Flinders
UNIVERSITY
ADELAIDE AUSTRALIA

2017

Autonomous Takeoff and Landing of a Small Drone from the TopCat ASV

Drone Platform Development and Software Control Integration.

Author: Stephen Keen

Supervisors: Assoc. Prof Karl Sammut
Dr. Greg Ruthenbeck
Mr. Jonathan Wheare

A thesis submitted in partial fulfilment of the requirements for the degree of
Bachelor Degree of Engineering (Electronic) (Honours)

Flinders University
School of Computer Science, Engineering and Mathematics (CSEM)

October 2017

DECLARATION

I hereby do certify that this work presented is my own original work, and does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief it does not contain any material previously published or written by another person except where due reference is made in the text.

ACKNOWLEDGEMENTS

The author wishes to extend a hearty thanks to Assoc. Prof. Karl Sammut for the bold faith and strong commitment he has shown in creating and nurturing this project. In particular, the author is grateful for the invitation to join this project and for the enthusiasm and interest the supervisor showed throughout this challenging project.

The author would like to thank the following for their valuable contribution in various ways throughout this project: Dr. Greg Ruthenbeck for his abundant advice conceptually and his willingness to teach the practical elements of Drone design, construction and flying; Mr. Jonathan Wheare for his broad advice on maritime navigation strategies, his help resolving teething software issues and his dedication throughout the project, including providing feedback on versions of this manuscript; Mr. Ben Chartier (Nova Systems) for his input in meetings and the Technical Services team, who provided timely advice, supplies and inquisitiveness to encourage us throughout.

The author would also like to express his sincere gratitude for cooperation and faithful commitment of the members of the Cat'sEyes UAV team, without whom, this project could not have been enjoyed so thoroughly. The willingness to confront difficult challenges, work through the decision making process, evaluate and improve the developed work, and achieve a memorable achievement has been a testament to members' character and teamwork. Together, the group has developed a sound understanding or a number of specializations that was not possible on one's own.

Lastly, I would like to acknowledge the dedicated support of my parents and family, faithfully providing physical, emotional and spiritual sustenance throughout the degree, and for their patient understanding through the numerous tests, that have led to this accomplishment.

ABSTRACT

Autonomous deployment and recovery of an unmanned aerial vehicle is essential if autonomous surface vessels are to exploit the greater perspective drones provide of the environment for better mission planning in uncertain marine environments. This report presents the development and field testing results of both the aerial vehicle platform and the system control software, utilizing the Mission Planning and Visual Guidance work from the companion projects. The project has delivered a durable, modular quadcopter design that maximized the use of existing technologies, and implemented a multiplatform software control that has undergone extensive component-wise field testing. The working software prototype has successfully commanded autonomous take-off, mission following and tracking of a slow-moving target in practical flight trials with the developed drone. Moreover, this project has delivered a vision based landing algorithm capable of precision landing in mild winds in both fair and overcast conditions. The precision landing system developed has achieved a 30% success rate of on a stationary one square meter target and 80% success within two square meters. It is expected to work without modification on a purpose built, horizontally moving platform. Significant additional work is required to complete the system integration and implement enhancements to achieve the ultimate goal of autonomous landing on a moving vessel. However, this project has laid a firm foundation for attaining this goal, and is one step closer to reaching the overall objective of cooperative autonomy in marine environments.

TABLE OF CONTENTS

DECLARATION 3
ACKNOWLEDGEMENTS 3
ABSTRACT 5
TABLE OF CONTENTS 6
LIST OF FIGURES 9
LIST OF TABLES 12
NOTATION 13
1. INTRODUCTION 1
2. LITERATURE REVIEW 4
2.1. Overview of Current Surface Vessel's Design 4
2.1.1. TopCat's Sensors 5
2.1.2. TopCat's Neural Capabilities 7
2.1.3. TopCat's Communication 8
2.2. Globally: Autonomous Landing of Unmanned Aerial Vehicles 9
2.2.1. Autonomous Co-operation between Unmanned Vehicles 9
2.2.2. Autonomous Landing of UAVs 9
2.2.3. Autonomous Landing of UAVs on moving platforms, outdoors 10
2.3. Lessons from Similar Implementations in Marine Environments 12
2.3.1. Take-off Strategy 12
2.3.2. Landing Strategy 12
2.3.3. Landing Platform Structure Design 17
2.4. Software: System Integration 19
2.4.1. Sensors 19
2.4.2. Computational Load Constraints 21
2.4.3. Communication Links 23
2.4.4. Summary of Overall UAV, ASV System Requirements 23
2.5. Unmanned Aerial Vehicle Design 24
2.5.1. Characteristics of Similar functioning UAVs 24
2.5.2. Requirements for chosen UAV 26
2.6. Project Design Overview 27
2.6.1. Design requirements 27
2.6.2. Research question defined 27
2.6.3. Key assumptions 27
3. METHODOLOGY/ EXPERIMENT DETAILS 29

3.1. Selection of Suitable Sensors and UAV 29
3.1.1. Suitable Flight Controller for Autonomous Flight 29
3.1.2. Selection of Key Sensors for Autonomous Flight 31
3.1.3. Refined specifications from requirements 32
3.1.4. Evaluation of Off-the-Shelf vs. Custom Built Drone 37
3.1.5. Drone Assembly Testing Process 38
3.1.6. Drone Landing Pad Development Process 38
3.2. Overall System Software Control of the Autonomous Drone 39
3.2.1. Detailed Control Sequence of the Autonomous Take-off & Landing 39
3.2.2. Functional Block Diagram of Drone Control Software 41
3.2.3. Information Flow between the Software Nodes 42
3.3. UAV Control Implementation Software for the Drone 42
3.3.1. Breakdown of UAV Control Implementation 42
3.3.2. ROS to Flight Controller Linking Software 45
3.3.3. UAV Control Software Testing Procedure 45
4. RESULTS & ANALYSIS 47
4.1. Drone Construction 47
4.1.1. Final Drone Components & Accessories List 47
4.1.2. Drone Assembly 49
4.1.3. Drone Flight Tests 53
4.1.4. Actual Specifications from the Drone's performance 57
4.1.5. Landing Pad Design 57
4.2. Software Development – UAV Control Implementation System 59
4.2.1. ROS to Flight Controller Link 59
4.2.2. UAV Control Software State Machine 60
4.2.3. Test Results of Key States of the UAV Control 62
4.3. Overall Software System integration 68
4.3.1. Handling Emergency & Failsafe Conditions 70
5. DISCUSSION 72
5.1. Evaluation of the Drone Design 72
5.2. Evaluation of UAV Control Software 72
5.3. Evaluation of Overall System Performance & Significance 74
5.4. Recommendations for Future Research 75
6. CONCLUSION 78
References 79
APPENDIX A: Configuration Parameters File for PIXHAWK 83

APPENDIX B: Procured Parts List & Links 98
APPENDIX C: Required Pre-requisites for Running the Software 102
APPENDIX D: Operational Issues & their Resolution 105

LIST OF FIGURES

Figure 2.1: Flinders' Autonomous Surface Vessel, named TopCat V2. Picture © CMECI, 2016. 4

Figure 2.2: Overview of TopCat V2's sensor array. Picture © CMECI, 2016. 5

Figure 2.3: Visualized data from the Velodyne LiDAR. Picture © CMECI, 2016. 6

Figure 2.4: Graphical representation of the Navigation and Guidance structure, where blue represents ROS nodes, green signifies existing sensors, and yellow and red indicate future work (Sammut et al.). 7

Figure 2.5: Quadrotor taking off from a moving platform (Ajmera et al., 2015) and tracking and landing on a platform identified by a unique AR tag (Lee et al., 2012). 9

Figure 2.6: Image of the quadrotor completing its landing on the moving platform (Kim et al., 2014). 11

Figure 2.7: Time Domain representation of the sequence of tasks required for autonomous landing. Each task, though independent, needs to be run in parallel, sharing its data with the other tasks (Djapic et al., 2015). 13

Figure 2.8: System level structure for performing autonomous landing on the moving platform (Kanellakis & Nikolakopoulos, 2017). 14

Figure 2.9: State Diagram of the Tracking process of the landing pad (Ajmera et al., 2015). 16

Figure 2.10: Left: CAD model of ASV & UAV (Weaver et al., 2013), Right: Conical metal platform for Vapor 55 Helicopter (Djapic et al., 2015). 17

Figure 2.11: Left: RIVERWATCH team of ASV and UAV, Right: showing the 1m by 1.3mm landing net centre marking pattern (Pinto et al., 2014) & (The Three Laws. 2017). 17

Figure 2.12: The trajectory paths of different landing tests by the drone onto the landing pad (Pinto et al., 2014). 18

Figure 2.13: Complete CAD Model of TopCat V2, showing the clear space for landing pad. Picture © Flinders CMECI, 2016 18

Figure 2.14: Common UAV sensors: a) Ultrasonic sensor, b) Visual stereo, c) Laser Range finder, d) monocular camera (Kanellakis & Nikolakopoulos, 2017). 19

Figure 2.15:Key system components for autonomous landing, showing the two main communication links (Chandra et al., 2016). 20

Figure 2.16: The PIXHAWK MAV system diagram, showing visual tracking, obstacle avoidance and path planning on a dedicated computer, feeding the low-level flight controller. (Meier et al., 2012) . 21

Figure 2.17: Block Diagram representation of the ROS bridge (Weaver et al., 2013). 22

Figure 2.18: Some of the different types of drones on the market (Kanellakis & Nikolakopoulos, 2017). 24

Figure 2.19: The PIXHAWK MAV drone, having on-board image processing capability. (Meier et al., 2012) 25
Figure 3.1: The PIXHAWK Autopilot showing the various connection interfaces available. The connectors to the motors and accessories are on the top (Autopilot, 2017). 30
Figure 3.2: Diagram illustrating the three-phase brushless DC motor coils and excitation sequence. © Servo Magnetics Inc.(Drone_Insider, 2017) 32
Figure 3.3: Breakout of three-phase brushless DC motor (FPV_Drone, 2017). 33
Figure 3.4: Typical Lithium-ion Polymer (LiPo) battery specifications (Liang, 2017). 35
Figure 3.5: Quadrotor Control Strategy to perform Autonomous Take-off and Landing on the Autonomous Vessel, TopCat. 40
Figure 3.6: Overall Diagram of the Quadrotor Software Control in relation to the ASV control. 41
Figure 3.7: ROS Topic messages between the main nodes of the Quadcopter control. 42
Figure 3.8: Quadrotor Control Software States and their component tasks. Note, the decisions governing transitions between the states is handled by the Mission Planner. 43
Figure 4.1: Diagram showing the typical PIXHAWK wiring and companion sensors (ArduCopter, 2017). 50
Figure 4.2: The completely assembled quadrotor in flight. 51
Figure 4.3: CAD model representations of custom brackets designed, a) Buzzer & Safety Switch mount, b) PIXY camera mount, c) Nylon landing feet suitable for landing on a net. 52
Figure 4.4: Snapshot of the Quadrotor's CAD model showing the PIXY camera's field of view unobstructed by the LeddarOne rangefinder (green), body frame of landing feet. 52
Figure 4.5: a) Mode 2 R/C Transmitter stick configuration and b) the corresponding aircraft rotations (Benson, 2017). 53
Figure 4.6: Compass/Motor Interference test showing the motor current (green) causes minimal interference (red) to the compass. 54
Figure 4.7: PID values after minor adjustments of the Auto tuned quadcopter. 54
Figure 4.8: Plots showing the effect of the PID tuning. Top, the underdamped pitch angles have not been tuned yet, while bottom, the tuned roll angles closely track the desired roll. 55
Figure 4.9: Total Current draw during four consecutive flights. 56
Figure 4.10. Proposed design of the 1m^2 Landing Pad on the TopCat vessel, showing the view from the side and from on top. 58
Figure 4.11: A Selection of the ROS Topics and Services available from the MAVROS package. 59

Figure 4.12: Simplified Transformation (left) between the vessel's camera frame and the corresponding downward looking camera frame fixed to the drone's body (Prasetyono, 2017). Right, the complex transformation possible in 6DOF. 61
Figure 4.13: Two autonomous take-offs in mild winds, demonstrating self-recovery of altitude. 63
Figure 4.14: Flight data showing two consecutive precision landings on a one square meter target. The altitude during flight and indication of whether the target can be seen (top), the manual throttle input (top-middle), the angular offsets during LOITER and LAND mode (bottom-middle), and bottom, the horizontal velocities in the drone's camera frame due to the angular corrections. 65
Figure 4.15: Quadcopter following a target moving at walking pace in Precision Loiter mode using the upward-looking camera and control software. Note, the minimal R/C roll and pitch corrections (bottom) and the stable altitude hold in Loiter mode (top). 67
Figure 4.16: The field trial landing deck of dimensions 1.1m by 0.9m. 68
Figure 4.17: Precision landing on a stationary net platform in mild winds. 69
Figure 5.1: Progress update for the Software Control of the Quadcopter. 73

LIST OF TABLES

TABLE 3.1: Comparison of available Rangefinders in the market. 31
TABLE 3.2: Refined Specifications of Essential Drone Components. 36
TABLE 3.3: Additional Drone Components Required. 36
TABLE 4.1: Final Drone Components Purchased. 47
TABLE 4.2: Final Drone Payload Specifications. 48
TABLE 4.3: Additional Components Procured. 48
TABLE 4.4: Actual Specifications of the Drone. 57
TABLE 4.5: Summary of Precision Landing Flight tests. 66
TABLE 5.1: Estimated work status given the relative complexity of each state. 73

NOTATION

AHRS - Altitude Heading Reference System

ASV – Autonomous Surface Vessel

ARF – Almost Ready to fly

AUW - All Up Weight

CoG - Centre of Gravity

BEC - Battery Eliminator Circuit

DoF – Degree of Freedom

DSP – Digital Signal Processing

ESC - Electronic Speed Controller

FOV – Field of View

FPV – First Person View

FCU – Flight Controller Unit

GCS – Ground Control Station

GPS – Global Positioning System

IMU – Inertial Measurement Unit

LiDAR - Light Detection and Ranging

LiPo - Lithium-ion Polymer

MAV – Micro Aerial Vehicle

MAVLink - Micro Air Vehicle Communication Protocol

OSD – On Screen Display – of telemetry information

OTS - Off-the-Shelf

PID – Proportional Integration, Derivative Controller

PNF – Plug and Fly

RTF – Ready-to-Fly

RTK – Real Time Kinematic

RTCM - Radio Technical Commission for Maritime Services

RTL - Return-to-Launch

UAV – Unmanned Aerial Vehicle

UART – Universal Asynchronous Receiver Transmitter Communications

VTOL - Vertical Take-Off and Land

1. INTRODUCTION

The trend to develop Autonomous Surface Vessels (ASVs) for complex or dangerous marine operations is being driven by the rapid enhancement in guidance, navigation and control capabilities (Liu et al., 2016). They will enable missions such as monitoring of remote marine environments and charting of flood-damaged waters or defence shorelines, where map data is scarce or out of date. In such locations, the vessel's ability to navigate itself through unfamiliar environments is not only essential for the mission's success, but is a tactical advantage (Weaver et al., 2013). Therefore, as part of the drive for complete autonomy, aerial vehicles are now being paired with ASVs to improve the vessel's sensing of the path ahead for better mission planning. Currently, ASVs, such as Flinders' own TopCat vessel, have been fitted with an array of sensors to estimate the vehicle's position, locate obstacles and measure environmental factors such as wave dynamics and wind speed (Sammut et al., 2016). These sensors include cameras, navigation radar and Light Detection and Ranging (LiDAR) sensors (Elkins et al., 2010). Although providing highly relevant spatiotemporal data in real-time, the sensors are limited by their short range, line-of-sight constraint and low viewpoint above the surface, hindering the vessel's ability to navigate through cluttered estuaries.

