter 2, however here we discuss these architectures with respect to the fusion of visual and inertial information.

[5]The work has used the state-of-the-art visual SLAM framework Parallel Tracking and Mapping [42].

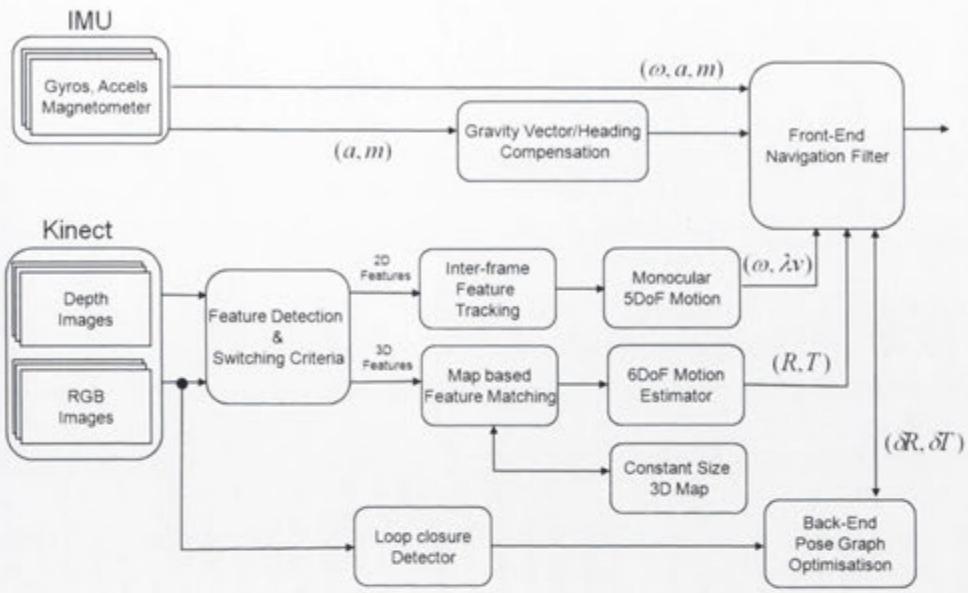



Figure 4.2: Flow chart of the Visual Inertial Framework.

approach with a direct-filter implementation [6]. Using visual directional constraints, we avoid the explicit estimation of scale to fuse monocular constraints to the inertial filter (details discussed in the section 4.5).

## 4.3 Visual-Inertial SLAM Framework with RGB-D sensors

We followed a loosely coupled direct architecture (features are not maintained in the SLAM filter), as in the EKF-tightly coupled approach [7] has a bottleneck due to the scalability issue [17]. SLAM systems generally consist of front-end and back-end subsystems. The front-end provides the initial vehicle state (filtering) and loop closure constraints [53] whereas the back-end turns the initial solution into a maximum

[6]The advantage of the loosely coupled approach is a constant time processing, which is suitable for real-time application

[7]Presented earlier in Chapter 3.

likelihood solution considering all the available data [101].

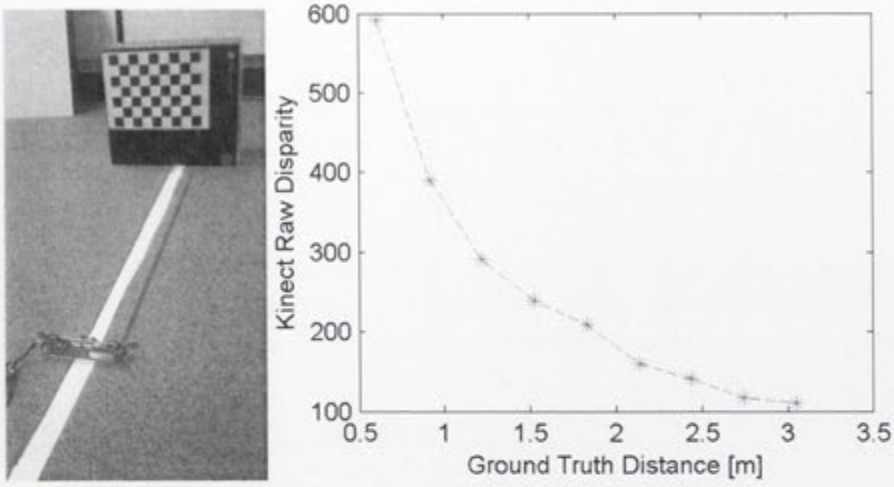
Figure 4.2 illustrates the overall system architecture where the vision node provides either RGB-D or monocular information, depending on the availability of the depth information. In our proposed framework, the front-end of the system consists of the probabilistic state estimator and Kinect based pose estimator (RGB-D and Monocular). The back-end of the system is based on the graph optimisation, which tries to compensate for the effect of loose coupling between the features and state estimator. The framework of loosely coupled architecture has the advantage of treating the Kinect based pose estimator as a black box and a benefit of constant computational costs for pose estimation. Indoor and outdoor flight results demonstrate the robustness of the proposed approach in challenging environments. The proposed framework and experimental results make the RGB-D sensor a viable solution for indoor and outdoor navigation. To the best of our knowledge, this has never been demonstrated before in outdoor flying settings. A video of the proposed framework for aerial dataset is available online [8].

## 4.4 Processing RGB-D Measurement

In this section, we will discuss the 6DOF pose constraints estimated from the RGB-D sensor (when both the depth and colour information is available). In SLAM theory the map of the environment is generally represented by a collection of landmarks (for 6DOF constraints we maintained a constant size map in order to obtain real-time performance). In this thesis, a features is a two-dimensional location in the Kinect frame, while the corresponding three-dimensional position will be referred to as a landmark (3D feature). The distinctive features help to identify unique landmarks in the environment and maintaining the set of these 3D landmarks help to identify motion/pose information (more details in Section 4.4.2).

To obtain the accurate pose information from the Kinect sensor, it is vital to correct the systematic errors present in the sensor (accounting for the calibration and depth to distance relation). In order to analyse the linearity of the depth sensor, we pointed the sensor to the planar object (checker board) at known distance. We estimated the depth provided

[8] `https://youtu.be/0h6vniZxwjk`



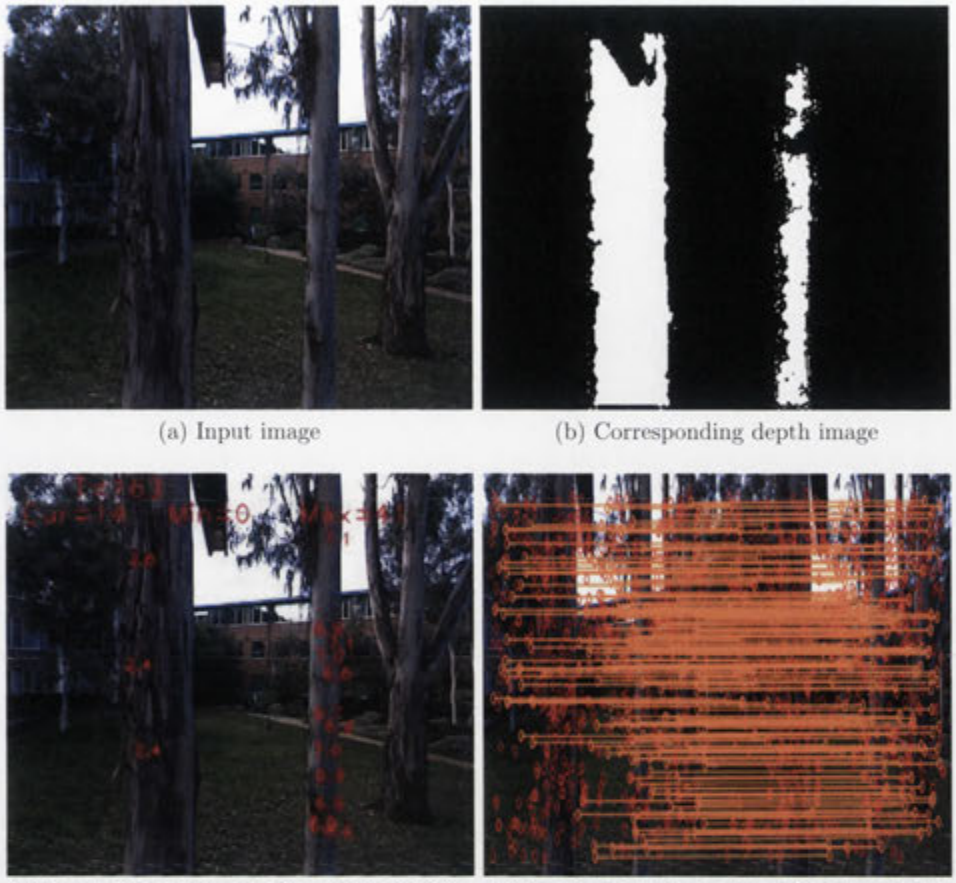

(a) (b)

Figure 4.3: Kinect measured depth compared to ground truth distance.

from the Kinect sensor for a region of interest (where the object is present) and average it. The raw output from the Kinect is an image that corresponds to the depth of the scene. Rather than providing the actual depth of the scene, Kinect returns the inverse depth. The Figure 4.3 shows the estimated disparity to actual known distance (indirectly proportional) relation for multiple observations at different distances. The relationship is calibrated along with camera calibration using the approach of [102] to obtain optimal disparity estimation results.

### 4.4.1 3D Feature Detection and Matching

In SLAM system the features are used to model the geometric information and are used to build the map of the environment. The two important aspects regarding the features are: detection and description. The detection phase identifies the area of interest (typically a region with strong variation in intensity or corner [103]) and its characteristics are modelled using descriptor (typically considering scale space and orientation information of the neighbouring pixels [104]). The use of features in our work is motivated by the fact that the complete processing of the image is computationally expensive whereas the feature presents a represent the small set of distinguishable information.



(a) Input image

(b) Corresponding depth image

(c) Image with corresponding depth (3D features) mapped

(d) Feature matching between time k and k+1 for all 2D features

Figure 4.4: Feature Detection and Matching Process.

To estimate the ego-motion between the two Kinect frames, the features need to be detected and tracked. In order to detect the distinguishable features, corners [103] are easier to detect and to manipulate than lines/edges [104], hence simplifying the implementation/computation. In this work, the Harris corners [103] are initially detected on the gray scale image from the Kinect frame. The corners for which the corresponding depth is not available are discarded. The spatial location of the feature in the pixel coordinates with depth gives $(u, v, d) \in \mathbb{R}^3$, which can be converted into 3D Euclidean feature position, $(X, Y, Z) \in \mathbb{R}^3$ (in camera coordinate frame). Figure 4.4 shows the feature detection and matching process on the acquired aerial imagery using a hexacopter. The mapping function

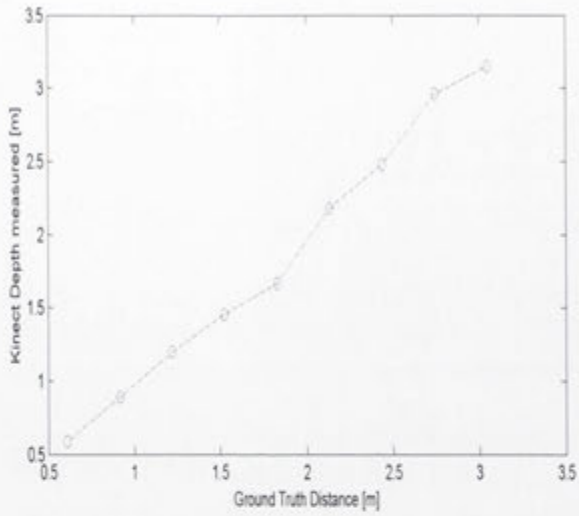



Figure 4.5: Kinect measured depth compared to ground truth distance.

$g : (u, v, d) \rightarrow (X, Y, Z)$ becomes where raw depth $\breve{\upsilon}$ is converted into 3D feature points as :

$$Z = \frac{z_{ref}}{1 + (\frac{z_{ref}}{fb})\breve{\upsilon}} \tag{4.1}$$

$$X = \frac{Z(u - u_0 + \delta u)}{f} \tag{4.2}$$

$$Y = \frac{Z(v - v_0 + \delta v)}{f} \tag{4.3}$$

Where $z_{ref}$ is the distance of reference plane to the sensor, f is the focal length, $(u_0, v_0)$ are the principal point, b is the baseline in meters and $(\delta u, \delta v)$ are the lens distortion terms. Applying the Equation 4.3, the relation between Kinect observed depth and ground truth calculated depth is shown in Figure 4.5.

The related covariance matrix $W$ of the transformed Euclidean 3D position can be computed by using Jacobian of the mapping and assuming independent noises in pixel

$(\sigma_u, \sigma_v)$ and depth measurements $(\sigma_d)$ as:

$$W = J \begin{pmatrix} \sigma_u^2 & 0 & 0 \\ 0 & \sigma_v^2 & 0 \\ 0 & 0 & \sigma_d^2 \end{pmatrix} J^T \qquad (4.4)$$
$$\text{where, } J = \frac{\partial g(X, Y, Z)}{\partial (u, v, d)}$$

After feature detection phase, the selection of feature descriptor approach plays an important role to achieve robust solutions. Among the well known local descriptors such as SIFT (Scalar Invariant Feature Transform) , SURF (Speeded Up Robust Feature) [104], NARF (Normal Aligned Radial Feature) [105],BRIEF (Binary Robust Independent Elementary Feature) [106], FAST/Harris with patches as descriptors [107, 108]. These descriptors exploit the characteristic of image by using intensity, gradient, curvatures and colour information for robust matching.

The local features are evaluated earlier by [109] in which they found that the SIFT features are highly discriminant but relatively slow to compute and match. This really affects the performance of real-time SLAM approach as various features have to maintained in the map. BRIEF features are faster to compute but lack the discriminator power required for accurate feature matching, however the SURF features provide faster computation while preserving the discriminative power of SIFT. The prefix (Speeded-up) is used to denote the computational efficiency of SURF against the SIFT features (by using fast approximation of SIFT algorithm [106])

In our work, SURF descriptors [104] are estimated from the detected features for matching purpose. The SURF descriptor finds a circular region against the detection feature locations. SURF also relies on local gradient histogram like SIFT whereas the computation speed-up is achieved by using integral images [110].

The covariance of each 3D feature is estimated using [111] approach where $\rho$ is treated

as random variable with standard deviation as $\sigma\rho$ as:

$$\sigma_Z = \frac{1}{fb} Z^2 \sigma\rho \tag{4.5}$$

$$\sigma_X = \frac{u}{fb} Z^2 \sigma\rho \tag{4.6}$$

$$\sigma_Y = \frac{v}{fb} Z^2 \sigma\rho \tag{4.7}$$

With a $3 \times 3$ covariance matrix ($\Omega$) for a 3D feature point is estimated from them.

In order to reduce the short term drift, we maintained a map of 3D features with their covariance and descriptors in a ring buffer. The buffer has a constant memory size and thus enables the front-end to perform in constant time.

### 4.4.2 Pose Estimation using RANSAC and Fine Adjustment

After the extraction of feature descriptors, the next vital step is to match the features to obtain the 6DOF pose estimates. The general approach for descriptor matching is to assumed that the Euclidean distance in feature space can be used for scoring the pair-wise matches (i.e. to set a fixed threshold on the Euclidean distance for declaring a match). The first frame features are declared as part of the map ($\mathbf{M}$) defined in the local navigational frame. All the subsequent feature measurement data ($\mathbf{D}$) are matched with the existing map features using the SURF descriptors. The comparing score is based on the sum-of-absolute-difference and if the score is within a specified threshold then it is declared as a matched-pair (details can be found in the implementation Chapter 5).

To determine the 6DOF pose information, RANSAC [112] is used with Horns approach [113]. RANSAC is capable of smoothing the noisy data (outliers) to find the correct correspondences using the model fitting approach. The RANSAC approach estimates the model with the Horn approach [113] and then fits this model to data points that agree with it. This procedure of model fitting is repeated with randomly selected set of features (among all the features) until the maximum number of iterations or condition of termination has reached.

The fixed number of iterations will ensure to return the least bad solution among the considered set of features (selected randomly) but not necessarily the optimal. Many enhancements to RANSAC has been suggested in research community (such as PROSAC, MSAC and MLESAC [114]) but this fundamental problem still remains a challenge. To get more accurate solution, we refine the results of the RANSAC with ICP. The rigid body transformation $(R, T)$ of the vehicle is obtained by weighted ICP for fine refinement.

$$\arg\min_{R,T} \left( \frac{1}{n} \sum_{i \in A} W_i ||\mathbf{M}_i - (R\,\mathbf{D}_i + T)\,||^2 \right) \tag{4.8}$$

where $i$ is the index of inlier feature set $A$ and $W$ is the weighting matrix given in (4.4).

The rigid body transformation is used to transform the feature data with their associated covariance to update or add new features into the map. The points which are within a predefined Euclidean vicinity are declared as update-points, whereas others are declared as new-points. The existing points are updated using a weighted averaging method whilst the inverse covariances are added together. The map feature descriptors are updated with the new ones (a different update scheme can be employed and researched but that was not the focus of this work). The new-points are added to the map with their descriptors and covariances. If the limit of ring buffer is reached then the old features are deleted [9].

## 4.5 Front-end Processing: EKF-SLAM Prediction

The estimation process of SLAM is based upon the inertial dynamic model and the relative observations of vision sensor to the environmental features. In this work, EKF is used as the state estimator whereas Inertial Measurement Unit (IMU) acts as a driver for the dynamic motion model. An IMU is comprised of tri-axial gyros and ac-

[9]Retaining the most spatially informative features in the map can further enhance the performance which is currently left for future investigation.

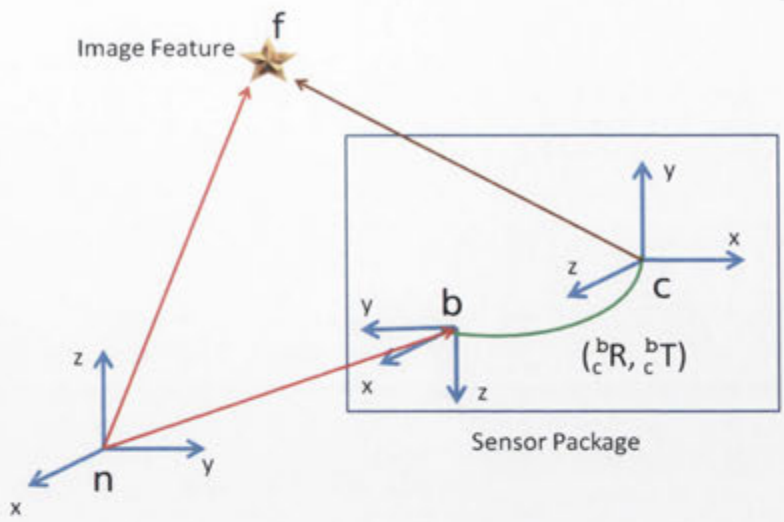



Figure 4.6: Frame of reference for camera/IMU system in navigational coordinate

celerometers arranged orthogonally and provides 6DOF measurements of the attached platform. The angular rate from the IMU's gyros is measured in the body frame and subsequently integrated to form the attitude (or orientation) of the platform. The calculated attitude is then used to transform the accelerometer measurements into the navigation frame quantity. The procedure used to estimate the position and velocity of the platform in the navigation frame is called the INS algorithm [83].

The process model includes the vehicle dynamic model and can be written in a discrete time as a first-order differential equation,

$$\mathbf{x}(k) \;=\; \mathbf{f}(\mathbf{x}(k-1), \mathbf{u}(k), \mathbf{w}(k), k) \tag{4.9}$$

where $\mathbf{f}(\cdot, \cdot, k)$ is a non-linear state transition function (at time $k$) which forms the current vehicle state, $\mathbf{x}(k)$, from the previous state, $\mathbf{x}(k-1)$, and the current control input, $\mathbf{u}(k)$. $\mathbf{w}(k)$ is the process noise vector.

In this work, the navigation frame is considered to be fixed (defined as a right-handed coordinate frame based on Newtonian mechanics) whereas the IMU is considered to be the sensor/body frame. It is assumed in this work that the position and orientation

of the navigation with respect to the real world is known e.g. the initial pose of navigational frame coincides with the body frame. The relative position and rotational parameters between the camera-frame and the body frame is actually estimated (calibration details are provided in Chapter 6). Figure 4.6 shows the relation between the navigation, inertial and the camera reference frames.

In this work the orientation in the navigation frame is is mechanised with Euler angle parameterisation at discrete time k. The state vector yields a 15 element state $\widetilde{\mathbf{x}}$ as:

$$\widetilde{\mathbf{x}}_k = \left[ \mathbf{P}^n(k), \mathbf{V}^n(k), \Psi^n(k), b_a^b, b_\omega^b \right]^T \tag{4.10}$$

$$\begin{bmatrix} \mathbf{P}^n(k) \\ \mathbf{V}^n(k) \\ \Psi(k) \end{bmatrix} = \begin{bmatrix} \mathbf{P}^n(k-1) + \mathbf{V}^n(k-1)\triangle t \\ \mathbf{V}^n(k-1) + [\mathbf{C}_b^n(k-1)[\mathbf{f}^b(k) - b_a^b(k)] + \mathbf{g}^n]\triangle t \\ \Psi(k-1) + \mathbf{E}_b^n(k-1)[\boldsymbol{\omega}^b(k) - b_\omega^b(k)]\triangle t \end{bmatrix},$$

where $\Psi(k)$ represents the Euler angles: roll ($\phi$), pitch ($\theta$) and yaw ($\psi$). $\mathbf{f}^b(k)$ and $\boldsymbol{\omega}^b(k)$ are acceleration and rotation rates measured in the body frame. The control input $\mathbf{u}$ consists of linear acceleration $\overline{\mathbf{f}}^b$ and angular velocity $\triangle\mathbf{q}^n(k)$ and $\mathbf{g}$ is the gravity vector. Where $g = [0, 0, 9.8m/s^2]^T$ is the gravity vector, $(bias_a^b, bias_\omega^b)$ are accelerometer and gyro biases respectively and $[\omega^b]_\times$ is a skew-symmetric form of the angular velocity vector. The dynamics of the accelerometer and gyro biases are modelled as random walk process with white Gaussian noise $\eta$:

$$\dot{b_a^b} = \eta_a$$
$$\dot{b_\omega^b} = \eta_\omega.$$

The $\mathbf{C}_b^n$ is the direction cosine matrix and $\mathbf{E}_b^n$ is the matrix that transforms the rotation rates in the body frame to the Euler angle rates

$$\mathbf{C}_b^n = \begin{bmatrix} C_\theta C_\psi & -C_\phi S_\psi + S_\phi S_\theta C_\psi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ C_\theta S_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix}$$

$$\mathbf{E}_b^n = \begin{bmatrix} 1 & S_\phi S_\theta / C_\theta & C_\phi S_\theta / C_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi / C_\theta & C_\phi / C_\theta \end{bmatrix}$$

where $S_{(\cdot)}$ and $C_{(\cdot)}$ represents $sin(\cdot)$ and $cos(\cdot)$ respectively.

The Jacobian $\nabla \mathbf{f}_x(k)$ of the non-linear vehicle model with respect to the vehicle state in equation 4.11 can be computed as,

$$\nabla \mathbf{f}_x(k) = \begin{bmatrix} \frac{\partial \mathbf{P}(k)}{\partial(\mathbf{P}(k-1),\mathbf{V}(k-1),\Psi(k-1))} \\ \frac{\partial \mathbf{V}(k)}{\partial(\mathbf{P}(k-1),\mathbf{V}(k-1),\Psi(k-1),bias_a^b(k-1))} \\ \frac{\partial \Phi(k)}{\partial(\mathbf{P}(k-1),\mathbf{V}(k-1),\Psi(k-1))} \\ \frac{b_a^b(k)}{\partial(\mathbf{P}(k-1),\mathbf{V}(k-1),\Psi(k-1))} \\ \frac{b_a^b(k)}{\partial(\mathbf{P}(k-1),\mathbf{V}(k-1),\Psi(k-1))} \end{bmatrix} \tag{4.11}$$

The Jacobian of the important terms in Eq. 4.11 are as follows:

$$\frac{\partial \mathbf{P}(k)}{\partial(\widetilde{x}_{k-1})} = \begin{bmatrix} \mathbf{I} & \triangle t I & 0 \end{bmatrix} \tag{4.12}$$

$$\frac{\partial \mathbf{V}(k)}{\partial(\widetilde{x}_{k-1})} = \begin{bmatrix} 0 & \mathbf{I} & \frac{\partial(\mathbf{C}_b^n(k)(\mathbf{f}^b(k)-bias_a^b(k)))}{\partial \Psi(k-1)} \triangle t \end{bmatrix} \tag{4.13}$$

$$\frac{\partial \Phi(k)}{\partial(\widetilde{x}_{k-1})} = \left[ \begin{array}{cccc} 0 & 0 & \mathbf{I} + \frac{\partial(\mathbf{E}_b^n(k)(\boldsymbol{\omega}^b(k) - bias_\omega^b(k)))}{\partial \Psi(k-1)} \triangle t \end{array} \right] \tag{4.14}$$

where the sub-matrix can also be computed by using Jacobians of $\mathbf{C}_b^n$ and $\mathbf{E}_b^n$ with respect to the Euler angles $(\rho, \psi, \theta)$. They are three element matrices and are computed as,

$$\frac{\partial \mathbf{C}_b^n}{\partial(\rho, \psi, \theta)} \triangleq \left[ \frac{\partial \mathbf{C}_b^n}{\partial \phi}, \frac{\partial \mathbf{C}_b^n}{\partial \theta}, \frac{\partial \mathbf{C}_b^n}{\partial \psi} \right] \tag{4.15}$$

$$\frac{\partial \mathbf{E}_b^n}{\partial(\rho, \psi, \theta)} \triangleq \left[ \frac{\partial \mathbf{E}_b^n}{\partial \phi}, \frac{\partial \mathbf{E}_b^n}{\partial \theta}, \frac{\partial \mathbf{E}_b^n}{\partial \psi} \right] \tag{4.16}$$

where,

$$\frac{\partial \mathbf{C}_b^n}{\partial \phi} = \left[ \begin{array}{ccc} 0 & S_\phi S_\psi + C_\phi S_\theta C_\psi & C_\phi S_\psi - S_\phi S_\theta C_\psi \\ 0 & -S_\phi C_\psi + C_\phi S_\theta C_\psi & -C_\phi C_\psi - S_\phi S_\theta S_\psi \\ 0 & C_\phi C_\theta & -S_\phi C_\theta \end{array} \right]$$

$$\frac{\partial \mathbf{C}_b^n}{\partial \theta} = \left[ \begin{array}{ccc} -S_\theta C_\psi & S_\phi C_\theta C_\psi & C_\phi C_\theta C_\psi \\ -S_\theta S_\psi & S_\phi C_\theta C_\psi & C_\phi C_\theta S_\psi \\ -C_\theta & -S_\phi S_\theta & -C_\phi S_\theta \end{array} \right]$$

$$\frac{\partial \mathbf{C}_b^n}{\partial \psi} = \left[ \begin{array}{ccc} -C_\theta S_\psi & -C_\phi C_\psi - S_\phi S_\theta S_\psi & S_\phi C_\psi - C_\phi S_\theta S_\psi \\ C_\theta C_\psi & -C_\phi S_\psi - S_\phi S_\theta S_\psi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ 0 & 0 & 0 \end{array} \right]$$

$$\frac{\partial \mathbf{E}_b^n}{\partial \phi} = \begin{bmatrix} 0 & C_\phi S_\theta / C_\theta & -S_\phi S_\theta / C_\theta \\ 0 & -S_\phi & -C_\phi \\ 0 & C_\phi / C_\theta & -S_\phi / C_\theta \end{bmatrix}$$

$$\frac{\partial \mathbf{E}_b^n}{\partial \theta} = \begin{bmatrix} 0 & S_\phi / C_\theta^2 & C_\phi / C_\theta^2 \\ 0 & 0 & 0 \\ 0 & S_\phi S_\theta / C_\theta^2 & -C_\phi S_\theta / C_\theta^2 \end{bmatrix}$$

$$\frac{\partial \mathbf{E}_b^n}{\partial \psi} = \mathbf{0}$$

The Jacobian $\nabla \mathbf{f}_w(k)$ of the non-linear vehicle model with respect to the input noise (or inertial sensor noise) in equation 4.11 is obtained by,

$$\nabla \mathbf{f}_w(k) = \begin{bmatrix} \frac{\partial \mathbf{P}(k)}{\partial (\mathbf{f}^b(k), \boldsymbol{\omega}^b(k))} \\ \frac{\partial \mathbf{V}(k)}{\partial (\mathbf{f}^b(k), \boldsymbol{\omega}^b(k))} \\ \frac{\partial \Psi(k)}{\partial (\mathbf{f}^b(k), \boldsymbol{\omega}^b(k))} \\ \frac{\partial bias_a^b(k)}{\partial (\mathbf{f}^b(k), \boldsymbol{\omega}^b(k))} \\ \frac{\partial bias_\omega^b(k)}{\partial (\mathbf{f}^b(k), \boldsymbol{\omega}^b(k))} \end{bmatrix} \tag{4.17}$$

## 4.6 Front-end Processing: EKF-SLAM Update

Given that the Kinect dropout can happen, we allow the EKF filter to update dynamically by switching between the monocular or RGB-D measurements. The monocular mode pipeline is similar to RGB-D pipeline by feature detection and matching phase except that the depth information is not available and visual directional constraints are estimated as explained in the chapter 3. The switching criteria is based upon the presence of the depth features and their spatial geometry. If the number of depth features are greater than a pre-specified threshold and uniformly distributed over the image, then RGB-D measurements are used to update the filter. Otherwise monocular measurements are utilised. The measurement uncertainty related to the visual pose output is dynamically scaled with the number of inliers to gauge the quality of the RGB-D/monocular motion estimation. The measurement delay involved in the

vision processing should also be handled carefully within the integration filter. The measurement update has to be synchronised in time with the inertial based prediction step.

Therefore we maintain time stamps with the predicted states within the ring buffer. When the measurement is obtained, the past EKF state with matching time is retrieved and updated accordingly. The corrected state is then propagated to the current state (by utilising the past states store in the ring buffer for EKF update).

Given that the Kinect drop-out can happen, we allow our EKF update step to dynamically switch between monocular or RGB-D constraints. The integration switching criteria is based upon presence of depth features and geometry of the features. If the depth features are greater than pre-specified threshold and uniformly distributed over the image (geometric property) then RGB-D innovation constraints are used for EKF-update step (otherwise monocular constraints are utilised (Chapter 3, Section 3.2)).

