UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING AND THE ENVIRONMENT

Aeronautics Astronautics and Computational Engineering

An Investigation into Visual Control of Small Remotely Piloted Aircraft in GPS-denied Environments

by

Chang Liu

Thesis for the degree of Doctor of Philosophy

May 2017

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND THE ENVIRONMENT
Aeronautics Astronautics and Computational Engineering

Doctor of Philosophy

AN INVESTIGATION INTO VISUAL CONTROL OF SMALL REMOTELY PILOTED AIRCRAFT IN GPS-DENIED ENVIRONMENTS

by Chang Liu

This thesis studies the design and implementation of a quadrotor system with the take-off mass lower than 1 kg, which is capable of autonomous navigation in unstructured and initially unknown environments, without the assist from any external localisation information. This is particularly useful for indoor or urban applications. While operating in such constrained space without GPS access poses multiple challenges for localisation, control, onboard sensor and computation. It focuses on monocular vision based methods to realise this goal, due to its size, power consumption and rich information captured. The aim of this research is to firstly explore the method for the real-time six degree-of-freedom (DOF) state estimation for the quadrotor, based on onboard inertial sensors and an onboard monocular camera; secondly develop the precise control method for the highly dynamic system, and thirdly realise a small platform with all the computation onboard.

It shows that the real-time 6 DOF state estimation can be realised by adding a smartphone computer onboard the quadrotor, to process the visual and inertial data. Specifically, it performs a loosely coupled visual-inertial sensor fusion algorithm, overlaid on top of a state-of-the-art monocular simultaneous localization and mapping (mSLAM) algorithm. Due to the complementary nature between visual SLAM and inertial measurement, the combination results in a fast state estimation with sub-centimetre accuracy. Moreover, given the high quality state estimation, we have shown that the quadrotor position can be precisely controlled by a cascaded model-based PID-like controller. A micro quadrotor testbed was developed from the ground up, including the vehicle mechanical structure and autopilot electronics. Since all the computation executes onboard the airborne platform, it results in a power-on-and-go system without the need for a ground station. Finally, a novel launching method and active SLAM planner were implemented, as well as a tether power solution was implemented to provide indefinite power to the platform. This summaries the thesis as a thorough work going from initial sensor preparation to detailed theory, through careful implementation downto the real world testing.

Contents

List of Figures

List of Tables

Declaration of Authorship

I, Chang Liu , declare that the thesis entitled *An Investigation into Visual Control of Small Remotely Piloted Aircraft in GPS-denied Environments* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as: (Liu et al., 2016a), (Liu and Prior, 2015c), (Liu et al., 2015) and (Liu et al., 2016b)

Signed:..

Date:..

Acknowledgements

First of all, I would like to thank Dr Stephen D. Prior and Prof. James Scanlan for having offered me the possibility of doing a PhD under their supervision. They have provided me a highly inspiring environment to achieve the present work. Being part of the Aeronautics, Astronautics and Computational Engineering group, the Autonomous Systems Lab, was a highly appreciated experience. The group's focus on practical applications, had a very positive effect on directing the research work presented in this thesis.

Secondly, I would like to thank my father Jianguo Liu, my mother Haimin Zhang, who have used their funding from their personal savings to support this research work. I would also like to thank the University of Southampton for providing funding as the deduction of the tuition fee.

Thirdly, I would like to thank my wife Liyuan Liu, and my parents for enormous encouragement they offered throughout this research. This is the key for the continuous process on this work.

Finally, I would like to thank Dr Luke Teacy, for his time and effort used to discuss technical and theoretical details in this work, as well as Mantas Brazinskas and Mehmet Ali Erbil for their continued technical support and encouragement

This work would not have been possible without the all the people involved. Here, I send all my deepest sincere thank and appreciation to all of you.

Acronyms

AR	Augmented Reality.
CEP	Circular Error Probable.
COTS	Commercial Off-The-Shelf.
CPU	Central Processing Unit.
CSAC	Chip Scale Atomic Clock.
DOF	Degree of Freedom.
DTAM	Dense Tracking and Mapping.
EKF	Extended Kalman Filter.
FOV	Field of View.
GPS	Global Positioning System.
GPU	Graphical Processing Unit.
HDOP	Horizontal Dilution of Precision.
IBVS	Image-based Visual Servoing.
IMU	Inertial Measurement Unit.
KLT	Kanade, Lucas and Tomasi Tracker.
LIDAR	Light Detection And Ranging.
LQE	Linear Quadratic Estimation.
MAV	Micro Aerial Vehicle.
MEMS	MicroElectroMechanical Systems.
MIT	Massachusetts Institute of Technology.
OF	Optical Flow.

PBVS	Position-Based Vision Servoing.
PCL	Point Cloud Library.
PDF	Pseudo-Derivative Feedback.
PID	Proportional-Integral-Derivative Controller.
PTAM	Parallel Tracking And Mapping.
RADAR	Radio Detection and Ranging.
RANSAC	RANndom SAmple Consensus.
ROS	Robotic Operating System.
RPA	Remotely Piloted Aircraft.
RPAS	Remotely Piloted Aircraft System.
SAR	Synthetic Aperture Radar.
SFM	Structure From Motion.
SLAM	Simultaneous Localization and Mapping.
SSH	Secure Shell.
SVO	Semi-direct monocular Visual Odometry.
TIMU	Timing and Inertial Measurement Unit.
TOF	Time of Flight Camera.
UAV	Unmanned Aerial Vehicle.
UKF	Unscented Kalman Filter.
VINS	Visual Inertial Navigation System.
VR	Virtual Reality.
VTOL	Vertical Take-off and Landing.

Chapter 1

Introduction

In nature, the sensor capturing the richest information is probably the eye. It is believed that the pigment eye spots on fossils show the origin of eye appeared from 540 million years ago, and since then it rapidly evolved towards various shapes and functionalities, ranging from the pinhole eyes, the compound eyes as insect eye to the sophisticated lens based colour eye as human eye. Even though we feel the perception of the scene as immediate actions, the processing of the vast visual information for the brain actually takes about 60 milliseconds, which means any consecutive events with smaller temporal distance can not be distinguished by the brain, thus this delay is generally treated as the approximate reference for the real-time standard for visual processing. Moreover, our brain subconsciously decides when and what part of the image to be fully processed, to further optimise the efficiency.

In the robotic world, all those subconscious actions in nature have to be explicitly programmed in artificial computer, which operates as the brain processing series of streaming images from lens-mounted colour camera. In the early research, image processing, such as structure from motion (SFM) and image filtering, was well considered as the off-line technique, due to the mismatch between the high computational requirement for real time image processing, and the limited computing capacity of the processors. Thus, at the time, sonar and laser were the favourite sensors for mobile robots to provide distance information about their surroundings, which leads to the beginning of the Simultaneous Localization And Mapping (SLAM) technique for robotic real-time online operation. However, due to the limitations of such sensors, i.e. limited range and low dimension information (generally one dimension), the techniques have to have strong assumptions over the operating artificial environment, such as small square space and negligible sensor orientation changes. However, in real world scenarios, none of the above assumption is true. Thus more powerful sensing solutions and more intelligent algorithms are needed to enable the real world robotic operation. Given the vast information gathered by the image sensor, it is certainly a key element to eliminate those assumptions.

In recent years, as the realisation of mass-producing good quality integrated cameras, their price and size has dropped significantly. Besides, over the last twenty years, the development of

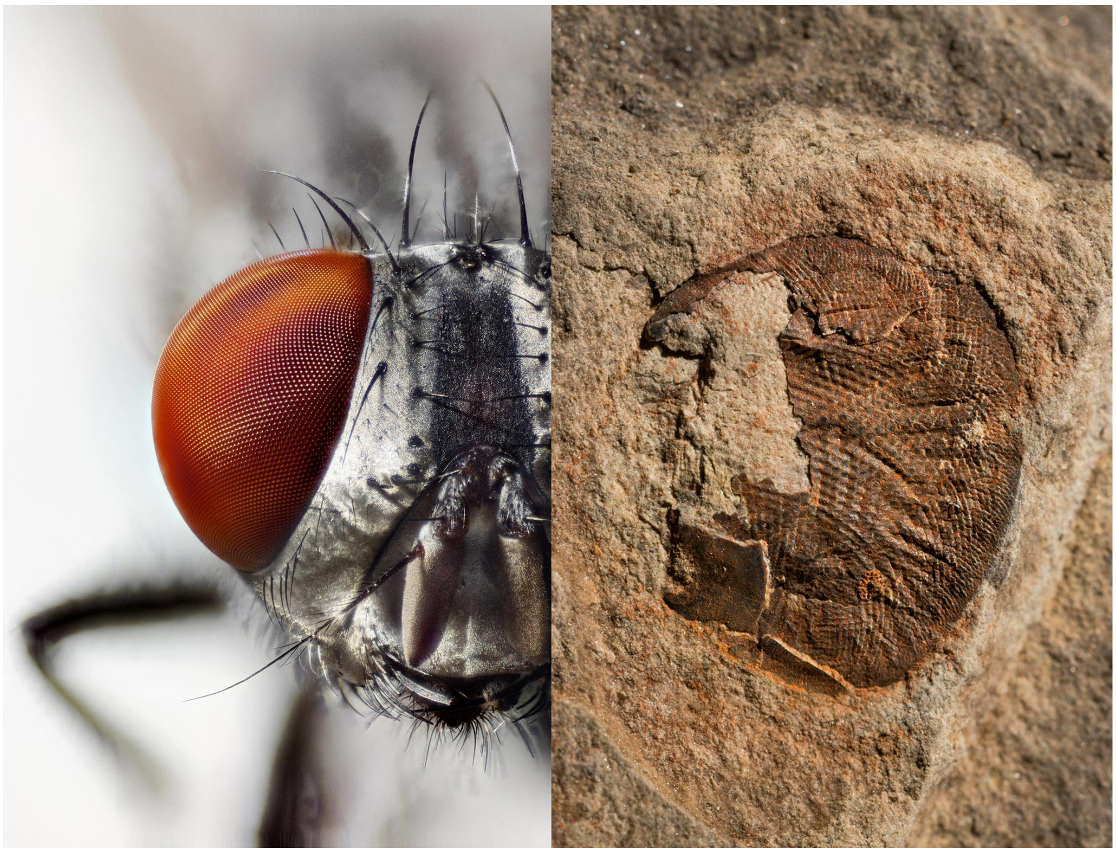


Figure 1.1: The modern insect compound eye in comparison with the arthropod eye 500 million years ago, showing similar level of complexity. Yong and Liittschwager (2016)

efficient algorithms for rapid image processing, mainly with the introduction of feature points extraction and optical flow techniques, dramatically reduces the computational requirements for real time image processing, and concurrently, thanks to the nano silicon manufacture techniques, system on chip technique and multicore parallel processing functionality, the computing speed of the same size processors grows exponentially over the years. As a result, the computing capacity of the small embedded processors, now starts to match the computation required by the rapid image processing algorithm capable of real-time operation. This shows the possibility to develop the computer vision system, which is small enough to be carried onboard a mobile robot and perceives the surroundings fast enough to be engaged in-line to mobile robotic operation. With all those in mind, one of the main goals in this thesis is to shrink the system in size, and use vision as the primary sense to localise the pose of a small robotic platform for its navigation in unstructured and unknown environments.

Certainly, for successful navigation and localization, vision is not the only sense in nature. For example bats developed sonar like sensors, birds can sense the earth magnetic field, and almost all animals can sense gravity field, acceleration and rotation through their vestibular system. When working with multiple sensors, nature finds its way to optimally fuse them. As an example, we can easily confuse our vestibular system, by spinning around our vertical axis. However if at the same time we fix one specific visual reference point, we can then subconsciously use this visual feedback to prevent the confusion. In this case, it can be considered as a system fusing fast dead-reckoning sensors for accelerations and angular velocities together with a slower sensor for visual correction to estimate a gravity-aligned pose. It is shown by nature again, that the optimal fusion of multiple sensor information can be very powerful for robotic localisation. Further with visual and inertial sensors as the main cues, it is sufficient for high performance localisation in three dimensional space. Therefore, we also seek to identify an appropriate sensor-fusion approach for state estimation.

1.1 Motivation

Remotely Piloted Aircraft (RPA) is one subset of Unmanned Aerial Vehicles (UAVs), which are mainly used to replace manned aircraft, for the purpose of reducing downside risk and rising confidence in mission success. In contrast to ground robots, the 3D manoeuvrability of UAVs in space, makes them unbounded by the terrain surface. This makes them ideal for reconnaissance, search and rescue, and coverage tasks.

Manually controlled RPAs generally require (a) highly skilled human pilot(s) performing continuous operation throughout the flight, while the autonomous and semi-autonomous RPAs are able to operate without the skilled human pilot. It greatly reduces the pilot training cost before the flight, and reduces the pilot working load in the flight. According to the author in RPAS Steering Group (2013), the safe integration of autonomous RPA into the European airspace will start in 2016, and the preparation for related regulation, technology research, and study on societal impact has already started.

The development of the Global Positioning System (GPS) has been the key to enabling the autonomous behaviours of RPA by providing the external localization information in global scope. Note that different providers use different terminologies (Russian service is called GLONASS, Chinese service is called BEIDOU and European service is called GALILEO), thus for the sake of simplicity, this thesis will use GPS to denote all the similar services. Given that the GPS by itself, is not very accurate ($\pm 2\ m$ Circular Error Probable (CEP, shown in Figure 1.2) with good Horizontal Dilution of Precision (HDOP), and $\pm 10\ m$ CEP with general HDOP) and not fast (generally 1 Hz update rate) enough to stabilise a RPA, thus, extensive research Abdelkrim et al. (2008); Yun et al. (2008), has brought the precision of position estimation up to decimetres accuracy by fusing the measurements from differential GPS (DGPS) and Inertial Navigation System (INS), called the INS/GPS approach. Nowadays, similar approaches drive most of the outdoor autonomous RPA systems.

However, there are multiple circumstances where GPS is not reliable or not available:

1. *Due to the natural weakness of the GPS signal, it can be easily blocked by any large object between the satellite and the receiver, which results in poor GPS position measurements*

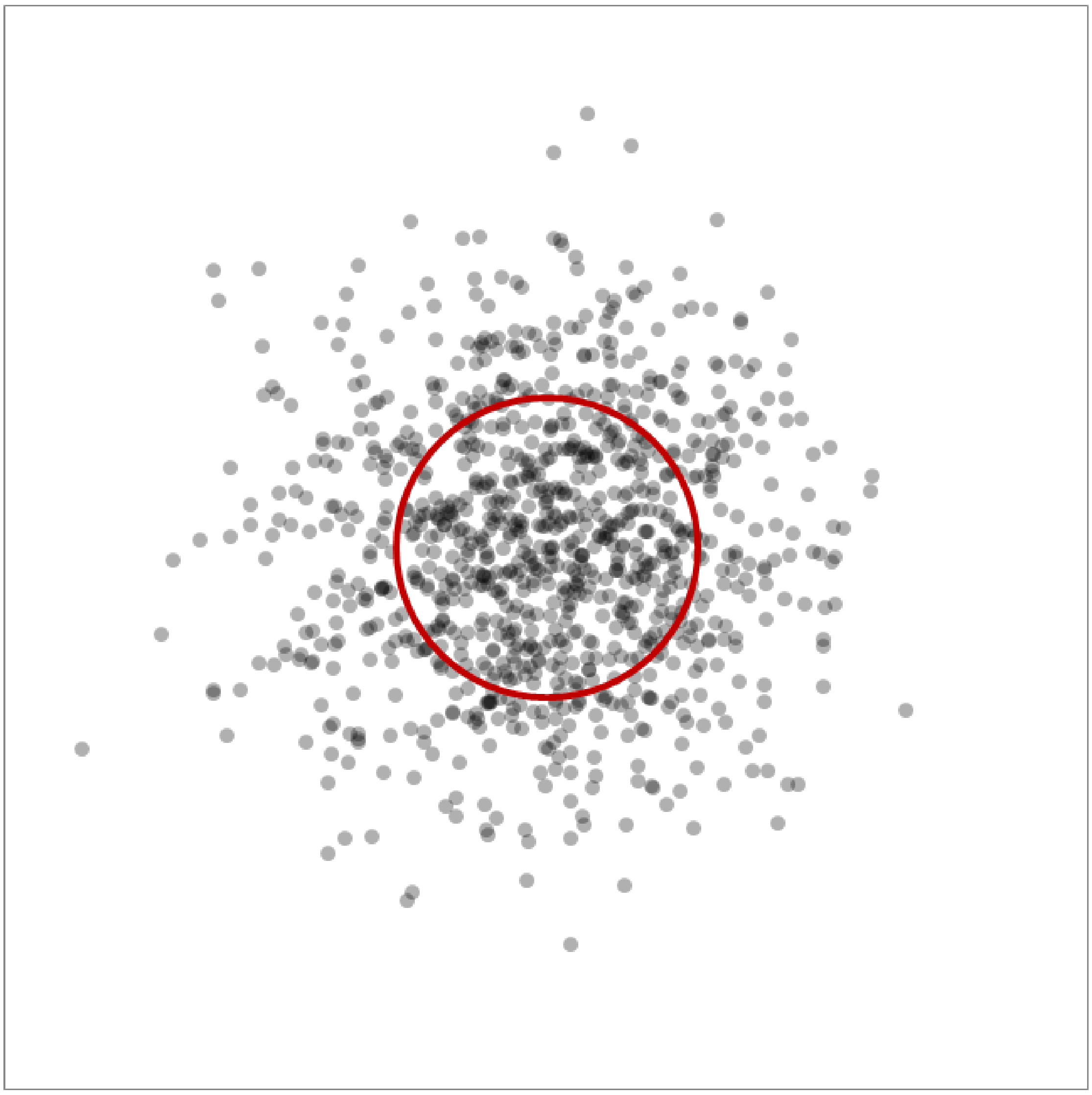


Figure 1.2: Circular error probable (CEP) shows the circular area which covers 50% of the sampling points. Zimmerman (2014)

or complete GPS signal loss. This is likely to happen indoor environments and even in outdoor urban area, where large buildings are close to each other.

2. *Given that there are a range of GPS jamming devices available in the market with affordable prices, it can be easily jammed by anyone. Thus it can happen in both civilian and military applications where GPS is purposely jammed by people.*

3. *Some careless RPA design can easily interfere with the GPS signal, such as placing the GPS antenna too close to a power wire or covering by a radio reflective material.*

4. *The GPS satellite failure happens occasionally, which results in direct failure in any GPS dependent devices.*

5. *GPS and similar services are only provided by a limited number of large organisations (US Airforce, Chinese Army etc.). In some large military activities, countries may decide to temporarily switch off their GPS service, which will also result in the corresponding GPS failure.*

Among the above, the indoor applications are the most general cases when GPS fails to operate, while there is high demand to remotely obtain information from inside a building, such as search and rescue, disaster response, safety and security, and military scenarios. Recently the UK Ministry of Defence (MOD) launched a series of activities to match the high demand in indoor applications: Unmanned Air Systems Capability Development Centre (UAS CDC) requested indoor flight demonstration; and Centre for Defence Enterprise (CDE) initialised the themed competition on the title 'What's inside that building?'.

Although autonomous RPA offers multiple advantages, and attracts a high demand for indoor operations, it leads to multiple challenges for robust operation. Firstly, for the autonomous flight in three dimensional space without GPS, the full 6 degree of freedom of the RPA has to be sufficiently measured without any physical contact to any fixed structure. Secondly, operations in the very constrained indoor space require the size of the RPA to be very small (less than 700 mm^3 occupying space for hovering), which significantly reduces its endurance, and payload capability for onboard sensors and computational power. Thirdly, the small size makes the RPA highly dynamic, which requires fast controller performance as well as measurement updates.

1.2 Contribution to Knowledge

This thesis investigates the methods for vision based navigation and control for a micro RPA. The main goal is to develop a complete micro quadrotor system, which uses an onboard camera and other available onboard sensors to achieve reliable and autonomous localisation and manoeuvre in indoor and close to ground outdoor environments, without the use of GPS or similar external aid. This goal requires intensive research in a multitude of areas. This thesis presents the core contributions, listed as following:

1. *Development and evaluation of a novel, loosely coupled, visual-inertial sensor fusion algorithm, based on a monocular SLAM, an accelerometer and a gyroscope, which requires so little computation that it is suitable for high speed 6 DOF state estimation on low power embedded computer.*

2. *Development of a complete micro (*250 mm *motor-to-motor size) quadrotor system capable of navigation in GPS-denied environments, based on onboard sensors and computation only.*

3. *Development and evaluation of a novel model based control algorithm for micro quadrotor autopilot system.*

4. *Design and Implementation of a novel low cost high performance quadrotor autopilot hardware based on commercial-off-the-shelf (COTS) components.*
5. *Design and Implementation of a novel hand-launching method for micro UAV, for intuitive operation.*
6. *Design and Implementation of a novel active planning algorithm, which leads to safer vision-based flight in low contrast environments.*
7. *Design and Implementation of an electrical tether system providing power to the micro UAV from offboard power sources.*

The detailed justifications and comparisons of the listed contributions are shown in the following subsections. They can be generalised as visual-inertial fusion framework, platform and control, active planning for safe vision operation, and tether power system for small UAV.

1.2.1 Novelty of the visual-inertial fusion framework

The following explains the novelty of the contribution (1) listed above. The implementation detail is shown in Chapter 4.

The combination of visual and inertial sensors has been shown to be viable, and the significant performance improvement over a single sensor system has attracted many researchers into the field after the success of SFly project in Weiss and Siegwart (2011), which enabled the worlds first autonomous unmanned aerial vehicle (UAV) in GPS-denied environments. Blösch et al. (2010)

In the past five years, many prominent research institutions began to develop advanced monocular visual-based simultaneous localization and mapping (mSLAM) algorithms based on structure from motion (SFM) theory Engel et al. (2014, 2013); Forster et al. (2014); Klein and Murray (2007); Montemerlo et al. (2003); Newcombe et al. (2011b); Pizzoli et al. (2014); Roussillon et al. (2011); Vogiatzis and Hernández (2011), which are suitable to modern onboard embedded computers. Moreover, the visual scale problem, which was the main challenge of involving monocular vision into the control loop, has been addressed by fusing onboard inertial measurements (accelerometer and gyroscope), called the visual-inertial navigation system (VINS) Dunkley et al. (2014); Jones and Soatto (2011); Kelly and Sukhatme (2009, 2016); Li and Mourikis (2013); Lobo and Dias (2003); Lynen et al. (2013); Shen et al. (2013b); Weiss et al. (2012a).

Almost all of the visual-inertial fusion algorithms, to the best of my knowledge, rely on nonlinear Kalman filter techniques (extended Kalman filter, unscented Kalman filter, etc.) to process both the orientation and the position measurement in the same process. This type of fusion has the following drawbacks. Firstly, it results in a large state vector (usually more than 20 dimensions) and a complex nonlinear system model, which causes high computational cost and relying on theoretical approximation to linearise the model. Secondly, it suffers from the lack of robustness in operation both in terms of filter initialisation and in terms of fail-safe implementation for aerial robotic applications, since the orientation estimation can be easily affected by the filter illness.

However, recent advances in computationally efficient inertial measurement unit (IMU) orientation estimation, Madgwick et al. (2011a), shows a competitive accuracy against Kalman-based algorithms by utilising optimisation based methods. Thus, in this thesis, a computationally efficient visual-inertial fusion algorithm is proposed by separating the orientation and the position fusion processes. This filter maintains the same level of accuracy against nonlinear Kalman filter approach, while significantly reduced the computational cost for real-time operation, as well as improved robustness due to that the orientation is separated from the position estimation. The algorithm is designed to perform a six degree of freedom state estimation, based on a gyroscope, an accelerometer and a mSLAM measurement. It also recovers the visual scale for the mSLAM.

1.2.2 Novelty of the platform and control

The following explains the novelty of the contributions (2–5) listed above. The implementation detail is shown in Chapter 3 and Chapter 5.

In order to make a UAV system practically functional in very constrained indoor space, the platform size becomes very important. Given that the UK typical doorway is $762\ mm$ wide, which limits the platform motor to motor distance up to $300\ mm$ with maximum $7\ inch$ propellers, to allow gap of minimum $100\ mm$ for safety margin. Given that the payload capacity (about $150\ g$) and endurance (about $15\ min$) is very limited for the platforms of this size, packing all the necessary computation and sensors onboard is almost impossible without the combination of software and hardware optimisation. This constraint significantly limits the available platforms to choose from.

In recent years, since the mass-production of micro-electro-mechanical systems (MEMS) based inertial measurement units (IMUs), their size and price have been reduced significantly. This leads to extensive research on mini quadrotor development, such as Elsamanty et al. (2013); Fernando et al. (2013); Jeong and Jung (2013); Mahony et al. (2012). Nowadays, the $250\ mm$ size platform is popular in first person view (FPV) racing communities, where the systems are designed for maximum controllability and simplicity for pilot to show off their skills. For them, only low computation is required for basic attitude stabilisation. However, none of the above considers vision feedback in the quadrotor control loop. On the contrary, the world's top computer science research groups are keen on advancing their high level computer vision algorithms, based on third party platforms such as AR-drone from Parrot or Hummingbird quadrotor from Ascending Technologies described in Kushleyev et al. (2013). Recent work includes Engel et al. (2012); Faessler et al. (2015); Huang and Bachrach (2011); Sa et al. (2013); Shen et al. (2013a,b); Weiss et al. (2012a). However, these solutions generally require very close collaboration with the platform manufacturers to access their source code or reverse-engineer from first principle. These problems stop most ordinary researchers from accessing the platform outside the communities.

Therefore, an all-in-one platform testbed designed specifically for vision based control is highly desirable. The testbed also needs to be highly integrated to keep the size within the constraint, and highly accessible both for hardware and software to allow all levels of researchers to explore its full potential. Hence, in this thesis, a $250\ mm$ size platform is developed from scratch with an industry level monocular camera and an onboard computer with standard quadcore smartphone processor. The onboard electronics were designed for maximum integration of commercial-off-the-shelf (COTS) components, which also allows maximum accessibility. The autopilot software was developed from scratch in the Arduino environment, based on the state-of-the-art control algorithm by Mahony et al. (2012), while taking into account the physical dynamics of the individual rotor and the whole UAV platform. Besides, due to the all-in-one nature, the autopilot was modelled and tuned specifically for this platform, which allows high accuracy control performance. The original mechanical design ensures maximum rigidity and the optimal component layout. This testbed allows all the onboard computer vision, sensor fusion and control algorithm to be developed and tested.

Additionally, a physical button was introduced onto the vehicle body. It allows the quadrotor to be launched in the place-and-hover manner for rapid deployment from hand or without a flat takeoff area. While in comparison against the throwing-launch of a quadrotor in Faessler et al. (2015), it is not only safer, but also allow the user to still have precise control on the quadrotor initial position.

1.2.3 Novelty of active planning for safe vision operation

The following explains the novelty of the contributions (6) listed above. The implementation detail is shown in Chapter 6.

Recent advances in onboard monocular vision-based simultaneous localisation and mapping (mSLAM) for unmanned aerial vehicles (UAVs) enables their autonomous navigation in GPS-denied environments. However, unlike the GPS-based navigation, the accuracy of the mSLAM-based navigation performance highly depends on the motion of the UAV and the scene within the camera's field of view. Therefore, instead of passively performing mSLAM while the UAV travels to the destination (target waypoint), we integrate the perception requirements of mSLAM into the navigation control of the UAV. Specifically, we present a novel active one-step-ahead planning algorithm which dynamically generates the optimal next action command in real-time for an autonomous quadrotor based on its current mSLAM observations, so that it not only approaches the target waypoint, but also maximises the localisation accuracy for the onboard mSLAM algorithm. Especially in scenarios similar to the one shown in Figure 6.1 in Chapter 6, where large area with low contrast occupies the significant portion of the direct path for UAV to approach the target position (indicated by yellow path), with the active planning algorithm, the UAV will choose the green path to avoid the low contrast area while approaching the target area. This active planning algorithm serves as the complementary to the state-of-the-art passive mSLAM algorithm, and aiming to improve the operational safety and robustness of the real world GPS-denied scenarios.

Therefore, to address this problem, the active SLAM planning algorithm serves as the complementary to the existing passive mSLAM algorithm. The objective is to actively command the vehicle movement to minimise the localisation uncertainty while performing the high level tasks, in a computationally efficient manner. In particular, given the next desired waypoint, a novel active planning algorithm was developed to choose the optimal UAV trajectory for an autonomous quadrotor, to maximise the accuracy of the onboard SLAM algorithm, while approaching the waypoint. More specifically, the active planner aims to command the UAV, so that it:

1. Tries to approach the next way point as close as possible;
2. Tries to minimise the localisation uncertainty of the mSLAM algorithm.

To the best of my knowledge, a similar robust active planner for monocular localisation on UAV has only been implemented by Mostegel et al. (2014). Based on a forward facing camera and PTAM (an open-source mSLAM algorithm by Klein and Murray (2007)), he measured the localization quality by the weighted sum of the number of mapped points in the frame, due to the lack of uncertainty measurement from PTAM. Furthermore, based on the measured localization quality, a threshold based planner was developed to switch among localization improvement mode, new point generation mode or way point following mode. although the system is capable of handling the challenging target command, like pure yaw rotation, it considers neither the optimal tradeoff between the two objectives, nor the physical dynamics of the quadrotor.

Different from the above approach, its aim is to develop a probability theory driven optimal planner, with a downward facing camera, which naturally avoids the pure yaw rotation problem. Therefore, the following contributions was made in part of this PhD work to the state of the art:

1. It presents the first theoretically optimal cost function for the two objectives from the probability theory perspective;
2. It presents the first approach to predict the localisation accuracy for any camera view point, based on the extended information filter (EIF);
3. The first active planning algorithm was designed for downward facing camera, which takes the physical dynamic constraints of the quadrotor into consideration.

1.2.4 Novelty of tether power system for small UAV

The following explains the novelty of the contributions (7) listed above. The implementation detail is shown in Chapter 7.

For the small-scale stable hovering platforms (such as standard multirotors) the available sources of power are limited to the on-board electric storage cells (batteries). There exists a range of possible crucial applications, where long term operation of such systems is required, such as precise inspection of large industrial structures, and tasks involving physical manipulation of the environments. Those applications can be achieved by UAVs, at the cost of significant power requirements. Also, there exist tasks which require close collaboration of aerial and ground robots, but inherently take a significant amount of time to accomplish, such as the aerial detection of land mines using radar-bullets and their safe removal using robotic agents. Within those tasks, agile flight and aggressive manoeuvre abilities might not be the key prerequisite, rather than stable operation while hovering at high altitudes.

There is little research currently on the tethered UAV, recent tethered UAV research mainly focuses on the control aspect of such system Abdelkrim et al. (2008); Zikou et al. (2015). There are also some commercial companies developing system tethered UAVs, such as Elistair, PARC from Cyphy work, Cardinal security, ECA group, while their systems are only suitable for large scale multirotors. Although, a small size tether powered UAV was developed by Fotokite, which is rather an expensive system.

The developed small tether system focuses on implementing a cost effective method for remotely powering of small-scale hovering UAVs via a ground-to-air Power-over-Tether link which carries power to the vehicle.

1.3 Publications and Public Engagements

1.3.1 Journal publications

1. Liu, C., Prior, S. D., and Scanlan, J. P. Design and Implementation of a Low Cost Mini Quadrotor for Vision Based Manoeuvers in GPS Denied Environments. *Unmanned Systems*, pages 4(3):185–196, DOI: 10.1142/S2301385016500059 (Liu et al., 2016a)

2. Liu, C., Prior, S. D., Teacy, W. L., and Warner, M. Computationally efficient visual- inertial sensor fusion for Global Positioning System-denied navigation on a small quadrotor. *Advances in Mechanical Engineering*, 8(3):1–11, DOI: 10.1177/1687814016640996 (Liu et al., 2016b)

3. Liu, C., Nash, J., and Prior, S. D. A Low-Cost Vision-Based Unmanned Aerial System for Extremely Low-Light GPS-Denied Navigation and Thermal Imaging. *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 9(10):1740–1747. URL: `http://waset.org/publications/10002835` (Liu et al., 2015)

4. Submitted to the *Journal of Intelligent and Robotic Systems*, in the title The Practical Implementation of an Autonomous Micro Quadrotor in GPS-denied Environments, on 3rd September, 2016. Author list: Liu, C, Prior, S.D., and Scanlan, J.P.

1.3.2 Conference publication

1. Liu, C. and Prior, S. D. Design and Implementation of a Mini Quadrotor Control System in GPS Denied Environments. *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015 - Conference Proceedings*, pages 462–469. DOI: 10.1109/ICUAS.2015.7152324 (Liu and Prior, 2015c)
2. Liu, C. and Prior, S. D. Computationally efficient visual-inertial sensor fusion for GPS-denied navigation on a small quadrotor. *2015 International Conference on Innovation, Communication and Engineering*. (Liu and Prior, 2015b)
3. Liu, C. and Prior, S. D. A low-cost vision-based unmanned aerial system for extremely low-light GPS-denied navigation and thermal imaging. *2015 17th International Conference on Intelligent Unmanned Systems*. (Liu and Prior, 2015a)
4. Presentation in *2015 Next Gen Drones conference* in Washington DC. US, with the title: Design and Implementation of an Open Source Small Quadcopter for GPS-Denied Environments. 23-24th June, 2015

1.3.3 Other public engagements

1. Attended *2016 Farnborough International Airshow* as exhibitor. 11-17th July, 2016.
2. Successful indoor flight demonstration to *Unmanned Air Systems Capability Development Centre (UAS CDC)* in Canary Wharf, London. 11th June, 2015.
3. Presented research work in *2015 Engineering@Southampton Autonomous Systems Showcase* at the Royal Academy of Engineering (RAe). 9th June, 2015.

1.4 Thesis Overview

Chapter 2 presents the literature review work conducted. It summarises all UAS navigation solutions in GPS-denied environment, derives the advantages of camera based methods, and then investigates into the monocular vision based solutions and visual inertial navigations.

Chapter 3 presents the design, implementation and testing work for the testing platform in terms of electronic and mechanical hardware, and high performance autopilot control algorithm for quadrotor.

Chapter 4 presents the design, implementation and testing work for the visual-inertial onboard sensor fusion algorithm developed for the vehicle six degree of freedom state estimation and visual scale recovery, aiming for low computation.

Chapter 5 presents the implementation detail of the final power-on-and-go platform, which is equipped with the controller from Chapter 3 and pose estimator from Chapter 4, as well as the final test result evaluating the performance of the overall system. Besides, it also described a novel hand-launching method implemented for the ease of consumer level usage.

Chapter 6 shows the mathematical derivation of a probability theory based active planning algorithm to avoid featureless environments by commanding the vehicle movement.

Chapter 7 presents the engineering and implementation detail of a tether powering solution for micro UAVs, to shift the power source from onboard UAV to the ground. This boost the endurance of micro UAVs to indefinite flight time.

Chapter 8 concludes and lists the publications and related project activities and future work.

Chapter 2

Literature Review

Over the last decades, there has been rich literature accumulated for localising and navigating mobile robots based on various methods. This chapter draws an overview of the state-of-the-art navigation solutions as an alternative to GPS. It starts with the review of different popular sensors used for the GPS-denied navigation research, with the emphasis on the sensors which are onboard the vehicle, which are called onboard sensors. It includes radio detection and ranging (RADAR) systems, Light Detection And Ranging (LIDAR) systems and camera(s) based systems, and also learned from the driverless car technologies. By comparing different sensors, it proves the advantages of the single camera based solution over other sensors. Then, it focus into the literature review specifically on monocular visual navigation and visual inertial navigation, and poses the visual scale problem in monocular vision and how it can possibly be recovered by fusing the inertial sensor. Finally, together with a brief review on platform and control theory, the literature review finishes with a summary of research methodology towards the following chapters.

2.1 Navigation Solutions in GPS-denied Environments

This section overviews the state-of-the-art navigation solutions as an alternative to GPS, using different sensors, regardless of application context. The autonomous car industry is also reviewed, due to their similar problems. Moreover, towards the end of this section, the comparison between different sensor is summarised in a single chart to highlight the advantages of the monocular vision based solutions. This forms the logic, which leads to the focus in the next section on the detailed and more specific review on monocular visual navigation in the next section.

2.1.1 External offboard localization methods

Recent research seeks alternative external localization systems. Some of them use external camera(s) Altug (2005); Park et al. (2005) or stereo camera(s) Klose et al. (2010). Some of them have recently gained promising results by applying the Vicon^{TM} (motion capture system) How et al. (2008); Michael et al. (2010), shown in Figure 2.1. However, those external localization approaches are not of our interest, because in those cases, the vehicle still depends on the assistance of external infrastructure. However, what we are looking for is a fully self-contained and extendable system, which means all sensors would be onboard, self-contained and immune from interference.

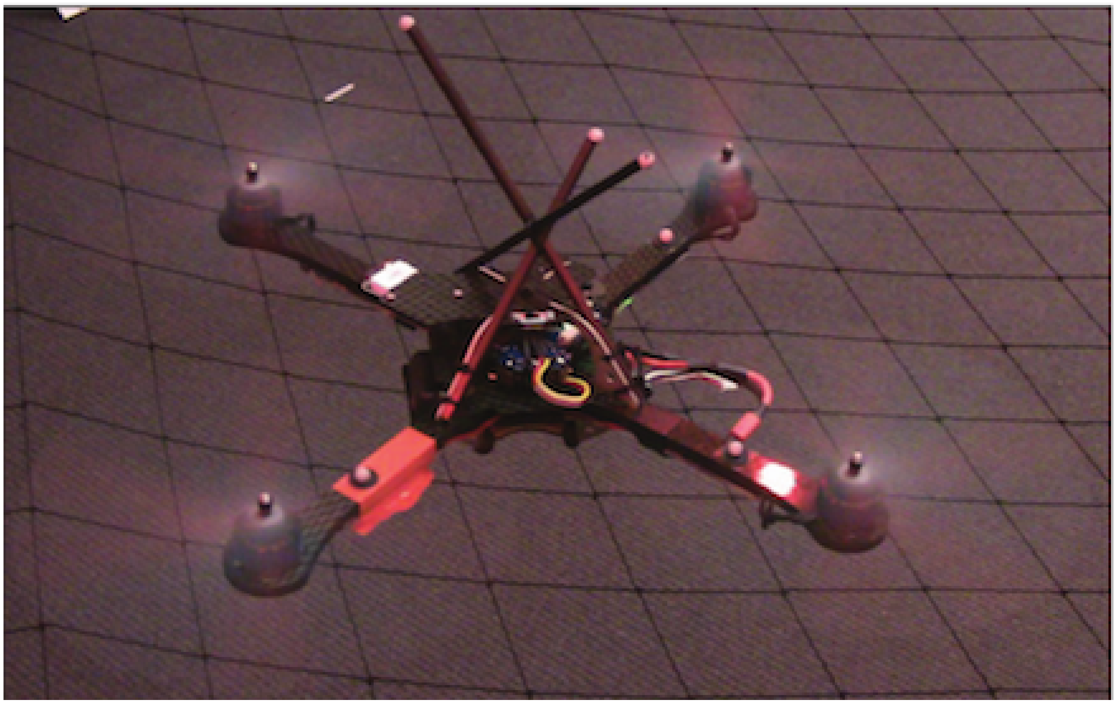


Figure 2.1: A quadrotor platform with Vicon markers Michael et al. (2010).

2.1.2 Onboard RADAR based methods

RADAR named after radio detection and ranging. The antenna transmits the radio wave or microwave, and by detecting the wave bounced back by object, it determines the range, altitude, direction, or speed of the object.

Automotive Micro RADAR is a group of low cost (around £700) RADAR systems mounted on a car, which are used to detect obstacles. These have been used on the autonomous cars, discussed in Section 2.1.5. They have the advantage to reliably operate in any weather with relatively long range. The detailed products are shown in Appendix A.2. Their general weight is 300 g.

Synthetic Aperture Radar (SAR) is a form of RADAR that uses the relative motion between its antenna and the target to obtain refined spatial information in the detected area. It is generally capable of obtaining very fine ($0.3\,\mathrm{m}$ resolution) at a range of over $1\,\mathrm{km}$, under any weather condition. However, the size of the sensor limits it to large scale RPA platforms. The smallest application, to the best of my knowledge, is over $0.5\,\mathrm{kg}$, such as IMSAR NanoSAR.[1] Please see Appendix A.1 for more SAR and specifications.

Because of its long range, robustness and resistance to any weather condition, the RADAR system is widely used in autonomous driving and aerospace industry. With the continuously improved integration level, its size and mass can be dropped in the future. While for their current level of pricing and weight, they are generally suitable for obstacle sensing or acting as an altimeter for medium size drone.

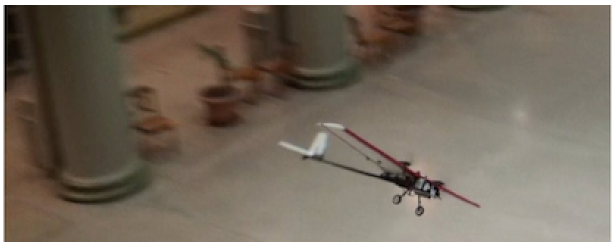
2.1.3 Onboard LIDAR based methods

LIDAR is the short for LIght Detection And Ranging. Two kinds of LIDAR systems are described in below.

Laser Scanner is a remote sensing technique, which illuminates the target by laser and processes the reflected light to obtain distance information. Recently, in Massachusetts Institute of Technology (MIT), an impressively accurate state estimation for aggressive flight is achieved by only using a laser scanner and IMU data,[2] Bry et al. (2012) as shown in Figure 2.2a. A picture of the opened vehicle body.[3] is shown in Figure 2.2b A laser scanner has the advantage of obtaining the distance information and operate well in textureless environments, however, limited range and field-of-view (normally 1D or 2D), and high mass, price and power consumption make it not optimal. The detailed products are shown in Appendix A.4.

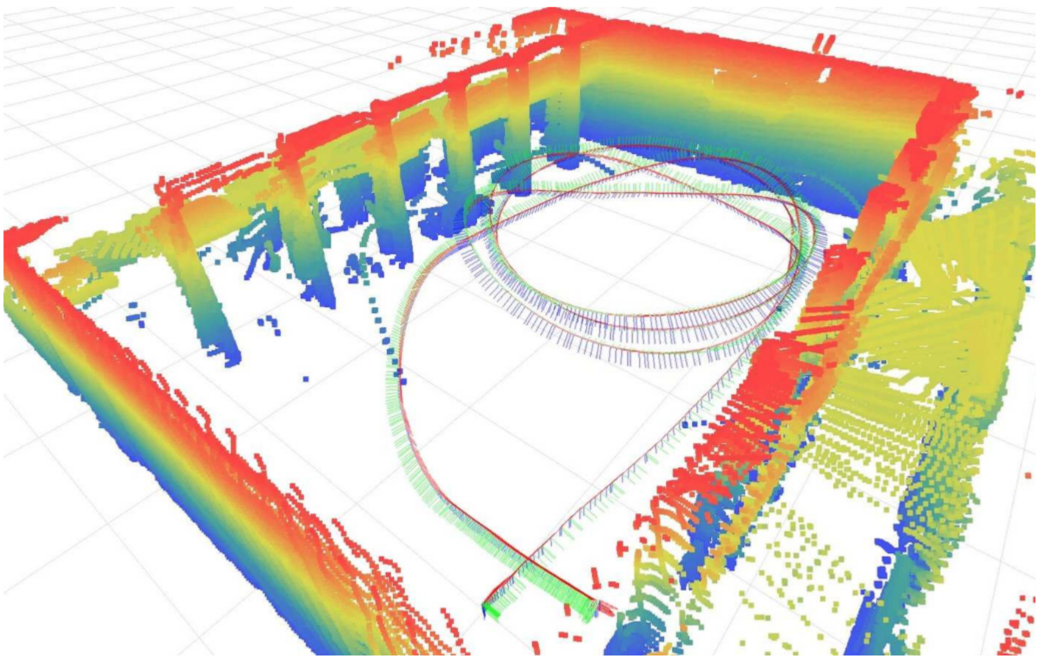
Flash LIDAR with the development of Time-of-flight (TOF or Flash LIDAR) camera, which is able to obtain distance information of every pixel in images and output as 3D point cloud, such as Microsoft Kinect[4] and PrimeSense, which is a subsidiary of Apple, and powering the Structure sensor[5], Please see Appendix A.3 for more Flash LIDAR sensors and specifications. MIT has successfully used the Kinect for navigation of a quadrotor[6] (shown in Figure 2.3), where a RGB-D SLAM algorithm is developed Huang and Bachrach (2011). Many researchers are keen to use it for 3D scanning to reconstruct dense model of an object Newcombe et al. (2011a), which is theoretically similar with SLAM, but designed to

[1]`http://www.imsar.com/pages/products.php?name=nanosar`
[2]`http://www.youtube.com/watch?v=VUeKKvKEvYI`
[3]`http://www.gizmag.com/mit-autonomous-indoor-uav/23686/pictures#5`
[4]`https://msdn.microsoft.com/en-us/library/jj131033.aspx`
[5]`http://structure.io`
[6]`http://www.youtube.com/watch?v=aiNX-vpDhMo`



(a) Flight view Bry et al. (2012).

(b) Vehicle body view.



(c) 3D map constructed.

Figure 2.2: MIT aggressive state estimation using laser scanner Bry et al. (2012).

be used in a small scene without considering error drifting (can be overcome by bundle adjustment and loop closure Strasdat et al. (2011)). Software tool libraries are available as open source software, such as Point Cloud Library (PCL)Rusu and Cousins (2011) and OpenNI.[7] Popular algorithms are developed to map the point cloud, such as Iterative Closest Points (ICP) algorithm. Some interesting results have been obtained on real-time dynamic 3D reconstruction Keller et al. (2013), which is capable of dynamic segmentation of moving and static objects, and dynamic scene update. However, similar to other LIDAR devices, the detection range is generally limited to several metres in consumer-end applications. Also those dense based SLAM Algorithm generally computationally expensive, and rely on massively parallel computation in GPU. With the more recent advance in commercial TOF cameras since 2015, . For example Intel realsense technology, which enables a range of smaller size TOF cameras optimized for robotic applications, such as Realsense R200 and ZR300.[8] Additionally, Intel also provides processor and software

[7]`http://www.openni.ru`
[8]`https://click.intel.com/realsense.html`

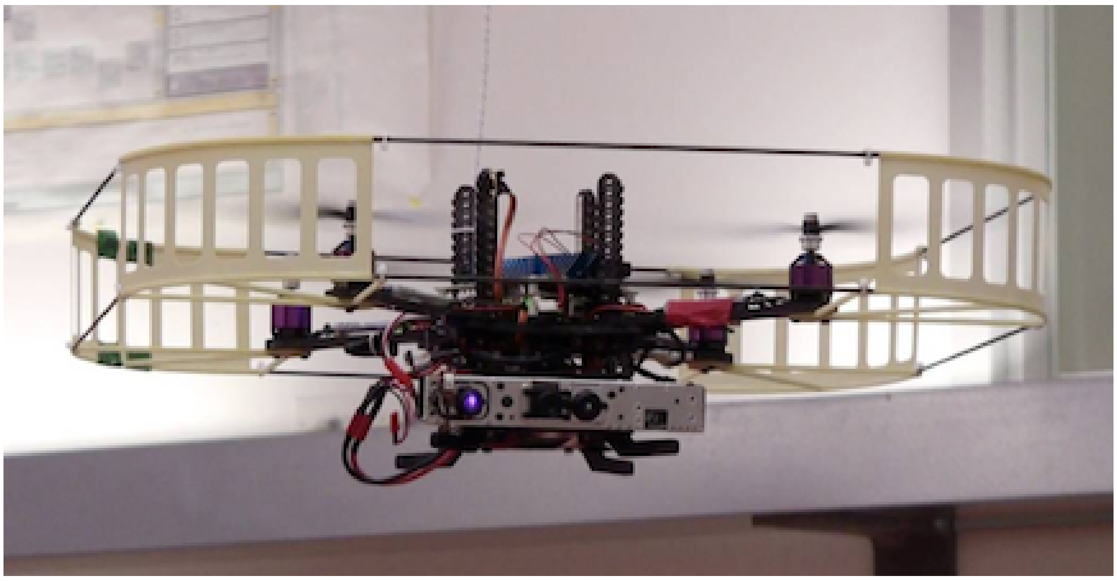


Figure 2.3: MIT Microsoft Kinect based airborne platform Huang and Bachrach (2011).

support for the system. similar systems can potentially form a lighter solution, and can also allow the easier integration of obstacle avoidance capabilities.

2.1.4 Onboard camera based methods

Camera as a passive sensor, generally has the advantage of low power consumption, comparing with active sensors such as LIDAR and RADAR, and it is capable of capturing very rich information, with adjustable field of view. However, obtaining the depth information in the scene is nontrivial with camera.

Stereo Camera is a pair of cameras with a fixed and known baseline (baseline refers to the relative position between the two cameras). Similar with animal eyes, triangulation is done in each frame to directly obtain a depth information in the scene. It outputs a point cloud, which is similar with Flash LIDAR. A pair of front-facing stereo camera has been implemented on a UAV (shown in Figure 2.6b), with the help of a downward-facing monocular camera computing optical flow, to achieve an autonomous navigation at the ASL research lab (ETHZ) Fraundorfer et al. (2012). Besides, stereo camera has also been used cooperating with a laser scanner for UAV autonomous flight in MIT Achtelik et al. (2009). Nikolic et al. (2014) have also shown a stereo camera designed specifically for robotic navigation. However, because of the fixed baseline, when the scene exceeds the maximum distance, it will lose the ability to recover the depth in the scene. Also, doing triangulation for each frame is computationally intensive for onboard computation. Moreover, it is not reliable in textureless or repetitive scenes. This is our potential interest for further development.

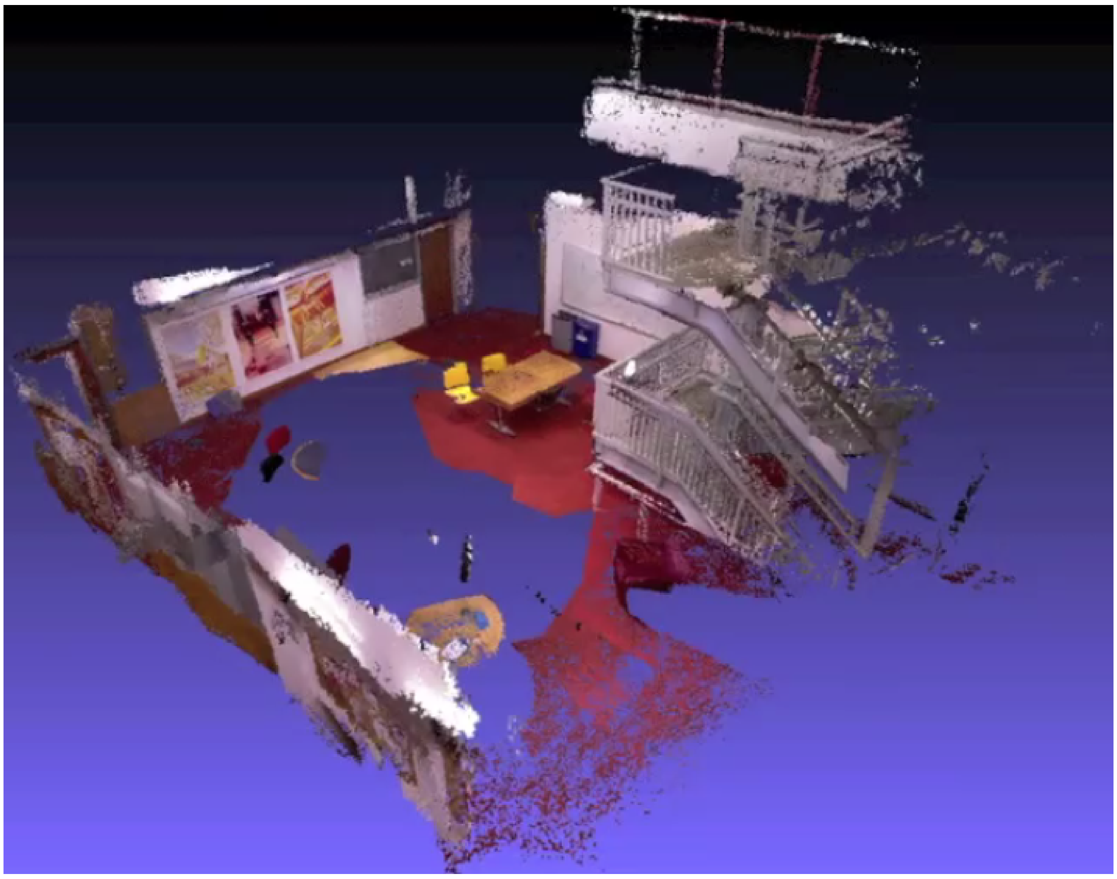


Figure 2.4: MIT Microsoft Kinect 3D reconstruction Huang and Bachrach (2011).

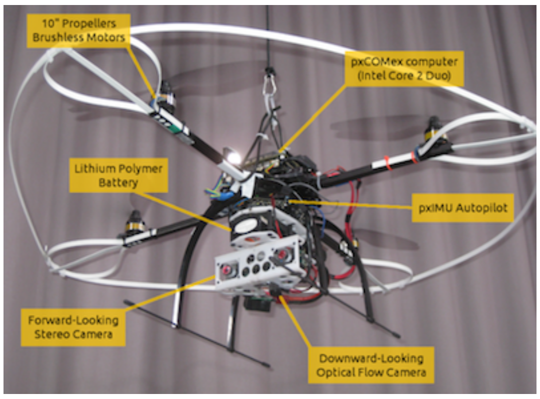
Monocular Camera in contrast, further reduces the mass and size required, but the depth information is completely lost in each frame. While the Structure From Motion (SFM) technique (discussed in Section 2.2.1.1) addresses this problem by triangulating a series of frames, which are captured by the same camera, but from different 3D positions. Therefore this technique has the advantage of flexible baseline, while because of this flexible but unknown baseline, it can only recover the depth upto scale. The scale can never be recovered without an external helper device. Therefore, to recover the scale for onboard monocular camera, many ideas have been proposed: an ultrasonic altitude sensor is used to recover the depth of a third party monocular vision algorithm Klein and Murray (2007), in the work[9] in Engel et al. (2012), which is implemented on a Parrot AR-Drone[10] to achieve the autonomous waypoint following in small area, as shown in Figure 2.6a; or Inertial Measurement Unit (IMU) data is used Nützi et al. (2011), and it has been successfully implemented by Weiss in Weiss (2012) to accurately navigate an autonomous quadcoptor in unknown environment, only using one downward facing camera and an IMU (shown in Figure 2.5).

[9]`http://vision.in.tum.de/data/software/tum_ardrone` (the published source code can also be found in this link)

[10]`http://ardrone2.parrot.com`

Figure 2.5: ETHZ monocular camera navigation Weiss (2012).

(a) TUM monocular camera based navigation Engel et al. (2012).



(b) ETHZ stereo camera based navigation Fraundorfer et al. (2012).

Figure 2.6: Monocular camera based navigation.

Optical Flow Sensors based on very low resolution cameras have been developed in recent years. Optical mouse was using the same sensor for human interaction with computer.[11] The same sensor was also used on modern UAV as a speed sensor, and resulting sensor achieves a very high update rate (around 6300 Hz) for fast flight.[12] The recent work by Gageik et al. (2013) achieved autonomous flight using this type of optical flow sensor. However, those sensors require bright lighting condition to operate, thus they are not suitable for indoor or low light outdoor environments. In order to avoid this limitation, Honegger et al. (2013) published a robust velocity and position estimator at high speed (250 Hz) based on optical flow, computed from a generic camera sensor, which is very sensitive to light. A detailed table about optical flow cameras can be found in Appendix

[11] `https://people.ece.cornell.edu/land/courses/ece4760/FinalProjects/s2009/ncr6_wjw27/ncr6_wjw27/docs/adns_3080.pdf`

[12] `http://diydrones.com/profiles/blogs/adns3080-optical-flow-sensor-now-available-in-the-d`

A.5. By assuming relatively flat surface, the optical flow camera is able to directly measure the horizontal velocity of the UAV, when combined with ultrasonic distance sensor and IMU.

Visual landmark based Methods Some researchers have explored localisation methods by using external landmarks, where prior knowledge about the landmark is used to derive relative 3D position between vehicle and the landmark. This method significantly reduces the computation, but requires direct visual contact between vehicle and the landmark, which limits its operation area.

2.1.5 Autonomous car

In recent years, autonomous car has be well investigated and developed. Very good results have been achieved to enable relatively reliable driver-less operation on the real road. Even though, it is a slight different problem to navigate a ground vehicle in the road, it still shows some guideline for a safe navigation system.

Google Autonomous Vehicle is recently stated to drive better than the human driver. Guizzo explained how a Google Autonomous Vehicle works.[13] A Velodyne 64-beam laser[14] is used as the primary sensor for mapping and localization, whose operation is illustrated in Figure 2.8. Moreover, as shown in Figure 2.7, four radars are mounted on the front and rear bumpers for long distance detection to deal with fast traffic on freeways; a camera is positioned near the rear-view mirror to detect traffic lights; and a GPS, inertial measurement unit, and wheel encoder, that determine the vehicle's location and keep track of its movements. A picture of the new google car released in late May 2014 is show in Figure 2.9.

VisLab International Autonomous Challenge (VIAC) is a recent autonomous vehicle test hold by VisLab (University of Parma), which is carried out from Parma to Shanghai between July and October 2010 Bertozzi et al. (2013); Broggi et al. (2012). As an extended test for heterogeneous road condition with real traffic over BRAiVE (VisLabs main testing prototype for well structured and predictable environments Grisleri and Fedriga (2010)), the sensor suite is based on the BRAiVE experience, except for an additional laser scanner, to frame the ground during off-road driving. Specifically, Stereo cameras and laser scanners are the primary sensors. The configuration is shown in Figure 2.10 and 2.11. As shown, seven cameras are installed on the vehicle (five forward and two backwards). The five forward facing cameras includes a pair of stereo cameras and a

[13]http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

[14]http://velodynelidar.com/lidar/hdlproducts/hdl64e.aspx

[15]http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

[16]http://recode.net/2014/05/27/googles-new-self-driving-car-ditches-the-steering-wheel/

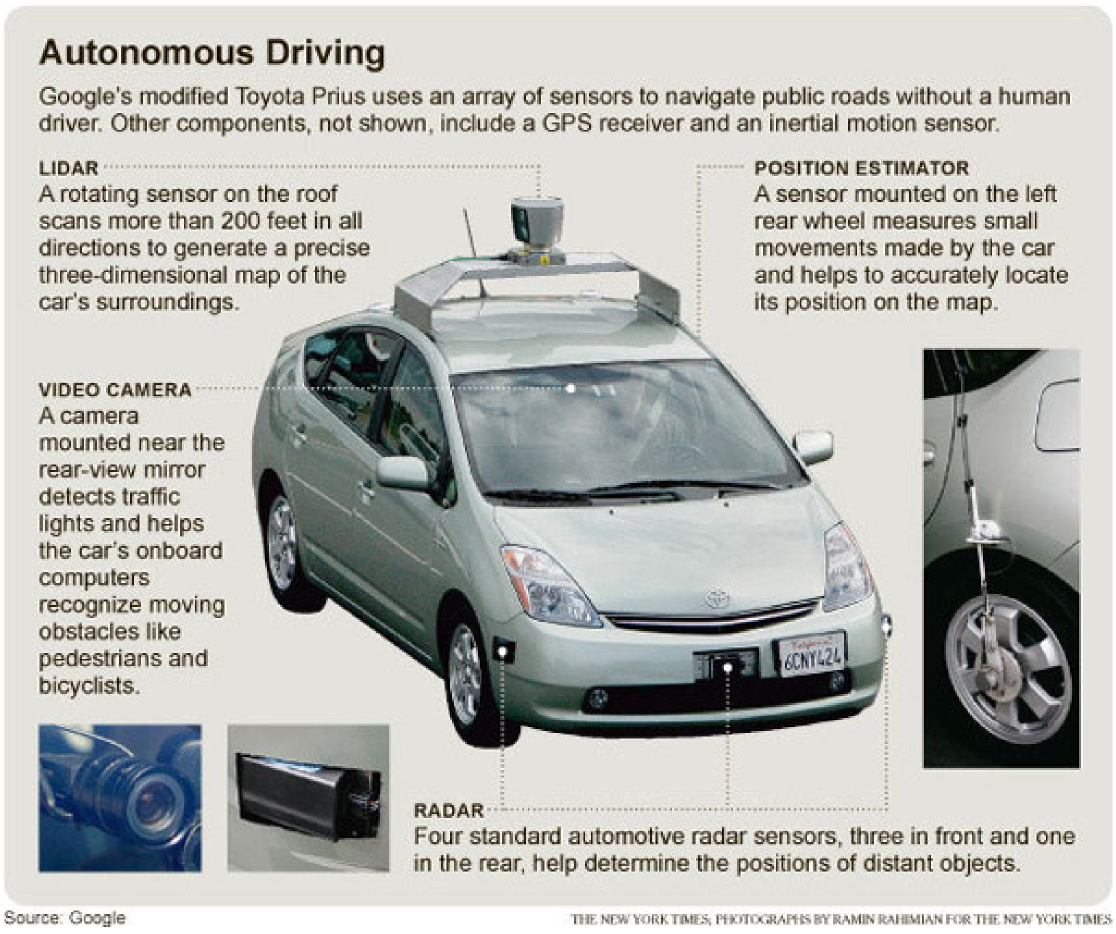

Autonomous Driving
Google's modified Toyota Prius uses an array of sensors to navigate public roads without a human driver. Other components, not shown, include a GPS receiver and an inertial motion sensor.
LIDAR
A rotating sensor on the roof scans more than 200 feet in all directions to generate a precise three-dimensional map of the car's surroundings.
POSITION ESTIMATOR
A sensor mounted on the left rear wheel measures small movements made by the car and helps to accurately locate its position on the map.
VIDEO CAMERA
A camera mounted near the rear-view mirror detects traffic lights and helps the car's onboard computers recognize moving obstacles like pedestrians and bicyclists.
RADAR
Four standard automotive radar sensors, three in front and one in the rear, help determine the positions of distant objects.
6CNY424
Source: Google
THE NEW YORK TIMES; PHOTOGRAPHS BY RAMIN RAHIMIAN FOR THE NEW YORK TIMES

Figure 2.7: Google autonomous vehicle configuration.

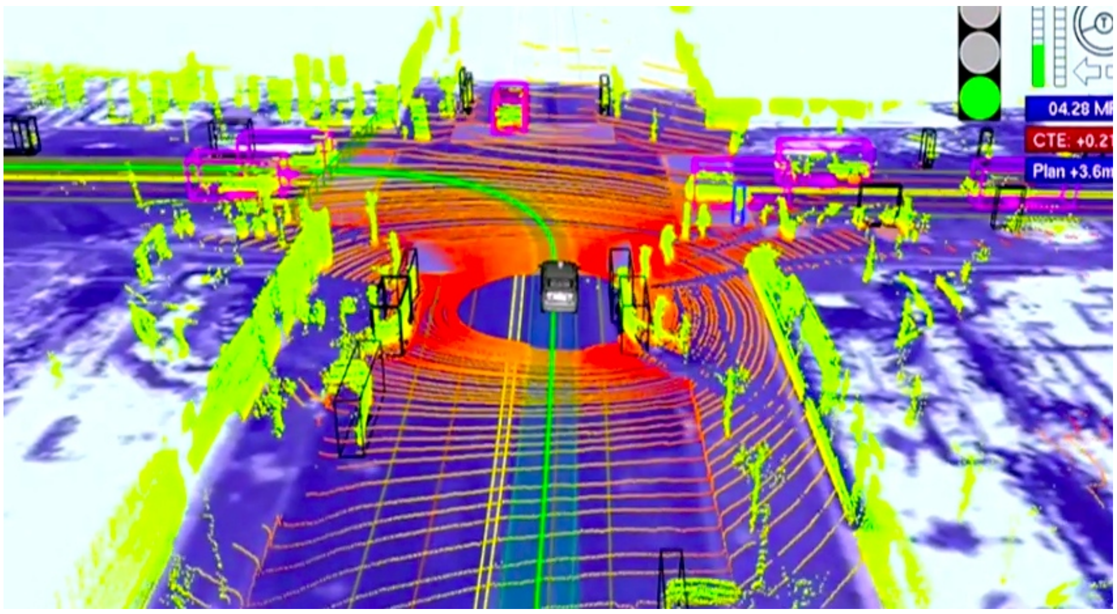


Figure 2.8: Velodyne 64-beam laser scanning output.[15]

Figure 2.9: Reported new google car.[16]

three-camera group to provide 180 ° panoramic view, while the two backward cameras are a pair of stereo camera. Moreover, four laser scanners are all mounted in front of the vehicle. A four-layer laser scanner in the middle of bumper and two normal laser scanners are mounted on each side of the bumper and The last one is on the roof rack, facing down to frame the ground. Also, the vehicle is equipped with GPS, IMU and Vehicle to Vehicle communication systems.

2.1.6 Summary

Therefore, as indicated in Table 2.1 (bottom two rows). Because of the low mass, low power consumption, adjustable field of view (FOV) and unlimited depth recovery ability, the monocular camera based navigation strategy in our scope is the most effective method for a small RPA.

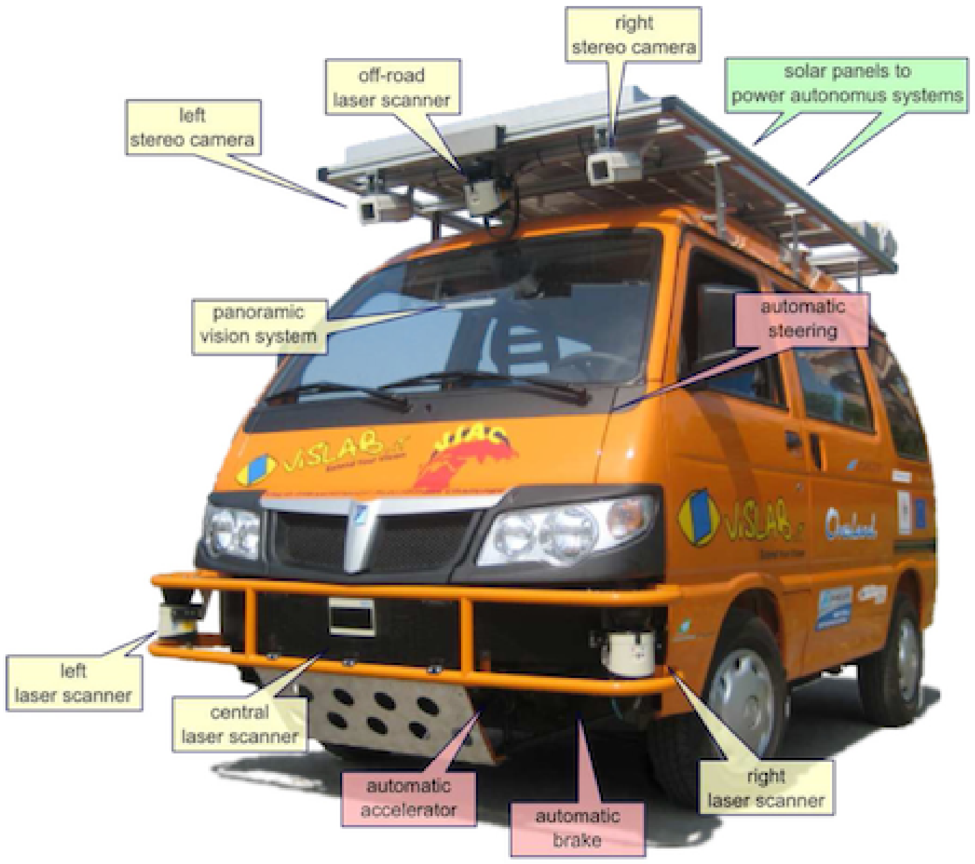


Figure 2.10: Front view of VIAC autonomous vehicle Broggi et al. (2012).