Several strategies have been tried to overcome these sensor shortcomings and provide more useful data for mission planning purposes. Boyd and Foody (2011) have reported on the developments in remote sensing, noting the improvements in the level of certainty in the datasets, and their representation with geographical information systems. Utilizing pre-captured terrain maps from satellite imaging, is valuable but often outdated, whilst pre-deploying aircraft is an effective but costly means of assisting with mission guidance (Pinto et al., 2014). However, these methods fail to provide the real-time data for dynamic navigation in harsh and unstructured marine environments that is relevant to the surface vessel's field of view (FOV).

Unmanned Aerial Vehicles (UAVs) are an ideal solution for overcoming this problem as they provide a deployable set of "forward-eyes" for the surface vessel, providing the aerial perspective that is necessary for navigation through unknown waterways (Djapic et al., 2015). A common form of UAVs, multirotor drones, have key limitations including restricted range, short flight times and increased susceptibility to wind disturbances. However, coupled with a surface vessel, the drone can be simply deployed and retrieved when needed, making long, remote operations possible and profitable (Djapic et al., 2015). Thus, providing Flinders' TopCat with a deployable, camera equipped drone will not only enhance the surface vessel's capability to complete long,

self-guided surveying missions, but also enable the surveying of neighbouring regions not accessible by the vessel.

Therefore, this report, and two related projects, are seeking to develop a drone that is capable of automatically launching from and landing on the TopCat vessel, under the guidance of Flinders' Centre of Maritime Engineering, Control and Imaging (CMECI). This report focusses on the selection of a suitable drone, the development of the UAV control software, and the overall system integration. The companion projects cover the mission planning and trajectory generation system (Swincer, 2017), and the vision-feedback guidance system (Prasetyono, 2017). The goal of these projects is to develop a prototype that will firstly, perform autonomous landing on a one square meter stationary platform, and if successful, then refine this design for a dynamically oscillating platform.

This project aims to design both the drone and overall controlling software in both a modular and scalable manner to give opportunity for future development of the research platform and enhancements of the task level algorithms. The design process will involve determining the necessary sensors and computing requirements for performing autonomous landing, leading to a preferred system synergy structure. This will guide the selection of a suitable drone and related sensors for both the surface vessel and drone. Both the drone and basic landing platform will be constructed while the software system is developed. Extensive testing will be performed both through simulation and real-time testing to validate the design. Lastly, the outcomes from the two companion projects will be incorporated into this system, for final implementation.

The remainder of this report is organized as follows:

- Section 2 will outline the extensive work done in the field of autonomous landing and describe the existing technology on the Flinders' TopCat vessel.
- Section 3 will detail the design process for the development of the drone & landing platform, the overall software co-operation and the nature of the UAV control software.
- In Section 4, the results of the development process are presented, while
- Section 5 will critically evaluate the overall performance of system.

The report concludes with a summary of the work completed and recommendations for future work.

2. LITERATURE REVIEW

The following sections present an assessment of the techniques used and proposed strategy for autonomous take-off and landing based on the extensive research conducted in the field. An overview of the relevant technology on the TopCat is followed by an assessment of the global research performed on autonomous landing. Detailed consideration will be given to the landing techniques, co-operative system composition and landing platform designs. These findings will be drawn together to recommend essential system level requirements and drone platform designs.

2.1. Overview of Current Surface Vessel's Design

Flinders University have developed their own ASV as both a competitor in the Maritime RobotX Challenges of 2014 and 2016 and a research platform (Sammut et al., 2016). The team is seeking to extend the capabilities of the vessel by developing an autonomous drone allied with the ASV, to acquire surveillance data for both mission planning and environmental monitoring purposes. The TopCat is built on the Wave Adaptive Modular Vessel (WAM-V) platform (WAM-V Marine Advanced Research Inc. 2017) and is designed for use in coastal waters or estuaries. Equipped with twin batteries and electric outboard motors, TopCat is capable of maintaining a cruising speed of four knots (eight km/hr) for up to five hours. The vessel is furnished with a set of navigation and object detection sensors that provide the on-board computers with the data to complete autonomous missions. The second version of the Flinders' TopCat vessel is shown in *Figure 2.1*, which features maximized sensor coverage and improved centre of gravity with the batteries mounted below the main platform. Its overall payload is 250 lbs, or 114 kg.

Figure 2.1: Flinders' Autonomous Surface Vessel, named TopCat V2. Picture © CMECI, 2016.

2.1.1. TopCat's Sensors

The TopCat uses a range of sensors to determine its location and sense its immediate environment. The main navigation sensor is the Trimble BX982 Global Positioning System (GPS) receiver, which, when coupled with dual GPS antennae, provides centimetre accurate position estimates and heading information to an accuracy of less than a tenth of a degree (Trimble, 2017). This GPS unit is capable of producing Real Time Kinematic (RTK) baselines in conjunction with the Trimble R10 base station to improve the accuracy of the estimates as well as providing corrections to the smaller GPS sensor on the drone. These corrections follow the Radio Technical Commission for Maritime Services (RTCM) standards. Additionally, the Trimble GPS provides an estimate of the roll of the vessel and its heading rate, in partnership with the Microstrain Altitude Heading Reference System (AHRS).

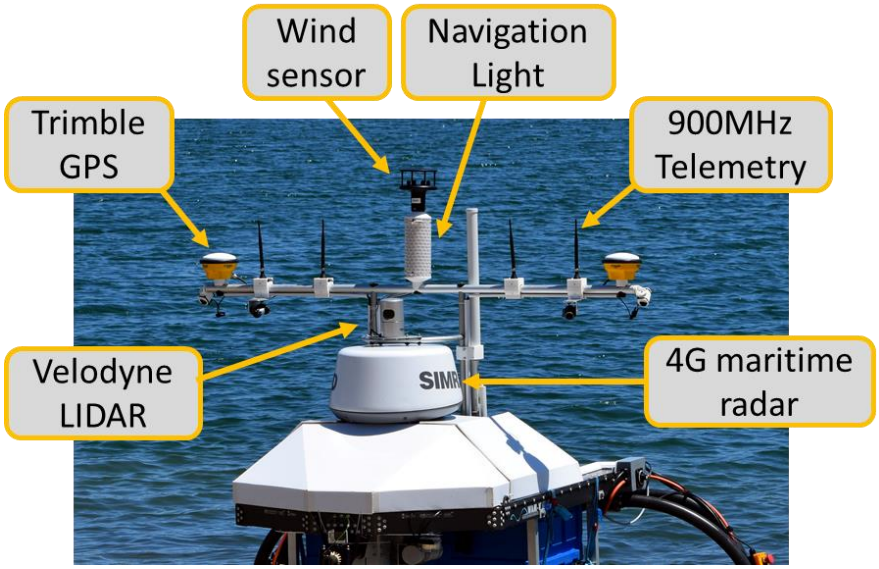


Figure 2.2: Overview of TopCat V2's sensor array. Picture © CMECI, 2016.

The primary sensors for detection of obstacles are the Simrad radar and the Velodyne LiDAR. The Simrad 4G maritime radar is a low power, reliable sensor, having been proven in the 2014 RobotX Maritime Challenge, and has a range up to 36 metres. The Velodyne HDL-32 LiDAR has 360° view, providing the vessel with a precise location and the information about the physical features of surrounding objects. This sensor functions by scanning the environment in horizontal layers, and detecting the reflections from nearby objects. *Figure 2.3* shows a sample of the LiDAR sensor data, where red indicates close objects while blue indicates objects further afield. From field tests, it is estimated that an object less than 200mm in height could fly horizontally between these layers without being detected until in close range. Consequently, to overcome this limitation and

capitalize on the LiDAR's precision, the approach path of the landing drone will be designed to maximize its visibility to the Velodyne LiDAR.

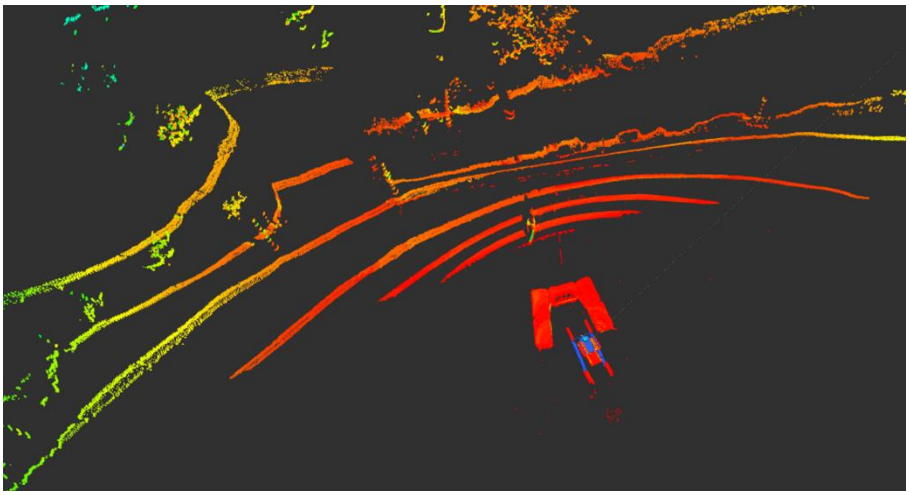


Figure 2.3: Visualized data from the Velodyne LiDAR. Picture © CMECI, 2016.

Other sensors available on the TopCat of interest to this project are the vessel's Inertia Measurement Unit (IMU) for providing yaw, pitch and roll data, and the Windsonic acoustic wind sensor mounted on the top of the mast. As the vessel has a shallow draft and stands more than a metre tall, it is very susceptible to wind disturbances, and therefore uses the wind sensor for drift compensation. Additionally, this data will be used to determine whether the wind conditions are conducive to the deployment of the drone. The vessel is also equipped with an array of four forward looking Microsoft LifeCam Studio web cameras for computer vision, but these have been angled downwards for a clearer view ahead. In addition, the drone's approach path is planned to be over the aft of the vessel, following long-standing naval recommendations (Djapic et al., 2015).

2.1.2. TopCat's Neural Capabilities

The TopCat is powered by a range of computing capabilities, ranging from low level actuator control to mission planning. The heart of the boat is the I7 mini ITX computer that has 16GB RAM and 250GB SSD. This is supported by the Digilent Max32 PIC based microcontroller and custom header board for control of the vessel's teleoperation, emergency stop, and CAN communication links. Additional hardware is also present for power distribution, motor control and underwater sensor data processing, but these are not relevant in the context of this project.

The software is built in Linux, using the open-source Robot Operating System (Kanellakis & Nikolakopoulos) that features a wide range of feature libraries and facilitates a modular development approach that is crucial for complex systems. The individual tasks have been organized into distinct programs, called nodes that can publish data or subscribe to ROS topics, for sharing of information. The software has been organized into three layers, namely mission planning, navigation and guidance, and hardware control, where only the top two layers use ROS. The mission planning level determines the mission strategy, the order of tasks and gives permission to individual task planners when sufficient information is available.

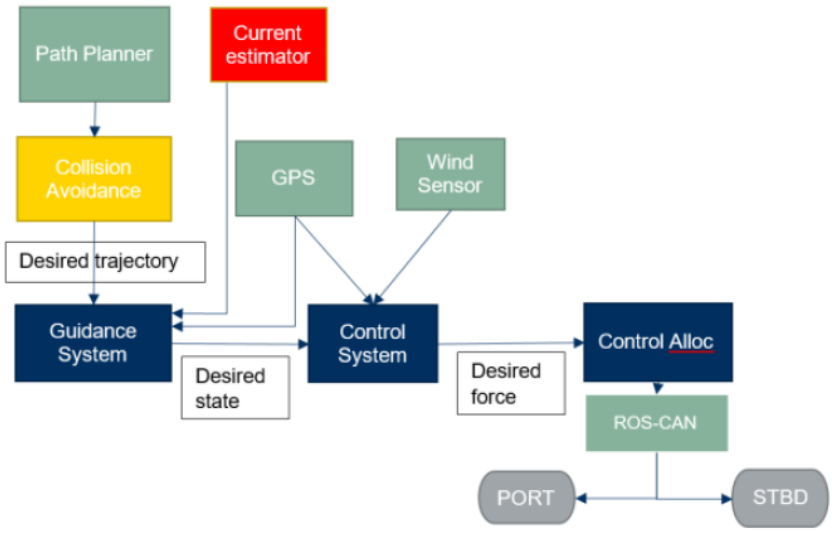


Figure 2.4: Graphical representation of the Navigation and Guidance structure, where blue represents ROS nodes, green signifies existing sensors, and yellow and red indicate future work (Sammut et al.).