The measurement covariance matrix (R) for the RGB-D and monocular odometry constraints are estimated from the motion estimation data of ground truth data against the proposed approach. The measurement covariance is dynamically scaled with number of inliers (to gauge the quality of the RGB-D/monocular motion estimation) before doing EKF update.

One aspects that is very important during EKF update step is; handling the measurement delay. The occurrence of the delay is usually due to the computational cost of vision processing. The measurement update has to be synchronised in time with the EKF-prediction step. We maintain a ring buffer of past states of EKF prediction with time information (a global time-stamp of IMU is used during inertial-vision data acquisition). When the measurement is obtained, the exact time of the past EKF state is retrieved and updated. The corrected state in then propagated to the recent state (current time) to propagate the error forward.

The EKF is implemented for the estimation of both the vehicle and map states. The state and its covariance are predicted using the control input which typically runs at

high-frequency to track the manoeuvring aircraft. Whenever a landmark is observed a data association process is conducted which checks to see if the landmark has been previously observed. If the landmark has been previously registered in the filter the observation is used to update the state and covariance, and if the landmark is a new one then a new landmark state is augmented to the filter state.

The state covariance is propagated using the Jacobian of the state transition model and process noise matrix by,

$$\mathbf{P}(k|k-1) = \nabla\mathbf{f}_{\mathbf{x}}(k)\mathbf{P}(k-1|k-1)\nabla\mathbf{f}_{\mathbf{x}}{}^{T} + \nabla\mathbf{f}_{\mathbf{w}}(k)\mathbf{Q}(k)\nabla\mathbf{f}_{\mathbf{w}}{}^{T}(k),$$

function with respect to the state and sensor noise respectively.

When an observation occurs, the state vector and its covariance are updated according to

$$\begin{aligned}
\widetilde{\mathbf{x}}(k|k) &= \widetilde{\mathbf{x}}(k|k-1) + \mathbf{W}(k)\nu(k) \\
\mathbf{P}(k|k) &= [\mathbf{I} - \mathbf{W}(k)\nabla\mathbf{h}_{\mathbf{x}}(k)]\mathbf{P}(k|k-1) \\
&\quad \times [\mathbf{I} - \mathbf{W}(k)\nabla\mathbf{h}_{\mathbf{x}}(k)]^{T} + \mathbf{W}(k)\mathbf{R}(k)\mathbf{W}^{T}(k),
\end{aligned}$$

where the innovation vector, Kalman gain, and innovation covariance are computed as,

$$\begin{aligned}
\nu(k) &= \mathbf{z}(k) - \mathbf{h}(\widetilde{\mathbf{x}}(k|k-1)) \\
\mathbf{W}(k) &= \mathbf{P}(k|k-1)\nabla\mathbf{h}_{\mathbf{x}}{}^{T}(k)\mathbf{S}^{-1}(k) \\
\mathbf{S}(k) &= \nabla\mathbf{h}_{\mathbf{x}}(k)\mathbf{P}(k|k-1)\nabla\mathbf{h}_{\mathbf{x}}{}^{T}(k) + \mathbf{R}.
\end{aligned}$$

$\nabla\mathbf{h}_{\mathbf{x}}(k)$ is the Jacobian of the non-linear state transition function $\mathbf{h}(\cdot)$ with respect to the predicted state $\widetilde{\mathbf{x}}(k|k-1)$.

## 4.7 Back-end Processing: Graph Optimisation

Loop closure within the vision node provides a global consistency of the system. A common approach is to represent the pose as a node and their relations as edges in a graph. This section provides the details of graph optimisation utilised in this work for completeness of the system however a comprehensive analysis on the back-end optimisation is presented in Chapter 6.

In this work, we have adopted the keyframe-based graph optimisation method as in [101]. The pose estimated from the EKF is fed to the back-end graph optimiser. Keyframe selection mechanism is carried out by applying a threshold to the accumulated relative ego-motion. The loop closure (or place recognition) approach has been extensively researcher, various authors used visual approaches [78, 80, 115], however other rely on the geometry of the environment [116, 117]. Loop closure is detected using the matching of the SURF feature descriptors (using the adopted approach [115]) between the current image and the existing keyframes. Once the loop is detected a new constraint (rigid transformation) is added to the graph and subsequently optimised till convergence. On convergence the EKF state (against the measured time-stamp) is updated with the help of the state ring buffer.

The dimension of the SLAM problem is very high when it is formulated as a non-linear least square [118] because all vehicle poses and feature locations are considered as parameters to be determined. The non-linear least square formulation is to minimise an objective function as follows:

$$F(\mathbf{x}) = \arg\min_{\mathbf{x}} \left( \sum (||E_o||_U^2 + ||E_m||_V^2) \right) \tag{4.18}$$

where the state vector is $\mathbf{x}$, $E_o$ is the error term for IMU based process model with $E_m$ error term for measurement model and $U, V$ are corresponding covariance matrices considering the models have Gaussian noise. The error term for the process model is expanded as:

$$E_o = \mathbf{x}(k) - \mathbf{f}\left[\mathbf{x}(k-1), \mathbf{u}(k), \mathbf{w}(k), k\right] \tag{4.19}$$

and the error term for observation model is defined as

$$E_m = \mathbf{z}(k) - \mathbf{h}(\mathbf{x}(k), \mathbf{M}) \tag{4.20}$$

whereas the $M$ is the map of the features maintained in the front-end (4.4.2).

The Mahalanobis distance for both relative errors in the cost function with zero-mean Gaussian noise with covariance $U, V$ can be written as

$$\begin{aligned} F(\mathbf{x}) \;=\; & \sum \left[\mathbf{x}(k) - \mathbf{f}(\mathbf{x}(k-1), \mathbf{u}(k), \mathbf{w}(k)))^T U^{-1} (\mathbf{x}(k) - \mathbf{f}(\mathbf{x}(k-1), \mathbf{u}(k), \mathbf{w}(k))\right] \\ & + \sum \left[\mathbf{z}(k) - \mathbf{h}(\mathbf{x}(k), \mathbf{M}))^T V^{-1} (\mathbf{z}(k) - \mathbf{h}(\mathbf{x}(k), \mathbf{M})\right]. \end{aligned} \tag{4.21}$$

The measurement and process models are both non-linear functions, and thus the non-linear objective function is linearised multiple times to reach local minima. Generally local approaches solve the objective function $F(x)$ as

$$F(\mathbf{x} + \delta\mathbf{x}) = F(\mathbf{x}) + J\delta\mathbf{x}, \tag{4.22}$$

where $J = \partial F(\mathbf{x})/\partial(\mathbf{x})$. Let $\varpi = (F(\mathbf{x} + \delta\mathbf{x}) - F(\mathbf{x}))$, then it becomes

$$J\delta\mathbf{x} = \varpi, \tag{4.23}$$

The solution can be found using pseudo-inverse of $J$

$$\begin{aligned} J^T J\delta\mathbf{x} &= J^T \varpi \\ \delta\mathbf{x} &= (J^T J)^{-1} J^T \varpi \end{aligned} \tag{4.24}$$

By including the covariance estimates $(U, V)$ as $\xi$, Equation 10 becomes

$$\delta \mathbf{x} = (J^T \xi^{-1} J)^{-1} (J^T \xi^{-1}) \varpi. \tag{4.25}$$

Where the Equation 4.25 is solved only ones by an optimisation based approach [62], it yields a similar result like the EKF or Extended information filter [119]. The advantage in optimisation based approach comes with repeatability of solution for Equation 4.25 with different linearisation points.

The error is fed-back to the EKF filter to improve the front-end results [10]. As the front-end is based upon the EKF filter so linearisation errors build up however the graph optimisation helps to minimise the effect of linearisation error using repeatability of solution in Equation 4.25. On the converge of the graph the results are used as an update to the EKF filter. One important thing to note is, if the update to the EKF filter is to the earlier state, then the error is propagated forward to the latest state using the ring-buffer approach (as mentioned earlier in section 4.6).

## 4.8 Summary and Discussion

This chapter addressed the theoretical and experimental development of monocular visual-inertial constraints for an RGB-D sensor to handle the depth dropout problem. We proposed an on-board approach based upon EKF, which fuses the full 6DOF pose or partial 5DOF from the Kinect sensor with dynamic bias estimation. Indoor and outdoor results were demonstrated for an autonomous aerial vehicle to evaluate robustness against the depth dropout problem. We followed a loosely coupled direct architecture, as in the EKF-tightly coupled approaches [17] (their scalability is a bottleneck). The front-end of the system is based on the decoupling of the two core modules: the state estimator and Kinect based pose estimator (RGB-D and monocular). The back-end of the system is based on the graph optimisation, which

[10] The idea is adaptation from the work of [120], however they have used tightly coupled EKF filter with the feature/pose based optimisation.

tries to compensate for the effect of covariance decoupling (of the features with the state vector) in the front end.

# Chapter 5

# Real-time Implementation

## 5.1 Introduction

In this chapter, we will describe the hardware development of the aerial vehicle and implementation of the algorithms that are needed for localisation, mapping and control. This chapter is the theoretical realisation of the concepts presented earlier in Chapters 3 & 4. The goal of this work was to develop a low-budget multicopter system capable of carrying a sensor payload (customized landing gear and powerful motors), which is truly self-contained for the autonomous navigation and flight control.

The overall framework is shown in Figure 5.1, with four main components that include: a multicopter system, an autopilot system, a ground station and a payload system (SLAM and position controller). It also shows the communications architecture of the system between the different components using the IEEE 802.11 and 915MHz radio channel. The payload system consists of a Pico-ITX computer, inertial sensor and Microsoft Kinect.

The brain of the whole system is located in the payload module which contains a real-time implementation of the localisation, mapping and position controller on the Pico-ITX board. Furthermore, the autopilot system provides the stabilisation and various navigational functionalities to the aerial platform. There are two sources of

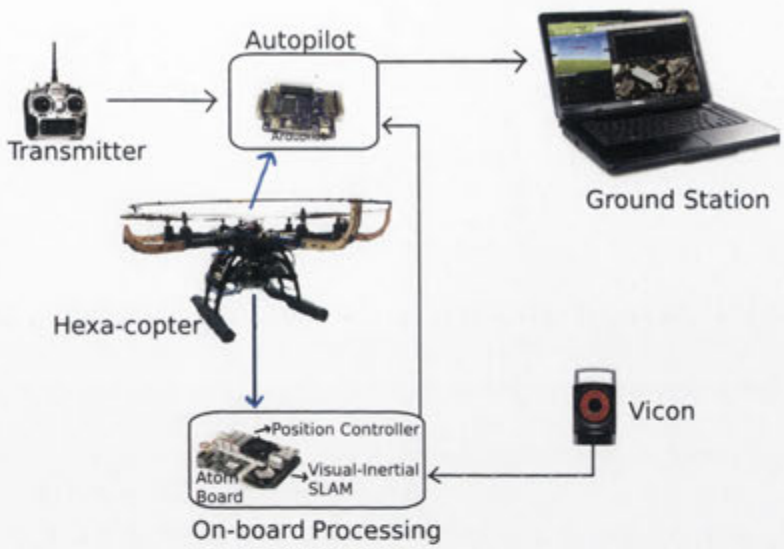



Figure 5.1: The conceptual framework for the multicopter system.

position data, only one of which is used by the position controller at any one time. The VICON system can only be used in an appropriately configured indoor environment (ground truth), while the Kinect-SLAM algorithm can provide position data in any environment that is appropriately feature-dense (indoor & outdoor). The autopilot can also receive a signal from the R/C transmitter (2.4GHz) which is used to override manual flying commands. There are three lithium batteries used to power the Kinect, multicopter platform and Pico-ITX computer. The telemetric ground station is used to monitor the flight data using a personal computer over a 915MHz radio channel.

Firstly, the hardware development of the aerial vehicle, (in our case) the hexacopter platform, will be discussed. The development target is a low-cost platform with commodity-level components easily available in the market. The hardware assembly involves the machining, soldering, design and construction of new components for the hexacopter system.

Secondly, the hardware development of the inertial unit will be discussed with the designing and manufacturing process. The autopilot system that provides the navigation and control capability to the aerial vehicle will also be presented. The in-house

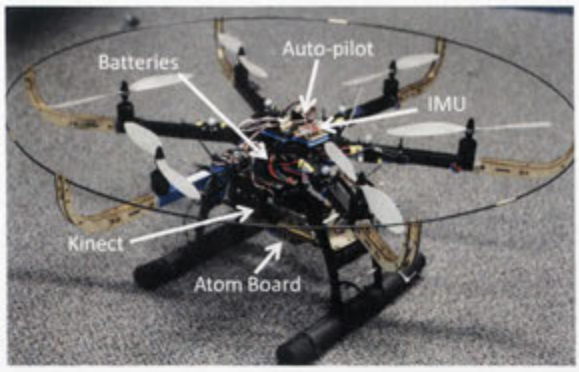



Figure 5.2: Hexacopter platform with its hardware components.

developed platform with its important hardware components is shown in Figure 5.2.

Thirdly, the real-time implementation of the on-board algorithms of vision based pose estimation (Chapter 3) in conjunction with the inertial based probabilistic framework (Chapter 4) will be described. The real-time implementation of the position controller will also be presented, which uses the steering information from the SLAM algorithm to enable autonomous hovering. The algorithmic libraries and parametric setting will also be provided.

Finally, a cost and performance comparison is provided for the custom built aerial platform and its payload system [121]. The total cost of the custom built hexacopter and payload system is under $1300, fulfilling the goal of a low-budget on-board real-time autonomous vehicle.

## 5.2 Developed Multicopter Platform

A custom build low cost hexacopter is the flying platform used in this thesis. The rotary-wing aerial vehicle with six (or more) rotors is gaining popularity due to its manoeuvrability and payload carrying capabilities. As compared to quadcopters (which are currently the most common form of aerial platform), the hexacopter provides the extra motors as redundancy and higher power/lift [123]. The technology of multicopters is made possible due to the enhanced performance of Electronic Speed Con-


Figure 5.3: The hexacopter platform with its major components.

trollers (ESC), brush-less motors and the increased capacity of lithium batteries. In addition to this, the different material used for the frame also improves the performance and efficiency of these flying vehicles.

The 3D model of the developed hexacopter with its axes convention is shown in Figure 5.3. The mode of the hexacopters are defined by the orientation of the rotors and arms relative to the x,y axes in the body fixed frame [123, 124]. In this thesis, we have used the + mode where the positive x-axis of the body fixed frame is aligned with one of the arms (blue colour). The positive y-axis is defined by the standard right-hand rule while positive z-axis is considered to be in the skyward direction.

Hexacopter flies in the 3D environment (6DOF) but it needs 4 DOF for its control i.e. roll, pitch, yaw and thrust [124]. The 6 motors are fixed pitch and produce thrust and torque when powered. The lift of the multicopter is controlled by speeding and slowing of the multiple motors (to change its net force). Higher speeds mean more thrust, and vice versa. The net torque of the flying vehicle is made zero by using three counter rotating pairs of rotors as shown in Figure 5.4. This helps maintain the platform to maintains its yaw angle. The yaw of the platform can be adjusted by the speed of the clockwise and counter-clockwise motions of the motors (changing the

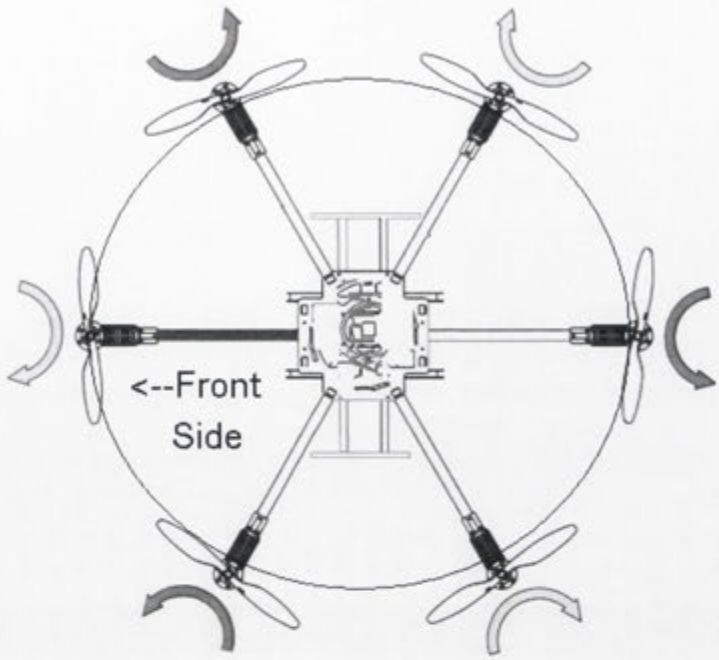



Figure 5.4: The motor angular configuration for the hexacopter platform.

angular velocity of the motors). In short, a net thrust helps to change the altitude while a net torque helps to rotate the aerial vehicle along z-axis. Meanwhile the roll is controlled by varying the speeds of the motors on either side of the y-axis. In addition, the Pitch is also controlled in the same manner, but by adjusting the speeds of the motors along the x-axis. An increase in the speed of the motors on the positive x-axis side, with a corresponding decrease in the speed of the motors on the opposite side, results in the hexacopter 'pitching' backwards.

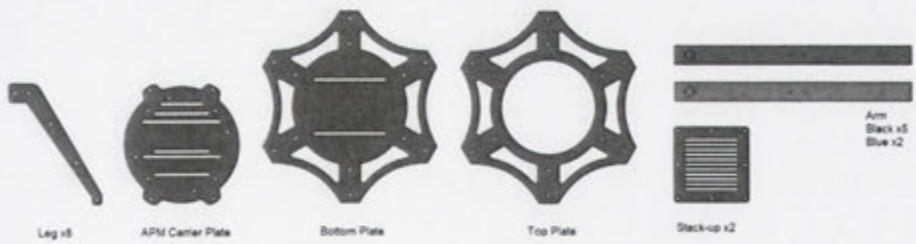



Figure 5.5: The parts for the hexacopter assembly [122].

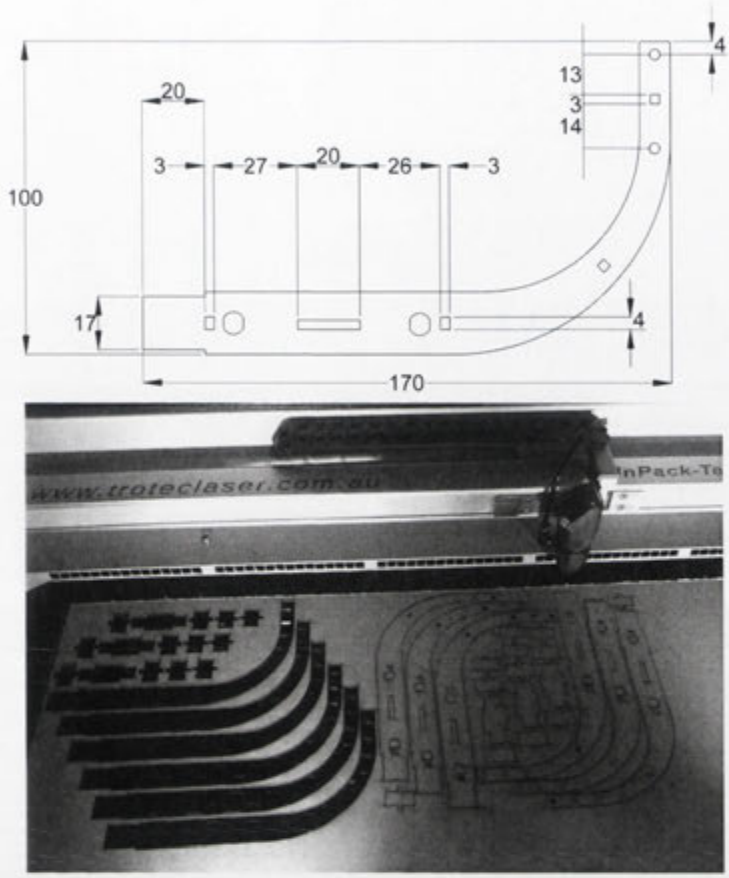



Figure 5.6: The design and laser printing of safety supports for hexacopter platform.

During the development phase, the assembly of the 6-armed Hexa-B kit (designed by 3D robotics [125]) was used. The frame was made of aluminium arms with different components as shown in Figure 5.5. Instead of 850Kv motors (which were default), we used more powerful 880Kv motors, with 11 inch propellers to provide increased lifting capability to the platform. The motors were soldered with the ESC, which were powered by a single 4-cell (14.8V) lithium-polymer battery with a capacity of 3300 mAh. The maximum flight time was limited by the battery capacity and was typically 10 minutes. It was realised during the testing that the force from the landing procedure had the potential to loosen the joints hence damaging the frame/payload system. The conventional landing gear was replaced with the 200mm landing gear

(a) Hexacopter with safety support


(b) Close-up view of saftey support

Figure 5.7: The safety ring for the hexacopter platform.

[126] for more durability and additional space to mount the payload. Moreover, the landing gear was encased with K-Flex elastomeric tubes to dampen the force of landing.

Early test flights also revealed that when landing, the hexacopter had a tendency to tip over and come to rest with part of its weight supported by its arms and propellers. As this resulted in the chipping or snapping of the propellers, it was necessary to design a safety ring encircling the propellers. This ring would also provide the additional benefit of protecting the propellers and objects in the environment should any mid-flight collisions occur. The ring of a 2D template was designed in CAD software to be

cut out of any 3mm thick material with the appropriate mechanical properties [127] as shown in Figure 5.6. The ring template was cut out from the thick sheets of medium density fibreboard (MDF) using a Trotec Speedy300 laser engraver (3mm thickness). The individual parts were glued together with E-Z Bond cyanoacrylate glue, and each assembly was inserted into the end of one of the hexacopter's arms, where it was secured with the application of hot glue to the outside of the joint. A length of carbon fibre rod was then passed through the circular holes in the MDF assemblies to complete the safety ring as shown in Figure 5.7. The weight of the assembled hexacopter without batteries is approximately 1.6kg. However with batteries, the final weight is approximately 2kg.

## 5.3 Developed Payload and Avionics System

The payload system consisted of a Microsoft Kinect RGB-D sensor, an in-house build IMU, a small Pico-ITX on-board computer, and two lithium-ion batteries (one to power the Kinect, and other to power the on-board computer). A Kinect unit was disassembled to make it lightweight by taking out the extra components i.e. pan motor, microphone and plastic covers. It was also rewired to accept the power from a lithium-ion battery, instead of its own AC wall socket lead. The on-board computer used for SLAM and position controller was the Pico-ITX (LP-170C) industrial single board computer. It was powered by a 1.66GHz Dual-core Intel Atom CPU with 2GB of RAM. The on-board computer required a 12V DC input with a peak current draw of up to 3A. The Atom board's power requirements were not compatible with the 4-cell 14.8V lithium-ion batteries available. To convert the power supply, a DE-SWADJ3 adjustable switching voltage regulator was used. The regulator has a very high efficiency of up to 96% and can provide $3-13$V output at up to 3A from a 5-35V input voltage. However, its maximum power output is 25W, which means that at the required 12V output, the supplied current is only 2.08A. While this does not meet the maximum requirements of the Pico-ITX (with reported tests from the manufacturer drawing up to 2.65W at 12V DC), in practice it was found that this was sufficient to

(a) Payload system on the hexacopter

(b) Close-up of the payload system

Figure 5.8: The payload system attached to the hexacopter platform (Kinect and its battery is placed above the base plate, while the on-board computer is attached below.)

run the Pico-ITX reliably.

The individual components were all mounted on a baseplate that was designed as a CAD template and cut out of 3mm MDF with the Trotec laser engraver. It was designed to allow the entire assembly to fit neatly into the space beneath the hexacopter and to be easily secured to the landing frame with zip-ties [127]. The payload was placed underneath the hexacopter and close to its centre of gravity (to avoid stability issues due to shift in centre of gravity).The Kinect was powered directly with a 3-cell lithium-ion battery with 1350mAh while a USB cable was connected to the Pico-ITX board for data transfer.

The assembled and mounted payload system is shown in Figure 5.8. As depicted, the Kinect and its battery are mounted above the baseplate, while the on-board computer

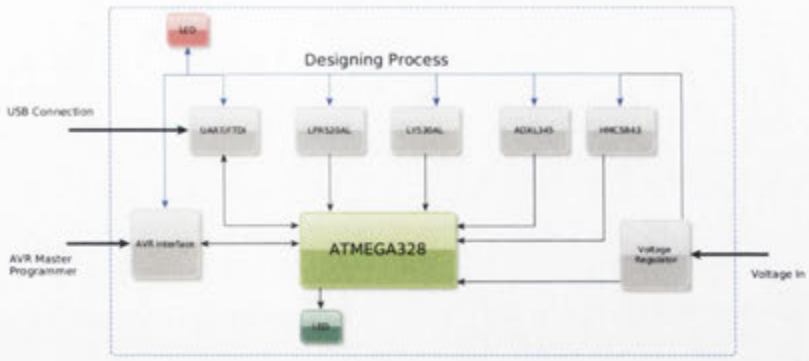



Figure 5.9: The conceptual layout of the in-house build IMU.

is attached below. The weight of the payload without batteries is approximately 0.7 kilograms, with the batteries bringing the total weight up to 1.2 kilograms. This makes the total flight ready weight of the combined hexacopter and payload system 3.2 kilograms.

### 5.3.1 In-house Built Inertial Measurement Unit

In this section, a low-cost customised inertial unit is presented with the details of its design and manufacturing process (our earlier work [68]). IMU plays a vital role in tracking the attitude of the flying vehicle due to its high data rates and is mostly used for attitude stabilisation. In our work, we have used the IMU pose information fused with the vision pose to autonomously control the aerial platform and perform SLAM. A prototype IMU board using Atmel micro-controller and MEMS sensors was designed and manufactured with a C++ embedded implementation (to process the raw inertial data and utilise customised attitude controller).

The developed IMU is a combination of gyroscope (LPR530AL and LY530AL), accelerometer (ADXL345), magnetometer (HMC5843), micro-controller (ATmega328), voltage regulator circuit, LEDs, FTDI basic, and an AVR SPI interface as shown in the Figure 5.9. The developed design allows the user to communicate with FTDI basic device for serial communication and ISP for programming/reprogramming of IMU firmware. The operating voltage in this system is 3.3 VDC. The maximum supply

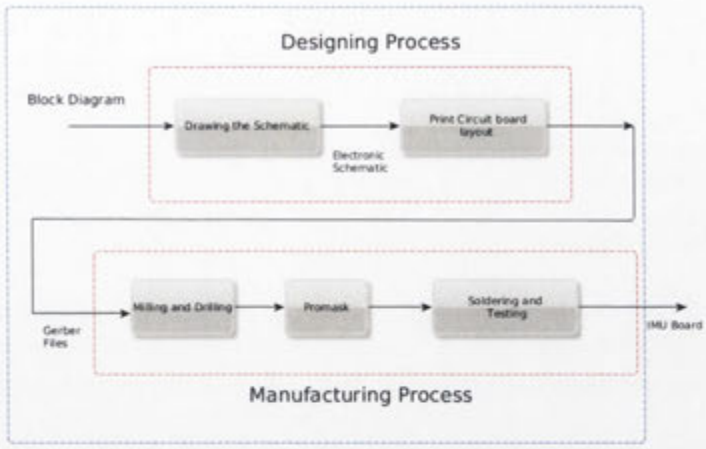



Figure 5.10: The design and manufacturing process involved for prototype build IMU.

voltage for this system is 16 VDC however, we have used the regulator to provide the required voltage.

ATmega328, an eight bit micro-controller was used in the developed IMU and a green LED was connected to indicate a serial communication. In the proposed design, the micro-controller got the clock source from 8MHz resonator. Five ADC pins were connected to gyroscope, two pins to the accelerometer and magnetometer (for I2C), three pins to the gyroscope, and two pins to the FTDI Basic interface and three pins to the AVR SPI interface.

In the gyroscope circuit, an external low pass filter was used in order to allow a band limiting output rate response. Resistor and capacitors were added to the gyroscope to implement a low pass filter. Digital output from the accelerometer was formatted as 16 bit 2's complement and in this design, it was accessible through I2C digital interface. In the developed IMU system, the magnetometer was also connected to the same I2C bus. The electrical connection of the magnetometer was based on single supply reference design [68]. In this case, the magnetometer was I2C slave and the ATmega328 was I2C master. The control of HMC5843 is carried out via the I2C bus and it used a 7-bit serial address (hence supporting the standard and fast modes i.e. 100kHz and 400kHz respectively).


Figure 5.11: Inertial Measurement Unit developed for Visual-Inertial SLAM framework.

In addition, the FTDI basic interface was added in the system for Universal Asynchronous Receiver Transmitter (UART). It was used for communicating serial data (text, number, etc.) between the ATmega328 and a computer via FTDI basic breakout board. AVR SPI interface was added for In-System Programming, which allowed the programming/reprogramming of the ATmega328 controller. For ISP, a programmer socket was connected to the ATmega328 using six-wire interface. The Serial Peripheral Interface consisted of three wires which were Master In Slave Out (MISO), Master Out Slave In (MOSI), and Serial Clock (SCK). In our case, ISP was operated as the Master and the ATmega328 was operated as the Slave. The schematic diagram of the designed IMU is shown in Figure A.2 (Appendix 1).

In the designing process of the IMU, there were two main processes involved: drawing the IMU schematic circuit and a PCB layout. Figure 5.10 shows the flow of designing and manufacturing process. The designing process was performed using EAGLE (Easily Applicable Graphical Layout Editor) software. All the components were added to the top side of the PCB layout. The accelerometer was mounted in a location close to a hard mounting point of the PCB in order to avoid measurement errors (due to the vibrations). Dual-axes gyroscope were placed as close as possible to single-axes gyroscope to get good result for roll, pitch, and yaw. After all components were added in board editor, the next step was routing the PCB. Manual routing was opted instead of auto-routing in-order to take care of the critical signals

while designing. When the routing process was finished, the next step was printing the PCB. Gerber files (file format used by PCB industry to describe the images of a PCB for milling, drilling, solder mask, copper layers) was generated from EAGLE software for the manufacturing process. The milling/drilling/manufacturing process has been done by Erasmo (technician in Research School of Engineering, ANU). The manufactured IMU was double sided and its dimension was 70mm ×39.5mm ×1.5mm.

Finally the circuit board was tested for continuity measurement by using multimeter. To ensure that each components was connected properly to each other, the probes were placed across the two points of the tested circuit. The value of the multimeter indicate the resistance (in ohms) and if no connection was present (open circuit) then 1 was shown as output. After the successful connectivity test of all the major components of IMU, then the voltage measurements were tested. The voltage output (3.3 VDC) from the voltage regulator was measured by powering it with 12 VDC power supply and using voltage meter. The black probe of the meter was placed on Ground (GND) and red probe was placed on the voltage output. The value of 3.3 VDC on the voltage meter indicates the successfully working of the IMU. After completing the connectivity and voltage measurement checks, the IMU was programmed with the firmware developed earlier in our work [68]. The in-house developed IMU is shown in Figure 5.11.