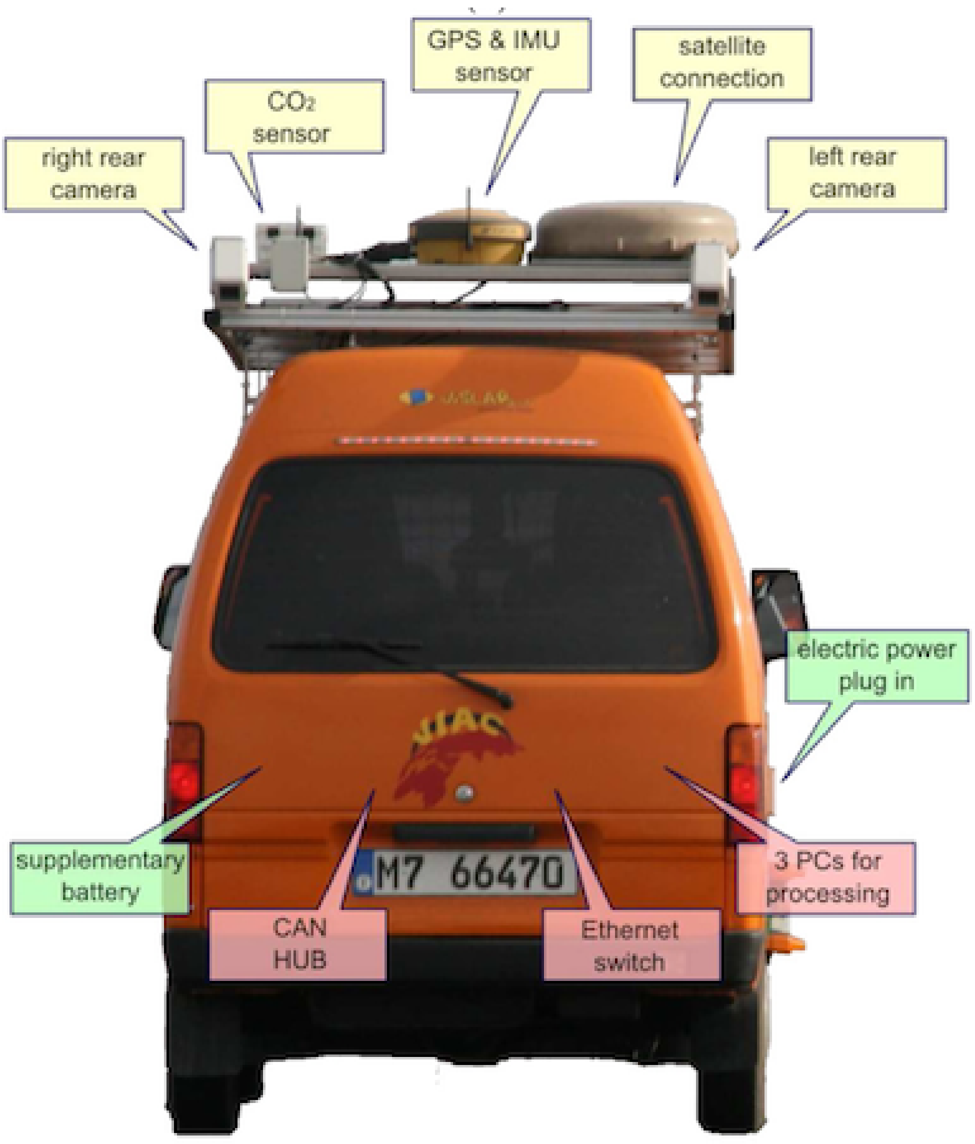


Figure 2.11: Rear view of VIAC autonomous vehicle Broggi et al. (2012).

Table 2.1: Comparison between different navigation solutions.

Solution Type	Independent	Mass	Power	Accuracy	FOV	Cost	Information Captured		
							Image Pixels	**Depth**	**Depth for Maximum Distance**
External Localization (Vicon Bonita 3)	No	0.226 kg	15 W	0.5 mm	$82^\circ \times 66^\circ$	around £20,000	Yes	Yes with multiple devices	12 m
Visual Land Mark	No	Approx. 0.03 kg	Approx. 0.3 W	Depends on Image Resolution	Adjustable	Approx. £50	Yes	Yes with multiple devices	1 m-50 m
Automotive RADAR	Yes	< 0.5 kg	3.7 W-12 W	0.3 m	$45^\circ \times 5^\circ$	Approx. £700	No	Inaccurate distance	60 m-500 m
SAR	Yes	> 2 kg	> 15 W	1 m	$5^\circ \times 5^\circ$	Approx. £60,000	No	3D Point Cloud	1000 m-2000 m
Laser Scanner	Yes	0.2 kg-0.8 kg	3 W-10 W	20 mm	180° to 270° 2D plane	Approx. £4,000	No	2D Plane	5 m-50 m
Flash LIDAR	Yes	0.05 kg-1 kg	2.5 W-15 W	Over 20 mm	$60^\circ \times 45^\circ$	£50-£2,000	No	3D Point Cloud	0.5 m-10 m
Stereo Camera	Yes	Approx. 0.07 kg	Approx. 0.7 W	Depends on Camera Resolution	Adjustable	Approx. £100 (two cameras)	Yes	3D Point Cloud	Approx. 10 m
Monocular Camera	Yes	Approx. 0.03 kg	Approx. 0.3 W	Depends on Camera Resolution	Adjustable	Approx. £50	Yes	3D Point Cloud without Scale	Unlimited

2.2 Monocular Visual Navigation

This section starts with introducing two classic background techniques: Structure from motion and Kalman-based filters, which enables monocular visual navigation. Then, it reviews and compares different types of monocular simultaneous localisation and mapping algorithms. Among them, a typical algorithm, called parallel tracking and mapping is discussed in detail to demonstrate the advantages and limitations of the similar algorithms. The optical flow techniques is also mentioned as a different solution using monocular vision. This section is ended by the discussion on the general scale problem of the monocular camera navigation, which leads to the introduction of the next section.

2.2.1 Core techniques briefing

2.2.1.1 Structure from motion (SFM)

Structure From Motion (SFM) is a range imaging technique in computer vision, which is derived from photogrammetry, and aims to recover 3D structure of the scene from a sequence of 2D images. When the relative motion is observed between object and camera, the depth information can be obtained. SFM problem involves many different techniques, such as kinetic depth effect, feature tracking, epipolar geometry and bundle adjustment. The basic principle is very similar with recovering 3D structure from stereo camera: the epipolar geometry is computed from different images, but the difference is that the stereo camera processes single frame from two cameras, while SFM processes series of frames from single camera. This is the core technique to enable the monocular Simultaneous Localization and Mapping (SLAM) in Section 2.2.2. The book by Opower (2002) is suggested for detailed reading.

2.2.1.2 Kalman filter

The Kalman filter, also known as a linear quadratic estimation (LQE), was firstly developed in the 1960s. It is an optimal recursive state estimator for linear system Kalman (1960), which combines the prior known linear model of the system with a series of noisy state measurement observation, to iteratively estimate the true state and filter out the noise. This estimator is originally from Bayesian Theorem and assumes a Gaussian noise distribution.

The general process can be summarised as:

1. *Predict:* predict current state estimation (including both estimation and covariance of the prediction probability) according to the prior linear system model and last state estimation.
2. *Update:* update the current state estimation considering the current state observation (in the form of measurement residual) in the fact of optimal Kalman Gain, which is computed after probability covariances of both prediction and measurement residual.

Then the updated state estimation is the output from Kalman Filter, and will be treated as the last state estimation in next iteration.

Extended Kalman filter is the nonlinear version of the Kalman Filter. By nonlinear, it means that instead of a linear function, the prior model of the system can also have differentiable functions (the classic EKF deals with first-order nonlinear functions). To adapt to this nonlinear model, instead of directly computing the covariances, the Jacobians matrix (a matrix of partial derivatives) of the model are used. This adaptation essentially linearises the non-linear function around the current estimate. Recently, higher order EKFs are described in Einicke (2012). These extension makes Kalman Filter more practical to real world problems with the cost of linearisation error, since majority of the real world systems are nonlinear, and the computational complexity is proportional to m^2 in theoretical best condition, where m is the number of elements in system state. Thus it works well only when the system is close to linear.

Unscented Kalman filter is a more sophisticated filter over EKF in terms of performance in nonlinear systems, which is originally proposed in 1997 in Julier and Uhlmann (1997). Essentially, because the linearisation in the classic EKF is done by computing the Jacobian of the model, which is the approximation based on Taylor Series Expansion, the reliable performance of the classic EKF requires the system to be almost linear on the time scale of the updates Julier and Uhlmann (2004). On the other hand, instead of computing the Jacobian, UKF uses Unscented transform to select Sigma points (a minimum set of sample points), and propagates them through the nonlinear functions, and then, the mean and covariance of the estimate are computed directly on the propagated Sigma points. By using the better way for linearisation, this method is proved to have significant improvement on estimating the true mean and covariance of system state. However, this improvement is cost by additional computation requirement (computational complexity is generally proportional to m^5, where m is the number of elements in system state), and it becomes significant when the size of the state is large.

2.2.2 Simultaneous localization and mapping (SLAM)

SLAM is a sub-field of mobile robotics research, which aims to build a map of its surrounding environment without prior knowledge, and at the same time dynamically localise itself with respect to the map. Recently a gap has been bridged between the SFM (as discussed in Section 2.2.1.1) and SLAM Strasdat et al. (2010), which enables the Monocular camera to be evolved in SLAM technique. In this subsection, we are only interested in Monocular camera based SLAM.

2.2.2.1 Filter based SLAM

Filter based SLAM employs filters (such as EKF or UKF) as the core to manage mapping and localisation (Typically refers to EKF-SLAM Civera et al. (2010), FastSLAM 2.0 Montemerlo et al. (2003) and RT-SLAM Roussillon et al. (2011)). In EKF-SLAM Civera et al. (2010), the state of the EKF is defined as a matrix including camera pose and observed feature positions. Therefore the EKF updates both camera pose and all feature positions, whenever camera captures a new frame. The computation complexity, which can be approximated from Section 2.2.1.2, is linearly proportional to n^2, where n is the number of observed features. It is generally known to be computationally expensive for high dynamic applications. The reason is that, those incremental SLAM methods updates camera pose and map feature positions in every frame, which wastes computation, particularly when there is no significant movement between two frames, the map is not necessarily to be rebuilt, and this is normally the case in general robotic operation, since the robot motion is slow relative to normal camera frame rate. In real-time application, where the computation efficiency is crucial for operation performance, This type of absolute synchronised SLAM has a significant limitation in principle. Therefore, it is not possible to involve this type if SLAM into the feedback loop of the RPAS flight controller.

2.2.2.2 Keyframe based SLAM

Keyframe based SLAM is a more straight forward solution to completely separate tracking (localization) and mapping as two separate threads, which enables the highly dynamic tracking operation while leaving map building slowly processed in background. This method has been proven to give the more accuracy per unit of computing time than filter based method Strasdat et al. (2010). Fundamentally different with previous SLAMs, it performs the localization solely with respect to a keyframe, based on the map (including tracked features and keyframes[17]) built previously whereas the map is built in a separate thread with the help from the localization output. This dramatically increases localization speed. The revolutionary framework is named as *Parallel Tracking and Mapping (PTAM)*,[18] Klein and Murray (2007) which was originally designed for Augmented Reality (AR) application. It demonstrates fast and accurate localization and relatively sophisticated mapping. The basic operation process has been summarised in Figure 2.12. As shown, the mapping thread and tracking thread are separated, which allows different operation rates: *tracking thread* processes every frame, prior pose is predicted with simple motion model and then coarse to fine pose update are carried out relative to a keyframe; whereas in *mapping thread*, sophisticated keyframe and feature insertion only operates when significant movement are observed, and local and global bundle adjustment are conducted periodically to maintain long term map consistency. The tracking computational complexity is linearly proportional to the number of features observed. In recent years, sinceWeiss et al. (2011) demonstrates

[17]keyframes are snapshots taken by the handheld camera at various points in time. Each keyframe has an associated camera-centred coordinate frame, and the observed features are in the position relative to its keyframe

[18]`http://www.youtube.com/watch?v=F3s3M0mokNc`

the feasibility to use PTAM for effective Micro UAV navigation, it becomes popular to be used in this area, such as Brockers et al. (2012); Engel et al. (2012); Jama (2011); Jama and Schinstock (2011); Nyman (2012). As shown in those papers, PTAM still needs significant modification before physically implemented in flight controller. However, keyframe based methods do not deal with uncertainty, and since they based on image features, it relies heavily on feature detection, matching threshold and robust estimation techniques (such as RANSAC), while in order for real-time performance, those techniques are all optimised for speed rather than accuracy. Therefore the drift of motion estimation is always significant, unless some computationally intensive optimisation methods (such as bundle adjustment) take place, which will in return reduce the speed.

ETHZ-PTAM Weiss and Siegwart (2011), which has been used as the primary visual navigation sensor in SFly project (world's first successful vision based autonomous micro aerial vehicle in GPS-denied environments), is a improved version of PTAM Klein and Murray (2007). PTAM is a efficient monocular visual SLAM algorithm, which successfully utilises multiple view geometry principles to recover the depth of tracked features in the video stream in real time. However, it is originally developed for augmented reality (AR) applications in small area. Several modifications from PTAM to ETHZ-PTAM , which enables it to be a feasible onboard vision navigation algorithm.

The testing was conducted under ROS (Robotic Operating System) in a consumer laptop. A customised USB camera driver node was made to continuously publish 640×480 image frame to a predefined topic in 30 Hz, and ETHZ-PTAM node subscript the same topic, and process whenever a new frame is available.

In terms of the algorithm operation, after the initialisation, the system starts the tracking operation (as shown in Figure 2.13), while dynamically building a virtual sparse 3D feature map (as shown in Figure 2.14) concurrently. Figure 2.13 indicates the operation from camera perspective. As shown, features in the video stream are tracked and indicated as colour-coded points, overlaid upon the original video stream. The colour-code indicates the level of pyramid, where the feature is tracked (indicating the size of the feature), and the grid indicates the virtual map that that the features are located. It is the same grid as the white grid in Figure 2.14. Therefore, when camera moves, the features will move together with map grid. On the other hand, Figure 2.14 indicates the operation from virtual map perspective. The colour-coded points are the same features as the above Figure. As we can see, the 3D position tracked features are accurately recovered in the virtual map, and the highlighted white-green-red frame indicates the current camera 3D position estimated by the algorithm, while the darkened frames are the last 16 keyframes used to construct the map. Therefore, when camera moves, the tracked feature will stay, while the camera frame will move the same way as the camera moves in 3D. Moreover, when camera gradually moves away from the current scene, the map will be dynamically updated and the old features and keyframes will be erased, which makes the algorithm to maintain the constant computation intensity, regardless of the increase the moving area. Furthermore, because of the dynamically updated map, the algorithm handles zooming nicely, so the camera

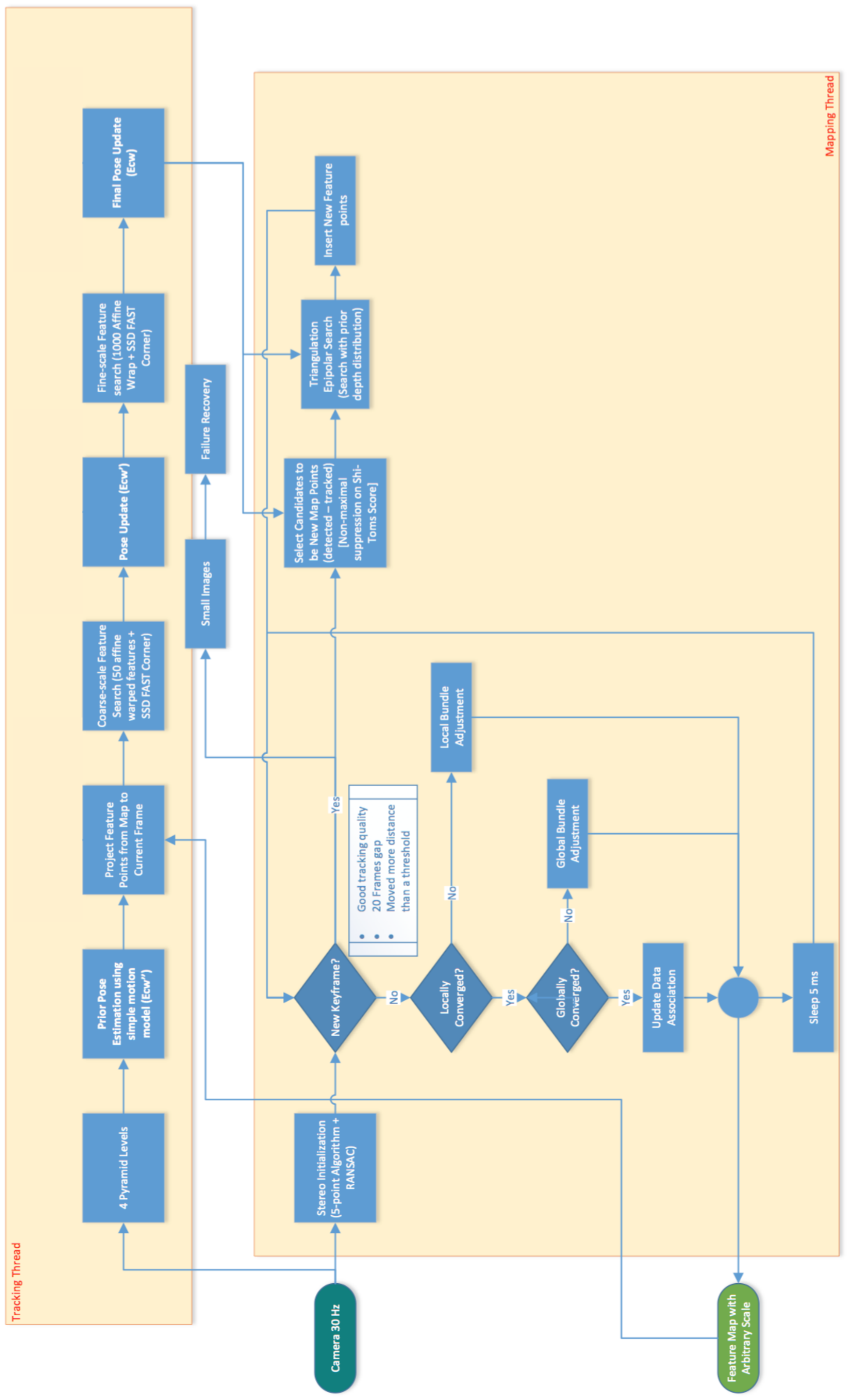
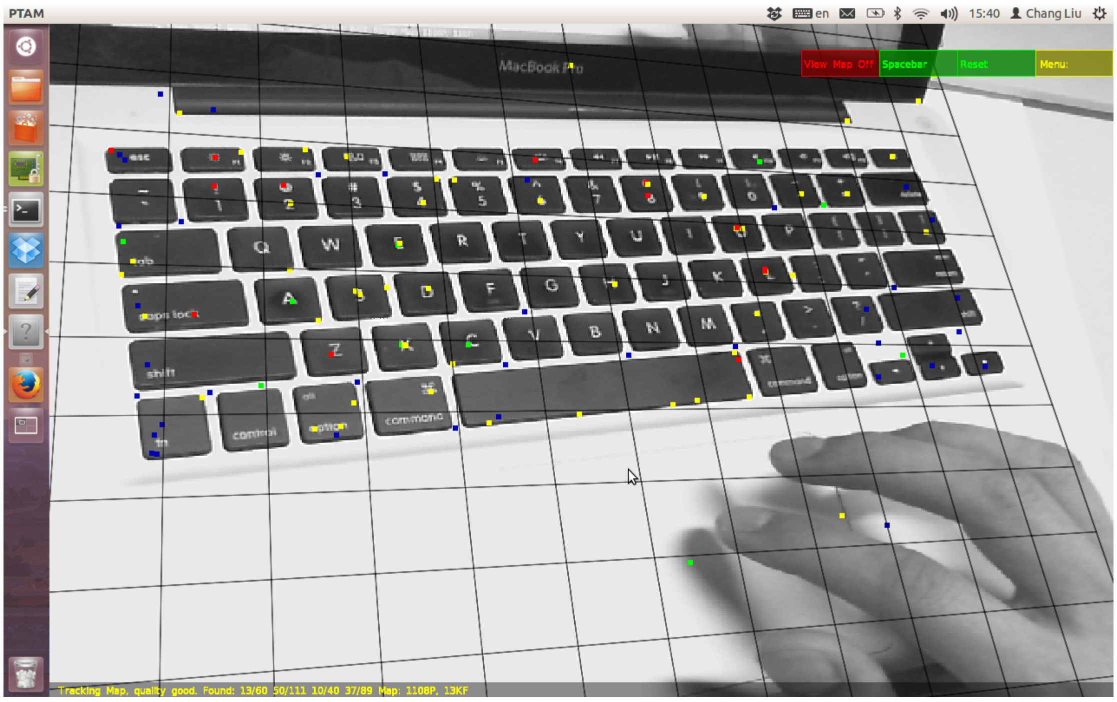

Tracking Thread
4 Pyramid Levels
Prior Pose Estimation using simple motion model (Ecw'')
Project Feature Points from Map to Current Frame
Coarse-scale Feature Search (50 affine warped features + SSD FAST Corner)
Pose Update (Ecw')
Fine-scale Feature search (1000 Affine Wrap + SSD FAST Corner)
Final Pose Update (Ecw)
Small Images
Failure Recovery
Mapping Thread
Camera 30 Hz
Stereo Initialization (5-point Algorithm + RANSAC)
New Keyframe?
Yes
No
Good tracking quality
20 Frames gap
Moved more distance than a threshold
Select Candidates to be New Map Points (detected – tracked) [Non-maximal suppression on Shi-Toms Score]
Triangulation Epipolar Search (Search with prior depth distribution)
Insert New Feature points
Locally Converged?
Local Bundle Adjustment
Globally Converged?
Global Bundle Adjustment
Update Data Association
Sleep 5 ms
Feature Map with Arbitrary Scale

Figure 2.12: PTAM summarised Block Diagram.



```
Tracking Map, quality good. Found: 13/60 50/111 10/40 37/89 Map: 1108P, 13KF
```

Figure 2.13: ETHZ-PTAM visualisation from camera perspective.

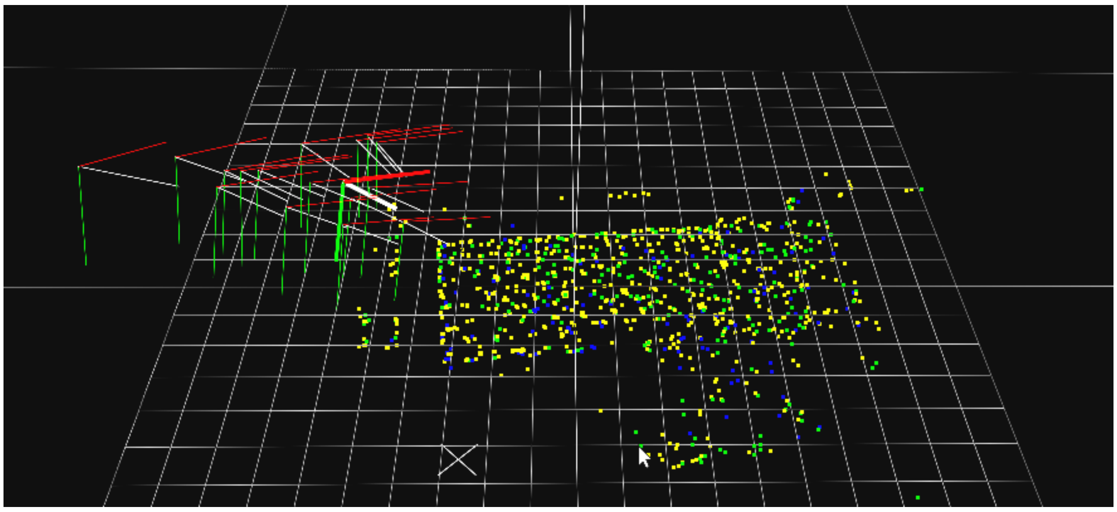


Figure 2.14: ETHZ-PTAM visualisation from virtual map perspective.

can start at very close to the keyboard and moves to 5 metres away from keyboard, while the algorithm easily remains tracking. Moreover, when camera is held in roughly fixed position (mocking RPA hovering), as expected, the map remains static (the mapping thread does not add new map points or update existing points). This saves significant amount of computation for the tracking thread. This is one of the big advantages of PTAM over filter-based SLAM algorithms.

Furthermore, many algorithm parameters can be adjusted dynamically by 'Dynamic-config' node provided by ROS. Two important parameters are the number of maximum keyframes maintained and number of maximum features tracked. More keyframes maintained effectively leads to bigger size of the map it will keep in the memory, which directly introduces more computation, but provides more features for the tracker to potentially track, thus makes the algorithm more robust. More features tracked for the tracker also means more computation for tracker thread to update and output denser feature map for mapping thread. However, this will give more accurate 3D camera state estimation, due to more features used to minimise reprojection error in 'Pose Update' step. Moreover, the algorithm also supports automatic reinitialisation procedure, which greatly improves the robustness of the system, especially when integrated into RPA control. Tracking failure therefore can be recovered, by reinitialising a new map.

PTAM optimisation towards onboard visual navigation sensor

Parallel Tracking and Mapping (PTAM) was the revolutionary design. because it was actually the first visual SLAM algorithm, which is fast enough to be involved into UAV control loop. Since Blösch et al. (2010) demonstrates the feasibility of employing PTAM as the primary positioning feedback for UAV navigation. It has also been improved by different researchers emphasising on solving different problems.

Automatic (re-) initialization: The PTAM uses manual stereo initialization for the first two frames. Making the PTAM initializes automatically, can make PTAM more robust to tracking failure, which means we can reinitialize another map when tracking failure cannot be recovered, and also this enables PTAM to operate in clustered and complicated environment. Jama and Schinstock (2011) have successfully used SURF matching for initialization. However Nyman (2012) has shown that this approach slightly costs mapping quality. Later in Weiss et al. (2012b), Stephen uses OF-inertial speed estimator to initialise the PTAM. In our method, we use the robust key frame extraction method based on Ahmed et al. (2010) for automatically selecting first two frame for stereo initialization.

Scale recovery with altimeter: The scale recovery is essential for fusing other sensor measurement into PTAM. The work in Nützi et al. (2011) uses EKF filter to estimate scale factor and gyro offset while fusing optical flow speed measurement with IMU reading. Then later in Engel et al. (2012), instead of filtering method, Jakob proposed a more computationally efficient approach, which use statistical formulation directly to estimate scale factor by fusing any monocular SLAM with other direct metric position or velocity measurement (such as ultrasonic sensor, pressure altimeter).

Gyro-aided tracking: Jama and Schinstock (2011) have proved that by replacing the simple motion model by the gyro integral as the prior pose estimation for tracking, the tracking performance can be significantly improved, especially for low frame rate and high rotational change circumstance. However, since PTAM tracks features in the scene, when high rotation takes place, the blurry scene will make it really difficult to track. The author of PTAM has already addressed this problem in Klein and Murray (2008), by adding edgelets to PTAM, which ends up better tracking robustness. We will implement gyro reading and edgelets into our application.

Map expansion with insignificant translation: In the case when the camera rotates significantly with no translation, the new scene needs to be mapped, which desires new keyframe to be introduced. This is a common operation for Vertical Take-off and Landing (VTOL) platform. However, the original PTAM examine the translation between current scene and last keyframe, and will not introduce new keyframe because of this condition. This will cause tracking failure, thus seriously effect the robustness of the PTAM navigation. Considering this, instead of using last keyframe for epipolar triangulation, Jama and Schinstock (2011) chooses the one has sufficient translation from current scene from all previous keyframes for mapping expansion. This method shows slight improvement on mapping ability over original PTAM. Since the keyframe-based SLAM does not include the probabilities in the map as EKF SLAM Civera et al. (2010) does, it saves computation but costs mapping ability when feature location is uncertain.

Map feature and keyframe management In terms of the computation complexity, firstly, it increases linearly with both the number of features and the number of keyframes in the map; secondly, the local and global bundle adjustment uses a big portion of the whole algorithm computation time, while the consistency of the map is not essential for short term control performance. In Weiss et al. (2012b), a keyframe manager is added to limit the number of keyframe stored in memory. Thus, only local bundle adjustment is performed. This approach significantly reduces the computation, while the system has to rebuild the map when revisiting the same place, where the keyframe has been deleted. In our approach, in addition to deleting the keyframe, the onboard computer sends the keyframes back to the ground station before deleting them. On the ground side, the ground station will save all the keyframes and do the global bundle adjustment, then sends the key frame back when the air system needs the keyframe again. In this way we reduces the computation while remaining the map consistency. Moreover, the feature management approach proposed in Weiss et al. (2012b) only stores the feature from highest three pyramid levels, which not only reduces computation, but also improves the tracking performance in self-similar scene. We will also implement this approach in our design.

Reusable feature map: One other problem with PTAM is that, any camera exposure change or light change will end up with tracking failure. In order to deal with that, we propose to normalize every individual feature in the map, and do normalization before patch search. By doing that, the map can be tracked in brightness-invariant manner.

2.2.2.3 Dense based SLAM

Dense SLAM was studied extensively most for ranging sensors such as Flash LIDAR (mentioned in Section 2.1.3). The framework for monocular camera is called DTAM (Dense Tracking and Mapping) Newcombe et al. (2011b), which is a monocular tracking and 3D reconstruction method. Based on the high performance GPU based Optical Flow estimator (FlowLib Werlberger et al. (2010)), to achieve real-time 3D dense reconstruction. Then it tracks the constructed dense model directly instead of extracted image features. Thus it makes use of every pixel in camera frame. It claims to achieve very robust performance in blurry and fast motion scenarios.[19] Moreover, an online generated 3D model of the environment greatly facilitates autonomous obstacle avoidance and recognition, path planning. Also, based on this SLAM, the technique for 3D reconstruction in dynamic scene in Keller et al. (2013) can be implemented.[20] However, the state-of-art dense methods still have the problem of high computation complexity, and heavily relies on massive parallel computation in the GPU. A recent paper from 2013 proposed semi-dense monocular SLAM Engel et al. (2013), which significantly reduced computation requirement by only estimating dense inverse depth map for the pixels on the edge (the areas have a non-negligible image gradient). The resulting algorithm is capable of running real-time in CPU only.[21]

2.2.2.4 Combination of keyframe and dense

The combination of keyframe and dense is the real state-of-the-art SLAM technology to obtain both speed and accuracy. Some very recent articles have investigated some potential approaches and obtained some promising results. Engel from TUM university proposed a semi-dense visual odometry method for monocular camera Engel et al. (2013), which significantly reduces the computation by performing dense reconstruction only on pixels with large gradient, and achieved CPU only operation in real-time. More excitingly, a fast semi-direct monocular visual odometry (SVO) method[22] by Forster et al. (2014). It is based on the parallel framework from keyframe SLAM, but eliminates feature extraction and robust matching techniques, which are slow and inaccurate for motion estimation. Instead, their algorithm operates directly on pixel intensities for motion estimation, which results in a highly accurate localization in a very high speed (300 fps). In addition to that, a probabilistic mapping method takes fully consideration of mapping uncertainty, which significantly out-performs the keyframe based methods in terms of outlier rejection. The Source code[23] was published in June 2014.

the fast semi-direct monocular visual odometry (SVO), to our knowledge, is the latest and most advanced open source SLAM implementation. It utilises the same keyframe-based parallel framework with PTAM, nonetheless, it tracks the features by directly conduct optimisation

[19] `http://www.youtube.com/watch?v=Df9WhgibCQA`
[20] `http://www.youtube.com/watch?v=2BdwMdh5M7Q`
[21] `http://www.youtube.com/watch?v=LZChzEcLNzI`
[22] `https://www.youtube.com/watch?v=2YnIMfw6bJY`
[23] `https://github.com/uzh-rpg/rpg_svo`

on image density. Thus, by eliminating the costly feature extraction step in tracking thread, it reduces more than half of the computation. This enables the onboard embedded computer to conduct all the essential process in near real-time.

In order to confirm the advantages and limitation of the SVO running on an embedded platform, a real experiment is conducted. As shown in Fig. 2.15, an embedded Linux single board computer, Odroid-U3[24], which features an 1.7 GHz quad-core processor, is installed on the quadrotor platform, and a global shutter high speed monocular camera with 90 degree FOV fisheye lens, is installed and connected to Odroid-U3.

In terms of the software setup, Robotic Operating System (ROS) is installed in both onboard Odroid-U3 and ground station laptop. The ground station laptop is able to access the onboard computer through wireless wifi link, through secure shell (SSH). SVO implementation is executed as an individual node in ROS on the onboard Odroid-U3 and all the data visualisation is performed remotely on the ground station laptop using RViz, which is a ROS package. The visualisation is shown in Fig. 2.16, which is a screen capture of one typical experiment trial, in which the camera points to the floor.

The core SVO process is able to perform 38 FPS in the onboard Odroid-U3 computer with real-time serial communication to autopilot and visual-inertial fusion algorithm running 100 Hz in parallel, which is sufficient for quadrotor position control purpose. The over all performance shows a promising tracking even with significant rotation, thanks to the wide angle fisheye lens. The 3D feature map constructed in real-time shows significantly less number of outliers than ETHZ-PTAM as tested before. That is because the map outlier in SVO is handled by the probability of each feature observed over multiple frames.

However, there are some situations, in which SVO loses track easily:

Camera too close to the scene: When the camera is less than 1 metre above the ground, the tracking quality will reduce significantly, and when the camera is less than 0.5 metre above the ground, the tracking will almost fail every time. This may caused by the inaccurate camera calibration with fisheye distortion, or the shadow of the quadrotor itself, which covers the area in the frame, when the camera is close to ground.

Unsuitable or sudden change of camera exposure: If the camera is setup as automatic shutter adjustment, when the camera changes the shutter speed while tracking, the SVO will immediately lose track, because all the contrast of the registered features are changed, thus there will not be any feature matching. Therefore, in most cases, the camera is force to setup as fix shutter speed, to prevent sudden exposure changes. However, the same shutter speed will never suitable for all the lighting conditions. therefore if the fixed shutter speed is too high or too low for the current lighting condition, then the captured image frame will be over or under saturated, which make it difficult for feature tracking. Thus

[24]`http://www.hardkernel.com/main/main.php`

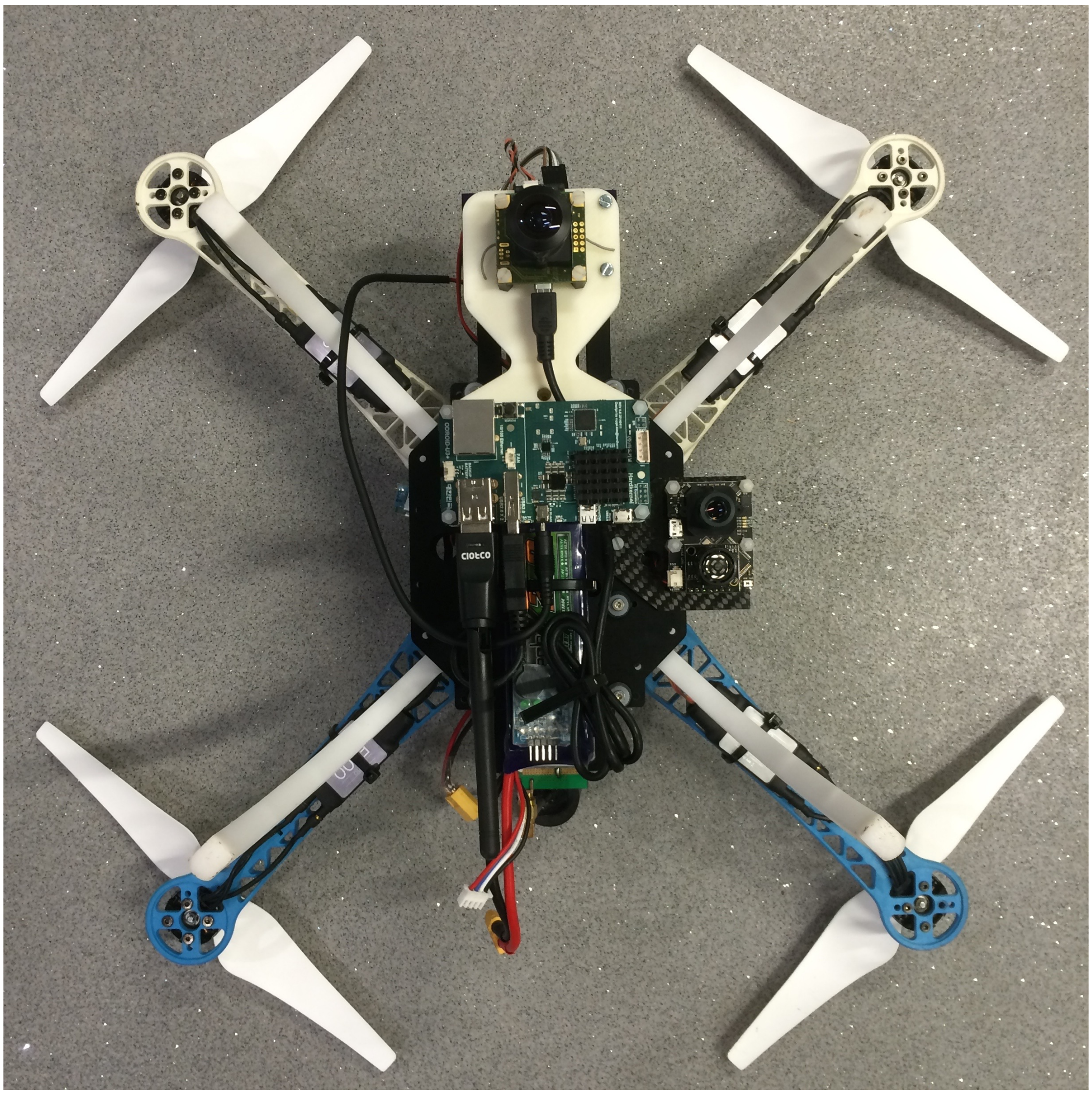


Figure 2.15: Hardware setup for testing SVO.

the robustness over dynamic lighting conditions will be an interesting future improvement for SVO.

Fast and sudden acceleration: The tracking optimisation of SVO assumes small changes between frames, which becomes untrue when the camera moves too fast or suddenly moves. Therefore, it is also interesting to fuse the short term inertial measurement back into the SVO as initial assumption for each optimisation step.

Rolling shutter camera: Generally for a rolling shutter camera, one image frame is captured by multiple shutter in a scanning manner, which means the different portion of the same image is captured in different time. The image can be easily distorted when the camera is in motion. This is widely called rolling shutter effect. While the global camera captures one image frame in a single shutter. The rolling shutter effect is not compensated in the SVO algorithm, thus a global shutter camera needs to be used for the algorithm for the maximal accuracy.

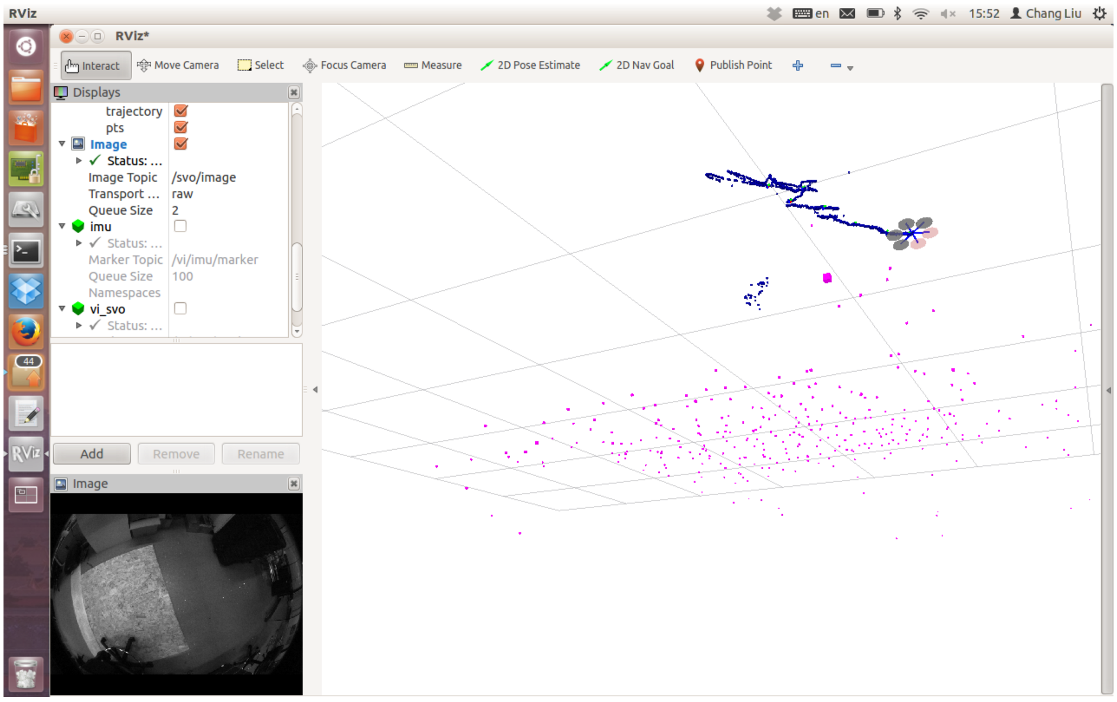


Figure 2.16: SVO visualisation.

Table 2.2: Comparison between SLAM algorithms. (n is the number of observed features)

Algorithms	**Computation**	**Update rate**	**Output Map**	**Handle Uncertainties**	**Handle Motion Blur**
Filter-based SLAM Section 2.2.2.1 (RT-SLAM)	n^2 at best	best at 50 Hz on Core2 2.2 GHz CPU	Sparse Feature Map (low dense)	Yes	No
Keyframe-Based SLAM Section 2.2.2.2 (PTAM)	n	at least 30 Hz on Core2 2.2 GHz CPU	Sparse Feature Map	No	No
Dense SLAM Section 2.2.2.3 (DTAM)	High (rely on GPU)	30 Hz on NVIDIA GTX 480 GPU hosted by an i7 quad-core CPU	Dense Map Model	Yes	Yes
Keyframe-dense SLAM Section 2.2.2.4 (SVO)	n	300 Hz on Intel i7, 8 cores, 2.8 GHz CPU, and 55 Hz on ARM Cortex A-9, 4 cores, 1.6 GHz Embedded processor	Sparse Feature Map	Yes	No (small blur with high frame rate)

Too little features in the scene: Same as all other feature based SLAM algorithms, SVO also relies on the features in the scene. The more feature rich is the scene, the more accurate will the SVO tracking measurement will be.

2.2.3 A comparison between SLAM algorithms

The comparison between different monocular SLAM is summarised in the Table 2.2. Note that this comparison chart is only for brief listing test results from corresponding papers. It is clear that the system configurations are different across different algorithms. Therefore, the fair comparison of the algorithm speed should take into account both the update rate and hardware configuration. From this perspective, it is worth noticing that SVO shows a significant speed boost over all the other algorithms. Reminding that it is also an open-source project. These facts encourage the usage of this specific algorithm in our following implementation works.

2.2.4 Optical flow (OF)

Optical Flow (OF) was a bionic computer vision technique, which computes image motion between two frames. It was originally derived by Horn and Schunk Horn and Schunck (1981) in 1980, based on brightness constancy assumption, which states the corresponding pixel intensities in the two consecutive frames are roughly the same, and smoothness constraint, which states the OF between the neighbour pixels are roughly the same, and the resulting algorithm iteratively computes the refined OF for every pixel between two frames. Then just one year later in 1981, Lucas and Kanade published a more efficient algorithm Lucas and Kanade (1981), which computes OF directly in a single iteration. This method is also based on brightness constancy assumption, and it further assumes the OF is same within a 3×3 pixel window. The resulting algorithm finds the compromise solution for the 3×3 window by the least squares principle. Moreover, since both of the methods computes the optical flow in 2×2 or 3×3 mask, they are not capable of handling larger OF than the mask. Thus the technique called Pyramid is used to reduce the image resolution to different levels, so that high OF can be estimated correctly. Several years ago, Kanade, Lucas and Tomasi came up with the KLT feature tracker Baker and Matthews (2004), based on the OF principle. The KLT tracks the features detected by Harris Corner Detector, by computing and linking the motion vector for each feature. The recursive manner is applied, since not only the OF (only indicate feature translation), but also other motion model (rigid, affine or projection model) is computed for more robust tracking.

2.2.4.1 PX4Flow optical flow camera

PX4Flow[25] is an optical flow camera, which is also equipped with an ultrasonic sensor and a gyroscope, as shown in Fig. 2.17. It has a native resolution of 752×480 pixels and calculates optical flow on a 4x binned and cropped area at 100 Hz and scales the optical flow value according to ultrasonic floor distance measurement to compute 2D translational velocity.

Because of the onboard gyroscope, it also compensates the optical flow caused by rotation. Unlike many mouse sensors, it has very high light sensitivity, thus it also works indoors and in

[25] https://pixhawk.org/modules/px4flow

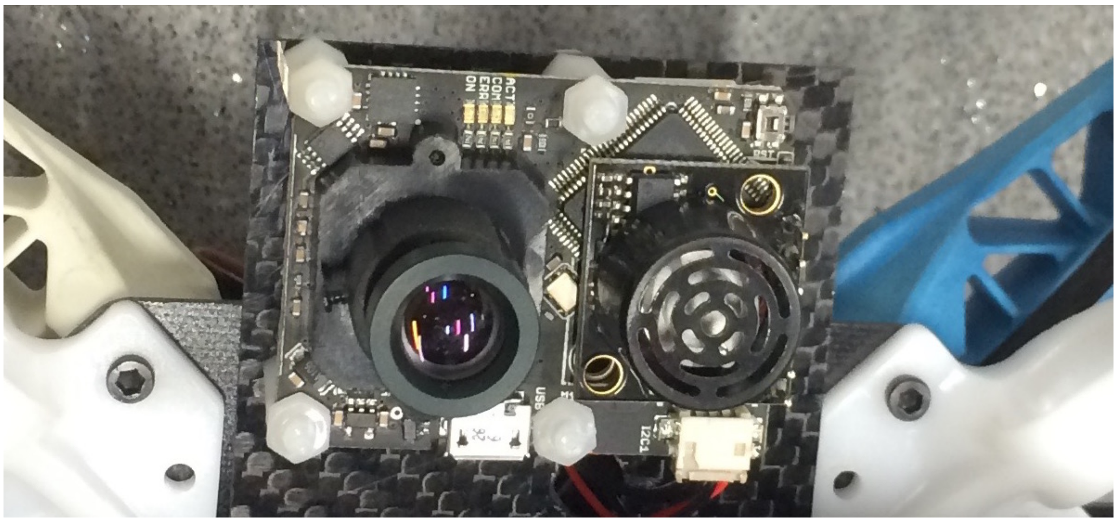


Figure 2.17: Installed PX4Flow camera.

low outdoor lighting conditions without the need for an illumination LED. It can also be freely reprogrammed to do any other basic, efficient low-level computer vision tasks.

The experimental test shows a very reliable performance, in terms of lighting condition and floor texture. It provides effective velocity measurement on almost any non-reflective surface. However, it limits the maximum distance from the floor to be 5 m.

2.2.5 Visual scale problem

All monocular navigation methods have the advantage of the ability to operate in the scene with any scale, since the flexible epipolar baseline is always relative to the scale of the local map. However, since the actual length of the epipolar baseline is unknown, the scale of the map is unknown. Thus, scale recovery becomes crucial when monocular SLAM is involved in the vehicle control, especially, when data fusion is desired between SLAM and other absolute scale sensors, such as an IMU or Ultrasonic altitude sensors. Generally, scale recovery can be achieved while the fusion is taking place. Different approaches has been proposed, Weiss and Siegwart (2011) estimates the scale continuously by fusing IMU measurement in EKF; the author in Engel et al. (2012) proposed a maximum likelihood (ML) approach with ultrasonic altitude sensor, which estimates the scale by minimising the negative log-likelihood. The latter shows faster convergence to the true scale. More interestingly, Hilsenbeck et al. (2012) presents a fast and robust scale recovery and visual reinitialisation method focusing on the long-term robustness and consistency of the system. The results demonstrate that errors and convergence times for scale estimation are considerably reduced, also recovery and reinitialization in parallel allowing almost seamless tracking.

2.3 Visual Inertial Navigation System (VINS)

Visual measurement is generally slow but accurate, while inertial measurement is fast but suffers from error accumulation. Due to this complementary nature between visual measurement and inertial measurement, the combination of visual and inertial sensors has been shown to be viable to achieve significant performance improvement over a single sensor system. Besides, since the absolute acceleration can be measured by inertial sensors, the visual scale problem discussed in Section 2.2.5 can be solved by the visual inertial fusion.

2.3.1 Inertial odometry

Inertia Measurement Unit (IMU) is a fundamental device used in RPA to stabilise flight. It generally consists of a 3-axis gyroscope, which senses vehicle angular velocity, a 3-axis accelerometer, which senses vehicle linear acceleration, and optionally a 3-axis magnetometer, which senses the absolute heading of the vehicle with respect to earth's magnetic field. Theoretically, when RPA is equipped with an IMU, the orientation can be obtained from the integral of angular velocity, while the position can be obtained from the combination of the orientation and double integral of the linear acceleration. However, in practice, since the current available IMUs are typically MEMS (Microelectromechanical systems) IMUs, they normally have non-negligible error. Also, the crystal oscillator clock is not sufficiently accurate. This results in the badly distorted pose estimation of the vehicle, especially for position estimation, because the double integral amplifies the error hugely. In order to overcome the problem, much research has been carried out to smartly fuse the three sensors measurements. Typical fusion is achieved by EKF, while recent investigations shows a effective improvements in orientation estimation using advanced complementary filter Calusdian et al. (2011); Mahony et al. (2008); Tian et al. (2013a) and gradient decent algorithm Madgwick et al. (2011a) (performance is demonstrated in open source IMU by x-io technologies[26]). This has been an interesting area for body movement tracking for movie and 3D gaming applications. While the position estimation still has non-negligible drift for long term applications. The advantage of the available IMU is that it is fast (typically 100 Hz) and responsive in short time intervals.

Recently, following the development of quantum mechanics, DARPA[27] is working on a Micro-PNT (Micro-Technology for Positioning, Navigation and Timing) program to TIMU (Timing and Inertial Measurement Unit), which essentially integrates a highly accurate chip scale atomic clock (CSAC) with an IMU, thus the accurate master clock significantly reduce the accumulated error when computing pose from integral of IMU data. Besides, extremely sensitive IMU is developed based on quantum interferometer. Those future technologies suggests possibility to achieve absolute pose estimation using Inertial information only Smith and Johnson (2013). However, those very expensive technologies are out of the scope in this review.

[26]http://www.x-io.co.uk
[27]http://www.darpa.mil

2.3.2 Visual inertial fusion

On the contrary to the high speed and high drift IMU state estimation, vision based state estimation has the advantage of low accumulated position drift, nonetheless, because of the massive data to be processed, the latency of the estimation is significant. Thus, visual measurement has the potential to correct the drift and observe the bias of the inertial measurement, whereas, inertial measurement has the potential to estimate scale, and bridge the between frames and visual failure. Therefore, fusing the inertial measurement with visual state estimation is the key operation to involve vision in the high performance control loop. The typical fusion method is Extended Kalman Filter (EKF), such as Engel et al. (2012); Tian et al. (2013b); Weiss and Siegwart (2011). Moreover, some recent implementations applied UKF for more accurate state estimation Shen et al. (2013a,b).

Tightly coupled fusion is based on filter based SLAM (see Section 2.2.2), where IMU measurements are introduced into the system state for filter estimation (normally EKF). Thus the filter is responsible to manage all observed features as well as IMU measurements and camera pose. The overall computational complexity that can be derived from Section 2.2.1.2, where for EKF the complexity is proportional to m^2. In this case, m is approximately the number of features.

Loosely coupled fusion Recently, majority of the promising results are achieved in the manner of loosely coupled structure, which treats the visual pose measurement as separate black box, and apply filtering methods (EKF or UKF) onto the outputs from the black box with IMU measurements. This method make the filter have constant computational complexity, regardless of the detail operation inside the visual black box, and this also leaves the choice for visual measurement open for different techniques. The following subsection discusses two very interesting recent achievements based on loosely coupled fusion method.

2.3.3 Two strong examples

Two very interesting recent achievements which are based on the loosely coupled fusion method, are discussed in this subsection. The first one is the Swiss SFly project in Autonomous System Lab (ETHZ university) in 2012, and the second one is the project of American GRASP lab in University of Pennsylvania.

The recent achievement in ETHZ SFly[28] project Weiss et al. (2012b); Weiss and Siegwart (2011) (shown in Figure 2.5 in Section 2.1), is the first to successfully demonstrate the capability of monocular vision-based control of autonomous MAVs, which uses a single downward facing

[28]SFly project: www.sfly.ethz.ch

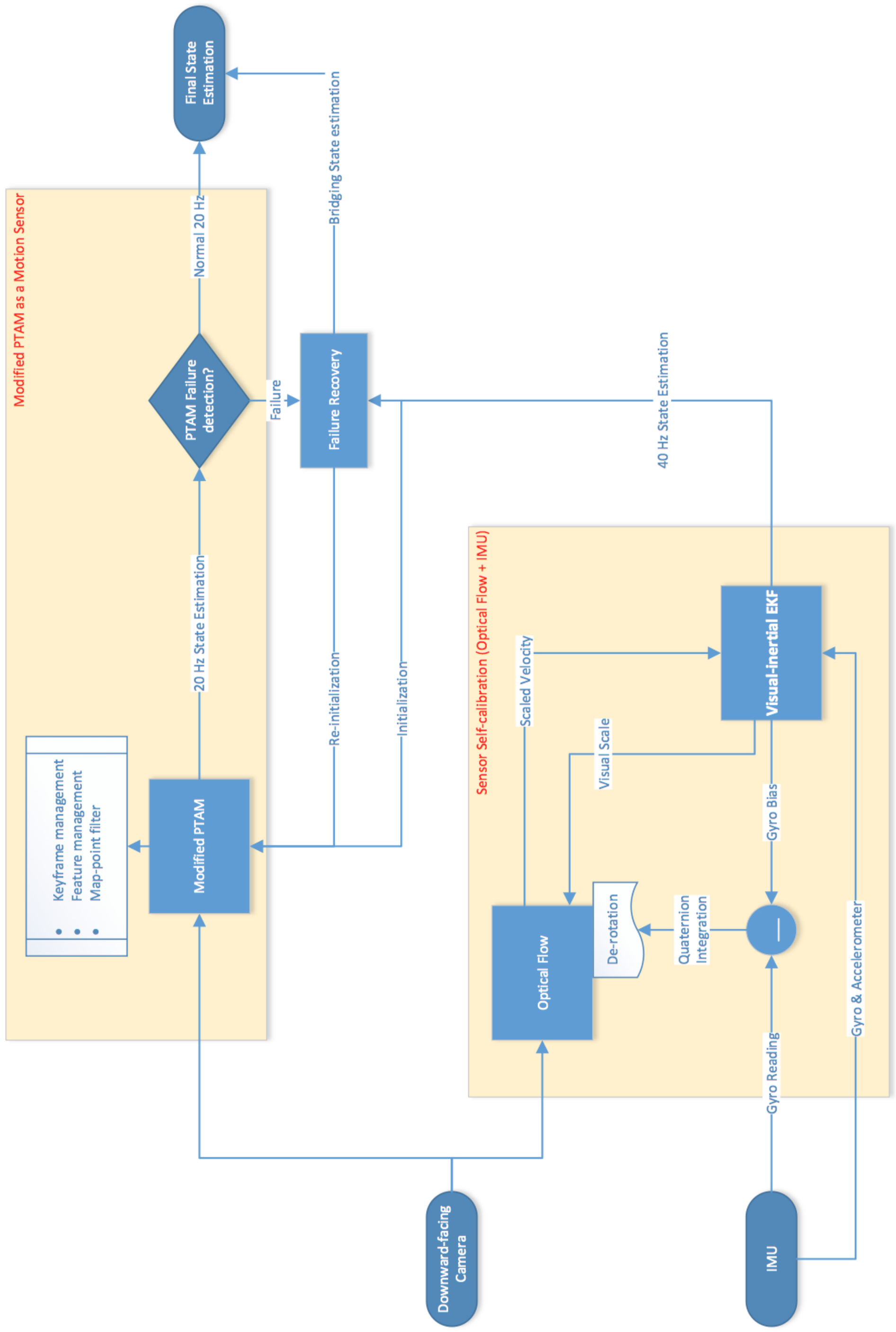


Figure 2.18: SFly summarised block diagram.

camera with complete onboard computation to achieve a power-on-and-go system for long-term navigation in large and initially unknown environments. As shown in Figure 2.18, It combines the modified version of PTAM Klein and Murray (2007) with Optical Flow (OF) based visual-inertial odometry. By refining the keyframe management, feature management and map point filter, the modified PTAM provides very efficient and powerful state estimation and mapping, whereas the OF-based visual-inertial odometry measures the scaled speed and calibrates the IMU online simultaneously, and then it is used to (re)initialise PTAM, correct drift, and bridge the pose estimation while tracking failure. In this setup, since the tracking computation for keyframe based SLAM has the linear computation cost against the number of features, and loosely coupled EKF has constant complexity, the overall computation is linear to the number of features. However, the system is limited to the operations which are far from ground or slow movement close to the ground, the reason is that *1)* the method used to recover the visual scale OF assumes the slowly varying image depth, which has the potential to change fast in close-to-ground fast flight; *2)* The PTAM assumes that all features can be seen from all directions in the map, which is not sure in clustered environment; *3)* downward facing camera is insufficient when close to ground.

Inspired by Weiss et al. (2012b), Shaojie in Shen et al. (2013a,b) proposed an hybrid visual approach (combination of monocular SLAM and stereo SLAM) towards high speed flight, as shown in Figure 2.19. He replaced PTAM by a redesigned monocular SLAM, which *1)* reduces the computation further by taking the initial guess from IMU measurement and decouple orientation and position estimation, *2)* becomes more robust for feature outliers, and instead of employing OF, he used a low speed secondary camera to obtain stereo correspondence of the tracked features, which is then used to recover the visual scale directly. This stereo correspondence *1)* removes the assumption of slowly-varying scale in OF method in [2], *2)* provides instant initialisation for monocular SLAM, *3)* continuously checks tracking failure and recovers scale/position drift. Finally, output visual state estimation and the IMU measurement are fused into an Unscented Kalman Filter (UKF), which results in an accurate state estimation in 100 Hz. The summarised block diagram is shown in Figure 2.20. However, in this approach, *1)* although scale/position drift is recovered continuously by stereo correspondence, the output feature map is assumed noiseless, without key-frame, where no global optimisation (bundle adjustment or loop closure) are considered. *2)* No sensor calibration is performed online; therefore, the sensor pre-calibration is required, and IMU bias is not compensated for longer-term flight. *3)* Forward facing cameras cannot capture enough features in higher altitude. *4)* The stereo camera pair with small fixed baseline will lose its ability to recover depth in far scene, which will be insufficient to correct monocular SLAM. Therefore, all the three reasons make it not suitable for long-term flight in large environments.

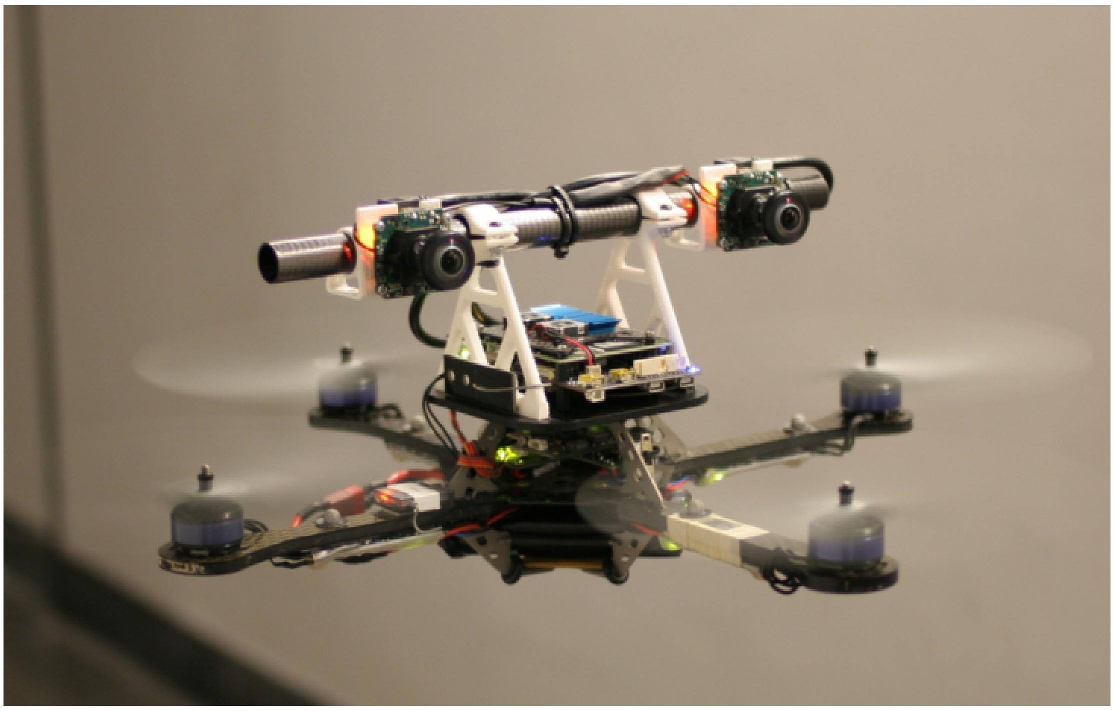


Figure 2.19: GRASP lab hybrid visual based navigation (Shen et al. (2013a)).

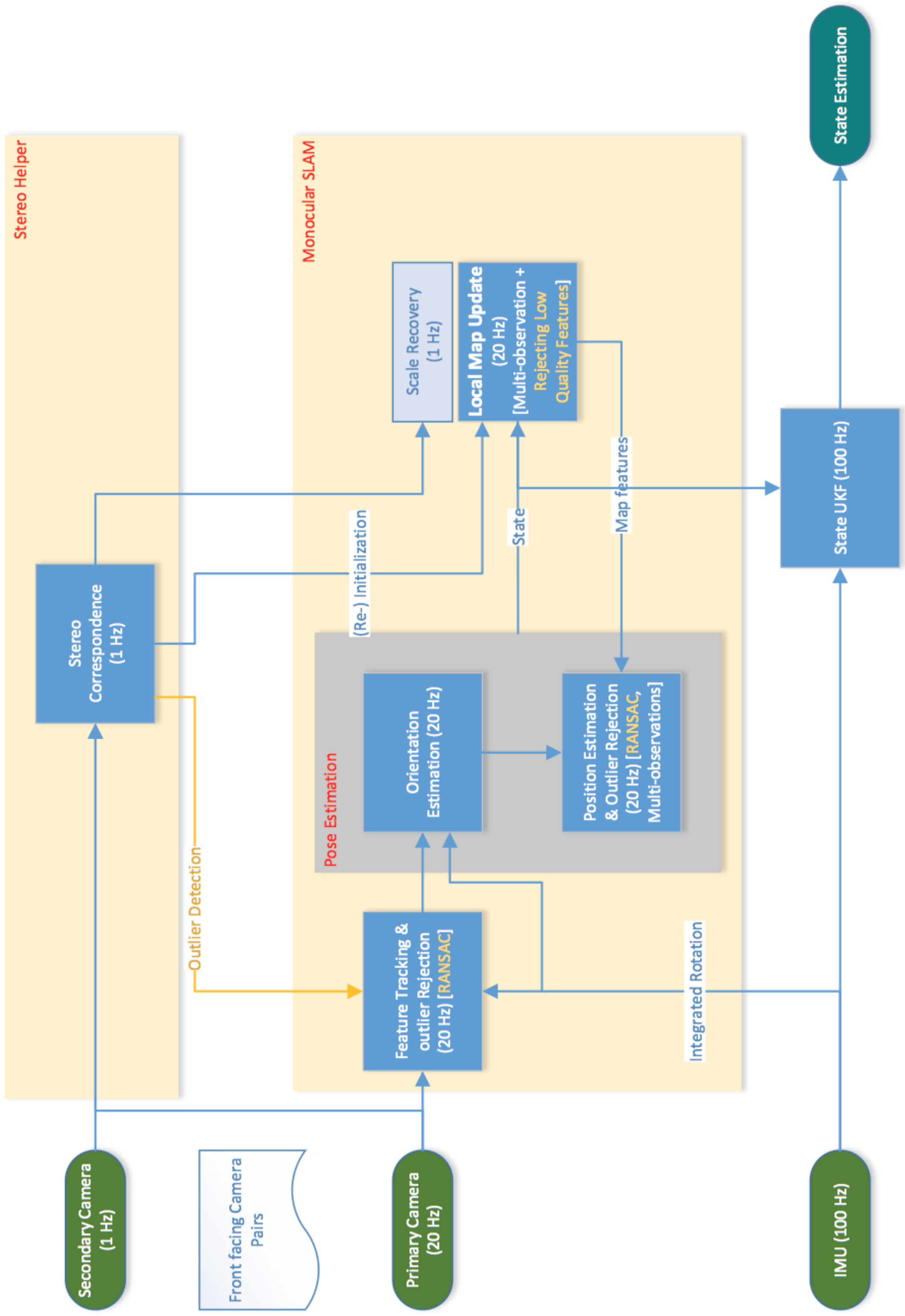
2.4 Platform and Control

Cascaded control structure is the most popular control framework for control of modern RPS, where inner loop controls three degree of freedom (DOF) attitude of RPS in very high speed (400–1000 Hz); outer loop controls the 3D position and then outputs the attitude command as the input of the inner loop.

2.4.1 Platform and modelling

Since this project focuses on civilian applications, where small or micro RPAS is more effective, it is better to employ RPAS as a testing platform of this project. So, in terms of small and micro RPAS constraints, it has very limited payload (less than 0.5 kg), which further limits the onboard power consumption, onboard computation and available sensors. Therefore, in future design, it is important to simplify onboard sensors configuration, to develop very efficient algorithm to process the rich vision information.

Moreover, it is very important to model and identify RPAS dynamics for high performance control method design and simulations. Kendoul in Bibuli et al. (2007) suggested various of method to achieve that, such as first-principle modelling, system identification technique Mettler et al. (2002) and combination of the two Abbeel et al. (2010). We are interested in the combination approaches, so that we can derive relatively accurate models with less mathematical complexity.



Stereo Helper
Monocular SLAM
Scale Recovery (1 Hz)
Local Map Update (20 Hz) [Multi-observation + Rejecting Low Quality Features]
State Estimation
State UKF (100 Hz)
Stereo Correspondence (1 Hz)
(Re-) Initialization
State
Map features
Pose Estimation
Orientation Estimation (20 Hz)
Position Estimation & Outlier Rejection (20 Hz) [RANSAC, Multi-observations]
Outlier Detection
Feature Tracking & outlier Rejection (20 Hz) [RANSAC]
Integrated Rotation
Secondary Camera (1 Hz)
Front facing Camera Pairs
Primary Camera (20 Hz)
IMU (100 Hz)

Figure 2.20: GRASP summarised block diagram.

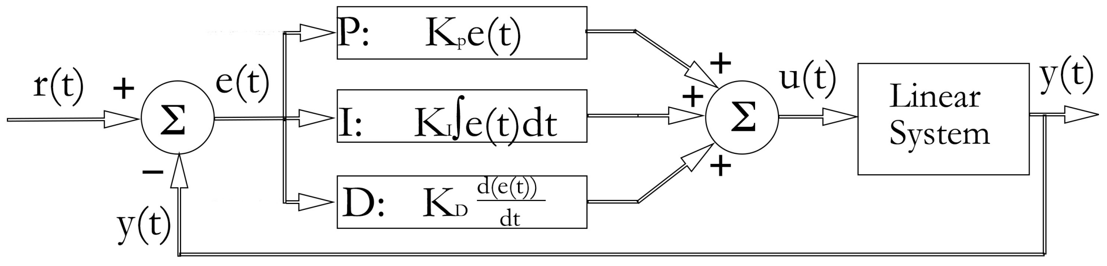


Figure 2.21: PID controller flow chart.

2.4.2 Proportional-integral-derivative (PID) controller

PID (Proportional-Integral-Derivative) controller is a popular industrial standard feedback controller for controlling most linear systems, which firstly appears in 1890s for automatic ship steering and been originally published in 1911 Bennett (1996). Generally speaking, a PID controller measures the difference (called error) between a measured process variable and a desired set-point, where the process variable is the system output variable, that the PID controller attempts to control. Then the controller intends to minimise the error, thus the process variable is close to the desired set-point.

To minimise the error, PID scales the error, integrate the error with respect to time and derivate the error with respect to time, thus the three terms (proportional, integral and derivative) are constructed. Then the sum of the weighted of the three terms forms the output from the controller to the linear system. The flow chart is shown in Figure 2.21. where r(t) is the user set point, y(t) is the measured output from the linear system, e(t) is the error, u(t) is controller output as well as input to the linear system, and K_P, K_I, K_D are the corresponding gains. The weighting of the three terms are called gains, which are the main parameters to be tuned to achieve different control performances. Also a PID controller will be called PI, PD or P controller when the gains for the corresponding terms are set to be zero.

To be more intuitive, the three term of PID controller can be interpreted in terms of time: proportional term is direct representation of current error; integral term is the accumulated error over the past time; and derivative term is the prediction of the future error based on the current rate of change of the error. Then, the gains determine the weighting of different terms effecting the controller output. A P controller is generally able to control a linear system, however it suffers from steady state error, which causes an offset from the set point when the system settles. It will be eliminated by sufficient integral gain, while too much integral gain will introduce instability. Moreover, a small derivative gain improves system stability.