The navigation and guidance level is organized into a number of nodes to simplify the complex task, and is visualized in *Figure 2.4*. The instructions from the path planner will be amended based on obstacles detected and the desired trajectory feeds into the guidance system. This system uses state-based estimation to keep the vessel on course and along with the control system, combines

several data sources to control the vessel's motors. A similar guidance and tracking system approach will be used when controlling and guiding the drone through take-off and landing.

2.1.3. TopCat's Communication

The design of the TopCat's communication system seeks to build in redundancy and fail-safe features into the vessel. The on-board electronics use Ethernet, Universal Serial Bus (USB) and Controlled Area Network (CAN) bus for communications with the host computer, battery and engine control modules. Four 900MHz radio modems are used for crucial communication with the base station and for providing teleoperation data. These modems have been tested and found to operate within Australia's allocated frequency range for the Industrial Scientific Medical (ISM) band (Sammut et al., 2016). The protocol TopCat uses for communicating over these modems is the Micro Air Vehicle Communication Protocol (MAVLink) that is a highly efficient, header-only message marshalling library specifically designed for micro air vehicles (MAVLink GCS, 2017). Furthermore, the flight controller of choice supports modems of the same frequency range and communication protocol, making the pairing of the drone with the vessel simpler. Lastly, the TopCat is equipped with a point-to-point 5GHz Wifi link for high data transfer communications where high Bandwidth is required such as streaming live video back to the base station.

2.2. Globally: Autonomous Landing of Unmanned Aerial Vehicles

2.2.1. Autonomous Co-operation between Unmanned Vehicles

The application of an aerial view has been used extensively for autonomous navigation, including guiding autonomous ground vehicles past obstacles (Weaver et al., 2013), and using ceiling mounted eye-bots to oversee teams of foot-bots navigating across wide clearings (Mathews et al., 2010). Research trends indicate growing support for this technique, suggesting the use of ASVs, UAVs and other vehicles working together as a means of overcoming the limitations of the constituent robots and thereby achieving higher efficiency (Djapic et al., 2015; Herissé et al., 2012; Mathews et al., 2010; Pinto et al., 2014; Weaver et al., 2013). This is commonly referred to in literature as co-operative autonomy of heterogeneous systems (Djapic et al., 2015; Weaver et al., 2013).

2.2.2. Autonomous Landing of UAVs

The challenge of autonomously landing an aircraft on a moving vessel has been accomplished earlier by the US Navy in 2013, using the Northrop Grumman's X-47B drone on an aircraft carrier (Weaver et al., 2013). However, this cost one billion dollars, and required the use of the whole vessel's airstrip to land the fixed-wing drone. Several research teams have, therefore, cheaply implemented autonomous landing through different techniques, using quadrotor drones, because quadrotors boast controlled Vertical Take-Off and Landing (VTOL), small all-up weight (AUW) and excellent agility during flight. These have included tracking and landing on self-directed platforms on a flat floor in indoor environments (Lee et al., 2012; Meier et al., 2012; Wenzel et al., 2011).

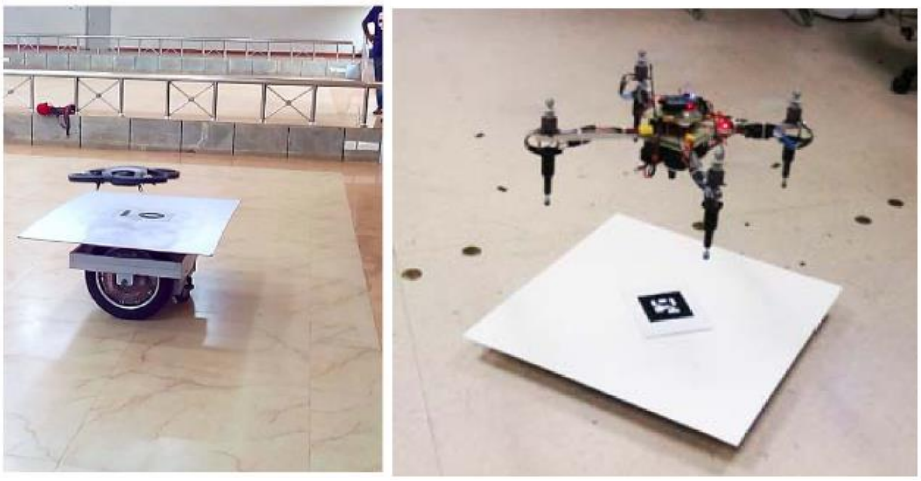


Figure 2.5: Quadrotor taking off from a moving platform (Ajmera et al., 2015) and tracking and landing on a platform identified by a unique AR tag (Lee et al., 2012).

Meier et al. (2012) and Chandra et al. (2016) demonstrated autonomous flight using a series of AR tags and computer vision linked with the reliable, open-source autopilot, PIXHAWK, which is the flight controller of choice for this project (Cai et al., 2014). Lee et al. (2012) implemented the autonomous landing using a low computation cost, image-based control only, using a two-dimensional image space to land on an AR code fastened to the platform. However, from observing the footage, the control system does not appear to handle wind disturbances robustly. Wenzel et al. (2011) also implemented a visual tracking approach for autonomous landing, using infra-red (IR) lights and a Wii remote IR camera, with a 90% success rate indoors. However, it is well known this will not function in direct sunlight due to the abundance of natural IR radiation. Herissé et al. (2012) designed and successfully demonstrated a control system that landed a quadrotor on an indoors, vertically oscillating platform that pitched similar to that of a boat's landing platform. In India, a team from the National Institute of Technology completed autonomous tracking and landing of a quadrotor on a helipad in a gymnasium, using only a downwards looking camera and vision processing performed off-board at the base station (Ajmera et al., 2015).

2.2.3. Autonomous Landing of UAVs on moving platforms, outdoors

In the outdoor environment, the goal of co-operative autonomy between the surface vessel and aerial vehicle can only be reached if the drone can reliably take-off from and land on the moving vessel (Herissé et al., 2012). This task has several challenges as the drone is vulnerable to wind gusts, the landing platform is moving in six degrees of freedom due to wave motion, and the vessel could be operating in environments where GPS is not available (Djapic et al., 2015; Gautam et al., 2014; Weaver et al., 2013). It must be noted that, compared with the US Navy ship, an autonomous marine vessel has a much smaller mass than that of an aircraft carrier, and therefore experiences far greater instability from the wave disturbances, making the landing on the unsteady platform more difficult. Kim et al. (2014) devised an innovative solution in outdoor environments, incorporating a smartphone attached to the underside of the drone to track a brightly coloured, moving platform in outdoors environment, as shown below. Here, the turbulences due to the outdoor environment were compensated for by an adaptive control scheme on-board the flight controller (Kim et al., 2014).

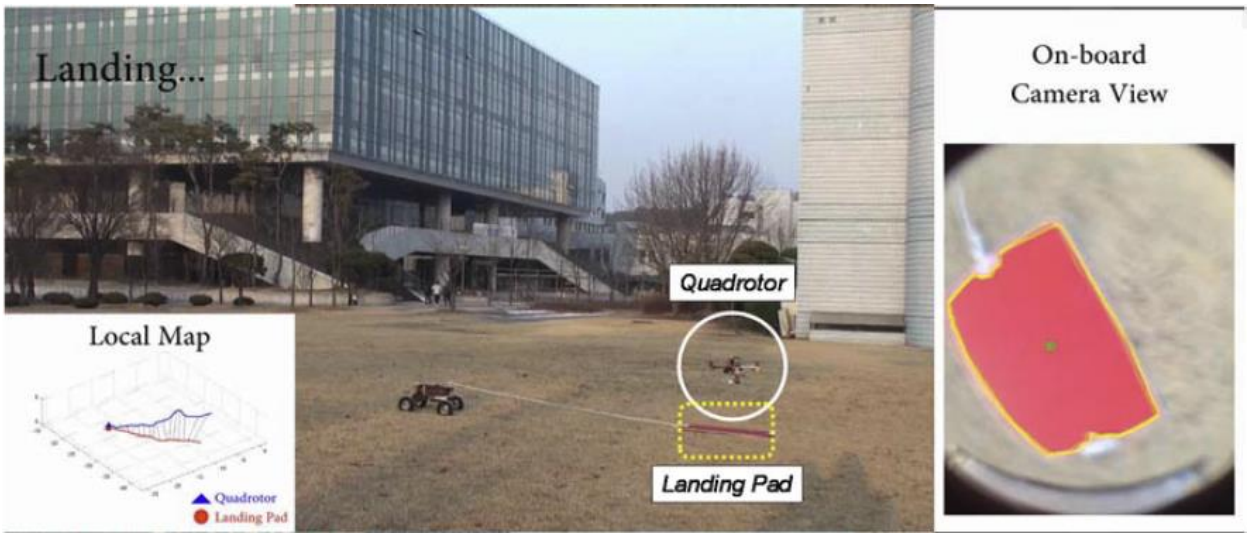


Figure 2.6: Image of the quadrotor completing its landing on the moving platform (Kim et al., 2014).

In the marine environment, Weaver et al. (2013) and Pinto et al. (2014) achieved autonomous launching and recovery missions from autonomous marine vessels, using quadrotor drones, while Djapic et al. (2015) used an autonomous helicopter to demonstrate the advantages of the aerial perspective in both environmental mapping and developing navigation cost maps for the vessel's mission planning. These key projects will be discussed in further detail in the following sections.

2.3. Lessons from Similar Implementations in Marine Environments

The following section describes in detail the strategies employed to achieve autonomous deployment from and onto moving platforms, particularly in the marine environment, and highlights the features that can be adapted to this study.

2.3.1. Take-off Strategy

Autonomous launching of an UAV is comparatively simple compared with the task of autonomous landing. It simply involves ascending quickly to a safe altitude while avoiding the known obstacles on the surface vessel, and then perform stabilization and correction to any issues encountered with trajectory following (Pinto et al., 2014). Djapic et al. (2015) emphasizes the need to check the wind conditions are favourable for flying before the aerial vehicle launches, which is critical as the drone's thrust specifications dictate the magnitude of wind disturbance it can tolerate. Additionally, the data from the vessel's Inertial Measurement Unit (IMU) can give an indication if the surf is too rough before attempting the complex drone deployment. The Californian team also implemented an unlatching sequence, involving an initial downwards thrust, to disconnect their SSC PAC Vapor 55 helicopter from its spring loaded catch. As the mass of this project's drone is much smaller than the helicopter's, a much simpler hold-down mechanism is being considered.

2.3.2. Landing Strategy

The landing strategy can be broadly summarized by the following three tasks (Djapic et al., 2015):

i. Trajectory following – involving geographical position, range and altitude calculations,
ii. Detection and Tracking of the ASV landing pad – primarily using vision, and
iii. Localization and Touchdown – requiring blending of vision, sonar, and other sensor data.

The first two tasks are described in detail in the companion projects of Swincer (2017) and Prasetyono (2017), respectively, while this project focusses on providing the software structure for their work, implementing the data fusion algorithms and assembling the control commands for the drone. Djapic et al. (2015) also highlighted the importance of these tasks being run in parallel, and all contributing to a common set of data for successful and robust landing, as illustrated in *Figure 2.7*. The following sections break down the landing task into smaller sections and describe the techniques that have been used successfully at each stage.

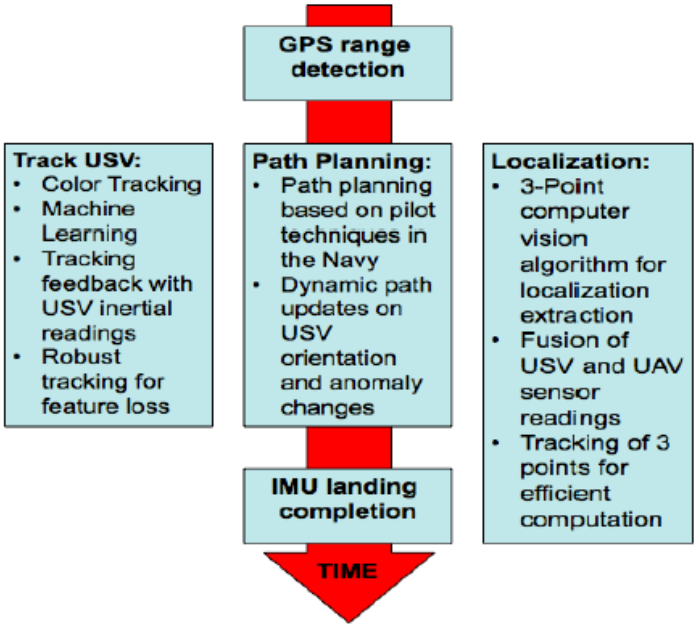


Figure 2.7: Time Domain representation of the sequence of tasks required for autonomous landing. Each task, though independent, needs to be run in parallel, sharing its data with the other tasks (Djapic et al., 2015).

2.3.2.1. Approaching the Surface Vessel

The flight path that the UAV follows is typically defined by a series of GPS waypoints that are sent to the flight controller of the drone (Weaver et al., 2013). Therefore, GPS tracking is used to bring the drone within sight of the surface vessel. The location estimation provided by the GPS modules can provide an estimate within two meters accuracy while moving. On their marsupial robot team called RIVERWATCH, Pinto et al. (2014) used a GPS-RTK Proflex 800 supplied by Ashtec SAS, claiming 2 cm horizontal accuracy. Similarly Djapic et al. (2015), finding that the existing GPS unit on their Vapor 55 UAV was insufficient, were looking to incorporate the Piksi RTK GPS units which boast similar accuracies. However, this accuracy is only possible when the tracked object has remained stationary for some time. It is for this reason that when the drone approaches within thirty meters of the surface vessel, more accurate, real-time position feedback sensors have been used in addition to the GPS units.

2.3.2.2. Identifying and Tracking the Landing Platform

A large number of studies have been dedicated to the identifying and tracking a landing platforms, with the results summarized in literature reviews such as Kanellakis and Nikolakopoulos (2017). This review focuses on the prevailing approach of using vision based feedback for position-altitude control, pose estimation, obstacle detection and target tracking. Importantly, it provides a

thorough review of each of the components identified in the system level structure below, and concludes that a large number of the techniques for visual-servoing, object tracking and Simultaneous Location and Mapping (SLAM) are experimental only and application specific (Kanellakis & Nikolakopoulos, 2017). However, it does highlight the challenge of achieving the level of on-board computer processing that is needed for short latency in the control loops, due to the limited payloads of the drones.