### 5.3.2 Autopilot System

The autopilot system is arguably the important piece of hardware in the multicopter system. It is responsible for attitude estimation, processing sensor data, controlling the aerial platform, flying on pre-defined waypoints and reporting/handling communications with the ground station. The core function of the autopilot system is to control the aerial platform using: the internal estimated states for stabilisation, the user-programmed mission files and the pre-defined fail-safe procedure.

The avionics system was based upon the open-source autopilot named ArduPilot-Mega (APM) 2.5 [125]. The purpose of using an open-source autopilot is to improve

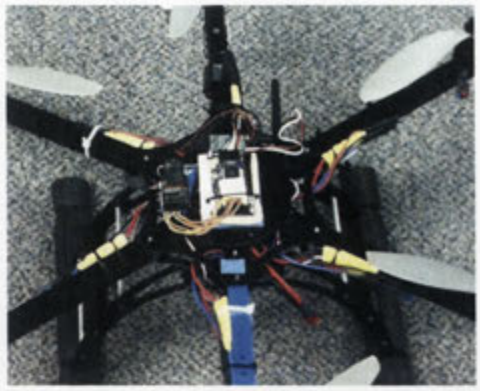


Figure 5.12: The autopilot was mounted onto the hexacopter platform and was placed on a the foam for vibration dampening.

the functionality and easily integrate with the customised payload system (for autonomous control purpose). The APM was designed for semi-autonomous control of both fixed-wing and rotary-wing powered aircraft. It acted as a bridge between the hexacopter sensors, motors,radio control receiver, ground based telemetry and the payload system. It offered both remote control and autonomous flying capabilities. For the manual flying, a 2.4GHz radio control receiver was attached to the autopilot system to follow the commands from the radio transmitter. The radio transmitter was operating in the Mode II (the left joystick controls the thrust and yaw, while the right joystick controls the roll and pitch) and was controlled by human pilot.

The heart of the APM is the low power Atmel 8-bit ATMega2560 micro-controller [122]. The micro-controller has 256KB ISP flash memory and achieves a throughput of 16 MIPS at 16 MHz. The sensor suit attached with the micro-controller contained: a 6-axis IMU(MPU-6000), GPS receiver (GTPA010), magnetometer (HMC5883l), barometric (MS5611-01BA) and sonar sensor (MB1240XL). The MPU-6000 provided the attitude estimates of the aerial platform, while the altitude information was provided by the MS5611 sensor (with a resolution of 10cm). A sonar sensor (resolution 1cm) for more accurate altitude measurements, as well as a GPS unit (4Hz) for position information outdoors were also attached. However, neither of these two sensors were used as the 3D vehicle pose information was provided from the SLAM/VICON system (will be discussed in section 5.4.2). Figure 5.12 shows the autopilot mounted

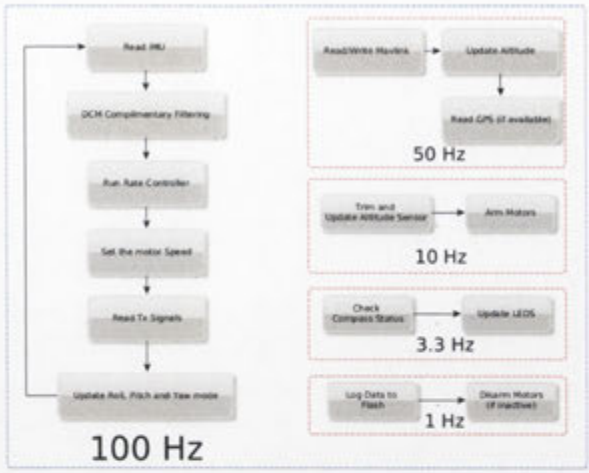



Figure 5.13: Algorithmic flow of Autopilot system.

onto the hexacopter platform and was placed on the foam for vibration dampening.

The autopilot was loaded with the modified version of the open-source Arducopter code ver. 2.9.1 [122]. The modification was done to communicate with the developed position controller and interface it with the autopilot attitude controller. The autopilot ran a series of loops, performing different tasks at specific frequencies. The main loop consisted of the core algorithms of the autopilot system, sensor data collection, our customised attitude estimation [68], and PI controller (discussed in section 5.4.2). The main loop was executed more frequently than any other algorithms of the autopilot code and executed at a rate of approximately 100 Hz as shown in the Figure 5.13.

Another important loop is the communication link between the autopilot and ground control station via Micro Air Vehicle Link (MAVLink). MAVLink is a protocol designed for communication, first released in 2009 [128]. Implemented as a header-only message library, it has only 8 bytes overhead per packet. Due to the low overhead, it can be used for both User Datagram Protocol (UDP) and Universal Asynchronous Receiver/Transmitter (UART) transport layers. The autopilot system communicates with the ground control station via an open-source 3DR radio modem (915MHz) at 50Hz. The MAVLink protocol [129] was used to send all the data to the ground


Figure 5.14: The ground control software [122] monitoring the sensor data while vehicle is flying near ANU building.

control station. The ground control station allowed the on-board sensors to be monitored and calibrated. The screen shoot of the ground control software [122] is shown in Figure 5.14.

During the flight test, it was also realised that the logging of the flight data plays important part to analyse the flight dynamics. The log files were recorded using two ways: using on-board 16Mb dataflash memory in a 1Hz loop and using the ground control software in a 50Hz loop. In addition to that, the real-time flight data was also used by the position controller running on Pico-ITX (details are provided in section 5.4.2).

## 5.4 On-board Real-time Implementation

The on-board implementation imposes a great challenge to handle the computation complexity and performed all the operation in real-time (for critical activities like autonomous control of dynamic flying vehicle). The airborne SLAM algorithm (theory described in Chapter 3 & 4) and position controller were implemented in real-time on

the developed payload of the developed hexacopter platform. The implementation was done in C++ using ROS framework [130] (which provides flexibility and modularity).

The algorithms that runs on ROS are normally composed of independent nodes (algorithms), each of which publish (send) or subscribe (receive) to information topics (data message structure) that are controlled by the operating system. Once a topic has been initialised, any node can publish values to it, or use values from it (via subscribing). This allows for the construction of modular systems where any node can be replaced, as long as the values required by other nodes are still being published by the replacement. In our real-time implementation, each program/node (data-acquisition node, vision node, EKF-prediction node, EKF-update node, Position-controller-node) was intended to be entirely self-contained and had the ability to pass the message/output to other node without knowing the implementation details of that node. For example, the acquisition of Kinect node passed the ROS messages to the vision processing node without knowing its implementation details (standard data structures passing procedure).

All computations were performed on a Pico-ITX board running Ubuntu 10.04. ROS latest version Fuerte was used as a framework for the on-board implementation. All the hardware components (VICON, Kinect, Inertial unit, Autopilot) communicated with each other via ROS pre-defined messages. ROS also provides access to standard packages across different hardware configurations i.e. the VICON_bridge driver package [131] was used to acquire the VICON data at 100Hz.

### 5.4.1 Real-time Implementation of Visual-Inertial SLAM

The IMU and vision data were processed in real-time in a fusion framework based on an Extended Kalman Filter. The 6DOF pose estimate of the vehicle and the map of the environment were estimated along side the accelerometer and gyroscope biases. The idea (in chapter 4) was to have a SLAM node which offered the loosely coupled integration approach, where the vision and inertial measurements were fused together in a probabilistic framework. The framework was divided into two steps, prediction

and update. The prediction was performed with the inertial sensor (process model) and the update was based on vision & Gravity/Magnetometer measurements (theory proposed in chapter 3 & 4) to reduce the overall error.

The Visual-Inertial fusion framework was build considering the modularity concept in mind as discussed earlier, where the EKF-prediction, EKF-update were also separated from other nodes. As the message passing was done using ROS message so each node did not have any information about the variables/implementation details of other node (modularity). The key advantage it offered was that different acquisition sensor and update sensor could be used while the main Visual-Inertial framework remain unchanged (durability). The libraries/ROS-packages used for this implementation are as follows:

- OpenNI [132]: to acquire the Kinect data.
- ROS Messages [133]: geometry_msgs (for vehicle poses), sensor_msgs (for image, camera calibration, point cloud, IMU), nav_msgs (for odometry), std_msgs (contain different data-types).
- OpenCV [134]: open source library to compute the feature detection/matching and image conversion to ROS messages vice-verse, image related visualisation.
- TF [133] : Keep track of multiple coordinate frames. It contains the relationship between different coordinate frame as a tree-structure.
- roscpp [133]: C++ implementation in ROS, the core library to use the ROS functionality in C++ code.
- rosbag [133]: set of tools for recording and playing back the ROS messages for data logging/playback purpose.
- Eigen3 [135]: for the matrices inversion and manipulation, mostly used in the EKF implementation and image storage types.
- Point Cloud Library [136], Storing 3D data and visualisation of the 3D point clouds.

- Boost [137]: support library, used to access the file-system and bind functionality, STL circular buffers (for ring-buffer implementation).

In the Visual-Inertial framework, firstly, the inertial acquisition node is presented which collected the raw data from the in-house build IMU. The inertial data was received at 50Hz on a serial port at 115200 baud rate. A wrapper code (written in C++) checked the checksum error for the validity of the data and if valid, then it was converted into the IMU ROS message (sensor_msgs). The time stamp form the Pico-ITX board was also embedded into the IMU message.

The EKF node was always listening for the new IMU message. As soon as new IMU message arrived, the EKF-prediction step took place. The state vector was defined using the geometry_msgs message containing the position, velocity and orientation in the form of quaternion (quaternion to Euler method was implemented as discussed in chapter 2, where-ever the Euler angles are required). The bias values for accelerometer ($bias_a^b$) and gyroscope ($bias_\omega^b$) in Equation 4.10 were defined as Vector3 [135]. The matrix manipulation for the prediction equations presented in Chapter 4 was implemented using Eigen [135] and the ring-buffer concept was implemented with the Boost library [137]. In the implementation, one of the key factor was the filter initialisation. The filter position and velocity were initialised to be zero (assuming the vehicle is stationary at the start) and attitude was initialised with the start-up calibrated attitude of the IMU (detailed provided in the calibration Section in Chapter 6).

The EKF update procedure was dependent upon the 6DOF or 5DOF constraints from the vision sensor and pose estimates from the gravity/magnetometer pose measurements. To estimate the visual measurements, the Kinect data stream was acquired using openni_driver [132, 133] at 22Hz data-rate. The data stream contained the registered depth and RGB image as well as camera calibration parameters (detailed provided in the calibration Section in Chapter 6). The feature extraction for Shi-Tomasi [103] was performed using GoodFeatureToTrack in Opencv library [134]. The feature matching was done using SURF descriptor [134] based on FLANN matcher [134].

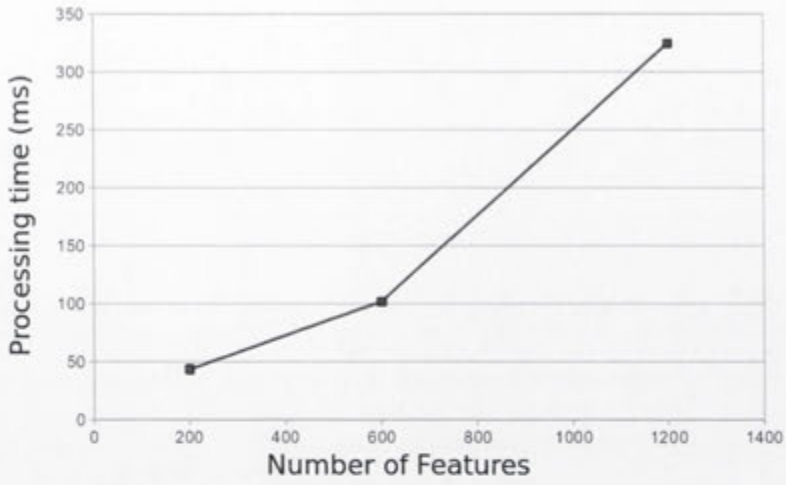



Figure 5.15: Computation requirements of feature and matching depending upon the number of input features in the image.

The number of features to be selected on the image plays a vital part in the computational complexity of the algorithm. It can be observed in the Figure 5.15 that when the number of features are increased then the computation time of the vision-node decreases. Considering the on-board real-time capability in mind the number of features were selected to be 200 per Kinect frame for matching. The input image resolution also effect the processing time as shown in Figure 5.16.

For the 6DOF constraints, the depth data and gray-scale image along with calibration data were converted into the 3D features (chapter 3 Equation 3.10). The relative transformation between the current Kinect frame features and map features (constant-size circular buffer) was estimated using Besl et al. approach [138] with RANSAC. The circular buffer [137] is used for the map features (the map update procedure is described earlier in Chapter 3). The benefit of using RANSAC for matching of the 3D features was to reduce the effect of false matches and to over-come the left-over imprecision, ICP was used. The relative transformation earlier was provided to the ICP for further refinement. For the ICP, the 3D features from the current Kinect frame and feature-map were matched using the KD-Tree nearest neighbour [134] approach. The matched points were declared as inliers, if the pair distance was less than

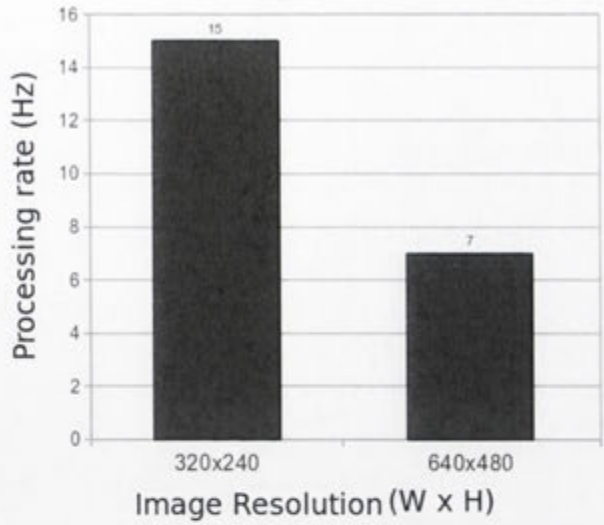



Figure 5.16: Computational comparison for different Kinect frame resolutions for the vision-node.

0.01m. The relative transformation using the weighted mean square error (details in chapter 3) was estimated. The algorithm was stopped, if the percentage of inliers was higher than 60% of the input features or the number of maximum ICP iterations reached. The relative transformation was transformed into 6DOF constraints as described earlier (in Chapter 4). Moreover, the 5DOF constraints had the same implementation of feature detection and matching as before. The major difference is that the formation of 3D feature step is not possible (unavailability of depth), so 2D-2D features were used to estimate the relative transformation. The relative transformation was estimated using the FundamentalMat [134] using RANSAC and were converted to 5DOF constraints (described in Chapter 3). The parameters selected for the experimentation are provided in Table 5.1.

The rigid transformation estimated from the vision node (6DOF and 5DOF constraints) were timestamped according to input Kinect frame. The status flag was also used to indicate the type of constraints to be used in the EKF-update step (to select the respective measurement function for $6DOF$ or $5DOF$, described earlier in Chapter 4). During fusion as there can be significant time delay between multiple

Table 5.1: Vision-node Selected Parameters

| | |
|---|---|
| Kinect Max Range | 5.5m |
| Number of Detected Features | 200 |
| Minimum Feature Matching | 50 |
| RANSAC Convergence Threshold | 85% |
| RANSAC Minimum Acceptable Percentage | 25% |
| Max. RANSAC iterations | 100 |
| Number of ICP correspondences | 400 |
| Max. ICP correspondence distance | 0.3m |
| Max. ICP iterations | 10 |
| Average ICP pair distance for convergence | 0.01m |
| KNearestNeighbour[134] | 4 |
| Image Resolution | $320 \times 240$ |
| Map Feature Circular Buffer Size | 7000 |

sensors, so the previous 20 vehicle states and their respective covariance matrix were kept in a circular buffer [137].

When the visual measurements were received as a ROS message, the EKF-update node performed the update procedure. In the update procedure, firstly the timestamps of the visual measurements were matched with the nearest neighbour from the circular buffer (experimentally the vision time delay was found to be less then $120msec$, so within 8 slots of ring-buffer the match was found). The covariance inversion was performed using the cholesky implementation found in the Eigen library [135]. After the EKF update was performed then the error was propagated to all the state stored in ring-buffer until the current state. The overall processing of the vision-node and EKF-update was provided in the in the Table 5.2. It can be seen that the feature matching is the most expensive part (can be speeded-up by using the GPU implementation of Opencv [134]).

The gravity/magnetometer measurements were estimated using the formulation provided in Chapter 4 (Equation 4.33) when-ever the IMU data was available. The weighted low-pass filter was implemented on the gravity/magnetometer measurements to avoid the abrupt jumps in the measurements (noise). As the aerial vehicle is dynamic in nature, so before EKF update, we make sure that the vehicle was in

Table 5.2: Front-end overall processing time on an embedded computer (Kinect update rate is 10Hz, whereas Front-end outputs at IMU rate 50 Hz

| Module | Processing time per frame ($ms$) |
|---|---|
| Data acquisition | 08.60 |
| Feature detection and matching | 43.17 |
| Motion estimation | 28.92 |
| Pose update | 19.05 |

approximate hover mode. For that situation, we had experimented with the norm of the gravity (less then 12g) and with the vision based relative translation (less then $0.4m$ for $6DOF$ measurements) for a decision criteria for the EKF update. The update procedure is similar to the earlier vision based update, however instead of full pose update, here we do only orientation update.

The front-end provides the vehicle pose and map (constant-size) estimates in real-time while the back-end node take care of global consistency. The key-frame selection implementation was based on distance/angular separation [139] between temporal frames. If the pose estimated from the front-end was greater than threshold (0.10m/0.174Rad/sec) then the key-frame was selected. The pose information was added into the SE3 array [135]. The RANSAC-SURF feature matching (as implemented earlier) was used to estimate relative transformation between the the current Kinect frame and earlier key-frames. If a valid relative transformation was found between them then the edge was created and the information was stored into SE3 array [135]. The edge between the non-consecutive poses were called loop closure constraints and other were called incremental constraints. The implementation of cholesky spare optimiser [101] was used to do the graph optimisation and to generate the consistent 3D trajectory. The update rate of the optimiser was approx 1Hz.

### 5.4.2 On-board Real-time Implementation of Position Controller

In this section the design and implementation details of the position controller for autonomous hovering is presented (that uses the SLAM output). There can be two

types of controller: one is a feedback controller, which estimate the error between measured output and expected output. The other is an open-loop controller, which act on a system without measuring the output of the system. In general, feedback controllers are used for autonomous platforms [18] and well known for their simplicity of implementation. So we have implemented a feedback PID controller as a proof of concept that Visual-Inertial SLAM is capable of controlling the vehicle autonomously.

One of the distinctive advantage of the PID controller is that, two controllers can be used together to yield another controller, called cascaded controller. The cascaded controllers yields betters performance as compare to single PID controller. Considering the dynamic nature of the hexacopter system, we have implemented the cascaded controller with two loops. The loops are

- Inner Loop: A attitude PI controller (100Hz) to control the attitude of the flying vehicle based upon gyro/accelerometer. This controller was implemented in the autopilot for vehicle stabilisation. It also contained a PI rate-controller to send the commands to the motors for stabilisation. The set point of this controller was controlled by outer loop (Visual-Inertial SLAM based Position Controller).
- Outer Loop: A position PID controller (50Hz) for autonomous hovering and based upon the 6DOF pose estimated from visual-inertial SLAM system.

The attitude controller provided a good foundation for the stable flight however the drift occurred (over time) due to IMU noises. The cascaded nature of the outer loop provided the position hold (drift-free flight) capability using the inner attitude controller (for flight stability).

Before going to the implementation details of the position controller, the foundation of the position controller are provided here. The aerial vehicle flies in the 3D environment (6DOF) but it requires 4DOF control commands i.e. roll, pitch, yaw and thrust [124]. Position control requires the hexacopter to be kept in place over a desired location. A position controller has four inputs, namely the desired position and

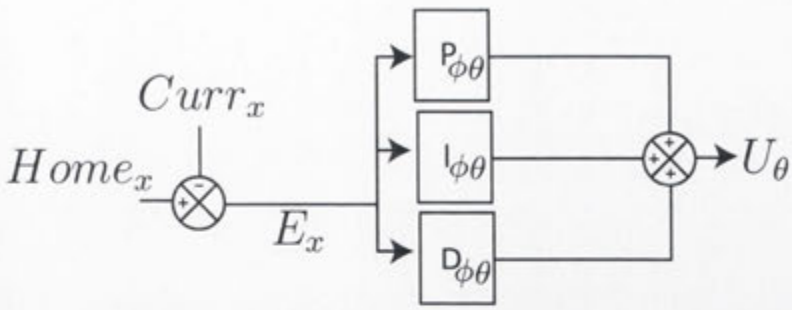



Figure 5.17: The PID position controller for Hovering.

the desired yaw angle ($\psi$). Movement in the $x-y$ plane is performed by tilting the hexacopter ($\phi, \theta$), so that the thrust vector had a component in the desired direction. The hexacopter's orientation in the plane was kept fixed on the original take-off heading. Altitude control refers to maintaining the height ($z$) of the hexacopter above the ground at a desired value. This is accomplished by controlling the magnitude of the thrust vector. The conversion from the input position information to the Euler angles commands i.e. roll, pitch and yaw commands ($U_\phi, U_\theta, U_\psi$ respectively) was achieved with this formulation:

$$\begin{aligned} U_\phi &= P_{\phi\theta} \times E_y + I_{\phi\theta} \times \int E_y dt + D_{\phi\theta} \times \frac{d}{dt} E_y \\ U_\theta &= P_{\phi\theta} \times E_x + I_{\phi\theta} \times \int E_x dt + D_{\phi\theta} \times \frac{d}{dt} E_x \\ U_\psi &= P_\psi \times E_z \end{aligned}$$

where $E_x, E_y, E_\psi$ are the errors in the $x, y, \psi$ axes, calculated with respect to a home position
($Home_x$,$Home_y$,$Home_\psi$ ) which is the vehicle position $\mathbf{P}$ (Equation 3.3) at reference hovering point. The current position $\mathbf{P}$ at time $k$ is defined as ($Curr_x$,$Curr_y$,$Curr_\psi$

). The errors are:

$$\begin{aligned} E_x &= Home_x - Curr_x \\ E_y &= Home_y - Curr_y \\ E_\psi &= Home_\psi - Curr_\psi \end{aligned}$$

$P_{\phi\theta}, I_{\phi\theta}, D_{\phi\theta}$ are the controller gains for roll and pitch; and $P_\psi, I_\psi, D_\psi$ are the controller gains for yaw. The proportional term ($P_{\phi\theta}$ & $P_\psi$ ) represents the present error. While too high values of this cause a system to turn unstable, low values affect the responsiveness of the system. The $I_{\phi\theta}$ accumulates the past errors and thus eliminates steady-state errors of purely proportional controllers. However, setting this too large causes overshoots. Finally, the differential term ($D_{\phi\theta}$) predicts future errors and thus can improve settling time and stability. The differential term is sensitive to noise, so large amounts of noise and a high value of this can turn the controller unstable as well.

The operation of the PID controller for the roll command is depicted in Figure 5.17. The y-axis corresponds to sideways movement of the hexacopter, and is therefore the direction in which error is measured. During the implementation, the integral term was calculated as a discrete running sum of all the error values since the algorithm was first initialised. To prevent the term from causing unstable control, the ceiling was placed on the absolute value of the integral term. The time derivative of the error was approximated as the difference between the present and previous error, divided by the time between updates. To place a maximum limit on the speed of the hexacopters movement, the absolute value of the sent roll command also was ceiling with 15o. The pitch controller worked identically to the roll controller, with the exception that error is measured in the x-direction. The yaw controller also worked similarly, with two exceptions. The error for the yaw controller was measured in radians. During experimentation it was also found that the yaw angle can be controlled by a proportional controller alone without resulting in oscillations or overshoot so it was a purely proportional controller and capped with 0.26 rad.

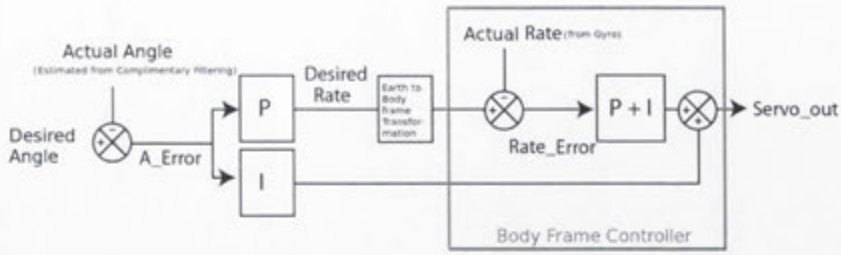



Figure 5.18: The Inner loop PI attitude controller implemented in the autopilot.

The inner loop of the cascaded position controllers was written in python on Pico-ITX using ROS framework. To be used for autonomous flight, some modifications were made to the Autopilot code. The most significant change was made to the MAVLink message processing module, to allow it to receive commands from the position controller i.e. the MAVLink packet (set_roll_pitch_yaw_thrust_send) was implemented in the autopilot code. It was able to retrieved the hexacopter position and orientation message that had been published by either the Visual-Inertial SLAM or VICON system. The first set of data it read from the topic was stored as the hexacopter's home position ($Home_x$,$Home_y$) and heading ($Home_\psi$). After this, it operated in a loop wherein it received a pose update. Then it calculated the appropriate error $E_x, E_y, E_\psi$ for estimating the roll, pitch, and yaw commands using PID controllers described earlier. The commands were sent to the autopilot and then the loop was repeated.

The inner loop of the cascaded controller was implemented on the hexacopter's autopilot. We implemented the attitude PI controller, keeping the hexacopter tilted at the desired angles sent from the position controller as shown in Figure 5.18. The attitude estimator implemented earlier [68] provided the actual angle of the flying platform however the desired angle was provided by the position controller. The error was fed to the PID gains and then earth to body transformation was carried out to send it to body frame controller. The body frame controller used the desire rate against the gyro rate to estimate the motor output signal.

Many theoretical methods for estimating the PID parameters exist [124] , but in

practice the parameters are often found experimentally. The process of choosing these parameters appropriately is called tuning the controller. Given some knowledge (transfer function) of the system, it is possible to systematically choose the values through the Zieger-Nichols method [140], or to manually estimate the parameters by trail and error. However, in our case due to the absence of any knowledge of transfer function, the parameters were tuned experimentally.

## 5.5 Cost Comparison of the Developed System

Table 5.3: Cost breakdown of a hexacopter and Payload System (less than 1300)

| Item | Qty. | Overall Cost |
|---|---|---|
| 3DR ArduCopter Hexa-B Frame Kit [125] | 1 | $130 |
| Hexa Power Distribution Board [125] | 1 | $13 |
| APC Propellers 11x47 Push-Pull Set | 3 | $14 |
| Propeller Fastener and Shaft Kit | 6 | $36 |
| 200mm clearance landing legs [126] | 1 | $95 |
| Autopilot system [125] | 1 | $68 |
| ESC 20 Amp | 6 | $68 |
| 880Kv AC2836-358 Motor [125] | 6 | $144 |
| Turnigy 3300 mAh 4 Cell 14.8V LiPo Pack | 2 | $65 |
| Mega Power 1350 mAh 3 Cell 11.1 Lipo | 1 | $28 |
| DE-SWADJ3 Voltage Regulator | 1 | $30 |
| IMU | 1 | $45 |
| Microsoft Kinect | 1 | $102 |
| LP-170C Atom Board | 1 | $279 |
| 32GB CompactFlash Memory | 1 | $97 |

A major motivation behind the development of the hexacopter platform and in-house build IMU, was to produce a low-cost alternative to the commercially available systems. The complete breakdown of the developed system is presented below in Table 5.3.

The total cost of the hexacopter system is under $US800$, meaning that the entire hexacopter + payload system can be assembled for less than $US1300$. It should also be noted that the prices for many items have dropped since the original items were

purchased, and that many equivalent parts are now available through third-party distributors (such as HobbyKing [141]). For example, an equivalent Frame-kit can also be purchased for $US80$, almost half of the paid cost [141]. The components for the combined hexacopter and payload could likely be purchased for under $US1000$ today.

Table 5.4: Comparison of cost of different hexacopters

| Model | Price (AUD) | Weight (kg) | Payload (kg) |
|---|---|---|---|
| Custom | $1,252 | 2.0 | 1.2 |
| ASCTech Firefly [121] | $9,100 | 1.0 | 0.6 |
| Droidworx XM-6 [142] | $10,660 | 3.0 | 1.0 |
| Cinestar 6 [143] | $6,600 | 2.6 | 2.0 |

The total price of the system is far cheaper than the alternatives, as shown below in Table 5.4. Although there is not much difference in the functional lifting ability of the different hexacopter models. However the difference become apparent, when we considered the autopilot and payload system. The autopilot and payload processing board is much inferior to its counter-parts [121], for an example, the Pico-ITX board is a dual-core atom board however the ascending technologies Firefly [121] supports the Intel i7 board, hence can afford more intensive data processing.

## 5.6 Summary and Discussion

In this chapter, we described the hardware development of the aerial platform and the implementation of the Visual-Inertial SLAM and attitude/position controller algorithms. The motivation behind this hardware and software development was to build a low-cost aerial platform with an efficient real-time SLAM pipeline (a theory developed in Chapters 3 & 4) to control the flying vehicle (position hovering).

Firstly, we described the developed hardware of the hexacopter system. The payload and autopilot system were also discussed. Furthermore, we provided the design and manufacturing process of the customised IMU development.

Secondly, a major portion of the chapter was devoted to the on-board real-time implementation. The Visual-Inertial SLAM implementation and parameter details were discussed. How the computational factor was considered while deciding some parameters was also discussed. Then the controller design and implementation were presented. All these provide the autonomous capability to the bare hardware of the hexacopter system.

Finally, the cost breakdown was provided and a comparison to a state-of-the-art off-the-shelf system was provided. The results of the implemented modules discussed here will be provided in the next chapter.

# Chapter 6

# Experimental Results

## 6.1 Introduction

This chapter provides the experimental results of the Visual-Inertial SLAM system (which was developed in Chapters 3 and 4). Firstly, the flight test environment and the ground station setup will be described, which are used for both real-time demonstrations. Secondly, the real-time results of the Visual-Inertial navigation system for an indoor environment with real-time hovering control are given. Finally, the results of the outdoor navigation system will be presented.