The general manual tuning process starts with K_P, while leave K_I and K_D as zero. Increase K_P until the system starts oscillate, then use half of that K_P value for a "quarter amplitude

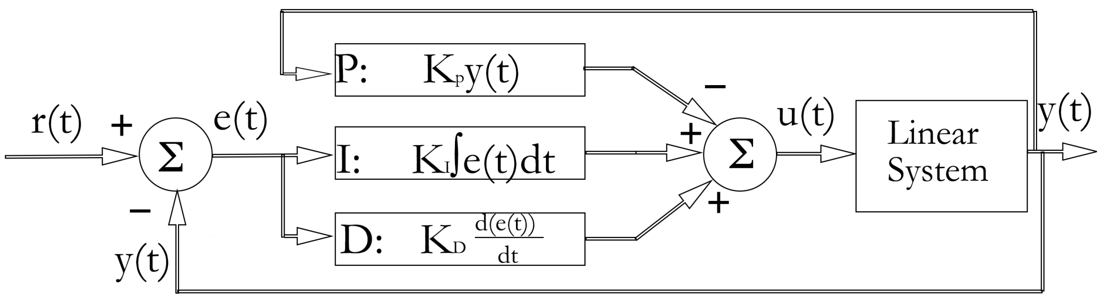


Figure 2.22: PDF controller flow chart.

decay" type response, and increase K_I until any offset is corrected within the sufficient time. Finally, gradually increase K_D until a load disturbance can be quickly recovered.

2.4.3 Pseudo-derivative-feedback (PDF) controller

A simple yet effective control structure was defined by Phelan (1977). This structure provides all the control aspects of PID control, but without system zeros that are normally introduced by a PID compensator. Phelan named this structure "Pseudo-derivative feedback (PDF) control" from the fact that the rate of the measured parameter is fed back without having to calculate a derivative.

The structure of the PDF and PID controllers are similar. The differences between PID and PDF control are best illustrated by comparing the controller flow chart of each structure. The PDF controller is shown in Figure 2.22. Instead of processing the error for all PID terms, for the PDF, the proportional gain only act on the system output as the negative feedback. This change effectively reduce the overall system overshoot. Thus it is considered to be a very useful controller type for UAV.

2.4.4 Visual servoing

Visual servoing is a concept of a special case in control problem which involves computer vision as its feedback to control the motion of robots. This research area focuses more on the control aspect of the system, which can be summarised as minimising the error between desired state and current state. Two schemes are generalised corresponding to how visual signal is used to describe the system state: *Image-based Visual Servoing (IBVS)* considers state as the features position in the image plane; whereas, *Position-based Visual Servoing (PBVS)* considers state as 3D camera pose with respect to reference coordinate frame, which can be derived based on computer vision (such as 3D localization). Chaumette and Hutchinson (2006, 2007) are a two-part series tutorial for visual servoing, which are recommended for detailed reading.

2.4.5 General flight control strategies

Based on the cascaded control framework, many different control theories have been applied. The linear control, such as classic PID controller, linear-quadratic regulator by Bergerman et al. (2007) and H-infinity control by Natesan and Bhat (2007), is the most popular and well supported method, while the performance suffers in aggressive manoeuvres. Besides, learning-based control, such as Fuzzy-logic by Garcia and Valavanis (2009), reinforcement learning by Abbeel et al. (2010) and neural-network by Dierks and Jagannathan (2010), has the advantages of flexibility on cross platforms and low computation requirement, while the stability and robustness have not been examined. Furthermore, model-based nonlinear control, such as feedback-linearization by Mellinger and Kumar (2011), adaptive control Kendoul et al. (2009), model-predictive control by Qi et al. (2010) and backstepping method by Liu et al. (2010), achieves higher level of flying capability. Versatile quadrotor controller has been achieved by Lee et al. (2010b), which demonstrates the quadrotor recovers from being initially upside down.

2.5 Summary towards Research Methodology and Framework

In this Section, a flow chart is presented as a summary of literature review, and also it represents the position-based visual servoing (PBVS) design framework, that will lead the present work and future work. Details are shown in Figure 2.23.

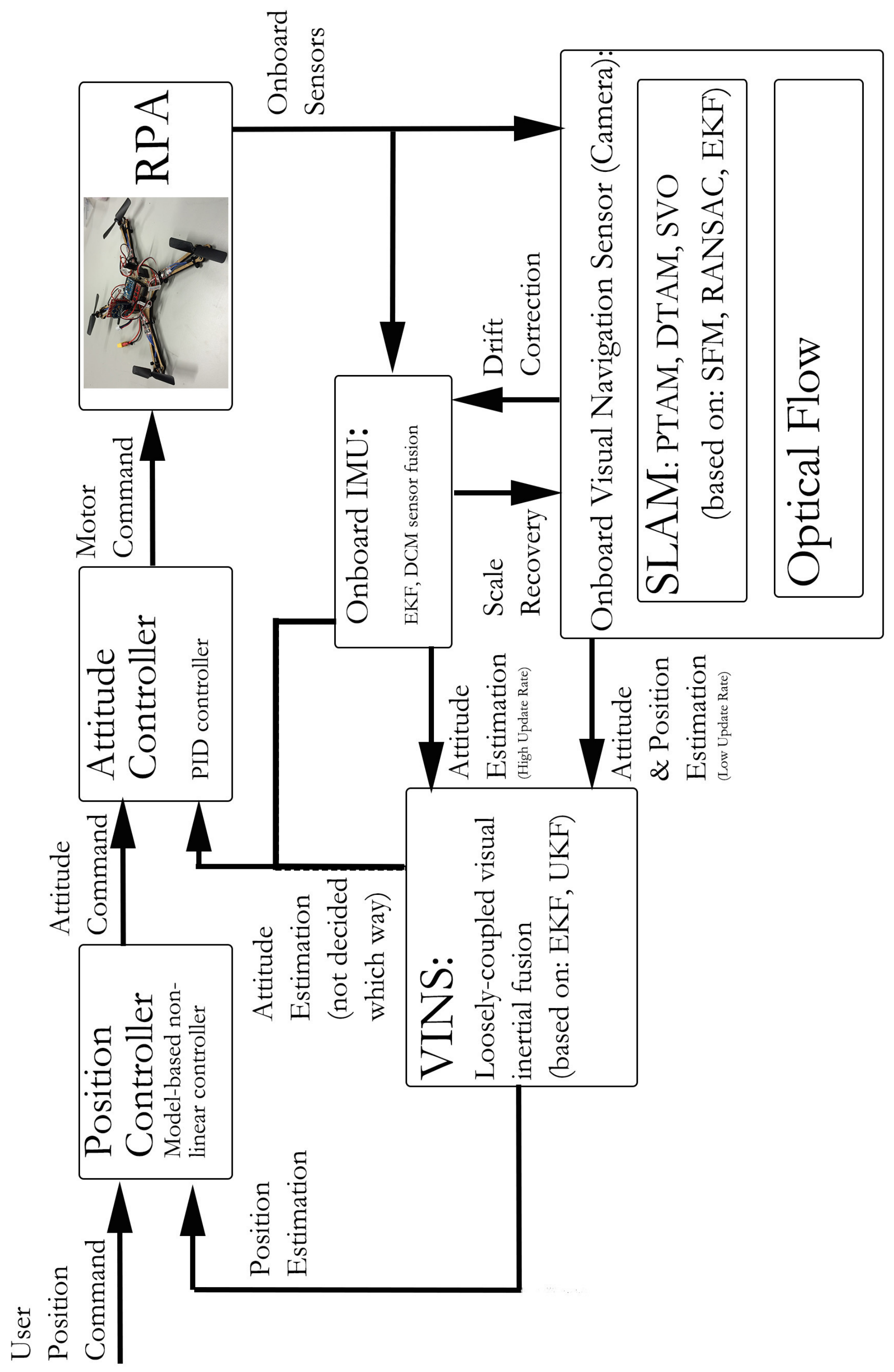

User Position Command
Position Controller
Model-based non-linear controller
Attitude Command
Attitude Controller
PID controller
Motor Command
RPA
Onboard Sensors
Onboard IMU:
EKF, DCM sensor fusion
Drift Correction
Scale Recovery
Onboard Visual Navigation Sensor (Camera):
SLAM: PTAM, DTAM, SVO
(based on: SFM, RANSAC, EKF)
Optical Flow
Attitude Estimation (High Update Rate)
Attitude & Position Estimation (Low Update Rate)
VINS:
Loosely-coupled visual inertial fusion
(based on: EKF, UKF)
Attitude Estimation (not decided which way)
Position Estimation

Figure 2.23: The proposed PBVS design framework.

Chapter 3

Quadrotor Modelling and Control Architecture Implementation

This chapter presents the design and implementation details of an advanced mini quadrotor system, including low cost commercial-off-the-shelf (COTS) electronics and advanced control algorithm. The proposed quadrotor has a gross takeoff mass of 758 g and 360 mm frame diagonal size. It is capable of semi-autonomous manoeuvre in GPS denied environments, solely relying on onboard sensors and computers. A globally defined quadrotor model is formularised, and a nonlinear velocity tracking controller is implemented on the special Euclidean group SE(3). An optical flow and ultrasonic based onboard downward-facing camera is used as the primary sensor to provide velocity and altitude measurement feedback for the controller. The control and sensor fusion algorithm is developed under Arduino compatible open source electronics.

The rest of this chapter is formed as follows: in Section 3.3, it explains the modelling of the quadrotor dynamics, and Section 3.4 describes the control architecture design based on the dynamic model. Then, Section 4.5 summarizes quadrotor implementation details, and then, Section 3.6 shows the test data to demonstrate the system performance. Lastly, Section 4.7 concludes and proposes future work.

Part of this chapter was accepted as Liu, C., Prior, S. D., and Scanlan, J. P. Design and Implementation of a Low Cost Mini Quadrotor for Vision Based Manoeuvers in GPS Denied Environments. *Unmanned Systems*, pages 1–12, DOI: 10.1142/S2301385016500059 (Liu et al., 2016a)

3.1 Introduction

Unmanned aerial vehicles (UAVs) are being considered in an increasing number of defence-related applications, for the purpose of reducing the risk of failure, and rising confidence in mission success. Moreover, the civilian market is predicted to rapidly expand over the next decade Kendoul (2012). The quadrotor is one of the most popular subset of UAVs. Because of its agile manoeuvrability, as well as its ability of vertical take-off and landing (VTOL) and stable hovering, it is commonly agreed to be an ideal candidate for search and rescue, surveillance, exploration, agriculture, monitoring and military applications in both indoor and outdoor environments.

Over the last decade, Global Positioning System (GPS) has been the key to enabling the autonomy of UAVs. It provides a global localization service with the best accuracy of 1-2 m. However, recently, due to the proven weakness of the GPS signal and rapid development of onboard sensing and computation capability, there has been growing interest in developing and researching alternative navigation methods for UAVs in GPS denied environments Achtelik et al. (2012); Bachrach et al. (2010); Bry et al. (2012); Engel et al. (2012); Jones and Soatto (2011); Shen et al. (2013a). The successful implementations will not only improve system robustness under GPS failure, but also enable a new range of applications out of GPS coverage.

A mini quadrotor is defined to carry under 2 kg payload by Kendoul (2012), which is sufficient for light weight perception sensors (such as cameras, laser scanner, radar and ultrasonic sensor) and embedded computer, which are essential for an autonomous navigation. Additionally, because they are low cost, easy to maintain, and safe to operate, these make them very good test-beds for research and development as stated by Mahony et al. (2012).

In this chapter, the vision-based method is believed to be the optimal sensor for navigation. The reason is that a camera has significant advantages over other sensors, such as low mass, low power consumption, low cost, adjustable field of view (FOV), high accuracy, additional colour information and long range. During the past five years, world's top research institutes had paid attention on developing advanced visual-based simultaneous localization and mapping (vSLAM) algorithms based on structure from motion (SFM) theory Engel et al. (2014, 2013); Forster et al. (2014); Klein and Murray (2007); Montemerlo et al. (2003); Newcombe et al. (2011b); Pizzoli et al. (2014). Those algorithms are efficient enough to execute in near real time on the modern onboard embedded computer, which makes it possible for a mini quadrotor to perform complete autonomous tasks in GPS-denied environments solely relying on onboard sensors and computers by utilising similar technology. Moreover, the visual scale problem, which was the main challenge of involving vision in control loop, is addressed to various extent by fusing onboard inertial measurements (accelerometer and gyroscope), which is named visual inertial navigation system (VINS) Dunkley et al. (2014); Jones and Soatto (2011); Kelly and Sukhatme (2009, 2016); Li and Mourikis (2013); Lobo and Dias (2003); Lynen et al. (2013); Shen et al. (2013b); Weiss et al. (2012a).

Therefore, in order to integrate the similar technologies into mini quadrotor platforms and for future improvements and developments, a suitable platform is required as the fundamental testbed. The most popular research platform in this category is the Hummingbird quadrotor sold by

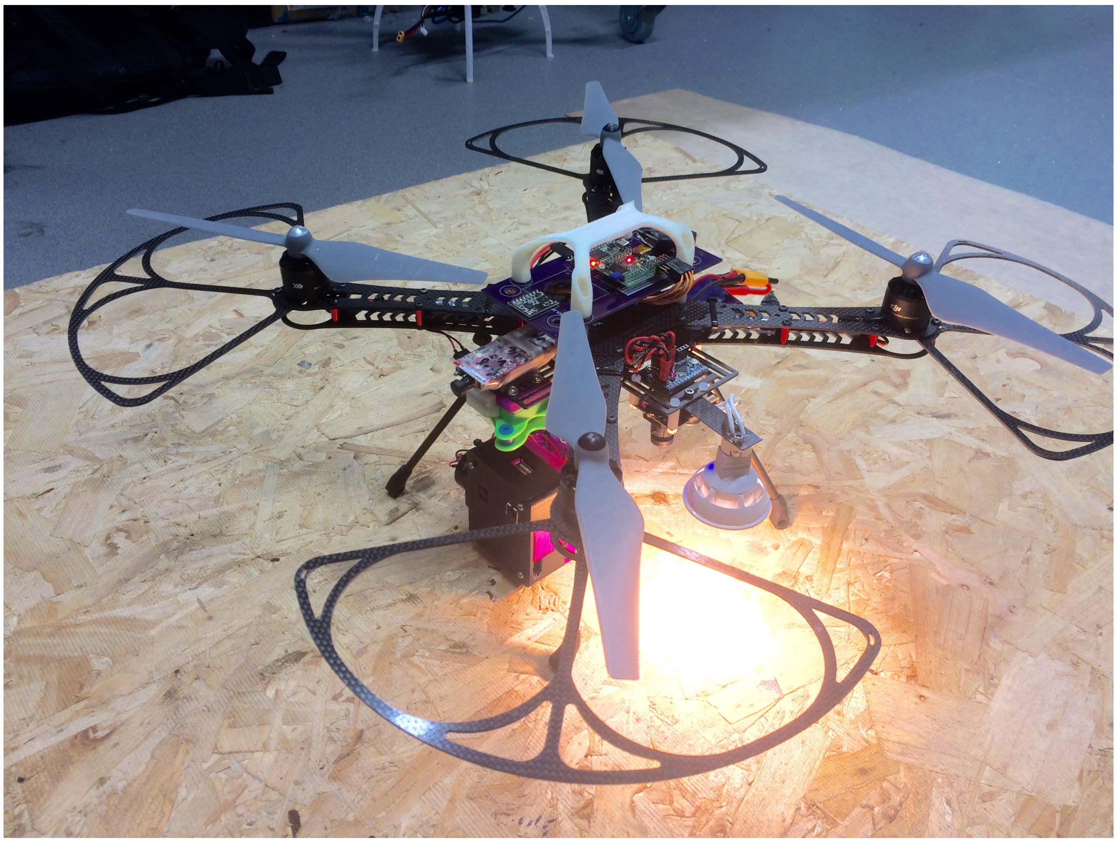


Figure 3.1: The developed quadrotor.

Ascending Technologies technical detail published in Kushleyev et al. (2013), with the state-of-the-art quadrotor autonomous control theory presented by Mahony et al. (2012). Thus, extensive research has been conducted on mini quadrotor development Elsamanty et al. (2013); Fernando et al. (2013); Jeong and Jung (2013). However, none of the above considers vision feedback in the quadrotor control loop. On the contrary, there are a few of commercial platforms, which are capable of vision based navigation, such as AR-drone and Bebop by Parrot, Phantom-3 and Inspire-1 by DJI. However, due to the nature of their consumer level application and the intellectual properties, they have very limited accessibility and extendibility for the usage as a research platform with onboard processing. Given this situation, as a continuous work based on the previous publication by Liu and Prior (2015c), this mini quadrotor, as shown in Fig. 3.1, is designed and implemented aiming to provide a test-bed for developing similar algorithms in the near future.

The rest of this chapter is formed as follows: in Section 3.3, it explains the modelling of the quadrotor dynamics, and Section 3.4 describes the control architecture design based on the dynamic model. Then, Section 4.5 summarizes quadrotor implementation details, and then, Section 3.6 shows the test data to demonstrate the system performance. Lastly, Section 4.7 concludes and proposes future work.

Symbols

D	Rotor Diameter, m
ρ	Air Density, $1.225\ kg/m^3$
C_T	Propeller Thrust Coefficient
C_Q	Propeller Torque Coefficient
C_P	Propeller Power Coefficient
λ	Advance Ratio
R_e	Blade Reynolds Number
η_p	Propeller Efficiency
η_m	Motor Efficiency
W	Total Air Flow Velocity, m/s
α	Angle of Attack, rad
T	Propeller Thrust, N
Q	Propeller Torque, Nm
K_V	Motor Speed Constant, $rev/s/V$
$\mathcal{R}$	Motor Resistance, Ω
i_0	Motor No-load Current, A
Q_m	Motor Torque, Nm
P_{Shaft}	Shaft Power, W

3.2 Propulsion System Fundamentals

Propellers are the only mechanical parts that produce force to lift the vehicle. If a propeller spins at a angular speed n with the forward axial travel speed perpendicular to the propeller plane at V_a, due to the angle of attack between total oncoming flow velocity and propeller pitch, the resultant force is generated by triangle-composing the blade lift force and drag force, which can in turn be triangle-decomposed into the propeller thrust (T) and propeller torque (Q). The theoretical expressions for T and Q are:

$$T = \rho n^2 D^4 C_T, \tag{3.1}$$

$$Q = \rho n^2 D^5 C_Q, \tag{3.2}$$

where n is rotor angular speed in rev/s, ρ is air density and D is propeller diameter. C_T is the thrust coefficient and C_Q is the torque coefficient (in many documentations it is also called power coefficient and denoted as C_P). The characteristic of a propeller can be specified by C_T and C_Q, which depend primarily on the advance ratio (λ), the blade Reynolds number (R_e) and the prop geometry. The mathematical expressions of advance ratio (λ) defined as:

$$\lambda = \frac{V_a}{nD}, \tag{3.3}$$

where V_a is also called the speed of advance, typically the true airspeed of the aircraft, and D is propeller diameter. The blade Reynolds number (R_e) are ratios defined as:

$$Re = \frac{inertial force}{viscous force} = \frac{\mathcal{V}L}{v}, \tag{3.4}$$

where $\mathcal{V}$ is mean velocity of propeller blade relative to the air, L is characteristic linear dimension (travelled length of air on the blade) and v is kinematic viscosity.

However, to measure the value of C_T and C_Q, we simply reverse equation (3.1) and (3.2), which yields:

$$C_T = \frac{T}{\rho n^2 D^4}, \tag{3.5}$$
$$C_Q = \frac{Q}{\rho n^2 D^5}. \tag{3.6}$$

Then, the propeller efficiency (η_p), which is defined by the ratio between thrust power and torque power, can be computed by:

$$\eta_p = \frac{Propulsive Powerout}{Shaft Powerin} = \frac{V_a T}{nQ} = \lambda \frac{C_T}{C_Q}. \tag{3.7}$$

The static test data in Brandt and Selig (2011); Deters and Selig (2008) implies the lack of efficiency for small propellers used on small RPA (maximum 60%). The reason is that they have low blade Reynolds number (R_e). The most important propeller geometric parameters are diameter and pitch value, which are quoted by almost every commercial propeller, such as 7×5 propeller corresponds to 7 inch diameter and 5 inch pitch. Here the pitch value is defined as the distance that the propeller would travel in one revolution if advanced into a solid material. The results in Brandt and Selig (2011); Deters and Selig (2008) also show that propellers with larger diameter and lower pitch were more efficient, in lower speed of advance, whereas higher pitch gives propeller better efficiency in higher speed of advance. A typical comparison graph[1] is shown in Fig. 3.2.

Motors drive propellers by applying torque. The characteristic of a electric motor can be mostly specified by motor speed constant (K_V, which is the ratio of motor internal back-EMF over rotation rate), motor resistance ($\mathcal{R}$), and no load current (i_0), which all can be tested

[1]`http://www.rcex.cz/?p=3593`

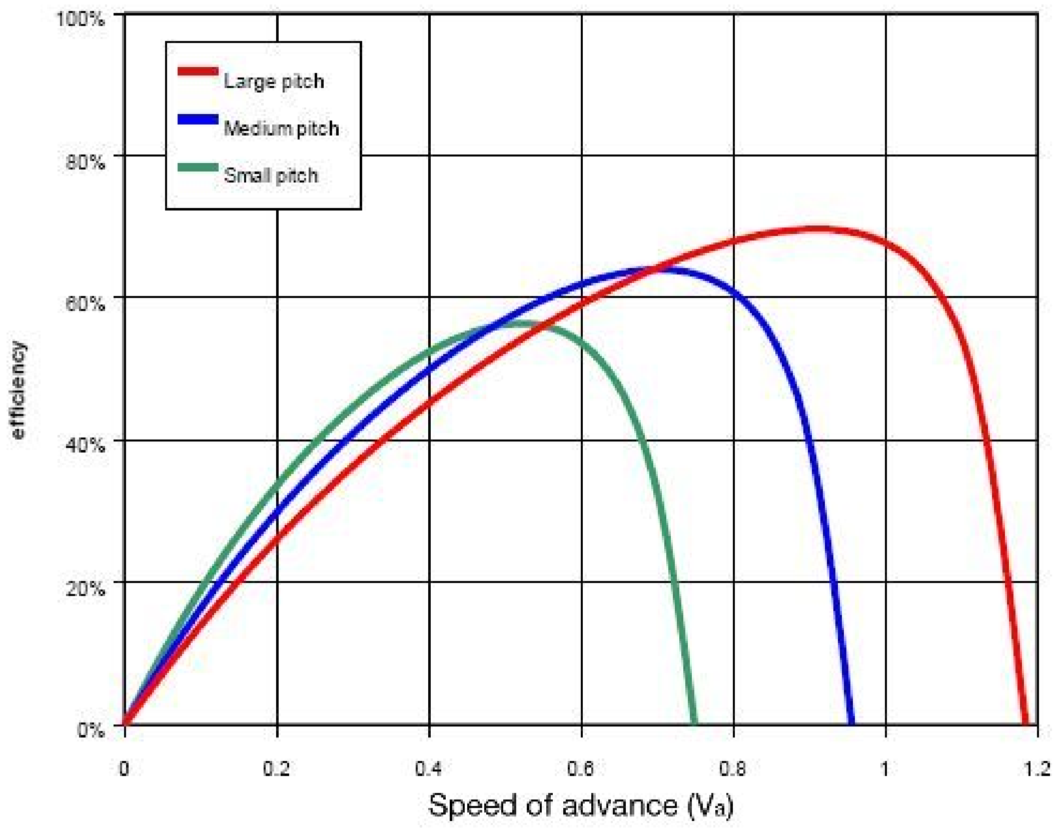


Figure 3.2: Typical propeller efficiency graph.

by bench experiments. Once the three motor constants are obtained, the motor torque (Q_m), shaft power (P_{Shaft}) and motor efficiency (η_m) can be computed as:

$$Q_m(n,v) = [(v - \frac{n}{K_V})\frac{1}{\mathcal{R}} - i_0]\frac{1}{K_V}, \tag{3.8}$$

$$P_{Shaft}(n,v) = [(v - \frac{n}{K_V})\frac{1}{\mathcal{R}} - i_0]\frac{n}{K_V}, \tag{3.9}$$

$$\eta_m(n,v) = [1 - \frac{i_0\mathcal{R}}{v - \frac{n}{K_V}}]\frac{n}{vK_v}, \tag{3.10}$$

where v is the applied voltage to the motor terminals. i is the current flow in the motor. Note here, we assume K_V is in rad/s/Volt, while it generally given in RPM/Volt. Thus the motor efficiency can be computed by:

$$\eta_m = \frac{P_{Shaft}}{P_{Elec}}, \tag{3.11}$$

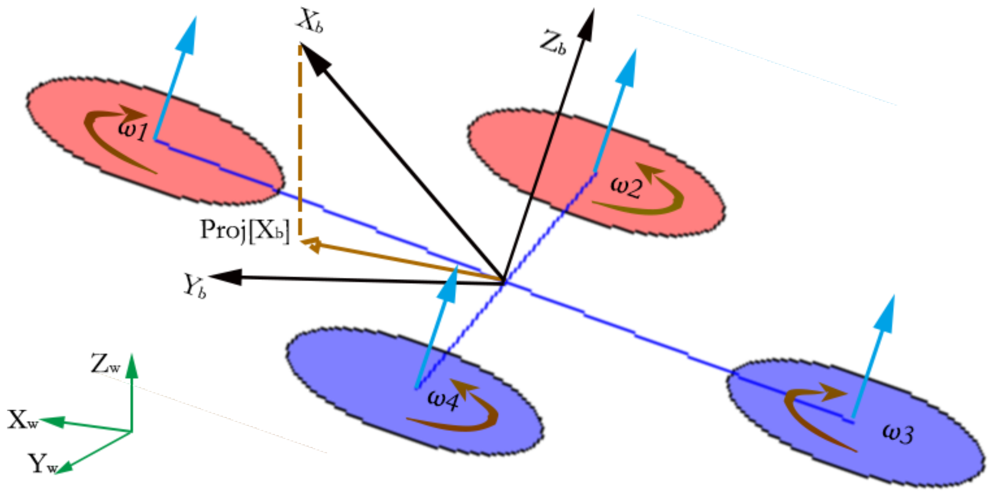


Figure 3.3: Coordinate system and quadrotor setup.

where electrical input power $P_{Elec} = vi$. A further problem for selecting motor is matching motor/propeller pair. Since the motor shaft and propeller always rotates at the same speed, the equilibrium operating speed occurs when $Q_m = Q$, the *impedance matching* takes place to ensure the peaks in η_p and η_m occur by roughly the same rotational rate and at the required thrust. This determines the size of the motor, given a propeller.

3.3 Quadrotor Dynamics Modelling

This section presents the nonlinear dynamic model of the mini quadrotor, which forms the basis for the controller synthesis in Section 3.4.

The coordinate frames and system setup is indicated in Fig. 3.3. Quadrotor body frame is fixed to the quadrotor body following right hand rule with X_b–axis pointing forward, Y_b–axis pointing left, Z_b–axis pointing up. World frame is fixed to the world, and is defined to have the same origin with body frame at the moment when the quadrotor connects to a battery. Z_w–axis points to the opposite direction of gravity and X_w–axis points to the same direction as quadrotor heading when connects to battery. The angles defined in the system follow the right hand rule. Fig. 3.3 also shows that the quadrotor has the cross configuration and four motors are numbered 1–4, with spinning directions as indicated.

The rest of the chapter uses $\mathbf{x_w}$, $\mathbf{y_w}$, $\mathbf{z_w} \in \mathbb{R}^3$ to denote unit vectors of the three world coordinates, thus $\mathbf{x_w} = (1, 0, 0)^\top, \mathbf{y_w} = (0, 1, 0)^\top, \mathbf{z_w} = (0, 0, 1)^\top$, and the unit vectors of the body frame are expressed in world frame as $\mathbf{x_b}$, $\mathbf{y_b}$, $\mathbf{z_b} \in \mathbb{R}^3$. The 3D rotation of the quadrotor body is represented by rotation matrix $\mathbf{R} \in SO(3)$. Therefore, we have $\mathbf{x_b} = \mathbf{R}\mathbf{x_w}$, $\mathbf{y_b} = \mathbf{R}\mathbf{y_w}$,

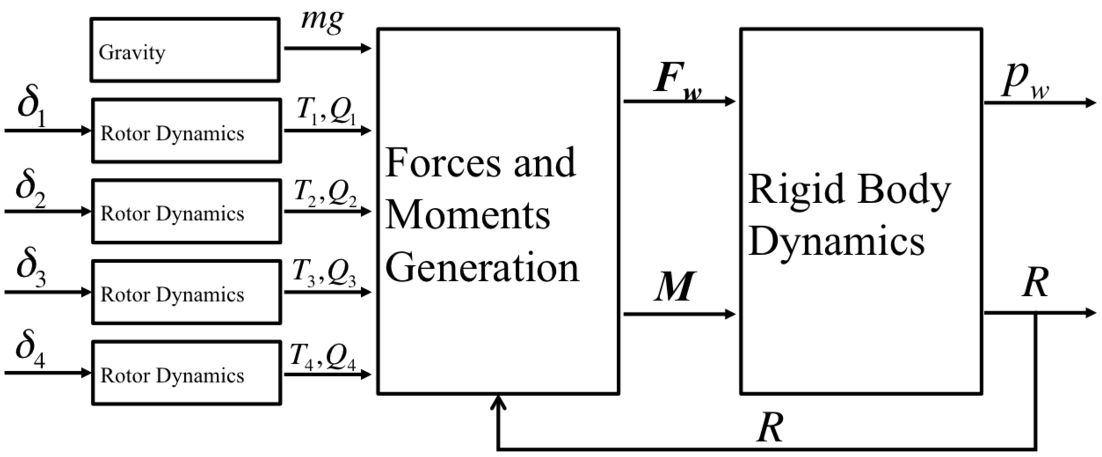


Figure 3.4: Overview of quadrotor dynamics model.

$\mathbf{z_b} = \mathbf{R}\mathbf{z_w}$, thus $\mathbf{R}$ can also be expressed as:

$$\mathbf{R} = \begin{bmatrix} \mathbf{x_b} & \mathbf{y_b} & \mathbf{z_b} \end{bmatrix}. \tag{3.12}$$

Note that we express the heading of the quadrotor as the unit vector parallel to the projection of X_b–axis onto the X_w–Y_w plane in world frame, denoted as $proj[\mathbf{x_b}] \in \mathbb{R}^3$, in Fig. 3.3.

The overview of the nonlinear model is shown in Fig. 3, where inputs of the model, δ_n, are the normalized pulse width modulation (PWM) command signal to the electronic speed controller (ESC) of motors, and outputs of the model are 3 dimensional (3D) position vector $\mathbf{p_w}$ ($= (x, y, z)^\top$) in world frame and body rotation matrix $\mathbf{R}$. The following subsections will explain the included components individually.

3.3.1 Rotor dynamics

The propulsion system of the quadrotor includes two pairs of counter-rotating ESC-motor-propeller systems. The dynamics of the four systems are identical and are approximated by the rotor dynamics model. The model receives the normalized PWM command δ and outputs thrust, T, in g and torque, Q, in Nm. If a propeller, with diameter D, rotates at n angular velocity in free air, whose density is ρ, assuming that the propeller drag is equal to the torque applied to spin, the thrust T and torque Q that it produces can be modelled as:

$$T = \rho n^2 D^4 C_T \tag{3.13}$$

$$Q = \rho n^2 D^5 C_Q, \tag{3.14}$$

where C_T and C_Q are thrust and torque coefficients respectively, which depend on propeller geometry, profile and Reynolds number. Furthermore, by assuming an ideal closed-loop ESC-motor system, which spins the propeller at the angular velocity that is linear to the normalised pulse width modulation (PWM) command, δ, ranging from 0 to 1, without mechanical delay:

$$n = k\delta - c, \tag{3.15}$$

where k and c are constants. Therefore, to model the propulsion system, simply substitute (3.15) into (3.13) and (3.14). However, in practice, to model the propulsion system with given parameters (ρ, D, C_T and C_Q), the derived equations can be simplified as:

$$T = c_T(\delta - c_o)^2, \tag{3.16}$$

$$Q = c_Q T, \tag{3.17}$$

where c_T, c_Q and c_o are constants determined by the given parameters, and c_T and c_o can be easily obtained from static thrust tests.

3.3.2 Force and moment generation

All the forces and moments applied to the quadrotor result in movement. They are generally generated by four different sources, stated byPhang et al. (2014), i.e., the gravitational force, the rotor thrust and moment, rotor reaction torques, and their gyroscopic effects. However, the last two have insignificant effect on overall forces and moments, thus we only consider the former two. Therefore, this module converts all the forces and moments into a force vector, $\mathbf{F_w} \in \mathbb{R}^3$ in world frame, and a moment vector, $\mathbf{M} \in \mathbb{R}^3$ about each body axis.

The gravitational force in world frame only applies to negative Z_w axis, which yields:

$$\mathbf{F_{gravity}} = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix}, \tag{3.18}$$

where m is quadrotor mass and g is standard gravitational acceleration.

Besides, each of the four rotors on the quadrotor generates thrust, T_n, and torque, Q_n, where $n = 1, 2, 3, 4$ (the numbering order is indicated in Fig. 3.3). The force generated by the four rotors applies to the positive Z_b axis in body frame, thus by rotating it into the world frame, we get:

$$\mathbf{F_{thrust}} = \mathbf{R} \begin{pmatrix} 0 \\ 0 \\ T_{total} \end{pmatrix} = \mathbf{z_b} T_{total}, \tag{3.19}$$

$$T_{total} = T_1 + T_2 + T_3 + T_4, \tag{3.20}$$

where T_{total} is the total thrust provided by the four rotors. $\mathbf{R}$ is the rotation matrix of the body frame, and $\mathbf{z_b}$ is the unit vector of Z_b–axis, as defined in (3.12).

Therefore, the total force applied onto the quadrotor in world frame can be derived as:

$$\mathbf{F_w} = \mathbf{F_{gravity}} + \mathbf{F_{thrust}}. \tag{3.21}$$

The other output from the module is the moment vector in body frame, which is approximated in this chapter to be generate by the thrusts and torques of the four rotors. Roll moment is contributed by the thrust difference between rotors 1, 4 and 2, 3. Pitch moment is contributed by the thrust difference between rotors 1, 2 and 3, 4. Yaw moment is contributed by the torque difference between rotors 1, 3 and 2, 4. Thus, by also substituting (3.17), it then can be formulated as:

$$\mathbf{M} = \mathbf{BCT}, \tag{3.22}$$

where:

$$\mathbf{B} = \begin{bmatrix} \frac{\sqrt{2}}{2}l & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2}l & 0 \\ 0 & 0 & c_Q \end{bmatrix}, \tag{3.23}$$

$$\mathbf{C} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}, \tag{3.24}$$

$$\mathbf{T} = (T_1, T_2, T_3, T_4)^\top. \tag{3.25}$$

The thrust vector, $\mathbf{T}$, represents the thrusts generated by the four rotors, and l is the distance between rotors and quadrotor centre of mass.

3.3.3 Rigid-body dynamics

Rigid-body dynamics formularises the translational and rotational dynamics of the quadrotor, by utilising the simplified Newton-Euler formalism. Therefore, the resulting position vector, $\mathbf{p_w} \in \mathbb{R}^3$ in world frame, and angular speed vector $\mathbf{\Omega} \in \mathbb{R}^3$ about each body axis, can be obtained as:

$$m\ddot{\mathbf{p}}_\mathbf{w} = \mathbf{F_w}, \tag{3.26}$$

$$\mathbf{J}\dot{\mathbf{\Omega}} = \mathbf{M}, \tag{3.27}$$

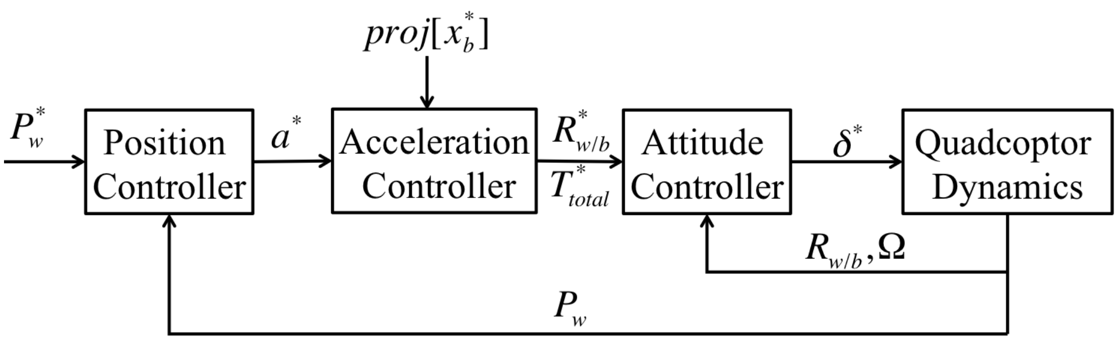


Figure 3.5: Overview of the nonlinear controller.

where $\mathbf{J}$ is the inertia matrix of the quadrotor, and since our quadrotor is approximately four way symmetrical, $\mathbf{J}$ is assumed to be a diagonal matrix, as:

$$\mathbf{J} = \begin{bmatrix} J_X & 0 & 0 \\ 0 & J_Y & 0 \\ 0 & 0 & J_Z \end{bmatrix}, \tag{3.28}$$

where J_X, J_Y, J_Z are the moment of inertia values around each axis of body frame.

3.4 Controller Design

Based on the dynamics model developed in the previous section, a nonlinear robust controller is designed to ultimately control the quadrotor 3D position $\mathbf{p_w}$ and heading $proj[\mathbf{x_b}]$ to match user input command $\mathbf{p_w^\star}$ and $proj[\mathbf{x_b^\star}]$. As shown in Fig. 3.5, three sub-controllers: attitude controller, acceleration controller and position controller, are developed. The quadrotor is controlled accordingly by taking the feedback measurement from inertial measurement unit (IMU) and vision based position sensor. Note that the detail of the position sensor design is not the focus of this section, thus here we assume the position is obtained from the position sensor.

3.4.1 Attitude controller

The attitude controller receives the desired body orientation represented as rotation matrix $\mathbf{R}^\star$, and desired total thrust provided by the four rotors, $T_{total}^\star$, from acceleration controller output. With the help of attitude feedback measurement, $\mathbf{R}$, from IMU attitude fusion, and angular velocity, $\mathbf{\Omega}$, directly measured from gyroscope, then it commands normalised PWM signals, $\delta_n^\star$, to the four ESCs of motors. It is designed to minimise both the attitude tracking error, $\mathbf{e_R} \in \mathbb{R}^3$, and angular velocity error, $\mathbf{e_\Omega} \in \mathbb{R}^3$, while maintaining the total thrust, T_{total}, as commanded.

Given the desired orientation, $\mathbf{R}^\star$, the attitude error, $\mathbf{e_R}$, is defined to be the sine of the angle of rotation about each body axis to go from $\mathbf{R}$ to $\mathbf{R}^\star$. It can be formularised as Lee et al. (2010a):

$$\mathbf{e_R}{\times} = \frac{1}{2}(\mathbf{R}^{\star\top}\mathbf{R} - \mathbf{R}^\top\mathbf{R}^\star), \tag{3.29}$$

which yields a skew-symmetric matrix in the form of:

$$\mathbf{e_R}{\times} = \begin{bmatrix} 0 & -e_{Rz} & e_{Ry} \\ e_{Rz} & 0 & -e_{Rx} \\ -e_{Ry} & e_{Rx} & 0 \end{bmatrix}, \tag{3.30}$$

where the cross map $\times : \mathbb{R}^3 \to \mathfrak{so}(3)$, thus we get $\mathbf{e_R} = (e_{Rx}, e_{Ry}, e_{Rz})^\top$. Moreover, the angular velocity error, $\mathbf{e_\Omega}$, is defined as Lee et al. (2010a):

$$\mathbf{e_\Omega} = \mathbf{\Omega} - \mathbf{R}^\top\mathbf{R}^\star\mathbf{\Omega}^\star, \tag{3.31}$$

where $\mathbf{\Omega}^\star \in \mathbb{R}^3$ is the angular velocity of the desired rotation, $\mathbf{R}^\star$, about each axis of the desired body frame. It can be obtained by the tangent operator equation of the desired rotation matrix, as:

$$\mathbf{\Omega}^\star = (\mathbf{R}^{\star\top}\dot{\mathbf{R}}^\star)^\vee. \tag{3.32}$$

Then, we can apply the proportional–derivative (PD) control law to compute the desired angular acceleration vector, $\boldsymbol{\alpha}^\star \in \mathbb{R}^3$, about each body axis to be applied to quadrotor body in order to minimise the difference between $\mathbf{R}$ and $\mathbf{R}^\star$, thus:

$$\boldsymbol{\alpha}^\star = -\mathbf{k_p}\mathbf{e_R} - \mathbf{k_d}\mathbf{e_\Omega}, \tag{3.33}$$

where $\mathbf{k_p}, \mathbf{k_d} \in \mathbb{R}^3$ are non-negative gain vectors, can be tuned, depending on the aggressiveness of the required manoeuvre.

Therefore, based on (3.27), the desired moment, $\mathbf{M}^\star \in \mathbb{R}^3$, to be generated onto quadrotor body can be computed by:

$$\mathbf{M}^\star = \mathbf{J}^{-1}\boldsymbol{\alpha}^\star, \tag{3.34}$$

And then we add total thrust control. Thus, by combining (3.20) and (3.22) we can say:

$$\begin{bmatrix} \mathbf{M}^\star \\ \frac{1}{4}T^\star_{total} \end{bmatrix} = \begin{bmatrix} \mathbf{B} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{C} \\ \mathbf{1}_{1\times4} \end{bmatrix} \mathbf{T}^\star, \tag{3.35}$$

where matrix $\mathbf{B}$ and $\mathbf{C}$ are defined in (3.23) and (3.24) respectively, and $\mathbf{T}^\star = (T_1^\star, T_2^\star, T_3^\star, T_4^\star)^\top$ is desired thrust vector, which represents the desired thrust command to each rotor. Therefore, the thrust command for individual rotors can be computed by reversing (3.35):

$$\mathbf{T}^\star = \begin{bmatrix} \mathbf{C} \\ \mathbf{1}_{1\times4} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{M}^\star \\ \frac{1}{4}T^\star_{total} \end{bmatrix}. \tag{3.36}$$

This expression of thrust commands not only applies the desired moment to the quadrotor, but also ensures the total thrust provided by the four rotors is equal to $T^\star_{total}$.

Finally, to generate the normalised PWM signal command, $\delta^\star_n$, for individual rotor, simply apply inverted (3.16) on $T^\star_n$. Then we get:

$$\delta^\star_n = \sqrt{\frac{T^\star_n}{c_T}} + c_o. \tag{3.37}$$

This nonlinear attitude tracking controller is demonstrated to recover from any initial orientation in simulation by Lee et al. (2010a), and it is proved to have exponential stability when the initial attitude error is less than 90°, and it yields almost global exponentially attractiveness when the initial attitude error is less than 180°.

3.4.2 Acceleration controller

The Acceleration controller receives the desired acceleration command vector, $\mathbf{a}^\star = (a^\star_x, a^\star_y, a^\star_z)^\top \in \mathbb{R}^3$, from the position controller output, and quadrotor heading command, $proj[\mathbf{x_b}]$, from user. It then converts the commands into the desired orientation, $\mathbf{R}^\star$, and desired total thrust, $T^\star_{total}$, commands for attitude controller.

For a given acceleration command, $\mathbf{a}^\star$, based on (3.26) and (3.21), the desired thrust force acts on the quadrotor in world frame, $\mathbf{F}^\star_{\mathbf{thrust}}$, can be computed as:

$$\mathbf{F}^\star_{\mathbf{thrust}} = m\mathbf{a}^\star - \mathbf{F}_{\mathbf{gravity}}. \tag{3.38}$$

Based on (3.19), $\mathbf{F}^\star_{\mathbf{thrust}}$ is shown to have the same direction with the desire body Z_b–axis, $\mathbf{z}^\star_\mathbf{b}$, with the magnitude equals to desired total thrust, $T^\star_{total}$. Thus:

$$T^\star_{total} = \|\mathbf{F}^\star_{\mathbf{thrust}}\|, \tag{3.39}$$

$$\mathbf{z}^\star_\mathbf{b} = \frac{\mathbf{F}^\star_{\mathbf{thrust}}}{T^\star_{total}}. \tag{3.40}$$

Then by assuming the desire heading command, $proj[\mathbf{x_b}]$, is not parallel to $\mathbf{z}^\star_\mathbf{b}$, we can obtain the unit axis vectors $\mathbf{y}^\star_\mathbf{b}$ and $\mathbf{x}^\star_\mathbf{b}$ by:

$$\mathbf{y_b^\star} = \frac{\mathbf{z_b^\star} \times proj[\mathbf{x_b}]}{\|\mathbf{z_b^\star} \times proj[\mathbf{x_b}]\|}, \tag{3.41}$$

$$\mathbf{x_b^\star} = \mathbf{y_b^\star} \times \mathbf{z_b^\star}. \tag{3.42}$$

Therefore, the output desired quadrotor rotation matrix will be:

$$\mathbf{R}^\star = \begin{bmatrix} \mathbf{x_b^\star} & \mathbf{y_b^\star} & \mathbf{z_b^\star} \end{bmatrix}. \tag{3.43}$$

However, when the acceleration command $\mathbf{a}^\star$ given to the acceleration controller results in a desired total thrust, $T^\star_{total}$, which is higher than 80% of the maximum total thrust, $T_{80\%}$, that the four rotors can provide, the system will encounter the tracking instability.

Therefore, an acceleration limit must be conducted in this case. Here, we scale down $\mathbf{a}^\star$ to $\widetilde{\mathbf{a}}^\star = \beta\mathbf{a}^\star$, with $0 < \beta < 1$, so that the quadrotor body acceleration maintain the same direction, while the resulting total thrust is equal to $T_{80\%}$. Thus, in other words:

$$\mathbf{F^\star_{thrust}} = m\widetilde{\mathbf{a}}^\star - \mathbf{F_{gravity}}, \tag{3.44}$$

$$T_{80\%} = \|\mathbf{F^\star_{thrust}}\|, \tag{3.45}$$

$$\widetilde{\mathbf{a}}^\star = \beta\mathbf{a}^\star, \tag{3.46}$$

which results in a quadratic equation of β, in the form of:

$$a\beta^2 + b\beta + c = 0, \tag{3.47}$$

where:

$$a = \|m\mathbf{a}^\star\|^2, \tag{3.48}$$

$$b = 2m^2 g a^\star_z, \tag{3.49}$$

$$c = m^2 g^2 - T^2_{80\%}. \tag{3.50}$$

Thus β can be obtained from the quadratic formula $\beta = \frac{-b+\sqrt{b^2-4ac}}{2a}$. Then $\mathbf{a}^\star$ can be replaced by $\widetilde{\mathbf{a}}^\star$, $T^\star_{total} = T_{80\%}$, and $\mathbf{R}^\star$ can be computed in the same form as (3.40), (3.41), (3.42), (3.43).

3.4.3 Velocity controller

The velocity controller outputs the desired acceleration vector, $\mathbf{a}^\star$, to the acceleration controller, in order to minimises the error between desired velocity, $\dot{\mathbf{p}}^\star_\mathbf{w}$, commanded from the user, and quadrotor velocity measurement, $\dot{\mathbf{p}}_\mathbf{w}$, in the world frame.

Table 3.1: Platform details.

Quantity name	**Value**	**Unit**
Takoff mass(m)	758	g
Arm Length (l)	180	mm
Propeller size	9.4×5.0	$inch$
Motor Kv	960	RPM/V

The position error vector, $\mathbf{e_p}$, and velocity error vector, $\mathbf{e_v}$, are defined as:

$$\mathbf{e_p} = \mathbf{p_w} - \mathbf{p_w^\star}, \tag{3.51}$$

$$\mathbf{e_v} = \dot{\mathbf{p}}\mathbf{_w} - \dot{\mathbf{p}}\mathbf{_w^\star}. \tag{3.52}$$

Then proportional-integral-acceleration (PI-A) control law is applied, which yields an expression as:

$$\mathbf{a}^\star = -\mathbf{k_i'}\mathbf{e_p} - \mathbf{k_p'}\mathbf{e_v} - \mathbf{k_a'}\ddot{\mathbf{p}}\mathbf{_w}, \tag{3.53}$$

where $\mathbf{k_i'}$, $\mathbf{k_p'}$ and $\mathbf{k_a'}$ are non-negative controller gain vectors, which can be tuned according to the require aggressiveness of the position tracking performance. $\dot{\mathbf{p}}\mathbf{_w^\star}$ is from user command and $\dot{\mathbf{p}}\mathbf{_w}$ is measured from velocity estimator described in next section. $\mathbf{p_w^\star}$ and $\mathbf{p_w}$ are obtained by integrating $\dot{\mathbf{p}}\mathbf{_w^\star}$ and $\dot{\mathbf{p}}\mathbf{_w}$, and $\ddot{\mathbf{p}}\mathbf{_w}$ is obtained directly from accelerometer measurement.

Additionally we add a position-error-integral term with gain value k_h' for z-axis position controller to remove altitude control steady-state error.

3.5 Implementation

This section summarises the implementation details for the working UAS system, including mechanical setup, and autopilot electronics and software description. The quadrotor basic details are summarised in Table. 3.1.

3.5.1 Chassis and propulsion system

We selected a GF360 carbon fibre quadrotor frame[2] for industrial standard design suitable for fast prototyping applications. The 360 mm motor span optimised for 9.4 $inch$ propeller, here we use DJI E310 propulsion system[3] for robustness and simplicity. This results in a 600 mm tip-to-tip span, which is almost the maximum safe size to manoeuvre through standard UK doorways (762 mm width).

[2] http://www.dhgate.com/product/gf-360-carbon-fiber-folding-four-axis-quadcopter/187566640.html

[3] http://www.dji.com/product/e310

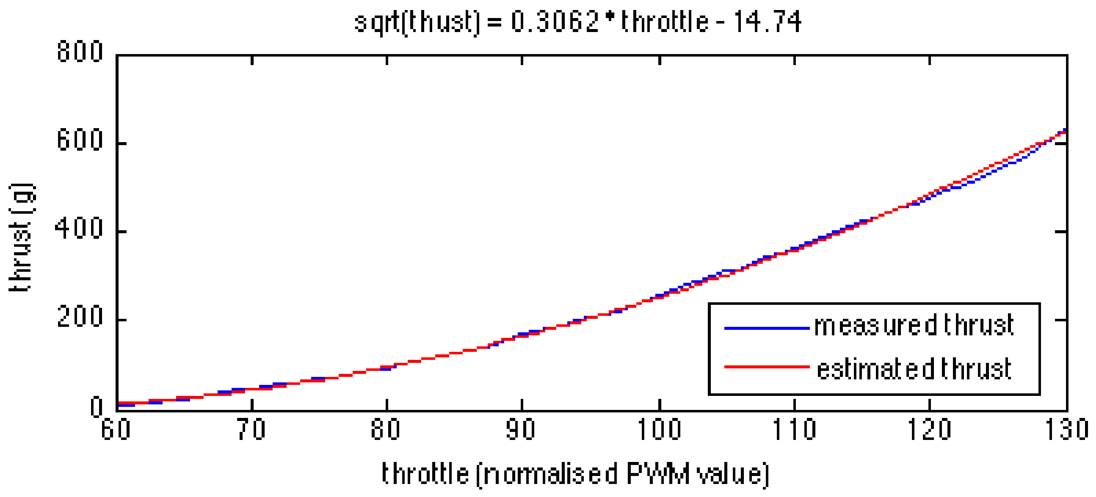


Figure 3.6: Propulsion system model.

We then conducted bench static thrust test of the E310 propulsion system and obtained c_T and c_o in (3.16) by applying linear regression on square root of T versus δ. The obtained c_o is 48.1385 and c_T is 0.09376, which gives 95% accuracy, as shown in Fig. 3.6, where the blue curve is the measured data from thrust test, and the red curve is generated from (3.16) with c_T and c_o equal to above values.

3.5.2 Moment of inertia

The moment of inertia of the quadrotor is obtained by averaging between analytical method and direct measurement.

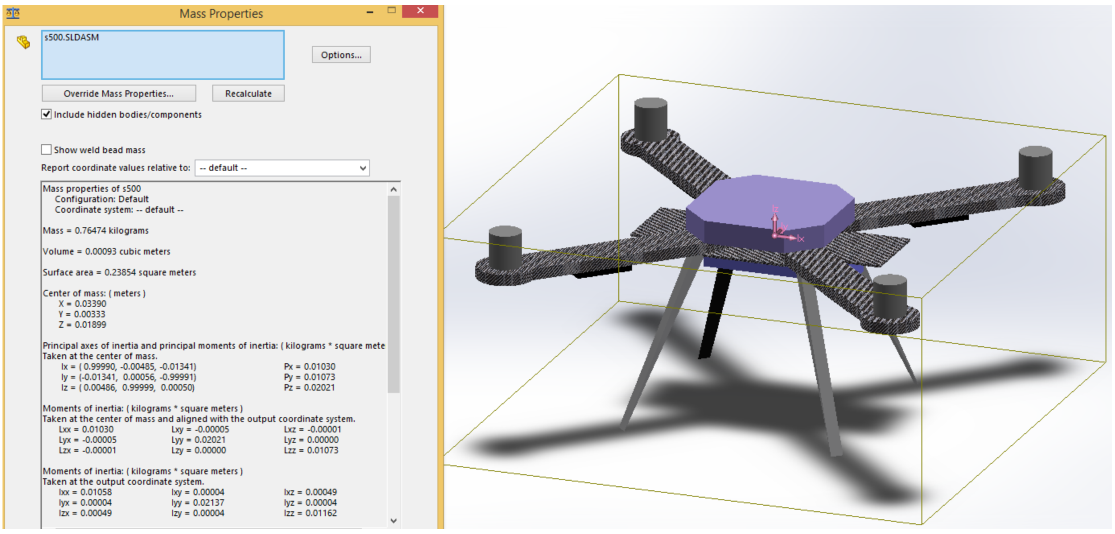
In order to obtain the analytical measurement of the moment of inertia, an approximated CAD model is constructed in SolidWorks®, with the mass assigned to all components individually. The constructed model is shown in Fig. 3.7.

The direct measurement of moment of inertia is obtained by applying the bifilar pendulum theory, where for each axis, the moment of inertia, J, can be computed by measuring the twist oscillation period with the setup as shown in Fig. 3.8, the equation used to compute the moment of inertia is:

$$J = \frac{mgT^2b^2}{4\pi^2L}, \tag{3.54}$$

where T is the period measured over one oscillation. Here, to improve the accuracy, the averaged period from 40 oscillations is obtained with manual stopwatch, and L, b is indicated in Fig. 3.8.

The results obtained from the CAD simulation, bifilar pendulum experiment and the average of both are summarised in Table 3.2. Note that the averaged value is used as the final measurement result.


Mass Properties
s500.SLDASM
Options...
Override Mass Properties...
Recalculate
☑ Include hidden bodies/components
☐ Show weld bead mass
Report coordinate values relative to: -- default --

```
Mass properties of s500
    Configuration: Default
    Coordinate system: -- default --

Mass = 0.76474 kilograms

Volume = 0.00093 cubic meters

Surface area = 0.23854 square meters

Center of mass: ( meters )
     X = 0.03390
     Y = 0.00333
     Z = 0.01899

Principal axes of inertia and principal moments of inertia: ( kilograms * square mete
Taken at the center of mass.
      Ix = ( 0.99990, -0.00485, -0.01341)                Px = 0.01030
      Iy = (-0.01341,  0.00056, -0.99991)                Py = 0.01073
      Iz = ( 0.00486,  0.99999,  0.00050)                Pz = 0.02021

Moments of inertia: ( kilograms * square meters )
Taken at the center of mass and aligned with the output coordinate system.
     Lxx = 0.01030           Lxy = -0.00005          Lxz = -0.00001
     Lyx = -0.00005          Lyy = 0.02021           Lyz = 0.00000
     Lzx = -0.00001          Lzy = 0.00000           Lzz = 0.01073

Moments of inertia: ( kilograms * square meters )
Taken at the output coordinate system.
     Ixx = 0.01058           Ixy = 0.00004           Ixz = 0.00049
     Iyx = 0.00004           Iyy = 0.02137           Iyz = 0.00004
     Izx = 0.00049           Izy = 0.00004           Izz = 0.01162
```

Figure 3.7: Quadrotor CAD model.

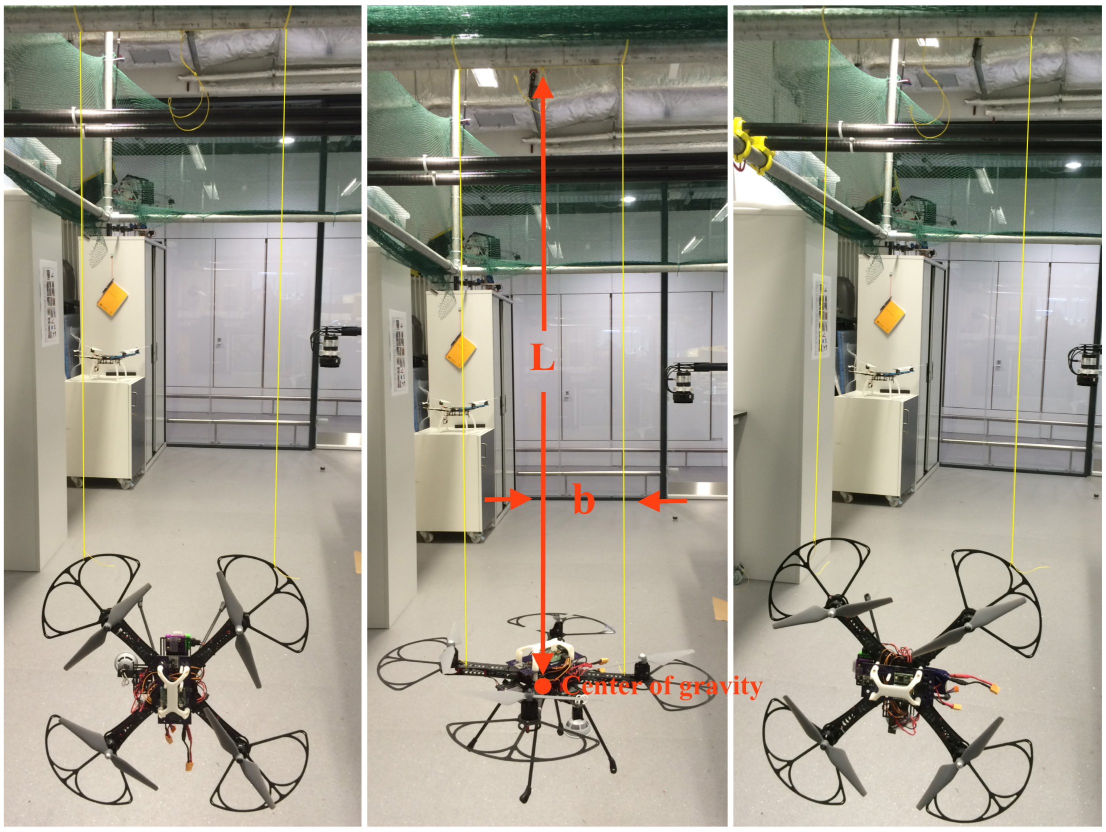


Figure 3.8: The bifilar pendulum experiment setup for the three body axis.

Table 3.2: Moment of inertia experiment results.

Quantity name	**Value from CAD**	**Value from Bi-filar Pendulum**	**Averaged Value**	**Unit**
J_X	0.01030	0.00875	0.00953	$kg \times m^2$
J_Y	0.02021	0.01863	0.01942	$kg \times m^2$
J_Z	0.01073	0.00921	0.00997	$kg \times m^2$

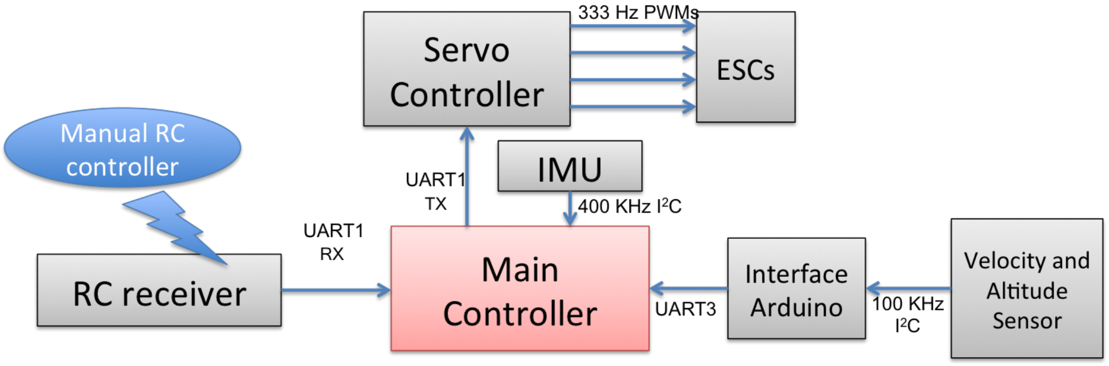
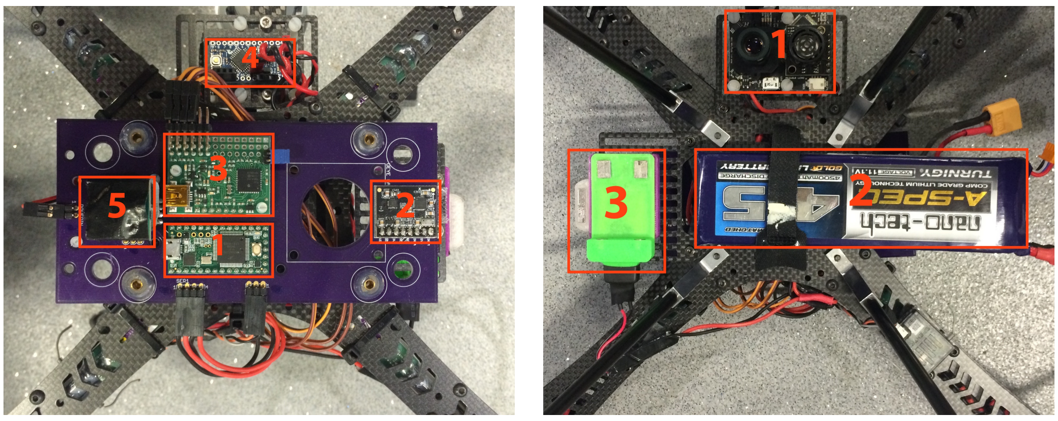


Figure 3.9: Block diagram.



(a) Top view.
1. Main Teensy controller;
2. FreeIMU;
3. Servo controller;
4. Interface Arduino;
5. Voltage regulator.

(b) Bottom view.
1. PX4Flow camera;
2. Li-Po battery.
3. Interface to additional camera payload.

Figure 3.10: Mechanical system layout.

3.5.3 Flight controller implementation

Thanks to the high speed Teensy 3.1 processor and a dedicated servo controller, the control loop implementing Section 3.4 executes within 3 ms. The physical layout is shown in Fig. 3.10a and Fig. 3.10b, and the block diagram in Fig. 3.9 shows the interactions between components.

1. **Main Controller Board** is based on Teensy 3.1 MCU board[4]. It is an ARM based Arduino compatible development board, which features very small form factor (35×18 mm) and fast processor (ARM Cortex–M4 with up to 96 MHz clock speed). It is ideal for a flight controller.

2. **IMU** is based on FreeIMU sensor suite[5] Varesano (2013), including a MPU6050 gyroscope-accelerometer combo-chip, a HMC5883L magnetometer and MS5611-01BA high resolution pressure sensor. However, only the MPU6050 chip is used in this implementation. The orientation fusion estimation uses the library provided with the sensor.

3. **Servo Controller** is based on Pololu Mini Maestro Servo Controller board[6]. It is a dedicated servo controller board, which features high resolution (0.25 μs) servo PWM output to 12 channels, with update rate up to 333 Hz, and the fast UART Serial protocol makes it easy to receive command from the main controller board.

4. **Ground Station Software** was developed in MATLAB® for graphical user interface (GUI) along with the quadrotor development. As shown in Figure 3.11, it was developed for the purpose of monitoring real time sensor measurement, 3D data visualisation, in-air parameter tuning, flight data logging and post-processing. The wireless communication is realised by XBee low power RF module. A customised bidirectional serial protocol is developed for reliable transmission and minimised data package. Two frame bytes are used at beginning and end of the package, also one byte checksum is used for each data package. A hand-shake procedure is followed for in-air parameter tuning, so that data transmission only happens when updating new parameters.

3.5.4 Velocity and altitude estimator implementation

The horizontal position is obtained by integrating the horizontal velocity. The horizontal velocity of the vehicle is obtained by fusing the measurements from PX4Flow Honegger et al. (2013) camera[7] and IMU. The vertical position is directly measured by the ultrasonic sensor on PX4Flow camera. On the PX4Flow, the CMOS high speed vision sensor with 21 degree field of view, measures the optical flow at 100 Hz. Then it obtains the ground velocity relative to quadrotor by scaling the average optical flow by the ground distance. Moreover, by subtracting the scaled gyroscope rate, thus it compensates for the optical flow caused by roll and pitch rotation.

In particular, the PX4Flow camera measures the horizontal velocity, $\mathbf{v_c} = [v_{cx}, v_{cy}, 0]^\top$ and vertical position, h. Note that $\mathbf{v_c}$ is always measured with respect to vehicle heading, $proj[\mathbf{x_b}]$,

[4]https://www.pjrc.com/teensy/teensy31.html
[5]http://www.varesano.net/projects/hardware/FreeIMU
[6]http://www.pololu.com/product/1352
[7]https://pixhawk.org/modules/px4flow

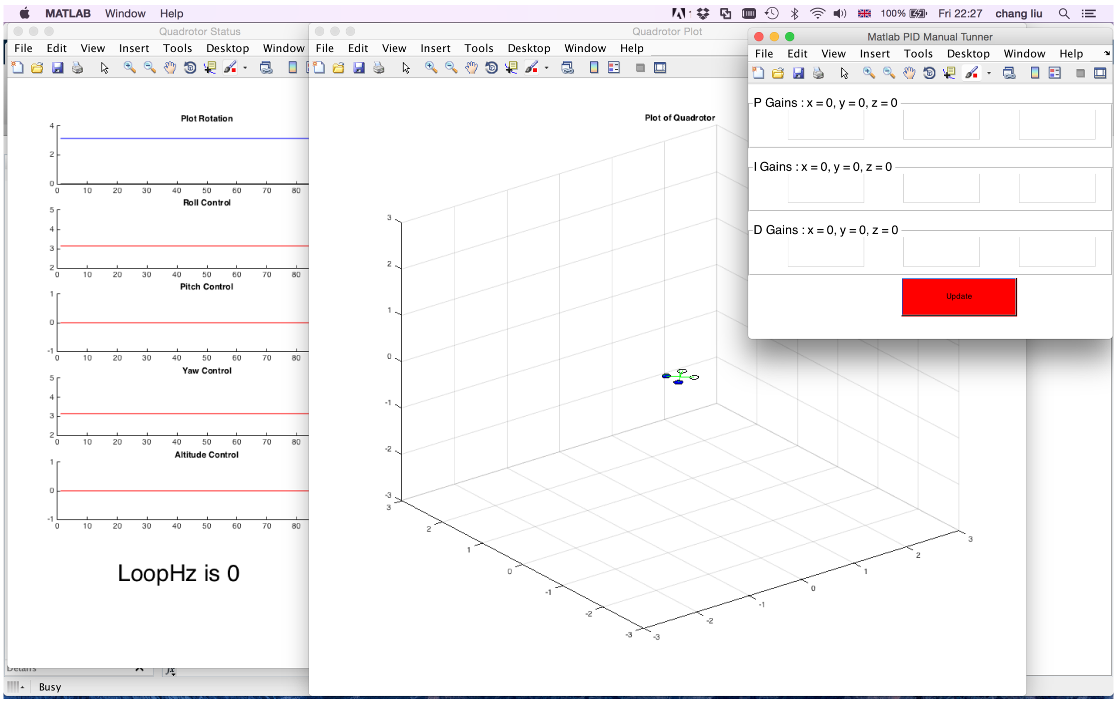


Figure 3.11: MATLAB® ground station graphical user interface.

thus the measured vehicle horizontal velocity $\mathbf{v}'_{\mathbf{w}}$ is:

$$\mathbf{v}'_{\mathbf{w}} = proj[\mathbf{x_b}] \odot \mathbf{v_c}, \tag{3.55}$$

where $\odot$ represents vector element-wise multiplication.