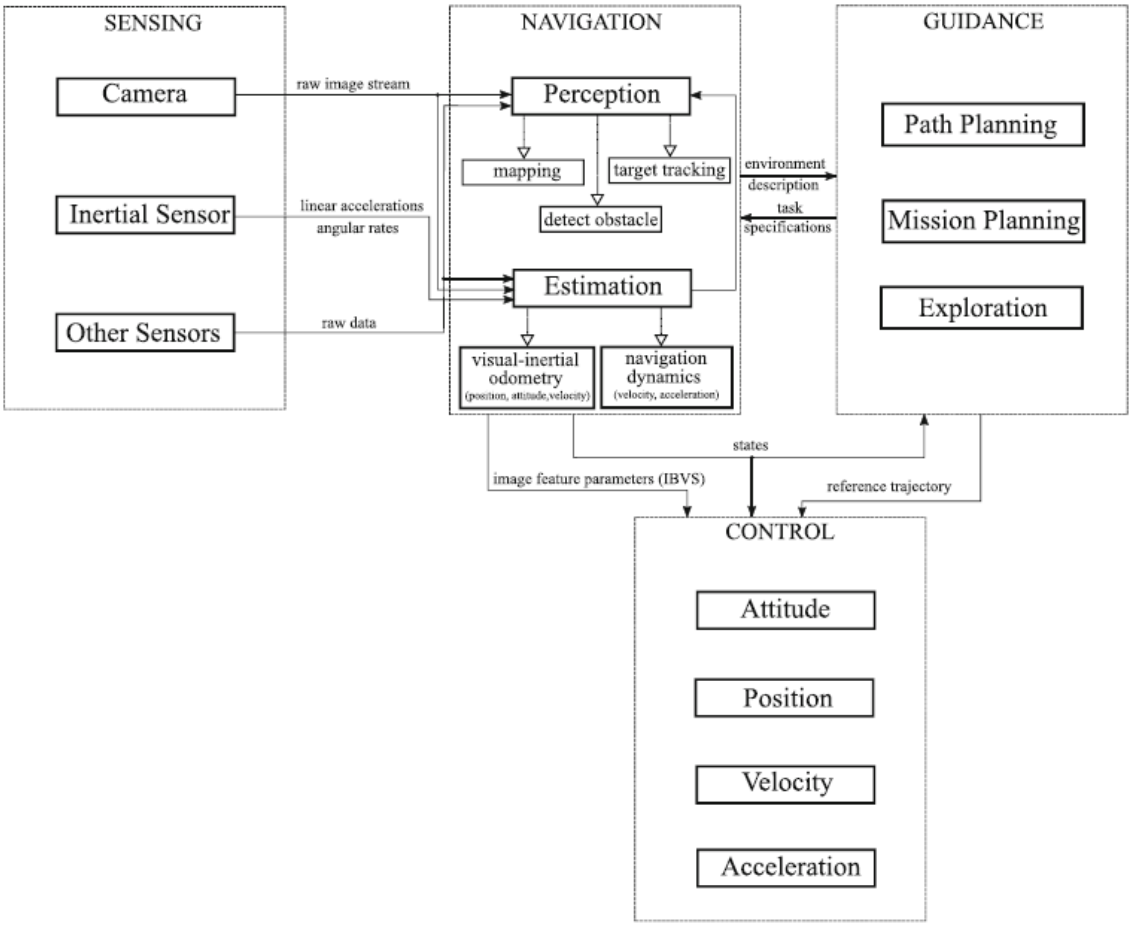


Figure 2.8: System level structure for performing autonomous landing on the moving platform (Kanellakis & Nikolakopoulos, 2017).

Additionally, Gautam et al. (2014) conducted a survey on autonomous landing techniques for a range of drone types, comparing both GPS guided and vision-based landing techniques. They also recommend vision-based landing since they have been verified on real systems, but robust controls need to be in place to handle image occlusion and fast changing image intensity, especially when close to landing (Gautam et al., 2014). This justifies the use of vision-based tracking for the following three marine applications.

Weaver et al. (2013) developed a compact design, shown in *Figure 2.10*, using a downward looking camera coupled with a powerful ODROID-U2 processor to provide on-board target detection, tracking and system control. This processor managed communications with the surface vessel and fed the control commands to the flight controller to manage the navigation and flying.

Pinto et al. (2014)'s RIVERWATCH also used the ODROID-U2 processor on-board their drone and camera for surveillance, but used an upward looking camera from the ASV's landing platform, for a more stable background against the sky, and to make use of the abundant computing power on the vessel. The vessel's heading and computed pose estimation of the UAV was then sent back to the UAV for control. However, this introduces significant latency into the vision-feedback control system.

Djapic et al. (2015) developed a robust design, using a camera on both the ASV and UAV, and fusing the data from both to create complete picture of the two vehicles. The upward looking stereo camera in the landing platform provided a pose estimate of the UAV relative to the vessel, once the UAV entered its FOV. This information was used to build a state representing the position of the drone, while the flight controller moved the UAV closer to the vessel by seeking to enlarge the object of interest in its downward looking camera. Concurrently, the ASV would orient itself into the wind, and travel at a constant speed, following Naval best practise. At a high level, the Flight Path Planner uses the UAV's state information and flight curvature limitations to derive an optimum descent path to the boat. This path is constantly adjusted to maintain clear view of the three points on the landing platform, as discussed in the next section.

2.3.2.3. Last five meters

Once within close range of the boat, the orthogonal landing pad alignment indicators were clearly visible, so Djapic et al. (2015) used a three point algorithm to derive a position estimate of the boat, while using the cross product of these points to verify the boat's rotation, as provided from its IMU. An Extended Kalman Filter (EKF) is commonly used for robust tracking during this stage, to recover from temporary image occlusion, or when the drone has lost sight of the platform (Ajmera et al., 2015; Djapic et al., 2015; Pinto et al., 2014; Weaver et al., 2013). A block diagram of the tracking process is shown in *Figure 2.9*, and for more detail, refer to Prasetyono (2017)'s report on Visual Feedback. Together, an overall system model was developed and constantly updated through closed loop feedback, reducing the error margin between the UAV and ASV to zero. The above process is illustrated in the flow diagram shown earlier in *Figure 2.7*.

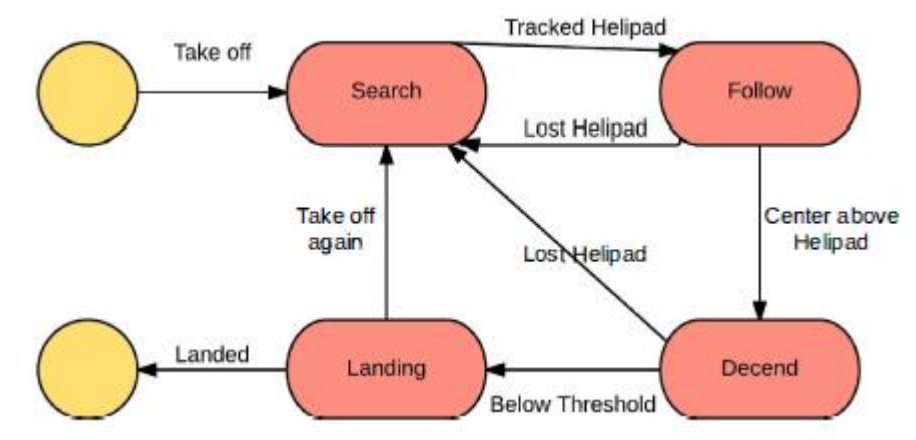


Figure 2.9: State Diagram of the Tracking process of the landing pad (Ajmera et al., 2015).

2.3.2.4. Last half a meter

The final approach to landing is made difficult by the rolling and pitching of the vessel's platform due to the sea motion. Extensive research has sought to model the boat's behaviour, both for landing military helicopters on aircraft carriers (Chartier, 2017), and for the more dynamic movement of smaller vessels. Predictive modelling of the low frequency vertical movement of the landing platform was conducted by Ajmera et al. (2015) and Yang et al. (2008). Herissé et al. (2012) demonstrated maintaining a safe distance from and landing on a rising and falling platform, indoors, using optical flow imaging. However, Herissé et al. (2012)'s modelling assumed the vessel's velocity was stationary, and the practical results revealed that significant latency, due to the off-board processing, and therefore recommended a feed-forward compensation approach.

A common approach is to use a couple of layers of Proportional, Integration, Derivative (PID) controllers, where the first layer aims to reduce the distance between the UAV and ASV to zero, while the second layer matches the yaw, pitch and roll of the drone, with that of the surface vessel (Djapic et al., 2015; Pinto et al., 2014; Weaver et al., 2013). Pinto et al. (2014) used four independent PID controllers to control the latitude, longitude, altitude and heading degrees of freedom.

At the final stage of landing, Djapic et al. (2015) proposed using the ASV's LIDAR to give an accurate reading of the landing gear's offset from the landing platform, for graceful touchdown. Despite the robustness of their design, Djapic et al. (2015), reported difficulties in reliably landing the helicopter in the centre of the platform, noting its susceptible to wind disturbances. In future they anticipate using some mechanical alignment mechanism to overcome this issue. Gautam et al. (2014)'s review also recognized the shortcomings of GPS and vision guidance, recommending the use of a dedicated altimeter to assist in the final stages of landing.

2.3.3. Landing Platform Structure Design

The landing pad design comprises two key components, namely the structural design of the platform, and the unique visual pattern for identifying the platform. This report focuses on the former component, and for details on the latter, see Prasetyono (2017)'s report as this is highly dependent on the nature of the Vision-Feedback system. The landing platforms from the three marine based implementations are shown below. Weaver et al. (2013) designed the platform for the 450mm diagonal, width drone with high contrast colour boundaries for easier thresholding in the image processing. The conical platform used by Djapic et al. (2015) was for a much larger helicopter.

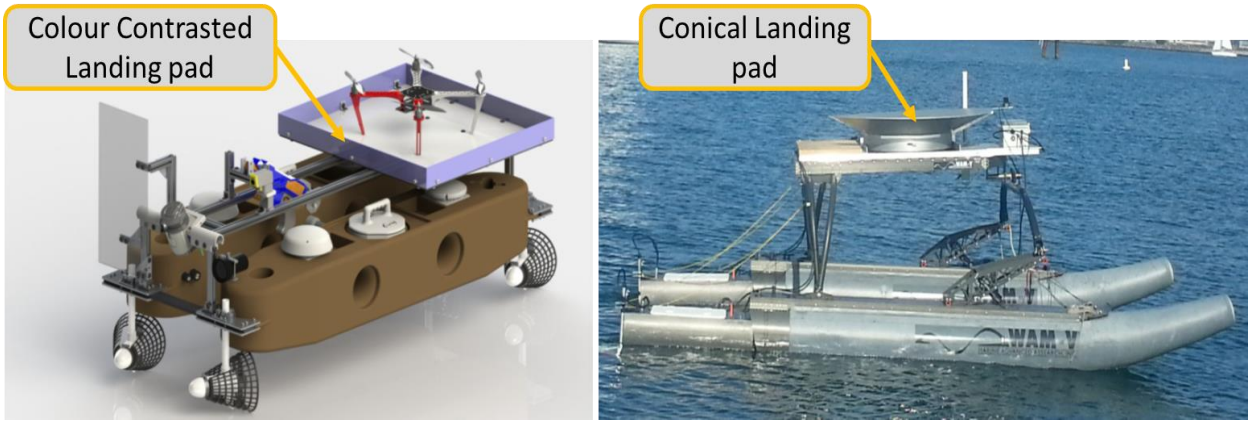


Figure 2.10: Left: CAD model of ASV & UAV (Weaver et al., 2013), Right: Conical metal platform for Vapor 55 Helicopter (Djapic et al., 2015).

Figure 2.11: Left: RIVERWATCH team of ASV and UAV, Right: showing the 1m by 1.3mm landing net centre marking pattern (Pinto et al., 2014) & (The Three Laws. 2017).

The RIVERWATCH team used a rubber texture net for the broad 1 m x 1.3 m platform, with angled sides to prevent the drone from sliding off. The drone is only held in place by the friction between the landing gear and the platform, but Pinto et al. (2014) concedes that this is not sufficient in high seas. The range of landing variations during a number of tests is shown below in *Figure 2.12*.

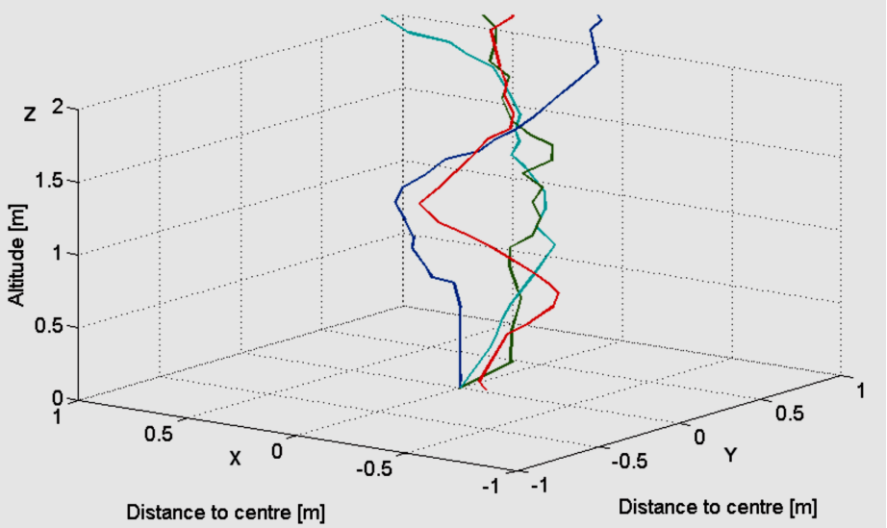


Figure 2.12: The trajectory paths of different landing tests by the drone onto the landing pad (Pinto et al., 2014).