The dataset includes the Kinect frames (RGB-D) taken at $22Hz$ and inertial data acquired from the in-house developed IMU at $50Hz$. The ground truth data (VICON data of less than $1cm$ and $1^{\circ}$ error) is also acquired at $100Hz$ for evaluation purposes in the indoor environment. All the data is processed in real-time on a dual-core Atom board (Pico-ITX computer) running the Robot Operating System (ROS) framework. All the data is synchronised with the time-stamp generated during the data acquisition of each respective sensor.

During the flight experiments, all the sensor data is published to ROS topics, which is recorded via bag format [130] [1]. This allows all commands, the VICON pose

[1] A built in data-recording package of ROS.

(a) (b)

Figure 6.1: Indoor environment with VICON setup.

information and the Visual-Inertial pose information to be retrieved for the purpose of analysis. In all the experiments, both the Visual-Inertial SLAM and VICON pose estimates were recorded simultaneously at the rate of 10Hz for comparison [2].

## 6.2 Experimental Setup

The indoor environment (at the Research School of Information Sciences and Engineering (RSISE) at ANU) is a lab with a volume of roughly 8m × 6m × 4m, cordoned off with a net to provide a safe and isolated flying environment. In this lab, a state-of-the-art VICON motion capture system is installed. The VICON system is an infrared marker-tracking system that can track 3D spatial displacement with 6DOF. The system uses an arbitrary arrangement of cameras outfitted with infrared (IR) optical filters and IR-LEDs in conjunction with an asymmetrical set of reflective markers placed on a subject. Due to the IR filters, the cameras only recognise the reflections from the markers. The indoor environment at KSISE has nine tripod-mounted VICON cameras distributed at different heights in three corners of the flying space. The windows are covered with plastic sheets to reduce ambient IR radiation during the

[2]The rate of the autopilot to send the messages back to the ground station was limited to 10Hz (in order to avoid extra computation for message parsing).

Figure 6.2: Outdoor unstructured forest environment.

day, as well as to protect the glass in the event of a crash. This indoor environment is shown in Figure 6.1.

Five reflective IR markers were fixed to the arms of the hexacopter, allowing the VICON cameras to estimate its pose. These markers are small balls that have been covered in reflective tape and are tracked by the VICON cameras using an on-board processor. Note that for an object to be tracked, the configuration of the markers must not be collinear or symmetrical. Although only three markers are strictly required for tracking, it is better to have more to prevent the loss of tracking in the event of the object becoming occluded. The quantitative comparison of the proposed work against the highly precise VICON reveals a robust approach in different challenging environments.

The outdoor flight tests were performed in a cluttered tree environment (i.e. forest) at the Australian National University, where depth dropouts happen due to limited range and ground clearance by the flying vehicle. A typical flight test time was around 5-10 minutes and the nominal flight height was 10m above the ground. The maximum ground speed was approximately 5-7m/s. The environment was challenging due to a lack of GPS aid and the unstructured nature, as shown in Fig 6.2. The unconstrained forest-like environment is different from indoor and outdoor urban environments, as specific assumptions regarding the structures (e.g. walls, roads and building) cannot be made. The outdoor environment provides a good test-bed for our framework to be evaluated for the RGB-D and depth dropout cases. Please note that for the

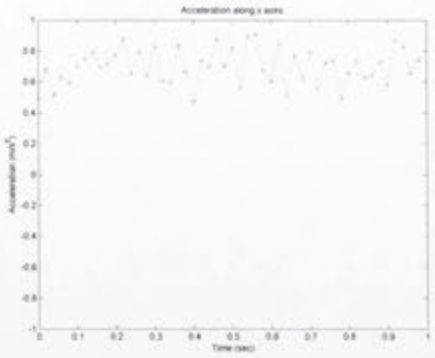
indoor environment the depth dropout case is simulated by intentionally dropping the depth frames (discussed in Section 6.3). However, for the outdoor environment, the depth dropout happens frequently due to aerial vehicle ground clearance or sunlight interference.

### 6.2.1 Sensor Calibration

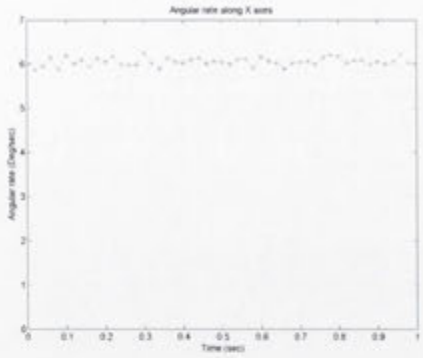
Calibration is the process of determining sensor systematic errors. Before the evaluation of the proposed approach, the calibration of the IMU and Kinect must be known precisely. The 6DOF transformation between the IMU and visual sensor also needs to be known in order to fuse the measurements of the complimentary sensors. This section provides the details of the calibration procedure carried out during the experiments.

For IMU calibration, the deterministic errors (such as the fixed bias and scale factor error) are estimated initially when the IMU is in static condition. The stochastic errors (such as random-walk) are calibrated dynamically using the online estimation of accelerometer and gyro biases (discussed in chapter 3). The calibration procedure which was proposed in our earlier work is adopted here [68]. The raw data samples were collected from IMU when it was left unchanged on the flat surface of table for 5 seconds. The data was acquired at 50Hz and only 2500 data samples were collected in order to determine the mean bias for gyroscope and accelerometer. The ideal gyroscope will produce an angular rate when a known rate of rotation is applied to it. This output is free from noise, perfect and no bias. However, a low cost sensor such as MEMS based IMU developed in this thesis is not perfect and needs calibration. Figure 6.2.1 illustrates the angular rate and acceleration along the three axes (x, y, z) of a stationary IMU without the calibration. In this experiment, the unit for angular rate of rotation is degrees per second and for acceleration is meter per second squared. Table 6.1 shows the mean bias of angular rate and acceleration along the three axes.

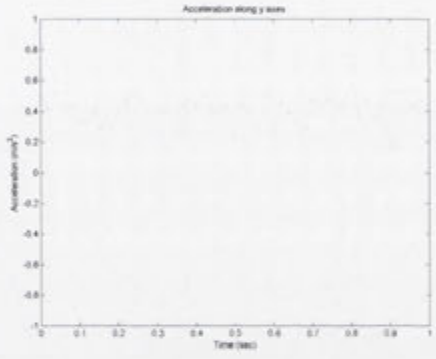
The following equations provide a simple calibration model for a MEMS based gyroscope and accelerometer bias (Eq. 6.1) and scale factor (Eq.6.3 6.4).



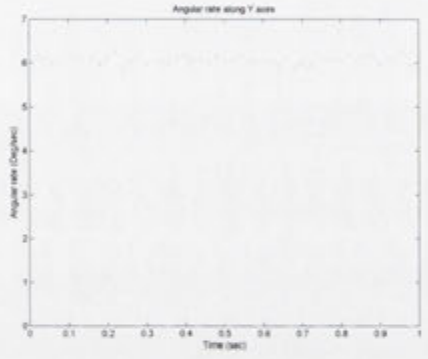

(a) Accelerometer samples along x-axis



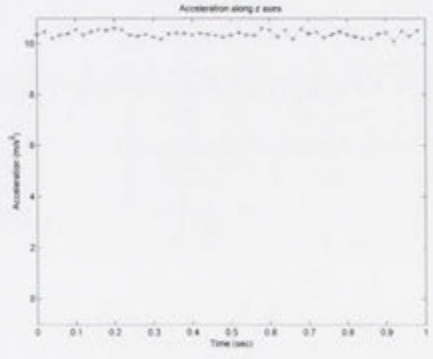

(b) Gyro samples along x-axis



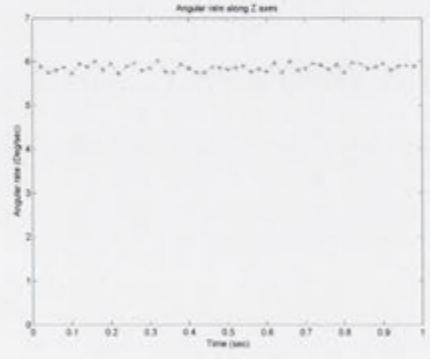

(c) Accelerometer samples along y-axis




(d) Gyro samples along y-axis




(e) Accelerometer samples along z-axis




(f) Gyro samples along z-axis

Figure 6.3: Un-calibrated IMU for Accelerometer and Gyro data along it's three orthogonal axes.

Table 6.1: Mean value of Acceleration and Angular rate for Un-calibrated IMU

| | Angular Rate (deg/s) | Acceleration ($m/s^2$) |
|---|---|---|
| x-axis | 6.0517 | 0.6967 |
| y-axis | 6.0643 | 0.4105 |
| z-axis | 5.8570 | 10.3795 |

$$\begin{aligned} f^b &= (\hat{f}^b - \beta_f)S_f \\ \omega^b &= (\hat{\omega}^b - \beta_\omega)S_\omega \end{aligned} \tag{6.1}$$

where $f^b$ and $\omega^b$ are the calibrated values of the quantity, and $\hat{f}^b, \hat{\omega}^b$ are the direct reading from the sensors, and $S_f, S_\omega$ are the scale factor for the accelerometer and gyro respectively. The fixed biases associated with accelerometer $\beta_f$ and gyro $\beta_\omega$ are estimated for static IMU experiments using Table 6.1.

The scale factor of the gyroscope is measured by using the information from the gyroscope's data sheet [144]. Scale factor of gyroscope is calculated:

$$S_\omega = \frac{\hat{S}_\omega}{\chi}\frac{\pi}{180} \tag{6.2}$$

where

$$\chi = \frac{Ref_\chi}{2^n - 1} \tag{6.3}$$

where $\hat{S}_\omega$ is the sensitivity of gyroscope, $\chi$ is the ADC steps ,$n$ number of ADC bits of gyroscope, and $Ref_\chi$ is the ADC reference voltage. The scale factor readings for gyros are in radians per second.

The scale factor of the accelerometer is measured by using the information from the accelerometer's data sheet [145]. Scale factor of accelerometer is calculated:

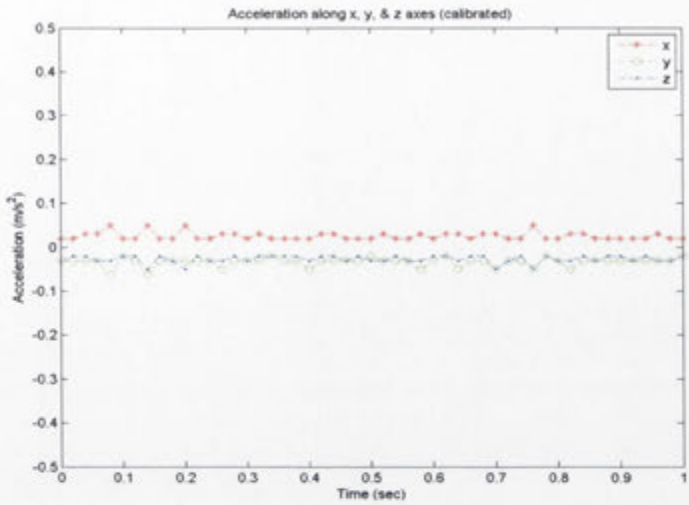



Figure 6.4: Calibrated Accelerometer.

$$S_f = g/t_s \tag{6.4}$$

Where $t_s$ are total steps for 1g that $t_s = \frac{1000mg}{S_{acc}}$. The $S_{acc}$ is the sensitivity of the accelerometer and is used as $S_acc = 4mg/LSB$ from the data sheet [145].

Figure 6.5,6.4 shows the angular rate and accelerometer reading along the three orthogonal axes of IMU after a stationary calibration. As can be seen that the angular rates and acceleration after the calibration are approximately fixed. Table 6.2 shows the average angular rate and acceleration values after calibration.

Table 6.2: Mean value of Acceleration and Angular rate for calibrated IMU

| | Angular Rate (deg/s) | Acceleration ($m/s^2$) |
|---|---|---|
| x-axis | 0.0262 | 0.0104 |
| y-axis | -0.0315 | -0.0100 |
| z-axis | -0.0274 | 9.8462 |

After the accelerometer and gyroscope calibration, we did the magnetometer and camera calibration. The magnetic measurement can be affected by two types of

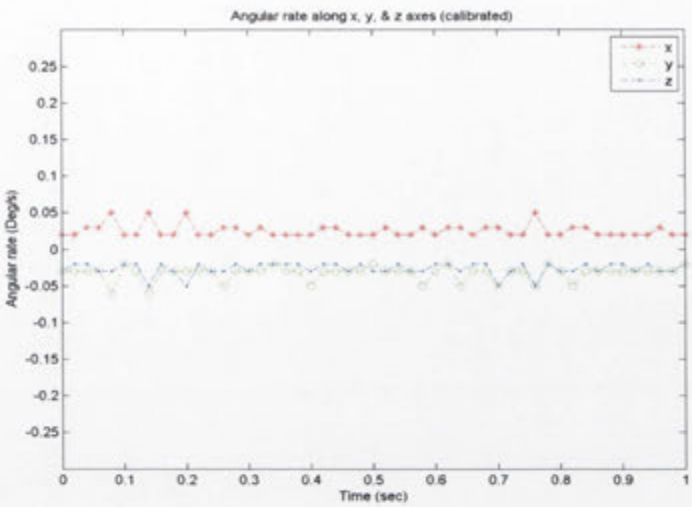



Figure 6.5: Calibrated Gyroscope.

distortion that is hard and soft iron distortion [3]. We performed a soft iron calibration using the approach proposed in the work of [146]. For the Kinect calibration the calibration parameters are stored in the device's internal memory however the factory setting's are not adequate for precise pose estimation. For the Kinect calibration, we adopted the recent work from [102] in which a checker-board is used to provide accurate intrinsic and extrinsic parameters using the re-projection error of the RGB and depth camera with bundle adjustment based refinement.

Finally the calibration between the vision sensor and IMU, which plays an important role to effectively fuse the information is provided. The angular alignment between the two sensors can be calculated in close-form [11] however the translation distance is difficult to compute [4]. For camera-IMU rotation calibration, we adopted the approach proposed by [147] [5]. The rotation alignment is estimated by using the vertical direction of the IMU using the direction of gravity and camera vertical direction.

[3]The hard iron distortion is causes by the nearby objects that produce magnetic field however the soft iron distortion is caused by the earth's magnetic field.

[4]The translation offset is the difference of the measured translation by both sensors due to a centric rotation axis.

[5]The translation offset is measured generally using spin-table however we used rough translation calibration by manually measuring the distance between IMU and camera centre.

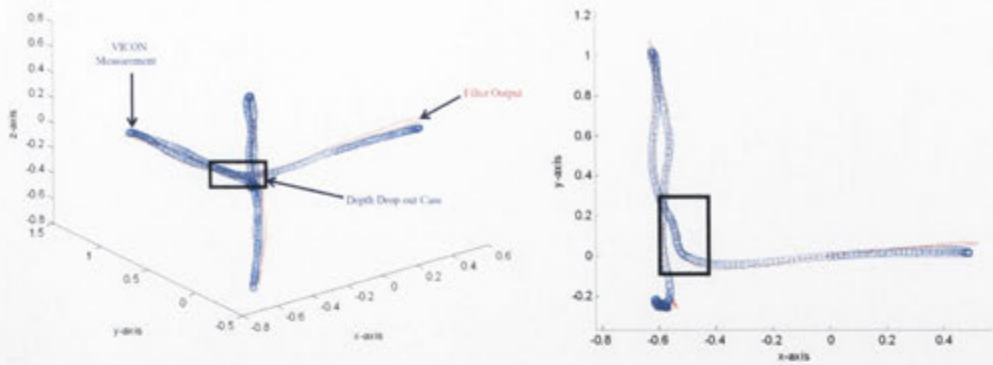
## 6.3 Real-time Indoor Flight Results

Six separate experimental flights were conducted in the indoor environment. The goals of these flights were twofold: firstly to test the ability of the position control to maintain a stationary hover, and secondly to evaluate the proposed approach with the ground truth recorded using the VICON system. The position controller was fed with the proposed Visual-Inertial SLAM, with the vision node running at $10 - 15$Hz and final output rate was sampled up to 50Hz (using the IMU data in the EKF filter).

For each flight, the hexacopter was placed in the approximate centre of the flight space. The rest of the flight is then conducted by following these steps:

- The VICON system is initialised (and calibrated if necessary [131]).
- The Atom board mounted on the hexacopter is turned on.
- The Atom board is remotely logged into via SSH [57] to send the commands from the remote computer.
- On the Atom board, VICON data capturing is run [131].
- On the Atom board, the Visual-Inertial SLAM package is initialised by running the vision and IMU nodes.
- On the Atom board, the position controller node is started. It stores the current position of the hexacopter as its home location upon initialisation.
- Finally, R/C transmitter of the hexacopter is used to switch from manually to automatic control. During all the times, the operator carefully monitors the flight performance and if necessary switches back to manual flight mode.

In the first flight 900 Kinect frames and 1501 IMU packets were recorded. Although the hexacopter deviated slightly from its home position throughout the flight, it always returned to the home location. Therefore, the flight was deemed a successful



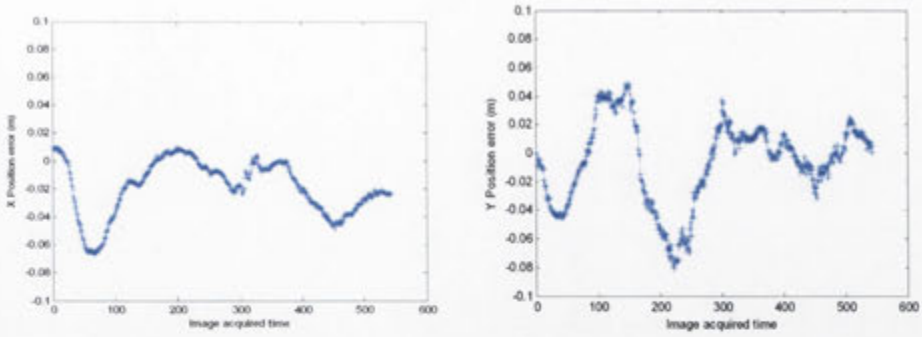

(a) 3D view (b) XY view

Figure 6.6: Trajectory comparison of proposed Approach against VICON system.

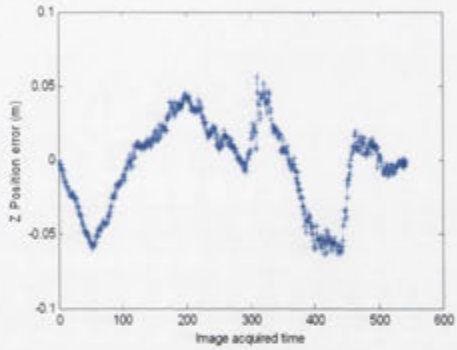
demonstration of the position controller (this experiment was conducted various times to confirm the repeatability of the result).

The hexacopters trajectory as recorded by the VICON and Visual-Inertial SLAM systems is shown in Figure 6.6. The trajectory depicts the take-off and wandering of the hexacopter platform, but there is no data for the landing. From this data, it is apparent that the Visual-Inertial SLAM approach matches the ground-truth VICON data well, and it was calculated that the root mean squared error between the two data sets was less than 0.2m and 0.5°. We can also observe that the hexacopter deviated a maximum of 0.5m and 1m from the home position in the x and y directions respectively.

Each time-step in the following plots (Figures 6.7) corresponds to the sending message rate, which was already stated to be 10Hz. The difference between the ground truth VICON data and the Inertial-Kinect SLAM data in the x, y, and z axes is shown below in Figure 6.7. It can be seen that with the exception of a few outliers, the data points agree within a tolerance of less than 0.1m. The root mean squared error was calculated to be 0.039m along the x axis, 0.035m along the y axis, and 0.01m along the z axis. Comparisons of the x position with the pitch of the hexacopter, and of the y position with the roll of the hexacopter are shown below in Figure 6.8. Note that the initial period until just after time-stamp 150 (the first 15 seconds of data)



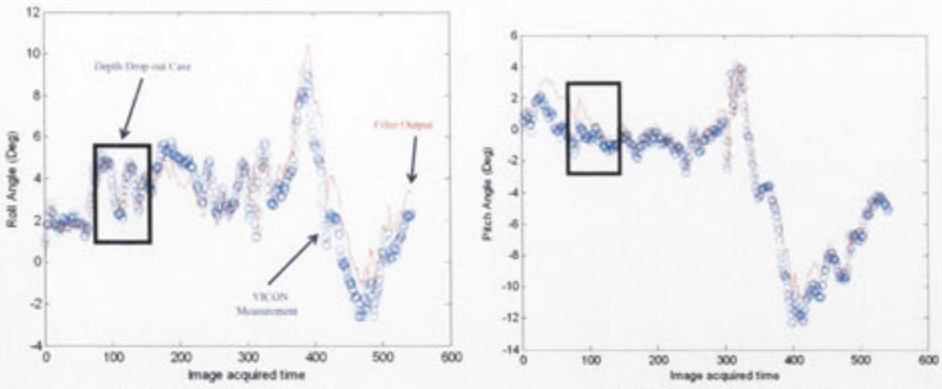

(a) x-axis difference (b) y-axis difference




(c) z-axis difference

Figure 6.7: Difference between VICON and Inertial-Kinect SLAM positions (The horizontal axis shows the time in second on acquiring image frames.).

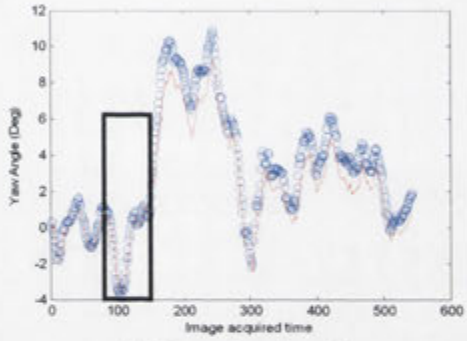
corresponds to when the position controller algorithm has been initialised but the hexacopter is still sitting on the ground. After that, it is interesting to note in Figure 6.8 that although the x position of the hexacopter returns to the home location, the pitch value never returns to zero but instead is always negative. This demonstrates the necessity of the integral gain in the PID controller, showing that there was some natural bias in the hexacopters movement that had to be compensated for. The pitch values from the Inertial-Kinect SLAM algorithm also appear to be significantly noisier than the roll values, but the reason for this is undetermined. In Figures 6.8 we can observe that although there is a clear maximum peak in the position error just after time-stamp 300 (30 seconds into the recording), there is no corresponding maximum peak in the roll values. This is evidence of the ceiling on the PID controller




(a) Roll attitude angle




(b) Pitch attitude angle

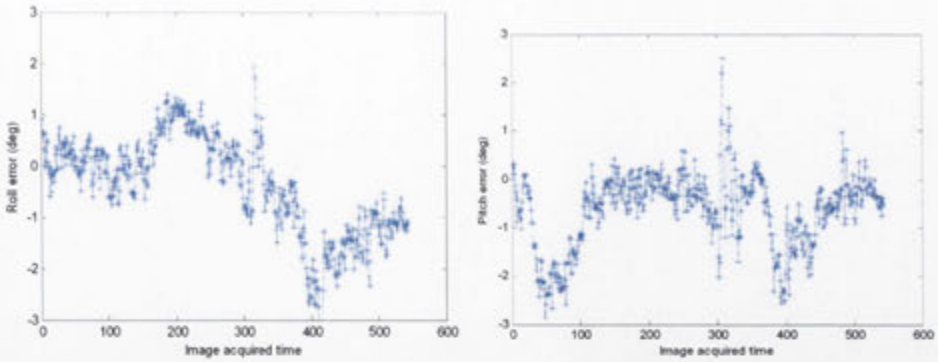



(c) Yaw attitude angle

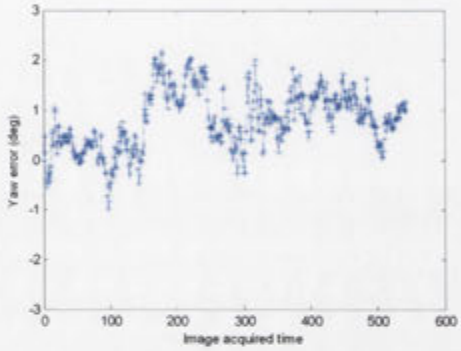
Figure 6.8: Attitude comparison between VICON and Inertial-Kinect SLAM approach.

values, which prevents the hexacopter from overcompensating and becoming unstable. The yaw is not maintained at 0 degrees, showing that the controller is not perfect. However, it is functioning, as is evidenced by the fact that the yaw values oscillate within a small range of the desired heading. We again see that the ground-truth VICON and Visual-Inertial SLAM estimates agree fairly closely. Unlike the roll and pitch commands, which are a function of the x and y positions, the yaw commands are a function of the current and past yaws. Therefore, there is nothing to compare this plot with.

The difference between the ground truth VICON data and the Inertial-Kinect SLAM data in the roll, pitch, and yaw angles is shown below in Figure 6.9. It can be seen




(a) Difference in Roll angle (b) Difference in Pitch angle




(c) Difference in Yaw angle

Figure 6.9: Attitude Error comparison between VICON and Inertial-Kinect SLAM approach.

that with the exception of a few outliers, the roll and pitch data points agree within a tolerance of less than 1 . However, the yaw data points seem to disagree more. The root mean squared error was calculated to be 0.11 for roll, 0.30 for pitch and 1.06 for yaw.

The second flight using the Inertial-Kinect SLAM algorithms output as the position controllers input also lasted approximately 45 seconds. Unlike the first flight however, the hexacopter became unstable and the thrust had to be cut to prevent the hexacopter from reaching the limits of the flight space and crashing into a wall. Repeated tests in this scenario last varying durations of around a minute in length, but ultimately had to be cut short for the same reason. The primary reason for this is believed to be the slower update rate of the Inertial-Kinect SLAM algorithm (50Hz)

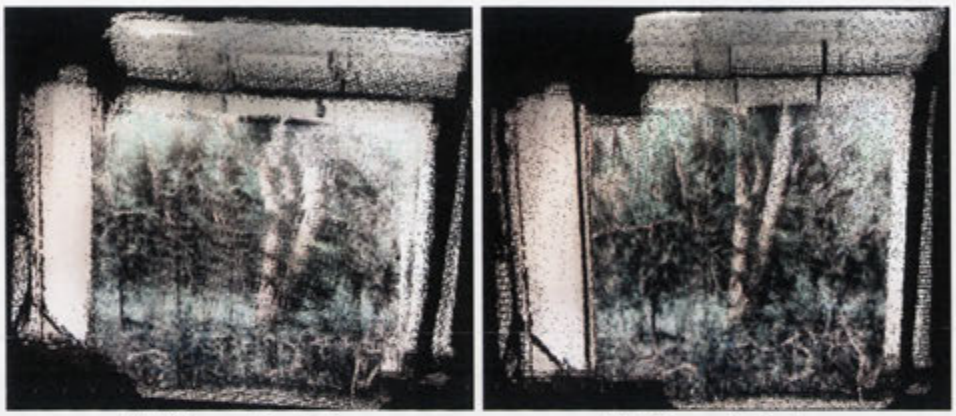
compared to the VICON motion capture system (100Hz). This problem could further exacerbated if MAVLink packets are being dropped. Such a problem is very possible and could be due to a variety of reasons such as overflowing buffers on the micro-controller or the quality of the wireless connection (which could be affected by noise from the motors and ESCs, as well as competing signals in a similar spectrum from the R/C transmitter). The slower update rate prevents smooth control, which is a significant issue in a high power and highly dynamic system such as the hexacopter.

Even in the first flight with high frequency VICON data, wandering of up to 1m was observed. One of the possible reasons for this is the manual control of altitude with the R/C transmitter. When the hexacopter is not perfectly level, some of the vertical thrust is lost (as it is converted to lateral movement). This causes the hexacopter to drop, meaning that the operator has to increase thrust temporarily to prevent the hexacopter from hitting the ground. However, a human operator lacks the reaction time and finesse to make this adjustment correctly. The resulting late overcompensation can cause the hexacopter to overshoot, increasing the instability of the system. Another potential cause of the systems lack of stability lies in the IMUs. This is particularly apparent in the payload IMU, which is of lesser quality than the one contained within the APM. Vibrations from the motors propagate through the hexacopter frame and introduce noise into the IMU data. This is obviously an issue in a system which is using the IMU directly for control, and not just navigation or pose estimation (the APM IMU is used to determine whether the hexacopters roll and pitch match the commands being sent, while the payload IMU affects the continuity and accuracy of the data from the Inertial-Kinect SLAM algorithm).

We have also compared the motion estimates of the ground-truth with the Inertial-Kinect framework (with/without monocular constraints). The Root-Mean-Square Error (RMSE) is used as a quality metric to obtain an error between the estimated pose estimate and the ground truth data as shown in Table 6.3. It can be observed that the inertial Kinect SLAM closely resembles with the ground truth data (with RMSE of less than $0.2m$ and $0.5°$), however monocular Visual-Inertial due to scale unobservability drift (and shows larger error).

Table 6.3: Evaluation of proposed approach against VICON outputs (RMSE error)

| | $P_x(m)$ | $P_y(m)$ | $P_z(m)$ | $\phi$ (°) | $\theta$ (°) | $\psi$ (°) |
|---|---|---|---|---|---|---|
| Monocular Visual-Inertial SLAM with VDC [17] | 2.61 | 3.13 | 1.79 | 1.07° | 1.01° | 0.81° |
| Inertial Kinect SLAM | 2.43 | 1.75 | 2.66 | 3.84° | 2.70° | 2.14° |
| Inertial Kinect SLAM with VDC | 0.05 | 0.04 | 0.19 | 0.40° | 0.04° | 0.48° |



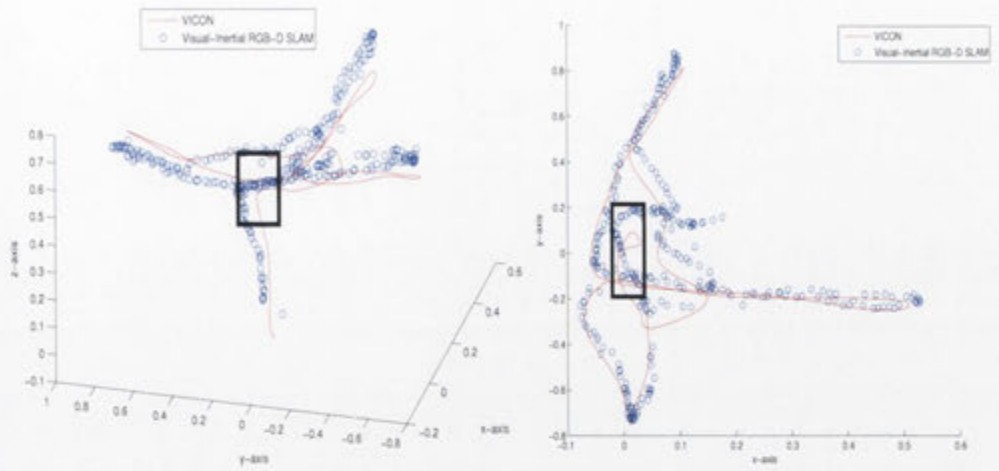
(a) Before optimisation (b) After optimisation

Figure 6.10: Graph optimisation, where the room wall was textured with forest like environment.