Then, given the IMU sampling time, ΔT, and measured body acceleration, $\mathbf{a}$, in body frame from accelerometer, we apply multiple rate complementary filter to compute the estimated vehicle horizontal velocity $\mathbf{v_w}^t = [v_{wx}, v_{wy}, v_{wr}]^\top$ at time t

$$\mathbf{v_w}^t = n(\mathbf{v_w}^{t-1} + \Delta T\mathbf{R}\mathbf{a}) + (1-n)\mathbf{v}'_{\mathbf{w}}, \tag{3.56}$$

where the coefficient n can be computed by

$$n = \frac{\tau}{\tau + T}. \tag{3.57}$$

Thus, τ is the time constant for the complementary filter. Therefore the vehicle position $\mathbf{p_w}$ can be computed by

$$\dot{\mathbf{p_w}} = [v_{wx}, v_{wy}, \dot{h}]^\top. \tag{3.58}$$

The filter executes at IMU sampling frequency, and $\mathbf{v}'_{\mathbf{w}}$ remains the latest measurement value before new velocity is updated from PX4Flow camera.

Table 3.3: Components cost summary with retail prices.

Item	Price (£)
Teensy 3.1 Main Controller	15
GF360 Frame	64.9
Turnigy 4500mah 3S Battery	37.8
DJI E310 Propulsion System	168.7
FreeIMU 4.0.3	45
Pololu Maestro Servo Controller	39.1
Optical Flow Camera (PX4FLOW)	120.8
Voltage Regulator	11.8
Prototype PCB Manufacture	25.7
Total	**£529**

Table 3.4: PID parameters.

Controller	x-axis	y-axis	z-axis
$\mathbf{k_p}$	2200	2200	15
$\mathbf{k_d}$	460	460	10
$\mathbf{k_i'}$	0.7	0.7	1.4
$\mathbf{k_p'}$	3.2	3.2	2.3
$\mathbf{k_a'}$	0.45	0.45	0.5
k_h'	–	–	0.7

3.5.5 Components cost summary

Table 3.3 summarises the cost of individual onboard components. The total cost of the entire system is £529, which is indicated in the last row. Note that the cost will be significantly reduced with higher quantity production.

3.6 Test Results

An indoor flight test was conducted as indicated in Fig. 3.12. The manual tuning was conducted in advance of this trial and the following tuning parameters in Table. 3.4 were used.

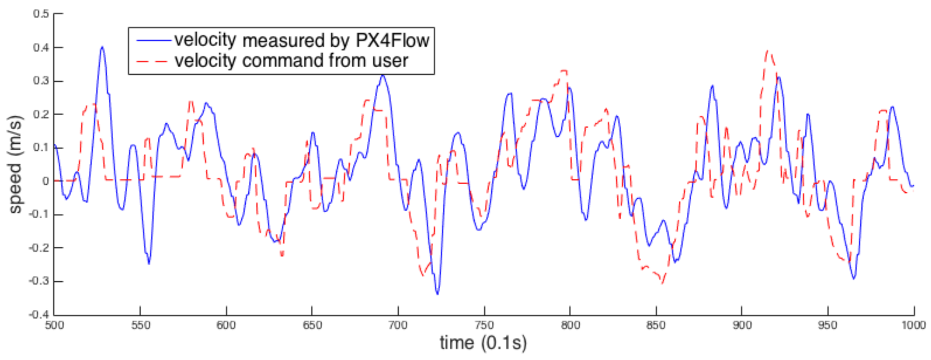
Flight data was recorded to demonstrate the control performance and validate the theory. Fig. 3.13 and Fig. 3.14 show the command-response graphs of terms directly controlled by the user, including horizontal velocity, altitude and heading angle, over 50 seconds duration (from $50s$ to $100s$). Fig. 3.13a and Fig. 3.13b shows the horizontal velocity can be effectively controlled within $\pm 0.2\ m/s$ accuracy and $0.5\ s$ response time. Due to the position feedback error introduced in the velocity controller in (3.51), the steady state velocity error was completely removed, since the position feedback error acts as the integral of the velocity feedback error. Moreover, it acts as the position hold effect when encounter an external turbulence. Fig. 3.14a shows that the altitude is controlled within $\pm 0.15\ m$ accuracy. Fig. 3.14b shows the heading angle is controlled within $\pm 0.1\ rad$ with a small steady-state error as a result of the PD attitude

Figure 3.12: Indoor flight test scene.

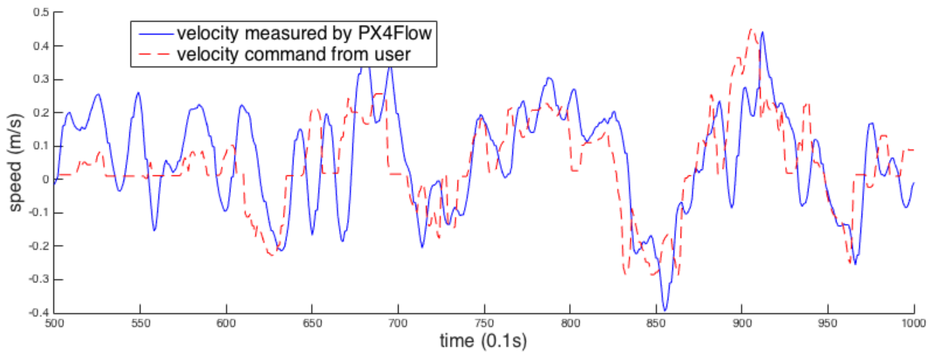
controller.

Moreover, to verify the cascaded control architecture, Two 28-second command-response graphs are shown in Fig. 3.15 ($122s$ to $150s$), indicating the velocity controller performance and the intermediate control signal between acceleration controller and attitude controller at the corresponding time. Note that we have rotated the velocity in the world frame to match the quadrotor heading so that the pitch angle of the quadrotor will result in the change in x-axis velocity change in match-heading frame.

It is clearly shown that the output from the acceleration controller (dashed in Fig. 3.15b reacts to the velocity error (indicated as the difference between dashed and solid in Fig. 3.15a), and acts as the command input to the attitude controller, although the output from acceleration controller also reacts to the acceleration measured directly by the accelerometer, which is not shown in the graph. Moreover, Fig. 3.15b also shows that there is significant offset (steady-state error) as expected from the PD attitude controller design with an imperfect mass balance of quadrotor body, which has been sufficiently compensated by the higher level velocity controller. It is indicated by the resulting pitch angle (solid in Fig. 3.15b centred at $0\ rad$.

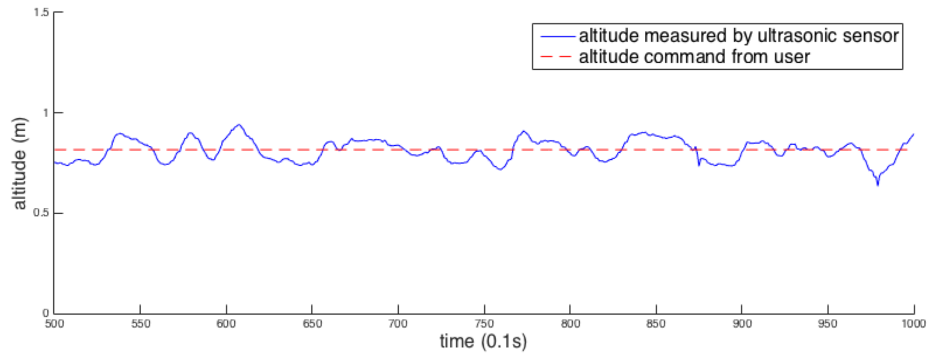


(a) Command-response graph of velocity in x-axis in the world frame.

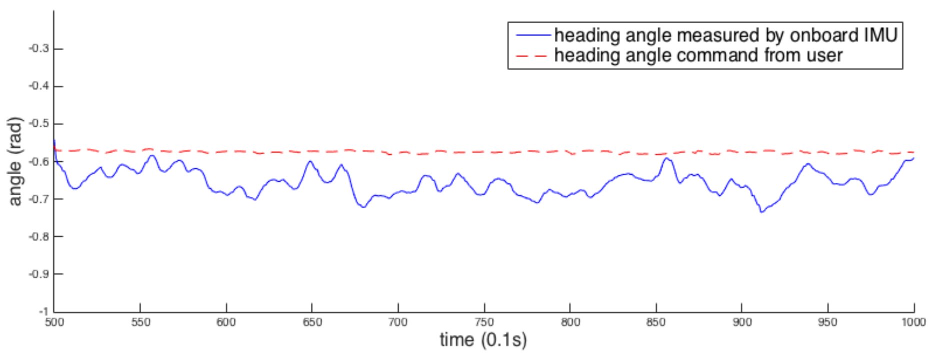


(b) Command-response graph of velocity in y-axis in the world frame.

Figure 3.13: Velocity control performance evaluation graphs.

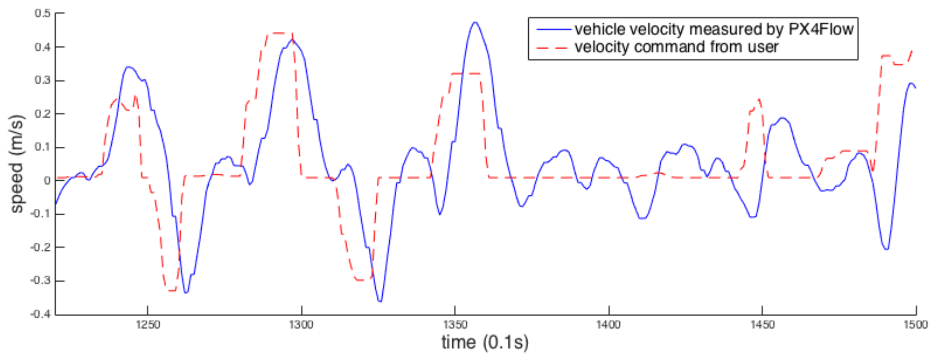


(a) Command-response graph of altitude.

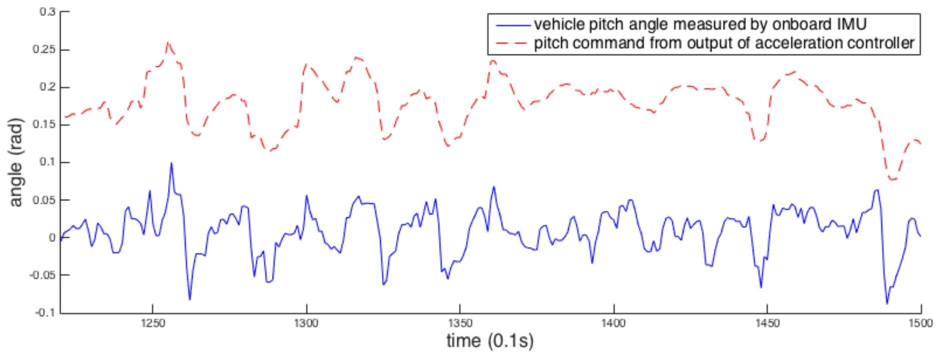


(b) Command-response graph of the heading angle.

Figure 3.14: Altitude and heading control performance evaluation graphs.



(a) Short-term command-response graph of velocity in x-axis in match-heading frame.



(b) Short-term command-response graph of pitch angle.

Figure 3.15: Cascaded control validation graphs.

Moreover, a noticeable delay (about 0.5 s) in the velocity responds graphs (Fig. 3.13a, Fig. 3.13b and Fig. 3.15a) were introduced by the physical inertia of the UAV after for acceleration.

3.7 Conclusion

This chapter has shown the quadrotor modelling and controller design principle, as well as implementation details. The flight test result showed a good attitude and altitude hold and an acceptable velocity control performance. The cascaded control architecture of the developed quadrotor is suitable for testing vision based localization algorithms, and testing new control strategies. The fully customised design makes it easy to integrate new sensors and manipulating controller. The further work includes implementing simultaneous localisation and mapping algorithm to provide positional feedback, and fine tuning the controller parameters.

Chapter 4

Onboard Sensor Fusion for State Estimation

Because of the complementary nature of visual and inertial sensors, the combination of both is able to provide fast and accurate six degree-of-freedom (6 DOF) state estimation, which is the fundamental requirement for robotic (especially unmanned aerial vehicle) navigation tasks in GPS-denied environments. This chapter presents a computationally efficient visual-inertial fusion algorithm, by separating orientation fusion from the position fusion process. The algorithm is designed to perform 6 DOF state estimation, based on a gyroscope, an accelerometer and a monocular visual-based simultaneous localisation and mapping (mSLAM) algorithm measurement. It also recovers the visual scale for the mSLAM. In particular, the fusion algorithm treats the orientation fusion and position fusion as two separate processes, where the orientation fusion is based on a very efficient gradient descent algorithm, whereas the position fusion is based on a 13-state linear Kalman filter. The elimination of the magnetometer sensor avoids the problem of magnetic distortion, which makes it a power-on-and-go system once the accelerometer is factory calibrated. The resulting algorithm shows a significant computational reduction over the conventional extend Kalman filter, with sub-centimetre accuracy. Moreover, the separation between orientation and position fusion processes enables the algorithm to be easily implemented onto two individual hardware elements and thus allows the two fusion processes to be executed concurrently.

Part of this chapter was published as Liu, C., Prior, S. D., Teacy, W. L., and Warner, M. Computationally efficient visual- inertial sensor fusion for Global Positioning System-denied navigation on a small quadrotor. *Advances in Mechanical Engineering*, 8(3):1–11, DOI: 10.1177/1687814016640996 (Liu et al., 2016b)

The position fusion algorithm is open sourced in the link:

`https://github.com/Changliu52/vi_ekf.git`

4.1 Introduction

The combination of visual and inertial sensors has been shown to be viable, and the significant performance improvement over a single sensor system has attracted many researchers into the field after the success of SFly projectWeiss and Siegwart (2011), which enabled the world's first autonomous unmanned aerial vehicle (UAV) in GPS-denied environments, as stated by Blösch et al. (2010).

In the past five years, many prominent research institutions began to develop advanced monocular visual-based simultaneous localization and mapping (mSLAM) algorithms based on structure from motion (SFM) theory Engel et al. (2014, 2013); Forster et al. (2014); Klein and Murray (2007); Montemerlo et al. (2003); Newcombe et al. (2011b); Pizzoli et al. (2014); Roussillon et al. (2011); Vogiatzis and Hernández (2011), which are suitable to modern onboard embedded computers. Moreover, the visual scale problem, which was the main challenge of involving monocular vision into the control loop, has been addressed by fusing onboard inertial measurements (accelerometer and gyroscope), called the visual inertial navigation system (VINS) Dunkley et al. (2014); Jones and Soatto (2011); Kelly and Sukhatme (2009, 2016); Li and Mourikis (2013); Lobo and Dias (2003); Lynen et al. (2013); Shen et al. (2013b); Weiss et al. (2012a).

Almost all of the visual-inertial fusion algorithms, to our knowledge, rely on nonlinear Kalman filter techniques (extended Kalman filter, unscented Kalman filter, etc.) to process both the orientation and the position measurement in the same process, this results in a large state vector (usually more than 20 states) and a complex nonlinear system model. However, recent advances in computationally efficient inertial measurement unit (IMU) orientation estimation, Madgwick et al. (2011a), shows a competitive accuracy against Kalman-based algorithms by utilising optimisation based methods. Thus, in this chapter, a computationally efficient visual-inertial fusion algorithm is proposed by separating the orientation and the position fusion processes, this maintains the same level of accuracy with nonlinear Kalman filter approach. The algorithm is designed to perform a six degree of freedom state estimation, based on a gyroscope, an accelerometer and a mSLAM measurement. It also recovers the visual scale for the mSLAM.

The rest of this chapter is organised as follows: Section "Algorithm Preliminaries" gives an overview of the visual-inertial fusion algorithm; section "Orientation Fusion Process" presents the mathematical expression of the orientation filter, and section "Position Fusion Process" presents the mathematical expression of the position filter. The implementation, test result and conclusion are shown in the last three sections.

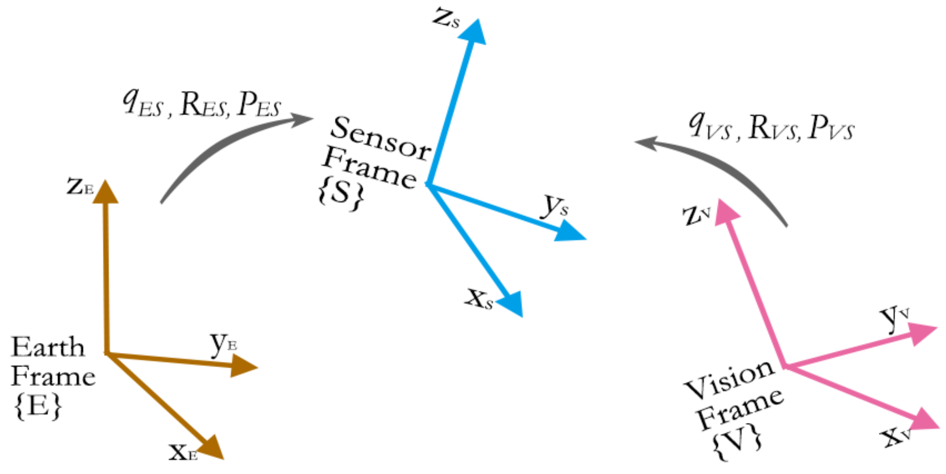


Figure 4.1: Coordinate system.

4.2 Algorithm Preliminaries

4.2.1 Coordinate system

The coordinate system used is shown in Fig.4.1. All the coordinate frames are defined following the right hand rule. The earth frame $\{E\}$ is fixed to the world, and z_E–axis is defined to be parallel to gravity vector. The sensor frame $\{S\}$ is shared by the gyroscope, the accelerometer and the camera, where x_S–axis points to sensor front, and z_S–axis points to sensor top. The vision frame $\{V\}$ is the world frame assumed in the mSLAM algorithm, in which the projection of x_v–axis on the x_E–y_E plane is parallel to x_E, and the orientation of z_V–axis is arbitrary depending on how the mSLAM is initialised. Note that there are two assumptions under this coordinate system: firstly, the origin of $\{E\}$ is assumed to be very close to the origin of $\{V\}$ (here we separate the two frames in Fig.4.1 for the sake of clearance); secondly, the x_V–axis is assumed to not be perpendicular to the x_E–y_E plane in $\{E\}$.

The orientation of $\{S\}$ with respect to $\{E\}$ can be expressed as $\boldsymbol{q}_{_{ES}} \in \mathbb{R}^4$ in quaternion or $R_{_{ES}} \in SO(3)$ in rotation matrix, and the position as $\boldsymbol{p}_{_{ES}} \in \mathbb{R}^4$. Similarly, the orientation of $\{S\}$ with respect to $\{V\}$ can be expressed as $\boldsymbol{q}_{_{VS}} \in \mathbb{R}^4$ or $R_{_{VS}} \in SO(3)$, and the position as $\boldsymbol{p}_{_{VS}} \in \mathbb{R}^4$.

4.2.2 Algorithm overview

As shown in Fig.4.2, the visual-inertial fusion algorithm assumes rotation $\boldsymbol{q}_{_{VS}}$, as well as the unscaled position $\overline{\boldsymbol{p}}_{_{VS}}$ are provided by a mSLAM algorithm, which is treated as a black box.

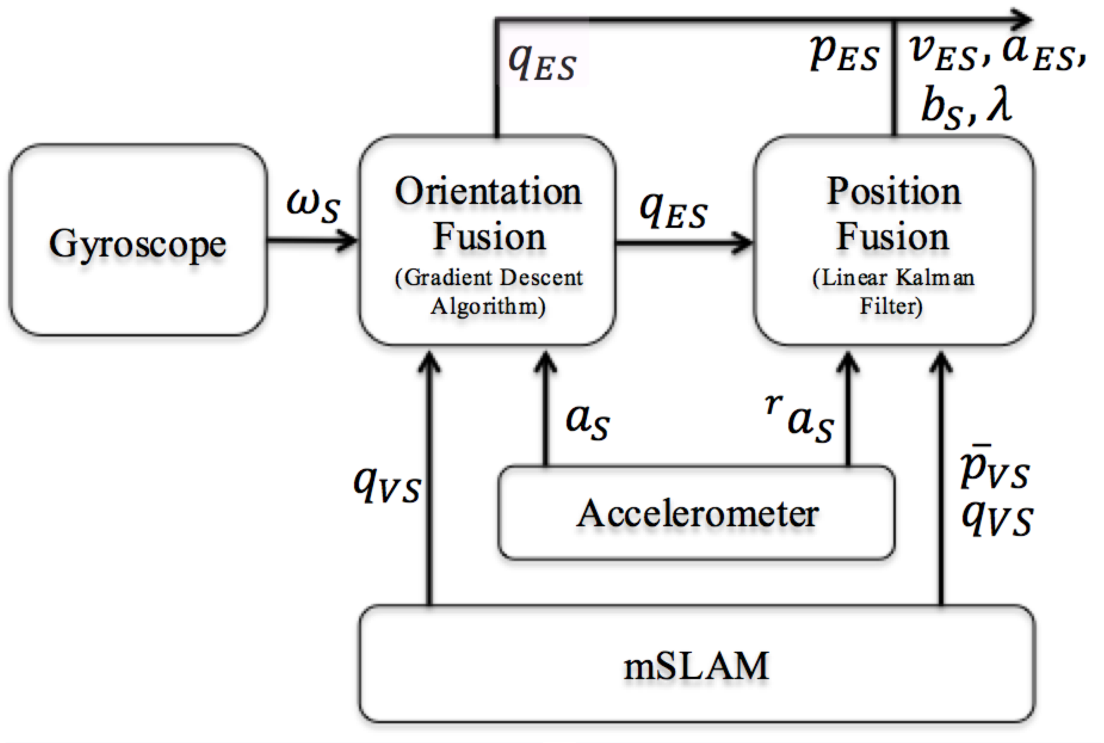


Figure 4.2: Algorithm overview.

Moreover, it receives angular rates measurement $\boldsymbol{\omega}_S \in \mathbb{R}^4$ from gyroscope, acceleration measurement $\boldsymbol{a}_S \in \mathbb{R}^4$ from accelerometer. The output of the fusion process is to estimate rotation $\boldsymbol{q}_{ES}$ and position $\boldsymbol{p}_{ES} \in \mathbb{R}^3$ of $\{S\}$ in $\{E\}$. Furthermore, the position filter also estimates the linear velocity $\boldsymbol{v}_{ES} \in \mathbb{R}^3$, linear acceleration $\boldsymbol{a}_{ES} \in \mathbb{R}^3$, and accelerometer bias $\boldsymbol{b}_{ES} \in \mathbb{R}^3$, as well as the metric scale of the mSLAM position measurement λ.

The fusion is separated into two fusion processes: orientation fusion process and position fusion process. The orientation fusion is based on very efficient gradient descent algorithm Madgwick et al. (2011a), and position fusion is based on a 13-state linear Kalman filter. The following two sections will present the mathematical expression of the two algorithms respectively.

4.3 Orientation Fusion Process

The orientation fusion algorithm fuses the gyroscope, accelerometer and vision (mSLAM) measurement to output a final orientation estimation. It is based on the gradient descent algorithm in quaternion representation. The origin of the algorithm comes from Madgwick et al. (2011a), where the detailed mathematical derivation and proof is presented. However, different from the original algorithm, the following fusion derivation eliminates the magnetometer sensor,

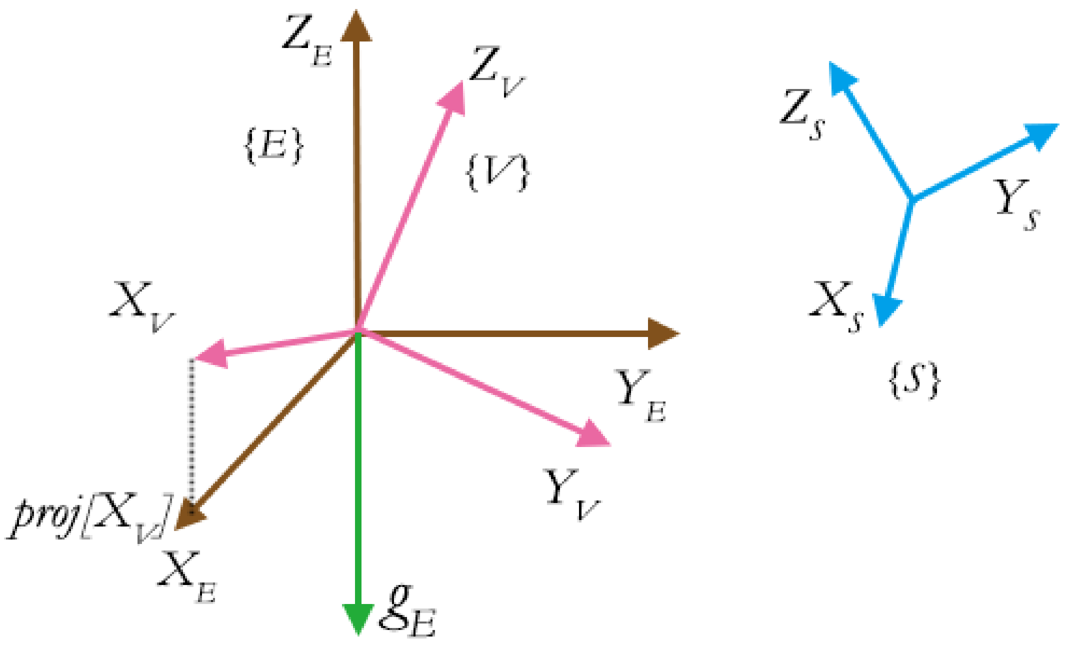


Figure 4.3: Gravity field and vision field.

while, instead, the rotation correction about gravity vector is compensated by fusing the vision (mSLAM) measurement. Therefore, it avoids the problem of magnetic field distortion, thus only factory calibration is required once for accelerometer.

Moreover, given that all the mSLAM orientation measurement tend to drift over time and distance due to the accumulated error, fusing the vision measurement in the same way as the magnetometer will result in the estimation error on gravity direction. This can be very sensitive for the quadrotor stabilisation and control. Thus, the following algorithm decouples vision measurement with the gravity direction estimation, while maintaining the effective fusion about the gravity vector.

In order to perform orientation estimation, as shown in Fig.4.3, three coordination frames are used: Sensor frame $\{S\}$ represents the orientation of all the coincide sensors (gyroscope, accelerometer and camera); Earth frame $\{E\}$ represents the reference frame of the inertial sensors (gyroscope and accelerometer); Vision frame $\{V\}$ represents the reference frame of the mSLAM algorithm based on the camera. Additionally, $\boldsymbol{g}_E$ is the unit vector representing the true gravity direction, which is also called gravity field vector in the rest of the chapter; and $proj[\boldsymbol{x}_V]$ is the unit projection vector of the x_V–axis onto the x_E–y_E plane, which is also called vision field vector in the rest of the chapter.

The purpose of the orientation fusion is to estimate optimal quaternion transformation $\boldsymbol{q}_{ES}$ from $\{E\}$ to $\{S\}$ so that, (1) the z_E–axis is align with the gravity field; (2) the x_v–axis is align with the vision field.

The essential mathematical expression of one iteration at time t is shown as follows. Note that the orientation estimation from last iteration $\boldsymbol{q}_{ES,t-1}$ is assumed to be known, and the sampling period is denoted as Δt.

4.3.1 Orientation derivative estimated by gyroscope

The 3-axis gyroscope measures the angular velocity (rate) in rad/s about the three axis of $\{S\}$ frame, which we denote as $\boldsymbol{\omega}_S = [0, \omega_x, \omega_y, \omega_z]^\top$. The quaternion derivative of the gyroscope estimation $\dot{\boldsymbol{q}}_{\omega,t}$ at time t can be computed by (4.1), given $\boldsymbol{q}_{ES,t-1}$ as the previous orientation estimation from all sensors.

$$\dot{\boldsymbol{q}}_{\omega,t} = \frac{1}{2}\boldsymbol{q}_{ES,t-1} \otimes \boldsymbol{\omega}_S, \tag{4.1}$$

where $\otimes$ denotes a quaternion product. Note that all the quaternions in this chapter follow $\boldsymbol{q} = [q_w, q_x, q_y, q_z]^\top$, and they are all unit quaternions ($q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1$).

4.3.2 Orientation optimisation from homogenous field

In order to obtain the optimal orientation estimation $\boldsymbol{q}_{ES} = [q_0, q_1, q_2, q_3]^\top$, the field measured from the sensor has to be aligned with the predefined reference field as close as possible. Thus, this can be formularised as an optimisation problem, where, for any homogenous field $\boldsymbol{b}_E \in \mathbb{R}^4$ in $\{E\}$, (4.2) is solved to minimise an objective error function.

$$\min_{\forall \boldsymbol{q}_{ES} \in \mathbb{R}^4} \boldsymbol{f}(\boldsymbol{q}_{ES}, \boldsymbol{b}_E, \boldsymbol{s}_S), \tag{4.2}$$

where $\boldsymbol{s}_S \in \mathbb{R}^4$ is the field vector measured by the corresponding sensor in $\{S\}$. The gradient descent algorithm is one of the most computationally efficient optimisation algorithms to solve the above problem. (4.3) describes it for n iterations, which starts from initial orientation estimation $\boldsymbol{q}_{ES,0}$ to final estimation $\boldsymbol{q}_{ES,n+1}$.

$$\boldsymbol{q}_{ES,k+1} = \boldsymbol{q}_{ES,k} - \mu \frac{\Delta \boldsymbol{f}(\boldsymbol{q}_{ES,k}, \boldsymbol{b}_E, \boldsymbol{s})}{\|\Delta \boldsymbol{f}(\boldsymbol{q}_{ES,k}, \boldsymbol{b}_E, \boldsymbol{s})\|}, k = 0, 1, 2..n \tag{4.3}$$

where μ is the an non-negative scalar, named as step-size, and the error direction $\Delta \boldsymbol{f}(\boldsymbol{q}_{ES,k}, \boldsymbol{b}_E, \boldsymbol{s})$ is computed by the objective error function $\boldsymbol{f}$ and its jacobian matrix $\boldsymbol{J}$.

$$\Delta \boldsymbol{f}(\boldsymbol{q}_{ES,k}, \boldsymbol{b}_E, \boldsymbol{s}) = \boldsymbol{J}^\top(\boldsymbol{q}_{ES,k}) \boldsymbol{f}(\boldsymbol{q}_{ES,k}, \boldsymbol{b}_E, \boldsymbol{s}) \tag{4.4}$$

4.3.2.1 Gravity field objective error function

The gravity field objective error functionrepresents the error between gravity vector in principle and the sensed gravity acceleration vector, expressed in $\{S\}$. Thus, given the gravity field vector $\boldsymbol{g}_E$, the gravity field objective error function $\boldsymbol{f}_g$ is defined as

$$\boldsymbol{f}_g(\boldsymbol{q}_{ES}, \boldsymbol{g}_E, \boldsymbol{a}_S) = \boldsymbol{q}^\star_{ES} \otimes \boldsymbol{g}_E \otimes \boldsymbol{q}_{ES} - \boldsymbol{a}_S, \tag{4.5}$$

where $\boldsymbol{q}^\star_{ES}$ is the conjugate of $\boldsymbol{q}_{ES}$, and $\boldsymbol{a}_S = [0, a_x, a_y, a_z]^\top$ is the normalised accelerometer measurement. Since $\boldsymbol{g}_E = [0, 0, 0, -1]^\top$, then it can be further simplified as (4.6).

$$\boldsymbol{f}_g(\boldsymbol{q}_{ES}, \boldsymbol{a}_S) = \begin{bmatrix} -2(q_1q_3 - q_0q_2) - a_x \\ -2(q_0q_1 + q_2q_3) - a_y \\ -2(\frac{1}{2} - q_1^2 - q_2^2) - a_z \end{bmatrix}. \tag{4.6}$$

Therefore, its jacobian matrix is

$$\boldsymbol{J}_g(\boldsymbol{q}_{ES}) = \begin{bmatrix} 2q_2 & -2q_3 & 2q_0 & -2q_1 \\ -2q_1 & -2q_0 & -2q_3 & -2q_2 \\ 0 & 4q_1 & 4q_2 & 0 \end{bmatrix}. \tag{4.7}$$

4.3.2.2 Vision field objective error function

The vision field objective error represents the difference between the vision field vector $proj[\boldsymbol{x}_V]$ and the x_E–axis of $\{E\}$ represented in $\{S\}$, thus the vision field objective error function $\boldsymbol{f}_V$ is defined as (4.8).

$$\boldsymbol{f}_V(\boldsymbol{q}_{ES}, \boldsymbol{x}_E, \boldsymbol{x}_{VS}) = \boldsymbol{q}^\star_{ES} \otimes \boldsymbol{x}_E \otimes \boldsymbol{q}_{ES} - \boldsymbol{x}_{VS}, \tag{4.8}$$

where $\boldsymbol{x}_{VS}$ is $proj[\boldsymbol{x}_V]$ represented in $\{S\}$ as shown in Fig.4.3. It is treated as a constant vector once been pre-computed by (4.9, 4.10, 4.11).

$$\boldsymbol{x}_V = \boldsymbol{q}^\star_{VS} \otimes (\boldsymbol{q}_{ES} \otimes \boldsymbol{x}_E \otimes \boldsymbol{q}^\star_{ES}) \otimes \boldsymbol{q}_{VS}, \tag{4.9}$$

$$proj[\boldsymbol{x}_V] = [0, \frac{x_{V1}}{x_{V1}^2 + x_{V2}^2}, \frac{x_{V2}}{x_{V1}^2 + x_{V2}^2}, 0], \tag{4.10}$$

$$\boldsymbol{x}_{VS} = \boldsymbol{q}^\star_{ES} \otimes proj[\boldsymbol{x}_V] \otimes \boldsymbol{q}_{ES}, \tag{4.11}$$

where $\boldsymbol{x}_V = [0, x_{V1}, x_{V2}, x_{V3}]^\top$ is the unit x_V–axis vector in $\{E\}$, and $proj[\boldsymbol{x}_V]$ is the normalised projection of $\boldsymbol{x}_V$ onto the x_E–y_E plane measured by mSLAM algorithm, which can be computed by (4.10). Since $\boldsymbol{x}_E = [0, 1, 0, 0]^\top$, it can then be simplified as

$$\boldsymbol{f}_V(\boldsymbol{q}_{ES}, \boldsymbol{x}_{VS}) = \begin{bmatrix} 2(\frac{1}{2} - q_2^2 - q_3^2) - x_{VS1} \\ 2(q_1 q_2 - q_0 q_3) - x_{VS2} \\ 2(q_0 q_2 + q_1 q_3) - x_{VS3} \end{bmatrix}, \tag{4.12}$$

where $\boldsymbol{x}_{VS} = [0, x_{VS1}, x_{VS2}, x_{VS3}]$. Thus, its jacobian matrix is

$$\boldsymbol{J}_V(\boldsymbol{q}_{ES}) = \begin{bmatrix} 0 & 0 & -4q_2 & -4q_3 \\ -2q_3 & 2q_2 & 2q_1 & -2q_0 \\ 2q_2 & 2q_3 & 2q_0 & 2q_1 \end{bmatrix}. \tag{4.13}$$

4.3.3 Fusion of three sensors

As stated in the gradient decent algorithm by Madgwick et al. (2011a), given that the convergence rate of the estimated orientation is equal or greater than the angular rate of the physical orientation, only one iteration is required to be computed per sample time, Δt. Therefore an unconventional gradient descent algorithm is derived to fuse all the three sensor measurements. To compute the orientation in next time stamp $\boldsymbol{q}_{ES,t+1}$, the process is summarised as

$$\boldsymbol{q}_{ES,t+1} = \boldsymbol{q}_{ES,t} + \dot{\boldsymbol{q}}_{ES,t+1}\Delta t, \tag{4.14}$$

$$\dot{\boldsymbol{q}}_{ES,t+1} = \dot{\boldsymbol{q}}_{\omega,t+1} - \beta\frac{\Delta \boldsymbol{f}}{\|\Delta \boldsymbol{f}\|}, \tag{4.15}$$

where $\frac{\Delta \boldsymbol{f}}{\|\Delta \boldsymbol{f}\|}$ can be assigned a physical meaning as the normalised direction of the error of $\dot{\boldsymbol{q}}_{ES,t+1}$, and it can be expressed as $\dot{\boldsymbol{q}}_{\epsilon,t+1}$,

$$\dot{\boldsymbol{q}}_{\epsilon,t+1} = \frac{\Delta \boldsymbol{f}}{\|\Delta \boldsymbol{f}\|}. \tag{4.16}$$

Moreover, β is the only adjustable parameter of this filter. It represents the magnitude of the gyroscope measurement error, which is removed in the direction according to the accelerometer and vision sensor. The higher β, the faster that the estimated orientation converge to the accelerometer estimation. The theoretically optimal value of β is

$$\beta = \sqrt{\frac{3}{4}}\tilde{\omega}_{max}, \tag{4.17}$$

where $\tilde{\omega}_{max}$ is the maximum gyroscope measurement error for each axis. Moreover, since IMU and the vision sensor operate in asynchronously, depending on which sensor measurement is available, $\Delta \boldsymbol{f}$ can then be expressed as:

$$\Delta \boldsymbol{f} = \begin{cases} \boldsymbol{J}_g^\top(\boldsymbol{q}_{ES})\boldsymbol{f}_g(\boldsymbol{q}_{ES}, \boldsymbol{a}_S) \\ \boldsymbol{J}_{g,V}^\top(\boldsymbol{q}_{ES})\boldsymbol{f}_{g,V}(\boldsymbol{q}_{ES}, \boldsymbol{a}_S, \boldsymbol{x}_{VS}), \end{cases} \tag{4.18}$$

where $\boldsymbol{f}_{g,V}(\boldsymbol{q}_{ES}, \boldsymbol{a}_S, \boldsymbol{x}_{VS})$ and $\boldsymbol{J}_{g,V}(\boldsymbol{q}_{ES})$ are the combined measurement of both field from the sensors, which can be expressed as:

$$\boldsymbol{f}_{g,V}(\boldsymbol{q}_{ES}, \boldsymbol{a}_S, \boldsymbol{x}_{VS}) = \begin{bmatrix} \boldsymbol{f}_g(\boldsymbol{q}_{ES}, \boldsymbol{a}_S) \\ \boldsymbol{f}_V(\boldsymbol{q}_{ES}, \boldsymbol{x}_{VS}) \end{bmatrix}, \tag{4.19}$$

$$\boldsymbol{J}_{g,V}(\boldsymbol{q}_{ES}) = \begin{bmatrix} \boldsymbol{J}_g(\boldsymbol{q}_{ES}) \\ \boldsymbol{J}_V(\boldsymbol{q}_{ES}) \end{bmatrix}. \tag{4.20}$$

4.3.4 Gyroscope bias online estimation

Given the fact that the gyroscope bias drifts with temperature and motion in practice, any high accuracy fusion algorithm must be able estimate the varying gyroscope bias online. Kalman based methods generally cope with this by introducing the bias variables into the state vector. However, a much more computationally efficient estimation method is used by DC component of the angular error feedback, similar with Madgwick et al. (2011b).

The normalised direction of the error in the rate of change of orientation, $\dot{\boldsymbol{q}}_\epsilon$, which is defined by (4.16), can be converted to the angular error $\boldsymbol{\omega}_\epsilon$ in $\{S\}$ frame by inverting (4.1). This yields

$$\boldsymbol{\omega}_{\epsilon,t} = 2\boldsymbol{q}^\star_{ES,t-1} \otimes \dot{\boldsymbol{q}}_{\epsilon,t}. \tag{4.21}$$

The gyroscope bias, $\boldsymbol{\omega}_b$, can then be represented as the DC component of $\boldsymbol{\omega}_\epsilon$, and thus can be removed from the gyroscope measurement, $\boldsymbol{\omega}_S$, as the integral of $\boldsymbol{\omega}_\epsilon$, weighted by a gain, ζ.

$$\boldsymbol{\omega}_{b,t} = \zeta \int \boldsymbol{\omega}_{\epsilon,t} \mathrm{d}t, \tag{4.22}$$

$$\boldsymbol{\omega}_{c,t} = \boldsymbol{\omega}_{s,t} - \boldsymbol{\omega}_{b,t}, \tag{4.23}$$

where $\boldsymbol{\omega}_c$ is the corrected gyroscope reading, thus it can be used to replace the raw gyroscope reading, $\boldsymbol{\omega}_s$ in (4.1).

Note here, the weighting factor, ζ, decides the convergence rate of the gyroscope bias estimation, where the higher the ζ is, the faster and noisier the convergence is. While the theoretical optimal value of ζ is defined as

$$\zeta = \sqrt{\frac{3}{4}}\dot{\tilde{\omega}}_{max} \tag{4.24}$$

where $\dot{\tilde{\omega}}_{max}$ is the maximum rate of change of the gyroscope measurement error of each axis.

4.4 Position Fusion Process

This position fusion algorithm assumes the orientation of the sensors is known, and only estimates the translational state of the system. being able to avoid estimating the orientation, not only reduces the length of the state vector significantly, but also keeps the system model almost linear. It takes three inputs: (1) the orientation estimation $\boldsymbol{q}_{_{ES}}$ in $\{E\}$ from the result of the orientation fusion process; (2) the raw sensor acceleration measurement ${}^{r}\boldsymbol{a}_{_S} \in \mathbb{R}^4$ in m/s^2 from accelerometer; (3) the unscaled position $\overline{\boldsymbol{p}}_{_{VS}} \in \mathbb{R}^4$ and orientation $\boldsymbol{q}_{_{VS}} \in \mathbb{R}^4$ in $\{V\}$ from the mSLAM. It outputs its state vector, which contains: position estimation $\boldsymbol{p}_{_{ES}} \in \mathbb{R}^3$, velocity estimation $\boldsymbol{v}_{_{ES}} \in \mathbb{R}^3$ and acceleration estimation $\boldsymbol{a}_{_{ES}} \in \mathbb{R}^3$, in $\{E\}$, and accelerometer bias $\boldsymbol{b}_{_S} \in \mathbb{R}^3$, as well as the metric scale $\lambda > 0$ of the mSLAM position estimation, which is defined as $\overline{\boldsymbol{p}}_{_{VS}} = \lambda \boldsymbol{p}_{_{VS}}$. The position fusion algorithm is formed of a coordinate frame management process and a 13-state linear Kalman filter. The Kalman filter conducts in the earth frame $\{E\}$, thus, all the sensor measurement values have to be converted to $\{E\}$ in the coordinate frame management process.

4.4.1 Coordinate frame management process

4.4.1.1 Dynamic acceleration in earth frame

Different with the orientation fusion process, in the position fusion process, we consider that the accelerometer not only measures the gravity, but also measurements the pure dynamic acceleration caused by the movement of the body frame, and since the orientation estimation $\boldsymbol{q}_{_{ES}}$, obtained from orientation fusion process, recovers the gravity vector $\boldsymbol{g}_{_E} = [0, 0, 0, -1]$ in $\{E\}$, therefore the dynamic acceleration ${}^{s}\boldsymbol{a}_{_{ES}} \in \mathbb{R}^3$ in $\{E\}$ can be easily obtained by:

$$\begin{bmatrix} 0 \\ {}^{s}\boldsymbol{a}_{_{ES}} \end{bmatrix} = \|{}^{r}\boldsymbol{a}_{_S}\|(\boldsymbol{q}_{_{ES}} \otimes \boldsymbol{a}_{_S} \otimes \boldsymbol{q}^{\star}_{_{ES}} - \boldsymbol{g}_{_E}), \tag{4.25}$$

recalling the normalised accelerometer measurement $\boldsymbol{a}_{_S} = \frac{{}^{r}\boldsymbol{a}_{_S}}{\|{}^{r}\boldsymbol{a}_{_S}\|}$.

4.4.1.2 Unscaled vision position in earth frame

The unscaled position estimation from mSLAM algorithm ${}^{s}\overline{\boldsymbol{p}}_{_{ES}} \in \mathbb{R}^3$ in $\{E\}$ can be obtained by:

$$\begin{bmatrix} 0 \\ {}^{s}\overline{\boldsymbol{p}}_{_{ES}} \end{bmatrix} = \boldsymbol{q}_{_{ES}} \otimes (\boldsymbol{q}^{\star}_{_{VS}} \otimes \overline{\boldsymbol{p}}_{_{VS}} \otimes \boldsymbol{q}_{_{VS}}) \otimes \boldsymbol{q}^{\star}_{_{ES}}. \tag{4.26}$$

Therefore, the resulting measurements in $\{E\}$ frame (${}^{s}\boldsymbol{a}_{_{ES}}$ and ${}^{s}\overline{\boldsymbol{p}}_{_{ES}}$) can be passed to the position Kalman filter as two individual sensor measurement, which will be described as follows.

4.4.2 Linear Kalman filter

The conventional Kalman filter (KF) framework consists of a prediction step, which performs the state vector time update in constant time interval; and a measurement update step, which performs the correction of the state vector based on the new sensor measurement. Here in order to encounter the asynchronous measurements from both accelerometer and mSLAM algorithm, two different measurement update models are constructed, and will be executed depending on which sensor measurement is available.

4.4.2.1 State representation and prediction model

The state of the Kalman filter is represented as a state vector $\boldsymbol{x} \in \mathbb{R}^{13}$:

$$\boldsymbol{x} = [\boldsymbol{p}_{ES}^{\top}, \boldsymbol{v}_{ES}^{\top}, \boldsymbol{a}_{ES}^{\top}, \boldsymbol{b}_{S}^{\top}, \lambda]^{\top}, \tag{4.27}$$

where position estimation $\boldsymbol{p}_{ES}$, velocity estimation $\boldsymbol{v}_{ES}$ and acceleration estimation $\boldsymbol{a}_{ES}$ are in $\{E\}$, and accelerometer bias $\boldsymbol{b}_S$ is in $\{S\}$, as well as the metric scale $\lambda > 0$ of the mSLAM position estimation, which is defined as $\overline{\boldsymbol{p}}_{VS} = \lambda \boldsymbol{p}_{VS}$.

The state vector is updated once every time interval, following the rule defined by the prediction model, which defines the physics of the inertial system. It is summarised as:

$$\dot{\boldsymbol{p}}_{ES} = \boldsymbol{v}_{ES}, \tag{4.28}$$

$$\dot{\boldsymbol{v}}_{ES} = \boldsymbol{a}_{ES}, \tag{4.29}$$

$$\dot{\boldsymbol{a}}_{ES} = \boldsymbol{n}_a, \quad \dot{\boldsymbol{b}}_S = \boldsymbol{n}_b, \quad \dot{\lambda} = n_\lambda. \tag{4.30}$$

The linear acceleration $\boldsymbol{a}_{ES}$, accelerometer bias $\boldsymbol{b}_S$ and mSLAM metric scale factor λ are modelled as Gaussian random walk. Thus, $\boldsymbol{n}_a \in \mathbb{R}^3$, $\boldsymbol{n}_b \in \mathbb{R}^3$ and n_λ are independent zero-mean normal distribution Gaussian process noise:

$$p(\boldsymbol{n}_a) \sim N(0, Q_a), \tag{4.31}$$

$$p(\boldsymbol{n}_b) \sim N(0, Q_b), \tag{4.32}$$

$$p(n_\lambda) \sim N(0, \sigma_\lambda^2), \tag{4.33}$$

where $Q_a = \sigma_a^2 \boldsymbol{I}_3$ and $Q_b = \sigma_b^2 \boldsymbol{I}_3$ are the process noise covariance of acceleration and accelerometer bias respectively. σ_a, σ_b and σ_λ are the standard deviations of $\boldsymbol{n}_a$, $\boldsymbol{n}_b$ and n_λ respectively, and $\boldsymbol{I}_3$ is three by three identity matrix.

Following Welch and Bishop (2006), the discrete Kalman filter time update includes two steps:

(1) state vector propagation to predict the state vector in next time step:

$$\hat{\boldsymbol{x}}_k = A\boldsymbol{x}_{k-1}, \tag{4.34}$$

where A is the state transition matrix, which is the Jacobian matrix of partial derivatives of the prediction model with respect to $\boldsymbol{x}$.

(2) state covariance matrix $P \in \mathbb{R}^{13\times 13}$ propagation to predict the state noise in next time step:

$$\hat{P}_k = AP_{k-1}A^\top + WQW^\top, \tag{4.35}$$

where W is the Jacobian matrix of partial derivatives of the prediction model with respect to the noise vector, P is the state covariance matrix and $Q = diag(\mathbf{0}_{6\times 6}, Q_a, Q_b, \sigma_\lambda^2)$ is the process noise covariance matrix.

4.4.2.2 Measurement model

The measurement model is derived in the form of:

$$\boldsymbol{z}_\star = H_\star \boldsymbol{x} + \boldsymbol{e}_\star, \tag{4.36}$$

where $\boldsymbol{z}_\star$ is the measurement from the mSLAM vision sensor or the IMU, $H_\star$ is the measurement model matrix, and $e_\star$ denotes the measurement error from the sensor, where $_\star$ can be $_{as}$ or $_{vs}$ depending on which sensor measurement is available between acceleration sensor measurement and vision sensor measurement. Here, $e_\star$ is also modelled as independent zero-mean normal distribution Gaussian process noise:

$$p(\boldsymbol{e}_\star) \sim N(0, R_\star), \tag{4.37}$$

where $R_\star$ is the measurement noise covariance of $e_\star$.

When the accelerometer measurement ${}^s\boldsymbol{a}_{ES}$ is available, the Kalman filter performs accelerometer measurement update, to adjust state vector and state covariance matrix according to the accelerometer measurement model. Here the accelerometer measurement model is defined by matrix $H_{as} \in \mathbb{R}^{3\times 13}$

$$H_{as} = [\mathbf{0}_{3\times 6} \quad \boldsymbol{I}_3 \quad \mathsf{R}_{ES} \quad \mathbf{0}_{3\times 1}], \tag{4.38}$$

where $\mathsf{R}_{ES} \in SO(3)$ is the corresponding rotation matrix to $\mathbf{q}_{ES}$, and the accelerometer measurement noise covariance $R_{as} = \sigma_{as}^2 \boldsymbol{I}_3$, where σ_{as} is the standard deviations of $\boldsymbol{e}_{as}$

When the unscaled position measurement ${}^s\overline{\boldsymbol{p}}_{ES}$ is available from mSLAM algorithm, the Kalman filter performs vision measurement update, to adjust state vector and state covariance matrix according to the vision measurement model. Here the vision measurement model is defined by matrix $H_{vs} \in \mathbb{R}^{3\times13}$

$$H_{vs} = [\lambda \boldsymbol{I}_3 \quad \boldsymbol{0}_{3\times9} \quad \boldsymbol{p}_{ES}]. \tag{4.39}$$

And the vision measurement noise covariance $R_{vs} = \sigma_{vs}^2 \boldsymbol{I}_3$, where σ_{vs} is the standard deviations of $\boldsymbol{e}_{vs}$.

Following Welch and Bishop (2006), the measurement update steps are summarised as:

(1) Compute Kalman gain $K_k \in \mathbb{R}^{13\times6}$:

$$K_k = \hat{P}_k H_\star^\top (H_\star \hat{P}_k H_\star^\top + R_\star)^{-1}. \tag{4.40}$$

(2) State vector update:

$$\boldsymbol{x}_k = \hat{\boldsymbol{x}}_k + K_t(\boldsymbol{z}_k - H_\star \hat{\boldsymbol{x}}_k). \tag{4.41}$$

(3) State covariance update:

$$P_k = (\boldsymbol{I}_{16} - K_k H_\star)\hat{P}_k. \tag{4.42}$$

Measurement update process handles different sampling rate between mSLAM and IMU estimation, by only updating state with the corresponding measurement, which becomes available. Thus by assuming the orientation fusion reaches steady state, the state vector $\boldsymbol{x}$ can be effectively estimated over time.

4.5 Implementation

This section describes the detail on the algorithm implementation on an embedded platform. The entire system was installed on a 250 mm size quadrotor platform, as shown in Fig.4.4.

FreeIMU v0.4.3[1] hardware was used as the inertial measurement unit, which includes an MPU6050 gyroscope-accelerometer combo chip, an HMC5883 magnetometer and MS5611 high resolution pressure sensor. However, only the MPU6050 was used in the state estimation algorithm. We performed orientation estimation in Teensy 3.1[2], which features an ARM Cortex-M4 processor with 96 MHz clock cycle. Both FreeIMU and Teensy 3.1 are soldered onto a custom designed autopilot board as shown in Figure 4.5a. The Autopilot board was designed to electronically and mechanically interface the FreeIMU, Teensy processor, voltage

[1]http://www.varesano.net/projects/hardware/FreeIMU
[2]https://www.pjrc.com/teensy

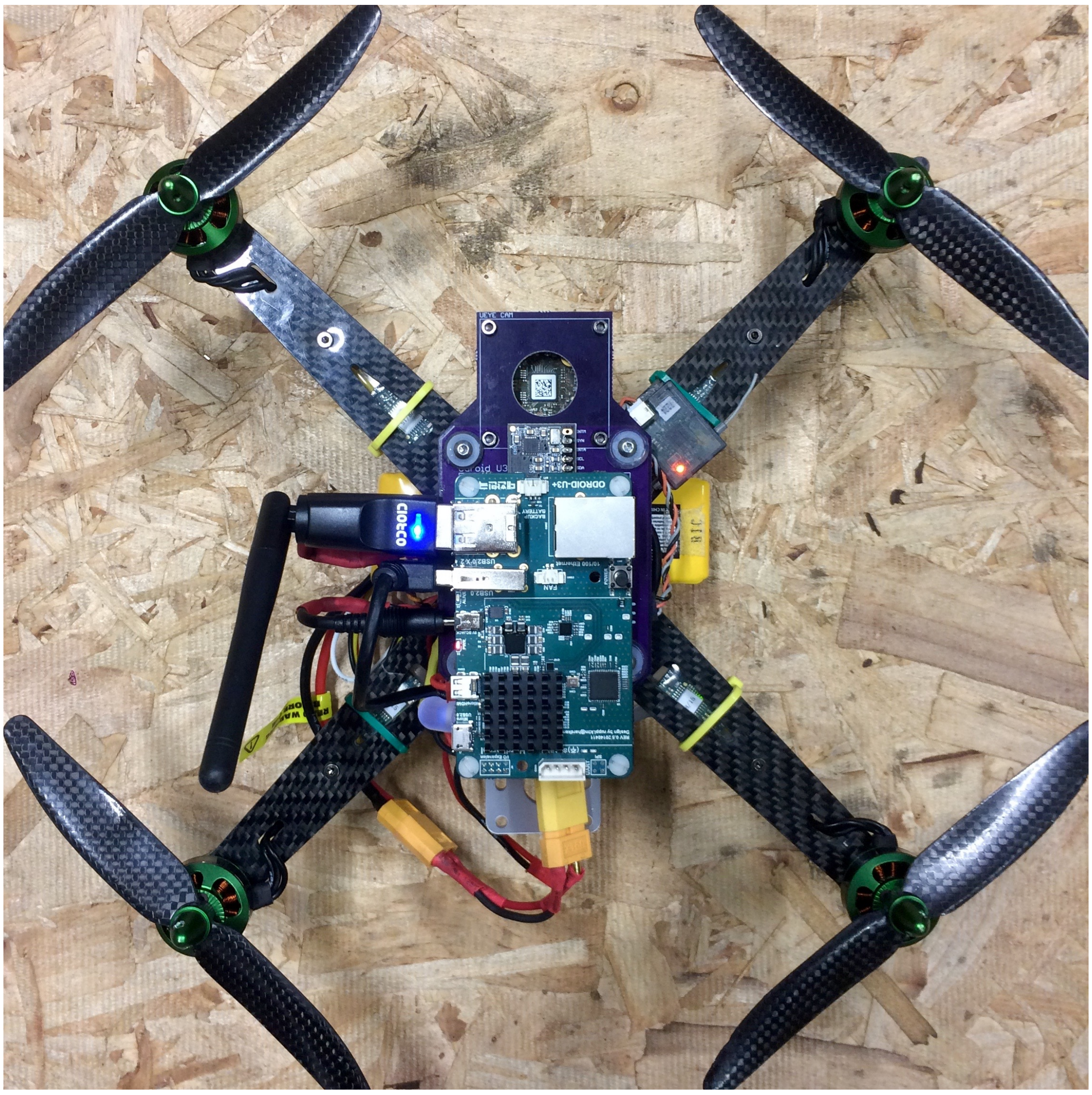
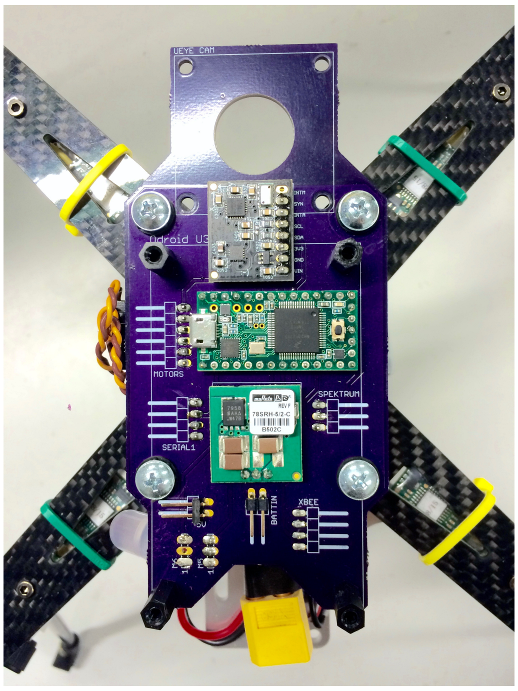


Figure 4.4: Entire micro UAV system equipped with the visual inertial controller.

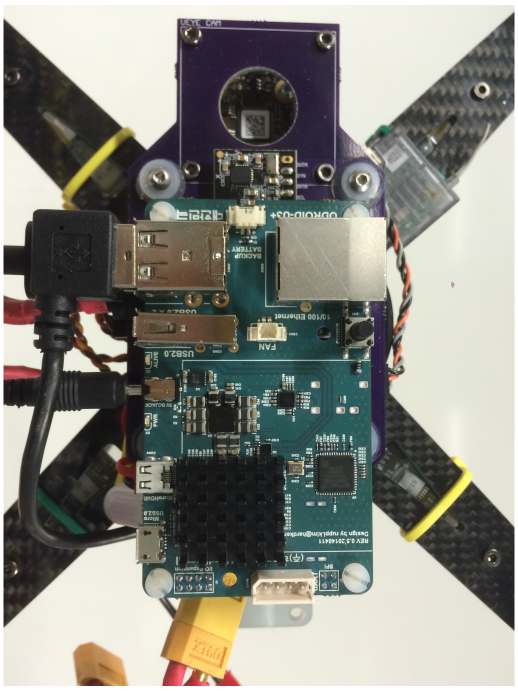
regulator and XBee Radio. This PCB board also provides mechanical interface to the onboard camera and onboard Linux computer.

Besides, The real-time video frame was captured by a downward-facing onboard global shutter uEye monocular camera (Figure 4.6) in maximum 80 frames per second. Then, both the video frame and the orientation estimation are processed by the SVO mSLAM framework with the EKF position fusion algorithm operating in parallel on a separate processor core in Odroid-U3[3], as shown in Figure 4.5b. The Odroid-U3 is single board embedded Linux smartphone computer, which features an 1.7 GHz Quad-Core processor with 2 GByte RAM. The communication between software packages is realised by Robot Operating System[4] (ROS).

[3]http://www.hardkernel.com/
[4]http://www.ros.org



(a) FreeIMU sensor (top) and Teensy 3.1 processor (middle), where it runs orientation fusion and autopilot algorithm, and a voltage regulator (bottom).



(b) Onboard Linux smartphone computer (Odroid-U3), where it runs position fusion and monocular visual SLAM based on ROS interface.

Figure 4.5: The hardware setup.

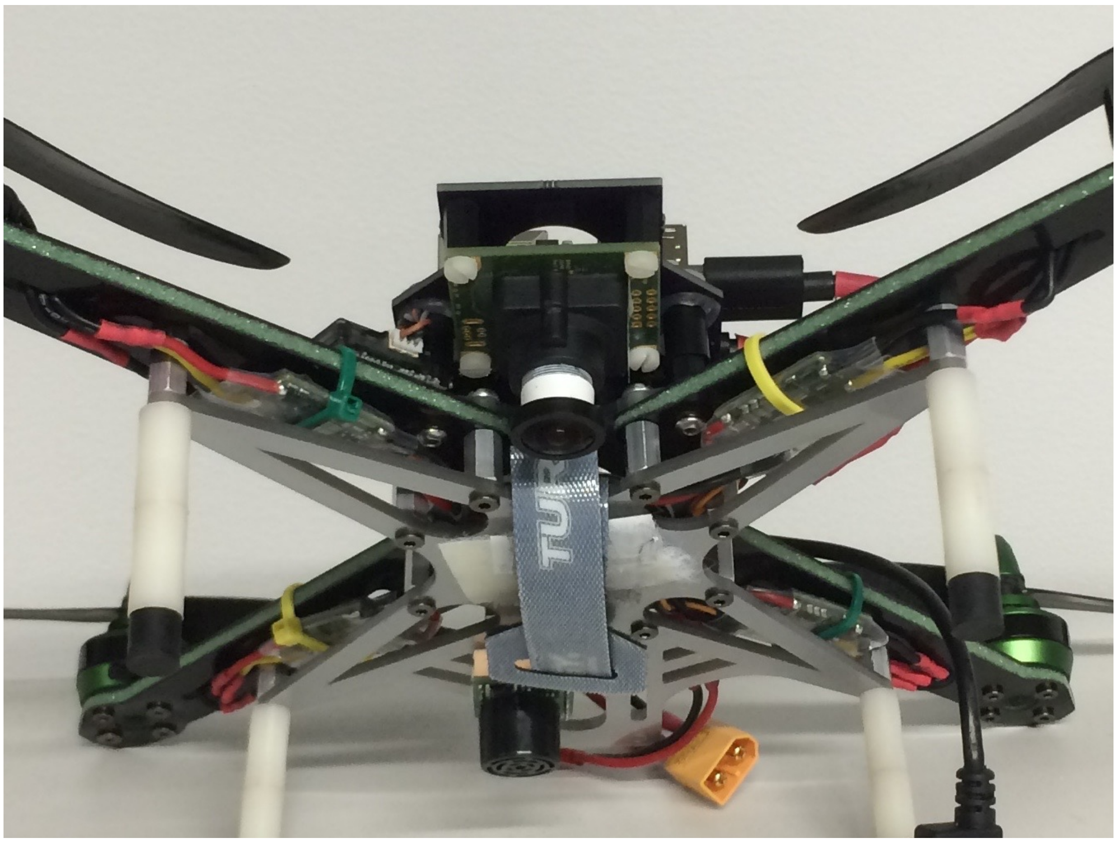


Figure 4.6: Downward facing global shutter monocular Ueye camera.

Table 4.1: Parameters setup.

Parameter	Value
β	0.5
σ_a	0.5
σ_b	$1 \times e^{-6}$
σ_λ	$1 \times e^{-6}$
σ_{as}	0.013
σ_{vs}	0.005

The Teensy processor is capable of executing the orientation fusion alongside with autopilot control algorithm at 300 Hz, while communicating with Odroid-U3 computer with ROS protocol, including publishing orientation estimation and acceleration measurement at 200 Hz and subscribing the pose estimation from SVO mSLAM framework in Odroid-U3. Moreover, in the Odroid-U3 computer, the SVO mSLAM is executed at 40 FPS with the KF position fusion algorithm running at 200 Hz in parallel.

We measured the SLAM processing delay and set the message buffer manually without hard synchronisation. The parameters setup for both orientation and KF position fusion algorithm is summarised in Table.4.1. β was left as default value, 0.5, by assuming $\tilde{\omega}_{max}$ was approximately $0.58 \quad rad/s$. The value of σ_{as} was selected according to the accelerometer standard noise from the data sheet of the MPU6050, and σ_{vs} was set to be 0.005 in the map scale. σ_a, σ_b and σ_λ were manually tuned through experiments. Here we assumed accelerometer bias and visual scale change very slowly, thus σ_b and σ_λ were set very small. σ_a determines the confidence level for the prediction model, which means the higher was σ_a, the less confidence is the estimator about its prediction model, thus the easier the sensor measurements effect the estimation.

4.6 Test Results

Three sets of test trials were performed to demonstrate the effectiveness of the algorithm. The trials were performed in real-time under general indoor condition, and handholding the quadrotor to produce motion, as shown in Figure 4.7.

4.6.1 Ground truth

Under the VICON[5] motion capture system, since the VICON cameras emits high power flashing 750 nm wave-length near infrared (IR) light, the Ueye camera had to use the lens with 650 nm wave-length IR filter. As shown in Figure 4.7, four reflective markers were rigidly installed onto the UAV body, in a pre-defined non-symmetrical form. The position of n^{th} marker in vehicle body frame was directly observed by the VICON system at 200 Hz, which can be expressed as

[5]https://www.vicon.com

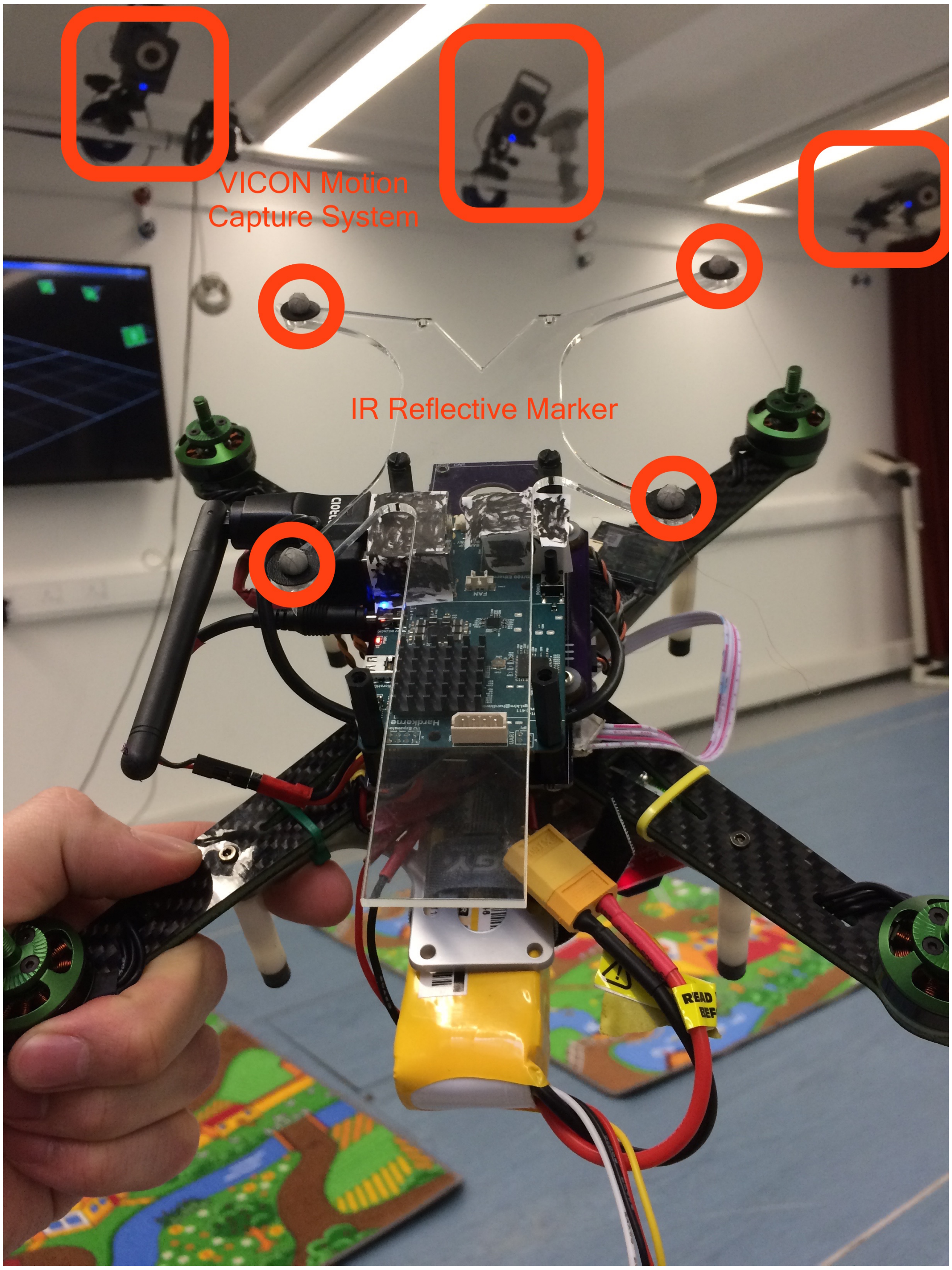


Figure 4.7: VICON indoor testing environment. (University of Southampton, Health Sciences Faculty, Active Living and Rehabilitation Group, VICON lab)

$\boldsymbol{p_n} = [p_{n0}, p_{n1}, p_{n2}, 1]^\top \in \mathbb{R}^4$. Then one can obtain the marker matrix $M \in \mathbb{R}^{4\times4}$ as:

$$M = [\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}, \boldsymbol{p_4}]. \tag{4.43}$$

Since the 3D position of individual marker can be recovered by the VICON system. The position of n^{th} marker in world frame can be expressed as $\boldsymbol{{}^w p_n} = [{}^w p_{n0}, {}^w p_{n1}, {}^w p_{n2}, 1]^\top \mathbb{R}^4 \in \mathbb{R}^4$. Then one can obtain the marker matrix as ${}^w M \in \mathbb{R}^{4\times4}$:

$${}^w M = [\boldsymbol{{}^w p_1}, \boldsymbol{{}^w p_2}, \boldsymbol{{}^w p_3}, \boldsymbol{{}^w p_4}]. \tag{4.44}$$

Thus in this case, the relation between M and ${}^w M$ can be expressed as a rigid transformation $T_t \in \mathbb{R}^{4\times4}$:

$$T_t M = {}^w M, \tag{4.45}$$

which can be seen as the ground truth pose measurement from VICON system. The affine transformation $T_t' \in \mathbb{R}^{4\times4}$ can be obtained by the linear least square solution:

$$T_t' = {}^w M M^+, \tag{4.46}$$

where M^+ is the pseudo inverse of the marker matrix M. However, in order to estimate the orientation and position, the affine transformation must be approximated into the form of the rigid transformation constraints. Given that the marker position was measured sufficiently accurate, it is safe to directly approximate the affine transformation.