Although the lock-down mechanism for the drone is not strictly within the scope of this project, the design of the landing platform and drone's landing gear are, and so consideration of potential lock-down mechanisms are factors influencing these designs. The plan for Flinders' TopCat vessel, is to use the available real-estate to the rear of the sensors rail for the landing platform, with the potential to expand this a future 0.5 m toward the vessel's aft, if needed. The vessel's remaining payload available is approximately 10-15 kg, so the landing platform of choice will need to be lightweight.

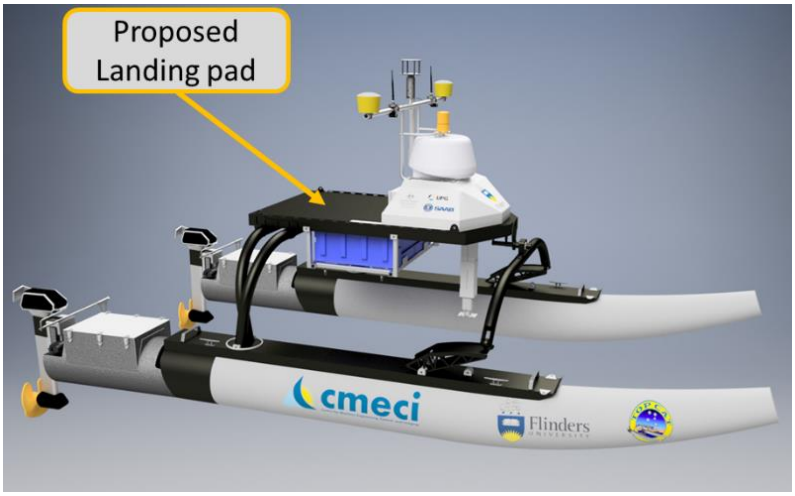


Figure 2.13: Complete CAD Model of TopCat V2, showing the clear space for landing pad. Picture © Flinders CMECI, 2016

2.4. Software: System Integration

The following section describes the system configurations used to achieve autonomous landing on moving platforms, particularly highlighting the sensors and software structure used, and concludes with recommendations for the preferred system structure.

2.4.1. Sensors

The rapid enhancements in technology have enabled the fusion from a range of sensors' data to create robust and capable systems (Djapic et al., 2015; Elkins et al., 2010). In their extensive review of computer vision for UAVs, Kanellakis and Nikolakopoulos (2017) notes the prevalent use of ultrasonics for obstacle avoidance, laser range finders for environmental surveying and range finding, and camera vision for depth sensing. In his review of autonomous landing strategies for UAVs, Gautam et al. (2014) identified that there was no single sensor that was ideal for autonomous landing, but recommended the use of multiple sensors, controlled by a robust controller. Vision tracking systems suffer from blindness when too close to the platform, as the pattern is outside the camera's FOV, or the aircraft shades the platform (Gautam et al., 2014). Also, GPS height readings are inaccurate for fine level control, and hence Gautam et al. (2014) suggested the use of laser range finders as dedicated altimeters.

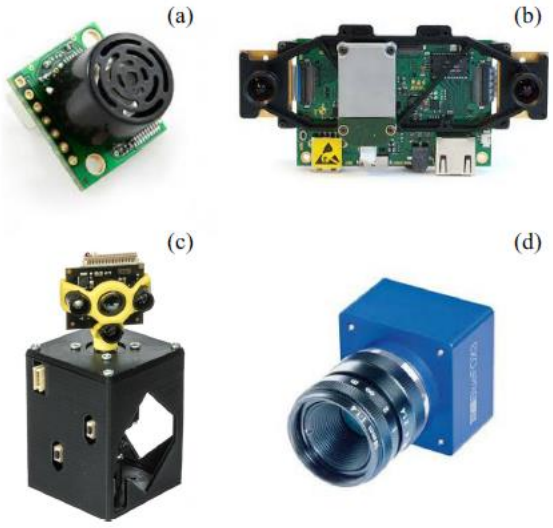


Figure 2.14: Common UAV sensors: a) Ultrasonic sensor, b) Visual stereo, c) Laser Range finder, d) monocular camera (Kanellakis & Nikolakopoulos, 2017).

Another critical sensor is the IMU that is built into modern flight controllers, providing feedback about acceleration and rotation rates, at approximately 100Hz rate. This has been combined with computer vision successfully to achieve autonomous flight in many applications (Kanellakis & Nikolakopoulos, 2017). For vision tracking, Djapic et al. (2015) used a single upward looking

camera (Leopard Imaging Video Camera) but anticipates using a stereo camera pair on the drone for better distance calculations. Weaver et al. (2013) used the Logitech HD Pro Webcam C920 camera and processed the images using the ODROID-U2 on-board, while Kim et al. (2014) used the omnidirectional camera on a smartphone, making use of its processor to compute the necessary guidance corrections for the flight controller. Herissé et al. (2012) and Kendoul et al. (2009) demonstrated successful tracking and landing using optical flow camera but with low response times due to off-board processing for optical-flow data.

The following implementations used the PIXHAWK flight controller with a suite of sensors, to achieve autonomous landing. Lee et al. (2015) developed their earlier quadrotor design (Lee et al., 2012) to enable autonomous landing within a 10 cm margin for automatic battery swapping applications. For vision guidance, they used the PIXY (CMUcam5) camera, equipped with a high-powered processor with proprietary software for colour-code detection and tracking. Other sensors included a LiDAR sensor, Ublox GPS and ultrasonic rangefinder, used in conjunction with the inbuilt magnetometer and barometer on the PIXHAWK. Additional on-board processing was provided by the Beagle Bone Black board. A Massachusetts team also used the PIXHAWK flight controller when attempting autonomous landing on research vessels, because of its open source hardware/software and its ability to reject minor environmental disturbances (Chandra et al., 2016). Their system included a gimbal mounted GoPro camera as shown in *Figure 2.15*, with all processing performed on the vessel's computer.

Figure 2.15:Key system components for autonomous landing, showing the two main communication links (Chandra et al., 2016).

2.4.2. Computational Load Constraints

The main computationally heavy algorithms are the vision-based tracking algorithms and the state-based pose estimators such as the SLAM or EKF algorithms. These have been overcome by performing the high level processing on the boat or by putting a low mass, digital signal processor (DSP) on the drone. Meier et al. (2012) implemented the latter technique in the PIXHAWK MAV system shown below, and a picture of the drone is shown in *Figure 2.19*.

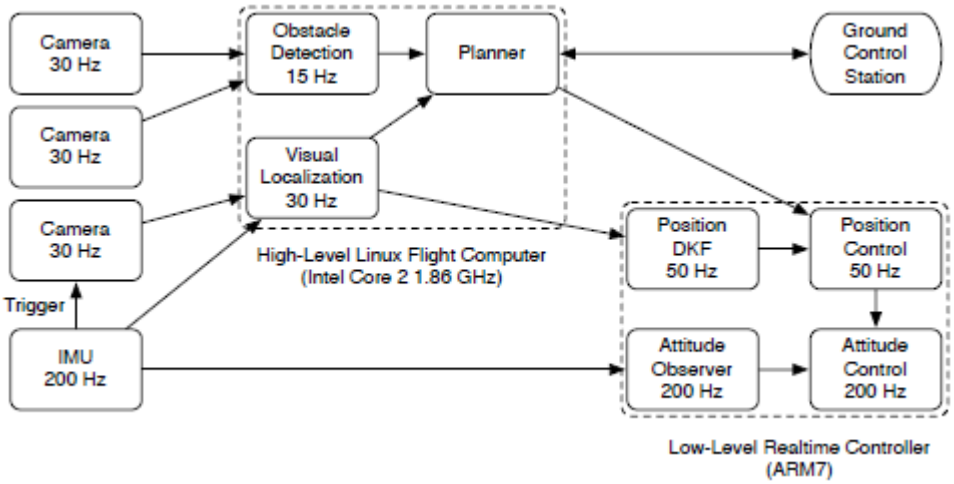


Figure 2.16: The PIXHAWK MAV system diagram, showing visual tracking, obstacle avoidance and path planning on a dedicated computer, feeding the low-level flight controller. (Meier et al., 2012)

Weaver et al. (2013) developed a similar, modular software design using the quad-core ARM ODROID-U2 processor on the drone, running Linux-Ubuntu and ROS. Coupled with the Logitech Webcam camera, this processor provided compact, high speed image processing on-board the drone for target detection and tracking. This processor also managed the trajectory planning, control commands and communications, leaving the surface vessel with the supervisory role of mission start, status and completion commands, and reporting the vessel's position and desired GPS waypoints.

Ajmera et al. (2015) and Herissé et al. (2012) perform the image processing and filtering calculations off the drone, at the base station, decreasing the drone's payload and offering the advantage of greater processing power, but at a cost of restricted autonomy, particularly in operating range (Wenzel et al., 2011). Pinto et al. (2014) also made use of the three, water cooled, i7-3770 Ivy Bridge systems on the ASV, even though they had an ODROID-U2 on the drone. This is because the environmental mapping required significant processing power. As the ultimate goal for this project is for the drone and surface vessel to operate in tandem without reliance on a base

station, the computationally heavy processing will need to be performed on the ASV, not at the base station.

The prevailing software system used was the Open Source ROS, on a Linux platform, the same configuration as Flinders' ASV (Ajmera et al., 2015; Kendoul et al., 2009; Pinto et al., 2014; Weaver et al., 2013). This platform has the advantages of compatibility with image processing libraries such as OpenCV, Point-Cloud-Library (PCL), excellent simulation tools such as ROS *vis* and Gazebo, and all the data is available in ROS *bags* for future simulation testing. Additionally, Weaver et al. (2013) provides a thorough explanation of the use of ROS nodes to design a modular system, and provides the high level diagram, shown here, illustrating the sharing of information between the ROS masters of each autonomous vehicle. The ROS core publishes its data to a topic that is visible to the ROS *bridge*, and when the ROS *bridge* observes an update, it notifies the other ROS core through updating its ROS *bridge*.

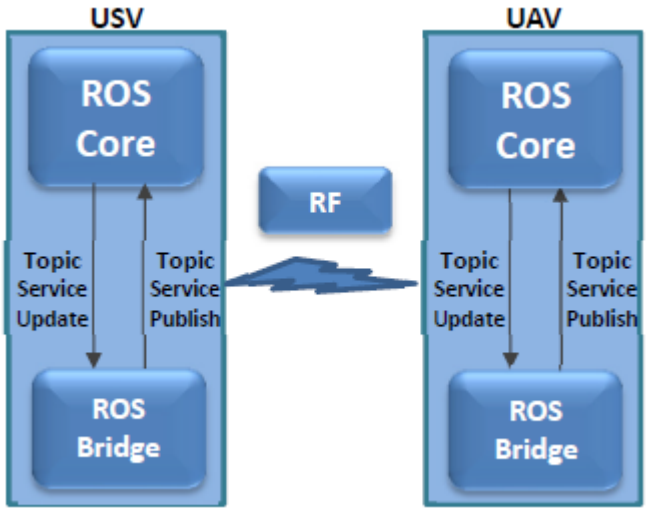


Figure 2.17: Block Diagram representation of the ROS bridge (Weaver et al., 2013).

Lastly, Chandra et al. (2016) used ROS, with the MAVLink communications protocol, to control the PIXHAWK flight controller during flight. Using the ROS *mavros* package, the PIXHAWK was set in "Guided Mode" to follow the GPS waypoints sent, until the fiducial marks on the platform were visible. Once overhead, "Loiter Mode" was selected that held the drone's position until commanded to move by the control points. ROS's *ARpose* package was used to determine a 3D position estimate of the drone's position relative to the platform from the Augmented Reality (AR) tags in the landing platform. The image feature extraction process was assisted by the ROS *img—proc* and *usb-cam* packages. The above techniques and structure will form the basis for the overall software system design for this project.

2.4.3. Communication Links

The communication links comprised primarily a critical communications link between the vessel and the drone, and a separate high bandwidth link to transmit live-video feeds to the boat or base station. Weaver et al. (2013) used the popular XBee RF modems (Cai et al., 2014), operating over a range of 300ft (91 m) to communicate between the ASV running ROS (*Mission Planner* node) and the UAV's ROS *core*. Pinto et al. (2014) also used a similar approach using XBee Pro & Ubiquiti Networks airMAX products. These communications included start, status requests and updates, GPS waypoints and ASV's position. Kendoul et al. (2009)'s "Quadrotor-based Aerial Platform" and "Real-time software" are very worthwhile the study for further explanation of a similar setup when performing processing on the boat.

The other main consideration is the bandwidth of the communications' channel as this is affected by the drone's distance from the vessel, and the volume of data to be sent. When the image processing is done on board, the communication channel is primarily receiving status updates, rather than providing flight control commands. This is preferred as the MAVLink bandwidth is dependent of the air data rate. This can be set to thirteen possible rates, namely 2, 4, 8, 16, 19, 24, 32, 48, 96, 128, 192, and 250, where the default value of 64 corresponds with 64kbps (uBlox, 2017). However, if error correction data is transmitted (recommended for long range), the air data rate is halved (SiK Radio. 2017). The higher the air data rate, the shorter the range of the devices.

2.4.4. Summary of Overall UAV, ASV System Requirements

In summary, the landing task is the most complex, and requires a combination of sensors to achieve robust landing in the more uncontrolled marine environments. For general flight navigation and approaching the surface vessel, the GPS sensors are ideal due to their long range, but within 30 meters, the visual feedback approach is a highly developed approach, and can be used for identifying the platform, tracking the platform and pose estimation. There are benefits to both a downward looking camera on the drone and an upward looking camera on the surface vessel, tracking the drone, such that it is advisable to have both, and use the downward looking camera for surveillance during flight, and a complementary pose feedback during landing. The processing can be adequately supported by computing power options either on the drone or vessel. The preferred software configuration is ROS, following the arrangements given in Weaver et al. (2013) and Chandra et al. (2016). It is advisable to develop the system level code on the boat initially, validate it, and then transfer it to a portable platform such as the ODROID-U4.

2.5. Unmanned Aerial Vehicle Design

Over a decade of development, many and various types of UAVs have been developed, as discussed in Kanellakis and Nikolakopoulos (2017) and Cai et al. (2014). One of the novel hybrid UAV designs was developed by von Frankenbeg (2016), comprising a conventional quadrotor with four orthogonally perpendicular rotors to the main rotors for increased manoeuvrability and rejection to disturbances, ideal for landing on inclined platforms. However, as this project is seeking to maximize the payload available for sensors, the addition of four extra motors, Electronic Speed controllers (ESCs) and mounting brackets, could not be justified.