In addition to evaluating the front-end motion estimates, the back-end mapping was applied on the selected key-frames (detailed in section 4.7), to further minimise the drift as shown in Figure 6.10 (before and after pose graph optimisation).

## 6.4 Real-time Hovering and Control

To fully evaluate the proposed idea, a comparison of the motion estimates of the proposed approach is provided against the VICON ground truth. The hexacopter is flown in a closed loop using a Proportional Integral Derivative (PID) controller from VICON measurements in a $8m \times 6m \times 3m$ indoor environment. The indoor aerial dataset consists of 900 Kinect frames and 1501 IMU packets. To simulate the

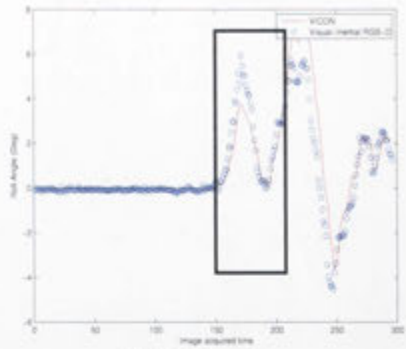



(a) 3D view for XYZ axis trajectory (in meters) (b) 2D view for XY axis trajectory (in meters)

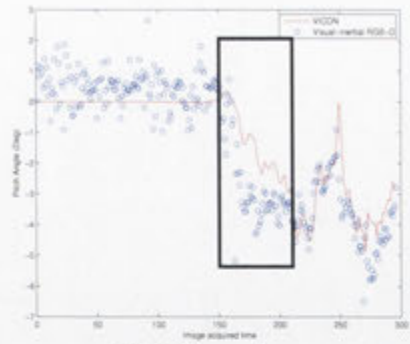
Figure 6.11: Real-time trajectory compared with the VICON system (The rectangular box shows the use of monocular constraints at depth dropout case.)

depth dropout case, we intentionally drop the depth data for some Kinect frames to evaluate the robustness of our proposed approach. Fig. 6.11 shows the translational comparison of proposed approach against the ground truth data whereas the depth dropout cases are highlighted with a rectangular boxes (during which monocular directional constraints are utilised by the proposed approach for motion estimation). The trajectory shows the take-off and lateral movements of the hexacopter platform. Fig. 6.12 shows the attitude comparison of the proposed approach with the ground truth where Euler angles are used for visualisation purposes. The difference between the ground truth VICON data and the Inertial-Kinect SLAM data in the roll, pitch, yaw angles and x,y,z in position is shown in Figure 6.13.

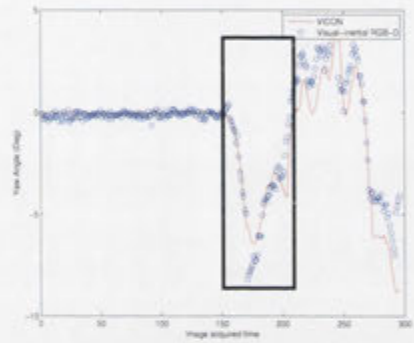
Before proper autonomous flight experiments could be conducted, it was essential that the parameters of the PID position controller were tuned. This was done manually, without deriving or testing an appropriate model for the hexacopter through kinematic analysis (as such a process can take up to a year on its own, with no promise of a use able result). The tuning process was conducted through trial and



(a) Roll angle



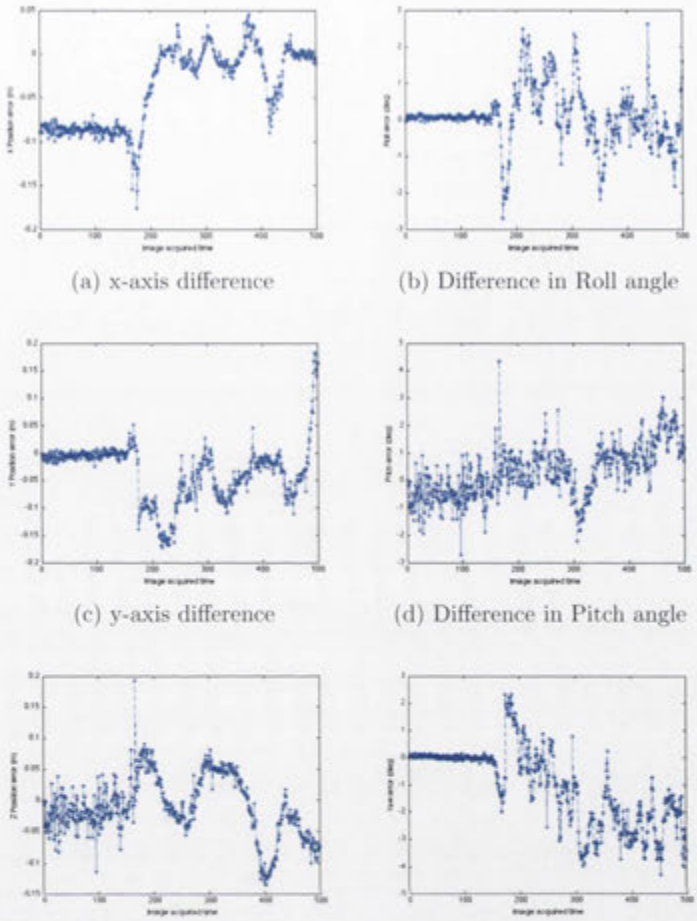
(b) Pitch angle



(c) Yaw angle

Figure 6.12: Attitude comparison of proposed approach against VICON system (the rectangular box shows the use of monocular constraints at depth dropout case.)

error over a large series of short flights. The proportional gain on the yaw controller was tuned first. This kept the hexacopter facing in one direction for all subsequent flights, allowing the results of adjusting the roll and pitch gains to be more easily understood. In the yaw experiments, the hexacopter was placed on the ground facing any direction. After initialising the position controller, a small amount of thrust was applied with the R/C transmitter until the hexacopter lifted off the ground. After a few seconds of flight in which the hexacopters behaviour was determined, the thrust was reduced and the hexacopter landed, concluding the experiment. Over the course of several experiments, the proportional gain for the yaw was slowly increased until the natural drift in yaw was counteracted. After this, several experiments were con-



(a) x-axis difference (b) Difference in Roll angle

(c) y-axis difference (d) Difference in Pitch angle

(e) z-axis difference (f) Difference in Yaw angle

Figure 6.13: Difference between VICON and Visual-Inertial SLAM pose.

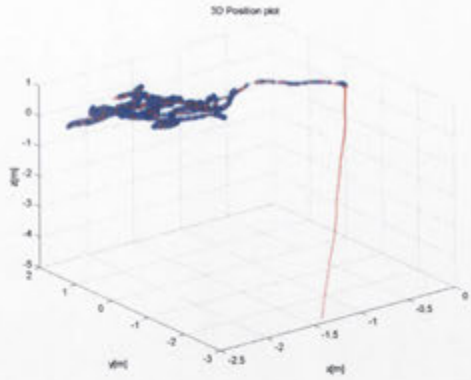
ducted where the position controller was initialised (thus storing the home heading) and then the hexacopter was rotated 45 or 90 before take-off. This allowed us to verify that the hexacopter would return to the home heading from any angle without noticeable overshoot. The yaw controller with only proportional gain proved to be highly reliable and effective, so the integral and derivative terms were not used in the controller With the yaw controller functional, the roll and pitch were simultaneously tuned (since they share identical PID gains). As before, the proportional gain $Kp$ was tuned first, with $Ki$ and $Kd$ set to be zero. To tune the proportional gain, the

position controller was initialised while the hexacopter was sitting on the ground to set the home position. Before take-off, the hexacopter was shifted 1m in the x or y direction. Thrust was then increased until the hexacopter took off, after which a constant thrust was maintained until the hexacopters behaviour was determined. At this point, the thrust was reduced, concluding the flight. With sufficient proportional gain, it was observed that the hexacopter returned to the home position. The proportional gain was slowly increased until it was observed that the hexacopter was beginning to overshoot the home position upon its return. After this, the proportional gain was left as it was. A new series of flights was conducted in which the hexacopter took off from the home position. As it was observed that the hexacopter slowly drifted from the home position during a longer flight, the integral gain $Ki$ was increased until this behaviour ceased. Finally, it was noted that although relatively stationary over the home position, the hexacopter was oscillating in attitude. Therefore the derivative gain $Kd$ was introduced to dampen the oscillations.

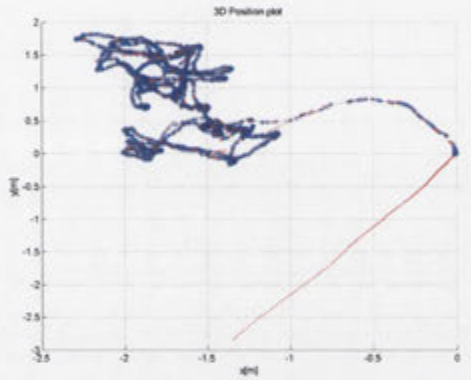
## 6.5 Real-time Outdoor Flight Results

The indoor experiments with simulated dropouts provides a quantitative error analysis of the proposed approach whereas the outdoor experiments were conducted to test the performance in real dropout cases. The testing environment is under heavy tree foliage (GPS-denied environment) with natural depth dropouts happen due to limited range and ground clearance of the flying vehicle. An area of $10m \times 12m$ is explored with 1701 RGB-D Kinect frames (whereas 240 depth frames are partially or completely unavailable due to range-limitation or sunlight interference). The 3D trajectory of the aerial vehicle which is estimated in real-time is shown in Figure 6.14. The input image of the outdoor test case along with map data is shown in Figure 6.15, the pose graph optimisation is also applied on the outdoor sequence.

As we are using the low-cost inertial sensor so the bias estimation plays an important part for stable navigation. The dynamic bias for accelerometer a noisy trend due to low-cost quality of inertial sensor (fluctuating pattern is due to frequent correction




(a) 3D view



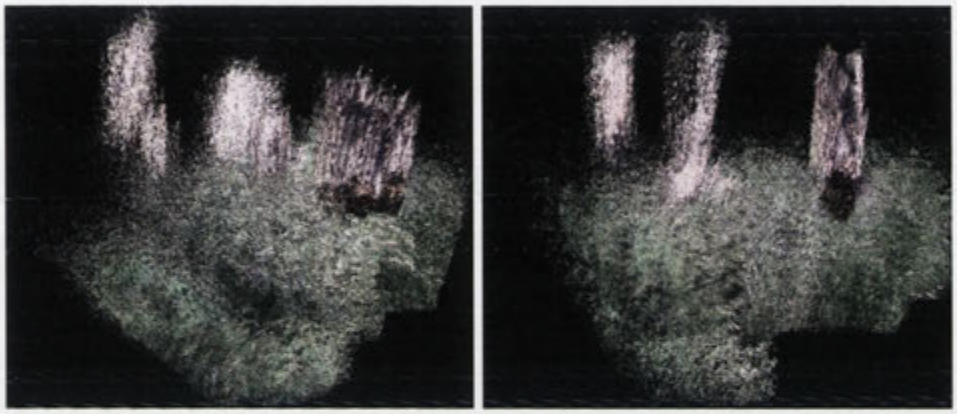

(b) XY view

Figure 6.14: 3D Flight trajectory of the hexacopter for outdoor sequence.

from the bias terms). Figure 6.16 shows the pose comparison of estimated output (with and without bias estimation) against the 6DOF/5DOF measurements. The vehicle attitude information with and without bias estimation is shown in Figure 6.17. As can be seen that the bias estimation results are correlated to the 6DOF measurements however without bias estimation the results diverge.

Figure 6.18 shows the normalised innovation from the EKF predicted pose and measured pose (a big surge at sample 500 is due to fast dynamic motion of hexacopter platform introducing a blur in the monocular image.) The normalised innovation is within 95.5% range of the Chi-square distribution showing the filter consistency (Chi-square conformity test) [64]. The position uncertainty (Figure 6.19) and at-

(a) Input image (b) Generated map



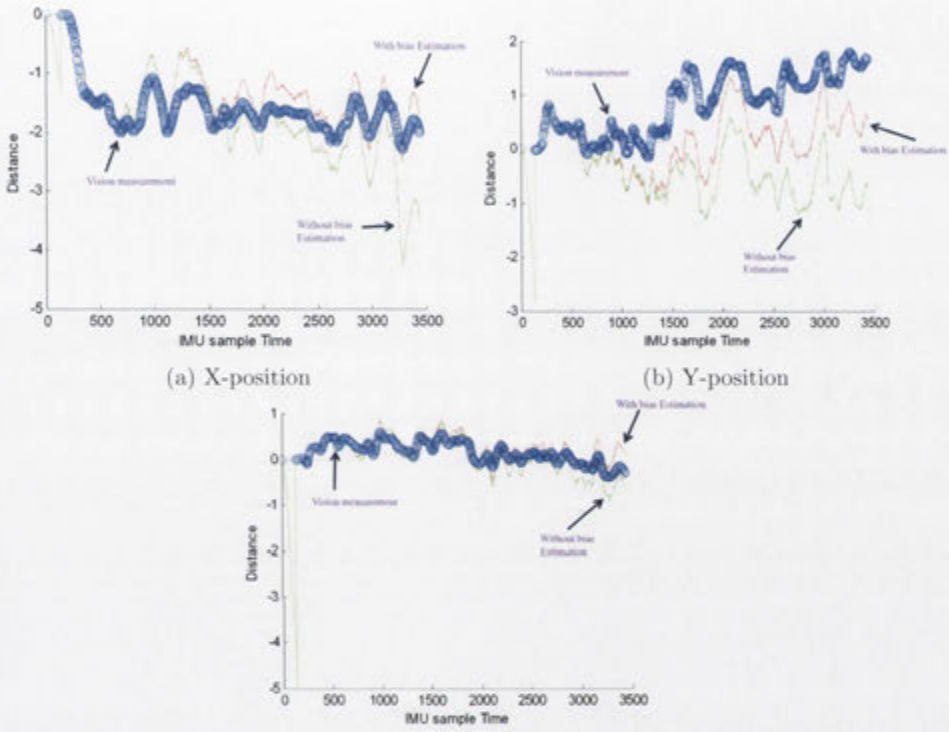
(c) Before pose-graph optimization (d) After pose-graph optimization

Figure 6.15: Proposed approach for the flight dataset in outdoor cluttered environment.

titude uncertainty (Figure 6.20) with $3\sigma$ is showing stable behaviour. The results shown here demonstrate that the proposed algorithm is capable of operating in different challenging environment, and producing very accurate motion estimates.

## 6.6 Summary and Discussion

This chapter described the experimental results of the Visual-Inertial SLAM system. The flight test environment and the ground station set-up have been described, which

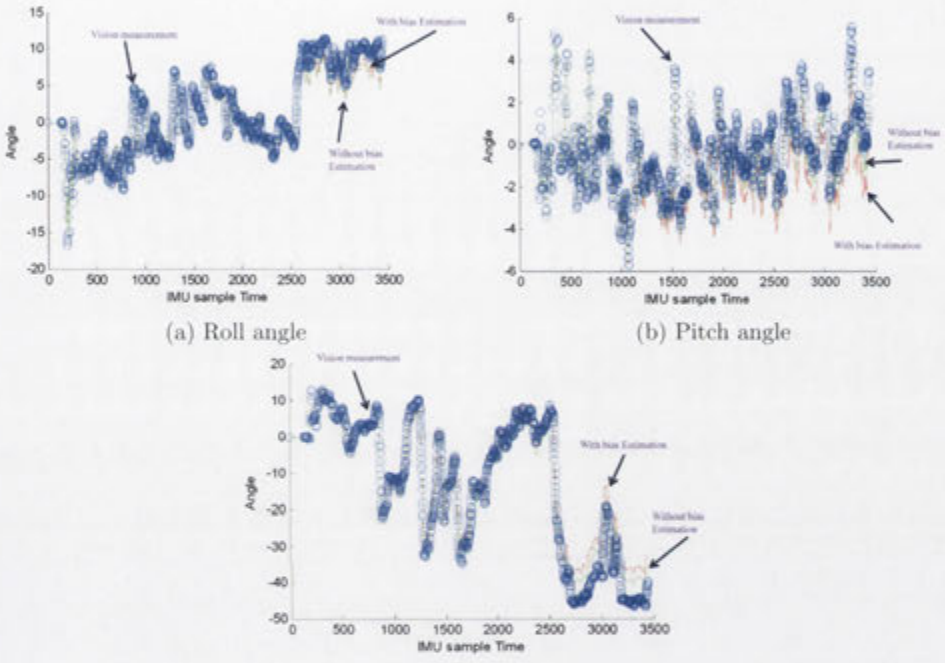



(a) X-position

(b) Y-position

(c) Z-position

Figure 6.16: Position information with/without accelerometer bias estimation for proposed approach.




(a) Roll angle

(b) Pitch angle

(c) Yaw angle

Figure 6.17: Vehicle attitude with/without gyro bias estimation for proposed approach.

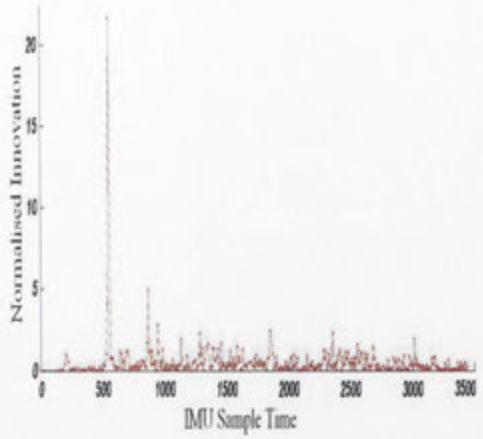
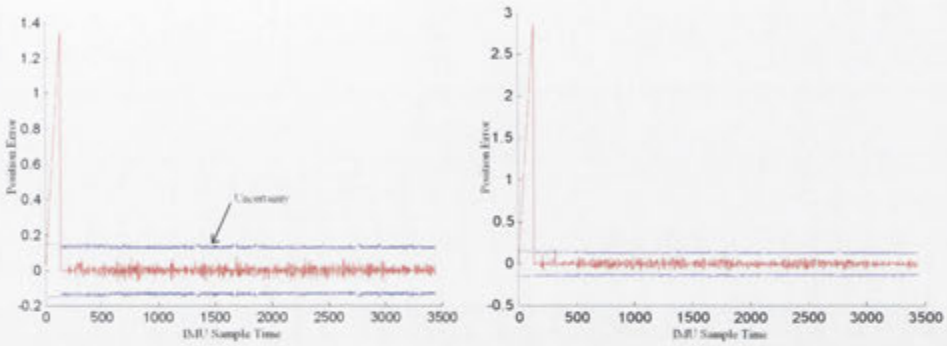
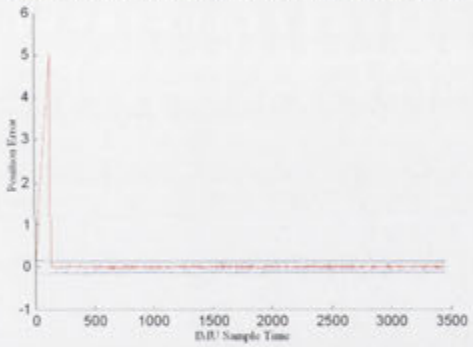



Figure 6.18: Normalised innovation from the EKF predicted pose and measured pose (innovation vectors are normalized by their standard deviation)

were used for both real-time demonstrations. Secondly, the real-time results of the Visual-Inertial navigation system for indoor environment were presented, followed by real-time hovering control. Finally, the results of the outdoor navigation system were described. The qualitative and quantitative evaluation of the proposed work against the highly precise VICON system reveals the robustness of the approach in different challenging environments.

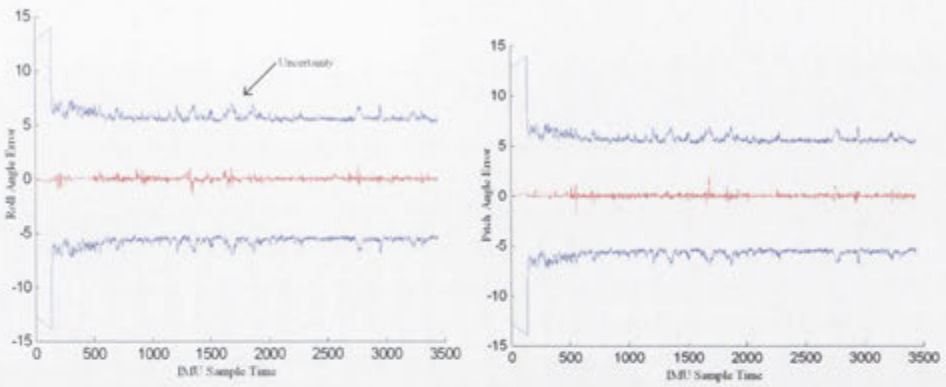



(a) X-position error with estimated covariance (b) Y-position error with estimated covariance

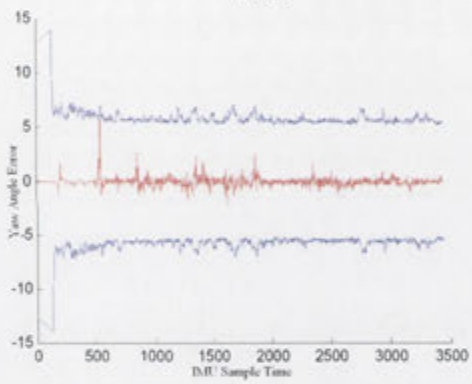



(c) Z-position error with estimated covariance

Figure 6.19: Estimated covariance (3 sigma deviation) and position error from the proposed approach.




(a) Roll angle error with estimated covariance (b) Pitch angle error with estimated covariance




(c) Yaw angle error with estimated covariance

Figure 6.20: Estimated covariance (3 sigma deviation) and attitude error from the proposed approach.

# Chapter 7

# Further Analysis on the Number of Local Minima

## 7.1 Introduction

In the previous chapter, we have presented the results of real-time Visual-Inertial SLAM and autonomous hovering of the aerial vehicle. In this chapter, our aim is to investigate and form a theoretical foundation for the SLAM problem by analysing its structure (number of local minima) and methodology to solve the problem optimally. As SLAM is a highly non-linear problem, so the understanding of the convex structure of the system is of much interest to robotics community. In particular, the study of the number of local minima can help to provide a guaranteed global minimum solution to highly non-convex problems.

Firstly, we discussed the conducted analysis on the SLAM problem which helped to understand the highly non-convex nature. The question of how many or total number of the local minima in the 3D SLAM problem is worth investigating, as until today, no such analysis exists in the current SLAM literature, to the best of our knowledge. We have investigated the number of local minima in SLAM problem (that is three positions and two orientations whereas heading was assumed to be known) using

stationary point analysis. We have provided a proof for the upper bound on the number of local minima and shown that the problem is equivalent to solving a two variable problem.

Secondly, we provided the theory behind a practical approach for finding the globally optimal solution in the 2D SLAM problem (two positions and one orientation unknown). The theoretical limit on the number of local minima provided earlier [148, 149] was exploited to solve the SLAM problem optimally. The proposed approach is not dependent upon the accuracy of the initial guess whereas existing approaches in SLAM literature [56, 101, 150] require a good starting point for convergence to the basin of global minima.

## 7.2 Analysis of Local Minima in 5DOF SLAM

SLAM literature is mostly categorised into two main streams: Filtering based [119] and graph based [151] approaches. As in our developed SLAM framework, the filtering based approach is backed by an optimisation based approach (Chapter 4) as:

- Front-end: Real-time pose estimation using Visual-Inertial information in probabilistic framework (EKF).
- Back-end: Non-linear least square based optimisation (graph based) approach for global consistency.

In this work, we have further investigated the non-linear based optimisation approach to understand the hidden structure of SLAM. In optimisation based SLAM approaches, measurements acquired during robot motions are modelled as constraints. The goal of these approaches is to estimate the configuration of parameters that maximally explain a set of measurements affected by Gaussian noise (minimises the non-linear least square error). The pioneering work in graph based SLAM is by [151] in which a brute force technique for range scan alignment was proposed. With the

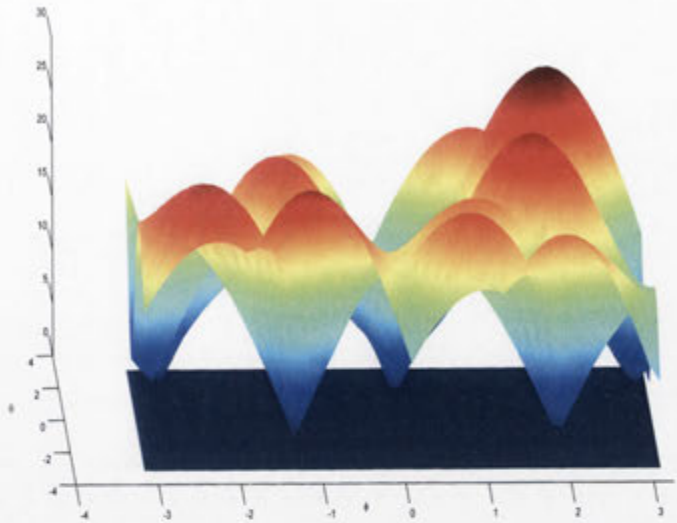


Figure 7.1: A simulation case for 5DOF SLAM problem with five stationary points.

assumption of known rotation [152], a Gauss-Seidel relaxation was introduced. The work was improved by [153] solving a network at different levels of resolution.

The work of [56] and tree-structure [55] based SGD has shown surprising results, in which the algorithm shows convergence with poor initial estimates. These convergence results indicate a lesser number of local minima as expected for high dimensional problems. Recently the work of [149, 154] has looked into this surprising result and provided a convexity analysis for the 2D SLAM problem. There is another work by [155], who has looked into the closed-form solution for the pose graph based 2D SLAM problem. Our work is different due to the highly non-linear nature of the 3D SLAM problem [17], involving multiple trigonometric terms so that the analytical evaluation of the roots and associated conditions is a not a straightforward extension.

Various work in computer vision [156, 157] regarding local minima have looked into resection and triangulation problems separately but simultaneous estimation is not addressed until today. Separately solving the resection and triangulation steps can lead to a suboptimal solution [158], whereas we are interested in the joint estimation of the problem with odometry and feature constraints.

We have formulated the SLAM as a non-linear least square problem where one orientation angle is assumed to be known in 6DOF case[1]. We have considered two poses at one time and solved the first and second order differential equations for the objective function. The rigorous manipulation of the objective function and trigonometric identities is performed to obtain a simplified form, providing insight into the hidden structure to be solved for optimisation. In this work, we have considered a 3D SLAM problem with range and bearing sensors and also provide a proof on the maximum number of local minima. Our analysis also shows that the 5DOF optimisation problem is equivalent to solving a two variable problem. Figure 7.1, shows a simulation case for a 5DOF SLAM problem with five stationary points (See simulation Section 7.2.4 for details).

### 7.2.1 Non-linear Least Square Feature based SLAM

The non-linear least square formulation of 3D SLAM requires control inputs and feature observations to estimate all the vehicle poses and feature locations. Control inputs provide odometry information depicting the relative pose between two consecutive vehicle poses whereas feature observations provide information regarding the relative position of the observed feature w.r.t to vehicle pose. The SLAM problem in the optimisation based approach [2] is to minimise an objective function as follows:

$$\arg\min_{x} \sum (||E_o||^2_{P_{o^i}} + ||E_f||^2_{P_{f^i_j}}), \tag{7.1}$$

where the state vector is $x = \{X, M\}$ with $X$ being the vehicle poses and $M$ the 3D features positions in absolute coordinate frame. $P_{o^i}$ and $P_{f^i_j}$ are odometry and feature measurement errors respectively. The $p$ vehicle poses are defined as

$$X = \left\{x_r^0, y_r^0, z_r^0, \phi^0, \theta^0, \Psi^0 \ldots, x_r^p, y_r^p, z_r^p, \phi^p, \theta^p, \Psi^p\right\} \tag{7.2}$$

[1]The assumption of knowing one orientation (heading) is not an ideal assumption [152], considering the availability of heading sensor i.e. magnetometer.

[2]Please note that the least square solution to SLAM is not necessarily equivalent to an application of Bayes theorem [152]

The $n$ map features as $M = \{x_1, y_1, z_1 \ldots, x_n, y_n, z_n\}$. The odometry observations describing the relative pose information at pose $i$ is defined as

$$O^i = \left\{\Delta x_r^i, \Delta y_r^i, \Delta z_r^i, \Delta\phi^i, \Delta\theta^i, \Delta\Psi^i\right\} \tag{7.3}$$

The relative feature observation of feature $j$ at pose $i$ is defined as

$$f_j^i = \left\{\Delta x_j^i, \Delta y_j^i, \Delta z_j^i\right\} \tag{7.4}$$

The odometry error is defined as an error between the odometry measurements and observation model as

$$E_o = O^i - H_{o^i}(X), \tag{7.5}$$

and the non-linear observation model for odometry information is defined as

$$H_{o^i}(X) = \begin{bmatrix} \left(R^T_{\phi^{i-1},\theta^{i-1},\Psi^{i-1}} \begin{pmatrix} x_r^i - x_r^{i-1} \\ y_r^i - y_r^{i-1} \\ z_r^i - z_r^{i-1} \end{pmatrix}\right) \\ \left(R^T_{\phi^{i-1},\theta^{i-1},\psi^{i-1}} \begin{pmatrix} \phi^i - \phi^{i-1} \\ \theta^i - \theta^{i-1} \\ \psi^i - \psi^{i-1} \end{pmatrix}\right) \end{bmatrix} \tag{7.6}$$

The $SO(3)$ rotation matrix is defined as $R(\phi, \theta, \psi)$

$$R_{\phi,\theta,\psi} = \begin{bmatrix} C_\theta C_\psi & -C_\phi S_\psi + S_\phi S_\theta C_\psi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ C_\theta S_\psi & C_\phi C_\psi + S_\phi S_\theta S_\psi & -S_\phi C_\psi + C_\phi S_\theta S_\psi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \tag{7.7}$$

where $c_{(\cdot)}$ and $s_{(\cdot)}$ represent $\sin(\cdot)$ and $\cos(\cdot)$ respectively.