The affine transformation T_t' can be seen in the form of:

$$T_t' = \begin{bmatrix} \boldsymbol{x_t'} & \boldsymbol{y_t'} & \boldsymbol{z_t'} & \boldsymbol{p_t} \\ r_1 & r_2 & r_3 & r_4 \end{bmatrix}, \tag{4.47}$$

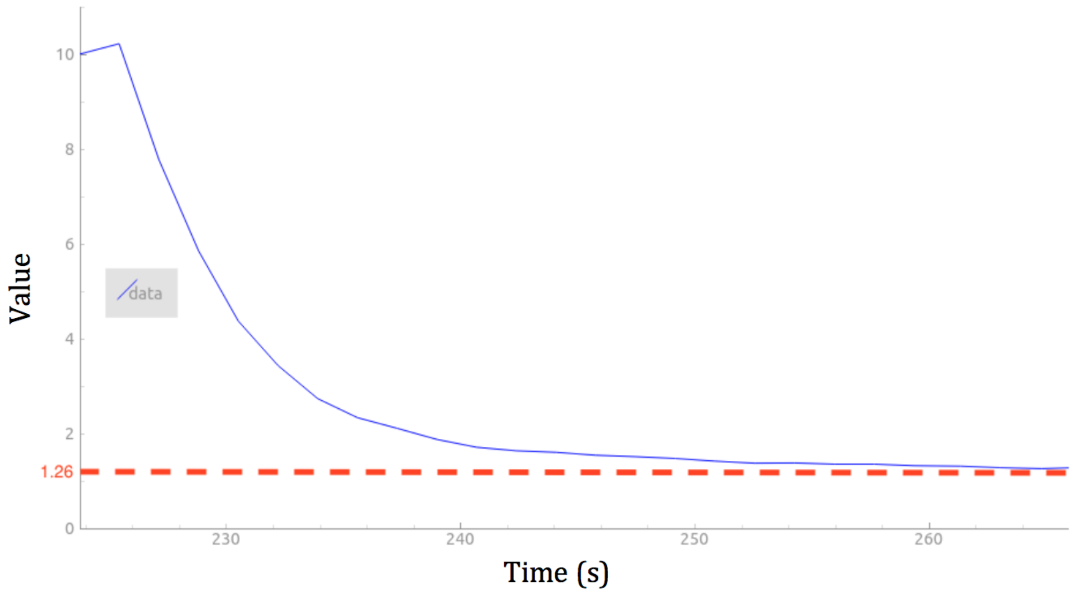
where $\boldsymbol{x_t'}, \boldsymbol{y_t'}, \boldsymbol{z_t'}, \boldsymbol{p_t} \in \mathbb{R}^3$, r_1, r_2, r_3 are random value close to 0, and r_4 is random value close to 1. Then the approximation can be achieved by:

$$\boldsymbol{x_t} = \frac{\boldsymbol{x_t'}}{||\boldsymbol{x_t'}||}, \tag{4.48}$$

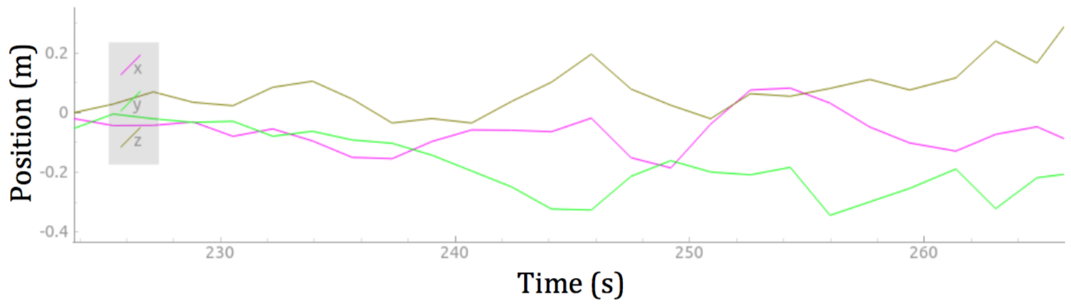
$$\boldsymbol{z_t} = \frac{\boldsymbol{x_t} \times \boldsymbol{y_t'}}{||\boldsymbol{x_t} \times \boldsymbol{y_t'}||}, \tag{4.49}$$

$$\boldsymbol{y_t} = \frac{\boldsymbol{z_t} \times \boldsymbol{x_t}}{||\boldsymbol{z_t} \times \boldsymbol{x_t}||}, \tag{4.50}$$

$$T_t = \begin{bmatrix} \boldsymbol{x_t} & \boldsymbol{y_t} & \boldsymbol{z_t} & \boldsymbol{p_t} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{4.51}$$



(a) Scale factor λ.



(b) The scaled position measurement from KF position estimator.

Figure 4.8: KF position fusion result.

Then the ground truth vehicle body rotation matrix is $R_t = [x_t, y_t, z_t]$ and ground truth vehicle body translation is p_t. This direct pose (orientation and position) measurement is treated as the ground truth, used as the comparison against the pose estimated by the implemented system.

4.6.2 First trial: scale convergence

The first trial focused on the evaluation of the scale factor (λ) estimation from an arbitrary initial value. The trial was conducted by handholding the quadrotor with gentle movement within $0.3 \times 0.3 \times 0.3\ m^3$ space. Note that the position fusion assumes the orientation fusion reaches the steady state before initialisation. The KF position fusion algorithm is initialised with the state vector $\boldsymbol{x_0} = [\mathbf{0}_{1\times 12}, 10]^\top$, note that we initialise the scale factor λ to 10 as an arbitrary positive initial value to show how it converges to the true value. As shown in Fig.4.8, The record starts at 227 second, when the initialisation occurred, and the record shows a 39-second trial. It is clear that the scale factor λ converged and sufficiently recovered as 1.26, despite that its initial value was set to 10, as shown in Fig.4.8a, and during the converging period, the position

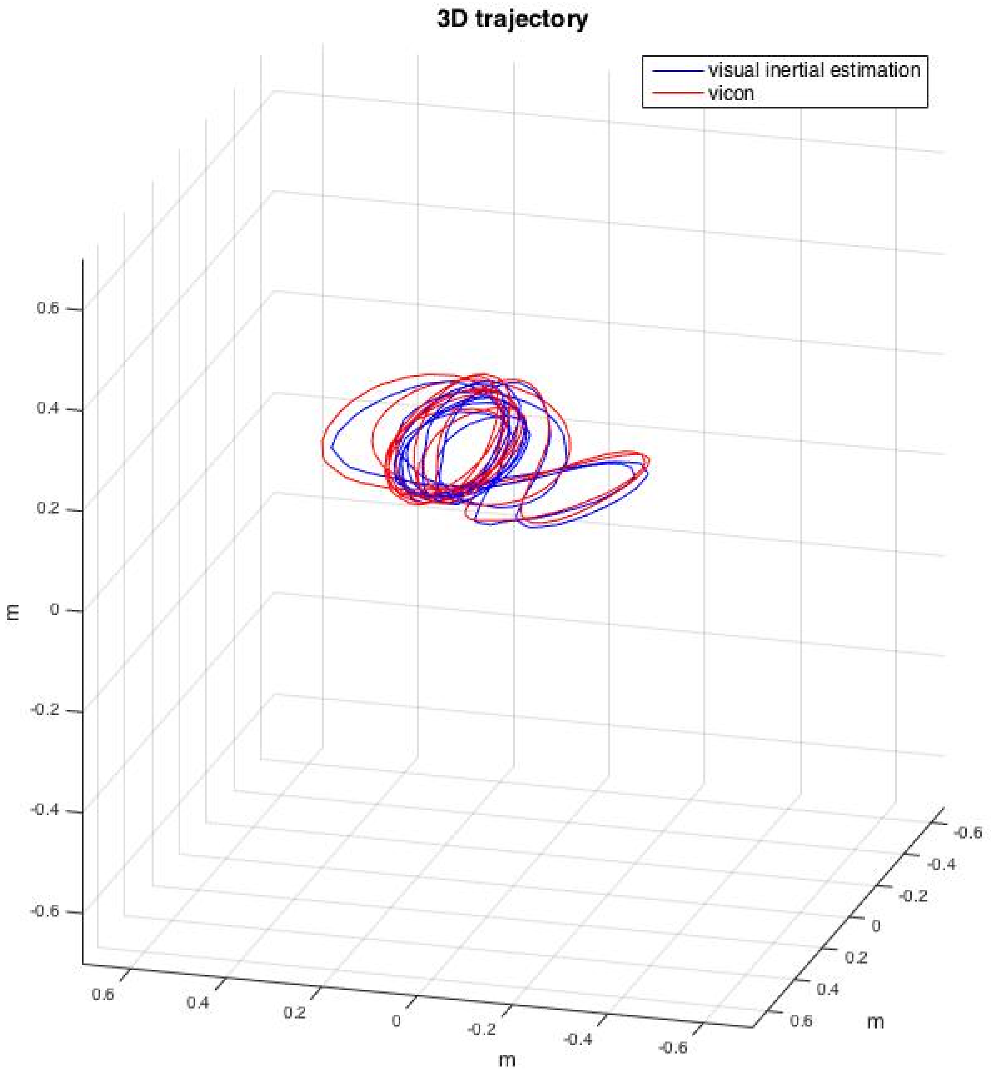


Figure 4.9: 3D trajectory illustration against ground truth comparison.

estimation output from KF position estimator is scaled accordingly with with the change of λ over time, as shown in Figure 4.8b.

4.6.3 Second trial: accuracy evaluation

The second trial evaluated the six degree-of-freedom (position and orientation) estimation performance, by comparing the pose estimation from the filter against the ground truth from VICON motion capturing system. The random movement was generated by handheld motion in about $2 \times 2 \times 2\ m^3$ space as shown in Figure 4.9 for 17 seconds duration.

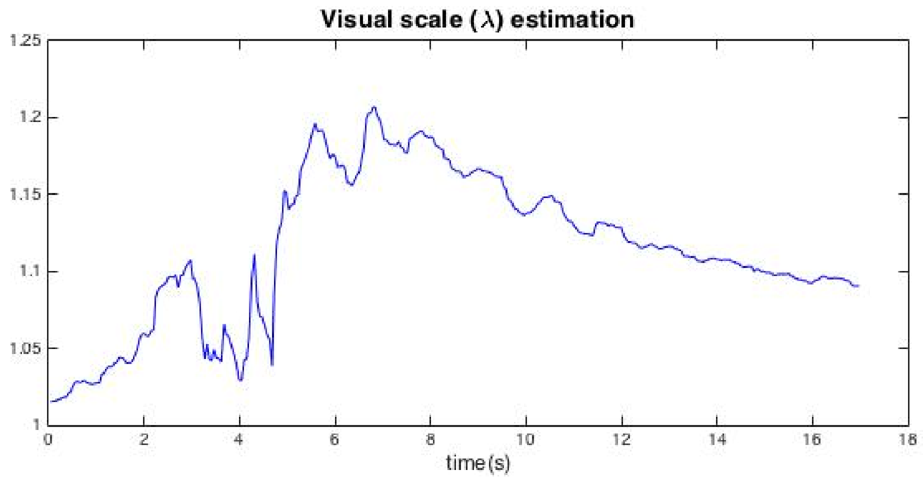


Figure 4.10: Scale factor estimation.

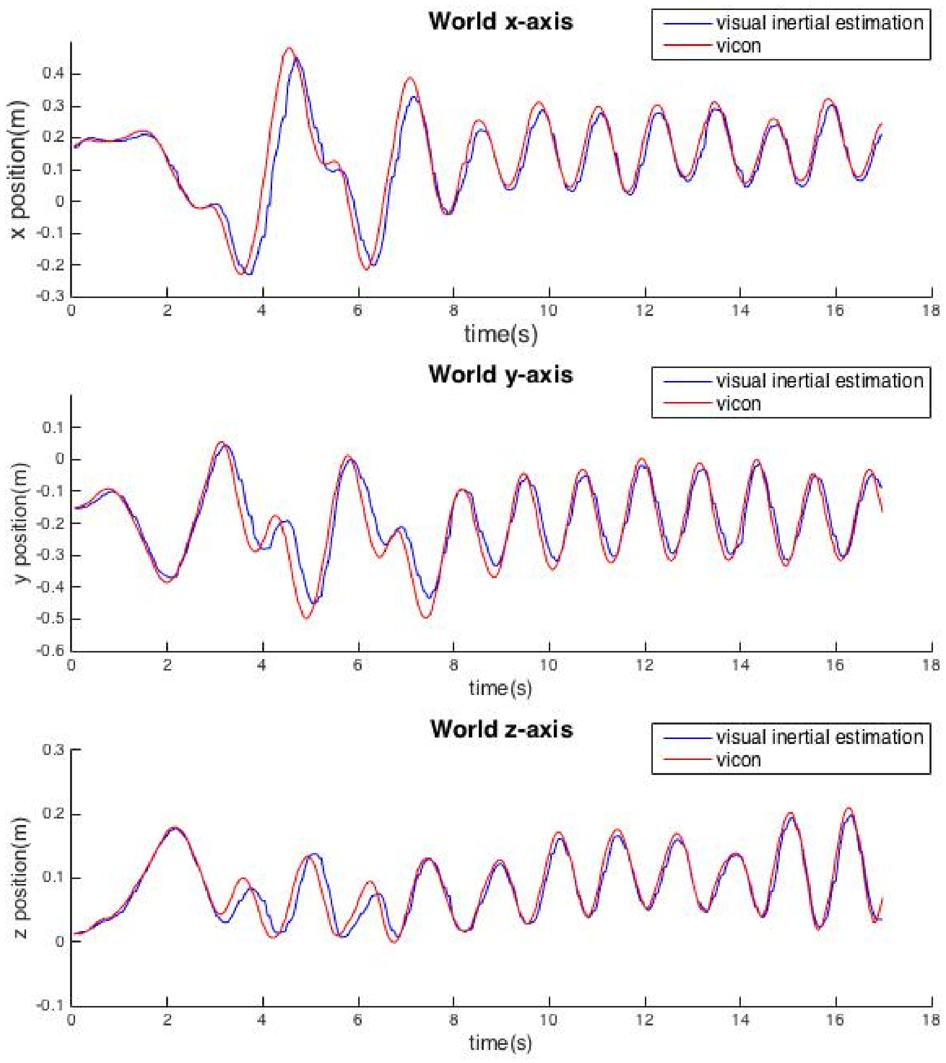


Figure 4.11: Position estimation ground truth comparison.

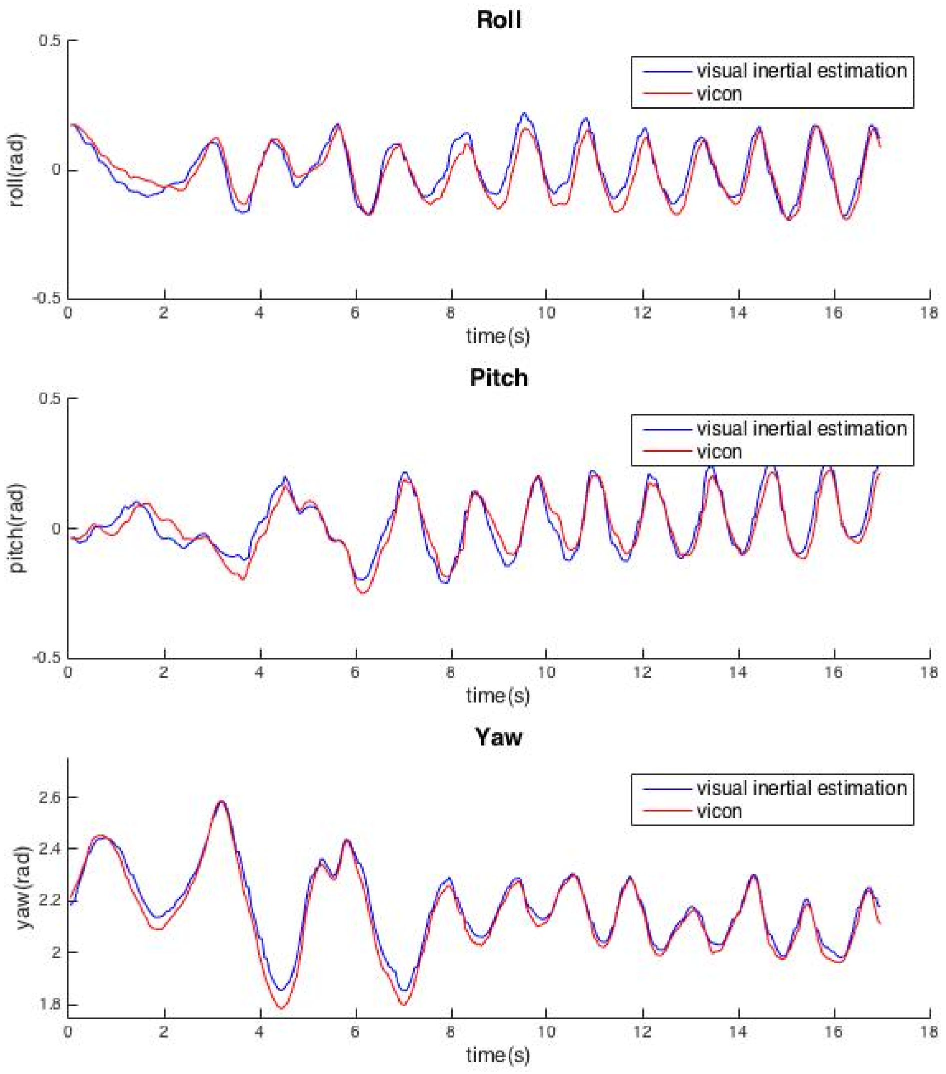


Figure 4.12: Attitude Euler angle estimation ground truth comparison.

The estimated scale factor of the SVO is shown in Fig.4.10. As shown, the scale factor was initialised close to the true scale. The initial scale varying was due to the large uncertainty at start up. Besides, since the SVO initialised a high volume of new map points at the initialisation, which results in the delay of the SVO pose measurement, thus it causes small timing error due to the soft manual synchronisation as mentioned in Section 4.5. Nonetheless, the scale estimation became more stable after 6 second and approaches towards the true value (as shown around 1.1).

The position estimation for each axis is shown in Fig.4.11. It is shown that the position estimation experienced the delay before 8 second. This is also due to the initial large uncertainty and timing error, and the position estimation approaches very close to the true value after 8 second.

The orientation estimation is represented as the Euler angle, as shown in Fig.4.12. As explained

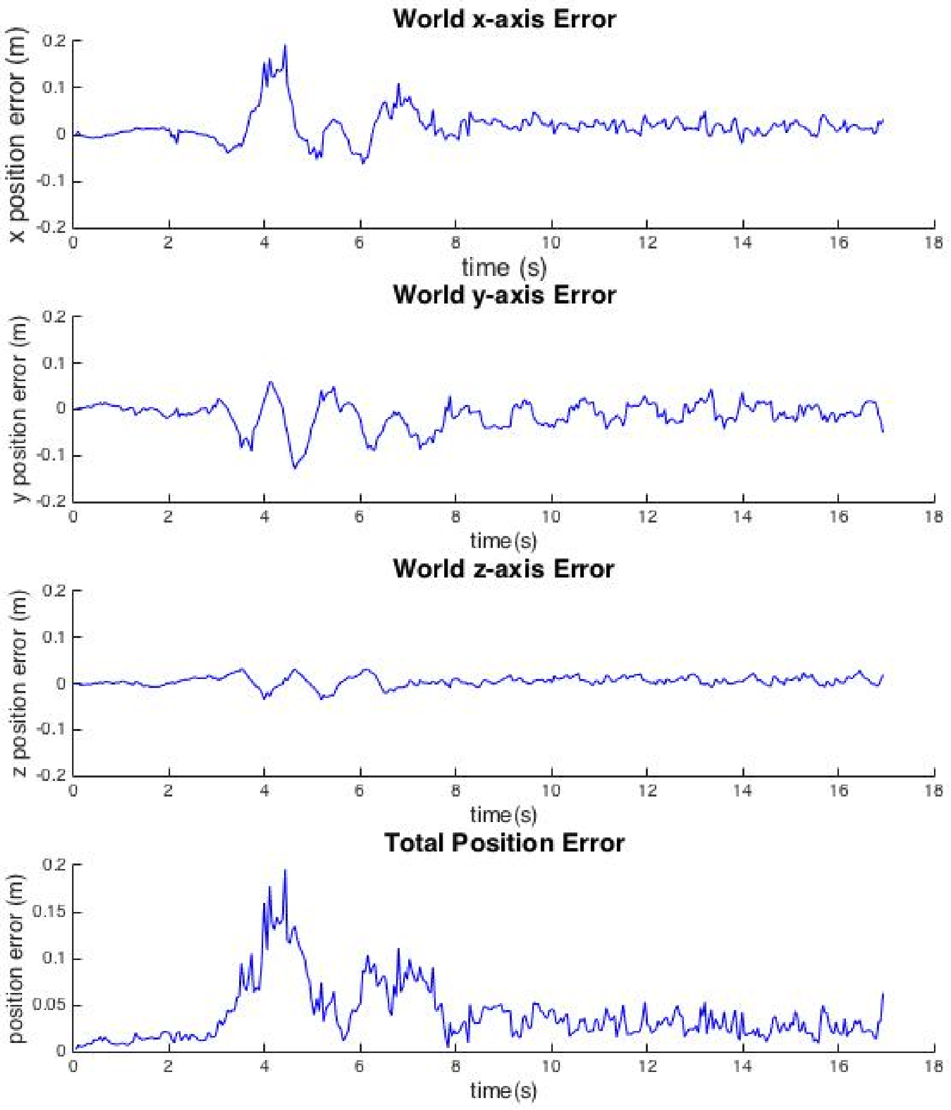


Figure 4.13: Overall position error.

in Section 4.3, since the roll and pitch angle are highly related to the vehicle control performance, in order to avoid the gravity vector drift from the SVO, only the yaw angle (heading) was computed from the fusion between SVO and IMU, while roll and pitch angle are computed directly from IMU fusion. It is shown clearly, that the attitude of the vehicle has been estimated accurately, especially when fusing the vision measurements to correct the yaw angle estimation in IMU, the final yaw angle experienced zero drift over the period. Roll and pitch angle can be sufficiently estimated without gyro and accelerometer only, which accuracy is sufficient enough and shows even faster responds than the ground truth VICON system, due to the fast IMU internal fusion process at 400 Hz.

Finally, the position estimation error was computed and shown in Fig.4.13. The positional error along the three dimensional axis were computed separately. Besides, the total positional error was computed as the root mean error (RME, also known as the quadratic mean) of all three axis errors. It was shown clearly that once the filter converges (after 8 second), the position error was kept below 0.05 m in all dimensions. Moreover, the remaining position error is believed to be mainly introduced by the small error in visual scale estimation, which could be the result of small timing error from manual synchronisation or slow converging rate close to the true value.

4.6.4 Third trial: figure movements

In order to clearly demonstrate the performance of the visual inertial system, two additional figure movements were conducted. The 3D trajectory of figure eight movement is shown in Figure 4.14, and the 3D trajectory of cube figure movement is shown in Figure 4.15. Moreover, as shown in Figure 4.15, despite that the system experienced several temporary wrong measurements from SVO, the system was still able to back track from the wrong estimation. This shows the good robustness of the system.

4.7 Conclusions and Future Work

This chapter has shown the design and implementation work of a sensor fusion framework, which is capable of performing the six degree of freedom sensor state estimation, by fusion a 3-axis gyroscope, a 3-axis accelerometer and a monocular vision based simultaneous localization and mapping algorithm.

Three test trials were carried out to demonstrate the effectiveness of the system. The first trial showed that scale factor of the mSLAM can be sufficiently estimated from an arbitrary value, thus the position output is scaled accordingly. The second trial showed in detail about the comparison of the real-time pose estimation against ground truth. It shows that the 3D position can be accurately estimated, and the drift-free attitude estimation, especially the yaw rotation can be estimated drift-free without the need for magnetometer. Further figure movements demonstrates the robustness of the system against different types of movement and also against the temporary tracking failure of the used SLAM system.

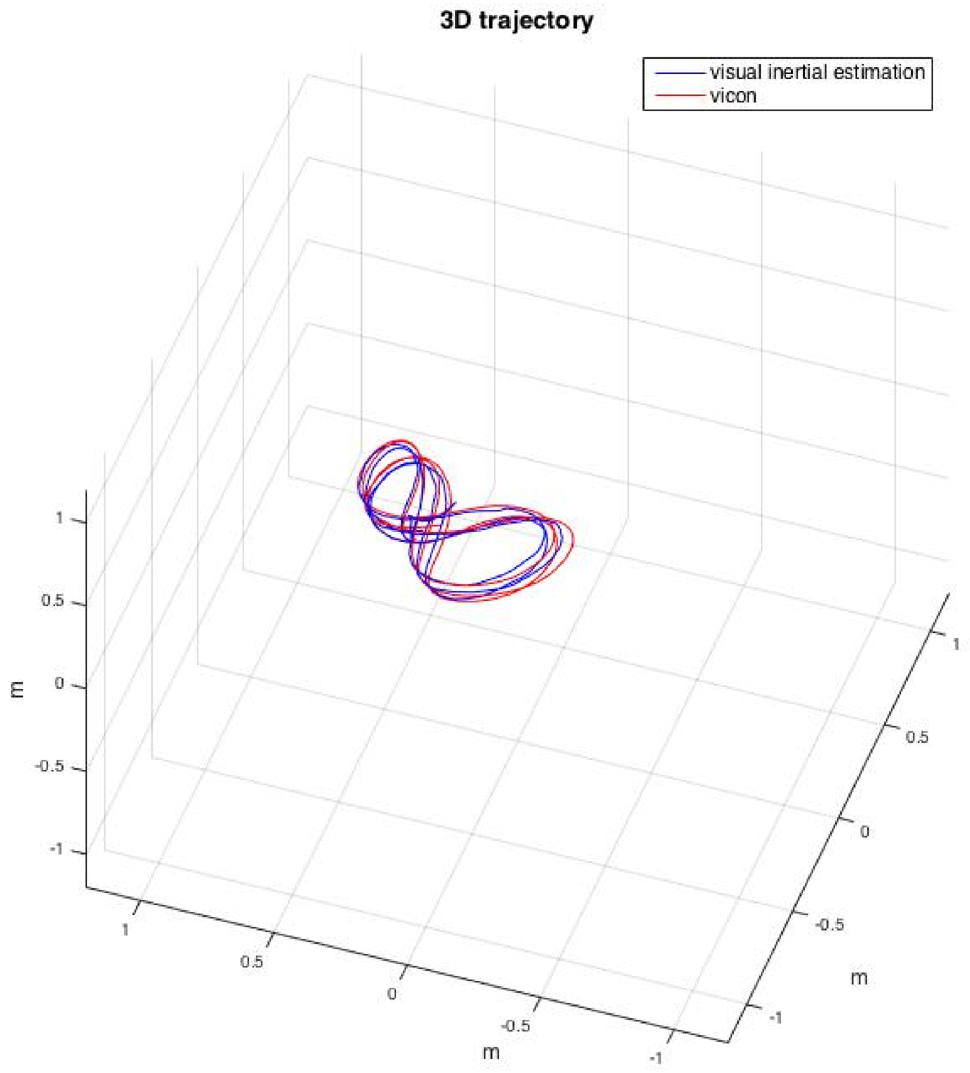


Figure 4.14: Figure '8' motion position estimation ground truth comparison.

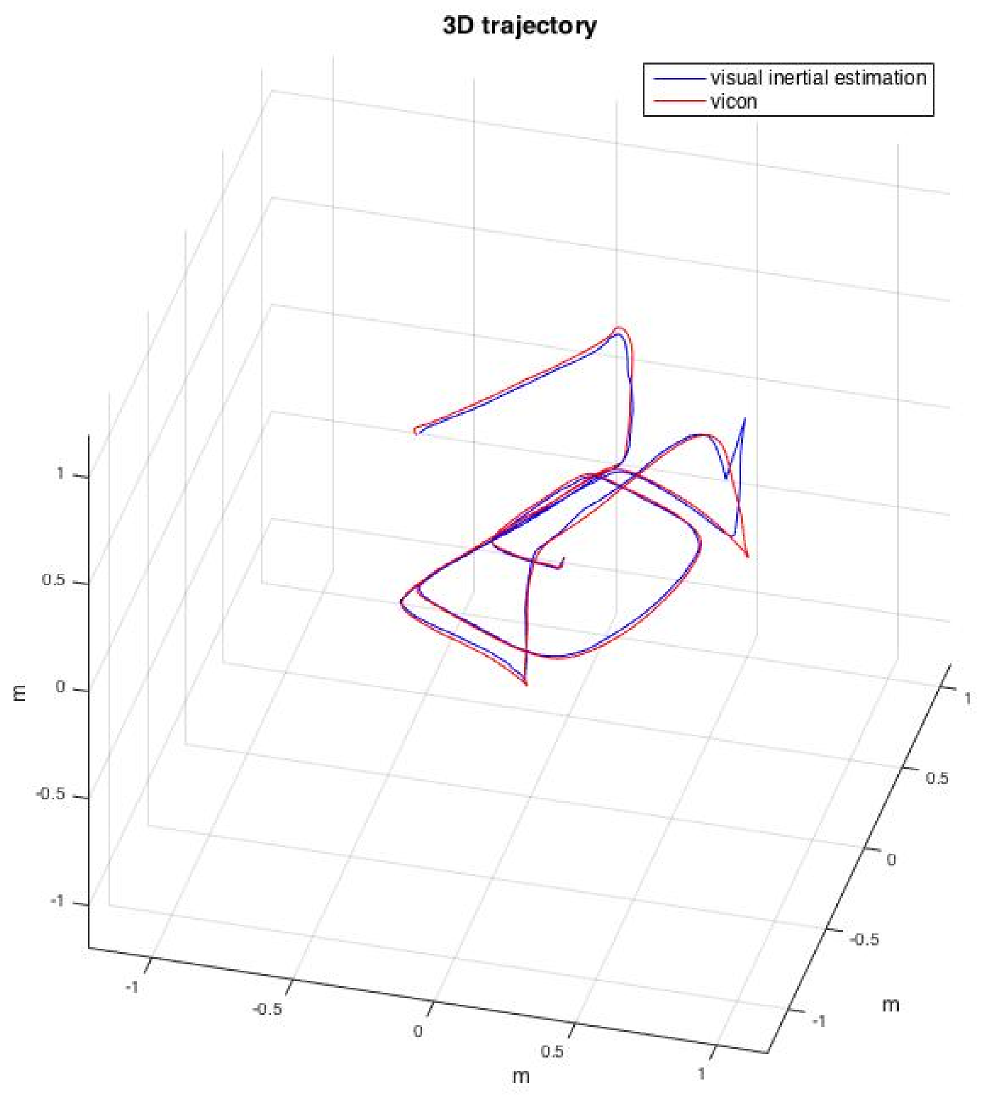


Figure 4.15: Cubic motion position estimation ground truth comparison.

Chapter 5

System Integration and Test Results

This chapter focuses on integrating the controller designed in Chapter 3 with the visual inertial state estimation system developed in Chapter 4 on a challengingly small 250 mm quadrotor platform. It describes the mechanical and electronics design behind the integration. It also describes the failure handling implementation towards the complete power-on-and-go system. Additionally, thanks to the small size of the quadrotor, it is safe to launch it from hands. A novel launching button designed onto the UAV body for fast and intuitive launch. Finally, test results are presented to demonstrate the effectiveness and robustness of the system performance.

Part of this chapter was Submitted to the *Journal of Intelligent and Robotic Systems*, in the title The Practical Implementation of an Autonomous Micro Quadrotor in GPS-denied Environments, on 3rd September, 2016. Author list: Liu, C, Prior, S.D., and Scanlan, J.P.

5.1 Platform and Electronics Implementation

This section summarises the development history over a range of platforms, which eventually leads to the final design. The mechanical and electronic design of the final platform is described in detail, showing the realisation of a 250 mm size autonomous quadrotor with all the sensor and computation onboard. The design includes the body structure, onboard components layout, electrical and electronic systems.

5.1.1 Platform iterations history

The evolution of the past implemented platforms is shown in Figure 5.1. From the top left, as the initial version 1, to bottom right, as the newest version 4. The shown platform-lineup clearly shows the variational trend of the platform size. Initially, the relatively large (500 mm size) platform, as shown in version 1, offers large onboard space and payload capacity, which provides flexibility for onboard components arrangement and reconfiguration. This allows different onboard electronic setup to be implemented and tested quickly without the need for tight integration. They were mainly used to develop the customised autopilot system described in Chapter 3 operating with an off-the-shelf PX4FLOW camera, for the purpose of verifying the control performance.

Then, when the onboard electronics are approaching stable configuration, the system was rebuilt while increasing the integration level and developing the visual inertial fusion algorithm. Thus, the following versions shows gradual size reduction. The size of the version 2 is $360\ mm$, and version 3 and 4 size are $250\ mm$. They were mainly used to implement and test the visual inertial fusion algorithm described in Chapter 4, as well as coupling it with the customised autopilot system, which takes the visual inertial position estimation as the feedback to the position controller. This will be discussed in detail later in this chapter.

Version 1 and 2 employed the COTS quadrotor chassis, since $500\ mm$ and $360\ mm$ class quadrotors are very popular in hobby aerial photography market. However, due to the higher integration requirement for the smaller platforms, components need to be in the exact place with non-standard configuration. This means that it is almost impossible to use the COTS chassis after the version 2. Hence, the version 3 and 4 were designed in SolidWorks. Version 3 was manufactured based on CNC routing the carbon-foam sandwich sheet from university EDMC workshop. However, due to the raw finish and the suboptimal components layout, the system was redesigned into version 4 to have much more stable flight performance. The version 4, which will be discussed in detail in the following sections, was manufactured by an external Chinese company.

5.1.2 Final platform design

The final platform, as the version 4 shown in Figure 5.1, was designed by learning from the failure of all the previous versions. The resulting platform is shown in Figure 5.2, which is a $250\ mm$ platform with $600\ g$ takeoff mass, including all the sensors and computation.

The vehicle chassis was designed as two parallel parts reinforcing each other. The CAD model was developed in SolidWorks as shown in Figure 5.3. The main body frame was a CNC routed $3\ mm$ thick 3K woven carbon fibre sheet for its omnidirectional force bearing ability and ease of machining, whilst the secondary reinforcement part was CNC routed $1\ mm$ thick 3K woven carbon fibre sheet to provide additional rigidity in the vertical axis. The finished sheets are shown in Figure 5.4, and the finished quadrotor system is shown from different viewing angle in Figure 5.5. The forward leaning design maintains the centre of mass of the final system at the geometrical centre between the four motors, whilst allowing the battery kept clear of the downward facing camera.

It is also shown in Figure 5.5 that, as the secondary part only reinforces the main frame, all the motors and electronics attach directly onto the main frame as the uniform-body mechanical

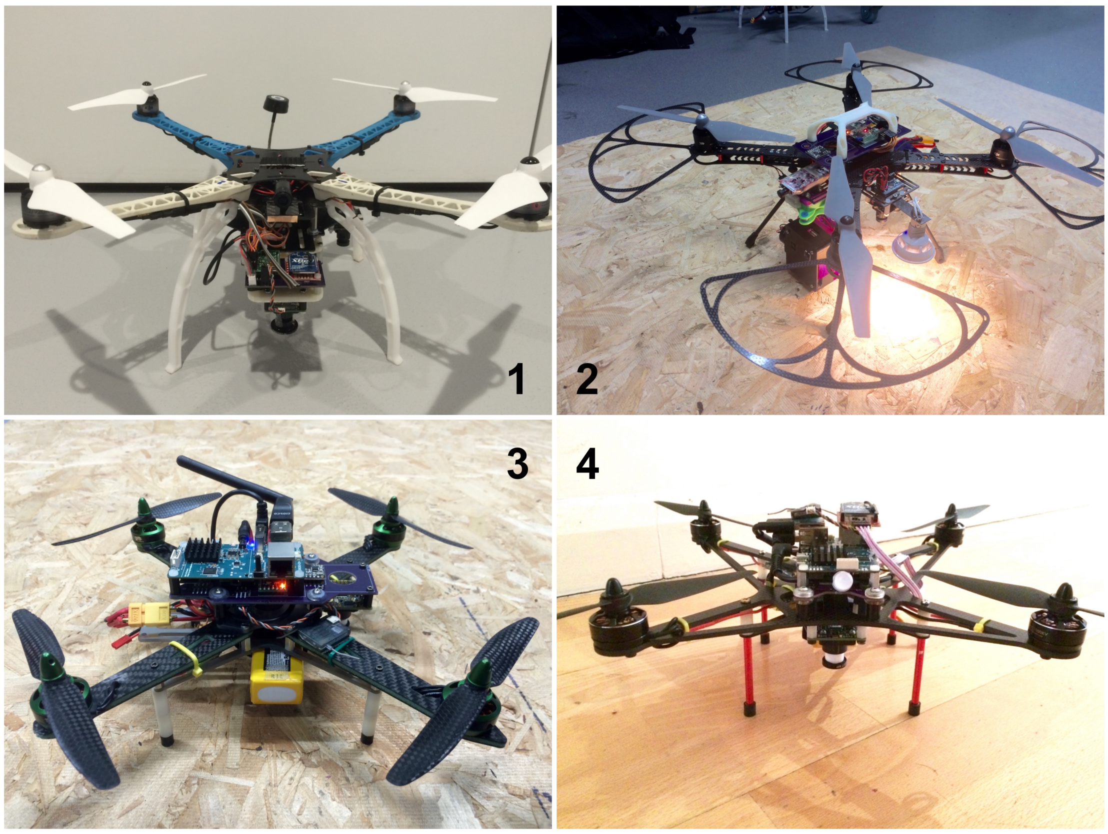


Figure 5.1: The evolution of the previous chassis.

interface. More specifically, the brushless motor propulsion system attaches directly onto the main carbon-fibre part, whose excellent stiffness rejects most of the low frequency vibration directly from the propulsion system, whilst the autopilot attached on the frame via four silicon vibration dampers, which filter out the most of the remaining high frequency vibration. Both methods combine to have a very good vibration isolation throughout the spectrum.

The T-motor AirGear200 propulsion system provides more than $500\ g$ lift per motor, providing over 3.3 thrust to weight ratio, which gives more than enough headroom for stable manoeuvre. Similar with the method used in Section 3.5.1, the propulsion system was tested using a static thrust test rig and curve-fitted as quadratic regression model described in 3.16. The obtained c_o is 2.575 and c_T is 0.1954, which gives 95% accuracy, as shown in Fig. 5.6, where the blue stars are the measured data from the static thrust test, and the red curve is generated from (3.16) with c_T and c_o equal to above values. Note here, the PWM value ρ was normalised to the range from 1 to 100.

The autopilot board was designed in the Eagle PCB design software, as shown in Figure 5.7 and Figure 5.8, for the purpose of minimising the onboard components by maximising the integration. It was designed as a printed circuit board (PCB) for the electronic interface for all the onboard COTS electronics, as well as the mechanical interface for all sensors and the additional Linux computer. The assembled autopilot is shown in Figure 5.9. As shown in dashed green, the

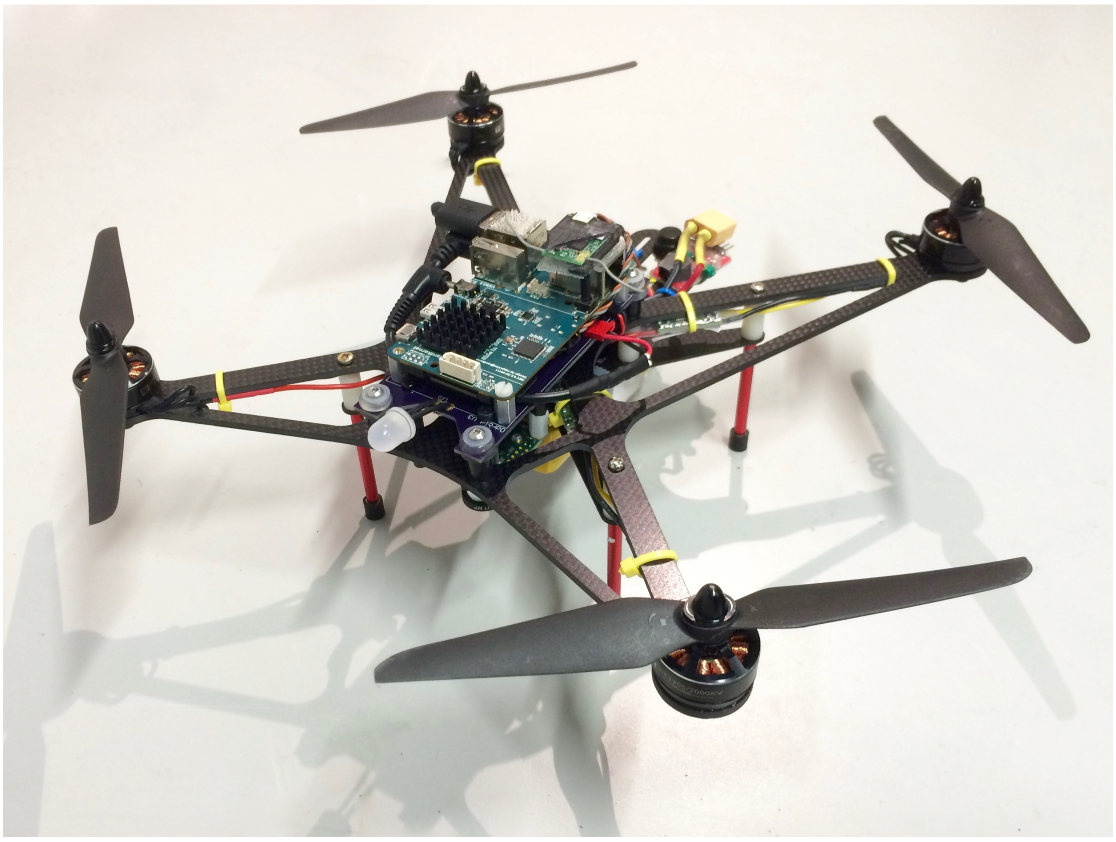


Figure 5.2: The final platform.

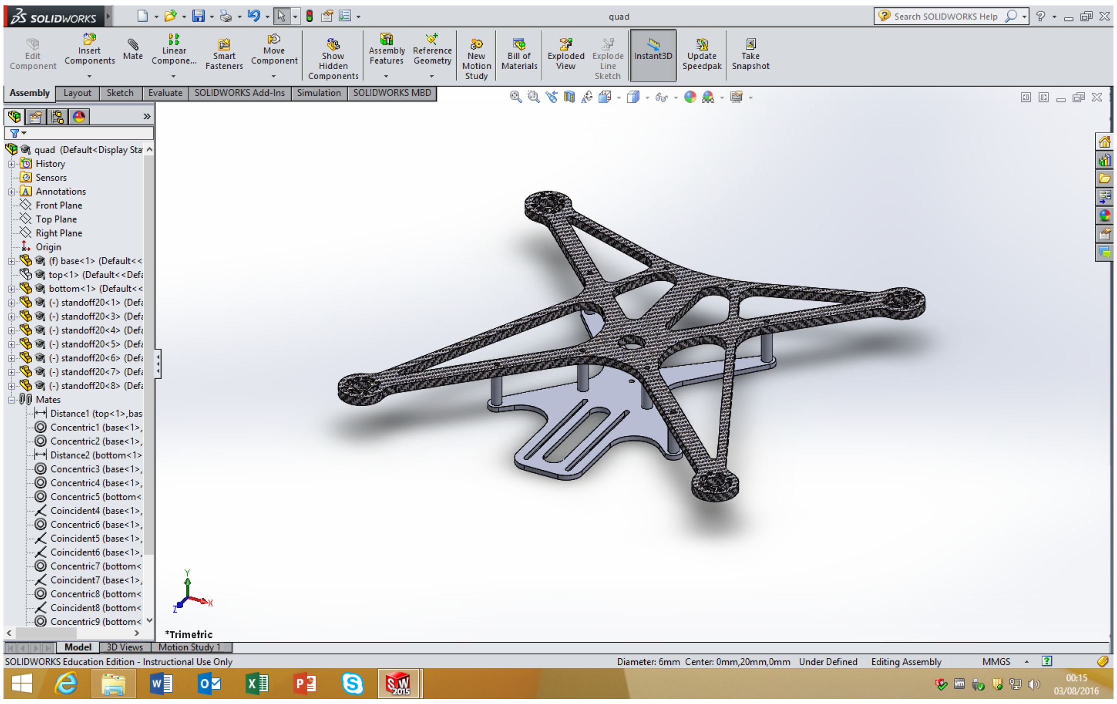


Figure 5.3: SolidWorks chassis model.

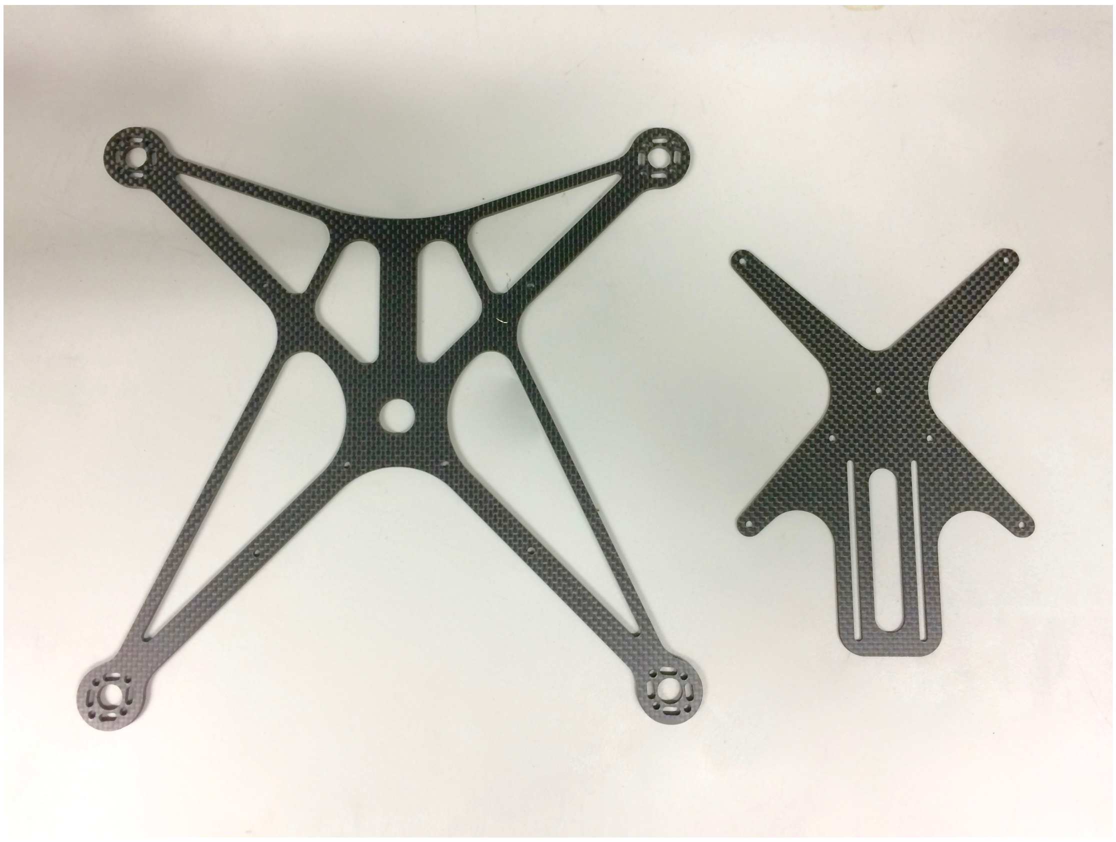


Figure 5.4: The main chassis structure.

interface for silicon damper, onboard linux computer and downward facing camera are designed on the board (four large holes at the corner). Therefore, the autopilot board directly attaches onto the main frame through silicon dampers, and the downward facing camera and the linux computer are hard fixed onto the autopilot by standoffs. This helps to fix all the sensors relative positions, additionally, the mass of all the electronics helps further damping the vibration from the motors. This is also shown in Figure 5.5.

The components, as shown in solid yellow in Figure 5.9, are hand-soldered on the autopilot board. They include a IMU sensor package board, an embedded processor board, a voltage regulator and an indicator LED:

1. **Main processor board** is based on Teensy 3.1 MCU board. It is an ARM based Arduino compatible development board, which features very small form factor ($35 \times 18\ mm$) and fast processor (ARM Cortex–M4 with up to 96 MHz clock speed). The fast speed and small size is ideal for a flight controller.

2. **IMU** is based on the FreeIMU sensor suite by Varesano (2013). It includs a MPU6050 gyroscope-accelerometer combo-chip, a HMC5883L magnetometer and MS5611-01BA high resolution pressure sensor. However, only the MPU6050 chip is used in this implementation. The orientation fusion estimation uses the library provided with the sensor.

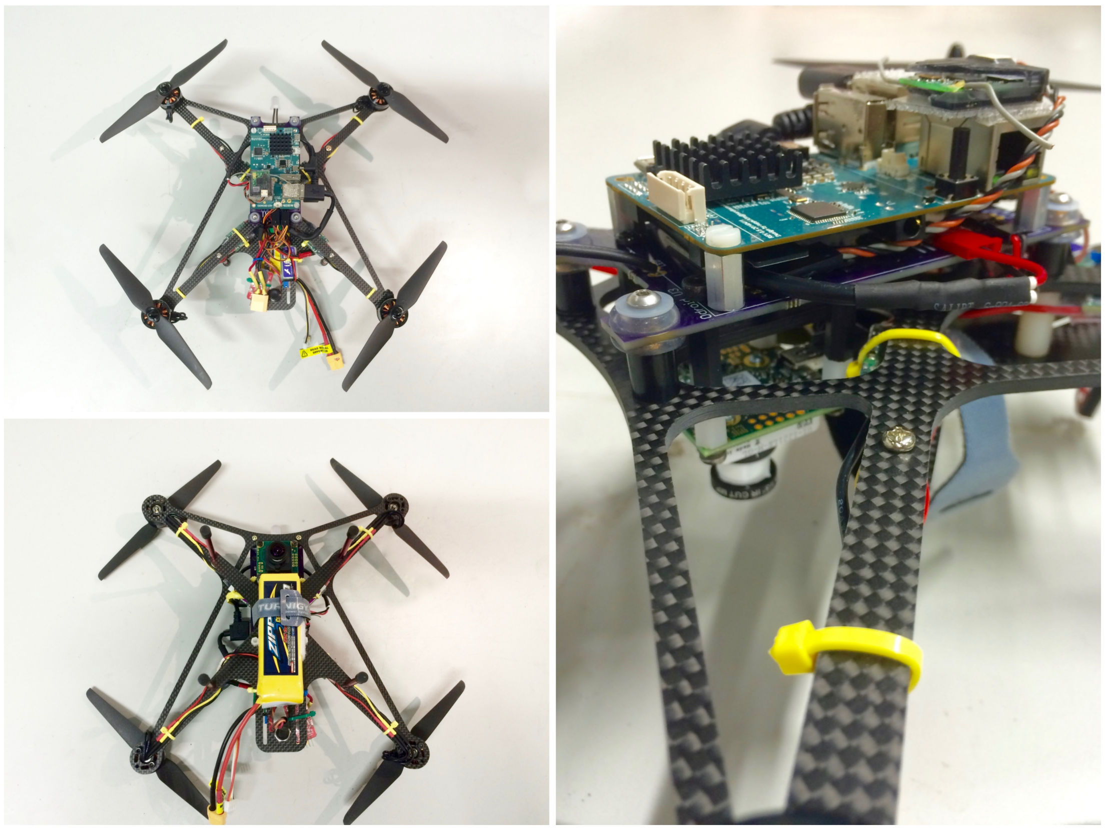


Figure 5.5: The final platform from different viewing angle.

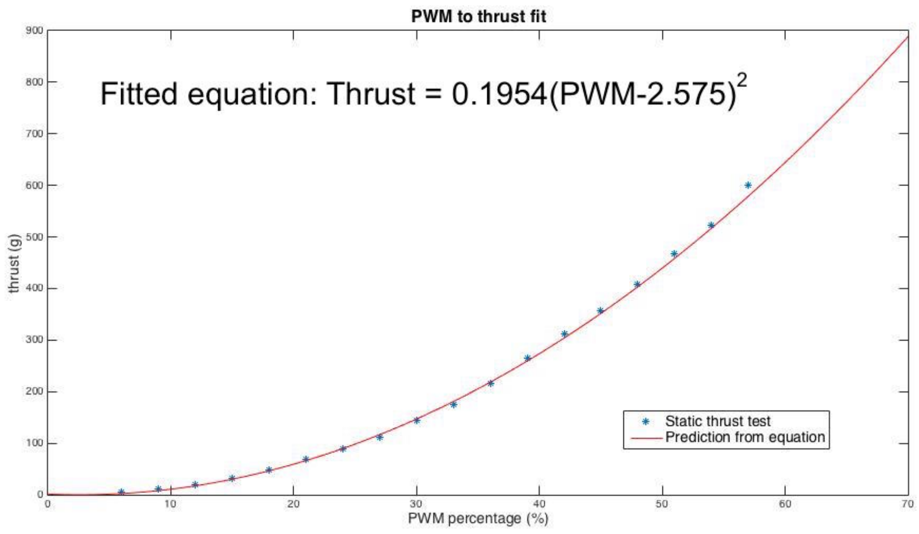


Figure 5.6: Propulsion system modelling at 12 v.

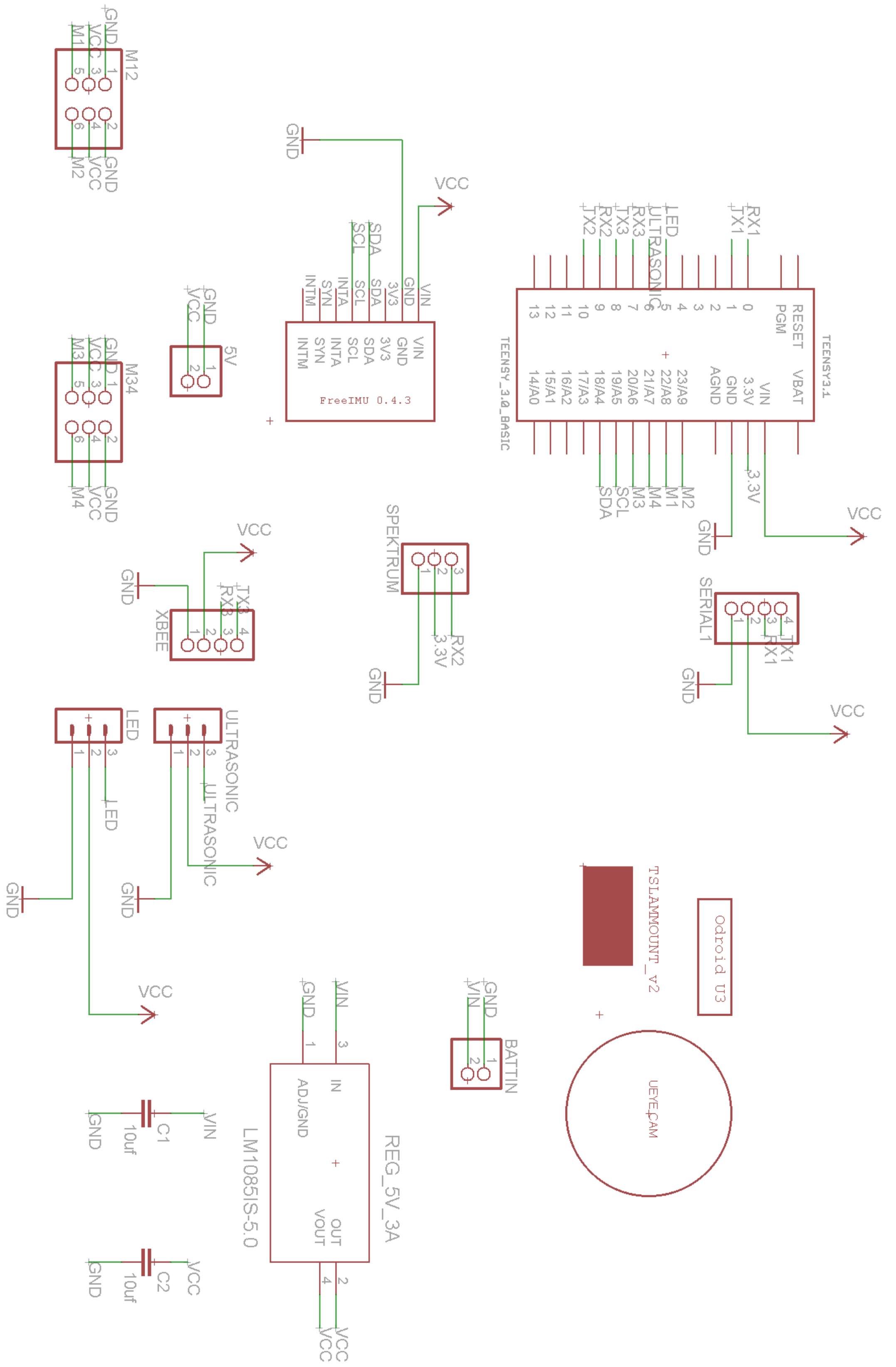


Figure 5.7: Autopilot schematic.

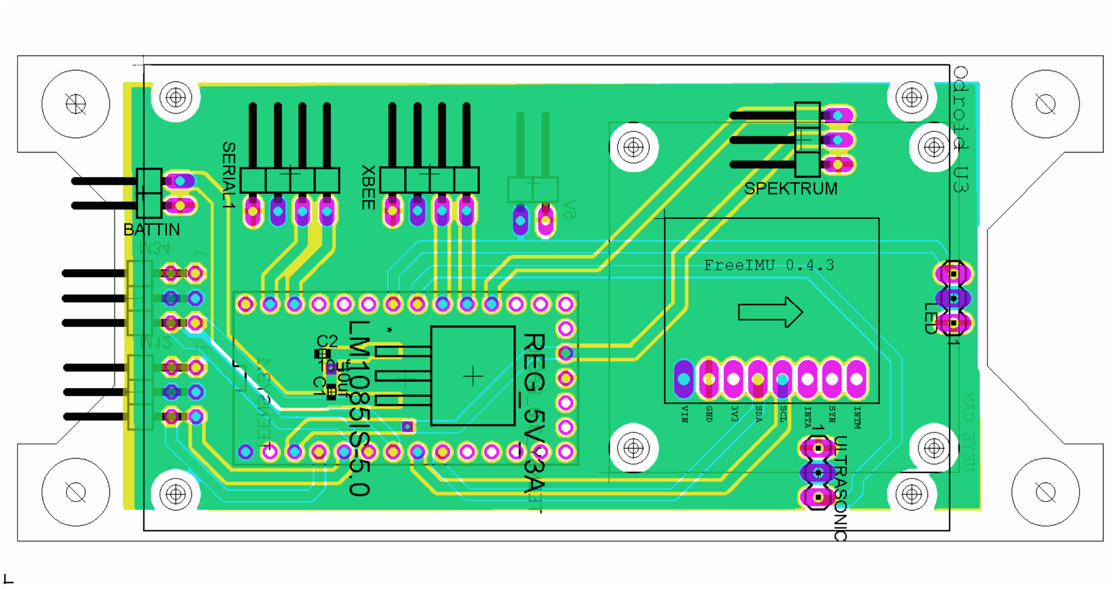


Figure 5.8: Autopilot PCB fabrication diagram.

3. **Voltage regulator** converts the 3 cell battery voltage (about 12 V) to stable 5V output, to power the autopilot as well as the onboard computer. It is a regular $5\ V\ 5\ A$ linear voltage regulator.
4. **Indicator LED** is controlled by the main processor board. The different lighting pattern indicates the system states and the healthiness of the pose estimation in real-time. This is very useful for debugging the faulty operation, and quick pilot recovery from system failure.

The electronic system signal interaction diagram between onboard electronics can be seen in Figure 5.10. Note that different from the previous autopilot designed described in Section 3.5.3, the Teensy board now commands the ESCs directly through PWM, instead of going through a dedicated servo controller in between. This design avoids the extra complexity and reduces the communication latency by about 0.02 second, which results in a much more responsive control.

5.2 Robust Integration of Visual-inertial Feedback

In order to involve the visual-inertial state estimation into the quadrotor control loop, we must ensure the reliability and safety of the state estimation to handle all the possible situations could encounter. Therefore, as shown in Figure 5.11, careful heuristic reasoning was designed to enable smooth initialisation and re-initialisation, ensure the tracking robustness, as well as interaction with autopilot for failure handling. This will be discussed in detail in following subsections.

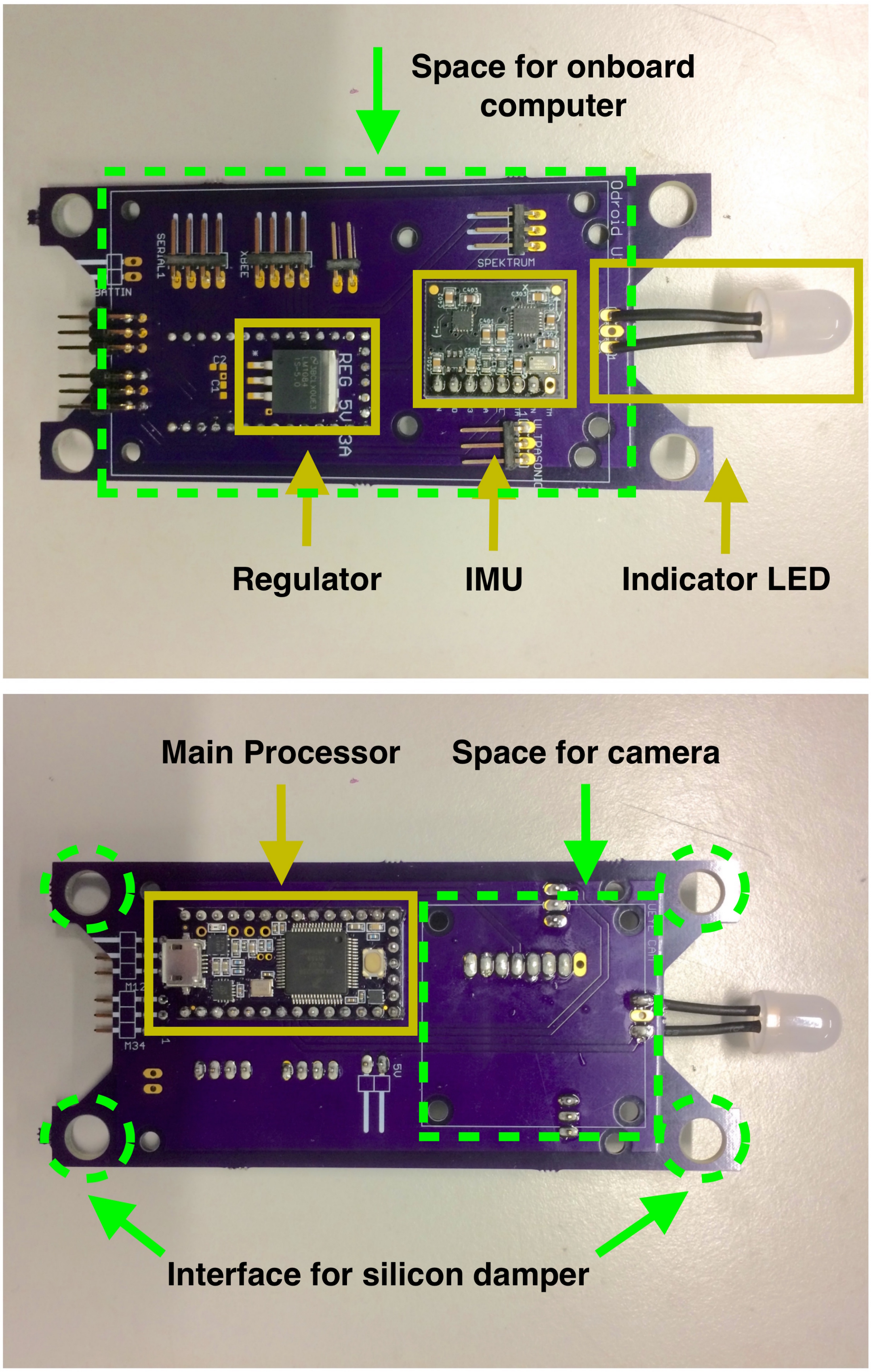


Figure 5.9: Autopilot board manufactured.

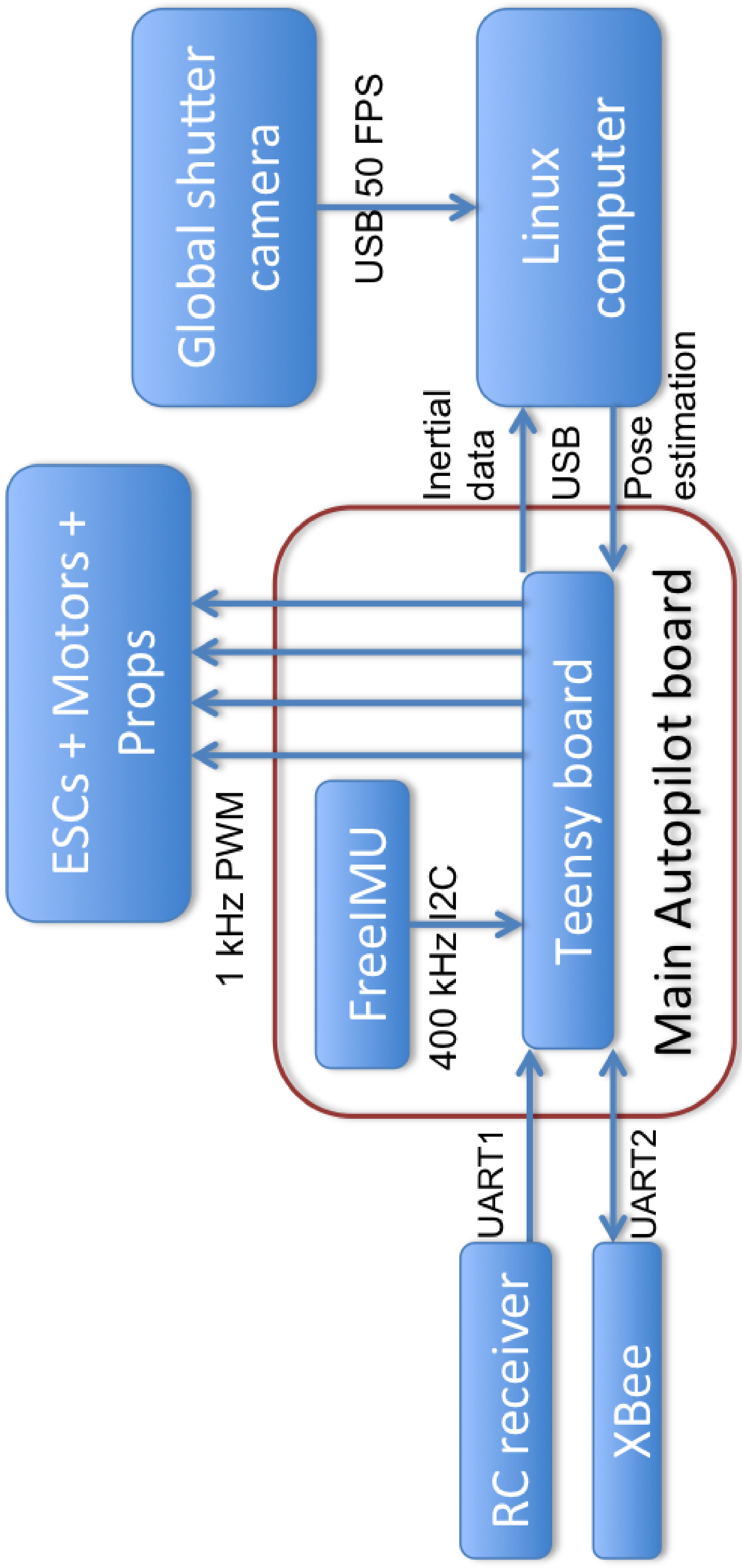


Figure 5.10: System signal interaction diagram.