Figure 2.18: Some of the different types of drones on the market (Kanellakis & Nikolakopoulos, 2017).

In this project, the VTOL, multi-rotor aircraft have been shortlisted due to the requirement to land on a relatively small landing platform, requiring manoeuvrability and hovering abilities, combined with a small footprint. Consequently, the limitations of short flight times and limited payloads have to be borne (Kanellakis & Nikolakopoulos, 2017).

2.5.1. Characteristics of Similar functioning UAVs

The following lists the specifications of drones used for similar applications in autonomous flight.

- Kendoul et al. (2009) used a X-3D-BL quad-copter from *Ascending Technologies GmbH*, which is 530 mm rotor-tip to rotor-tip, weighing 400g with the battery. Designed for 300g payload, it is capable of 12 minute flight times with full payload and 22 minutes unloaded.

It used the Gumstix flight controller with built in IMU, GPS and pressure sensor, and a *wifistix* card for communication over 500m (50Mbps). An analog camera and 1.3GHz transmitter with base station frame grabber were used for visual feedback.

- Lee et al. (2012) used a quadcopter with 600 mm from motor centre to motor centre diagonally, and equipped with two batteries, weighed 1700g. A 50° FOV camera (320x240 pixels) provided a 30fps live feed to an off-board processor via a 1.2GHz link. Main RC commands were sent over a 2.4GHz link through a USB dongle.
- Meier et al. (2012)'s drone shown below, was 550 mm diameter and weighed 1000-1200 gms, achieving flight times up to 16 minutes. It was able to carry between 400-800g payload, using 8" or 10" propellers to provide 450-600gms thrust each. Additionally, it used a form of PIXHAWK and the preferred QGroundControl program on their base-station.

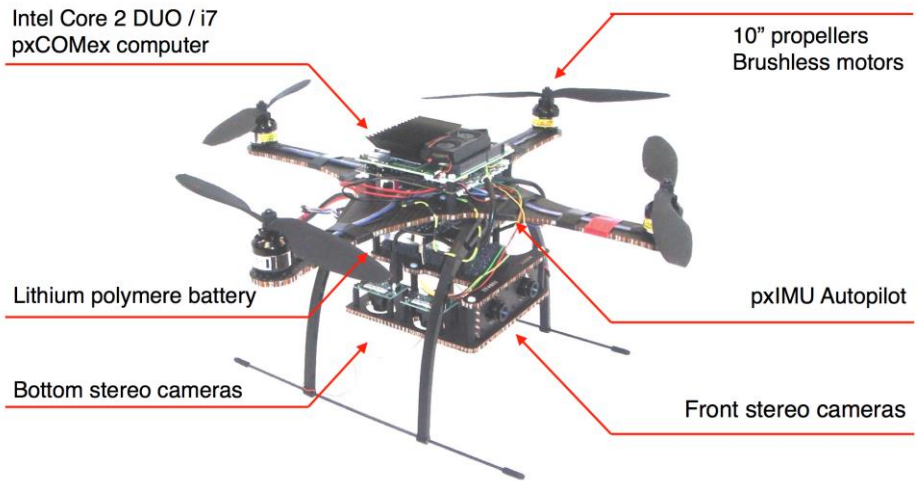


Figure 2.19: The PIXHAWK MAV drone, having on-board image processing capability. (Meier et al., 2012)

- Pinto et al. (2014) used a hexacopter, while Weaver et al. (2013) and Kim et al. (2014) both used the F450 frame from DJI Flame Wheel. Weaver et al. (2013) used a 6000mAh, 3 cell LiPo battery to achieve 15 minute flight times with a 3.8 lb (1700gm) quadcopter lifted by 2100 gms thrust collectively from the motors.

2.5.2. Requirements for chosen UAV

Therefore, from the above examples, the following general guidelines will guide the drone selection:

- The approximate size is around the 400 -550 mm diagonal width to be able to carry around 400 gms payload.
- Estimated flight times depend on the payload but should be around 15 minutes.
- At least a camera and two independent wireless links are needed with the surface vessel.

Overall, the system will be designed to ensure the drone platform is scalable and the senor and software network is modularity, for future enhancements to develop the research platform. Sufficient payload capacity will be built into the design, whilst sufficient margin will be given to ensure the drone is resistant to normal wind loads.

2.6. Project Design Overview

The following defines the scope of the project.

2.6.1. Design requirements

- Preferred flight controller to be the PIXHAWK, using Ardupilot software.
- Utilize Off-the-Shelf (OTS) componentry as much as possible to simplify development.
- The budget for the overall project is $1800 (Companion projects included).
- Communications via the MAVLink Protocol, over 900MHz.
- The platform design and software developed must be compatible with the TopCat.
- The design shall be scalable and the software development, modular.
- Drone capable of carrying the required sensory payload with additional capacity.

2.6.2. Research question defined

Overall Project:

To autonomously take-off and land a small drone from a platform that is initially, stationary, and then, if time available, on a platform moving according to the dynamics of a vessel.

Individual Project:

To identify and design the most suitable drone and software system control to achieve autonomous take-off and landing from Flinders' autonomous surface vessel, TopCat. This involves the evaluation of the current techniques used for autonomous landing, and the implementation of the autonomous landing firstly on a stationary platform, and then one moving in six degrees of freedom, to simulate the marine environment's disturbance.

2.6.3. Key assumptions

- The standalone Quadrotor control software will be developed, but integration with the ASV's existing software is beyond the scope of this project.
- Only a purely functional implementation of each software system is required at this early point in development. Future work can involve optimizing component systems, including full error handling and working moving from semi-automatic to autonomous control.
- Obstacle avoidance is not required for this project, except to avoid the surface vessel's sensor array.
- The drone is being designed to only operate in relatively calm sea swells.

3. METHODOLOGY/ EXPERIMENT DETAILS

The following section describes the refinement of the design options based on market availability, whilst providing justification for the preferred software systems used. The first sub-section expounds the drone's composition, particularly the reasons for the sensor suite chosen, while the second sub-section outlines the control sequence and preferred software configuration to achieve autonomous deployment from/to a moving platform. Additionally, an outline of the experiments to verify the system performance is provided.

3.1. Selection of Suitable Sensors and UAV

The development of an autonomous drone and its controlling software is heavily dependent on the technology available in the market and the ability to successfully integrate the individual components.

3.1.1. Suitable Flight Controller for Autonomous Flight

The most essential component of the aerial vehicle is its flight controller, providing not only the control signals to each motor and maintaining dynamic stability, but the ability to follow paths and complete missions autonomously. Many cheap flight controllers exist on the market, such as the Lumenier LUX, CC3D and Flyduino KISS (<$50), designed for miniature or sports drones (with diagonal arms less than 250mm). Additionally, flight controllers designed for smooth flight and photography applications, including the DJI NAZA-M V2 ($300), DJI A3 ($900) and Snapdragon ($675), are expensive and lack the flexibility to customize the firmware. The PIXHAWK autopilot, however, is a fully featured, cost effective ($200) flight controller that has been a popular choice by researchers, including Chandra et al. (2016); Lee et al. (2015); Meier et al. (2012). Its specifications include:

- 32bit STM32F427 ARM Cortex® M4 core with FPU
- 168MHz/256kB RAM/2 MB Flash
- 32-bit STM32F103 failsafe co-processor
- ST Micro L3GD20H 16 bit gyroscope
- ST Micro LSM303D 14 bit accelerometer / magnetometer
- Invensense MPU 6000 3-axis accelerometer/gyroscope
- MEA MS5611 barometer

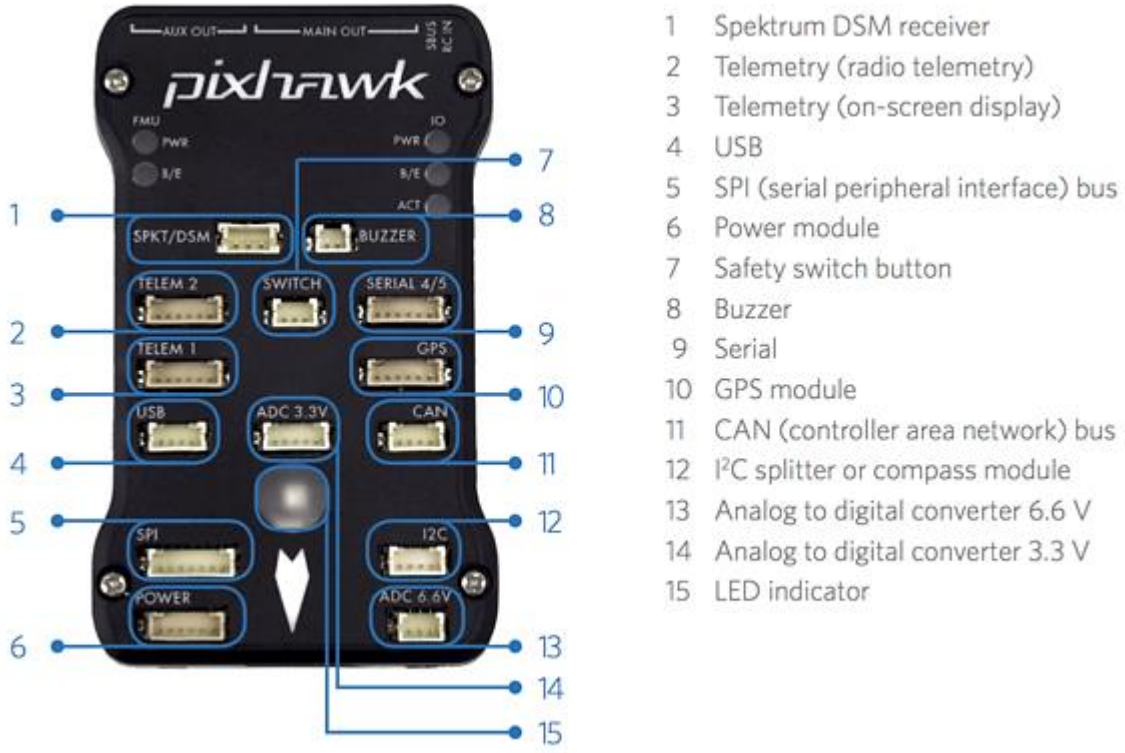


Figure 3.1: The PIXHAWK Autopilot showing the various connection interfaces available. The connectors to the motors and accessories are on the top (Autopilot, 2017).

Its features include a suite of flight modes (uBlox, 2017) with the following particularly useful for the project:

- LOITER – Uses the GPS three dimensional lock to hold the vehicle's position & altitude, compensating for wind disturbance to approximately 25kmhr.
- AUTO – Automatically perform missions, involving take-off, following a series of waypoints (GPS points) and landing, amongst other commands.
- GUIDED – Enables asynchronous commanding of the drone's position, using user or computer generated waypoints, to enable features such as "Follow Me" mode.
- LAND / Return-to-Launch (RTL)– Automatic controlled decent at a user configurable speed, either at the current location or a predetermined "home" position.
- STABILIZE – Manual flight mode, where the pilot has complete control of the drone's movement (i.e. yaw, pitch, roll and throttle).

In addition, the PIXHAWK was selected because it has a well-documented online support, has previously been paired with Linux systems running ROS, uses the MAVLink communications protocol that is compatible with the TopCat, and supports many sensors essential for autonomous landing.

3.1.2. Selection of Key Sensors for Autonomous Flight

The main drone-mounted sensors needed for autonomous landing include an on-board camera and rangefinder, as outlined in Section 2.4.1. The Pixy IR-Lock sensor was chosen as the downward looking camera for detecting the landing pad and assisting with Precision Landing. (Please refer to Prasetyono (2017)'s report for full details). The following table describes shortlisted rangefinders that were considered, revealing the comparative ranges, power consumption and accuracies available. The optical rangefinder from LeddarOne was chosen due to its full support by PIXHAWK, ease of interfacing and its popularity, including companies such as Draganfly Innovations Inc. which have adopted it as their predominant UAV altimeter. Using a signature modulated infrared beam, this sensor provides altitude readings up to 40m at 140Hz refresh rate, for an affordable price (Leddartech, 2017).

TABLE 3.1: Comparison of available Rangefinders in the market.

Rangefinder Sensor Options							
Supported by PIXHAWK	**N**	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
Type	Adafruit VL53LOX	LV-MaxSonar-EZO	XL-MaxSonar-EZO	MB7040 I2CXL-MaxSonar-WR	LeddarOne optical range finder	LIDAR-Lite 3 Laser Rangefinder	SF10/A
Reading rate	30Hz	20Hz	10Hz	42kHz	140Hz	500Hz	32Hz
Resolution		25.4mm	10mm	10mm	3mm	±1cm	10mm
Accuracy	< ±12%			±0.5%	±5cm	±2.5cm	±0.05m
Interface	I2C	RS-232	RS-232 serial	I2C	3.3V UART or RS-485	I2C or PWM	I2C
Voltage	3-5V	2.5 to 5.5V/2mA	3.3 and 5VDC	3-5.5V/3.4mA	5V/260mA	4.75-5V/135mA	5V/150mA
Range	5cm to 1.2m	20cm to 6.45m	20cm to 7.65m	20cm -7.65m	20cm to 40m	0-40m	0 to 25m
Weight				32g	14g	16g	35g
Price	$18.90 AUD	$37.70 AUD	$63.00 AUD	$99.95	$148.63	$169.50	$352 AUD
Speciality	Time of Flight distance sensor - invisible laser	Ultrasonic Range finder	Ultrasonic Range Finder	Time of Flight distance sensor - Ultrasonic sesnor	Time of Flight - modulated IR , 3deg diffused beam	Time of Fligth - Signature encoded laser beam	Otical aperature 51mm, 0.4deg beam, micro USB plug
Product Link	https://www.adafruit.com/product/3317	https://www.sparkfun.com/products/8502	https://www.sparkfun.com/products/9491	https://www.maxbotix.com/Ultrasonic_Sensors/i2c_distance_sensors-2.htm	http://leddartech.com/modules/leddarone/	http://www.robotshop.com/en/lidar-lite-3-laser-rangefinder.html	http://lightware.co.za/shop2017/drone-altimeters/26-sf10a-25-m.html

Another sensor used for localization purposes, was the uBlox Neo-7M digital Global Positioning System (GPS), receiving GPS L1 C/A, GLONASS (Russian GPS) corrections, and provides a position estimate to within 2.5m for GPS at a 10Hz update rate (uBlox, 2017). As part of the PIXHAWK kit, it connects to the flight controller using serial and I^2C interfaces, and is capable of receiving Real-Time Correction Messages (RTCM v3) from a base-station to improve the position accuracy. Augmented by a built-in compass, this sensor provides a global reference for the drone, and is supported by the vision guidance system for greater accuracy.