The feature observation error $E_f$ is defined as

$$E_{f_j^i} = f_j^i - H_{f_j^i}(X, M), \tag{7.8}$$

whereas the non-linear feature observation model, describing relative distance between observed feature and vehicle pose, is a function of $M$ and $X$ as

$$H_{f_j^i}(X, M) = \begin{bmatrix} R^T_{\phi^i, \theta^i, \psi^i} \begin{pmatrix} x_j - x_r^i \\ y_j - y_r^i \\ z_j - z_r^i \end{pmatrix} \end{bmatrix}. \tag{7.9}$$

The Mahalanobis distance for both relative errors in the objective function with covariance $P_{o^i}, P_{f_j^i}$ having zero-mean gaussian noise can be written as

$$J(x) = \sum_{i=1}^{p} (O_i - H_{o_i}(X))^T P_{o^i}^{-1} (O_i - H_{o_i}(X)) + \sum_{ij} (f_j^i - H_{f_j^i}(X, M))^T P_{f_j^i}^{-1} (f_j^i - H_{f_j^i}(X, M)) \tag{7.10}$$

### 7.2.2 One-Step 5DOF SLAM Problem

We have considered a one-step SLAM problem, a special case of Equation 7.10 for the purpose of analysis, with two poses and $n$ features. Figure 7.2 shows the representation of one-step SLAM with odometry and feature observations. The first pose is considered to be a world coordinate, with respect to which the second pose and other features are defined. The spherical form of the covariance matrix $P_{o^i}, P_{f_j^i}$ is used in this analysis to represent the simplified form of the non-linear cost function $J(x)$ as

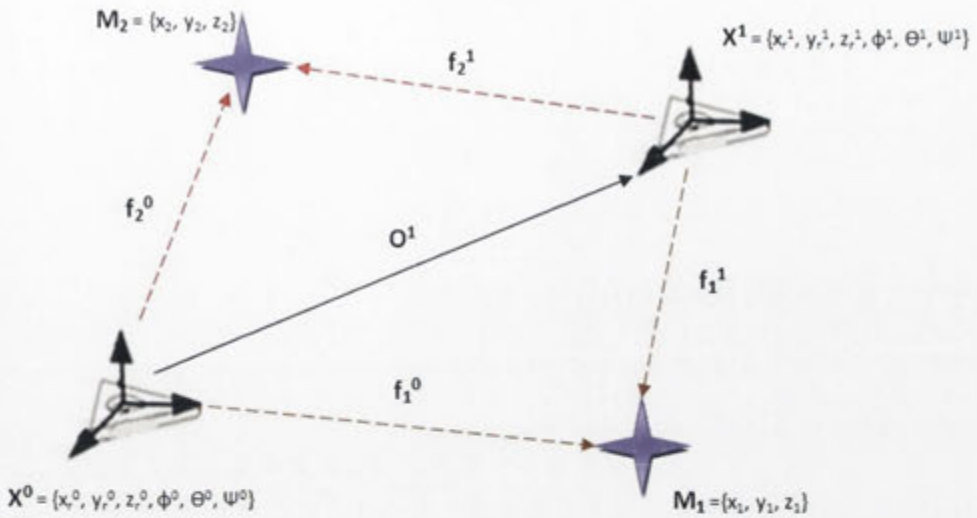



Figure 7.2: One-Step SLAM representation with two poses and features.

$$J(x) = (O^1 - H_{o^1}(X))^2 + \sum_{i=0}^{1}\sum_{j=1}^{n}(f_j^i - H_{f_j^i}(X, M))^2 \tag{7.11}$$

By substituting the values of odometry and feature observations, we can see the non-linearity is coming from $R_{\phi^1,\theta^1,\psi^1}$, when features are observed at second pose whereas observations at first pose are linear

$$J(x) = \alpha + \beta + \sum_{j=1}^{n}\left(\begin{pmatrix}\Delta x_j^1\\ \Delta y_j^1\\ \Delta z_j^1\end{pmatrix} - R^T_{\phi^1,\theta^1,\psi^1}\begin{pmatrix}x_j - x_r\\ y_j - y_r\\ z_j - z_r\end{pmatrix}\right)^2 \tag{7.12}$$

where $\alpha$ and $\beta$ are the first and second terms of objective function

$$\alpha = \begin{pmatrix} \Delta x_r - x_r \\ \Delta y_r - y_r \\ \Delta z_r - z_r \\ \Delta\phi - \phi \\ \Delta\theta - \theta \\ \Delta\psi - \psi \end{pmatrix}^2, \qquad \beta = \sum_j^n \begin{pmatrix} \Delta x_j^0 - x_j \\ \Delta y_j^0 - y_j \\ \Delta z_j^0 - z_j \end{pmatrix}^2$$

The benefit of using spherical matrix is not only for simplicity but the error in the world coordinate or robot coordinate has a similar meaning, so we can estimate the error in either coordinate. In this analysis, by calculating the feature observation error in the world coordinate, we can avoid the product of the state variables with rotation matrix whereas the non-linearity left in the objective function is due to the rotation matrix

$$J(x) = \alpha + \beta + \sum_j^n \left( R_{\phi^1,\theta^1,\psi^1} \begin{pmatrix} \Delta x_j^1 \\ \Delta y_j^1 \\ \Delta z_j^1 \end{pmatrix} - \begin{pmatrix} x_j - x_r \\ y_j - y_r \\ z_j - z_r \end{pmatrix} \right)^2 \tag{7.13}$$

We relax the original 6DOF SLAM by assuming the heading angle to be zero, therefore the objective function becomes

$$J(x) = \alpha' + \beta + \sum_j^n \left( \begin{pmatrix} U_j \\ V_j \\ W_j \end{pmatrix} - \begin{pmatrix} x_j - x_r \\ y_j - y_r \\ z_j - z_r \end{pmatrix} \right)^2 \tag{7.14}$$

where $\alpha'$ is without the $\psi$-related term

$$\begin{aligned} U_j &= \Delta x_j^1 c_\theta + \Delta y_j^1 s_\phi s_\theta + \Delta z_j^1 c_\phi s_\theta \\ V_j &= \Delta y_j^1 c_\phi - \Delta z_j^1 s_\phi \\ W_j &= -\Delta x_j^1 s_\theta + \Delta y_j^1 s_\phi c_\theta + \Delta z_j^1 c_\phi c_\theta \end{aligned}$$

Note that the following equalities hold for partial derivatives of $U_j, W_j$

$$\begin{aligned} \frac{\partial U_j}{\partial \theta} &= W_j \\ \frac{\partial W_j}{\partial \theta} &= -U_j \end{aligned}$$

The gradient/hessian are evaluated for the objective function $J(x)$ to conduct stationary point analysis.

The gradient of the cost function $J(x)$ in Eq. 7.11 is estimated w.r.t to $x$ as

$$\frac{\partial J(x)}{\partial x} =$$

$$2\begin{bmatrix} -\Delta x_1^0 + x_1 - (U_1 - (x_1 - x_r)) \\ -\Delta y_1^0 + y_1 - (V_1 - (y_1 - y_r)) \\ -\Delta z_1^0 + z_1 - (W_1 - (z_1 - z_r)) \\ \vdots \\ -\Delta x_n^0 + x_n - (U_n - (x_n - x_r)) \\ -\Delta y_n^0 + y_n - (V_n - (y_n - y_r)) \\ -\Delta z_n^0 + z_n - (W_n - (z_n - z_r)) \\ -\Delta x_r + x_r + \sum_{j=1}^{n}(U_j - (x_j - x_r)) \\ -\Delta y_r + y_r + \sum_{j=1}^{n}(V_j - (y_j - y_r)) \\ -\Delta z_r + z_r + \sum_{j=1}^{n}(W_j - (z_j - z_r)) \\ -\Delta\phi + \phi + \sum_{j=1}^{n}\Big((U_j - (x_j - x_r)W_j) + (W_j - (z_j - z_r)(-U_j))\Big) \\ \begin{pmatrix} -\Delta\theta + \theta + \sum_{j=1}^{n}(\ (U_j - (x_j - x_r)s_\theta V_j) + (V_j - \\ (y_j - y_r)(-\Delta y_j^1 s_\theta - \Delta z_j^1 c_\theta)) + (W_j - (z_j - z_r)c_\theta(V_j))) \end{pmatrix} \end{bmatrix}$$

The first gradient $\frac{\partial J(x)}{\partial x} = 0$ reveals all the stationary points. As it is seen that 3n+3 equations are linear when $\phi, \theta$ are fixed. This shows that the actual 5DOF problem is in fact equivalent to solving a two variable optimisation problem. By solving the first $3n + 3$ equations simultaneously, we obtain

$$
\begin{aligned}
x_j - x_r = -\frac{1}{n+2}\Delta x_r + \frac{1}{2}\Delta x_j - \frac{1}{2(n+2)}\sum_{k=1}^{n}\Delta x_k + \frac{1}{2}U_j \\
+\frac{1}{2(n+2)}\sum_{k=1}^{n}U_k \\
y_j - y_r = -\frac{1}{n+2}\Delta y_r + \frac{1}{2}\Delta y_j - \frac{1}{2(n+2)}\sum_{k=1}^{n}\Delta y_k + \frac{1}{2}V_j \\
+\frac{1}{2(n+2)}\sum_{k=1}^{n}V_k \\
z_j - z_r = -\frac{1}{n+2}\Delta z_r + \frac{1}{2}\Delta z_j - \frac{1}{2(n+2)}\sum_{k=1}^{n}\Delta z_k + \frac{1}{2}W_j \\
+\frac{1}{2(n+2)}\sum_{k=1}^{n}W_k
\end{aligned} \tag{7.15}
$$

By substituting Equation 7.15 into the last two equations of partial derivatives w.r.t $(\phi, \theta)$ and considering $n = 1$ (as the actual 5DOF problem is about solving for two optimisation variables $(\phi, \theta)$ so this will yield same result as if $n > 1$), we have

$$
\begin{aligned}
\frac{\partial J(x)}{\partial \phi} = \phi - \Delta\phi + \Delta y_1^0 \Delta z_1^1 c_\phi + \Delta y_r \Delta z_1^1 c\phi + \Delta y_1^0 \Delta y_1^1 s\phi + \Delta y_r \Delta y_1^1 s_\phi \\
-\Delta y_1^1 \Delta z_1^0 c_\theta c_\phi - \Delta y_1^1 \Delta z_r c_\theta c_\phi - \Delta x_1^0 \Delta y_1^1 c_\phi s_\theta - \Delta x_r \Delta y_1^1 c_\phi s_\theta \\
+\Delta z_1^0 \Delta z_1^1 c_\theta s_\phi + \Delta z_r \Delta z_1^1 c_\theta s_\phi + \Delta x_1^0 \Delta z_1^1 s_\theta s_\phi + \Delta x_r \Delta z_1^1 s_\theta s_\phi
\end{aligned} \tag{7.16}
$$

$$
\begin{aligned}
\frac{\partial J(x)}{\partial \theta} = \theta - \Delta\theta + \Delta x_1^1 \Delta z_1^0 c_\theta + \Delta x_1^1 \Delta z_r c_\theta + \Delta x_1^0 \Delta x_1^1 s_\theta + \Delta x_r \Delta x_1^1 s_\theta \\
-\Delta x_1^0 \Delta z_1^1 c_\theta c_\phi - \Delta x_r \Delta z_1^1 c_\theta c_\phi - \Delta x_1^0 \Delta y_1^1 c_\theta s_\phi - \Delta x_r \Delta y_1^1 c_\theta s_\phi \\
+\Delta z_1^0 \Delta z_1^1 c_\phi s\theta + \Delta z_r \Delta z_1^1 c_\phi s_\theta + \Delta y_1^1 \Delta z_1^0 s_\theta s_\phi + \Delta y_1^1 \Delta z_r s_\theta s_\phi
\end{aligned} \tag{7.17}
$$

By collecting the odometry and feature observation terms as constant, we obtain

$$\frac{\partial J(x)}{\partial \phi} = \phi - \Delta\phi + q_0 c_\phi + q_1 c_\phi + q_2 s_\phi + q_3 s_\phi + q_4 c_\theta c_\phi + q_5 c_\theta c_\phi + q_6 c_\phi s_\theta + q_7 c_\phi s_\theta + q_8 c_\theta s_\phi + q_9 c_\theta s_\phi + q_{10} s_\theta s_\phi + q_{11} s_\theta s_\phi \tag{7.18}$$

$$\frac{\partial J(x)}{\partial \theta} = \theta - \Delta\theta + r_0 c_\theta + r_1 c_\theta + r_2 s_\theta + r_3 s_\theta + r_4 c_\theta c_\phi + r_5 c_\theta c_\phi + r_6 c_\theta s_\phi + r_7 c_\theta s_\phi + r_8 c_\phi s\theta + r_9 c_\phi s_\theta + r_{10} s_\theta s_\phi + r_{11} s_\theta s_\phi \tag{7.19}$$

Where q and r are constants used instead of given odometry/feature observations. Furthermore by combining the similar terms, we have

$$\frac{\partial J(x)}{\partial \phi} = (\phi - \Delta\phi) + q_1' c_\phi + q_2' s_\phi + q_3' c_\theta c\phi + q_4' s_\theta s\phi + q_5' c_\phi s_\theta + q_6' c_\theta s_\phi \tag{7.20}$$

$$\frac{\partial J(x)}{\partial \theta} = (\theta - \Delta\theta) + r_1' c_\theta + q_2' s_\theta + q_3' c_\theta c_\phi + q_4' s_\theta s_\phi + q_5' c_\theta s_\phi + q_6' c_\phi s_\theta \tag{7.21}$$

Again q' and r' are constants after simplification. Using trigonometry identities, we obtain further simplification as

$$\frac{\partial J(x)}{\partial \phi} = a_0\phi + a_1 \sin(\phi - \theta + b_1) + a_2 \sin(\phi + b_2) + a_3 \sin(\phi + \theta + b_3) \tag{7.22}$$

$$\frac{\partial J(x)}{\partial \theta} = c_0\theta + c_1 \sin(\phi - \theta + d_1) + c_2 \sin(\theta + d_2) + c_3 \sin(\phi + \theta + d_3) \tag{7.23}$$

The first gradient is defined as a function of two variables $(\phi, \theta)$ as

$$f_1(\phi,\theta) = \frac{\partial J(x)}{\partial \phi}, \quad f_2(\phi,\theta) = \frac{\partial J(x)}{\partial \theta}$$

The hessian of the objective function is also obtained similarly to first gradient derivation as

$$\begin{aligned}\frac{\partial f_1(\phi,\theta)}{\partial \phi} &= 1 + a_4\cos(\phi - \theta + b_4) + a_5\cos(\phi + b_5) \\ &\quad + a_6\cos(\phi + \theta + b_6) \qquad (7.24)\\ \frac{\partial f_2(\phi,\theta)}{\partial \theta} &= 1 + c_4\cos(\phi - \theta + d_4) + c_5\cos(\theta + d_5) \\ &\quad + c_6\cos(\phi + \theta + d_6) \qquad (7.25)\end{aligned}$$

In this section, we have provided a derivation of gradient and hessian estimation and also observed that solving $(\frac{\partial J(x)}{\partial x} = 0)$ for 5DOF problem is same as solving an optimisation problem with two variables $(\phi, \theta)$.

### 7.2.3 Stationary Point Analysis

For a Jacobian function $f(\phi,\theta) = [f_1(\phi,\theta), f_2(\phi,\theta)]^T$

$$f(\phi,\theta) = \begin{bmatrix} a_0\phi + \sum_{i=1}^{3} a_i \sin\left(\phi + (i-2)\theta + b_i\right) \\ c_0\theta + \sum_{i=1}^{3} c_i \sin\left((i-2)\phi + \theta + d_i\right) \end{bmatrix}, \qquad (7.26)$$

evaluate the number of roots $(\phi,\theta)$ with Hessian matrix being positive definite

$$\begin{bmatrix} f_1(\phi,\theta) = 0 \\ f_2(\phi,\theta) = 0 \end{bmatrix} \quad with \quad \begin{bmatrix} \frac{\partial f_1}{\partial \phi} & \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial \phi} & \frac{\partial f_2}{\partial \theta} \end{bmatrix} > 0. \qquad (7.27)$$

As the functions are non-linear with multiple trigonometric terms, an analytical eval-

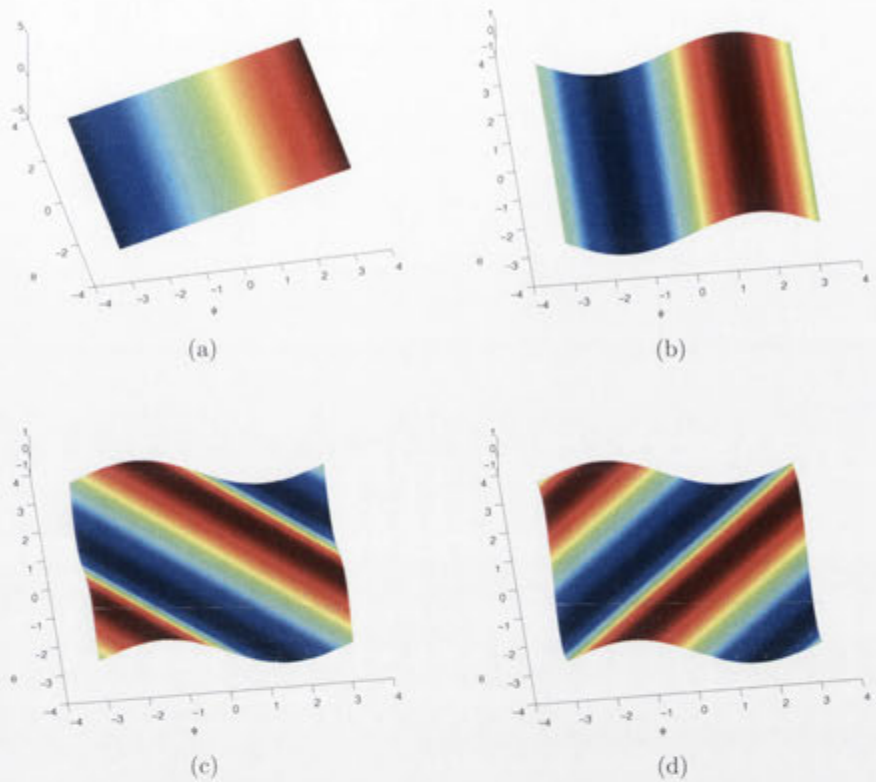
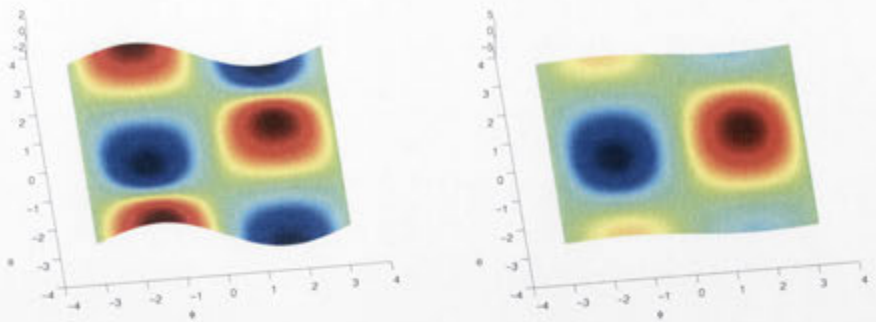


Figure 7.3: Four wave components of $f_1(\phi, \theta)$ consisting of a) $\phi$, b) $\sin(\phi)$, c) $\sin(\phi+\theta)$ and d) $\sin(\phi - \theta)$. Note the last two sinusoids have $\pm 45^\circ$ wave directions which are orthogonal to each other.

uation of the roots becomes intractable. Thus we will rather focus on the maximum number of roots for these non-linear simultaneous equations. The functions consist of a linear term and a sum of 2D sinusoidal surfaces with different magnitudes and phase offsets.

Figure 7.3 illustrates the four components of $f_1(\phi, \theta)$ which are a linear plane $\phi$, $\sin(\phi)$, $\sin(\phi + \theta)$ and $\sin(\phi - \theta)$. Figure 7.4(a) shows a superposition result for the two orthogonal sinusoids $sin(\phi + \theta)$ and $sin(\phi - \theta)$. The orthogonal nature of two sinusoids create a grid pattern with two alternating peaks and troughs within the boundary. The addition of $sin(\phi)$ reinforces the peaks and troughs along the



(a) (b)

Figure 7.4: Superposition for $f_1(\phi, \theta)$: a) two orthogonal sinusoids ($\sin(\phi + \theta)$ and $\sin(\phi - \theta)$) and b) three sinusoids including $\sin(\phi)$. The orthogonal nature of two sinusoid creates a grid-like pattern with two alternating peaks and troughs within the boundary. The addition of $\sin(\phi)$ reinforces the peaks and troughs along the $\phi$-direction.

$\phi$-direction as shown in Figure 7.4(b) while still maintaining the grid structure.

Figure 7.5 shows the four components of $f_2(\phi, \theta)$ which are similar to $f_1(\phi, \theta)$ except a linear plane $\theta$ and $\sin(\theta)$. Thus it has the same grid pattern and the addition of $\sin(\theta)$ simply reinforces the peaks and troughs along the $\theta$-direction as shown in Figure 7.5(c).

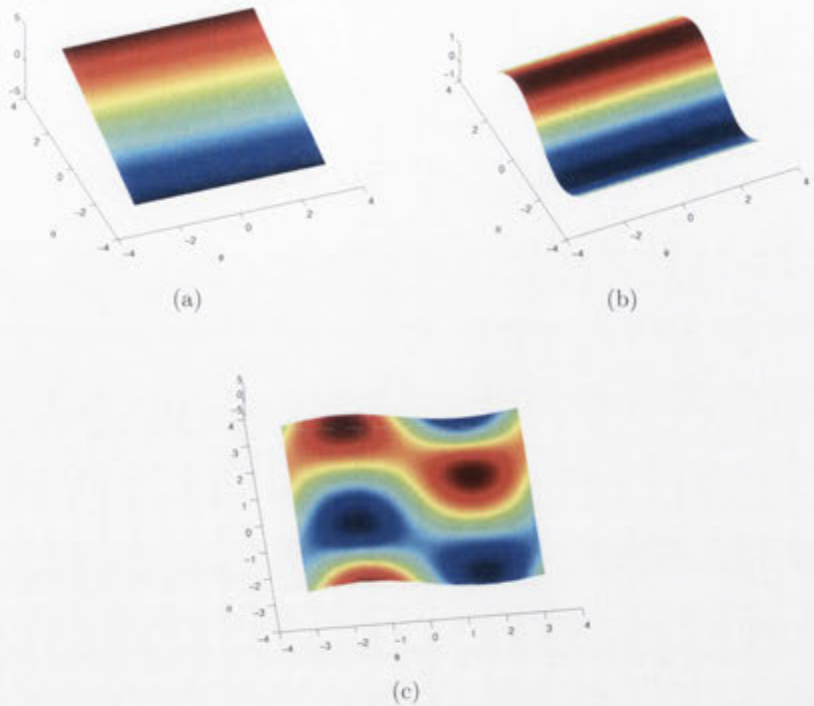
#### 7.2.3.1 Maximum number of roots

The sinusoidal terms can be further simplified as

$$\begin{bmatrix} f_1(\phi, \theta) \\ f_2(\phi, \theta) \end{bmatrix} = \begin{bmatrix} a_0\phi + A(\theta)\sin(\phi + B(\theta)) \\ c_0\theta + C(\phi)\sin(\theta + D(\phi)) \end{bmatrix} \tag{7.28}$$

where $(A, C)$ are magnitude functions with $(B, D)$ being the phase functions. Please note that the resulting wave has the same wavelength to the original components but with a varying amplitude and phase offset.

This can be shown by using the complex signal representation of the sinusoids, the



(a) (b)

(c)

Figure 7.5: Additional wave components of $f_2(\phi, \theta)$: a) $\theta$ and b) $\sin(\theta)$ ($\sin(\phi+\theta)$ and $\sin(\phi-\theta)$ are not shown here). Superposition result with three sinusoids including $\sin(\theta)$. The grid pattern is preserved but the peak and trough along the $\theta$-direction were reinforced.

magnitudes and phases of the superposed signals can be computed

$$A(\theta)e^{jB(\theta)} = a_1 e^{j(-\theta+b_1)} + a_2 e^{j(b_2)} + a_3 e^{j(\theta+b_3)}$$
$$C(\phi)e^{jD(\phi)} = c_1 e^{j(-\phi+d_1)} + c_2 e^{j(d_2)} + c_3 e^{j(\phi+d_3)}$$

$$A(\theta) = \sqrt{(a_1 c_{-\theta+b_1})^2 + (a_2 c_{b_2})^2 + (a_3 c_{\theta+b_3})^2} \tag{7.29}$$

$$B(\theta) = tan^{-1} \frac{a_1 s_{-\theta+b_1} + a_2 s_{b_2} + a_3 s_{\theta+b_3}}{a_1 c_{-\theta+b_1} + a_2 c_{b_2} + a_3 c_{\theta+b_3}} \tag{7.30}$$

$$C(\phi) = \sqrt{(c_1 c_{-\phi+d_1})^2 + (c_2 c_{c_2})^2 + (c_3 c_{\phi+d_3})^2}, \tag{7.31}$$

$$D(\phi) = tan^{-1} \frac{c_1 s_{-\phi+d_1} + c_2 s_{d_2} + c_3 s_{\phi+d_3}}{c_1 c_{-\phi+d_1} + c_2 c_{d_2} + c_3 c_{\phi+d_3}} \tag{7.32}$$

**Lemma 1**: The solution curves for $f_1(\phi, \theta) = 0$ consist of at most three open and/or close branches.

*proof*: The solution curves are the intersections between a sinusoidal surface and a plane $z = -a_0\phi$ with varying slope $a_0$

$$A(\theta) \sin(\phi + B(\theta)) = -a_0\phi. \tag{7.33}$$

The surface is a superposition of three sinusoidal waves, so the analysis of the intersection with the plane is not so straightforward. However, as all sinusoids have the same wavelength of $2\pi$, the resulting wave also has the same wavelength but with a varying amplitude and phase as a function of $\theta$. That is, for a fixed $\theta$-value, $\theta_0$, the cross-section of the surface is a simple sinusoid as illustrated in Figure 7.6. Therefore if the slope of the intersecting plane satisfies $-A(\theta_0) < a_0 < A(\theta_0)$, then the maximum number of intersections becomes *three* within $-\pi \leq \phi < \pi$. In addition, the number of intersections with positive gradient are at most *two*. Figure 7.7(a) shows a set of 2D solution curves for $f_1(\phi, \theta) = 0$ as the plane slope varies, with intersection points forming curves of an open branch with a closed loop. It can be seen that any horizontal line intersects the curve at most three points.

**Lemma 2**: The solution curves for $f_2(\phi, \theta) = 0$ also consist of at most three open and/or closed branches.

$$C(\phi) \sin(\theta + D(\phi)) = -c_0\theta. \tag{7.34}$$

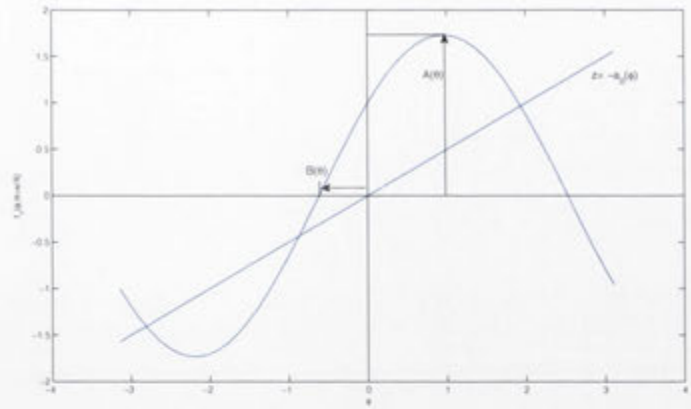


Figure 7.6: A cross-section showing the intersection of the surface and plane.

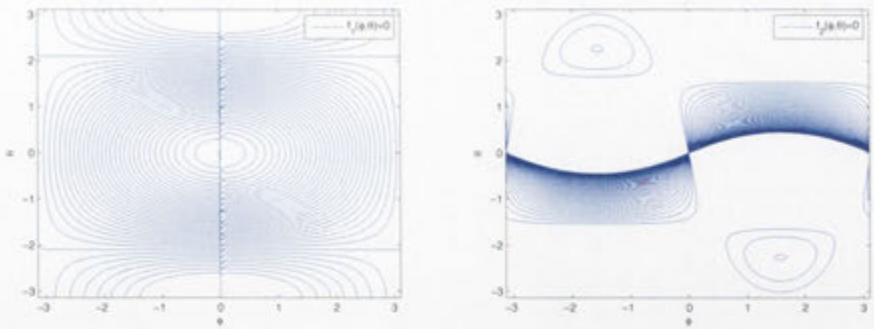
*proof*: As the function has a similar form to $f_1$ along the $\theta$ direction, the same argument holds yielding a maximum number of three intersections for varying slope $c_0$. Solution curves for $f_2(\phi, \theta) = 0$ are also shown in Figure 7.7(b), and again any vertical line meets with the curves at most three points.

**Theorem 1**: The maximum number of roots for the simultaneous equations is *nine*.

$$\begin{aligned} f_1(\phi, \theta) &= 0 \\ f_2(\phi, \theta) &= 0 \end{aligned} \tag{7.35}$$

*proof*: As each solution curve along each axis consists of three open branches or one open branch with one closed loop, the maximum number of intersections between curve (Left-hand-side of Equation 7.33 and 7.34) with plane (Right-hand-size) jointly becomes nine due to orthogonality characteristic of the curves (similar to Figure 7.4(a) for $(\phi, \theta) \in [-\Pi, \Pi]$). This can be easily seen in Figure 7.8(c) where the number of intersection becomes nine (each forming as a convex region). If the slopes of the plane change, the number of root can drop as shown in Figures 7.8(a) and (b).

**Theorem 2**: The maximum number of roots for the simultaneous equations



(a) (b)

Figure 7.7: Solution curve sets as the intersections between the surface and the plane with varying slope for a) $f_1(\phi, \theta) = 0$ and b) $f_2(\phi, \theta) = 0$. The curve set consists open branches or closed loops but the number of crossings with any horizontal line for $f_1(\phi, \theta)$ or any vertical line for $f_2(\phi, \theta)$ is at most three (see the proof in text for details).