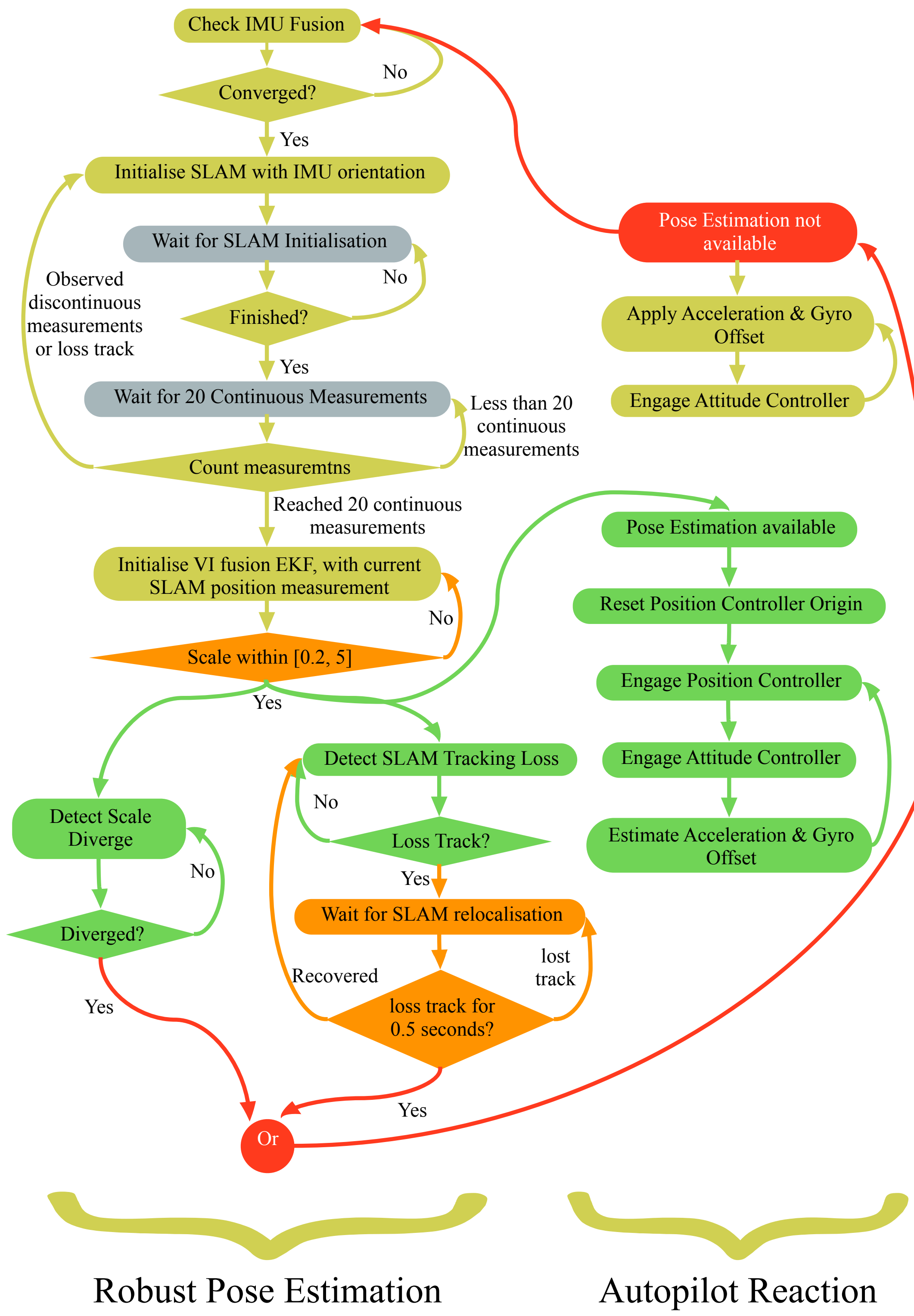

Check IMU Fusion
Converged?
No
Yes
Initialise SLAM with IMU orientation
Wait for SLAM Initialisation
No
Observed discontinuous measurements or loss track
Finished?
Yes
Wait for 20 Continuous Measurements
Less than 20 continuous measurements
Count measuremtns
Reached 20 continuous measurements
Initialise VI fusion EKF, with current SLAM position measurement
No
Scale within [0.2, 5]
Yes
Detect Scale Diverge
No
Diverged?
Yes
Detect SLAM Tracking Loss
No
Loss Track?
Yes
Wait for SLAM relocalisation
Recovered
lost track
loss track for 0.5 seconds?
Yes
Or
Pose Estimation not available
Apply Acceleration & Gyro Offset
Engage Attitude Controller
Pose Estimation available
Reset Position Controller Origin
Engage Position Controller
Engage Attitude Controller
Estimate Acceleration & Gyro Offset
Robust Pose Estimation
Autopilot Reaction

Figure 5.11: Smooth initialisation and reinitialisation logic flow chart.

5.2.1 Pose healthiness check

In order to ensure control safety, when the visual-inertial system is in healthy and active operation, the healthiness of the pose estimation must be constantly monitored at all time, in parallel with the active autopilot positional control. In the case of estimation failure, the system has to properly detect the failure event, and perform reinitialisation procedure. In our case, when the visual inertial system is initialised, the scale value is used as the quick indication of the healthiness of the EKF estimation. Here the scale is treated as valid when it is between 0.2 to 5. While concurrently, the SLAM tracking status is given by the SVO package, which is a direct indication of the SLAM performance. Note that it allows SVO to temporarily lose track in order to give the SVO package a chance for its internal re-localisation procedure, however, it has been tested that when temporary tracking loss persists more than 0.5 second, the chance for re-localisation is very low.

Only in the case that both the scale is in valid range and SVO is in track, the autopilot position controller can be engaged. Furthermore, whilst the position controller is engaged, the vehicle body acceleration offset, which might be caused by a constant wind direction or the body centre of mass offset, are estimated by integrated the positional error while control. Besides, the gyroscope rate measurement offset are estimated, when fusing the SLAM heading measurement back to IMU attitude estimation.

The pose healthiness check is indicated as green and orange in the 'robust pose estimation' column in Figure 5.11. Any continuously unhealthy estimation must trigger the reinitialisation of both SLAM and visual inertial EKF, and disable the position control, which will be discussed in the following sections.

5.2.2 Smooth initialisation and re-initialisation

The smooth initialisation and re-initialisation process is summarised as the yellow and grey in the 'robust pose estimation' column in Figure 5.11.

The sensor fusion framework developed in Chapter 4 requires the heading of the IMU sensor frame and SLAM sensor frame are aligned before initialisation. This is crucial for both initialisation before the flight, and re-initialisation in the flight. Particularly, assuming that the IMU orientation estimation already converged, in the event of SLAM (re)-initialisation, the SLAM algorithm was modified, so that it reads the current orientation from IMU, and use this orientation and (0, 0, 0) as its initial pose. By doing this, the both frames are aligned at (re)-initialisation.

Once the SLAM is initialised, the visual inertial EKF filter would initialise after 20 continuously successful SLAM measurements. This was set in order to void unreliable SLAM measurements, which generally presents after unsuccessful initialisation. It is often shown as discontinuous pose giant jump immediately after initialisation, with repeating tracking loss, due to the faulty triangulation. If not taken care of, the EKF can easily diverge, resulting in a corrupted filter estimation.

In terms of re-initialisation, it effectively performs the same with the initialisation procedure. However, since the SLAM re-initialises at the origin, the EKF also has to re-initialise at the origin. Indeed, when re-initialising in the flight, this would reset the origin of the pose estimation to the current position. In this case, the position controller of the autopilot must be notified for this re-initialisation action, and reset the origin of the position controller at the same time. This was achieved by the communication between Teensy board and Linux computer through the USB.

Note as stated in the pose healthiness check in Section 5.2.1, since it takes more than 0.5 second to confirm the SLAM tracking loss, the total reinitialisation process takes about 1 second. Moreover, the EKF filter takes roughly 1 second to converge the map scale factor from initial guess to the true value, while the exact converging time depends on the difference between initial guess and the true value. During this re-initialisation period, no positional or velocity estimation is available, thus the SLAM failure handling process must takes over.

5.2.3 SLAM failure handling

A short (less than 0.5 second) and temporary tracking loss can be recovered internally in the SVO package. The tracking loss persist for more than half a second will hardly have a chance to recover, thus a reinitialisation of the map is needed, which can be handled by the smooth reinitialisation stated in Section 5.2.2. If the reinitialisation continuously fails, it is likely that the system encounters hardware failure or suffers from bad lighting condition.

During reinitialisation process, the autopilot does not have any positional or velocity information, thus the it must disengage the position controller, whilst maintaining an open loop control to minimise the vehicle acceleration in all directions. This can be effectively achieved when acceleration offset and gyroscope rate measurement offset has been estimated during active position control. This process can be found in as the yellow steps in autopilot reaction column in Figure 5.11.

5.2.4 Visual-inertial synchronisation and delay compensation

The synchronisation between visual and inertial data significantly effects the performance of the fusion system. Here, the typical data sampling flow is shown in Figure 5.12. Given the 5.5 ms IMU sampling period, the 20 ms processing delay from SLAM computation causes a mismatch against the IMU measurement update, by about four measurements. Here, in order to deal with this situation, a software synchronisation was used. Specifically, a delay buffer was developed for the IMU in the EKF software package, which stores and lists the latest four measurements from the IMU. Then, at each IMU sampling action, the buffer publishing and popping the oldest

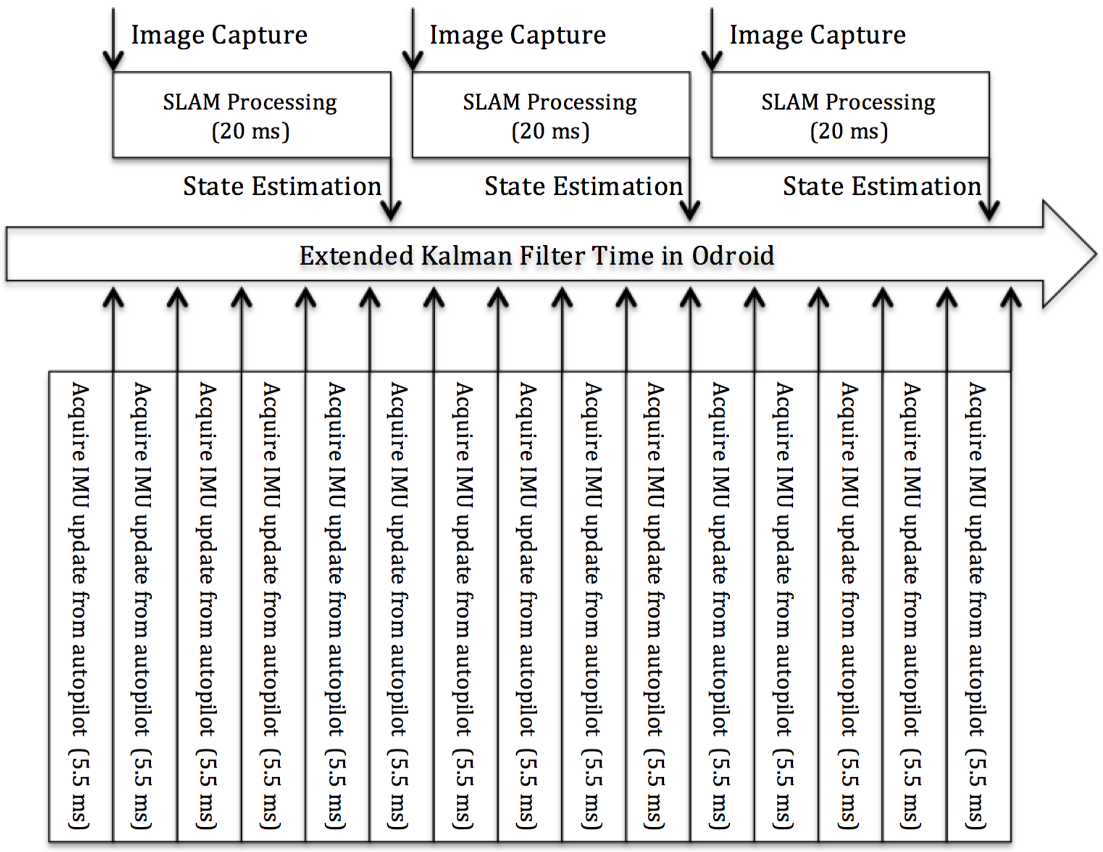


Figure 5.12: Sampling time mapping indication.

IMU measurement, while storing the new measurement. This buffer action effectively delays the IMU measurements by 4 measurements, which roughly synchronised visual and inertial sampling action.

However, this delay operation will result in significant delay (20 ms) in the overall output from the EKF package. This delay in the feedback consequently impacts the overall control performance of the UAV. Especially for the vehicle velocity feedback, the 20 ms delay will easily cause control oscillation with high differential gain, whilst the delay in position measurement only has a minor effect. Therefore, in order to compensate for this delay in the velocity feedback, a pre-integration buffer was developed in the autopilot, which always stores the latest 10 IMU measurements (note: IMU update rate in the autopilot is 400 Hz), and integrates the 10 dynamic accelerations to obtain the velocity compensation value.

5.3 Overall Position Control Performance Result

The final control performance was tested and the test results are shown in this section, demonstrating the accuracy and robustness of the system. Three trials were carried out: hovering test, trajectory following test and disturbance recovery test. For each trial, the 3D trajectory graph and 2D command-response graph of all 6 DOF are shown individually.

The high level positional waypoints and trajectory are commanded from user remote controller directly. The remote controller stick was mapped to velocity command and the velocity is integrated to obtain the positional command. Given the high measurement accuracy of the onboard visual inertial sensor fusion as demonstrated in the Section 4.6, the true vehicle position can be sufficiently obtained by the onboard visual inertial estimation. The test trials were conducted in an indoor environment, as shown in Figure 5.13. Colourful carpets were used to provide rich contrast on the ground for the ease of visual tracking.

5.3.1 Hovering test

This test shows the stability of the system hovering against towards a single 3D position point. Figure 5.14 shows that the vehicle hovers around the target position. Note that the target position point (red) is covered by the vehicle position measured (blue), and the position motion in each axis were recorded individually in Figure 5.15, with the position command from user are represented by red line, and real time position measurements are represented by blue line. In the case of hover the red lines are constants, and blue lines approach them as close as possible. Finally, Figure 5.16 shows the performance of the inner attitude controller, with the angle commanded by the outer position controller about each axis is shown in red and the true angles reaction measured from IMU are shown in blue.

As shown in Figure 5.14 and 5.15, the system hovers within 10 centimetre accuracy with no external disturbance. While noticeably, as shown in Figure 5.15, it experienced micro (about 0.05 rad) oscillation at about 1 Hz especially in world x-axis, this is believed to be caused by the imperfect 3D velocity measurement from visual inertial fusion, due to the manual synchronisation error between visual slam and IMU. Although the attitude has the integral term to compensate small centre of mass offset, it is limited to a very small value. Thus, as shown in Figure 5.16, the pitch angle shows an offset, due to the significant center of mass offset along the quadrotor body x-axis, while it was compensated by the integral term of the outer position controller, as the result, the true pitch angle still kept about 0 rad.

5.3.2 Trajectory following

This test shows the stability of the system tracking a dynamic target 3D trajectory. Figure 5.17 shows the target horizontal square-like trajectory (red) followed by the vehicle estimated position measurement (blue), and the position motion in each axis were recorded individually in Figure 5.18, with the position command from user are represented by red line, and real time position measurements are represented by blue line. In the case of hover the red lines are trajectories commanded by user, and blue lines approach them as close as possible. Finally, Figure

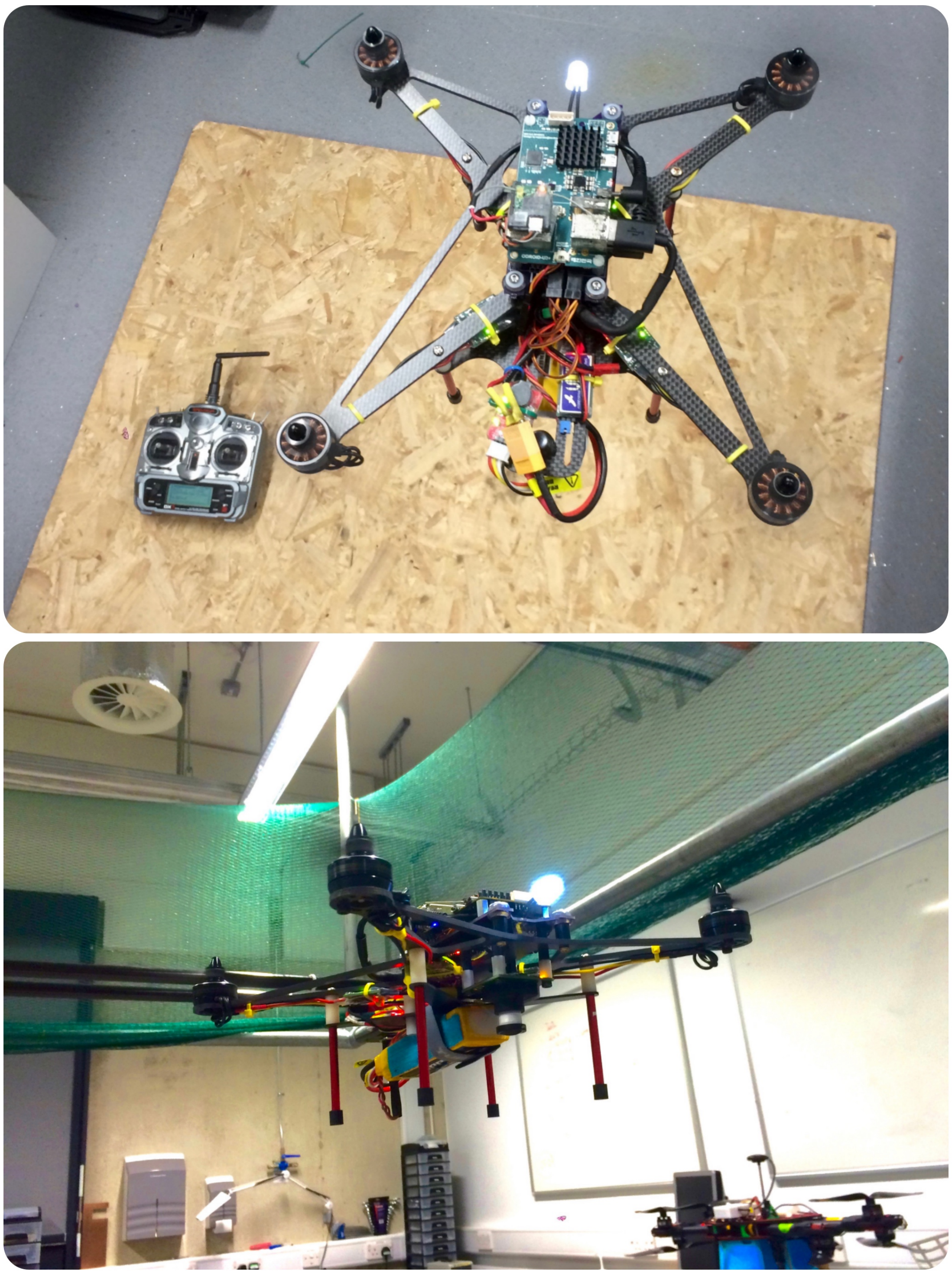


Figure 5.13: Testing environment setup.

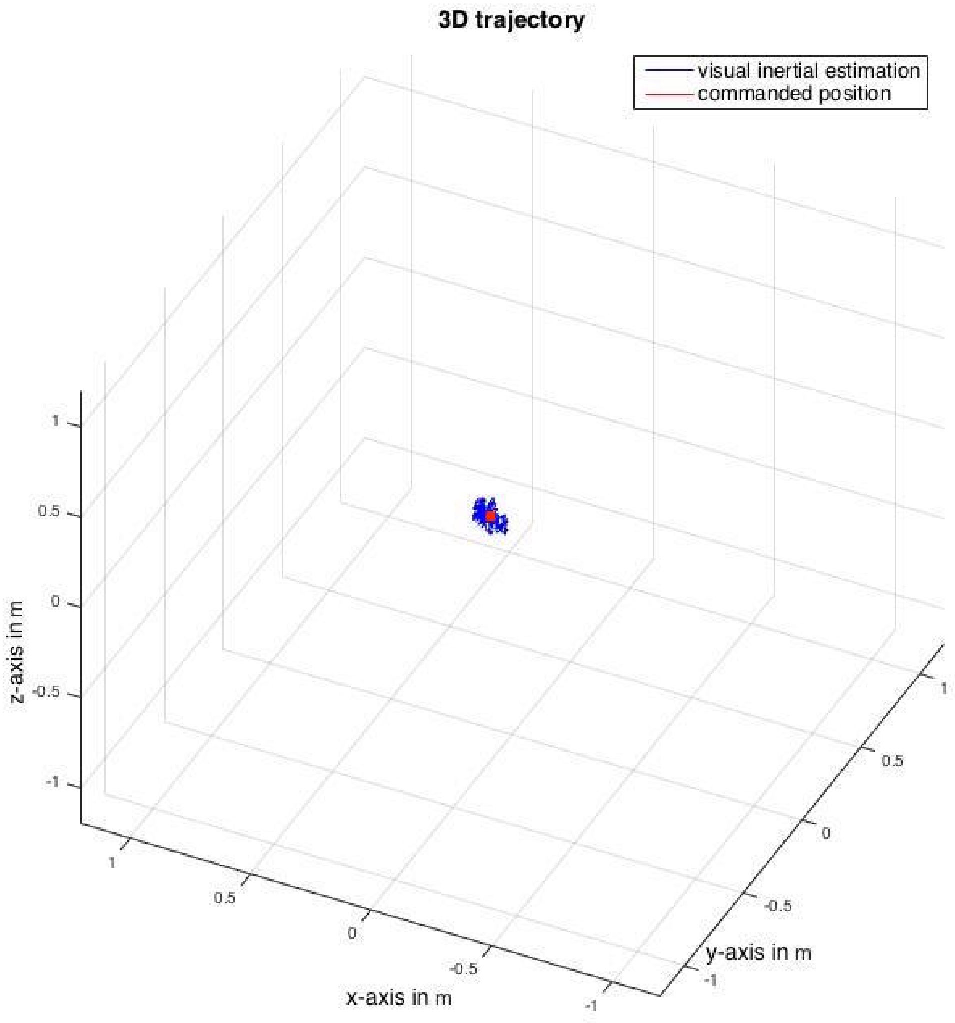


Figure 5.14: 3D trajectory of hovering.

5.19 shows the performance of the inner attitude controller, with the angle commanded by the outer position controller about each axis is shown in red and the true angles measured from IMU are shown in blue.

The horizontal square-like trajectory was commanded from user remote controller. As shown in the 3D view in Figure 5.17 and in time series in Figure 5.18, the trajectory can be sufficiently tracked with the error less than 20 centimetre, for any motion directions. Besides, as shown in Figure 5.19, the excellent attitude controller performance was demonstrated in this dynamic operation. It is shown that the pitch control integral were built up, allowing the control without steady state error. This is different from the hovering test, because the battery offset is different, due to the manual installation.

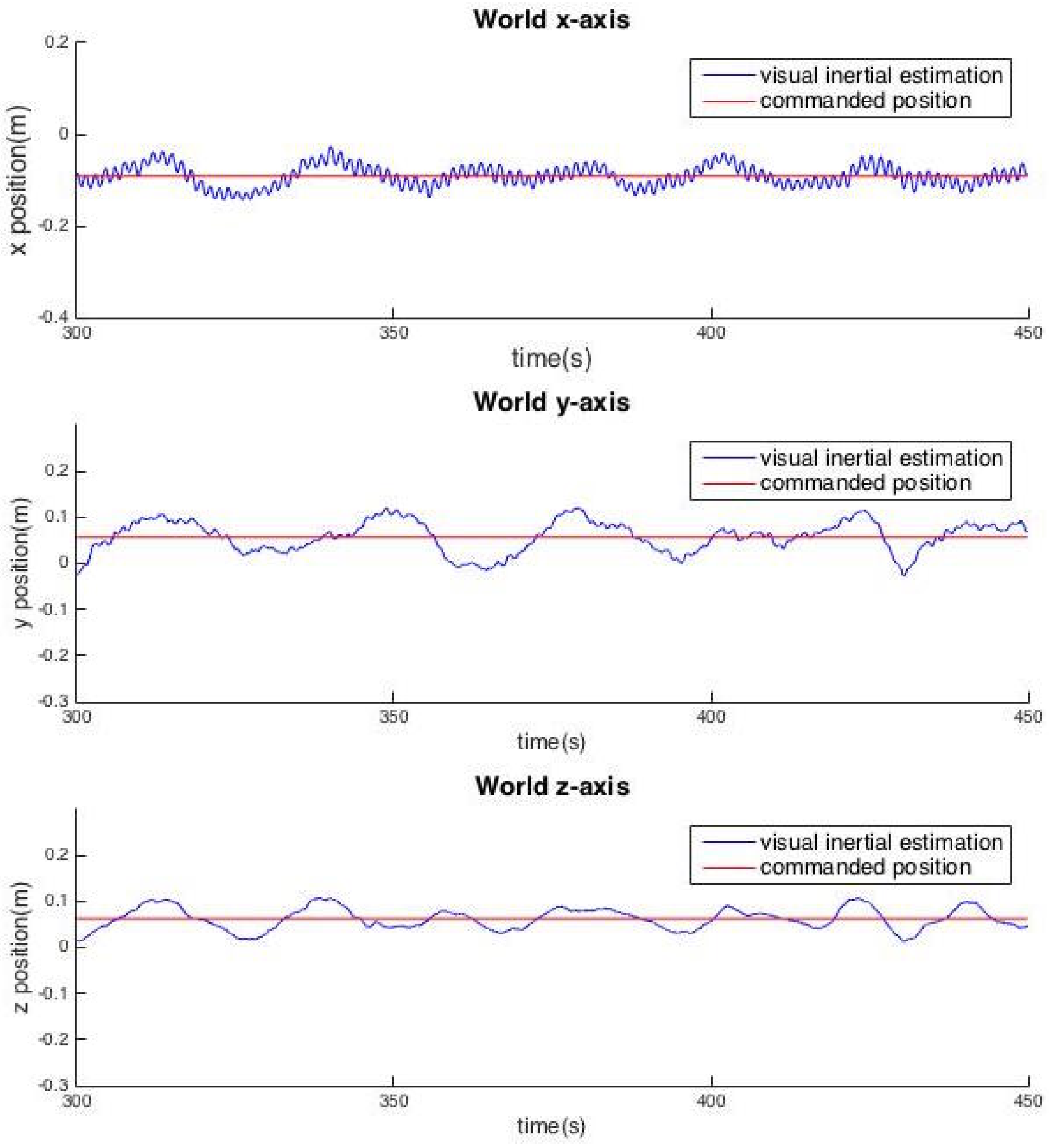


Figure 5.15: Positional trajectory of hovering in m.

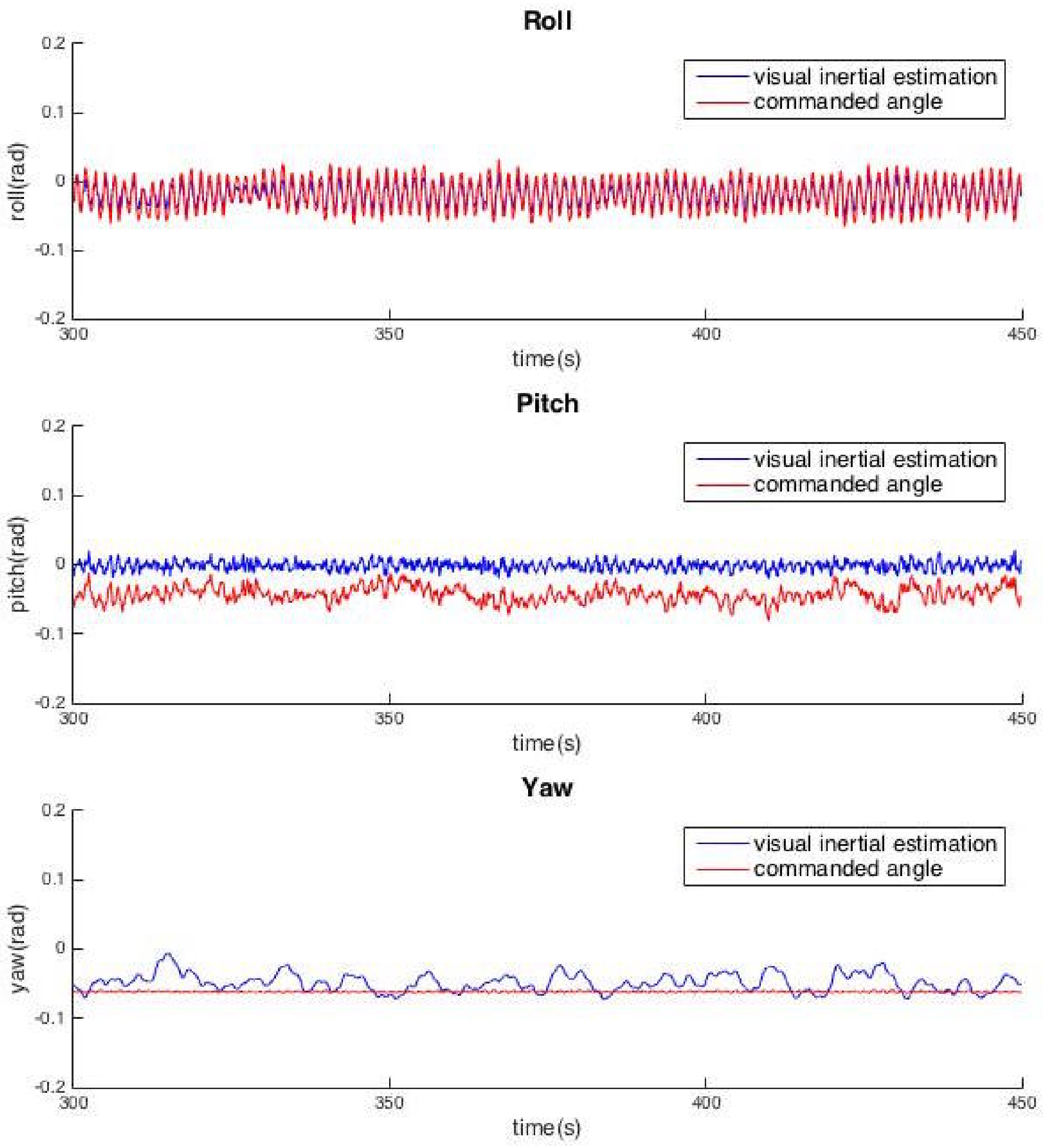


Figure 5.16: Angular trajectory of hovering in rad.

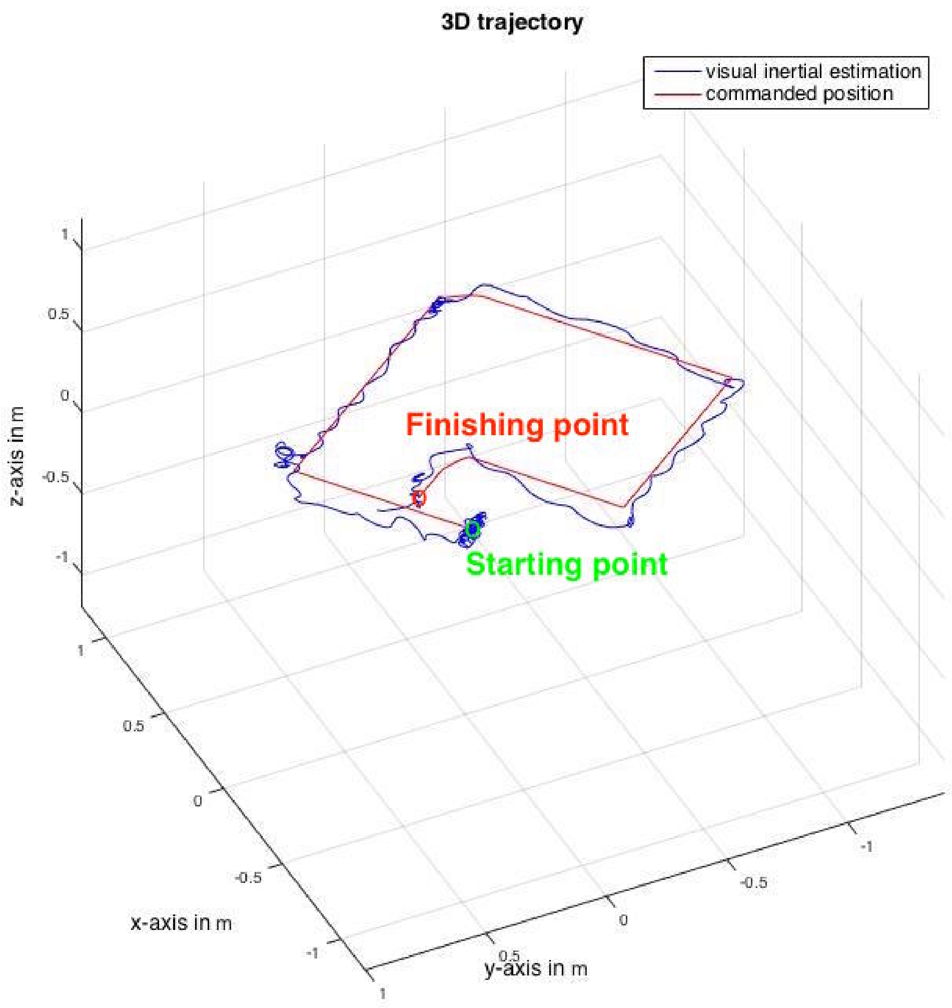


Figure 5.17: 3D trajectory of tracking square-like path.

Also note that the trajectory following was tested in a relatively small room, where the down wash from the propeller can be reflected by walls, causing the disturbance.

5.3.3 Disturbance recovery

This test shows the stability of the system with the focus on disturbance recovery in the hovering situation. The system was initially commanded to hover, then the quadrotor was pulled three times to one direction, simulating the disturbance. When released, the quadrotor recovers to the initial hover position. Figure 5.20 shows the vehicle position measured (blue) along the process against the target position point (red). It is shown clearly that the vehicle was pulled towards one direction and recovers with a small overshoot.

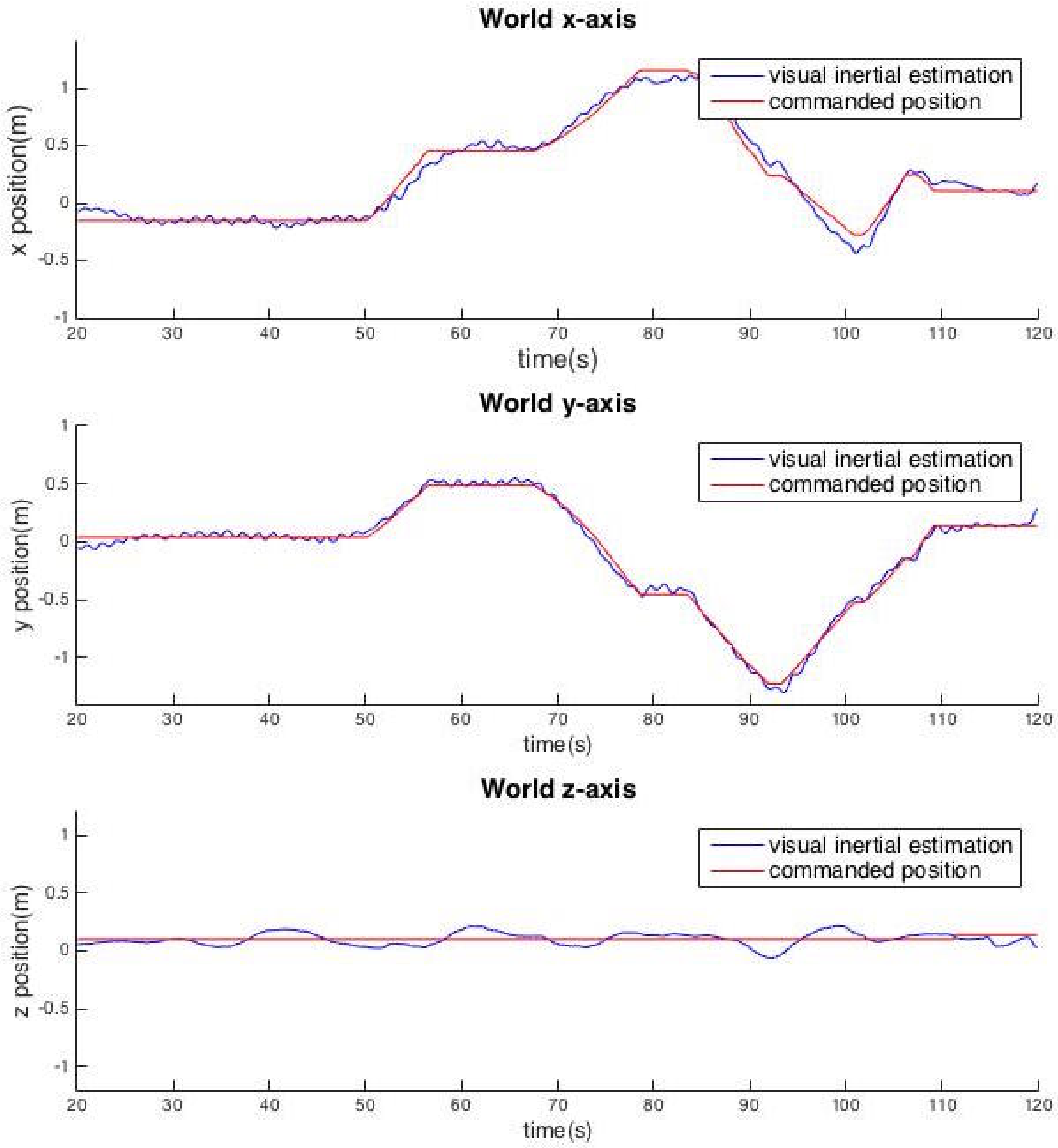


Figure 5.18: Positional trajectory of tracking square-like path for hovering in m.

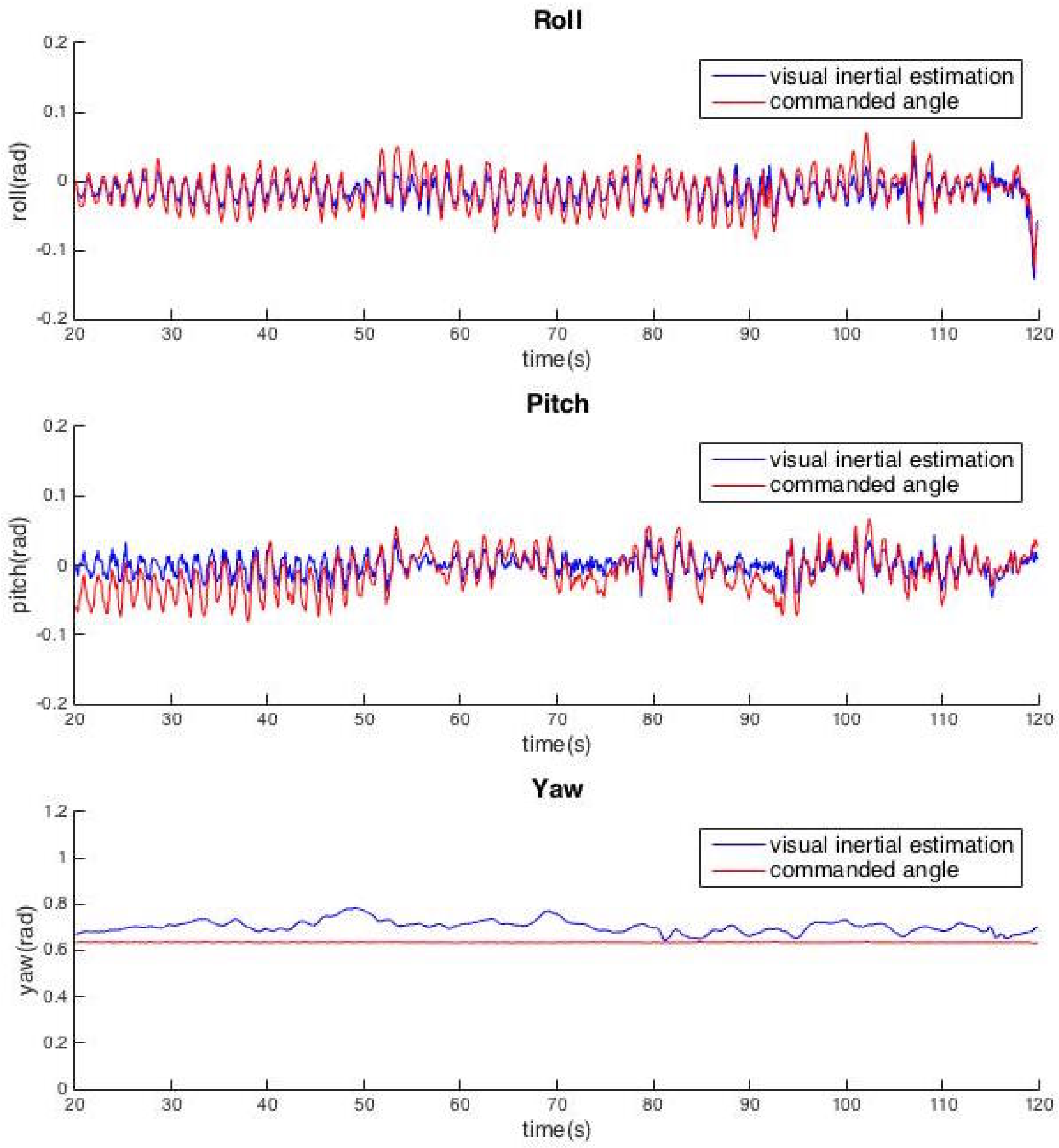


Figure 5.19: Angular trajectory of tracking square-like path in rad.

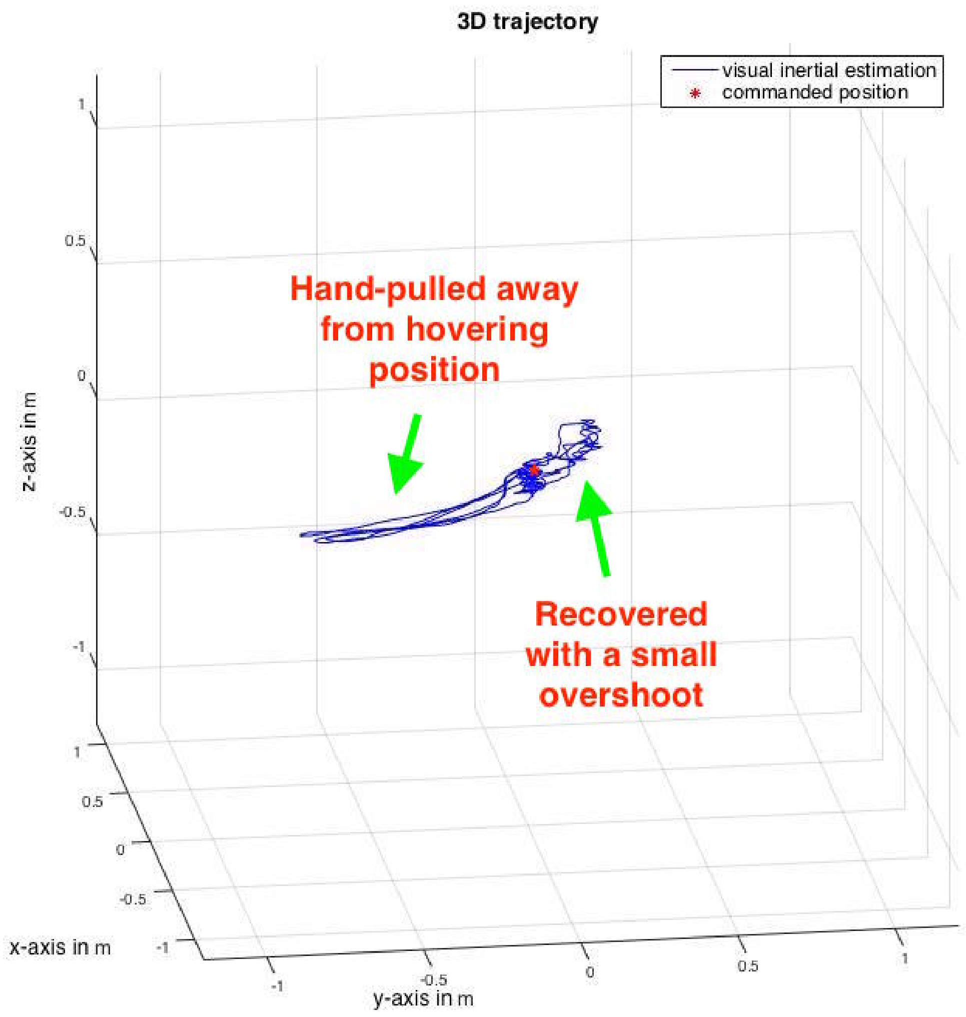


Figure 5.20: 3D trajectory of disturbance recovery.

For more detail, the positional motion in each axis was recorded individually and shown in Figure 5.21, with the position command from user are represented by red line, and real time position measurements are represented by blue line. In the case of hover the red lines are trajectories commanded by user, and blue lines approach them as close as possible. It can be clearly seen that three pull actions were conducted at 230 second, 250 second, 290 second respectively. The quadrotor was pulled towards negative x-axis for 0.7 m and negative z-axis for about 0.3 m.

Finally, Figure 5.22 shows the performance of the inner attitude controller, with the angle commanded by the outer position controller about each axis is shown in red and the true angles measured from IMU are shown in blue.

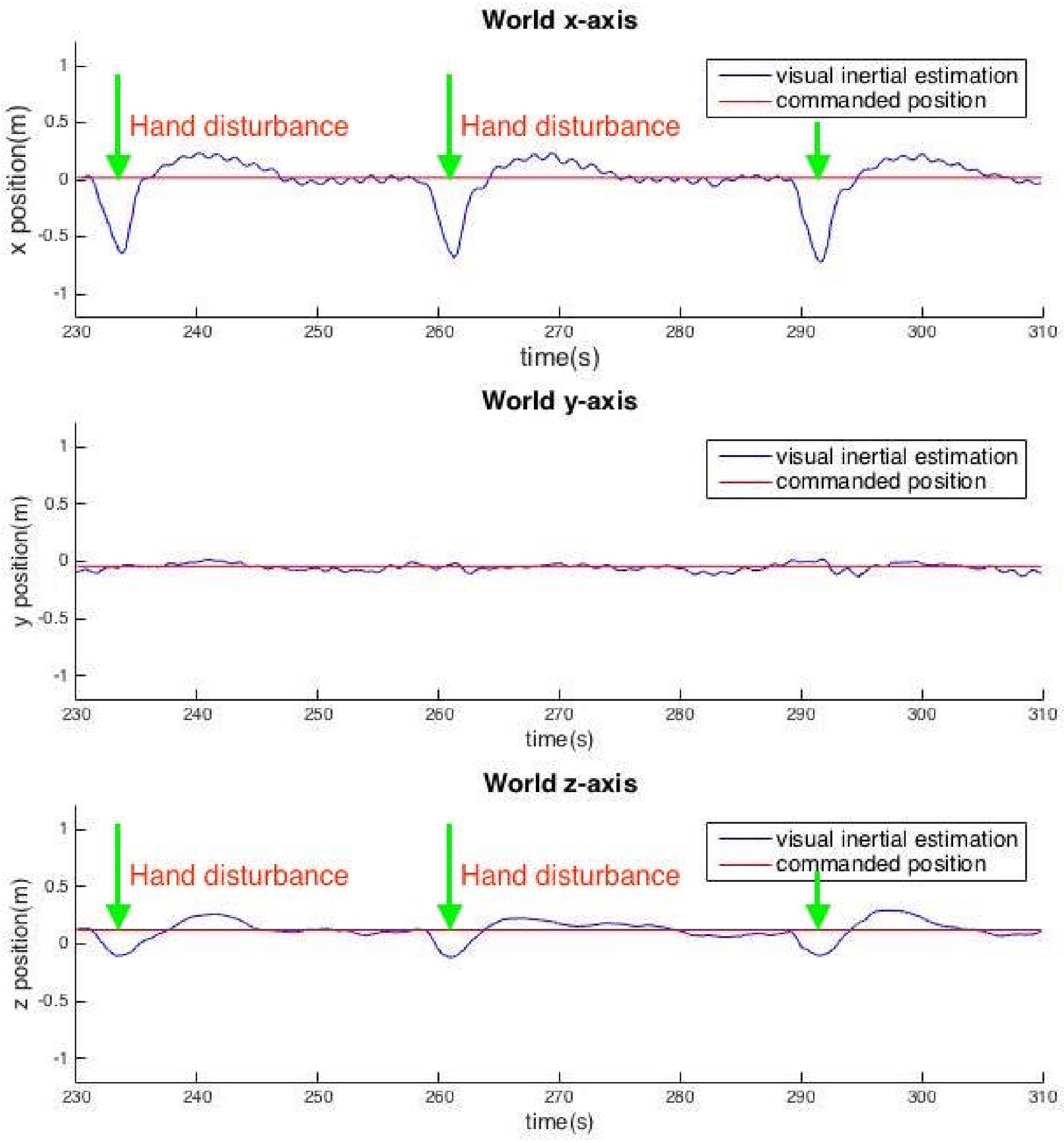


Figure 5.21: Positional trajectory of disturbance recovery in m.

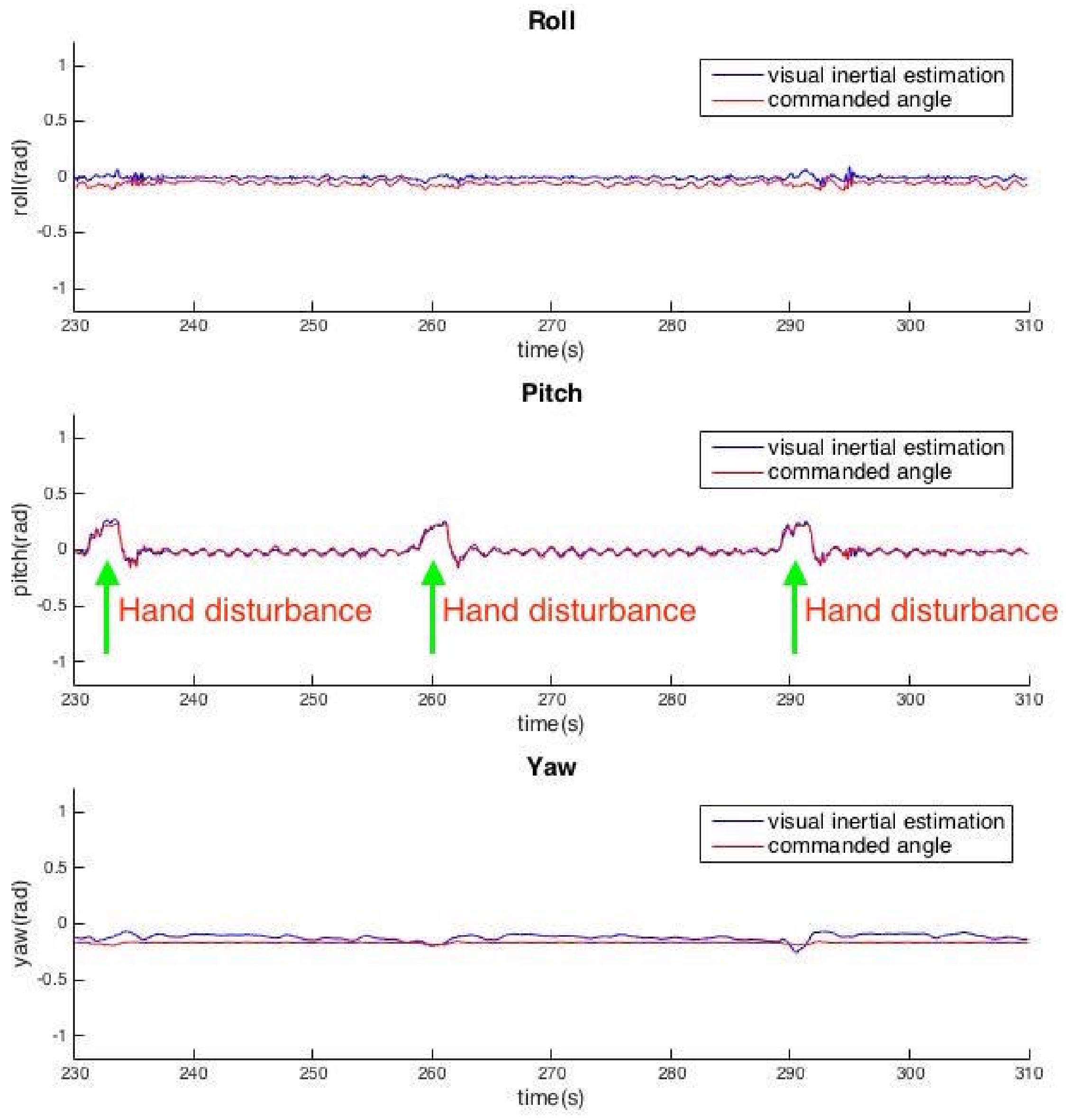


Figure 5.22: Angular trajectory of disturbance recovery in rad.

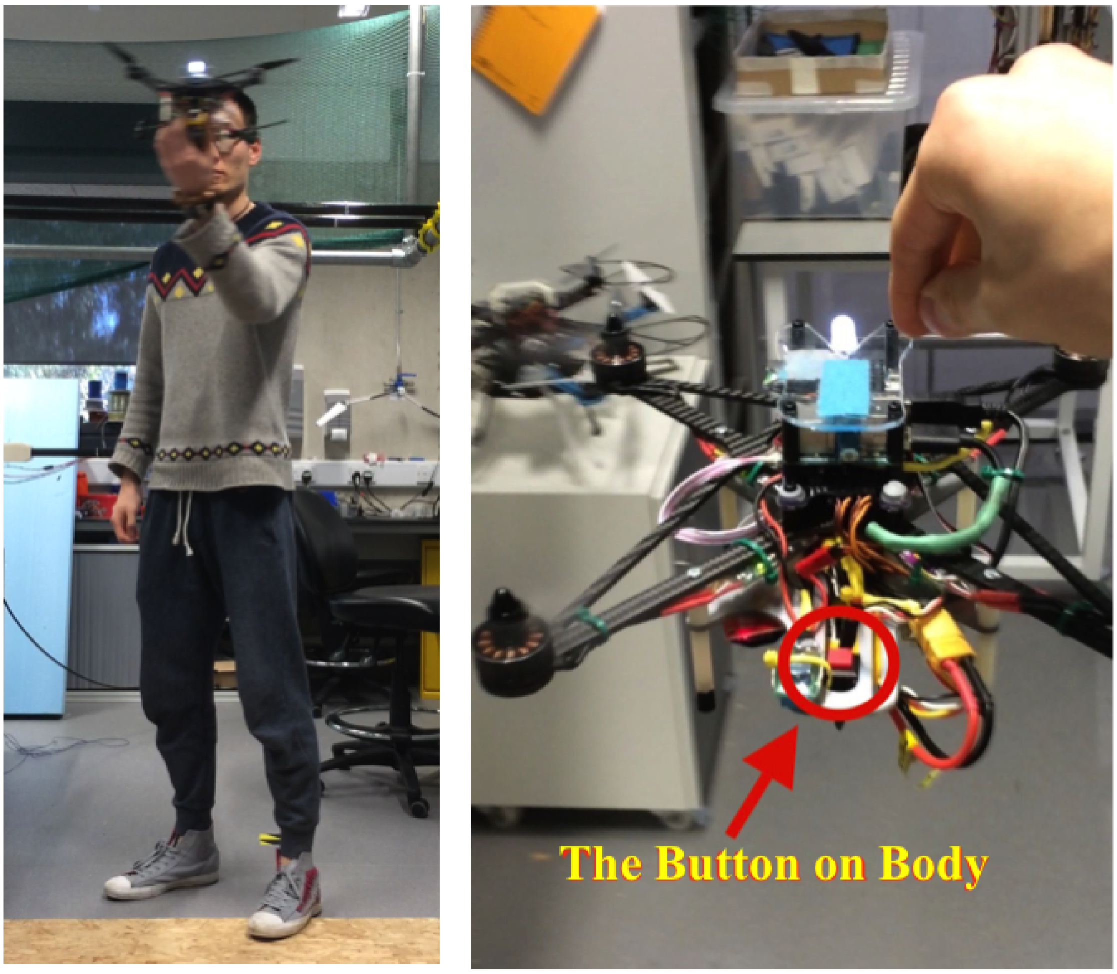


Figure 5.23: Hand-launching quadrotor.

5.4 Hand Launching with a Button on Vehicle Body

Previous sections have shown that the integrated system is capable of manoeuvre with centimetre accuracy. However, with a downward facing camera, it struggles to perform close to ground operation (less than 10 cm between camera and the ground), and it does require movement for visual SLAM initialisation. Especially when launching the quadrotor from ground, the pilot always needs to perform a manual takeoff to initialise the visual SLAM, which is not ideal for the general usage of such power-on-and-go system.

The results in previous section shows that the system is accurate and robust enough to perform close to human flight. In this section, we demonstrate the design of an intuitive method to launch this system by hand(s). This launch method not only allows the human pilot to intuitively position the quadrotor in the 3D space within reach, but also solves the problem of downward-facing camera when operating close to the ground. The typical use case is demonstrated in Figure 5.23.

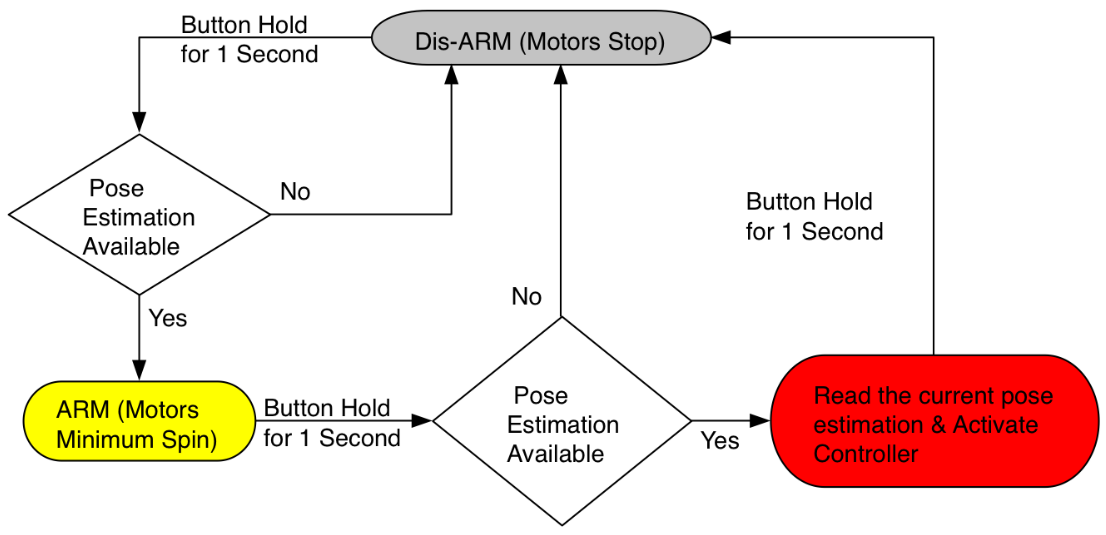

Button Hold
for 1 Second
Dis-ARM (Motors Stop)
Pose Estimation Available
No
Yes
ARM (Motors Minimum Spin)
Button Hold
for 1 Second
No
Pose Estimation Available
Yes
Read the current pose estimation & Activate Controller
Button Hold
for 1 Second

Figure 5.24: The state machine for hand-launching quadrotor.

Specifically, this hand launching method was achieved by installing a push button on the quadrotor body as shown in Figure 5.23. Therefore, by pressing the button on quadrotor body, user can easily launch the quadrotor from anywhere it is placed. To dis-arm the quadrotor, the user simply holds the same button. The entire procedure does not require any remote control activity. However, in order to ensure the visual inertial pose estimation is available before launching, user is required to slowly wiggle the platform and confirm the successful visual inertial system initialisation by watching the indicating LED on the autopilot. Thus to avoid users' miss operation, a finite state machine is designed for the safe button control. The state chart is shown in Figure 5.24.

5.5 Conclusion

This chapter has shown the implementation of closing the UAV control loop by feeding the visual-inertial state estimation back to the controller. It summarised the details of practically integrating the controller designed in Chapter 3 with the visual inertial state estimation system developed in Chapter 4 on a challengingly small 250 mm quadrotor platform. It has shown the mechanical and electronics design behind the integration. It also described the failure handling implementation towards the complete power-on-and-go system. Besides, thanks to the small size of the vehicle, it is safe to launch it from the hand. A novel launching button was designed onto the UAV body for fast and intuitive launch. Finally, test result was presented to demonstrate the effectiveness and robustness of the system performance.

The overall system accuracy, robustness and user-friendly design has demonstrated the feasibility of deploying such system in the real world applications. By relying on the drift free (for hovering) pose estimation from the SLAM based visual-inertial sensor fusion, the system enables fully autonomous drone operation in GPS-denied environments.

Chapter 6

Active Simultaneous Localisation and Mapping

Recent advances in onboard monocular vision-based simultaneous localisation and mapping (mSLAM) for unmanned aerial vehicles (UAVs) enables their autonomous navigation in GPS-denied environments. However, unlike the GPS-based navigation, the accuracy of the mSLAM-based navigation performance highly depends on the motion of the UAV and the scene within the camera's field of view. Therefore, instead of passively performing mSLAM while the UAV travels to the destination (target waypoint), we integrate the perception requirements of mSLAM into the navigation control of the UAV. Specifically, we present a novel active one-step-ahead planning algorithm which dynamically generates the optimal next action command in real-time for an autonomous quadrotor based on its current mSLAM observations, so that it not only approaches the target waypoint, but also maximising the localisation accuracy for the onboard mSLAM algorithm. Especially in scenarios similar to the one shown in Figure 6.1, where large area with low contrast occupies the significant portion of the direct path for UAV to approach the target position (indicated by yellow path), with the active planning algorithm, the UAV will choose the green path to avoid the low contrast area while approaching the target area. This active planning algorithm serves as the complementary to the state-of-the-art passive mSLAM algorithm, and aiming to improve the operational safety and robustness of the real world GPS-denied scenarios.

6.1 Introduction

Over the last decade, the Global Positioning System (GPS) has been the key to enabling the autonomy of UAVs. It provides global localization service with the best accuracy of 1-2 metres. However, recently, due to the proven weakness of GPS signal and rapid development of onboard sensing and computation capability, there has been growing interest in developing and researching alternative navigation methods for UAVs in GPS denied environments Achtelik et al.

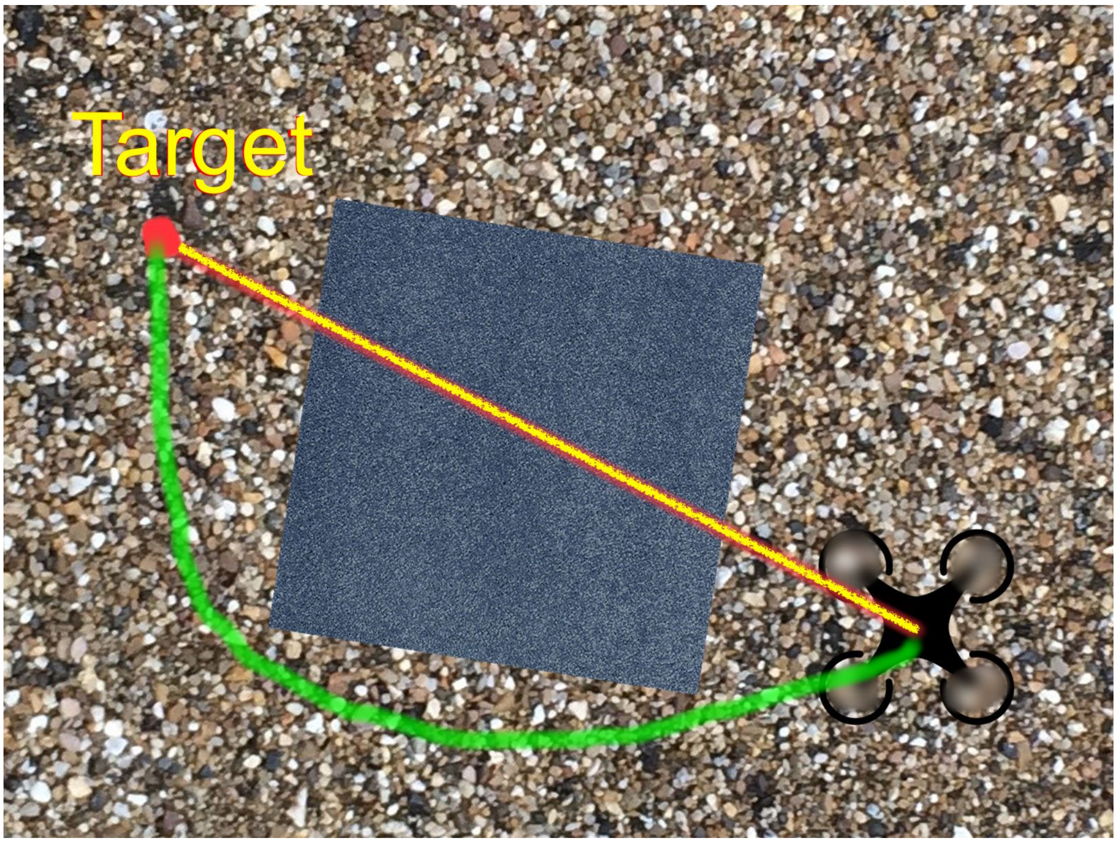


Figure 6.1: Active SLAM demonstration. (UAV automatically avoids area with low contrast.)

(2012); Bachrach et al. (2010); Bry et al. (2012); Engel et al. (2012); Jones and Soatto (2011); Shen et al. (2013a). Successful implementations will not only improve system robustness under GPS failure, but also enable a new range of applications out of GPS coverage.

Monocular visual-based simultaneous localization and mapping (mSLAM) algorithm, based on structure from motion (SFM) theory, is widely considered to be the optimal navigation solution for GPS denied environments, due to the minimum number of sensors required (a single monocular camera) and unlimited operation range. Recent advance in computationally efficient mSLAM Engel et al. (2014, 2013); Forster et al. (2014); Klein and Murray (2007); Montemerlo et al. (2003); Newcombe et al. (2011b); Pizzoli et al. (2014); Roussillon et al. (2011); Vogiatzis and Hernández (2011) enables such system to effectively operate in near real-time on highly dynamic platforms, such as UAVs. while most of them focus on passively estimating the camera state for any given image frames, regardless how well the camera view point is suitable for this localization task.

However, not all the camera views are equally suitable for mSLAM localisation tasks. Unlike GPS-based navigation, the accuracy of the mSLAM-based navigation performance highly depends on the motion of the UAV and the scene within the camera's field of view. In real world scenarios, due to the lack of texture or contrast in the scene, the localisation performance of the mSLAM can be dramatically influenced. Besides, the 3D uncertainty of the mapped features mainly depends on the triangulation angle between observations, which in turn effects the localisation uncertainty.

Similar problems have been widely explored as the Next-Best-View (NBV) problem in computer vision community Haner and Heyden (2011); Vasquez-Gomez et al. (2014); Wenhardt et al. (2007), which suggests that given an approximate reconstruction it seeks a single next view that minimises the reconstruction error. While similarly, the active SLAM problem Carlone et al. (2014); Chaves et al. (2014); Kim and Eustice (2013); Kontitsis et al. (2013); Leung et al. (2006); Valencia et al. (2012) in robotics community tries to find the optimal action that can improve map building for exploration purpose. Wenhardt Wenhardt et al. (2007) compared the three general criteria to determine the NBV: (D)terminant-optimal, (E)igenvalues-optimal and (T)race-optimal based on covariance matrix, and showed no significant advantage of one criteria over another. Besides, ChavesChaves et al. (2014) and Kim Kim and Eustice (2013) proposed active visual SLAM path planning algorithms that plans coverage efficient loop-closure paths. Moreover, Carlone Carlone et al. (2014) investigated the usage of particle filter in active SLAM and exploration to allow robot to autonomously decide between exploration and place revisiting actions, and Valencia Valencia et al. (2012) proposed a computational efficient one step look ahead exploration strategy for pose SLAM.