3.1.3. Refined specifications from requirements

In order to assess the existing market options for a suitable platform for the autonomous drone research, one should understand the essential components of a drone and their specifications.

Frame:

Generally speaking, small quadrotors (up to 200mm diagonally across) exhibit faster dynamics than larger drones and are fine-tuned with regard to weight, making them most suitable for racing or hobbyist applications. However, where stability and long flight times are required, such as in aerial surveillance, larger frames provide the robustness against wind disturbances and increase the flexibility regarding payloads, whilst being easier to fly. Drawing on the findings presented in Section 2.5, the frame size was narrowed to between 350-500mm diagonally across, with a target weight (AUW) of less than 1.4 kilograms, subject to further refinement of components.

Motors

The prevailing multirotor motor used is the brushless DC motor, due to its high responsiveness, long lifetime, minimal wearable components, and high efficiency. It comprises fixed stator core, wound in diagonally opposite pairs, forming three-phases as shown in *Figure 3.2*. An electronic speed controller (ESC) generates a rotating magnetic field that drives the motor shaft, by pulsing the three motor phases in the sequence shown on the left.

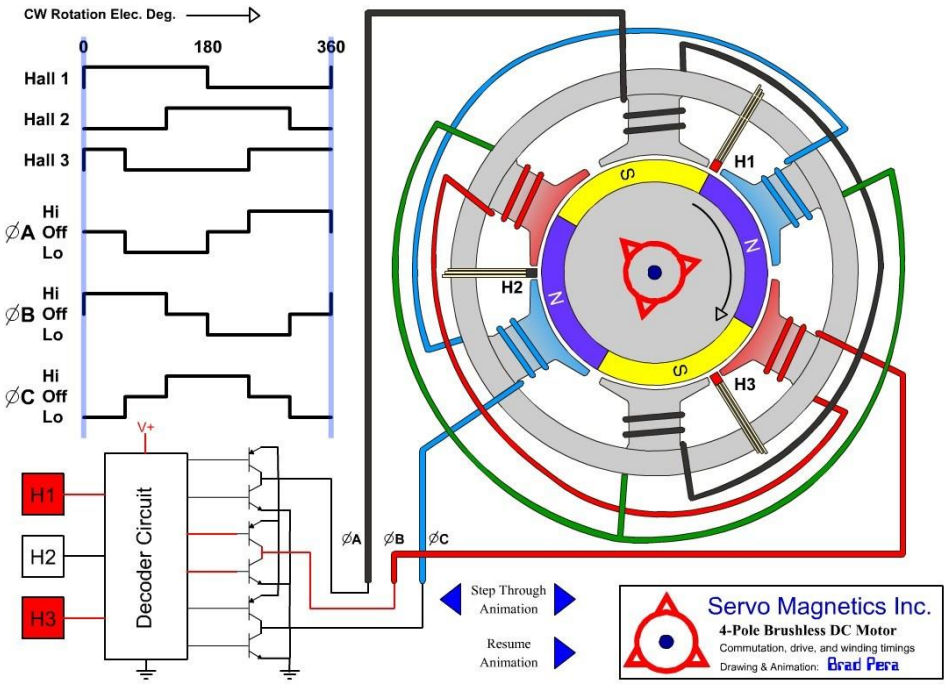


Figure 3.2: Diagram illustrating the three-phase brushless DC motor coils and excitation sequence. © Servo Magnetics Inc.(Drone_Insider, 2017)

Brushless DC motors are typically specified by the following information, where the first two letters "RS" identify the brand (Race Spec), while "22" refers to the stator width and "05", the stator height in millimetres. The KV value represents the motor's revolutions per minute per volt, when it is unloaded, where the higher the number, the more acrobatic or high performance the motor. However, lower KV value motors are generally more efficient and possess greater torque, and therefore used with large propellers to carry large payloads for long periods. The last term lists the number of electromagnets in the stator (N) and permanent magnets in the rotor (P), where higher values correspond to finer controlled motors with higher torque.

$$Common\ Motor\ Specifications: \quad RS2205 \quad 2300kv \quad 12N14P$$

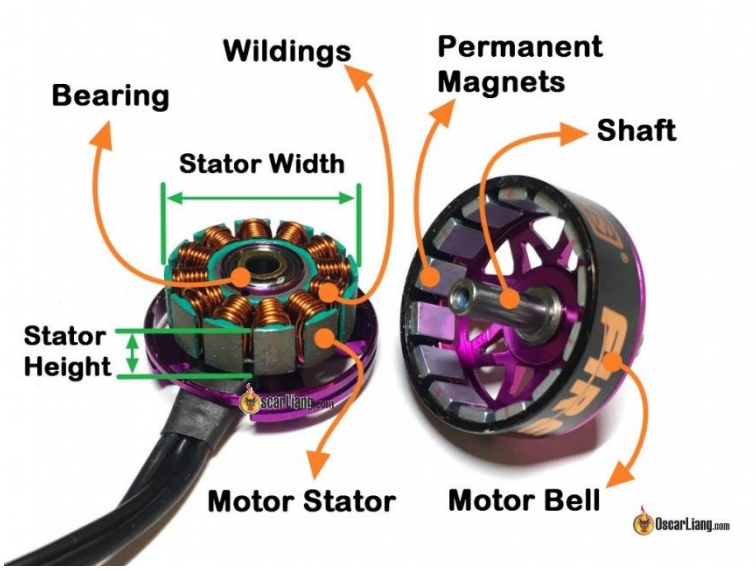


Figure 3.3: Breakout of three-phase brushless DC motor (FPV_Drone, 2017).

Market research indicates mini-quads used for racing use 1806 or 2204 motors, while 2212 motors are more suited to heavier loads. Before settling on a preferred motor range, the following extra considerations were understood:

- Thrust to Weight ratio – The combined thrust of the four motors must not only lift the fully loaded quadrotor, but have sufficient capacity to achieve the desired responsiveness of the drone during flight. Quadrotors that hover at 70% throttle will exhibit slow but smooth response to user control, so a rule of thumb for design is at least a 2:1 ratio, leaving sufficient capacity to carry larger batteries for increased flight time.
- Operational voltage – The chosen battery voltage level affects the theoretical motor speed which is calculated by the KV rating multiplied by the battery voltage.

- Current Draw and Efficiency – One motor can power a range of propeller sizes, but these are restricted by the frame size and the corresponding efficiency of each motor-propeller combination. Larger motors better handle steeper pitch propellers but add to the overall weight of the aircraft. Therefore, thrust tables are provided by suppliers, indicating the thrust, efficiency and power consumption of various propeller combinations and voltage levels, for the specific motor.

Propellers:

Propellers are typically specified by a diameter (inches) and pitch. Thus, a 10 x 45 propeller is 10" diameter, and will screw 4.5" forward for one revolution through a solid medium. This project needs stable hovering provided by vertical take-off and land (VTOL) propellers, ranging in size between 7"-10" to achieve at least double the thrust as the drone's AUW.

ESCs:

The motor speed controllers need to handle the peak current draw of the motors which is between 12-18Amps for larger motors. Additionally, fast motor response is achieved using the popular ESC firmware, chiefly BLHeli BLHeli_S and SimonK, firmware. The newer BLHeli firmware are ideal for racing mini-quads, boasting over 20x faster response times than PWM using protocols such as MultiShot and DShot 150 – DShot 900. The SimonK firmware has been extensively used for larger aircraft, with a recent addition of support for the OneShot 125 protocol that reduces response times by four times compared with PWM signalling (Ribeiro, 2017). This is the preferred firmware due to the greater available of high capacity ESCs.

Additionally, some ESCs provide a regulated 5V power supply through an in-built battery eliminator circuit (BEC), removing the need for additional voltage regulators. Although useful as a backup power supply for the Flight Controller, it is more suited for ancillary components, due to issues around conflicting voltage rail references.

Battery:

Lithium-ion Polymer (LiPo) batteries are the most prevalent multirotor battery due to their high energy density, capacity and light weight. However, their selection is heavily dependent on the nature of motor chosen. These batteries comprise individual LiPo cells (3.0-4.2V range) arranged in parallel or series combinations, where $3S1P$ indicates three cells in series, and one connected in parallel, producing a fully charged voltage of 12.6V or nominally, 11.1V. The C-rating (Discharge rate) is related to the capacity as shown in Eq.1, which specifies the maximum

continuous discharge current from the battery, that should exceed the peak combined current draw from all motors. Also, the flight time can be estimated using Eq. 2. Additionally, the burst current rating specifies the maximum current that can be drawn from the battery within a 10 second period.

$$Max\ \ Continuous\ Discharge\ Current = C\_Rating \times Capacity \qquad Eq.1$$

$$Max.Expected\ Flight\ Time[min] = \frac{Capacity[mAh]}{Average\ Current\ Draw[A]} \times \frac{60}{1000} \qquad Eq.2$$

The ideal battery choice maximizes the battery capacity whilst minimizing the weight within the physical constraints, in order to extend the flight time. From similar implementations described in Section 2.5.1 and market research, 3S or 4S batteries with capacities between 3000-6000mAh can sustain quadrotors around 1.2kg for flight times up to 15 minutes.

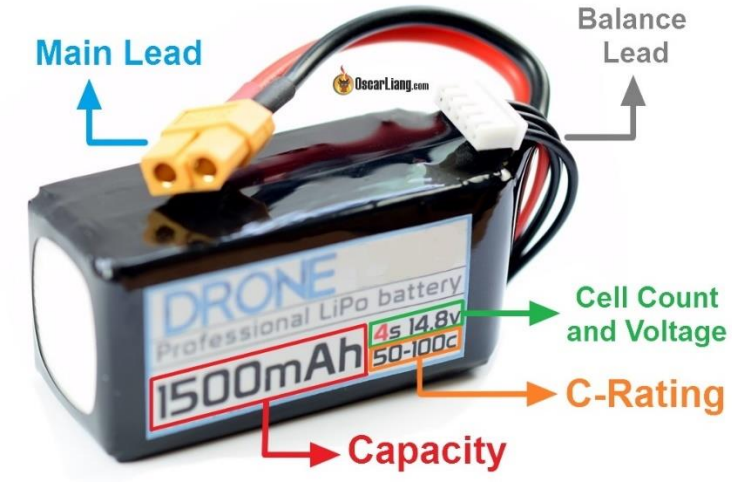


Figure 3.4: Typical Lithium-ion Polymer (LiPo) battery specifications (Liang, 2017).

The logistical considerations regarding batteries include the physical size and connector types. The selected battery should fit within the allotted space of the drone's battery compartment whilst remaining short enough to allow for fine adjustment of the drone's Centre of Gravity (CoG). The connectors fitted were to be quick release, safe and able to handle the current draw.

Telemetry & R/C Receiver:

Additional essentials for the drone include manual control and telemetry feedback to the base station and controlling computer. The typical R/C Transmitter/Receiver combinations for drones use the 2.4GHz spectrum, which has the least environmental distortion. The preferred telemetry frequency in Australia is the 900MHz spectrum, providing long range communications over 1km, using the MAVLink protocol.

The following tables present the narrowed specification range upon which the Off-the-Shelf options were evaluated for suitability for this application.

TABLE 3.2: Refined Specifications of Essential Drone Components.

COMPONENTS	REFINED SPEC. RANGE
TOTAL WEIGHT (AUW)	Up to 1.4kg & 400gms payload
FRAME	350-500mm
MOTORS	2212 or greater, less than 1300kv
PROPELLERS	7”-10”, VTOL, >600gms thrust
ESCS	4x 15-30 Amps, Oneshot125, 5V BEC
BATTERY	3S-4S & 3000-6000mAh
FLIGHT CONTROLLER	PIXHAWK, ArduPilot
TELEMETRY	900MHz, MAVLink
R/C RADIO RECEIVER	2.4GHz
FLIGHT TIME	10 – 15 minutes

Therefore, due to the interdependence of the essential components of a drone, the designer needs to iteratively re-evaluate the above combinations, based on the refined weights and power specifications as components are selected.

TABLE 3.3: Additional Drone Components Required.

COMPONENTS	REFINED SPEC. RANGE
LOW VOLTAGE ALARM	Audible Buzzer & Numerical display
ALTIMETER	LeddarOne Optical Rangefinder
ONBOARD VISION SYSTEM	FPV camera, transmitter (5.8GHz), antennas
DOWNWARD FACING CAMERA	Pixy CMUcam5
LANDING FEET	Flexible, net landing

3.1.4. Evaluation of Off-the-Shelf vs. Custom Built Drone

Given the above required specifications, the market research identified a range of ready-to-fly (RTF) or almost-ready-to-fly (ARF) drone kits, with the following final shortlist. Given the need to use the PIXHAWK flight controller, and the potential for the drone application to be expanded due to research needs, the commercial drones such as DJI Phantom and MAVIC Pro were not considered due to the lack of flexibility and their cost exceeding the allowed budget of $1800.

- Storm Drone AntiGravity (RTF) – 640mm diagonally, 1.21kg weight w/o battery, 1.83kg with 6S 4200mAh battery and T-Motor MN4006 380kv motors, 40A ESCs with Tarot 1555 folding propellers, controlled by Naza V2 Flight Controller Unit (FCU) and capable of around 30 minutes flight time. The basic kit costs US$849, or ~$1070AUD, and will need reworking for sensor mounts (Helipal, 2017).
- QAV400 FPV Quadcopter RTF – prebuilt & tested, 1.17kg with 4S 2200mAh battery, GoPro, using FXC2216-9 1100kv motors, 35A ESCs and Graupner 8x5 propellers. Capable of 12 minute flight time using 3300mAh battery, this kit retails for US$800 with options of supplying one's own motors and FCU, with an additional $200 for tuning. Other variations include the QAV500 (out of stock) and QAV540 (US$900) kits from getFPV, which supply the local Queensland provider, Desert Aircraft (getFPV, 2017).
- Turnigy SK450 Quad Copter (PNF) – 680g weight w/o battery, using Multistar 2213 935kv motors, 20A ESCs, 8045-1045 propellers, on a proven frame, using the KK2.1 FCU. The batteries and R/C controller would need to be procured independently, whilst the frame is not large enough for camera and other accessory mounts. Despite the lack of flexibility for research, this kit is an excellent starting point for medium size quadrotors (Hobbyking, 2017).