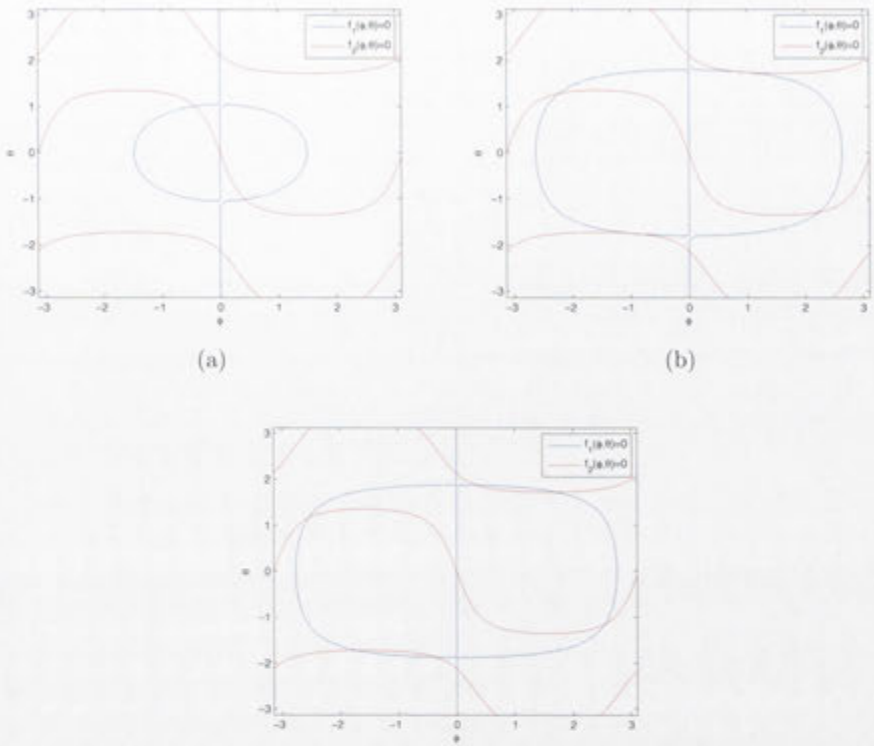
$f_1(\phi, \theta) = 0$ and $f_2(\phi, \theta) = 0$ with the following conditions is *four*.

$$\begin{aligned} \partial f_1(\phi, \theta)/\partial\phi > 0 \\ \partial f_2(\phi, \theta)/\partial\theta > 0 \end{aligned} \tag{7.36}$$

*proof*: This due to the fact that the nine roots can have positive or negative gradients along $\phi$ and $\theta$ directions. However each direction, there exist at most two roots with a positive slope. As this analysis focuses on the upper bound of local minima, the discriminant of the Hessian matrix was not fully evaluated here and left for future work, which can reduce the maximum number of minima (due to any potential saddle points).

### 7.2.4 Simulation Study

The proposed analysis suggests that at-most four local minima can exist, so to evaluate our findings, a simulation study was also conducted. In the simulation two poses with 10 features were considered with their respective odometry and relative feature

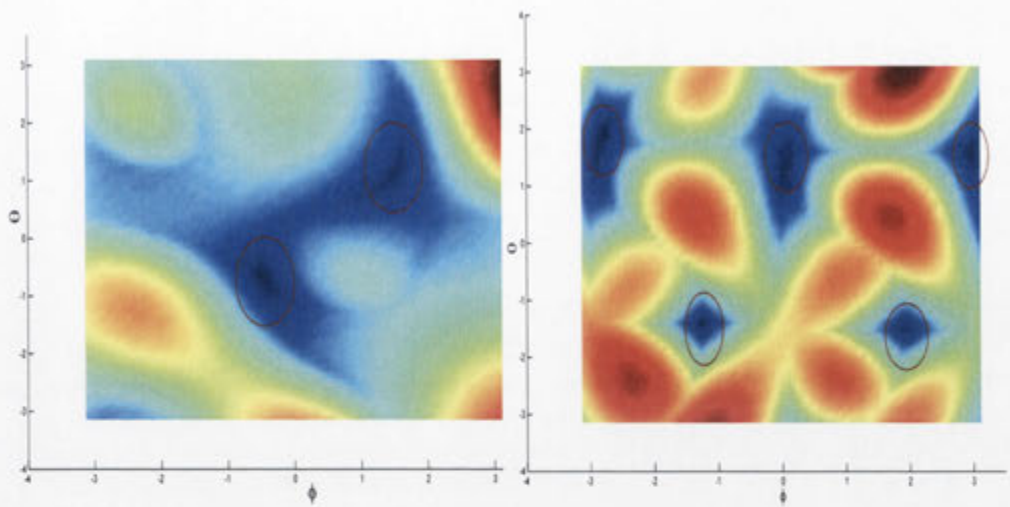


(a) (b)

(c)

Figure 7.8: Examples of solution for two non-linear simultaneous equations with a) five, b) seven and c) nine intersection points.

observations with added white gaussian noise. The simulation was used to generate two dataset with different relative observations assuming heading is fixed (known). For each experimental dataset, Equation 7.27 and 7.28 were used to evaluate the number of stationary points, in which gradient of the objective function w.r.t two variables $(\phi, \theta)$ were calculated. As shown in the Figure 7.9, experimental datasets were evaluated to obtain different number of stationary points. These stationary points are declared local minima when second gradient (Equation 7.27) at their respective $(\phi, \theta)$ is greater than zero. Monte carlo simulation [159] with 1000 trials was conducted with different initial guess (for estimated variables given the observation) revealed at most 4 local solutions. This simulation study helps us to numerically validate the



(a) (b)

Figure 7.9: Number of stationary points (a) two (b) five with 2D projection of $(||f_1(\phi,\theta)|| + ||f_2(\phi,\theta)||)$. Number of local minima after second gradient evaluation becomes (a) One (b) Two.

number of local minima in 5DOF SLAM problem. The Section 7.3 is dedicated to the theory behind finding the optimal solution using the estimated upper bound in this analysis for 2D SLAM problem.

## 7.3 Global Optimal Solution for 2D SLAM Problem

Recent research [101] has focused on making the optimisation based algorithms more efficient and robust, showing that its on-line implementation is feasible. These graph based approaches solve the problem by considering the convex approximation of the original problem and works in iteration to improve the local solution. Although these approaches perform efficiently in practice, very little attention has been paid on the convergence condition and none of them can guarantee a global minimum over different initial guesses. Figure 7.10 shows the results of a Gauss-Newton approach on

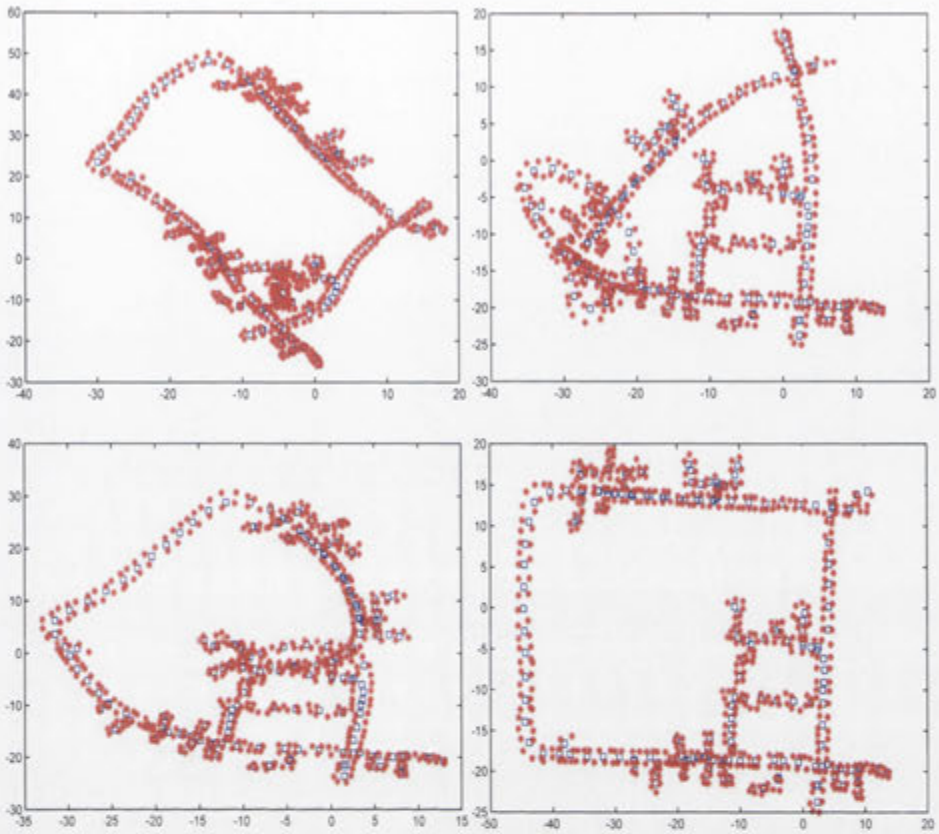


Figure 7.10: Examples of local solution for DLR dataset [160] with different random initial guess.

publicly available dataset [160] with different random initial guesses, resulting in different local solutions. These optimisation based approaches assumes a local convexity, that is with a reasonable initial estimate (near the global basin of attraction), the algorithm will converge to global minima. However when the odometry and feature observation are not consistent with each other, then the local optimiser can struck in local minima.

Global optimal solutions to highly non-linear problems has been shown to NP-hard [161]. In structure from motion research, the guaranteed global optimal solution is investigated with known rotation framework for $L_\infty$ [162] and branch and bound based approaches [157] (typically the non-linearity in SLAM is in the odometry/observation

models due to robot orientation). In recent work [163] proposed a swarm optimisation based approach to estimate (almost) optimal maps. Their work was based upon meta-heuristic optimisation approach where they have presented the map as a tree of fragments with particle filtered based sampling approach. Finally they have conducted an ant colony search to obtain (almost) optimal solution from 2D SLAM problem. However in our work we have exploited the condition on the number of minima in the SLAM problem to obtain the optimal solution.

The dimension of the SLAM problem is very high when it is formulated as a non-linear least square problem [118], because all vehicle poses and feature locations are considered as parameters to be determined as discussed earlier. Here instead of considering the full 3D SLAM problem, which is computationally intractable [3] for optimal solution [163], we have considered a 2D SLAM problem. The equation 7.33 and 7.34 derived earlier for maximum number of roots can be simplified for the 3DOF problem (for two translation and one orientation unknown components):

$$A_1 \sin(\psi + B_1) = -a_1 \psi. \tag{7.37}$$

Where $A_1$ is magnitude with $B_1$ being the phase constant and $a_1$ being an orientation constant (estimated using equations 7.29 to 7.32). This simplification came by fixing the unnecessary unknown terms (i.e. $z_r, \phi, \theta$) for 3DOF case (for two translation and one orientation (heading) unknowns, our earlier work [164]). It is important to note that the resulting function is similar to the earlier work of [149]. The research findings in their work suggest at most two and at least one local minima for 2D SLAM problem. The approach, proposed in this work, exploits the upper theoretical bound (at most two minima) under noisy observations to obtain global minima by a meta-heuristic approach (discussed in following section).

Feature based 2D SLAM problem is considered in this work with two poses or relative maps (one-step SLAM [148]). For the one-step SLAM problem, we considered two poses as two individual relative maps. The assumption we undertake in this research

[3] due to SO(3) group involving multiple trigonometry rotation terms.

is that, every SLAM problem can be decomposed into small relative maps. The relative relation of each map has a fluid behaviour whereas the internal structure of each map is well known and can be optimised independently with respect to local coordinate [118] [4]. Necessary and sufficient condition for the existence of at most two local minima [149] is exploited by a meta-heuristic approach called GRASP (greedy randomised adaptive search procedure) [165] which is combinatorial optimisation to obtain global optimal solution. Meta-heuristic approaches optimise by iteratively refining the candidate solution by combining randomness with local search methods [166]. The global optimality of the SLAM solution is with respect to the cost function 7.1 (with an assumption of spherical covariance).

### 7.3.1 Greedy random adaptive search procedure

GRASP is a multi-start meta-heuristic approach to solve combinatorial problems [165]. Previously, it has been successfully deployed in travelling salesman problem [167] and firstly proposed here for SLAM problem. GRASP basically consist of two phases: local search and feasible solution construction. The construction phase builds a feasible solutions (using greedy approach), whose neighbourhood is searched by a local search phase to find local optima. By using different feasible solutions as starting points the local search will usually lead to good, though most often, suboptimal solutions. Whereas in our case the upper bound is searched by multi-start meta-heuristic approach until a optimal solution is achieved, which is the global optimal solution. The pseudo code in algorithm 1, details the working of GRASP approach, where local search is performed by the gauss-newton [101].

### 7.3.2 Randomised greedy algorithm

We proposed long-term memory based greedy algorithm to determine a feasible solution. Figure 7.11 (left) details a simple example on 1D in which at most two local

[4]The decomposition into relative maps provides a computational benefit as not all the poses within the relative map need to be estimated.

**Algorithm 2** GRASP-Algorithm: Determination of Optimal Solution

*inputs* : $observations, x = \{X, M\}$
$x^* = \{\}$
**while** check local minima condition or max iteration are not reached **do**
  $x \leftarrow RandomizedGreedyAlgo(.) \rightarrow Algo2$
  $x \leftarrow LocalSearch(x, observations)$
  **if** (f(x) < f(x*)) **then**
    $x^* = x$
  **end if**
**end while**
**return** $x^*$

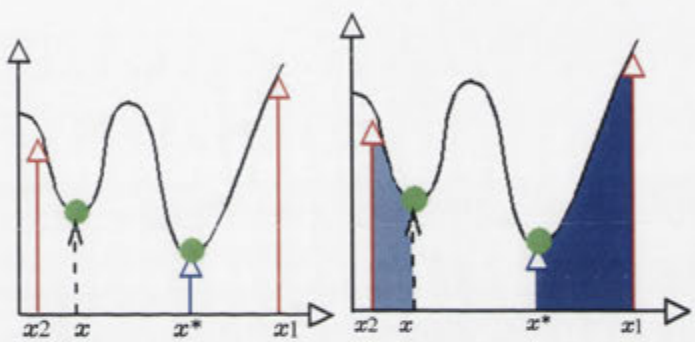



Figure 7.11: 1D problem with 2 local minima revealing possible feasible solutions from local optimisation

minima are considered. The two feasible solutions $(x_1, x_2)$ are generated by GRASP approach and among them, the two minima are found (whereas $x_1$ reveals the global optimal solution $x^*$). The generation of feasible solution is the key to obtain the optimal solution, in a high dimensional search space of parameters. Figure 7.11 (right) describes a search space reduction mechanism in which the initial guess $x_1$ is first hypothesised, which determines a local optimal $x^*$ (the intermediate traversal solution of the local optimiser are stored in long-term memory for search space $x_1$ to $x^*$). The next hypothesis of initial guess is being made by greedy algorithm in consideration with the already traversed solution space in long-term memory (the goal is to avoid making an initial guess from already traversed space). The selection of new initial guess is based upon absolute distance criteria (greedy approach), in which the new initial guess for local optimiser is not from the already considered search space. The

greedy algorithm helps to reduce the search space and improves the time efficiency as against the exhaustive brute force search in solution space. The pseudo-code in algorithm 2, describes the greedy algorithm, in which the long-term memory of all the solution space is maintained. The selection of new feasible solution is determined similar to 1D explained approach on orientation space (as vehicle and feature positions are linear with respect to orientation). Selection of $\epsilon$ in radians decides the selection of new hypothesis for initial guess to boot the local optimiser.

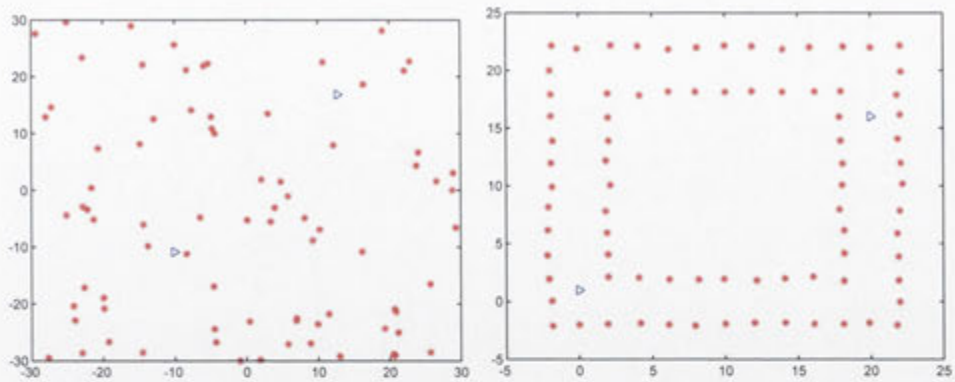
**Algorithm 3** GRASP-Algorithm: Randomised greedy algorithm

```
while Initial guess not found | Max iter not reached do
    x = random(.)
    found = 1
    for i=1:num of elements in LTmemoryX do
        Oldx = LTmemoryX[i]
        if |(x − Oldx)| < ε then
            found = 0
            Break → select another x
        end if
    end for
end while
return x
```

An appealing characteristic of GRASP approach is the ease of implementation by setting and tuning few parameters. The computation time of the approach does not vary much from iteration to iteration and increases linearly with the number of iterations whereas the time increases combinatorially with the increase of searched spaced parameters.

### 7.3.3 Optimal solution to Map-joining

Most of the map joining approaches start with linear approximation of map states and do optimisation as a post processing step to estimate the local solution [118, 150, 168]. Our approach is not dependent upon the initial guess and bound to search the optimal results by modified GRASP approach. We initialised each map randomly (unknown feature and vehicle pose) and solve the map joining problem as illustrated in pseudo



(a) Random initial guess with two relative maps

(b) GRASP based map-joining

Figure 7.12: GRASP based map-joining for simulation dataset [168] with 2 relative maps.

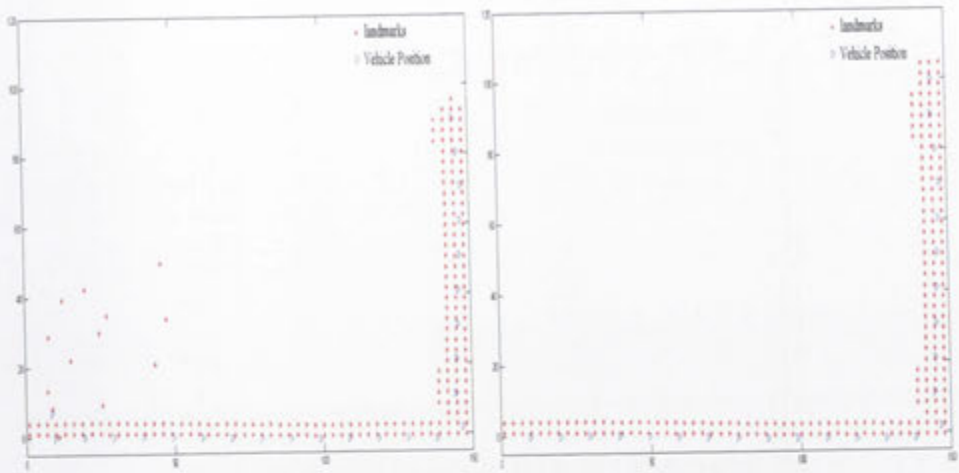
code algorithm 3. The data association is assumed to be known (still the problem is highly non-linear due to unknown vehicle pose and feature position [1]). Two maps are considered at each time and provided to GRASP algorithm, which returns the optimal solution for those two sets of maps observation. The input to algorithm 1 will be set of observations (relative feature and odometry information of relative maps) to obtain $x^*$.

**Algorithm 4** MAP Joining Optimal: GRASP variant for Map Joining

```
x = first relative map (k)
while Fuse relative map k+1(l) into x do
    Data Association assumed to be known
    Initialize random pose/landmark for l into x
    x = GRASP-Algorithm (x, Local Map k + 1)
    k = k + 1
end while
return (x* = x)
```

### 7.3.4 Results and Discussion

The performance of the proposed algorithm is tested on simulated [168] as well as on the real dataset [160], which are both available publicly.




(a) Before joining (b) After joining

Figure 7.13: Intermediate results of simulation dataset [168] with 50 relative maps

The $150 \times 150m^2$ simulation environment [168] containing 2500 features uniformly spaced in rows and columns is tested first. The robot started from the left bottom corner of the square and followed a big loop. A sensor with a field of view of 180 degrees and a range of 6 meters was simulated to generate relative range and bearing measurements between the robot and features. Two relative maps are considered from the dataset with unknown initial guesses (covariance matrices are set to identity). Multiple trials were performed with random initial guesses to be processed with the proposed approach for map-joining, which always converge to a global optimal solution without being affected by the variation in initial guess, as shown in Figure 7.12. The approach was also tested on a multiple relative maps (as the upper bound is available for joining two individual relative maps), so the obtained solution is an approximation of the original problem. 50 relative maps in total with 612 observed features and 1374 odometry/feature measurements were made from the robot poses. Figure 7.13(a) shows the intermediate results of the $25^{th}$ relative map where the new relative map has random initial guesses of landmark position and pose. The GRASP based smoothing is performed and approximated optimal results obtained are shown in Figure 7.13(b). Final results with ground truth result are shown in Figure 7.14, showing estimated position against the interpolated ground truth positions.

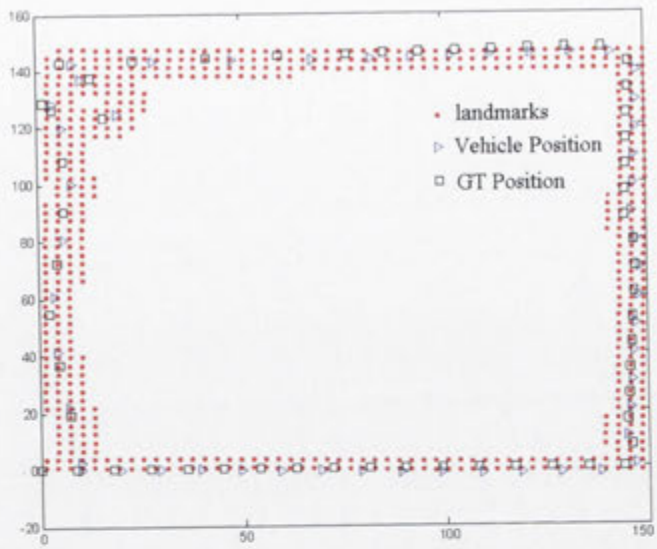
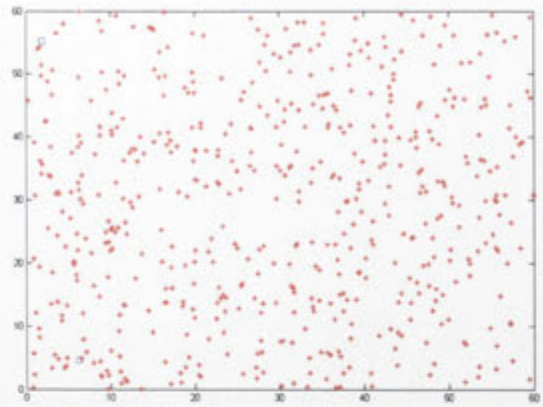


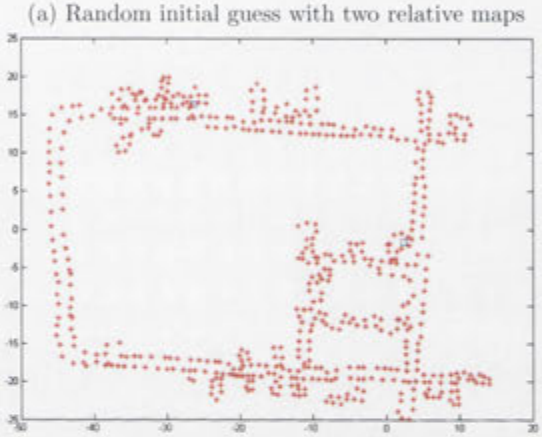

Figure 7.14: Ground truth and estimated vehicle/landmark positions for simulation dataset [168] with 50 relative maps

The GRASP based approach with unknown initial guess is tested on DLR dataset [160] and compared with the state-of-the-art map-joining approach [168]. The DLR dataset is acquired with a camera attached on a wheeled robot and odometry. The robot moved around a building detecting scattered artificial white/black landmarks, placed on the ground. Odometry measurements and relative positions of the observed landmarks are being provided. This dataset is also divided into two relative maps [5]. Figure 7.15 shows the results of the global optimal solution obtained after joining relative maps with unknown initial guesses (multiple trials resulted in the same optimal solution hence showing the independence to a good initial guess). The proposed approach is also tested on 200 relative maps with 540 observed features and 1680 odometry/feature observations when is considered with known data association. Figure 7.16(a) shows the random initial guess for vehicle pose and feature positions in global frame of reference, to be processed by GRASP approach. Figure 7.16(b and c) shows a visual comparison of proposed approach with [168] (which is booted with

[5]The concept was to have a one-step SLAM example in the form of two relative maps. The initialization of features and pose is done as random (unknown) to find the global minima solution. The benchmark dataset was selected to test the approach against a precise ground truth.



(a) Random initial guess with two relative maps

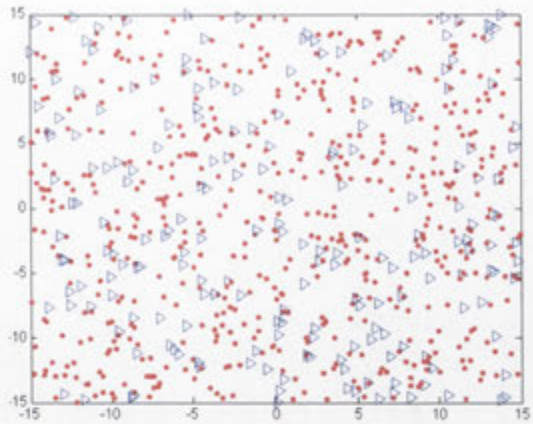


(b) GRASP based map-joining

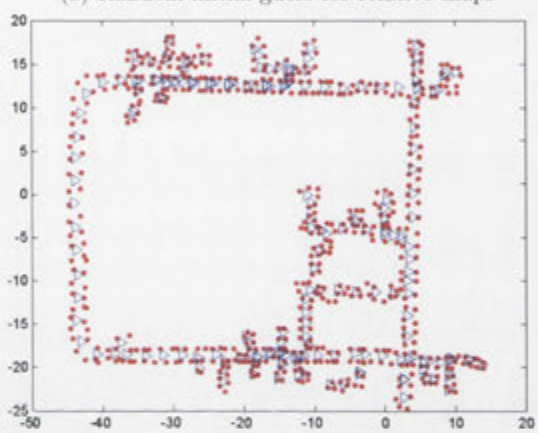
Figure 7.15: GRASP based map-joining for simulation dataset [160] with 2 relative maps

linear initialisation whereas our proposed approach is not dependent upon known initial guess).

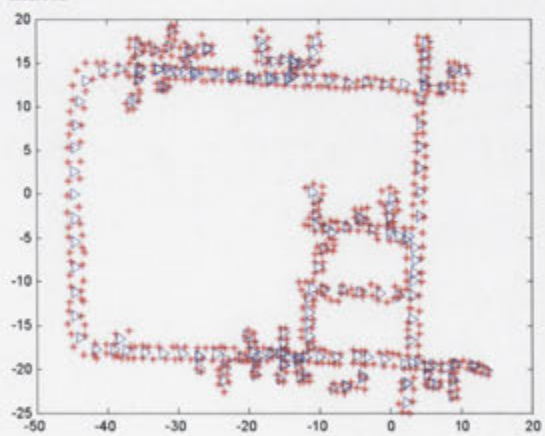
The performance of the experimental results validates the proposed idea and makes the SLAM problem solvable by global optimisation based approach without initial guess.



(a) Random initial guess for relative maps



(b) GRASP based vehicle/landmark position estimates



(c) I-SLSJF with EIF based initialization [168]

Figure 7.16: DLR dataset results using GRASP with 200 relative maps, showing global convergence (b) with random initial guesses (a).

## 7.4 Summary and Discussion

In this chapter, we analysed a theoretical foundation of SLAM by analysing its hidden structure. The problem was formulated as a non-linear least square feature based SLAM and stationary point analysis was conducted to determine the maximum number of local minima. The analysis conducted in this chapter helped to understand the highly non-convex nature of SLAM and showed that the two-pose SLAM is equivalent to solving a problem with two unknown variables. We also provided an upper bound for the number of local minima in a 5DOF SLAM problem, where one of the orientation angles was assumed to be known. For the two-pose 3D SLAM problem, the analysis revealed that there exist at most four local minima.

The second half of the chapter presented a practical approach to finding the globally optimal solution to the 2D SLAM problem. Local optimisation based strategies, which are mostly adopted for SLAM problems, are highly susceptible to the local minima problem due to the non-convex structure of the problem. By exploiting the theoretical limit on the number of local minima, we proposed a framework to estimate a global optima in a defined non-linear cost function with the assumption of spherical covariance. The proposed approach is not reliant on good initial guesses, which is the primary condition of local optimisation based approaches for global convergence. Experimental results were provided on the different datasets (available on-line) to validate the robustness of the approach.

# Chapter 8

# Summary and Conclusions

## 8.1 Introduction

In this thesis, we have focused on the vision sensor to provide the visual cues to aid and calibrate a low-cost inertial sensor in a probabilistic framework for the localisation, mapping and control of an aerial vehicle. We proposed a novel idea of aiding a low-cost inertial sensor with visual non-holonomic constraints. The idea is similar to the non-skidding assumption in wheeled robots [1, 2] while we extended it to the full 6DOF domains aiming for all-terrain navigation. The framework developed in this thesis demonstrated the utility and effectiveness of using visual non-holonomic constraints for maintaining inertial stability and performance.

This thesis comprises of eight chapters, in which Chapter 1 provided a research background as well as a rationale for conducting this research. Chapter 2 discussed the basic theory behind Bayesian estimation and sensor fusion architectures. Chapter 3 formulated the theory behind visual non-holonomic constraints. The idea of using hundreds of features to constrain the drift of the inertial sensor was developed in a monocular SLAM framework. Chapter 4 addressed the problem of depth dropout in the RGB-D sensor. A computationally efficient map management method was also developed for real-time implementation. Chapter 5 provided the details of the developed hardware and on-board implementation. Chapter 6 presented the results of the

Visual-Inertial SLAM for RGB-D sensor on aerial vehicles for both indoor and outdoor environments. The qualitative and quantitative evaluation of the proposed work was provided against the highly precise motion capturing system. The comparison showed that the visual non-holonomic constraints have better accuracy as compared to existing state-of-the-art approaches. Chapter 7 presented the theoretical analysis of the 3D SLAM problem and also provided the theoretical limit on the number of local minima (to solve the SLAM problem optimally).

This chapter summarised the contributions made in this thesis and also discussed the possible future directions.

## 8.2 Summary of Contributions

The two key challenges addressed in this work were: a) the computational complexity issue in the monocular feature based SLAM and the related scale-ambiguity problem, and b) the depth dropout problem in RGB-D (Kinect or Stereo) sensors was addressed. Alongside with the $3D$ SLAM framework investigation, a theoretical study was carried out on the convexity structure of a simplified Inertial-SLAM. The analysis shed light on solving the non-linear SLAM problem with a global optimisation based approach. In addition, we designed and developed an in-house inertial sensor and an aerial platform. All the processing was performed in real-time, on-board, with a closed-loop controller (through an autopilot) for the hovering of the aerial vehicle.