Both topics focus on ground robotic applications, which benefit from their naturally stable vehicle dynamics and the ability to perform direct odometry such as wheel encoding. Thus, the main purpose for mSLAM in such systems is to correct the long term drift accumulated from the direct odometry. Therefore, it can operate in a very low update rate, which allows costly computation, and the tracking failure can be easily handled and recovered. On the contrary, UAV, as an extremely dynamic platform, does not allow any direct odometry without GPS. Thus the mSLAM is mostly used as its primary localisation method, and any short tracking lose could cause a crash, with little chance for recovery. Therefore, the safety, robustness and update rate of the mSLAM for UAV is the primary concern, when involved in its high speed control loop. Moreover, the long term drift for UAV is less of concern, given their short operation endurance (10 to 30 mins in general).

Therefore, to address this problem, the active SLAM planning algorithm serves as the complementary to the existing passive mSLAM algorithm. The objective is to actively command the vehicle movement to minimise the localisation uncertainty while performing the high level tasks, in a computationally efficient manner. In particular, given the next desired waypoint, an novel active planning algorithm is developed to choose the optimal UAV trajectory for an autonomous quadrotor, to maximise the accuracy of the onboard SLAM algorithm, while approaching the waypoint. More specifically, the active planner aims to command the UAV, so that it:

1. Tries to approach the next way point as close as possible;
2. Tries to minimise the localisation uncertainty of mSLAM algorithm.

To the best of my knowledge, similar robust active planner for monocular localisation on UAV has only been implemented by Mostegel et al. (2014). Based on a forward facing camera and PTAM Klein and Murray (2007) mSLAM algorithm, he measured the localization quality by the weighted sum of the number of mapped points in the frame, due to the lack of uncertainty measurement from PTAM. Furthermore, based on the measured localization quality, a threshold based planner was developed to switch among localization improvement mode, new point generation mode or way point following mode. although the system is capable of encountering the challenging target command, like pure yaw rotation, it considers neither the optimal tradeoff between the two objectives, nor the physical dynamics of the quadrotor.

Different from the above approach, its aim is to develop a probability theory driven optimal planner, with a downward facing camera, which naturally avoids the pure yaw rotation problem. Therefore, the following contributions was made in part of this PhD work to the state of the art:

1. It presents the first theoretically optimal cost function for the two objectives from the probability theory perspective;
2. It presents the first approach to predict the localisation accuracy for any camera view point, based on the extended information filter (EIF);
3. The first active planning algorithm was designed for downward facing camera, which takes the physical dynamic constraints of the quadrotor into consideration.

In the next section, the problem is formularised from the probability distribution prospective. In the following sections, it starts with formularising the problem in Section 6.2, and presents the extended information filter based localisation quality measurement in Section 6.3. Then the optimisation search procedure is describe in Section 6.4 and experimental test result in Section 6.5, and finally conclude in Section 6.6.

6.2 Problem Formulation

In highly unstructured unknown environments, multiple steps look ahead planning hardly makes any sense, thus the purpose of this active planner is to compute no more than one step look ahead trajectory waypoint action $\boldsymbol{p}_\mu^\star \in \mathbb{R}^3$ that satisfies the overall objectives the most. As shown in Fig. 6.2, the active planner receives the next desired waypoints $\boldsymbol{w}^\star \in \mathbb{R}^3$ from user or higher level planner, and command the planned next trajectory waypoint $\boldsymbol{p}_\mu^\star \in \mathbb{R}^3$, by monitoring the camera position estimation $\boldsymbol{p}_\mu \in \mathbb{R}^3$ with its covariance matrix P_p and the camera orientation R_c, and the mapped features $\boldsymbol{l}_n \in \mathbb{R}^3$, $(n = 1, 2, ...)$, with their covariance matrix R_{l_n} from mSLAM engine. As shown, the system assume a separate mSLAM engine running concurrently, and able to provide current camera pose estimation and mapped features. Note that in this chapter, we do not consider obstacle avoidance or occlusion of the mapped sparse points.

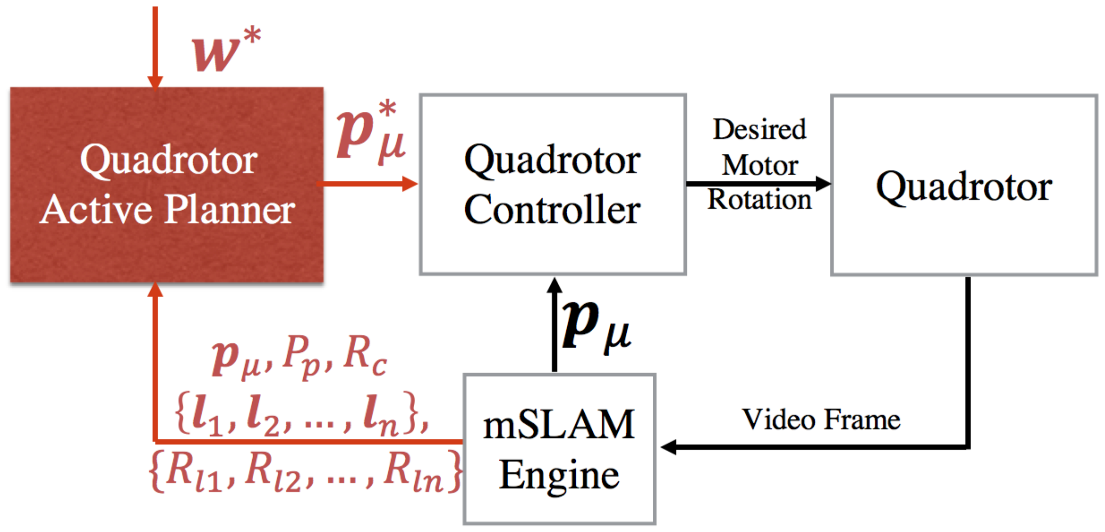


Figure 6.2: System block diagram.

We then consider this as an optimisation problem based on probability distribution. Therefore, we define our objective cost (energy) function C as the expectation of the squared difference between the UAV next actual position $\boldsymbol{p}_a^*$ and desired waypoint $\boldsymbol{w}^\star$:

$$C(\boldsymbol{p}_a^\star, \boldsymbol{w}^\star) = E[(\boldsymbol{p}_a^\star - \boldsymbol{w}^\star)^2], \tag{6.1}$$

where the superscript $'2'$ represents the inner product of itself. It can then be derived as a function of the position mean $\boldsymbol{p}_\mu^\star$ and desired waypoint $\boldsymbol{w}^\star$:

$$C(\boldsymbol{p}_\mu^\star, \boldsymbol{w}^\star) = Tr[P_p^\star] + (\boldsymbol{p}_\mu^\star - \boldsymbol{w}^\star)^2, \tag{6.2}$$

where $P_p^\star \in \mathbb{R}^{3\times3}$ is the predicted noise covariance matrix for odometry in the next trajectory waypoint, $Tr[\star]$ denotes the trace of the matrix, and $\boldsymbol{I}_3$ is the identity matrix. Therefore, the objective of the planner is to search for the optimal next trajectory waypoint $\boldsymbol{p}_\mu^\star$ within a search space $S \subset \mathbb{R}^3$, that minimise the cost function in (6.2):

$$\boldsymbol{p}_\mu^\star = \arg\min_{\boldsymbol{p}_\mu^\star \in S} C(\boldsymbol{p}_\mu^\star, \boldsymbol{w}^\star). \tag{6.3}$$

This expression theoretically defines the optimal trade-off between the future localisation uncertainty $P_p^\star$, and the distance between the next trajectory waypoint $\boldsymbol{p}_\mu$ and desired waypoint $\boldsymbol{w}^\star$.

The next section will focus on computing the localisation uncertainty (quality) $P_p^\star$.

6.3 Localisation Quality Measurement

The localisation quality of the mSLAM algorithm can be represented as the covariance matrix of the localization estimation. Thus, the purpose of the localisation quality prediction process is to predict the covariance matrix $P_p^\star$ of any given future positions $\boldsymbol{p}_\mu^\star$ within the search space S, based on the current observations from the concurrently running mSLAM algorithm. The current observations include the current camera position estimation $\boldsymbol{p}_\mu$ with its covariance P_p, the estimated 3D location of the mapped features points $\boldsymbol{l}_n$ in the map, with their covariance matrices R_{l_n}.

Then the localisation quality measurement model can be derived in the similar fashion with the measurement step of the extended information filter (EIF) Yaakov et al. (2013) based on fisher information matrix Frieden and Binder (2000).

6.3.1 Expected orientation and active points

For a given future position $\boldsymbol{p}_\mu^\star$, the planner assumes the UAV approaches the future position in a fixed acceleration $\boldsymbol{a}^\star$, which can be computed by:

$$\boldsymbol{a}^\star = \frac{2(\boldsymbol{p}_\mu^\star - \boldsymbol{p}_\mu) - 2\dot{\boldsymbol{p}}_\mu \delta t}{\delta t^2}, \tag{6.4}$$

where δt is the time interval for UAV to reach the future position, and $\dot{\boldsymbol{p}}_\mu$ is the current velocity measurement. Then given the current user yaw command $\boldsymbol{x}_{yaw}$ as a unit vector on X-Y plane in world frame, the expected UAV orientation matrix R_c can be computed as:

$$\boldsymbol{z}_R = \frac{\boldsymbol{a}^\star - \boldsymbol{g}}{||\boldsymbol{a}^\star - \boldsymbol{g}||}, \tag{6.5}$$

$$\boldsymbol{y}_R = \frac{\boldsymbol{z}_R \times \boldsymbol{x}_{yaw}}{||\boldsymbol{z}_R \times \boldsymbol{x}_{yaw}||}, \tag{6.6}$$

$$\boldsymbol{x}_R = \boldsymbol{y}_R \times \boldsymbol{z}_R, \tag{6.7}$$

$$\mathsf{R}_c = [\boldsymbol{x}_R \quad \boldsymbol{y}_R \quad \boldsymbol{z}_R], \tag{6.8}$$

where $\boldsymbol{x}_R$, $\boldsymbol{y}_R$ and $\boldsymbol{z}_R$ are the three unit axis vector of the UAV frame, and $\boldsymbol{g} = [0, 0, -9.8]^\top$ is the gravity vector.

All the mapped points can be projected onto the image plane based on the camera projection model $\boldsymbol{\pi} : \mathbb{R}^3 \rightarrow \mathbb{R}^2$, which is determined by the intrinsic camera parameters which is known from camera calibration. Thus, the 2D position of n^{th} projected feature $\boldsymbol{u}_n \in \mathbb{R}^2$, can be expressed as:

$$\boldsymbol{u}_n = \boldsymbol{\pi}(\boldsymbol{l}_n^c), \tag{6.9}$$

where the superscript c in $\boldsymbol{l}_n^c$ denotes that the 3D point coordinates are expressed in camera frame, and it can be obtained by applying the rigid body transformation on the mapped points:

$$\boldsymbol{l}_n^c = \mathsf{R}_c^\top(\boldsymbol{l}_n - \boldsymbol{p}_\mu^\star). \tag{6.10}$$

Note here the UAV frame and camera frame is assumed coincide. Therefore, all the projected points which lie inside the dimension of the image are named active points, which will be used to measure the localisation quality.

6.3.2 Fisher information modelling for one active point

Given that the keyframe based visual mSLAM provides independent measurement on consecutive frames over time, thus the uncertainty propagation over time is ignored.

The measurement model formularises the geometric relationship between the feature points and the mSLAM localisation process. Here we take the k^{th} active point as an example. Then based on (6.9) and (6.10) the camera projection of the active point can be modelled as:

$$\boldsymbol{u}_k = \boldsymbol{\pi}(\mathsf{R}_c^\top(\boldsymbol{l}_k - \boldsymbol{p} + \boldsymbol{e}_{_{lk}})) + \boldsymbol{e}_{_{ck}}, \tag{6.11}$$

where $\boldsymbol{u}_k \in \mathbb{R}^2$ is the corresponding feature position in image coodinate. $\boldsymbol{l}_k \in \mathbb{R}^3$ is the position of the active point in the 3D sparse map; $\boldsymbol{e}_{_{lk}} \in \mathbb{R}^3$ is the mapping error of the active point, and $\boldsymbol{e}_{_{ck}} \in \mathbb{R}^2$ is the tracking error of the active point in camera frame. Both of them are modelled as zero-mean Gaussian white noise:

$$p(\boldsymbol{e}_{_{lk}}) \sim N(0, R_{_{lk}}), \tag{6.12}$$

$$p(\boldsymbol{e}_{_{ck}}) \sim N(0, R_{_{ck}}), \tag{6.13}$$

where the point covariance matrix $R_{_{lk}}$ and the tracking covariance matrix $R_{_{ck}}$ are assumed to be diagnose and provided by the mSLAM algorithm.

Then, the information gained from the k^{th} active points can be represented as its information matrix, I_k, which can be propagated as:

$$I_k = H^\top(VR_{_{lk}}V^\top)^{-1}H, \tag{6.14}$$

where $H \in \mathbb{R}^{2\times3}$ is the Jacobian matrix of the measurement model (6.11) with respect to $\boldsymbol{p}_\mu^\star$; and $V \in \mathbb{R}^{2\times3}$ is the Jacobian matrix of the measurement model (6.11) with respect to $\boldsymbol{e}_{_{lk}}$.

6.3.3 Probability of point recognition

In real operation, not all the active points can be successfully recognised, due to the change of viewing angle, viewing distance and potential occlusion. In our model, only the viewing angle is considered.

Based on the result in Mostegel et al. (2014), by assuming that each feature point is a flat planar patch, the recognition accuracy depends on the viewing angle towards the patch. Thus the point recognition probability for k^{th} active point p_k can be reasonably approximated by:

$$p_k = \begin{cases} cos\alpha_k & \text{if } \alpha \in [-\pi/2, \pi/2] \\ 0 & \text{otherwise} \end{cases},$$

where α_k is the viewing angle from the norm of the planar patch.

6.3.4 Likelihood of keeping track

Besides the tracking accuracy, the reliable continuous tracking of the visual mSLAM system is directly related to the number of features being tracked in the current frame, and in the real implementation, this number of tracking features usually limited by a hard threshold. In other words, any measurements with the number of tracking features less than the threshold will be treated as insufficient measurement, and the closer the number to the threshold, the less likely that the mSLAM will keep the continuous tracking. This cut off behaviour will result in zero information gain from the current frame, when triggered.

In order to take this behaviour into account, the likelihood of keeping track, p_t, linearly scales the information matrix, and it is assumed that the p_t is in second order relationship to the number of active points m in the scene:

$$p_t = \begin{cases} (\frac{m-m_{min}}{m_{max}-m_{min}})^2 & \text{if } m \in [m_{min}, m_{max}] \\ 1 & \text{if } m \in [m_{max}, +\infty) \\ 0 & \text{otherwise} \end{cases},$$

where m_{max} and m_{min} are the maximum and minimum allowed number of tracking features in the frame, which are defined in the mSLAM algorithm.

6.3.5 Computing the localisation quality for one future position

Given any future position, $\boldsymbol{p}_\mu^\star$, the localisation information is gained from all the active points projected to that future position. Since the probability of point recognition is also considered, given m active points, the information gained from all the active points is then represented as the total information matrix, I, which can simply be computed by the weighted sum of the information matrix of the individual active point Thrun (2004):

$$I = p_t \sum_{k=1}^{m} p_k I_k. \tag{6.15}$$

Then the localisation quality (covariance matrix $P_p^\star$) can be computed as:

$$P_p^\star = I^{-1}. \tag{6.16}$$

6.4 Optimal Trajectory Action Point Search

Given the ability to predict the localisation quality $P_p^\star$ for one future position $\boldsymbol{p}_\mu^\star$, as described in the last section, the planner is able to compute the cost energy for this position based on (6.2). While in order to select the optimal next trajectory point in the search area S, to minimise the cost energy, as described in (6.3), the following action search process will be conducted. The action search executes in a fixed time interval δt. Specifically, the Monte-Carlo tree search is performed within a 3D space constrained by the UAV physical dynamics.

6.4.1 UAV dynamics constraints

Considering the physical limitation of the UAV, in order to ensure the selected next trajectory point is feasible for UAV to smoothly reach within the time interval δt, we define our search space S, so that the resulting trajectory point commands the UAV with less than maximum acceleration a_{max} and less than maximum velocity v_{max}. Thus given the current position measurement $\boldsymbol{p}_\mu$ and the current velocity measurement $\dot{\boldsymbol{p}}_\mu$, the search space S can be expressed as the intersection between the two ellipsoids:

$$S_a = \{\boldsymbol{p} \in \mathbb{R}^3 | ||\boldsymbol{p} - \boldsymbol{p}_\mu - \dot{\boldsymbol{p}}_\mu \delta t|| < \frac{a_{max}\delta t^2}{2}\}, \tag{6.17}$$

$$S_v = \{\boldsymbol{p} \in \mathbb{R}^3 | ||\boldsymbol{p} - \boldsymbol{p}_\mu|| < v_{max}\delta t\}, \tag{6.18}$$

$$S = S_a \cap S_v. \tag{6.19}$$

Here, this constrained space is further simplified as a cube by defining the maximum edge point $\boldsymbol{p}_{max}$ and minimum edge point $\boldsymbol{p}_{min}$ as:

$$\boldsymbol{p}_{max} = min\{\boldsymbol{p}_\mu + \dot{\boldsymbol{p}}_\mu \delta t + \frac{a_{max}\delta t^2}{2}\mathbf{1}, \boldsymbol{p}_\mu + v_{max}\delta t\}, \tag{6.20}$$

$$\boldsymbol{p}_{min} = max\{\boldsymbol{p}_\mu + \dot{\boldsymbol{p}}_\mu \delta t - \frac{a_{max}\delta t^2}{2}\mathbf{1}, \boldsymbol{p}_\mu - v_{max}\delta t\}, \tag{6.21}$$

6.4.2 Monte-Carlo tree search

The Monte-Carlo tree search technique has been widely explored in the field of machine learning, to rapidly solve the optimisation problems without the knowledge of the hidden underlying system model, widely called multi-armed bandit problem. Bubeck et al. (2008); den Broeck and Guy Driessens (2011); Weinstein and Littman (2012)

The optimal next action within searchable space as the constrained cube, which is defined by the two edge points $\boldsymbol{p}_{max}$ and $\boldsymbol{p}_{min}$, can be discovered by executing the spatial Monte-Carlo tree search algorithm. The algorithm generates an hierarchical tree of nodes by expanding from the root, which represents the entire searchable space. The tree is formed of multiple connected nodes. Starting from the root node as the top hierarchy, the nodes in lower level present the subspace of the nodes in higher level, which is called child and parent nodes. Besides, the action, which is a sampling position generated from the corresponding node, is used to input to the simulator in order to generate a reward, which in this case is generated by the cost energy function defined. The leaf is defined as the node with no child or has never be visited. This algorithm is described in Algorithm 1.

Algorithm 1 Spatial Monte-Carlo tree search

```
1:  procedure MCTS
2:  loop:
3:      $n_{best} \leftarrow root.$
4:      while $n_{best} \neq leaf$ do
5:          $n_{best} \leftarrow argmax_{c \in CHILDREN(n_{best})} B(c).$
6:      $action \leftarrow ACTION(n_{best}).$
7:      $reward \leftarrow REWARD(action).$
8:      $n_{associated} \leftarrow n_{best}.$
9:      while $n_{associated} \neq root$ do
10:         $UPDATE\_STATISTICS(n_{associated}).$
11:         $n_{associated} \leftarrow PARENT(n_{associated}).$
12:     $UPDATE\_STATISTICS(n_{associated}).$
13:     $CHILDREN(n_{best}) \leftarrow \{C_1(n_{best}), C_2(n_{best})\}.$
14:     goto loop.
```

The line 3 to the line 5 show that from the root, it finds the best leaf node as the highest B-value (ties are broken arbitrarily). In the i^{th} iteration, to minimise the regret rate, the B-value of the node n can be computed by applying the upper confidence bound policy, which is defined as:

$$U(n) = \bar{R}(n) + \sqrt{\frac{2ln(i)}{v}} + v_1 \rho^h, \tag{6.22}$$

$$B(n) = min\{U(n), max\{B(C_1(n)), B(C_2(n)\}\}, \tag{6.23}$$

where $\bar{R}(n)$ is the average reward per visit and v is the number of visit for the corresponding node. $v_1 > 0, 0 < \rho < 1$ are preset parameters, and h is the hierarchical level of the node. $C_1(n)$ and $C_2(n)$ are the two children of the node.

Line 6 generates the the action from the best node selected, by selecting a random position $\boldsymbol{p}_{\mu}^{\star}$, and line 7 shows that this action (position $\boldsymbol{p}_{\mu}^{\star}$) sent to the simulator by calling the $REWARD(action)$ function. Here, the simulator, is the cost energy function defined in (6.3).

Then after the reward function, line 8 to line 11 propagate the reward back to all the higher level nodes associated (or nodes containing the chosen action). This includes updating their total rewards and the number of visits.

Finally, the best selected node is expanded by line 13. This is achieved by creating two child nodes, following the space splitting policy. The policy defines that the parent node splits into half along one axis, while this splitting axis is chosen randomly from the three in each iteration.

6.5 Implementation

This active SLAM algorithm performs the one-step-look-ahead planning for the UAV to make action decision, which offers the optimal balance between approaching the target position and avoiding the featureless scene.

The algorithm was implemented on the onboard linux computer (Odroid U3 from HardKernel) on a real UAV platform, which was built in Chapter 5, with a downward-facing monocular camera. All the computations, including a monocular SLAM algorithm, Extended Kalman Filter for visual-inertial fusion and active planning algorithm, were executed simultaneously utilising all four CPU cores of the onboard Odroid U3, with Robotic Operating System (ROS) as background interface. The localisation quality measurement algorithm was based on a third party SLAM algorithm called Semi-direct Visual Odometry (SVO), which also offers ROS interface.

The first test aimed to show the effectiveness of the localisation quality measurement algorithm described in Section 6.3. The localisation quality was expressed by the trace of the position covariance matrix obtained by (6.16). This was tested by computing the quality value only for the current UAV position, in real time. The generated map points and the camera state from SVO were passed into the localisation quality measurement to compute the active points, then the localisation quality measurement algorithm was executed to obtain the quality measurement.

The test was conducted by hand-holding the quadrotor to move above a feature-rich square carpet, which is laid on top of the pure grey ground, as shown in Figure 6.3. With this setup, the onboard downward-facing camera can only track the features on the carpet, thus when moving towards the edge of the carpet, the localisation quality should be measured with large uncertainty. The localisation uncertainty was graphically expressed as the thickness of the UAV trajectory (thicker means larger uncertainty), which is merged with the 3D feature map generated from SVO. Note that since the uncertainty is very small when SVO is in good tracking, in order to show the readable effect in the map, the quality value was scaled 300 times to directly define the thickness of the trajectory.

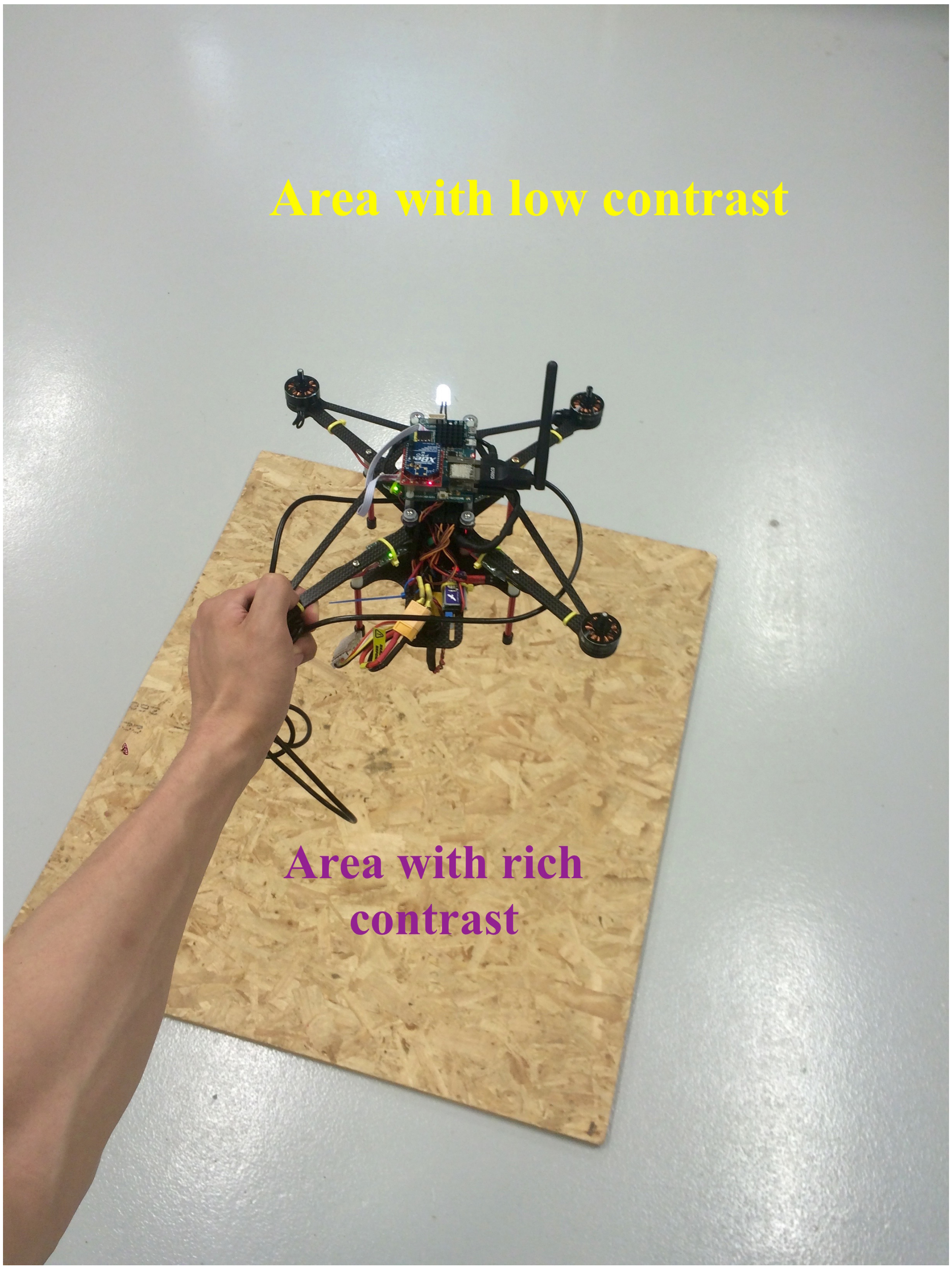


Figure 6.3: Active SLAM testing environment.

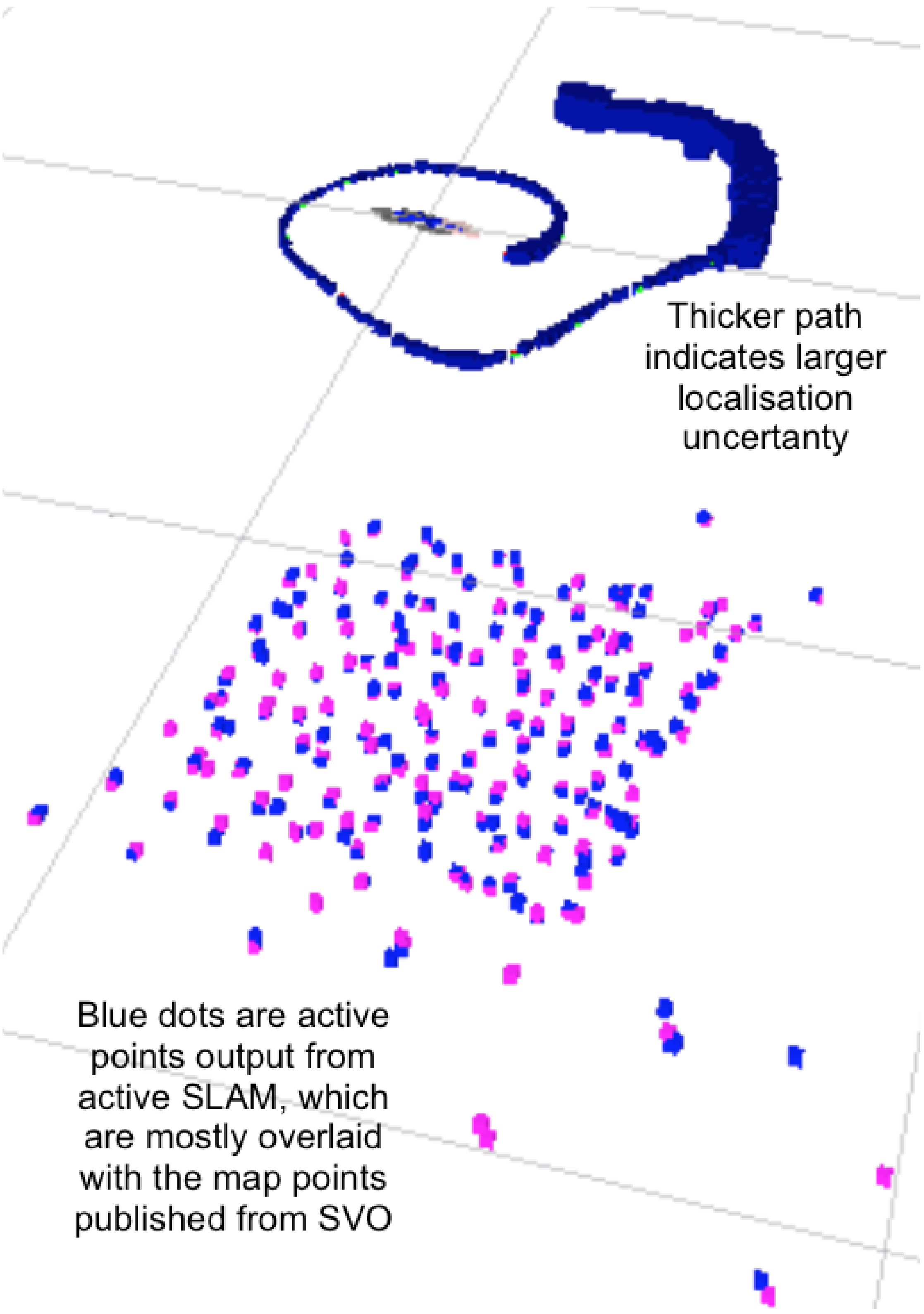

Thicker path indicates larger localisation uncertanty
Blue dots are active points output from active SLAM, which are mostly overlaid with the map points published from SVO

Figure 6.4: Active SLAM testing horizontal movement (side view).

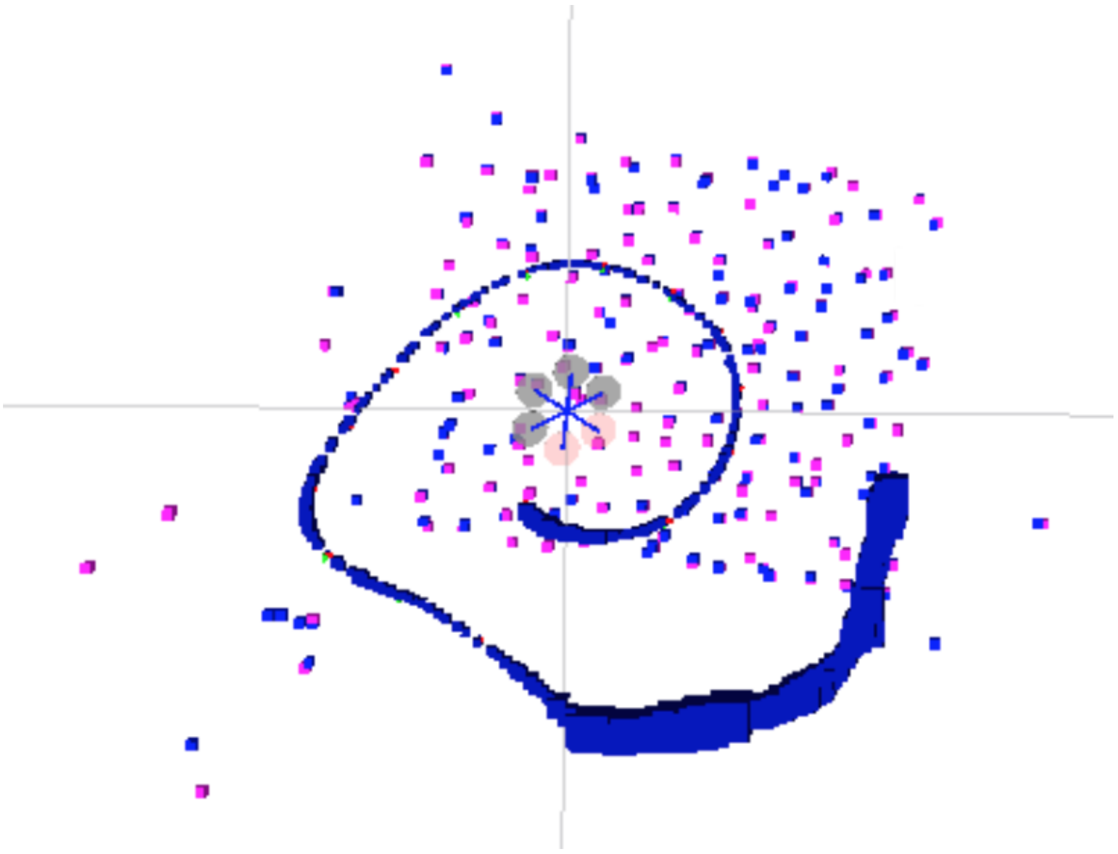


Figure 6.5: Active SLAM testing horizontal movement (top view).

Figure 6.4 and 6.5 show the side view and the top view of a horizontal spiral trajectory towards the edge of the carpet. The pink points are the features mapped by the SVO, while the blue points are the active points estimated by the quality estimation algorithm. Note that most blue points are overlaid on top of the pink points with negligible error, due to the unsynchronised projection. It is shown clearly that the localisation uncertainty increases significantly at the edge of the carpet.

Then, a vertical helix movement test was performed to show how the scene distance effect the quality estimation. As shown in Figure 6.6, the localisation uncertainty increases when UAV moves far away from the scene. This is because when features are further away, the translational motion of the camera causes less motion of the feature in the image frame. Besides, since most of the tracking features are limited inside the carpet, when the carpet is far away, the carpet occupies a small part of the image frame, which significantly increases the localisation uncertainty along the vertical axis.

Finally, a complex motion was performed to show the robustness of the estimation. As shown in Figure 6.7, the localisation quality of the system was estimated in the same manner regardless of the complex motion.

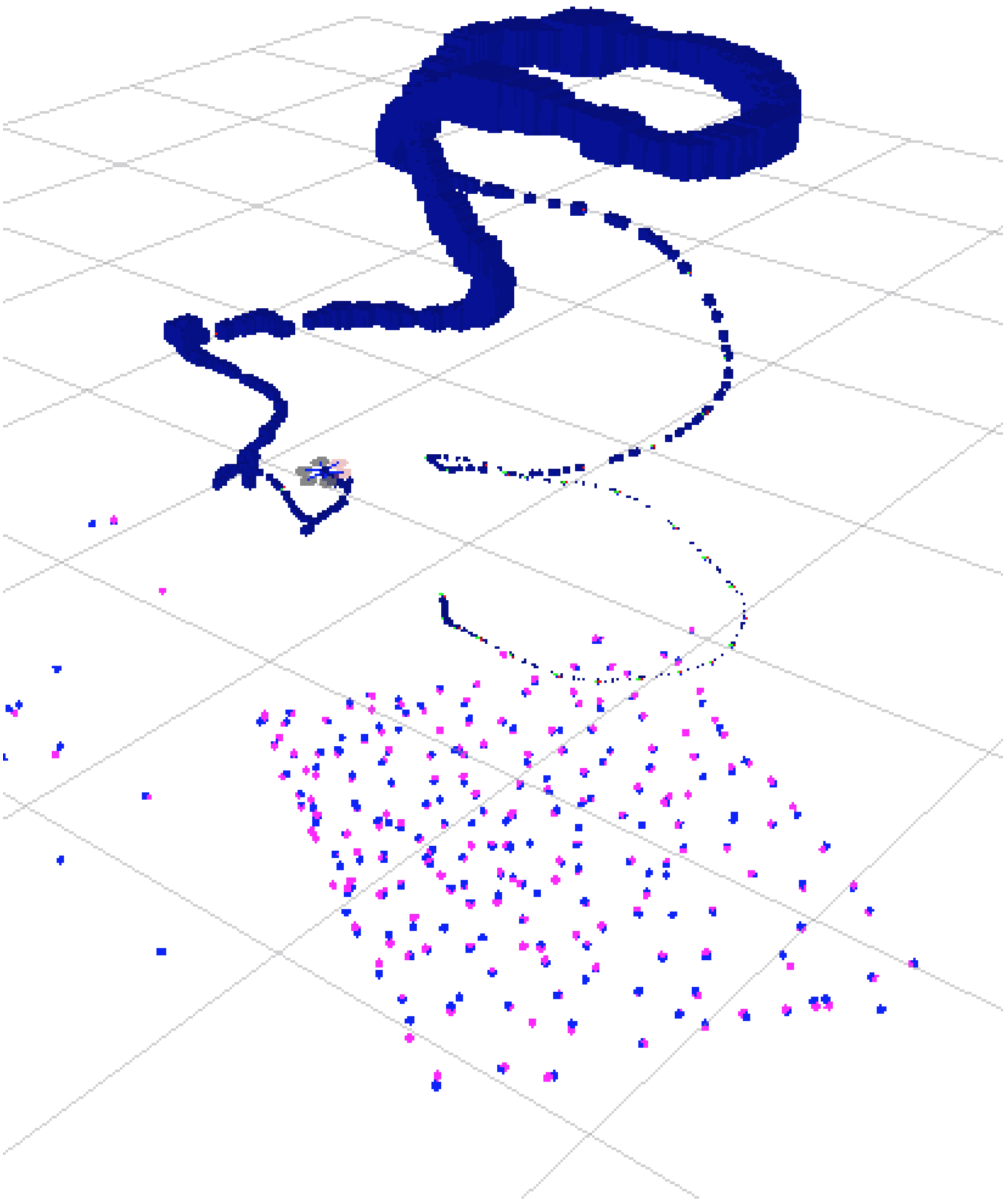


Figure 6.6: Active SLAM testing vertical movement.

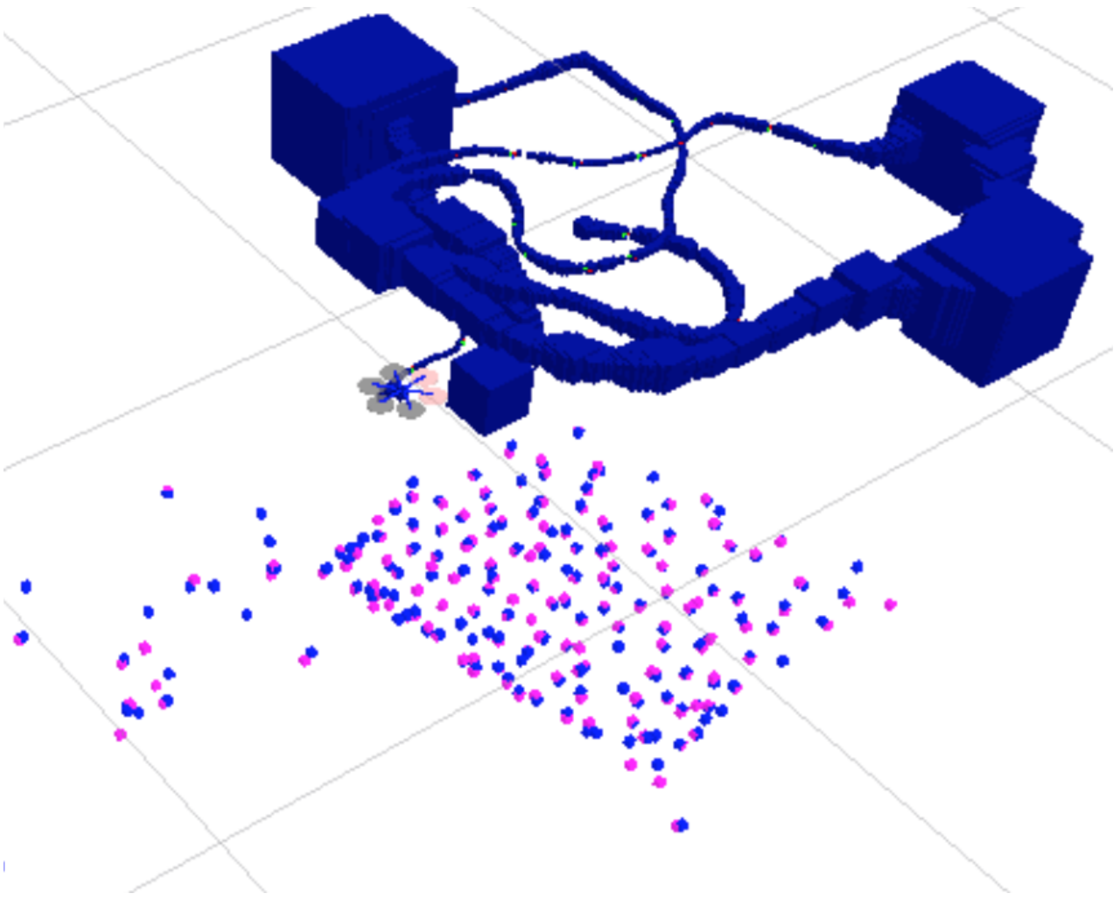


Figure 6.7: Active SLAM testing complex movement.

6.6 Conclusion

Instead of passively performing mSLAM while directly traveling to the destination (target waypoint), we integrated the perception requirements of the mSLAM into the navigation control of the UAV. Specifically, we present a novel active one-step-ahead planning algorithm which dynamically generates the optimal next action command in real-time for an autonomous quadrotor based on its current mSLAM observations, so that it not only approaches the target waypoint, but also maximising the localisation accuracy for the onboard mSLAM algorithm. This active planning algorithm serves as the complementary to the state-of-the-art passive mSLAM algorithm, and aiming to improve the operational safety and robustness of the real world GPS-denied scenarios. We have incorporated the active planning algorithm with a third party mSLAM algorithm, and implemented the localisation uncertainty estimation algorithm in real UAV system. We have shown that the algorithm is able to reasonably estimating the localisation uncertainty from the given pose. However, further work needs to be done to implement the Monte-Carlo tree search, which employs the estimation algorithm.

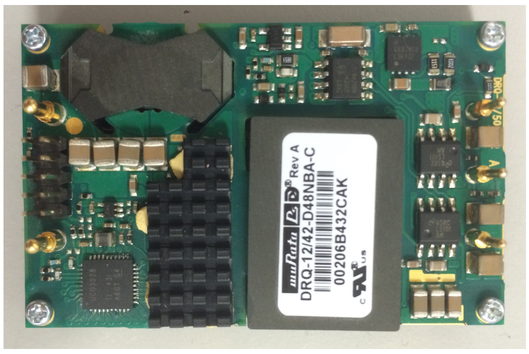
Chapter 7

Tether Power Solution

For a micro size platform, the average endurance is between $10 \sim 20\,min$, which is significantly lower than larger size platforms. The improvements in efficiency and battery technology can only have marginal effects on the endurance. In order to break the barrier of flight endurance for the small size platform, a tethering solution is proposed to provide power from offboard ground station. Due to the complete separation between the airborne and its main power source, this implementation will dramatically boost the flight endurance to indefinite. However, the challenge for such tether power system for the micro size platform is the trade-off between the mass and power capacity of the system, so that to allow substantial power draw for UAV manoeuvre, while minimising system size, which minimises the impact of such system on the payload capacity of the platform. This chapter summarises the design and engineering of the tether power system for the long endurance flight.

7.1 Introduction

Unmanned Aerial Vehicles (UAVs) have become one of the main field of robotics research in recent years, with a number of significant breakthroughs being achieved in order to tackle the challenges associated with their airborne nature. More specifically, the field of small-scale UAV design, the field of high precision model-based control and autonomous navigation, the field of robotic manipulation, and the field of robotic visual perception, along side with the development of small form-factor and high speed embedded systems. This momentum has sparked a keen interest in employing small-and-medium size UAVs in a wide range of civilian applications, without being limited to environment mapping and precision agriculture, industrial infrastructure inspection, and disaster response missions. For the small-scale stable hovering platforms (such as standard multirotors) the available sources of power are limited to the on-board electric storage cells (batteries), in which optimising the battery and design only gives minor improvement on the endurance.



(a) Murata 500 W voltage regulator on airborne platform.

(b) 500 W, 48 V AC-DC converter on ground station.

Figure 7.1: Voltage converters for ground and air.

However, there exists a range of possible crucial applications, where long term operation of such systems is required, such as precise inspection of large industrial structures, and tasks involving physical manipulation of the environments. Those applications can be achieved by UAVs, at the cost of significant power requirements. Also, there exist tasks which require close collaboration of aerial and ground robots, but inherently take a significant amount of time to accomplish, such as the aerial detection of land mines using radar-bullets and their safe removal using robotic agents. Within those tasks agile flight and aggressive manoeuvre abilities might not be the key prerequisite, rather than stable operation while hovering at high altitudes.

There is little research currently on the tethered UAV, recent tethered UAV research mainly focus on the control aspect of such system Abdelkrim et al. (2008); Zikou et al. (2015). There are also some commercial companies developing system for large scale tethered UAVs, such as Elistair [1], PARC from Cyphy work [2], Cardinal security [3], ECA group [4], and a small size drone Fotokite [5]. Those solutions mainly focus on large or expensive powered tether UAVs.

This work focuses on implementing a cost effective method for remotely powering of small-scale hovering UAVs via a ground-to-air Power-over-Tether link which carries power to the vehicle. In the rest of this chapter, the power method selection process is shown in Section 7.2, then tether selection in Section 7.3, furthermore, the transient current and backup battery are investigated in Section7.4. Finally, conclude in Section 7.5.

7.2 Power Method Selection

While one cannot deny the fact that the physical tether will limit the flight operational range, there are many applications, such as continuous monitoring, inspection, which does not require

[1] `http://elistair.com`
[2] `http://cyphyworks.com/parc/`
[3] `http://cardinalsecurity.co.uk/tethered-drone-systems/`
[4] `http://www.ecagroup.com/en/aerospace/airborne-monitoring-survey`
[5] `http://fotokite.com`

long range of operation, but long endurance are highly desirable. Given that the range is the biggest factor, the target tether length must be decided before hand. Here 10 m was selected considering the payload capacity of the tiny platform.

Moreover, the power supply method was decided to be direct current (DC), since only 2 wires are required in this case, and market available DC power switching regulator is much smaller than AC transformer, since the switching regulator does not require any inductive coil. In order for the maximum efficiency, and wider margin for tether resistance selection, the voltage supplied into the tether should be as high as possible. However, due to the limitation of current available off the shelf DC power switching regulator, 48 V was selected. Here the Murata DRQ-12/42-D48NBA-C 500 W voltage regulator (mass: 72.5 g), as shown in Figure 7.1a, is selected for the airborne to convert the 48 V tether output voltage into 12 V for the direct three cell battery replacement for aircraft operations. On the ground side of the tether, a 500 W AC to DC power supply was selected convert from 230 VAC main socket to 48 V stable voltage into the tether, as shown in Figure 7.1b.

Table 7.1: General copper wire.

Standard Wire Gauge	**Cross-sectional area** (mm^2)	**Ω/20m**	**g/20m**
10	8.30	0.0416	1482
11	6.82	0.0506	1216
12	5.48	0.063	978
13	4.29	0.0804	766
14	3.24	0.1064	578
15	2.63	0.1312	468
16	2.08	0.1662	370
17	1.59	0.218	284
18	1.17	0.296	208
19	0.811	0.426	145
20	0.657	0.526	117
21	0.519	0.664	93.6
22	0.397	0.868	70.8
23	0.292	1.182	52.0
24	0.245	1.406	43.8
25	0.203	1.702	36.2
26	0.164	2.1	29.2
27	0.136	2.54	24.4
28	0.111	3.1	19.8
29	0.094	3.68	16.7
30	0.078	4.42	13.9
31	0.068	5.06	12.2
32	0.059	5.84	10.5
33	0.051	6.8	9.04
34	0.043	8.04	7.66
35	0.036	9.64	6.38

7.3 Tether System Selection

Given the 10 m target tether length, and about 100 W power consumption to hover the craft and 500 W maximum, the engineering challenge is to select right thickness of the coper wire, subject to the constraints that the total mass of the copper wire should be lower than 100 g, and the maximum output power should be more than 500 W, which is determined by the wire resistance.

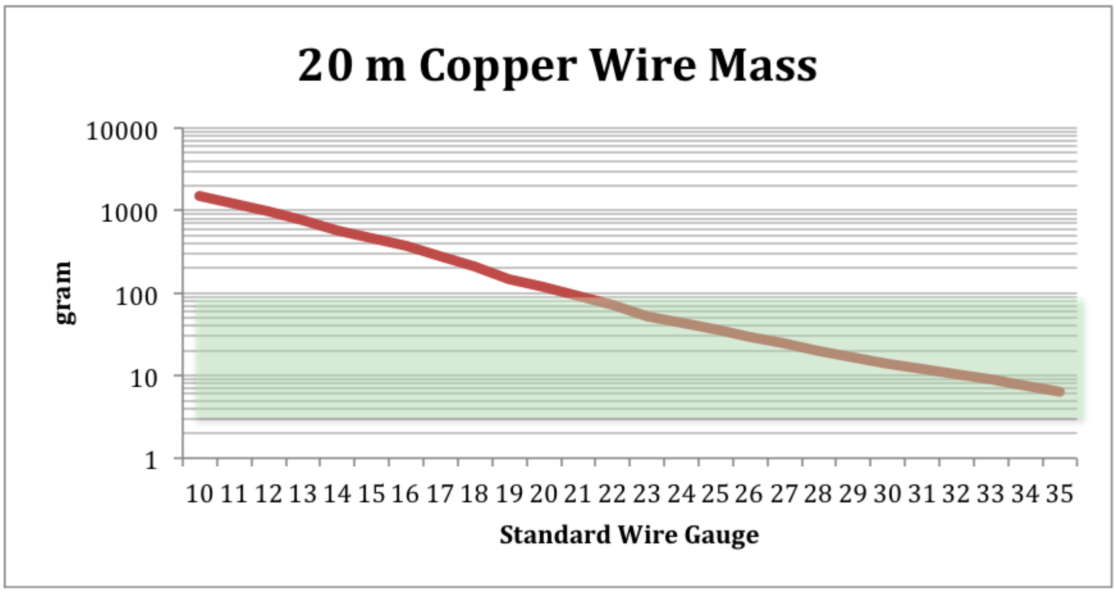


Figure 7.2: 20 m copper wire mass. (as a 10 m twin wire tether)

Given the conversion chart from standard wire gauge (SWG) to copper wire cross sectional area (A_c), as shown in Table 7.1, the total mass (M_c) of the 20 metre copper wire (l_c, as a 10 metre twin-wire tether) can be computed as:

$$M_c = \rho_c A_c l_c, \tag{7.1}$$

where ρ_c is the copper density (8.96 g/cm^3). Note that this only indicates the mass of the pure copper. However, depending on the insulation, the tether mass can vary significantly. Thus in the design $100\ g$ copper wire target was set to allow additional payload for insulation. The copper mass is plotted in Figure 7.2, where green zone is highlighted showing the satisfaction of the mass requirement.

On the contrary, the resistance increases when wire gets thinner, which then limits the maximum power output from tether. From Biot and Lowrie (2007), the resistance of the copper wire (R_c) can be computed as:

$$R_c = \frac{\rho_\Omega l_c}{A_c}, \tag{7.2}$$

where ρ_Ω is the resistivity of copper ($1.68 \times 10^{-8}\ \Omega m$). Then the possible maximum power output from tether (P_o) at 48 V input voltage (V_i) can be computed from Ohm's law as:

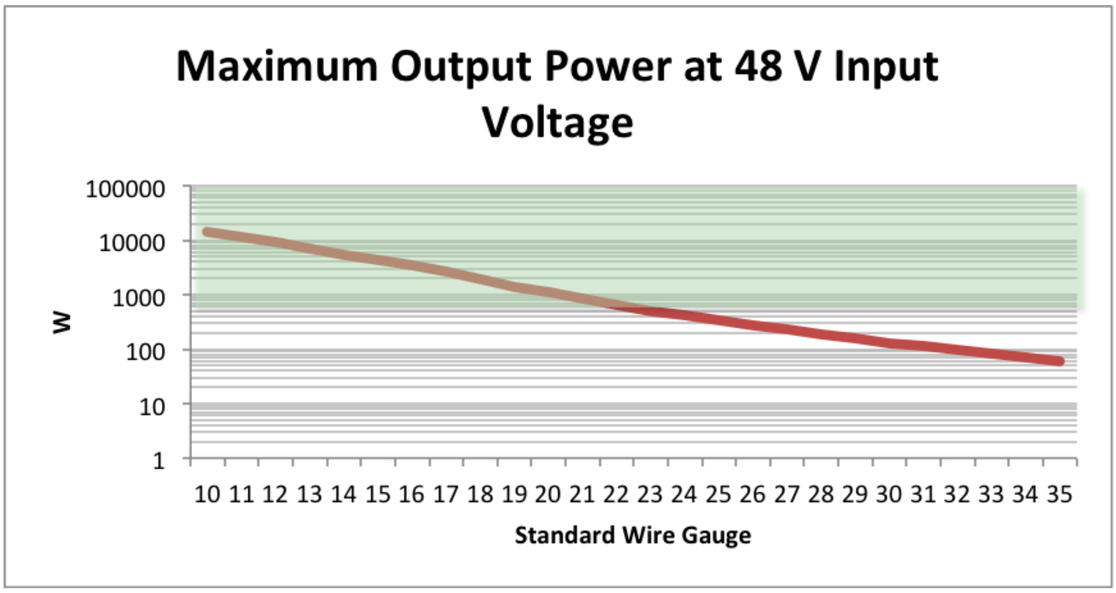


Figure 7.3: 20 m copper wire output power with 48 V input voltage. (as a 10 m twin wire tether)

$$P_o = \frac{V_i^2}{4R_c}. \tag{7.3}$$

This maximum output power from tether is then plotted in Figure 7.3, where the green zone highlights the wires that satisfies $500\ W$ output requirements. It can be easily seen that the two satisfaction zones overlaps at 22-23 SWG wire, thus 22 SWG tether was selected.

7.4 Transient Current and Backup Battery

Theoretically, with the above setup, the system would work. However, in practice the transient current for sudden motor acceleration results in spikes up to $90\ A$, which causes the power draw reaches about $1\ KW$, as shown in Figure 7.4, when testing with direct battery power. This generally happens when quadrotor takes off with sudden throttle increase. When using the converter for a direct replacement of the onboard battery, this power spike can easily trigger the over current protection of the voltage converters. The converters reset themselves results in sudden power loss.

Therefore, an onboard small 350 mAh 3 cell backup battery is introduced to provide the extra power demanded at the transient time, in order to smooth down the spike, and with this backup battery, even if the current protection was accidentally triggered, the backup battery is able to bridge out the converter reset time. Besides, in any cases, at the event of converter failure, this backup battery is capable to provide 1 minute flight time for safely landing the quadrotor.

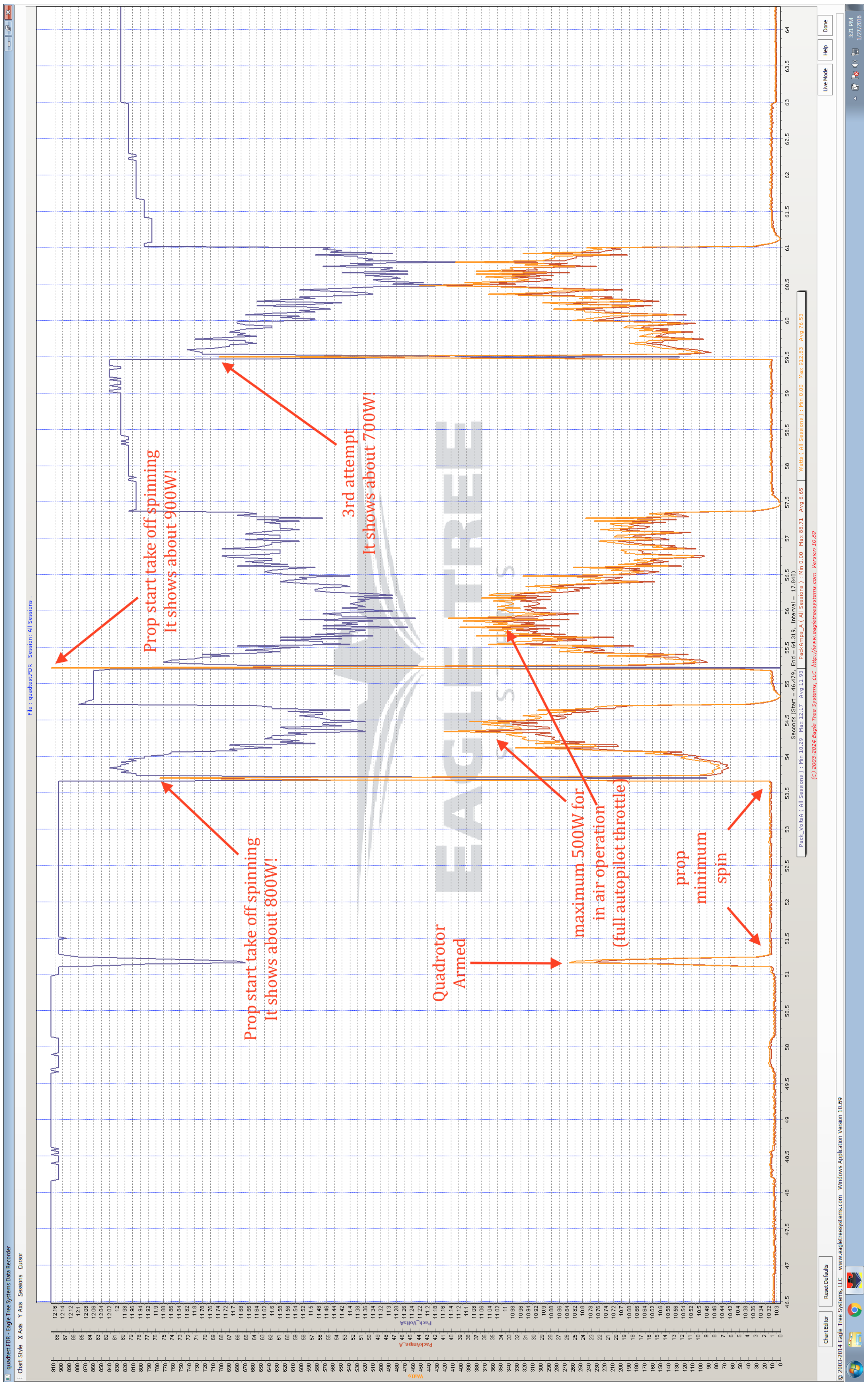


Figure 7.4: Transient power output from battery test log at 100 Hz. (blue is voltage, red is current and yellow is power)

Therefore, to interface the onboard converter with the backup battery, an interface PCB board was designed in Eagle PCB design software as Figure 7.5 and Figure 7.6a, and manufactured as Figure 7.6b.

An Schottky power diode is used at the output of the onboard converter to stop the current flows from backup battery into the converter. A small resistance was added to the output of the onboard converter to adjust the tradeoff between the backup battery and the converter, and determines the equilibrium voltage at hover. Here a very small 22 $m\Omega$ power resister is used. Besides, a Transient Voltage Suppressor (TVS) diode is used at the input of the converter to bypass the input voltage spike from tether. This generally happens when the converter was reset by its over current protection, sudden current drop in tether creates the voltage spike due to the relatively large inductance in the tether.

A further test was conducted to confirm the current smoothing performance. The dynamic voltage, current and power were logged directly from the output of onboard converter. Although, due to the internal resistance of the Li-Po battery, the transient current spike still existed, its magnitude is significantly reduced to 35 A (400 W) maximum, which is small enough to avoid triggering the converter reset.

The resulting tether system is then installed onto a quadrotor as shown in Figure 7.8. The tether attachment point locates at the centre-of-mass of the aerial platform, and the tension force is offloaded from the tether electrical connection.

The complete tether system in operation is shown in Figure 7.9. Note that the system can be powered from main socket, or a large 12 cell battery, when considering a mobile ground station.

7.5 Conclusion

In this chapter, the engineering detail of a Power-over-Tether system was presented, which was used to sufficiently power a small size quadrotor from a ground source. The corresponding low price tether powering system utilises a 10 metre standard twin 22 SWG copper cable, running 48 VDC through the tether. The air and ground voltage converters/transformers were out sourced from commercial-off-the-shelf components, while an additional backup battery interface for the airborne was developed in-house, for the purpose of reducing the transient current in the tether and backup power in case of tether failure.

The real flight test was conducted with a manually controlled quadrotor, while the electrical performance of the tether system was demonstrated through the recorded data. The future work includes designing the automatic tether management system on the ground, and integrating the GPS-denied navigation system with the tether. Besides, in addition to the power transmission in the tether, the Power-line communication (PLC) technology can be implemented into the system to allow ethernet connection on the tether. It not only offloads the power source to the ground, but also offloads computation to the ground, without additional cable.

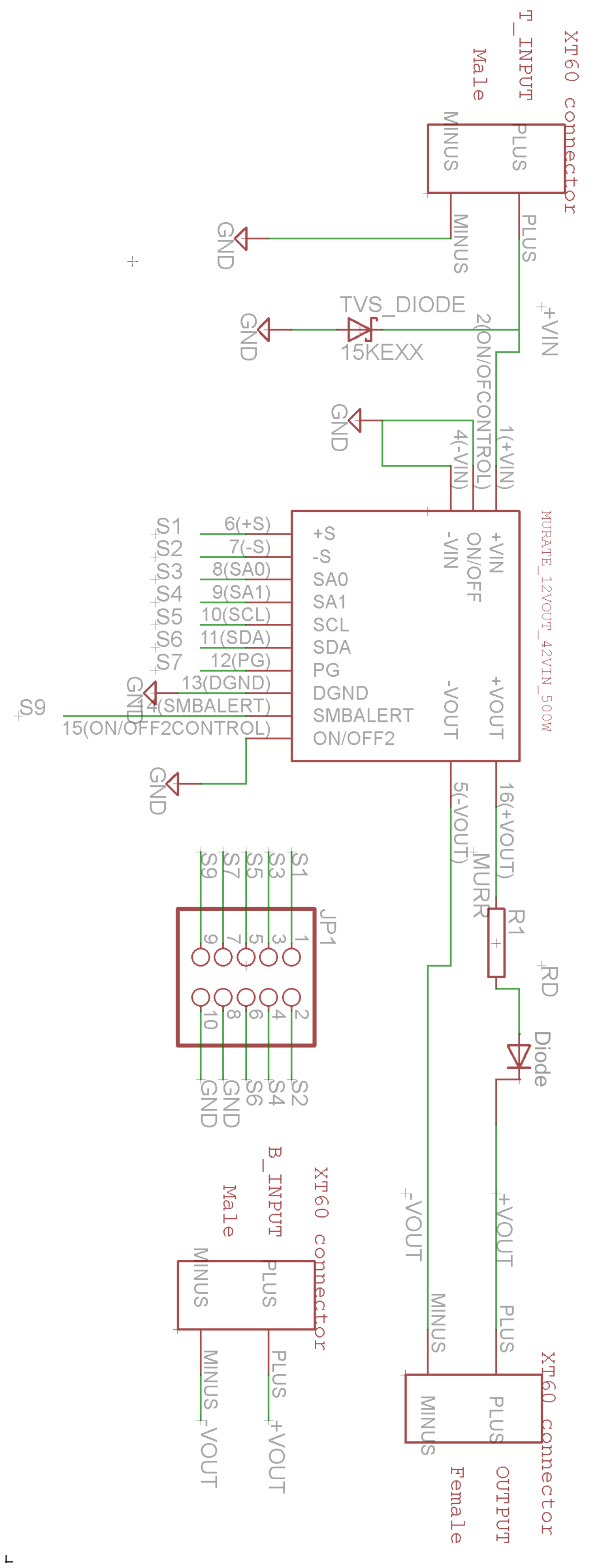
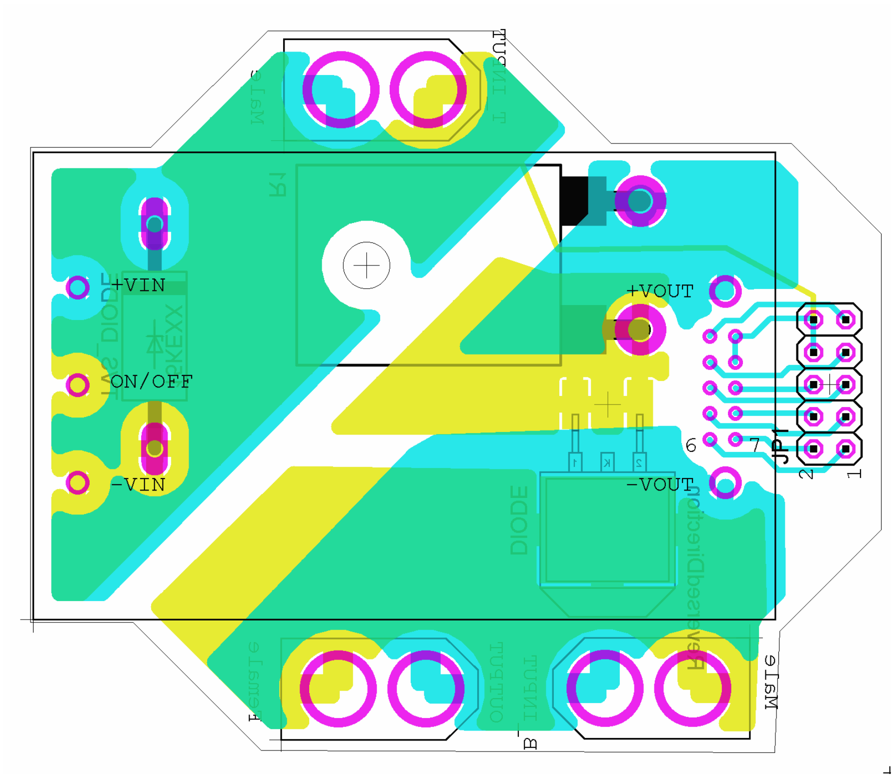
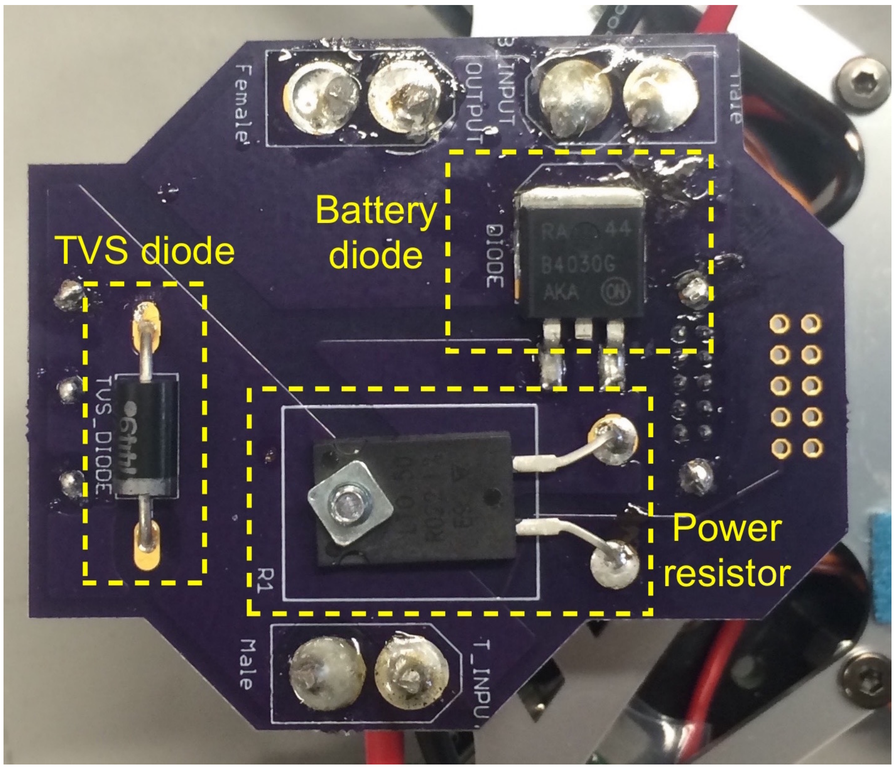


Figure 7.5: Backup battery interface board schematic.



(a) PCB fabrication diagram.



(b) Assembled PCB.

Figure 7.6: Backup battery interface board.

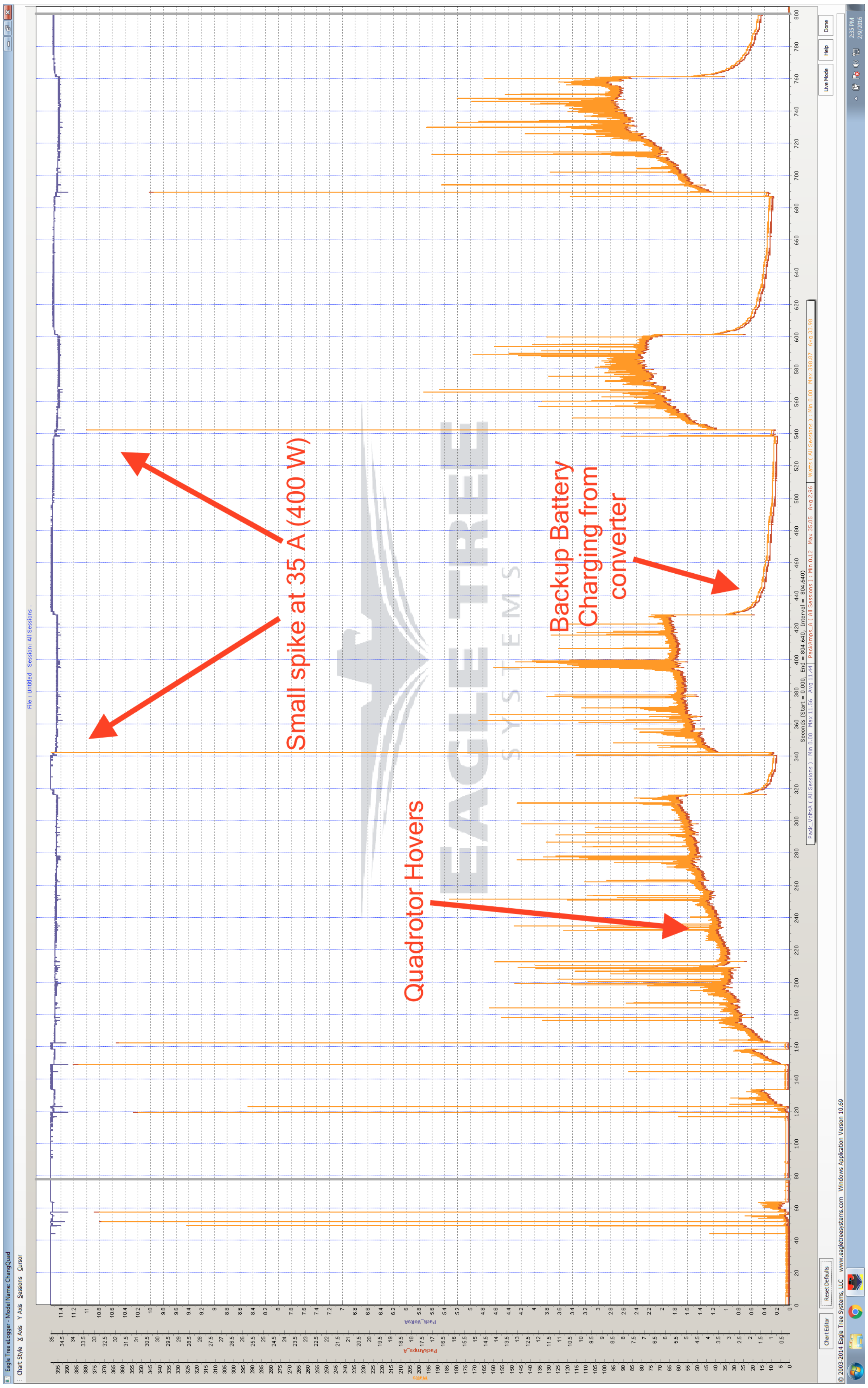


Figure 7.7: Transient power output from converter test log at 100 Hz. (blue is voltage, red is current and yellow is power)

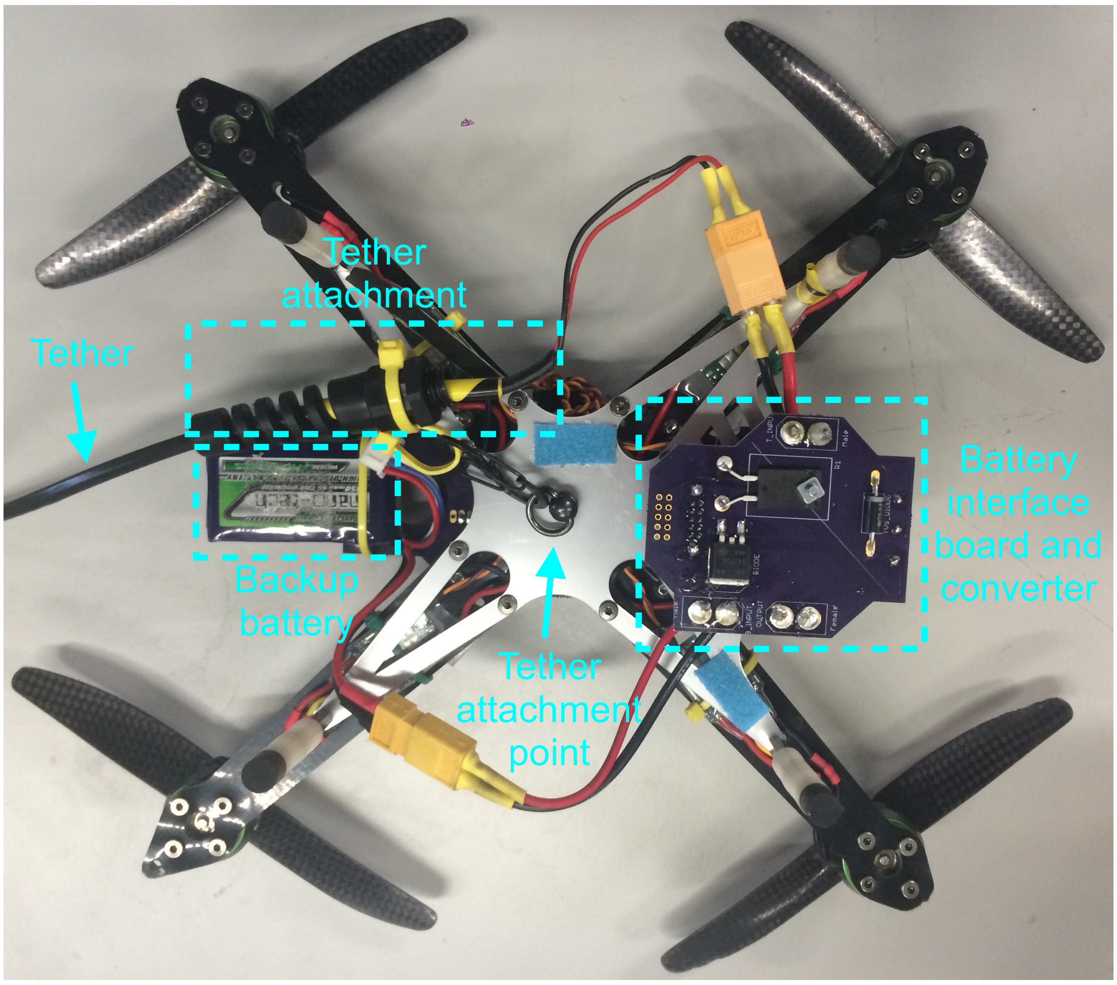


Figure 7.8: The onboard tether power system.

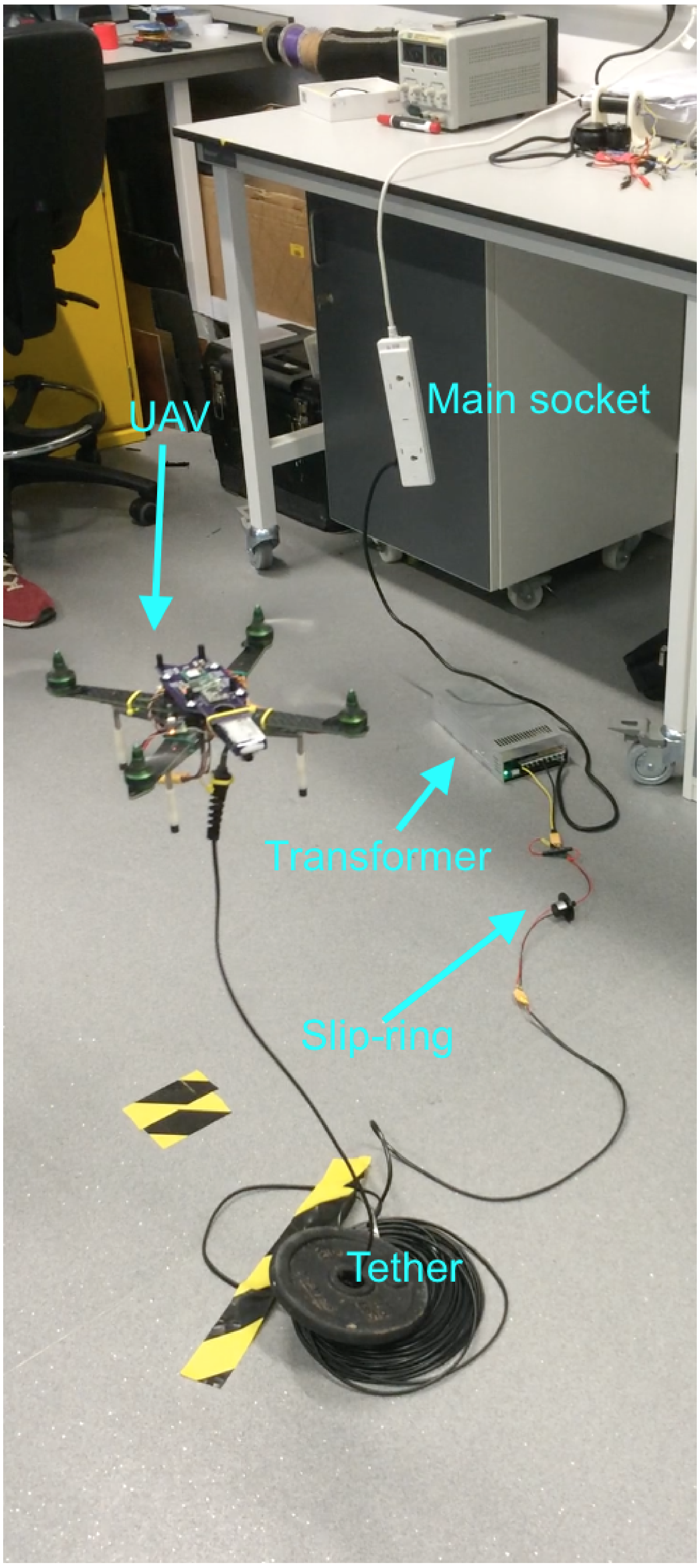


Figure 7.9: The tether power system in operation.

Chapter 8

Conclusion and Further Work

8.1 Conclusion

In this dissertation, we realised a complete, power-on-and-go $250\ mm$ size quadrotor system, using a monocular camera and an onboard IMU as its sensors, relying only on the onboard processing power, to perform autonomous navigation without the assistance from any external infrastructure or offboard sensors.

The model based control architecture implementation and testing work in Chapter 3 demonstrated that the system, which is based on commercial-off-the-shelf components, is able to semi-autonomously operate using a camera as the primary velocity sensor and potentially integrate additional sensors. Besides, the visual inertial sensor fusion architecture implementation and testing work in Chapter 4 has shown the potential of combining the visual SLAM based localisation method with inertial measurement to provide fast and accurate six degree of freedom state estimation, and finally, in Chapter 5, when combining the model based control with visual inertial fusion algorithm, and some additional hardware design optimisation, the entire system was packed into a tiny $250\ mm$ size quadrotor platform, and realising impressive performance. Additionally, with the novel button-on-body design, this quadrotor can be launched from hand without involving remote control. Finally, an innovative tether system was engineered to provide indefinite power to the small size platform. The experimental test result provide a detailed proof of the effectiveness of the flight control performance as well as the sensor localisation performance in real world scenarios. All the mathematical derivations behind the design are clearly presented in this thesis.

Furthermore, instead of passively performing mSLAM while directly traveling to the destination (target waypoint), we integrated the perception requirements of the mSLAM into the navigation control of the UAV. Specifically, we present a novel active one-step-ahead planning algorithm which dynamically generates the optimal next action command in real-time for an autonomous quadrotor based on its current mSLAM observations, so that it not only approaches the target waypoint, but also maximising the localisation accuracy for the onboard mSLAM algorithm. This active planing algorithm serves as the complementary to the state-of-the-art passive mSLAM algorithm, and aiming to improve the operational safety and robustness of the real world GPS-denied scenarios. We have cooperated the active planning algorithm with a third party mSLAM algorithm, and implemented the localisation uncertainty estimation algorithm in real UAV system. We have shown that the algorithm is able to reasonably estimating the localisation uncertainty from the given pose. However, further work needs to be done to implement the Monte-Carlo tree search, which employs the estimation algorithm.

Finally, the engineering detail of a Power-over-Tether system was presented, which was used to sufficiently power a small size quadrotor from a ground source. The corresponding low price tether powering system utilises a 10 metre standard twin 22 SWG copper cable, running 48 VDC through the tether. The air and ground voltage converters/transformers were outsourced from commercial-off-the-shelf components, while an additional backup battery interface for the airborne platform was developed in-house, for the purpose of reducing the transient current in the tether and backup power in case of tether failure.

8.2 Discussion of Contributions

The findings and results presented in this thesis are a step towards practically realising vision based navigation for agile, airborne power-on-and-go systems in a challengingly small form factor. Although this is not the first work using vision processing assist navigating on a UAV, this work is the first realising the full vision navigation functionality on a 250 mm size platform, with all the computation onboard, and with the additional features including button-on-body design and tether power solution. These functionalities could not be achieved without the development of the model-based control algorithm, the highly computationally efficient sensor fusion framework and highly integrated system layout developed along this thesis.

8.2.1 Recall the list of contributions

1. *Development and evaluation of a novel loosely coupled visual-inertial sensor fusion algorithm, based on a monocular SLAM, an accelerometer and a gyroscope, which requires less computation enough to be suitable for high speed 6 DOF state estimation on low power embedded computer.*

2. *Development of a complete micro (*250 mm *motor-to-motor size) quadrotor system capable of navigation in GPS-denied environments, based on onboard sensors and computation only.*

3. *Development and evaluation of a novel model based control algorithm for micro quadrotor autopilot system.*

4. *Design and Implementation of a novel low cost high performance quadrotor autopilot hardware based on commercial-off-the-shelf (COTS) components.*

5. *Design and Implementation of a novel hand-launching method for micro UAV, for intuitive operation.*

6. *Design and Implementation of a novel active planning algorithm, which leads to safer vision-based flight in low contrast environments.*

7. *Design and Implementation of an electrical tether system providing power to the micro UAV from offboard power sources.*

8.2.2 Journal publications

1. Liu, C., Prior, S. D., and Scanlan, J. P. Design and Implementation of a Low Cost Mini Quadrotor for Vision Based Manoeuvers in GPS Denied Environments. *Unmanned Systems*, pages 1–12, DOI: 10.1142/S2301385016500059 (Liu et al., 2016a)

2. Liu, C., Prior, S. D., Teacy, W. L., and Warner, M. Computationally efficient visual- inertial sensor fusion for Global Positioning System-denied navigation on a small quadrotor. *Advances in Mechanical Engineering*, 8(3):1–11, DOI: 10.1177/1687814016640996 (Liu et al., 2016b)

3. Liu, C., Nash, J., and Prior, S. D. A Low-Cost Vision-Based Unmanned Aerial System for Extremely Low-Light GPS-Denied Navigation and Thermal Imaging. *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 9(10):1740–1747. URL: `http://waset.org/publications/10002835` (Liu et al., 2015)

4. Submitted to the *Journal of Intelligent and Robotic Systems*, in the title The Practical Implementation of an Autonomous Micro Quadrotor in GPS-denied Environments, on 3rd September, 2016. Author list: Liu, C, Prior, S.D., and Scanlan, J.P.

8.2.3 Conference publication

1. Liu, C. and Prior, S. D. Design and Implementation of a Mini Quadrotor Control System in GPS Denied Environments. *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015 - Conference Proceedings*, pages 462–469. DOI: 10.1109/ICUAS.2015.7152324 (Liu and Prior, 2015c)

2. Liu, C. and Prior, S. D. Computationally efficient visual-inertial sensor fusion for GPS-denied navigation on a small quadrotor. *2015 International Conference on Innovation, Communication and Engineering*. (Liu and Prior, 2015b)

3. Liu, C. and Prior, S. D. A low-cost vision-based unmanned aerial system for extremely low-light GPS-denied navigation and thermal imaging. *2015 17th International Conference on Intelligent Unmanned Systems*. (Liu and Prior, 2015a)

4. Presentation in *2015 Next Gen Drones conference* in Washington DC. US, with the title: Design and Implementation of an Open Source Small Quadcopter for GPS-Denied Environments. 23-24th June, 2015

Other public engagements

1. Attended *2016 Farnborough International Airshow* as exhibitor. 11-17th July, 2016.

2. Successful indoor flight demonstration to *Unmanned Air Systems Capability Development Centre (UAS CDC)* in Canary Wharf, London. 11th June, 2015.

3. Presented research work in *2015 Engineering@Southampton Autonomous Systems Showcase* at the Royal Academy of Engineering (RAe). 9th June, 2015.

8.3 Implication and Application

After the successful test of the system, it have been shown that vision only navigation is possible as long as it only concerns about local consistency of the map. Thanks to the advantage of the SLAM algorithm, when hovering or moving within a small area, since the camera points at the same scene all the time, the system produces zero drift for all three dimensions. With a sub-centimetre accuracy, the developed UAV system can easily achieve stable and responsive manoeuvre. The precise control and the small form factor, make it safe to operate close to humans. Besides, the 3D sparse map produced from the SLAM algorithm has the potential to be used to generate an elevation mesh map indicating the rough structure of the terrain. Additionally, with the image contained in the keyframe of the SLAM algorithm, the structure-from-motion 3D reconstruction can be conducted by a separate software module for detailed 3D reconstruction.

Moreover, with the button-on-body design and the intelligent self stabilisation capability, it demonstrates an extremely intuitive method for a non-technical person to easily operate a UAV by the place-and-hover action. The capability takes almost all the control work from the human operator to the UAV itself. This ease of use functionality has a huge potential application in consumer personal UAV market, which has shown an exponential growth over the last decade.

Lastly, with a tether power solution, the endurance was boosted from 10 $mins$ to indefinite. This essentially opens up a whole range of applications, which requires long endurance, such as long term monitoring, and some journalism operation, as well as tasks, which require the UAV to physically interact or manipulate the environment.

The resulting small size intelligent hovering platform is an ideal candidate to perform autonomous indoor information gathering tasks in various domains. For example, in military or search and rescue scenarios, such system can be used to gather room structure information, potential hazards information or search for human targets or item of interest, before or without human entry. Besides, the same system can also be used for logistic applications in large warehouses, such as autonomous inventory monitoring. Additionally, for large industrial power plant, it can be used for pipe or other system inspections. Also, with the help of the tether powering system, the restriction of the endurance is removed completely, for long term operation.

8.4 Future Work

8.4.1 Robust operation against extreme and dynamic lighting

There are still multiple limitations to the developed system. The camera based system is very sensitive to the lighting condition, such as extremely dark and bright, and highly dynamic lighting. Therefore, in the current implementation, the camera shutter speed has to be manually fixed to avoid flicking effect by the dynamic shutter speed adjustment. However, this poses the problem that when the lighting condition changes significantly, the vision system might not work due to improper video exposure. This problem can be potentially solved by adding a lighting factor for all tracked features to cancel out the effect of exposure changes. Or in an other way, instead of processing the raw images, the image can be preprocessed by the canny edge filter.

Additionally, thermal/infrared camera can be used for extremely dark situations when no possible light source can be carried with the UAV.

8.4.2 Fusion between PX4Flow camera with SLAM

The camera uses optical flow to measure translational velocity. Vehicle position control can be achieved by integrating the velocity. Since it only requires two consecutive image frames to compute the optical flow, it has the advantage over the SLAM algorithm for its relatively robust performance in terms of fast motion, tracking loss and close to scene operation, whilst it suffers from accumulating error (drifting) by integrating the velocity, and the maximum operational height is 3 m. On the contrary, the visual SLAM solution has much smaller drift with theoretically unlimited height limit (the higher the less accurate, but the more tracking robustness), however, it suffers from the challenge of smooth reinitialisation and close scene operations. Therefore, the proper combination of both sensors will result in a much more robust localisation solution. Given the fusion framework developed in Chapter 4, only minor changes need to be done to fuse PX4Flow measurement, SLAM and inertial measurement.

8.4.3 Active SLAM further implementation

Unlike the GPS-based navigation, the accuracy of the mSLAM-based navigation performance highly depends on the motion of the UAV and the scene within the camera's field of view. Especially with a downward facing camera, when flying above the featureless ground, the SLAM algorithm is very likely for failure. Therefore, in order to avoid this situation in the first place, An active SLAM algorithm was developed in Chapter 6 to integrate the perception requirements of mSLAM into the navigation control of the UAV, instead of passively performing mSLAM while the UAV travels to the destination (target waypoint). Specifically, a novel active one-step-ahead planning algorithm was proposed, which dynamically generates the optimal next action command in real-time for an autonomous quadrotor based on its current mSLAM observations, so that it not only approaches the target waypoint, but also maximised the localisation accuracy for the onboard mSLAM algorithm. The aim was to trade-off between the two conflicting objectives.

However, there are still many area for improvements, such as detailed implementation for Monte-Carlo tree search to find the optimal trajectory based on the localisation quality prediction and tradeoff energy function. Also, the further improvement energy function taking into account the tracking robustness prediction when camera is close to the scene.

8.4.4 Tether system further development

Given the physical connection to the ground station, as a tether. There are multiple advantages that the system can utilise. First of all, this contact to the ground can be seen as a form of sensor, which can be potentially used to localise the quadrotor. This can then be used as a secondary method for GPS-denied navigation. Secondly, given this physical connection, the power-line communication (PLC) method can be used to establish connection between the ground station and the air platform. This form of wired communication becomes extremely secure compared with any form of the wireless communication.

8.4.5 Front facing stereo camera and obstacle avoidance

Although the system has the ability to precisely control its position, it does not yet have any obstacle sensing capability. When a new position is commanded, the UAV will approach the target position blindly. This may introduce a safety hazard by operator fault, or by a dynamic object running into the aircraft. In order to avoid this issue, obstacle sensing and avoidance are required. In this case, stereo vision might be used to add dynamic depth measurement on all pixels.

Moreover, the same stereo camera can also be used for state estimation based on monocular camera odometry techniques with the help from stereo depth perception. In comparison to the

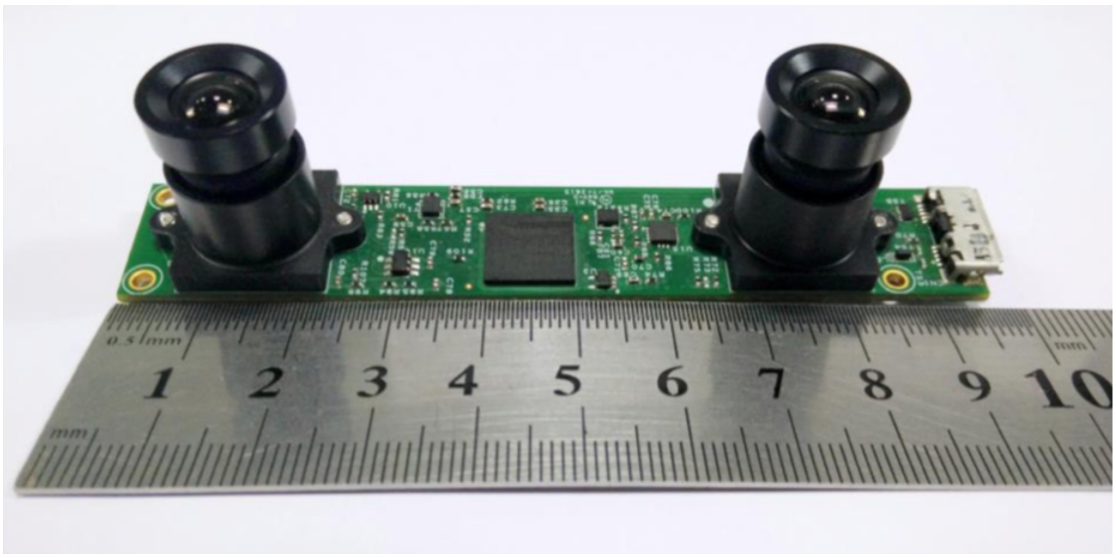


Figure 8.1: Tara stereo camera by E-con systems. (sourced from e-consystems.com)

downward facing monocular camera, it is not only independent to the ground texture, but also potentially suitable for additional 3D reconstruction and obstacle avoidance capabilities. With the current stereo camera production improvements, there are several newly released off-the-shelf small form factor stereo camera, which is factory calibrated and with IMU integration. Tara stereo camera [1] as shown in Figure 8.1.

8.4.6 Multiple UAV collaboration

Since the developed system performs the computation completely onboard, this gives the system outstanding scalability, which allows multiple UAV collaboration in distributed computation approaches. However, in order for the sufficient collaboration between multiple agents, there must be a common coordination shared between different UAVs. In the current stage, the system only has a local coordination system, which resets at the power-on. Thus, the key challenge is to share, match and stitch the sparse map between different UAVs, so that all UAVs are localising themselves with respect to the same master map.

8.4.7 Extend the use case for fixed wing operations

For the fixed wing UAV platforms, although they are not likely to fly indoors, the presented vision based GPS-denied system can be a very useful back-up and enhancement system in the case of GPS failure.

[1] `https://www.e-consystems.com/3D-USB-stereo-camera.asp`

When flying outdoor at higher altitude, the demonstrated monocular SLAM solution has a significant range advantage over other systems, such as optical-flow-ultrasonic solutions, laser scanner solutions. Since theoretically, there isn't a range limit for such system, given its flexible stereo baseline, while only experience the degrade of accuracy with higher altitude, depending on the camera resolution. Moreover, it naturally avoids the featureless ground problem for a downward-facing camera, since the camera viewing area covers significantly large ground space, so that it is almost impossible to encounter a featureless scene. Besides, the forward flying motion also helps the algorithm initialisation procedure.

However, there are still challenges for the extension. The fixed wing UAV launching motion generally require fast horizontal speed close to ground, which will cause problem for the current system to track the rapid moving scene. Besides, the SLAM algorithm operates on local coordinate which originated at its initialisation, thus the fusion of GPS measurement, which operates on global coordinate, must be implemented carefully.

8.4.8 Modular design with 'head-body' separation

It is clear that with the extended intelligence on RPAs, the system is able to obtain substantial autonomy, which is characterised by the robotic field. In other words, the system can be treated more as an aerial robot rather than the remotely controlled aircraft. Although the actuation principle is different from a ground robot, the high level perception and navigation is fundamentally the same with a robot.

Based on such principle, the development of the uniform 'head', which is the hardware to perform high level perception and navigation, with a standard interface to different types for 'bodies', which perform low level perception and actuation, so that the same 'head' can be easily installed onto a specific type of body for the best fit of a specific mission.

8.4.9 Navigation with bio-inspired camera

Conventional camera sensors see the world as a series of image frames with fixed image size. Successive frames contain enormous amounts of redundant information, wasting memory access, RAM, disk space, energy, computational power and time. In addition, each frame imposes the same exposure time on every pixel, making it difficult to deal with scenes containing very dark and very bright regions.

Inspired by the biological eyes, the event-based cameras (popularly produced by DVS[2], as shown in Figure 8.2) solve these problems by using the artificial retina. Instead of wastefully sending entire images at fixed frame rates, only the local pixel-level changes caused by movement in a scene are transmitted at the time they occur. The result is a stream of events at

[2] `http://inilabs.com/products/dynamic-vision-sensors/`

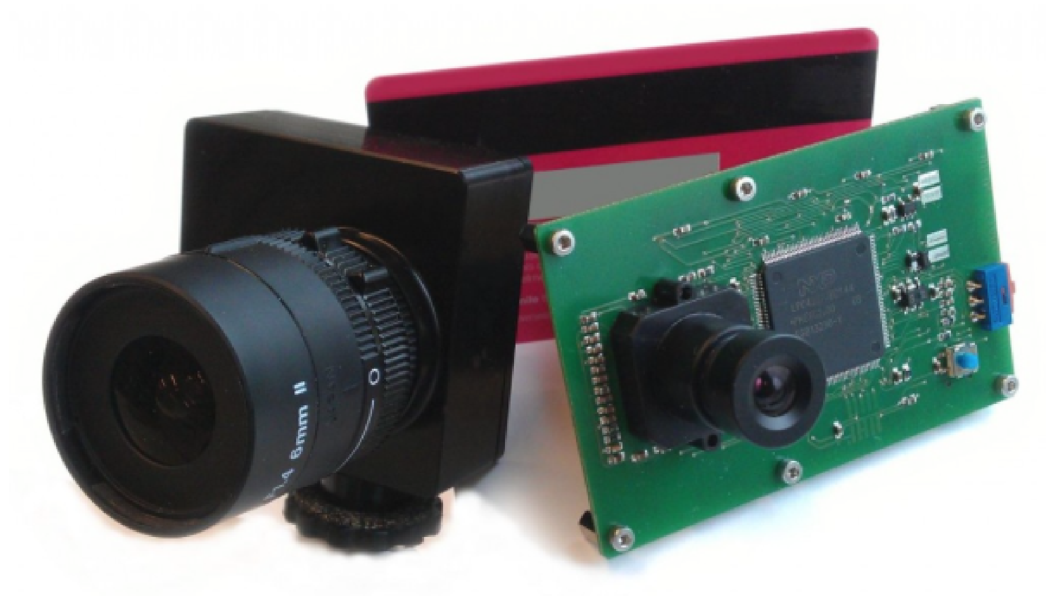


Figure 8.2: DVS event-based cameras. (sourced from inilabs.com)

microsecond time resolution, equivalent to or better than conventional high-speed vision sensors running at thousands of frames per second. Power, data storage and computational requirements are also drastically reduced, and sensor dynamic range is increased by orders of magnitude due to the local processing.

The successful integration of the event-based camera into the robotic applications especially for aerial robots, whose situational perception suffered from its fast manoeuvre and little payload for onboard computation, will open up the door to practical solutions to the current limitations of real-world high performance and robust robotic operation. World's top research institutes just started to pioneer the investigation of using such event-based cameras to perform high speed SLAM since 2014. Bardow et al. (2016); Gallego et al. (2015); Kim (2014); Kim et al. (2016) It is worth restating that the measurement rate of the event-based camera is on the order of a microsecond, its independent pixel architecture provides very high dynamic range, and the bandwidth of an event stream is much lower than a standard video stream. These superior properties of event-based cameras offer the potential to overcome the limitations of real-world computer vision applications relying on conventional imaging sensors. One very recent implementation Kim et al. (2016) from Imperial College shows the impressive initial result of the real-time 6-DOf tracking and 3D reconstruction based on one single event-based camera on a general PC, with the impressive performance specifically on the fast motion.

It is believed that the SLAM algorithm based on the event-based camera will soon become practical, and then the focus will be how to implement efficient algorithm into the embedded platform, and utilise the tracking and reconstruction on the UAV to control its motion for the next level performance.

8.4.10 Google Tango system deployment

Given the recent advance in Augmented Reality (AR) and Virtual Reality (VR) on mobile devices like smartphone and tablet, they have gained the core capabilities for spatial perception. The spatial perception particularly refers to 6 DOF motion tracking, scene depth perception and place recognition. Although, the capabilities were developed for Augmented Reality and Virtual Reality applications, all of them are crucial for robotic perception.

In this field, the Project Tango by GOOGLE is leading the technological advancement. By utilising the modern mobile parallel computing architecture, and advanced imaging and sensor technology (inferred time-of-flight camera, stereo vision and fisheye monocular vision, IMU), a complete hardware-software solution is developed into a single package. It is able to perform real-time 6 DOF motion tracking (state estimation) at over 100 Hz. With the depth sensing, it can also perform real-time 3D reconstruction at over 1Hz, and perform loop closure with recognising the same place being revisited. Such devices is a key advance for the enabling the autonomous robot.

Their developer level smartphone and tablet[3] platform has been released to public. More excitingly, with the partnership with LENOVO, the world's first consumer level tango-enabled smartphone is released[4].

One example is shown in Figure 8.3, where the room is 3D reconstructed in real-time using the tango developer tablet.

Although, The integration of Tango phone on a quadrotor has been implemented in Kumar's Lab[5], further implementation for the higher level perception is left to be investigated.

[3]https://store.google.com/product/tango_tablet_development_kit
[4]http://shop.lenovo.com/us/en/tango/
[5]https://www.youtube.com/watch?v=RE1YHlI1XHo

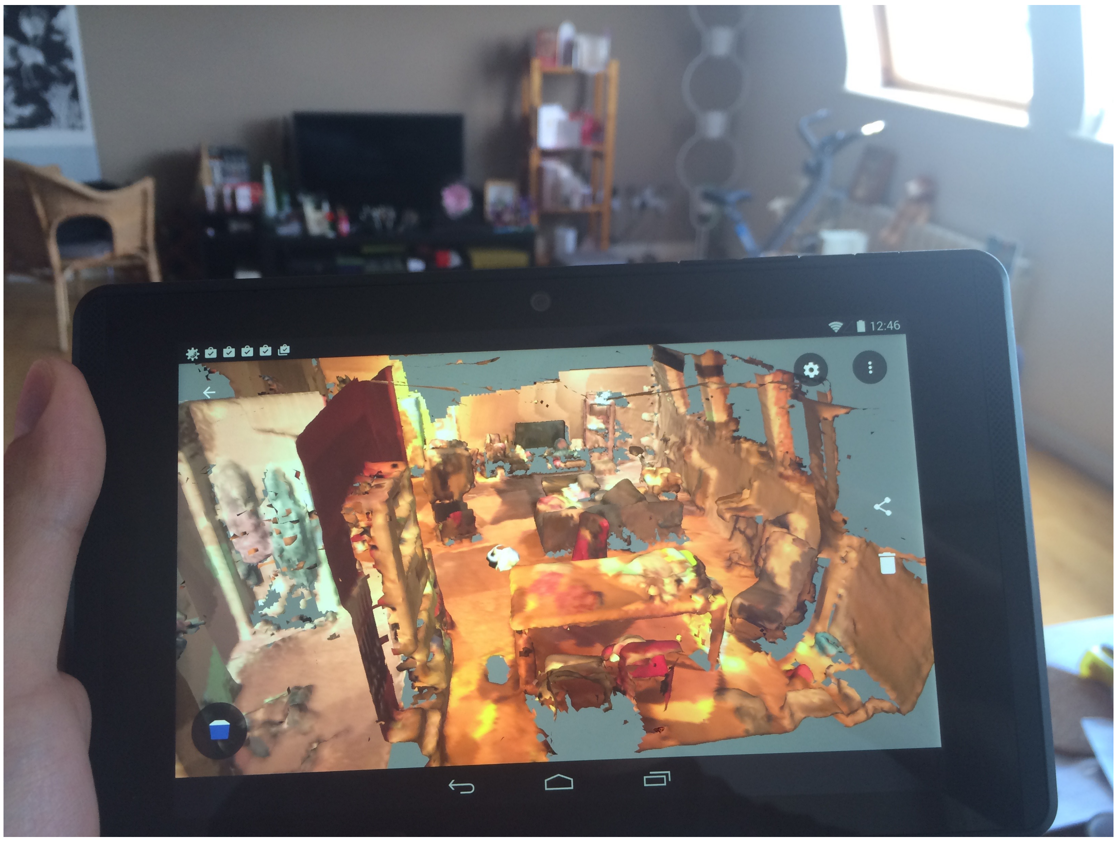


Figure 8.3: Tango developer tablet testing.

References

Abbeel, P., Coates, A., and Ng, a. Y. (2010). Autonomous Helicopter Aerobatics through Apprenticeship Learning. *The International Journal of Robotics Research*, 29(13):1608–1639.

Abdelkrim, N., Aouf, N., Tsourdos, A., and White, B. (2008). Robust nonlinear filtering for INS/GPS UAV localization. In *2008 16th Mediterranean Conference on Control and Automation*, pages 695–702. IEEE.

Achtelik, M., Bachrach, A., He, R., Prentice, S., and Roy, N. (2009). Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments. *Proceedings of SPIE The International Society for Optical Engineering*, 1:733219–733219–10.

Achtelik, M. W., Lynen, S., Weiss, S., Kneip, L., Chli, M., and Siegwart, R. (2012). Visual-inertial SLAM for a small helicopter in large outdoor environments. *IEEE International Conference on Intelligent Robots and Systems*, pages 2651–2652.

Ahmed, M. T., Dailey, M. N., Landabaso, J. L., and Herrero, N. (2010). Robust Key Frame Extraction for 3D Reconstruction from Video Streams. *Proceedings of the Fifth International Conference on Computer Vision Theory and Applications (VISAPP 2010)*, pages 231–236.

Altug, E. (2005). Control of a Quadrotor Helicopter Using Dual Camera Visual Feedback. *The International Journal of Robotics Research*, 24(5):329–341.

Bachrach, A., De Winter, A., He, R., Hemann, G., Prentice, S., and Roy, N. (2010). RANGE - Robust autonomous navigation in GPS-denied environments. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1096–1097.

Baker, S. and Matthews, I. (2004). Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255.

Bardow, P., Davison, A. J., and Leutenegger, S. (2016). Simultaneous Optical Flow and Intensity Estimation from an Event Camera. *29th IEEE Conference on Computer Vision and Pattern Recognition*.

Bennett, S. (1996). A brief history of automatic control. *Control Systems, IEEE*, 16(3):17–25.

Bergerman, M., Amidi, O., Miller, J. R., Vallidis, N., and Dudek, T. (2007). Cascaded position and heading control of a robotic helicopter. *IEEE International Conference on Intelligent Robots and Systems*, pages 135–140.

Bertozzi, M., Broggi, a., Coati, a., and Fedriga, R. I. (2013). A 13,000 km Intercontinental Trip with Driverless Vehicles: The VIAC Experiment. *IEEE Intelligent Transportation Systems Magazine*, 5(January):28–41.

Bibuli, M., Caccia, M., and Lapierre, L. (2007). Path-following algorithms and experiments for an autonomous surface vehicle. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 7(PART 1):81–86.

Biot, M. a. and Lowrie, W. (2007). Fundamentals of geophysics. *Cambridge University press*, 28(2):168–178.

Blösch, M., Weiss, S., Scaramuzza, D., and Siegwart, R. (2010). Vision based MAV navigation in unknown and unstructured environments. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 21–28.

Brandt, J. B. and Selig, M. S. (2011). Propeller Performance Data at Low Reynolds Numbers. *49th AIAA Aerospace Sciences Meeting*, (January):1–18.

Brockers, R., Susca, S., Zhu, D., and Matthies, L. (2012). Fully self-contained vision-aided navigation and landing of a micro air vehicle independent from external sensor inputs. *SPIE Defense, Security, and Sensing*, pages 83870Q–83870Q–10.

Broggi, A., Cerri, P., Felisa, M., Laghi, M. C., Mazzei, L., and Porta, P. P. (2012). The VisLab Intercontinental Autonomous Challenge: an extensive test for a platoon of intelligent vehicles. *International Journal of Vehicle Autonomous Systems (IJVAS)*, 10(3):147–158.

Bry, A., Bachrach, A., and Roy, N. (2012). State estimation for aggressive flight in GPS-denied environments using onboard sensing. In *2012 IEEE International Conference on Robotics and Automation*, pages 1–8. IEEE.

Bubeck, S., Munos, R., Stoltz, G., and Szepesvári, C. (2008). Online optimization in X-armed bandits. *Multi-Armed Bandits*, pages 1–4.

Calusdian, J., Yun, X., and Bachmann, E. (2011). Adaptive-gain complementary filter of inertial and magnetic data for orientation estimation. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1916–1922.

Carlone, L., Du, J., Kaouk Ng, M., Bona, B., and Indri, M. (2014). Active SLAM and exploration with particle filters using Kullback-Leibler divergence. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 75(2):291–311.

Chaumette, F. and Hutchinson, S. (2006). Visual servo control. I. Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90.

Chaumette, F. and Hutchinson, S. (2007). Visual servo control. II. Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1):109–118.

Chaves, S. M., Kim, A., and Eustice, R. M. (2014). Opportunistic sampling-based planning for active visual SLAM. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, number Iros, pages 3073–3080. IEEE.

Civera, J., Grasa, O. G., Davison, A. J., and Montiel, J. M. M. (2010). 1-Point RANSAC for EKF Filtering. Application to Real-Time Structure from Motion and Visual Odometry. *Journal of Field Robotics*, 27(5):609–631.

den Broeck, V. and Guy Driessens, K. (2011). Automatic Discretization of Actions and States in Monte-Carlo Tree Search. *Proceedings of the ECML/PKDD 2011 Workshop on Machine Learning and Data Mining in and around Games*, pages 1–12.

Deters, R. and Selig, M. (2008). Static testing of micro propellers. *26th AIAA Applied Aerodynamics Conference*, (August).

Dierks, T. and Jagannathan, S. (2010). Output feedback control of a quadrotor UAV using neural networks. *IEEE Transactions on Neural Networks*, 21(1):50–66.

Dunkley, O., Engel, J., Sturm, J., and Cremers, D. (2014). Visual-Inertial Navigation for a Camera-Equipped 25g Nano-Quadrotor. *IROS2014 Aerial Open Source Robotics Workshop*, page 2.

Einicke, G. a. (2012). *Smoothing, Filtering and Prediction - Estimating The Past, Present and Future*. InTech.

Elsamanty, M., Khalifa, a., Fanni, M., Ramadan, a., and Abo-Ismail, a. (2013). Methodology for identifying quadrotor parameters, attitude estimation and control. *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics: Mechatronics for Human Wellbeing, AIM 2013*, pages 1343–1348.

Engel, J., Sch??ps, T., and Cremers, D. (2014). LSD-SLAM: Large-Scale Direct monocular SLAM. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8690 LNCS, pages 834–849.

Engel, J., Sturm, J., and Cremers, D. (2012). Camera-based navigation of a low-cost quadrocopter. *IEEE International Conference on Intelligent Robots and Systems*, pages 2815–2821.

Engel, J., Sturm, J., and Cremers, D. (2013). Semi-dense visual odometry for a monocular camera. *Proceedings of the IEEE International Conference on Computer Vision*, pages 1449–1456.

Faessler, M., Fontana, F., Forster, C., and Scaramuzza, D. (2015). Automatic Re-Initialization and Failure Recovery for Aggressive Flight with a Monocular Vision-Based Quadrotor. *International Conference on Robotics & Automation*.

Fernando, H., De Silva, A., De Zoysa, M., Dilshan, K., and Munasinghe, S. R. (2013). Modelling, simulation and implementation of a quadrotor UAV. *2013 IEEE 8th International*

Conference on Industrial and Information Systems, ICIIS 2013 - Conference Proceedings, pages 207–212.

Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO : Fast Semi-Direct Monocular Visual Odometry. *IEEE International Conference on Robotics and Automation (ICRA).*

Fraundorfer, F., Heng, L., Honegger, D., Lee, G. H., Meier, L., Tanskanen, P., and Pollefeys, M. (2012). Vision-based autonomous mapping and exploration using a quadrotor MAV. *IEEE International Conference on Intelligent Robots and Systems*, pages 4557–4564.

Frieden, B. R. and Binder, P. M. (2000). Physics from Fisher Information: A Unification. *American Journal of Physics*, 68:1064.

Gageik, N., Strohmeier, M., and Montenegro, S. (2013). An autonomous UAV with an optical flow sensor for positioning and navigation. *International Journal of Advanced Robotic Systems*, 10:1.

Gallego, G., Forster, C., Mueggler, E., and Scaramuzza, D. (2015). Event-based Camera Pose Tracking using a Generative Event Model. *Arxiv*, 1:1–7.

Garcia, R. D. and Valavanis, K. P. (2009). The implementation of an autonomous Helicopter testbed. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 54(1-3 SPEC. ISS.):423–454.

Grisleri, P. and Fedriga, I. (2010). The BRAiVE autonomous ground vehicle platform. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 7(PART 1):497–502.

Haner, S. and Heyden, A. (2011). Optimal view path planning for visual SLAM. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6688 LNCS:370–380.

Hilsenbeck, S., Moller, A., Huitl, R., Schroth, G., Kranz, M., and Steinbach, E. (2012). Scale-preserving long-term visual odometry for indoor navigation. *2012 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2012 - Conference Proceedings*, (November):13–15.

Honegger, D., Meier, L., Tanskanen, P., and Pollefeys, M. (2013). An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1736–1741.

Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203.

How, J., Bethke, B., Frank, a., Dale, D., and Vian, J. (2008). Real-time indoor autonomous vehicle test environment. *Control Systems, IEEE*, 28(2):51–64.

Huang, A. and Bachrach, A. (2011). Visual odometry and mapping for autonomous flight using an RGB-D camera. *International Symposium on Robotics Research (ISRR)*, pages 1–16.

Jama, M. (2011). Monocular Vision Based Localization and Mapping. *PhD Thesis*.

Jama, M. and Schinstock, D. (2011). Parallel tracking and mapping for controlling VTOL airframe. *Journal of Control Science and Engineering*, 2011:1–10.

Jeong, S. and Jung, S. (2013). Design, Control, and Implementation of Small Quad-Rotor System Under Practical Limitation of Cost Effectiveness. *International Journal of Fuzzy Logic and Intelligent Systems*, 13(4):324–335.

Jones, E. S. and Soatto, S. (2011). Visual-inertial navigation, mapping and localization: A scalable real-time causal approach. *The International Journal of Robotics Research*, pages 1–38.

Julier, S. and Uhlmann, J. (1997). A New Extension of the Kalman Filter to Nonlinear Systems. *Int Symp AerospaceDefense Sensing Simul and Controls*, 3(2):26.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME-Journal of Basic Engineering*, 82(Series D):35–45.

Keller, M., Lefloch, D., Lambers, M., Izadi, S., Weyrich, T., and Kolb, A. (2013). Real-time 3D Reconstruction in Dynamic Scenes using Point-based Fusion. *3Dv*, pages 1–8.

Kelly, J. and Sukhatme, G. S. (2009). Visual-inertial simultaneous localization, mapping and sensor-to-sensor self-calibration. *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA*, pages 360–368.

Kelly, J. and Sukhatme, G. S. (2016). Visual Inertial Sensor Fusion : Localization , Mapping and Sensor-to-Sensor Self-calibration. *The International Journal of Robotics Research*.

Kendoul, F. (2012). Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378.

Kendoul, F., Nonami, K., Fantoni, I., and Lozano, R. (2009). An adaptive vision-based autopilot for mini flying machines guidance, navigation and control. *Autonomous Robots*, 27(3):165–188.

Kim, A. and Eustice, R. M. (2013). Perception-driven navigation: Active visual SLAM for robotic area coverage. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3196–3203.

Kim, H. (2014). Simultaneous Mosaicing and Tracking with an Event Camera. *25th British Machine Vision Conference*, pages 1–12.

Kim, H., Leutenegger, S., and Davison, A. J. (2016). Real-Time 3D Reconstruction and 6-DoF Tracking with an Event Camera. *The 14th European Conference on Computer Vision*.

Klein, G. and Murray, D. (2007). Parallel tracking and mapping for small AR workspaces. *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*, pages 1–10.

Klein, G. and Murray, D. (2008). Improving the agility of keyframe-based SLAM. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5303 LNCS(PART 2):802–815.

Klose, S., Wang, J., Achtelik, M., Panin, G., Holzapfel, F., and Knoll, A. (2010). Markerless, vision-assisted flight control of a quadrocopter. *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pages 5712–5717.

Kontitsis, M., Theodorou, E. a., and Todorov, E. (2013). Multi-Robot Active SLAM with Relative Entropy Optimization. *2013 American Control Conference (ACC 13)*, pages 2757–2764.

Kushleyev, A., Mellinger, D., Powers, C., and Kumar, V. (2013). Towards a swarm of agile micro quadrotors. *Autonomous Robots*, 35(4):287–300.

Lee, T., Leok, M., and McClamroch, N. H. (2010a). Control of Complex Maneuvers for a Quadrotor UAV using Geometric Methods on SE(3). *arXiv preprint arXiv:1003.2005*, (i):8.

Lee, T., Leok, M., and Mcclamroch, N. H. (2010b). Geometric Tracking Control of a Quadrotor UAV on SE (3). *Control*, math.OC(3):5420–5425.

Leung, C., Huang, S., and Dissanayake, G. (2006). Active SLAM using model predictive control and attractor based exploration. *IEEE International Conference on Intelligent Robots and Systems*, pages 5026–5031.

Li, M. and Mourikis, a. I. (2013). High-precision, consistent EKF-based visual-inertial odometry. *The International Journal of Robotics Research*, 32(6):690–711.

Liu, C., Nash, J., and Prior, D. S. (2015). A Low-Cost Vision-Based Unmanned Aerial System for Extremely Low-Light GPS-Denied Navigation and Thermal Imaging. *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 9(10):1740–1747.

Liu, C. and Prior, S. D. (2015a). A low-cost vision-based unmanned aerial system for extremely low-light GPS-denied navigation and thermal imaging. *2015 17th International Conference on Intelligent Unmanned Systems*.

Liu, C. and Prior, S. D. (2015b). Computationally efficient visual-inertial sensor fusion for GPS-denied navigation on a small quadrotor. *2015 International Conference on Innovation, Communication and Engineering*.

Liu, C. and Prior, S. D. (2015c). Design and Implementation of a Mini Quadrotor Control System in GPS Denied Environments. *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015 - Conference Proceedings*, pages 462–469.

Liu, C., Prior, S. D., and Scanlan, J. P. (2016a). Design and implementation of a low cost mini quadrotor for vision based maneuvers in gps denied environments. *Unmanned Systems*, 04(03):185–196.

Liu, C., Prior, S. D., Teacy, W. L., and Warner, M. (2016b). Computationally efficient visual-inertial sensor fusion for Global Positioning System-denied navigation on a small quadrotor. *Advances in Mechanical Engineering*, 8(3):1–11.

Liu, H., Li, J., Yao, J., and Hu, F. (2010). Backstepping based adaptive control for a mini rotorcraft with four rotors. *ICCMS 2010 - 2010 International Conference on Computer Modeling and Simulation*, 1:472–476.

Lobo, J. and Dias, J. (2003). Vision and Inertial Sensor Cooperation Using Gravity as a Vertical Reference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1597–1608.

Lucas, B. D. and Kanade, T. (1981). An Iterative Image Registration Technique with an Application to Stereo Vision. *Imaging*, 130(x):674–679.

Lynen, S., Achtelik, M. W., Weiss, S., Chli, M., and Siegwart, R. (2013). A robust and modular multi-sensor fusion approach applied to MAV navigation. *IEEE International Conference on Intelligent Robots and Systems*, pages 3923–3929.

Madgwick, S. O. H., Harrison, A. J. L., and Vaidyanathan, R. (2011a). Estimation of IMU and MARG orientation using a gradient descent algorithm. *IEEE International Conference on Rehabilitation Robotics*, (1945-7898):1–7.

Madgwick, S. O. H., Harrison, A. J. L., and Vaidyanathan, R. (2011b). Estimation of IMU and MARG orientation using a gradient descent algorithm. In *2011 IEEE International Conference on Rehabilitation Robotics*, pages 1–7. IEEE.

Mahony, R., Hamel, T., and Pflimlin, J. M. (2008). Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions on Automatic Control*, 53(5):1203–1218.

Mahony, R., Kumar, V., and Corke, P. (2012). Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor. *IEEE Robotics & Automation Magazine*, 19(3):20–32.

Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2520–2525.

Mettler, B., Tischler, M. B., and Kanade, T. (2002). System Identification Modeling of a Small-Scale Unmanned Rotorcraft for Flight Control Design. *Journal of the American Helicopter Society*, 47(1):50–63.

Michael, N., Mellinger, D., Lindsey, Q., and Kumar, V. (2010). The GRASP multiple micro-UAV testbed. *IEEE Robotics and Automation Magazine*, 17(3):56–65.

Montemerlo, M., Thrun, S., Roller, D., and Wegbreit, B. (2003). FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. *IJCAI International Joint Conference on Artificial Intelligence*, 18:1151–1156.

Mostegel, C., Wendel, A., and Bischof, H. (2014). Active Monocular Localization : Towards Autonomous Monocular Exploration for Multirotor MAVs. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3848–3855.

Natesan, K. and Bhat, M. (2007). Design and flight testing of H; lateral flight control for an unmanned air vehicle. *2007 IEEE International Conference on Control Applications*.

Newcombe, R. a., Davison, A. J., Izadi, S., Kohli, P., Hilliges, O., Shotton, J., Molyneaux, D., Hodges, S., Kim, D., and Fitzgibbon, A. (2011a). KinectFusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136. IEEE.

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. (2011b). DTAM: Dense tracking and mapping in real-time. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2320–2327.

Nikolic, J., Rehder, J., Burri, M., Gohl, P., Leutenegger, S., Furgale, P. T., and Siegwart, R. (2014). A Synchronized Visual-Inertial Sensor System with FPGA Pre-Processing for Accurate Real-Time SLAM. *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 431–437.

Nützi, G., Weiss, S., Scaramuzza, D., and Siegwart, R. (2011). Fusion of IMU and vision for absolute scale estimation in monocular SLAM. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 61(1-4):287–299.

Nyman, J. (2012). Designing an Autonomous Exploration Architecture for an Indoor Quadcopter. *Master Thesis*, pages 2011–2012.

Opower, H. (2002). Multiple view geometry in computer vision. *Optics and Lasers in Engineering*, 37(1):85–86.

Park, S., Won, D. H., Kang, M. S., Kim, T. J., Lee, H. G., and Kwon, S. J. (2005). RIC(Robust Internal-loop Compensator) based flight control of a quad-rotor type UAV. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pages 1015–1020.

Phang, S. K., Li, K., Yu, K. H., Chen, B. M., and Lee, T. H. (2014). Systematic Design and Implementation of a Micro Unmanned Quadrotor System. *Unmanned Systems*, 02(02):121–141.

Phelan, R. M. (1977). *Automatic control systems*. Cornell University.

Pizzoli, M., Forster, C., and Scaramuzza, D. (2014). REMODE : Probabilistic , Monocular Dense Reconstruction in Real Time. *Proc. IEEE International Conference on Robotics and Automation (ICRA)*.

Qi, J., Song, D., Dai, L., Han, J., and Wang, Y. (2010). The New Evolution for SIA Rotorcraft UAV Project. *Journal of Robotics*, 2010(Figure 1):1–9.

Roussillon, C., Gonzalez, A., Solà, J., Codol, J.-M., Mansard, N., Lacroix, S., and Devy, M. (2011). RT-SLAM: A Generic and Real-Time Visual SLAM Implementation. In *Lecture Notes in Computer Science*, volume 6962 LNCS, pages 31–40.

RPAS Steering Group (2013). Roadmap for the integration of civil Remotely-Piloted Aircraft Systems into the European Aviation System. *Final report*, page 16.

Rusu, R. B. and Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1–4.

Sa, I., He, H., Huynh, V., and Corke, P. (2013). Monocular Vision based Autonomous Navigation for a Cost-Effective Open-Source MAVs in GPS-denied Environments. pages 1–6.

Shen, S., Mulgaonkar, Y., Michael, N., and Kumar, V. (2013a). Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor. In *Robotics: Science and Systems*.

Shen, S., Mulgaonkar, Y., Michael, N., and Kumar, V. (2013b). Vision-based state estimation for autonomous rotorcraft MAVs in complex environments. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1758–1764.

Smith, S. P. and Johnson, D. (2013). Advances in the Science and Technology of Position, Navigation, and Timing (PNT). In *AIAA Guidance, Navigation, and Control (GNC) Conference*, Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics.

Strasdat, H., Davison, A. J., Montiel, J. M. M., and Konolige, K. (2011). Double window optimisation for constant time visual SLAM. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2352–2359.

Strasdat, H., Montiel, J. M. M., and Davison, A. J. (2010). Real-time monocular SLAM: Why filter? *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2657–2664.

Thrun, S. (2004). Simultaneous Localization and Mapping with Sparse Extended Information Filters. *The International Journal of Robotics Research*, 23(7-8):693–716.

Tian, Y., Wei, H., and Tan, J. (2013a). An adaptive-gain complementary filter for real-time human motion tracking with MARG sensors in free-living environments. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 21(2):254–264.

Tian, Y., Zhang, J., and Tan, J. (2013b). Adaptive-frame-rate monocular vision and IMU fusion for robust indoor positioning. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2257–2262.

Valencia, R., Valls Miro, J., Dissanayake, G., and Andrade-Cetto, J. (2012). Active pose SLAM. *IEEE International Conference on Intelligent Robots and Systems*, pages 1885–1891.

Varesano, F. (2013). FreeIMU: An Open Hardware Framework for Orientation and Motion Sensing. *arXiv preprint.*

Vasquez-Gomez, J. I., Sucar, L. E., Murrieta-Cid, R., and Lopez-Damian, E. (2014). Volumetric Next-best-view Planning for 3D Object Reconstruction with Positioning Error. *International Journal of Advanced Robotic Systems*, pages 1–13.

Vogiatzis, G. and Hernández, C. (2011). Video-based, real-time multi-view stereo. *Image and Vision Computing*, 29(7):434–441.

Weinstein, A. and Littman, M. (2012). Bandit-Based Planning and Learning in Continuous-Action Markov Decision Processes. *Twenty-Second International Conference on Automated Planning and Scheduling*, pages 306–314.

Weiss, S., Achtelik, M., Kneip, L., Scaramuzza, D., and Siegwart, R. (2011). Intuitive 3D maps for MAV terrain exploration and obstacle avoidance. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 61(1-4):473–493.

Weiss, S., Achtelik, M. W., Chli, M., and Siegwart, R. (2012a). Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 31–38.

Weiss, S., Achtelik, M. W., Lynen, S., Chli, M., and Siegwart, R. (2012b). Real-time Onboard Visual-Inertial State Estimation and Self-Calibration of MAVs in Unknown Environments. *(ICRA), 2012 IEEE*, 231855:957–964.

Weiss, S. and Siegwart, R. (2011). Real-time metric state estimation for modular vision-inertial systems. *Proceedings - IEEE International Conference on Robotics and Automation*, 231855:4531–4537.

Weiss, S. M. (2012). *Vision Based Navigation for Micro Helicopters (PhD Thesis - Weiss 2012)*. PhD thesis.

Welch, G. and Bishop, G. (2006). An Introduction to the Kalman Filter. *In Practice*, 7(1):1–16.

Wenhardt, S., Deutsch, B., Angelopoulou, E., and Niemann, H. (2007). Active visual object reconstruction using D-, E-, and T-optimal next best views. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–7.

Werlberger, M., Pock, T., and Bischof, H. (2010). Motion estimation with non-local total variation regularization. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2464–2471.

Yaakov, B.-S., X. Rong, L., and Thiagalingam, K. (2013). *Estimation with Applications to Tracking and Navigation*, volume 53. John Wiley & Sons, Inc., New York, USA.

Yong, E. and Liittschwager, D. (2016). Inside the eye: Nature's most exquisite creation. `http://ngm.nationalgeographic.com/2016/02/evolution-of-eyes-text`.

Yun, B., Peng, K., and Chen, B. M. (2008). Enhancement of GPS signals for automatic control of a UAV helicopter system. *2007 IEEE International Conference on Control and Automation, ICCA*, pages 1185–1189.

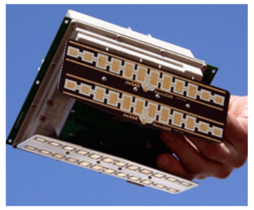
Zikou, L., Papachristos, C., and Tzes, A. (2015). The Power-over-Tether system for powering small UAVs: Tethering-line tension control synthesis. *2015 23rd Mediterranean Conference on Control and Automation, MED 2015 - Conference Proceedings*, pages 681–687.

Zimmerman, D. (2014). Understanding Rifle Precision. `http://www.thetruthaboutguns.com/2014/12/daniel-zimmerman/understanding-rifle-precision/`.

Appendices

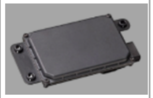
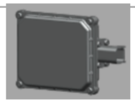
A.1 SAR Systems Comparison

Table A.1: SAR comparison.

Nano Synthetic Aperture Radar (nanoSAR)				
Product	IMSAR NanoSAR	ScanEagle NanoSAR	Sandia National Labs SAR	
Picture				
Release Year	2013	2008	2005	
Weight (without antenna)	1.18 kg without antenna	0.9 kg	4 kg	
Size	13.97 cm x 8.9 cm x 5 cm + Antenna + IMU	15.75 cm x 19 cm x 11.4 cm	17.8 cm cube	
Range	1 - 3 km	1 km	10 km@4"res	
Resolution	very fine, 0.3, 0.5, 1, 2, 5, 10 m	1 m	23 km@12"res	
Power Consumption	< 25 W	15 W	60 W	
Resources	ww.imsar.com/products/display/2064/38/n	rdmicrosystems.com/L4E_sar.htm#SANDIA_SAR		
Price	~£60,000	Non-commercial	Non-commercial	

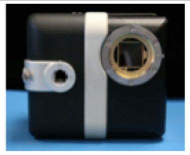
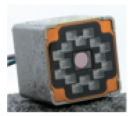
A.2 Automotive RADAR Systems Comparison

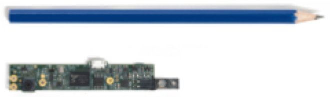
Table A.2: Automotive RADAR comparison.

Automotive RADAR								
Manufacturer	**RADAR**	**Picture**	**Mass**	**Power**	**Range**	**Horizontal FOV**	**Vertical FOV**	**Update Rate**
Delphi	ESR 9.21.21		575 g	< 12 W	1 m to 60 m / 174 m	45 degree / 10 degree	4.2 - 4.75 degree	>= 20 Hz
	ESR 9.21.15							
	RSDS		380 g	7 W	0.5 m to 80 m	75 degree	5 degree	
Smart Micro Automotive RADAR	SMS UMRR Type 29		330 g	3.7 W	1 m to 160 m	18 degree	4 degree	
	SMS UMRR Type 30		295 g		1 m to 90 m	35 degree	5 degree	
	SMS UMRR Type 31		295 g		1 m to 45 m	50 degree	5 degree	
Smart Micro RADAR Altimeter	SMS UMRR-0A		350 g standard housing		0.5 m to 500 m	NA	NA	60 Hz
			160 g integrated version					

A.3 Flash LIDAR Systems Comparison

Table A.3: Flash LIDAR comparison.

Name	TigerEye 3D Flash LIDAR Camera Kit™	TigerCub 3D Flash LIDAR Camera™	SwissRanger™ SR4000	FOTONIC e-series	PrimeSense Carmine 1.08	Kinect 1.0
Picture						
Range	60m, 150m, 450m, 110m		0.1-5m, 0.1-10m	0-7m, 0-10m	0.8m-3.5m	0.7–6 m
Accuracy			+/-10mm, +/-15mm	+/-10~+/-40mm		
Field of View	45°, 45°*22°, 9°, 3°	60°, 45°, 30°, 8.6°, 3°	43.6° (h) x 34.6° (v) or 69° (h) x 56° (v)	70° x 53°, 45° x 34°	57.5° ×45°	43° × 57°
Resolution	128*128, 128*64	128 x 128	176 x 144	160 (h) x 120 (v)	640x480	640x480
Max. Frame Rate (fps)	30	20-30	50	55	60	30
Weight	2kg, 1.6kg	0.8kg (1.8lbs)	470 g (USB), 510 g(Ethernet)	800 g		
Size (mm)	110 x (113/150) x 107	(without lens): 110 x 100 x 120	65 x 65 x 68 (USB), 65 x 65 x 76 (Ethernet)	80x80x86.3	180 x 25 x 35	
Voltage Supply	24 V DC (+/- 4V)	24VDC	12 V (-2%; +10%), maximum 1.0 A (typical 0.8 A)	24 V		
Power		15 W	12W	Typical 10 W max 20 W	2.25W	
Features	The TigerEye Cooling System	Able to Integrate External Zephyr Laser Camera	C, C++, Matlab API	C API: FZ-aPI for C, C++, OpenNI, PC	OpenNI	OpenNI
				1.5GHz Dual-core ARM Cortex-a9 onboard wi	Can't run below 5 degree temperature	
Source	dvancedscientificconcepts.com/products/	w.advancedscientificconcepts.com/products/tigo	http://www.ebay.com/itm/MESA-Imaging-swissranger-sr4000-3	ww.fotonic.com/assets/documents/fotonic_E40-70	se.com/wp-content/uploads/2012/12/PrimeSenses_3DsensorsWeb.pdf	
Price			~£2000		£120	£60

Name	Kinect 2.0	Asus Xtion Pro	PrimeSense Capri 2.5	pmd[vision]® CamBoard nano	Leap Motion	
Picture						
Range	0.8 m - 4.0 m	0.8m and 3.5m	0.4m-3.5m	0.05 - 0.5m	~ 1 m	
Accuracy			1%			
Field of View	70° × 60°	58° × 45°	57.5° × 45°	90° x 68°	150°	
Resolution	500×500	640x480, 320x240	640x480, 320x240			
Max. Frame Rate (fps)	30	30, 60	30, 60	< = 90	~166	
Weight			99.2g		46g	
Size (mm)		180 x 35 x 50	119.2 x 27.9	37 x 30 x 25	76.2 x 12.7 x 30.5	
Voltage Supply				5 V @ 500 mA		
Power		< 2.5 W		2.5 W		
Features	1080p widescreenVideo and active IR	OpenNI	OpenNI	Very small size and low power	Wide Angle, High Frame rate	
	No OpenNI suport		KickStarter	C API	Complete SDK	
Source		http://www.newegg.com/Product/Product.aspx?Item=N82E16826785030		http://pmdtec.com/products_services/referenc	https://www.leapmotion.com/developers	
Price	£268	£100	~£80	£400	£50	

A.4 Laser Scanners Comparison

Table A.4: Laser scanner comparison.

Laser Scanner									
Manufacturer	**Picture**	**Laser Scanner**	**Mass**	**Power**	**Range**	**Best Resolution (distance/angle)**	**Horizontal FOV**	**Scan Rate**	**Price**
Hokuyo		UTM-30LX	370 g	8.4 W	0.1 m to 60 m	30 mm / 0.25 degree	270 degree	40 Hz	£4180
		UTM-30LX-EW							£3830
		UXM-30LX-EW	800 g	10 W	0.1 m to 50 m	50 mm / 0.25 degree	190 degree	20 Hz	£3613
		UHG-08LX	500 g	3.6 W	0.3 m to 11 m	30 mm / 0.36 degree	270 degree	15 Hz	£2460
		URG-04LX	160 g	2.5 W	0.2 m to 5.6 m	10 mm / 0.36 degree	240 degree	10 Hz	£1900
		URG-04LX-UG01	160 g	2.5 W	0.2 m to 5.6 m	30 mm / 0.36 degree		10 Hz	£1000
		PBS-03JN	500 g	6 W	0.2 m to 3 m	/ 1.8 degree	180 degree	3.6 Hz	£1000
SICK		SICK LMS100	1100 g	20 W	0.5 m to 20 m	12 mm / 0.25 degree	270 degree	50 Hz	£2360
		SICK LMS111		60 W					£2850
		SICK LMS151			0.2 m to 50 m				£3647
		SICK TiM310	150 g	3 W	0.05 m to 4 m	30 mm / 0.25 degree		7.5 Hz	£916
Velodyne		VLP-16	600 g	<10 W	150-200 meters		360 x 30 degree		£4864
RoboPeak		RPLIDAR360	170 g	1 W	0.2 m to 6 m	10 mm / 1 degree	360 degree	5.5 Hz	£259
PulsedLight		LIDAR-Lite Laser Rangefinder	20 g	4d 4h 48m	40m	25 mm / x	0 degree	100 Hz	£49

A.5 Camera Systems Comparison

Table A.5: Camera comparison.

Camera				
	Optical Flow Cameras			
Name	ADNS3080 board	PX4FLOW	Caspa™ VL	e-CAM56 37x GSTIX 5 MP
Picture				
Mass	40 g	17.5 g	22.9 g	10 g
Power	200 mW	575 mW	320 mW	420 mW
Resolution	30 x 30	752 x 480	752 x 480	2592 x 1944
Frame Rate	6400 Hz	250 Hz	60 Hz	30 Hz (720p)
Spectrum	Not suitable for indoor or low light outdoor condition	High light sensitivity	Visible	Visible
price	£24	£91	£45	£42