This thesis made four main contributions:

- To utilise the visual aiding information for an inertial sensor at higher update rates, this work proposes a novel visual non-holonomic constraint. Firstly, these visual constraints were utilised in the monocular SLAM to resolve the computational complexity problem for a large scale environment. We also utilised these visual constraints to solve the problem of depth dropout in RGB-D sensors. We addressed the depth dropout problem by proposing a novel Visual-Inertial

fusion framework (a graceful degradation mechanism that uses the 2D sensing mode in case of the unavailability of 3D information).

- The hardware development of the low-cost IMU using MEMS sensors (with gyroscope, accelerometer, and magnetometer). The IMU was calibrated statically in the laboratory and dynamically during the flight.
- The architecture of our implementation was developed with modularity and robustness in mind. A demonstration of the developed framework was evaluated on an in-house built multi-rotor platform with hovering control. For the purpose of demonstration, an attitude and position PID controller was also developed through an open-source autopilot system.
- The convexity structure of an Inertial-SLAM was investigated by analysing the number of local minima. We analysed the number of local minima of a two-pose 5DOF Inertial SLAM system by utilising stationary point analysis, and hence proved that when one of the orientation angles is assumed known, there exist a maximum of four local minima in the system.

## 8.3 Future Directions

The following section summarises the future research directions on the topics dealt with in this thesis (Visual-Inertial aerial SLAM) and the possible enhancements of the proposed framework.

In the SLAM convexity analysis, global optimal research needs to be extended to the 6DOF scenario where the complete rotation transformations must be considered. Another possible extension for the 5DOF analysis is to look into the multi-step SLAM (which is highly non-linear).

This thesis has demonstrated the developed framework of Visual-Inertial mapping and navigation using a low-cost IMU and monocular/Kinect sensor. The proposed

approach needs to be tested on the stereo system and evaluated in an outdoor large scale environment [1].

From the implementation point-of-view, there are also a few works which need to be improved upon. IMUs are known to be more temperature dependent than time dependent, meaning the change in bias will be larger whenever there is a change in temperature than it will be just over time. So tests should be performed to determine the ratio between temperature and bias and hence corrected. Further work can also be done to optimise the performance of the on-board computer. Currently it is running Ubuntu 10.04, which is a full-featured installation that continuously runs a graphical user interface (GUI). Since the on-board computer is mounted to the hexacopter and controlled via SSH, this GUI is not utilised and is therefore an unnecessary overhead. To remove this waste, a minimal installation of Ubuntu (which is optimised for low-end computers) could be used instead.

We have seen that the position controller can adequately maintain its position in 2D with 5DOF (spatial displacement in the plane as well as maintaining orientation). In the future, an important extension to the position controller implementation would be to implement altitude hold as well. It could then be said that the position controller works in 3D with 6 DOF. However, altitude control is hard to implement, as one of the reasons for this is the low frequency of the controller loops in the autopilot (APM [122]). The autopilot runs the altitude controller at 50Hz, which is half the frequency of the roll, pitch, and yaw controllers. This leads to a poorer response in the throttle control, but the APM is not capable of running all controllers at the full 100Hz frequency. If the APM were to be replaced with the PX4 flight controller [129] (which could run the same modified code), this would resolve the problem and increase the frequency of all the controllers. However, this still leaves the problem that altitude data is not updated frequently enough if the VICON system is not used, which prevents outdoor flights. Although there is an on-board barometric sensor, its data is not reliable. To aid in this implementation, a possible approach would be to incorporate sonar readings of height into the altitude controller. This would provide

[1] The large scale environment tested in this thesis is up to 1.1Km (Chapter 3)

high-frequency altitude readings as long as the hexacopter was within a certain distance to the ground. Beyond that, a downward facing laser scanner could be used to obtain readings.

From the point-of-view of application, further work in the direction of path following is also required. This could be done by updating the position controller so that the home position is now the desired position, which can be changed over time. The next logical development, especially if autonomous outdoor flight is desired, is to implement an obstacle avoidance algorithm. As the current controller is only capable of moving the hexacopter to a desired position, this will require a path planning algorithm to be developed and run on the Visual-Inertial SLAM generated map. However, this will not be completely safe unless the entire environment surrounding the hexacopter has first been mapped (for example, by running the Kinect-SLAM algorithm while controlling the hexacopter manually through R/C), as the hexacopter is capable of flight in directions that the Kinect sensor cannot observe. Therefore, either additional sensors will need to be installed or the hexacopter's movement will need to be constrained to forward movement only. The next potential development would be to network multiple hexacopters together, enabling them to simultaneously contribute to a single map.

# Appendix A

# Sensor Specification

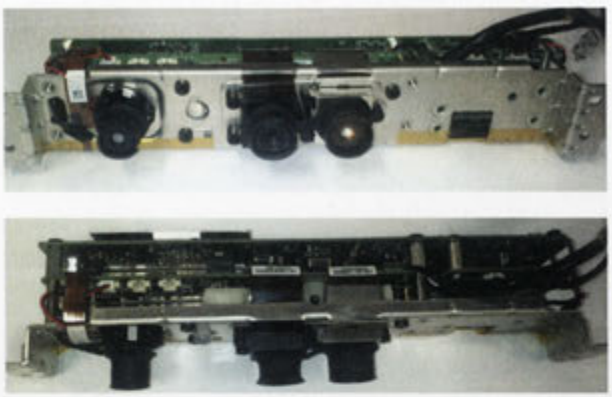
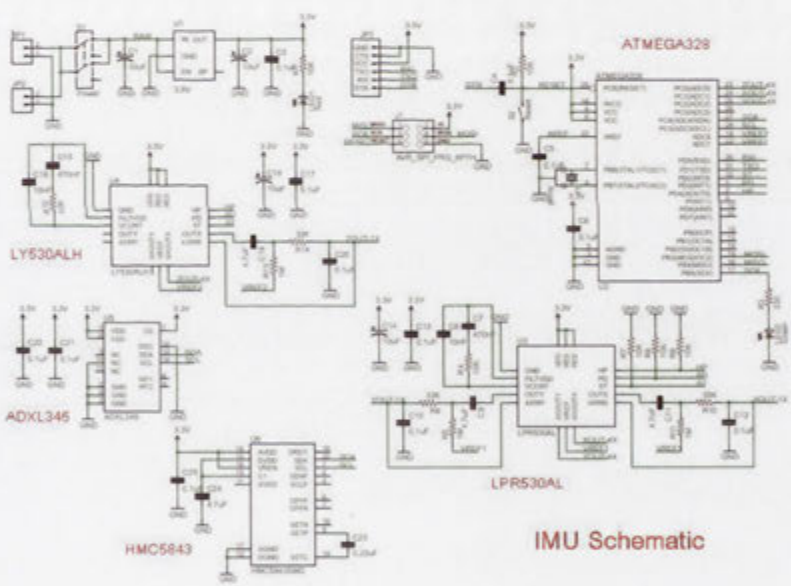


Figure A.1: Vision Sensor Used for Pose Estimation (Microsoft's Kinect)




(a)

Figure A.2: Schematic diagram of the developed IMU

# Bibliography


[1] G. Dissanayake, S. Sukkarieh, E. Nebot, and D.-W. H., "The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications," in *In IEEE Tran. on Robotics*, pp. 731–747, 2001.

[2] D. Scaramuzza, F. Fraundorfer, and R. Siegwart, "Real-time monocular visual odometry for on-road vehicles with 1-point ransac," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4293–4299, 2009.

[3] B. Williams and I. Reid, "On combining visual slam and visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[4] G. Nuetzi, S. Weiss, D. Scaramuzza, , and R. Siegwart, "Fusion of imu and vision for absolute scale estimation in monocular slam," in *Journal of Intelligent and Robotic Systems*, 2011.

[5] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments," *Journal of Robotics Research (IJRR)*, 2012.

[6] D. Herath, S. Kodagoda, and G. Dissanayake, "Simultaneous localisation and mapping: A stereo vision based approach," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[7] S. Thrun, W. Burgard, and D. Fox, "Probabilistic robotics," in *The MIT press, Cambridge*, 2005.

[8] M. Bryson and S. Sukkarieh, "Building a robust implementation of bearing-only inertial slam for a uav," *Journal of Field Robotics*, vol. 24, no. 1-2, pp. 113–143, 2007.

[9] N. Nourani-Vatani, J. Roberts, and M. Srinivasan, "Imu aided 3d visual odometry for car-like vehicles," in *Australasian Conference on Robotics and Automation (ACRA)*, 2008.

[10] R. Eustice, O. Pizarro, and H. Singh, "Visually augmented navigation for autonomous underwater vehicles," in *IEEE Journal of Oceanic Engineering*, 2008.

[11] J. Kelly and S. Sukhatme, "Visual-inertial sensor fusion:localization mapping and sensor-to-sensor self-calibration," in *The International Journal of Robotics Research*, 2011.

[12] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-slam-based navigation for autonomous micro helicopters in gps-denied environments," in *In Journal of Field Robotics*, 2011.

[13] Amazon, "Prime air, http://www.amazon.com/b," in *Amazon web announcement*, 2013.

[14] J. Fink, N. Michael, S. Kim, and V. Kumar, "Planning and control for cooperative manipulation and transportation with aerial robots," in *International Symposium of Robotics Research, Switzerland*, 2009.

[15] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Vision-based state estimation and trajectory control towards aggressive flight with a quadrotor," *Robotics Science and Systems Conference (RSS)*, 2013.

[16] A. Vedaldi, G. Guidi, and S. Soatto, "Moving forward in structure from motion," in *IEEE International Conference on Computer Vision and Pattern Recognitions*, pp. 1–7, 2007.

[17] U. Qayyum and J. Kim, "Seamless aiding of inertial-slam using visual directional constraints from a monocular vision," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[18] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart, "Onboard imu and monocular vision based control for mavs in unknown in- and outdoor environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[19] THawk, "The fukushima inspection, spectrum.ieee.org/robotic-aerial-vehicle-at-fukushima-reactors," in *IEEE Spectrum*, 2011.

[20] D. Helmick, S. Roumeliotis, Y. Cheng, D. Clouse, M. Bajracharya, , and L. Matthies, "Slip compensation for a mars rover," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2005.

[21] L. Kneip, M. Chli, and R. Siegwart, "Robust real-time visual odometry with a single camera and an imu," in *British Machine Vision Conference (BMVC)*, 2011.

[22] J. Kim and S. Sukkarieh, "Complementary slam aided gps/ins navigation in gnss denied and unknown environments," in *International Symposium on GNSS/GPS*, (Sydney, Australia), 2004.

[23] A. Gutierrez, J. Nieto, T. Calleja, and E. Nebot, "Large scale visual odometry using stereo vision," in *Australasian Conference on Robotics and Automation (ACRA)*, 2009.

[24] A. Conway, "Autonomous control of an unstable model helicopter using carrier phase gps only," in *PhD Thesis, Stanford University, USA*, 1995.

[25] S. Sukkarieh, E. Nebot, and H. Durrant-Whyte, "A high integrity imu/gps navigation loop for autonomous land vehicle applications," *IEEE Transactions on Automatic Control*, vol. 15, pp. 572–578, June 1999.

[26] R. Greenspan, *GPS and Inertial Navigation*, ch. 7. American Institute of Aeronautics and Astronautics, Inc, 1996.

[27] S. Snyder, B. Schipper, L. Vallot, N. Parker, and C. Spitzer, "Differential gps/inertial navigation approach/landing flight test results," *IEEE Transactions on Aerospace and Electronic Systems Magazine*, pp. 3–11, 1992.

[28] S. Sukkarieh, E. Nettleton, J. Kim, M. Ridley, A. Goktogan, and H. Durrant-Whyte, "The anser project: Multi-uav data fusion," 2002.

[29] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Int. Symposium on Experimental Robotics*, 2010.

[30] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," in *IEEE Control Systems Magazine*, vol. 28, pp. 51–64, 2008.

[31] S. Bieniawski, D. Halaas, and J. Vian, "Micro-aerial vehicle flight in turbulent environments: Use of an indoor flight facility for rapid design and evaluation," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008.

[32] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the rgb-d slam system," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[33] M. Bosse, "Atlas: a framework for large scale automated mapping and localization," in *Ph.D. dissertation, Massachusetts Institute of Technology*, 2004.

[34] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unknown indoor environments," in *International Journal of Micro Air Vehicles*, 2009.

[35] J. Civera, A. Davison, and M. Montiel, "Inverse depth parametrization for monocular slam," in *IEEE Transactions on Robotics*, 2008.

[36] C. Nguyen, "Constructing drivability maps from 3d laser range data for autonomous vehicles," in *Technical Report: The University of Texas at Austin, 2010*, 2010.

[37] P. Agarwal and T. Brady, "Slam strategy for an autonomous quadrotor," in *Report, Computer Science and Engineering, University of Michigan*, 2008.

[38] P. Corke, J. Lobo, and J. Dias, "An introduction to inertial and visual sensing," *International Journal of Robotics Research*, vol. 26, pp. 519–535, December 2007.

[39] J. Kim and S. Sukkarieh, "Airborne simultaneous localisation and map building,"

[40] D. Eberli, D. Scaramuzza, S. Weiss, and R. Siegwart, "Vision based position control for mavs using one single artificial landmark," in *International Conference on Unmanned Aerial Vehicles*, 2010.

[41] J. Kim, "Autonomous navigation for airborne applications," in *Australian Centre for Field Robotics, The University of Sydney*, 2004.

[42] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[43] J. Engel, J. Sturm, and D. Cremers, "Camera-based navigation of a low-cost quadrocopter," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[44] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[45] A. Gutierrez, J. Nieto, T. Calleja, and E. Nebot, "Large scale visual odometry using stereo vision," in *Australasian Conference on Robotics and Automation (ACRA)*, 2009.

[46] K. Tim, H. Andrew, B. Max, W. Garrison, and M. Larry, "Gamma-slam: Stereo visual slam in unstructured environments using variance grid maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[47] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid, "Rslam: A system for large-scale mapping in constant-time using stereo," in *International Journal of Computer Vision*, 2011.

[48] J. Kelly, S. Saripalli, and G. Sukhatme, "Combined visual and inertial navigation for an unmanned aerial vehicle," in *6th International Conference on Field and Service Robotics (FSR)*, 2007.

[49] S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an rgb-d camera," in *International Symposium on Robotics Research (ISRR)*, 2011.

[50] Parrot, "Ar. drone," in *http://ardrone2.parrot.com/*, 2013.

[51] T. Lupton and S. Sukkarieh, "Removing scale biases and ambiguity from 6dof monocular slam using inertial," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3698–3703, 2008.

[52] U. Qayyum and J. Kim, "Visual odometry and inertial slam integrated for large scale environment," in *International Global Navigation Satellite Systems Conference (IGNSS)*, 2011.

[53] K. Konolige, M. Agrawal, and J. Sola, "Large scale visual odometry for rough terrain," in *International Symposium on Research in Robotics (ISRR)*, 2007.

[54] H. Strasdat, J. Montiel, and A. Davison, "Visual slam: Why filter?," in *Image and Vision Computing (IMAVIS)*, 2012.

[55] G. Grisetti, D. Rizzini, C. Stachniss, E. Olson, and W. Burgard, "Online constraint network optimization for efficient maximum likelihood mapping," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1880–1885, 2008.

[56] E. Olson, J. Leonard, and S. Teller, "Fast iterative optimization of pose graphs with poor initial estimates," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2006.

[57] U. Qayyum and J. Kim, "Omni visual-laser 3d scanner for outdoor mapping," in *IEEE International Conference. on Robotics and Biomimetics (ROBIO)*, 2013.

[58] M. Grewal and A. Andrew, *Kalman Filtering: Theory and Practice.* Prentice–Hall, Inc., 1993.

[59] R. Brown and Y. Hwang, *Introduction to Random Signals and Applied Kalman Filtering.* Wiley, New York, 1992.

[60] P. Maybeck, *The Kalman filter: an introduction to concepts.* Academic Press, Stochastic models estimation and control, 1979.

[61] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "Fast-slam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *International Joint Conference on Artificial Intelligence*, pp. 1151–1156, 2004.

[62] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing," in *International Journal of Robotics Research*, 2006.

[63] P. Jung, G. Lim, and K. Kong, "A mobile motion capture system based on inertial sensors and smart shoes," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 692–697, 2013.

[64] G. Dissanayake, S. Sukkarieh, E. Nebot, and H. Durrant-Whyte, "The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications," in *IEEE Transaction on Robotics*, 2001.

[65] A. KING, "Inertial navigation - forty years of evolution," in *Marconi Electronic Systems Ltd*, 1998.

[66] M. Baumker and A. Mattissek, "Integration of a fiber optical gyro attitude and heading reference system with differential gps," in *In Proceedings of Institute of Navigation (ION)*, pp. 1093–1101, 1992.

[67] Inertial Science Inc., *Technical Manual: Inertial Measurement Unit, ISIS Rev.C*, 1999.

[68] D. Derwai, U. Qayyum, and J. Kim, "Development of a low cost imu system for small-scale flyers," in *Undergraduate report, ANU*, 2011.

[69] J. Craig, *Introduction to Robotics. Mechanics and Control.* PEARSON. Prentice. Hail. Pearson Education International, 2004.

[70] S. Sukkarieh, *Aided Inertial Navigation Systems for Autonomous Land Vehicles.* PhD thesis, Australian Centre for Field Robotics, The University of Sydney, 1999.

[71] J. Kim, S. W. Moon, S. H. Kim, D. H. Hwang, S. J. Lee, M. S. Oh, and S. W. Ra, "Design of a loosely-coupled gps/ins integration system," *Journal of the Korea Institute of Military Science and Technology*, vol. 2, pp. 186–196, December 1999.

[72] J. Wendel and G. F. Trommer, "Tightly coupled gps-ins integration for missile applications," *In Proceeding of Aerospace Science and Technology*, vol. 8, pp. 627–634, 2004.

[73] U. Qayyum and J. Kim, "Inertial-kinect fusion for outdoor 3d navigation," *Australian Conference on Robotics and Automation (ACRA)*, 2013.

[74] S. W. Moon, J. Kim, S. H. Kim, D. H. Hwang, S. J. Lee, M. S. Oh, and S. W. Ra, "An implementation of a loosely-coupled gps/ins navigation system," in *International Symposium on Satellite Navigation Technology*, (Brisbane, Australia), 1999.

[75] L. Armesto, J. Tornero, and M. Vincze, "Fast egomotion estimation with multirate fusion of inertial and vision," *International Journal of Robotics Research*, vol. 26, pp. 577–589, 2007.

[76] I. Mourikis and S. Roumeliotis, "A multistate constraint kalman filter for vision-aided inertial navigation," *IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[77] U. Qayyum and J. Kim, "Visual-inertial motion priors for robust monocular slam," in *Towards Autonomous Robotics Systems Conference (TAROS)*, 2011.

[78] K. Konolige and M. Agrawal, "Frameslam: From bundle adjustment to real-time visual mapping," in *In IEEE Tran. on Robotics*, 2008.

[79] R. Hartley and H. Li, "Five-point algorithm made easy," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2012.

[80] H. Strasdat, J. Montiel, and A. Davison, "Real-time monocular slam: Why filter? (pdf format)," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[81] W. Brain and I. Reid, "On combining visual slam and visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[82] M. Milford and G. Wyeth, "Single camera vision-only slam on a suburban road network," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[83] J. Kim and S. Sukkarieh, "Real-time implementation of airborne inertial-slam," in *Robotics and Autonomous Systems*, 2007.

[84] J. Civera, O. Grasa, A. Davison, and J. Montiel, "1-point ransac for ekf filtering: Application to real-time structure from motion and visual odometry," in *In Journal of Field Robotics*, 2010.

[85] P. Pinies, T. Lupton, S. Sukkarieh, and J. Tardos, "Inertial aiding of inverse depth slam using a monocular camera," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[86] J. Blanco, F. Moreno, and J. Gonzalez, "A collection of outdoor robotic datasets with centimeter-accuracy ground truth," in *In Journal of Autonomous Robots*, pp. 327–351, 2009.

[87] I. Dryanovski, R. Valenti, and J. Xiao, "Fast visual odometry and mapping from rgb-d data," *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[88] J. Sola, A. Moninand, and D. M, "Bicamslam: Two times mono is more than stereo," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[89] G. Inc., "Project tango [online] available at `https://developers.google.com/project-tango/concepts?hl=en`," 2014.

[90] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera," in *24th annual ACM symposium on User interface software and technology, ACM*, 2011.

[91] S. Scherer, D. Dube, and A. Zell, "Using depth in visual simultaneous localisation and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[92] T. Whelan, J. McDonald, H. Johannsson, M. Kaess, and J. Leonard, "Robust real-time visual odometry for dense rgb-d mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[93] L. Matthies, "Dynamic stereo vision," *PhD thesis, Dept. of Computer Science, Carnegia Mellow University*, 1989.

[94] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for rgb-d cameras," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[95] G. Hu, S. Huang, L. Zhao, A. Alempijevic, and G. Dissanayake, "A robust rgb-d slam algorithm," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[96] E. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," in *International Journal of Robotics Research*, 2010.

[97] B. Bouvrie, "Improving rgbd indoor mapping with imu data," *Delft University of Technology Master's Thesis in Embedded Systems*, 2011.

[98] H. Ovren, P. Forssen, and D. Tornqvist, "Why would i want a gyroscope on my rgb-d sensor?.," *Proceedings of IEEE Winter Vision Meetings, Workshop on Robot Vision (WoRV13)*, 2013.

[99] S. Rusinkiewicz and M. Levoy, "Effcient variants of the icp algorithm," in *In 3D Digital Imaging and Modeling Conference (3DIM)*, 2001.

[100] D. Dial, P. DeBitetto, and S. Teller, "Epipolar constraints for vision-aided inertial navigation," *Proceedings of IEEE WACV/MOTION*, pp. 221–228, 2005.

[101] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[102] C. Herrera, D.Kannala, and J. Heikkila, "Joint depth and color camera calibration with distortion correction," 2012.

[103] J. Shi and C. Tomasi, "Good features to track," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[104] H. Bay, A. Ess, T. Tuytelaars, and L. Gool, "Surf: Speeded up robust features," in *Computer Vision and Image Understanding (CVIU)*, 2008.

[105] R. Rusu, N. Blodow, Z. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[106] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, "BRIEF: Computing a Local Binary Descriptor Very Fast," *IEEE Transaction of Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, no. 7, pp. 1281–1298, 2012.

[107] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns," vol. 32, pp. 448–461, 2010.

[108] S. Winder, G. Hua, and M. Brown, "Picking the best daisy," in *IEEE International Conference on Computer Vision and Pattern Recognitions*, 2009.

[109] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," vol. 27, pp. 1615–1630, 2004.

[110] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision (IJCV)*, 2001.

[111] K. Khoshelham and O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," in *Sensors (doi:10.3390/s120201437)*, 2012.

[112] D. Nister, "Preemptive ransac for live structure and motion estimation," in *IEEE International Conference on Computer Vision*, pp. 199–206, 2003.

[113] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Journal of the Optical Society of America*, 1987.

[114] J. Frahm and M. Pollefeys, "Ransac for (quasi-)degenerate data (qdegsac)," in *IEEE International Conference on Computer Vision and Pattern Recognitions*, pp. 453–460, 2006.

[115] H. Strasdat, A. Davison, J. Montiel, and K. Konolige, "Window optimisation for constant time visual slam," in *IEEE International Conference on Computer Vision*, 2011.

[116] E. Eade and T. Drummond, "Monocular slam as a graph of coalesced observations," in *ICCV*, 2007.

[117] S. Williams and I. Mahon, "Simultaneous localization and mapping on the great barrier reef," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

[118] K. Ni, D. Steedly, and F. Dellaert, "Tectonic sam: exact, out-of-core, submap-based slam," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[119] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii," in *IEEE Robotics and Automation Magazine*, 2006.

[120] A. Mourikis and S. Roumeliotis, "A dual-layer estimator architecture for long-term localization," *Proceedings of the Workshop on Visual Localization for Mobile Platforms*, 2008.

[121] AscTech, "Ascending technologies research catalogue," in *http://www.asctec.de/downloads/flyer/*, 2013.

[122] Arducopter, "An arducopter repository -open source autopilot system," in *Arducopter, http://copter.ardupilot.com/*, 2013.

[123] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 153–158, 2007.

[124] A. Alaimo, V. Artale, C. Milazzo, A. Ricciardello, and L. Trefiletti, "Mathematical modeling and control of a hexacopter," in *International Conference on Unmanned Aircraft Systems*, 2013.

[125] 3DRobotics, "Supporting open-source aerial platform," in *http://store.3drobotics.com/*, 2013.

[126] Quadframe, "Landing legs 200mm," in *http://www.quadframe.us/, 2013*, 2013.

[127] J. Khor, U. Qayyum, and J. Kim, "Hovering control of a hexacopter using vision-based slam," in *Undergraduate report, ANU*, 2013.

[128] T. E. Zurich, "Mavlink micro air vehicle communication protocol," in *http://qgroundcontrol.org/mavlink/start*, 2009.

[129] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A system for autonomous flight using onboard computer vision," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[130] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: an open-source robot operating system," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[131] M. Achtelik, "Vicon drivers," in *https://github.com/ethz-asl/ros-drivers/tree/master/viconbridge*, 2011.

[132] Openni, "Online documentation of openni library, an opensource framework for 3d sensing," 2013.

[133] ROS, "Online documentation of robot operating system," in *http://www.ros.org/*, 2013.

[134] OpenCV, "Online documentation of opencv library," in *http://code.opencv.org/projects/opencv/wiki/*, 2013.

[135] Eigen, "Online eigen library documentation," in *http://eigen.tuxfamily.org*, 2013.

[136] R. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[137] Boost, "Online documentation of boost library," in *http://www.boost.org*, 2013.

[138] P. Besl and N. McKay, "A method for registration of 3d shapes," in *IEEE Transaction Pattern Analysis and Machine Intelligence (PAMI)*, 1992.

[139] B. Clipp, B. Lim, J. Frahm, and M. Pollefeys, "Parallel, real-time visual slam," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[140] G. Franklin, J. Powell, and A. Emami-Naeini, "Feedback and control of dynamic systems," in *Sixth Edition of the Book from Pearson Higher Education*, 2010.

[141] Hobbyking, "Online store," in *http://www.hobbyking.com.au*, 2013.

[142] Droidworx-Aeronavics, "Xm6 hexacopter," in *http://quadcopters.co.uk/droidworx-xm-6/hexacopter-ready-to-fly-669-p.asp*, 2013.

[143] Cinestar-Quad, "Cinestar 6 ready to fly," in *http://www.quadrocopter.com*, 2013.

[144] LPR530AL, "Gyroscope sensor,mems motion sensor, dual axis pitch and roll analog gyroscope and analog yaw-rate gyroscope," in *from http://www.sparkfun.com/datasheets/Sensors/LY530ALH.pdf retrieved on*, 2013.

[145] ADXL345, "3-axis, digital accelerometer, analog devices, inc," in *http://www.analog.com/static/imported-files/datasheets/ADXL345.pdf retrieved on*, 2013.

[146] D. Gebre-Egziabher, "Magnetometer autocalibration leveraging measurement locus constraints," in *Journal of Aircraft*, vol. 44, 2007.

[147] J. Lobo and J. Dia, "Relative pose calibration between visual and inertial sensors," in *International Journal of Robotics Research, Vol 26*, 2007.

[148] U. Qayyum and J. Kim, "Global optimal solution to slam problem with unknown initial estimates," *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pp. 76–83, 2012.

[149] S. Huang, H. Wang, U. Frese, and G. Dissanayake, "On the number of local minima to the point feature based slam problem," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 265–272, 2012.

[150] S. Huang, Z. Wang, and G. Dissanayake, "Sparse local submap joining filter for building large-scale maps," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1121–1130, 2008.

[151] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," in *Journal of Autonomous Robots*, 1997.

[152] T. Duckett, S. Marsland, and J. Shapiro, "Fast, on-line learning of globally consistent map," in *Journal of Global Optimization*, 2002.

[153] U. Frese, P. Larsson, and T. Duckett, "A multilevel relaxation algorithm for simultaneous localization and mapping," in *IEEE Transactions on Robotics*, 2005.

[154] S. Huang, Y. Lai, U. Frese, and G. Dissanayake, "How far is slam from a linear least squares problem," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3011–3016, 2010.

[155] D. Rizzini, "Towards a closed-form solution of constraint networks for maximum likelihood mapping," *14th International Conference on Advanced Robotics (ICAR)*, 2009.

[156] R. Hartley and F. Kahl, "Global optimization through rotation space search," in *International Journal of Computer Vision (IJCV)*, 2009.

[157] C. Olsson, F. Kahl, and M. Oskarsson, "Branch and bound methods for euclidean registration problems," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2008.

[158] K. Mitra and R. Chellappa, "A scalable projective bundle adjustment algorithm using the l $\infty$ norm," in *Computer Vision, Graphics Image Processing (ICVGIP)*, 2008.

[159] D. Gamerman, "Markov chain monte carlo: Stochastic simulation for bayesian inference," in *Book: Chapman and Hall/CRC Press*, 1997.

[160] J. Kurlbaum and U. Frese, "A benchmark data set for data association," in *Technical Report, University of Bremen*, 2010.

[161] R. Freund and F. Jarr, "Solving the sum-of-ratios problem by an interior-point method," in *Journal of Global Optimization*, 2001.

[162] K. Astrom, O. Enqvist, C. Olsson, F. Kahl, and R. Hartley, "An l$\infty$ approach to structure and motion problems in 1d-vision," in *IEEE International Conference on Computer Vision*, 2007.

[163] R. Iser and F. Wahl, "Antslam: global map optimization using swarm intelligence," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[164] U. Qayyum and J. Kim, "Global optimization for 2d slam problem," *Lecture Notes in Electrical Engineering Volume 283*, pp. 35–49, 2014.

[165] M. Resende and C. Ribeiro, "Greedy randomized adaptive search procedures," in *Handbook of Metaheuristics*, 2003.

[166] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," in *ACM Computing Surveys*, 2003.

[167] S. LEE, "Greedy randomized adaptive search procedure fortraveling salesman problem," in *Master of Science Thesis in Texas A & M University*, 2005.

[168] S. Huang, Z. Wang, G. Dissanayake, and U. Frese, "Iterated slsjf: A sparse local submap joining algorithm with improved consistency," in *Australasian Conference on Robotics and Automation (ACRA)*, 2008.