Working to a budget of $1800, it was discovered that purchasing the QAV400 kit components locally, including the preferred essential drone components would cost approximately $670AUD for a basic platform, compared with $1010AUD for the equivalent quadrotor from the US, requiring retrofitting, not to mention shipping costs. Therefore, the decision was made to proceed with a custom drone design based on the QAV400 frame, maximizing the authors' learning experience, whilst creating an ideal solution that is suitable for future development.

3.1.5. Drone Assembly Testing Process

The construction of the drone was validated by the following tests before flight and its overall performance qualitatively evaluated through a range of test flights. The initial manual flight tests sought to build confidence in the quadrotor's ability for controlled flight, followed by tests validating the existing capabilities of the PIXHAWK flight controller for autonomous flight.

Bench Tests:

- Verify AUW of Quadrotor, including battery options and extra sensors' weight.
- Basic power-up test, confirming wiring setup, voltage levels.
- R/C transmitter & receiver pairing, validating vision system operation.
- Flight controller configuration – orientation, compass, GPS, R/C channels, flight modes, fail-safe actions, R/C control of motors.

Manual Flight Tests:

- Maiden flight – take-off, hovering throttle level, basic flight modes.
- PID tuning of Roll, Pitch & Yaw axes.
- Qualitative assessment of Motor/Propeller combination.
- Flight time with different batteries, flight characteristics - responsiveness (sufficient thrust capacity), and robustness against wind disturbance (Loiter).

Autonomous Flight Features Validation:

- LOITER, POSHOLD/ALTHOLD (indoors), auto LAND, RTL, GUIDED modes.
- Load a Mission file using the Base station software, test AUTO mode including AUTO take-off, waypoint following and RTL.
- System Integration – test Pixy, LeddarOne rangefinder.
- UAV software control components partnered with manual control.

3.1.6. Drone Landing Pad Development Process

For independent system testing of the drone's precision landing algorithms on the land, a provisional landing pad was constructed that incorporates the vessel's upward looking camera as well as another flight controller, to simulate the vessel's orientation and heading. A CAD model of the proposed landing platform for the TopCat was generated, finalizing the expected position of the camera, but physical implementation was left until the landing net is procured. In addition, the landing feet for the quadrotor was constructed, with features incorporated for landing on a low amplitude, oscillating net.

3.2. Overall System Software Control of the Autonomous Drone

Having discussed the creation of the quadrotor platform, the following sections outline the development of the software control of the drone through ROS, describing the planned overall software system function and structure to achieve autonomous take-off and landing. Specifically, this section outlines the breakdown of this complex project between the group members, before outlining in detail the Control Implementation software that is the focus of this report.

3.2.1. Detailed Control Sequence of the Autonomous Take-off & Landing

Firstly, the strategy shown on the following page gives an overview of the planned control sequence to achieve the autonomous take-off, mission and landing, providing insight into the level of system integration required. This combines lessons learnt from similar implementations discussed earlier, particularly Chandra et al. (2016); Djapic et al. (2015); Pinto et al. (2014) and the design decisions from partner group members.

The goal of the overall project is to produce a proof-of-concept software control system that initially runs on the surface vessel's computers, utilizing its computational power, with future work separating this into independent ROS Masters that communicate over ROS Bridge, enabling parallel operation, as implemented by Weaver et al. (2013). Therefore, the interface between the UAV software control and the Vessel's existing system for this year's project is shown in *Figure 3.6*.

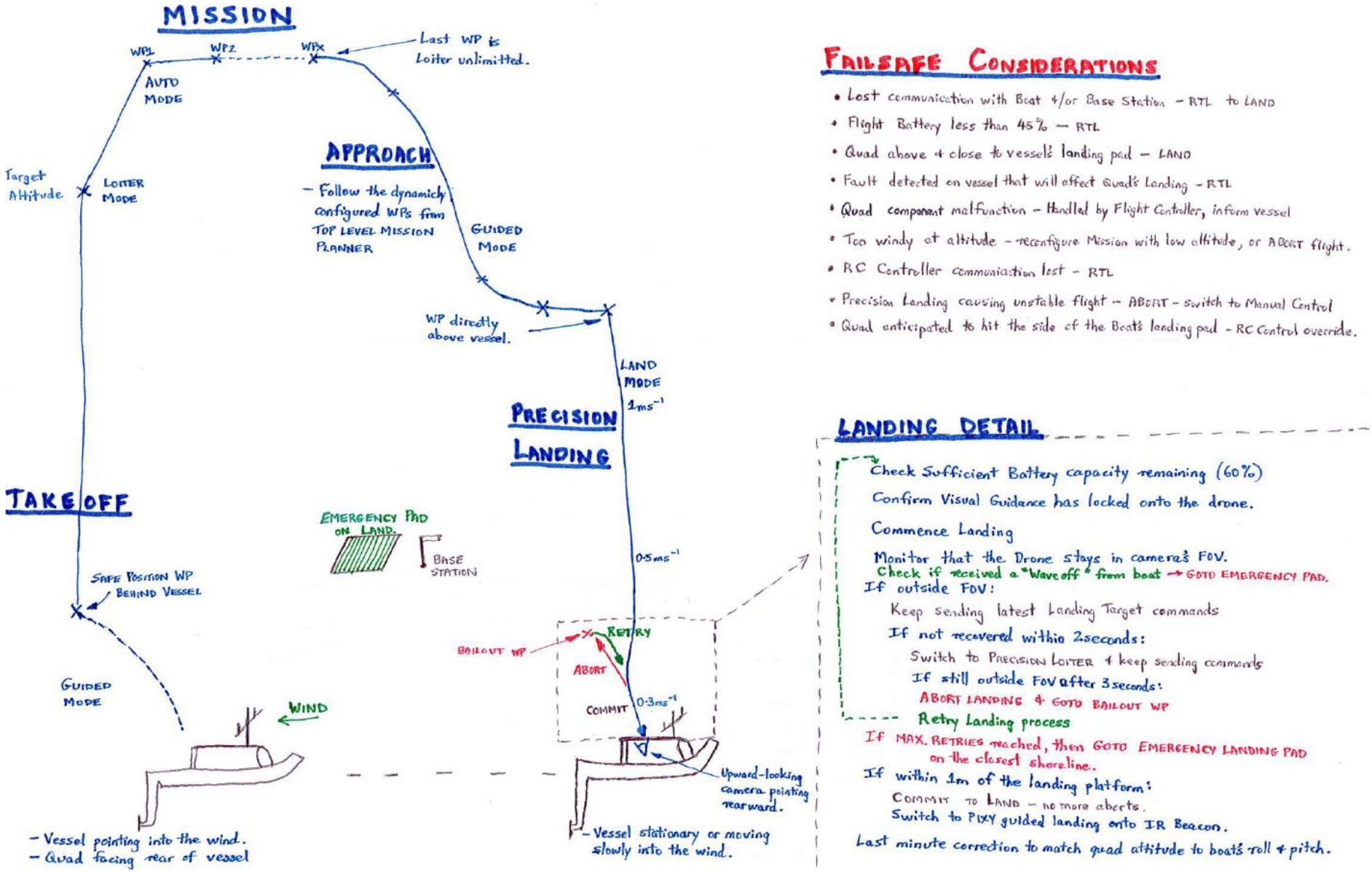


Figure 3.5: Quadrotor Control Strategy to perform Autonomous Take-off and Landing on the Autonomous Vessel, TopCat.

3.2.2. Functional Block Diagram of Drone Control Software

The functions outlined in *Figure 3.5* will be broadly divided into three interdependent elements illustrated in *Figure 3.6*, where the UAV Control Implementation is the author's responsibility.

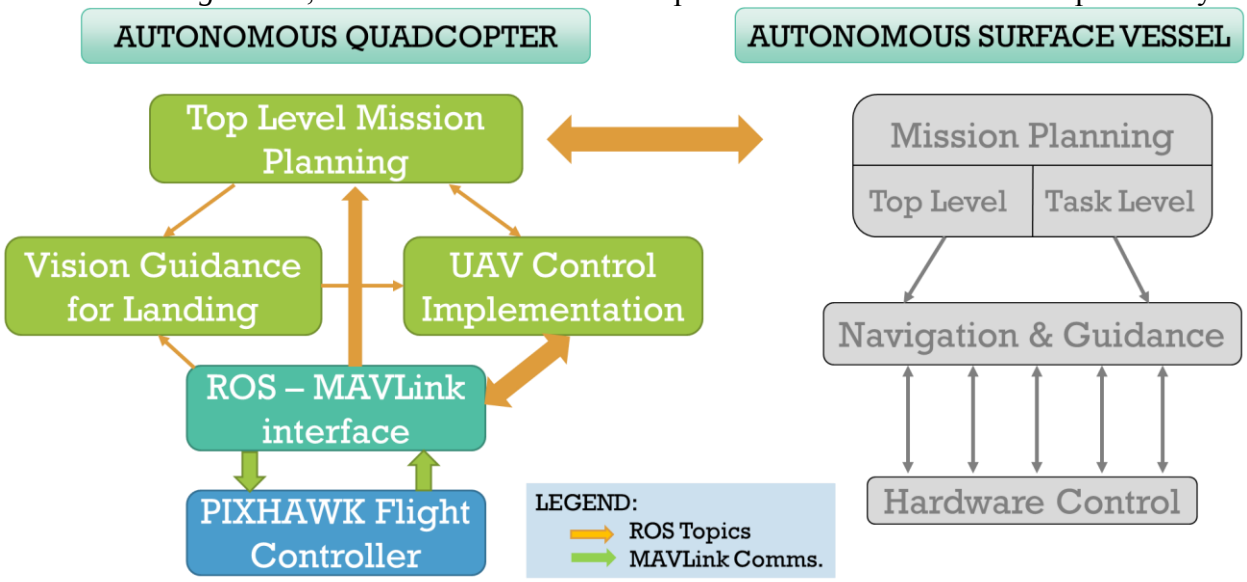


Figure 3.6: Overall Diagram of the Quadrotor Software Control in relation to the ASV control.

The information shared between the Software nodes is outlined in the following section, using ROS topics as detailed in *Figure 3.7*, but the primary control and feedback communication from the PIXHAWK on the quadcopter is through the MAVLink telemetry radios. As the ROS-MAVLink interface returns the flight status information to a series of ROS topics, this information is publically visible to the three main nodes above, but the UAV Control node is the only program that sends commands to the FCU. This structure was chosen to avoid conflicting or simultaneous commands to the Flight Controller, by assigning the sending responsibility to one program. Additionally, this allowed extra bandwidth for receiving flight data that would inform the decision making of all subprograms.

3.2.3. Information Flow between the Software Nodes

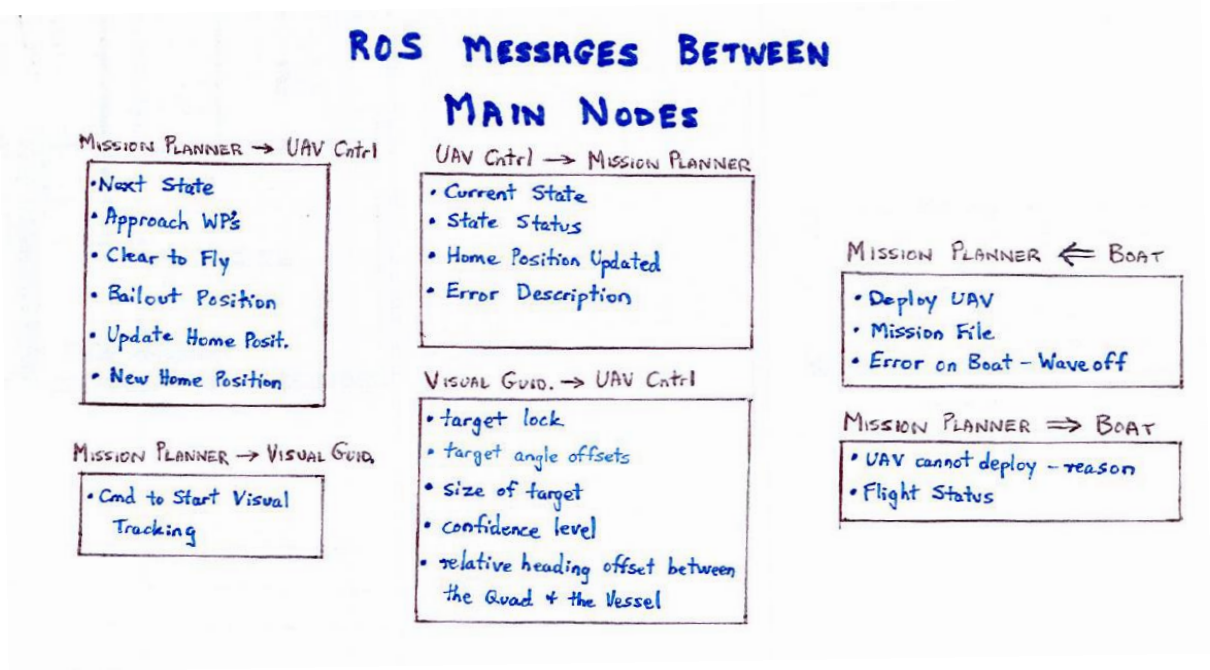


Figure 3.7: ROS Topic messages between the main nodes of the Quadcopter control.

3.3. UAV Control Implementation Software for the Drone

Given the overview of the complete system, the following section expounds the development of the Control Implementation software, including the low-level interface for commanding the PIXHAWK flight controller.

3.3.1. Breakdown of UAV Control Implementation

The UAV Control software comprises a set of states that command the quadcopter throughout the stages of the flight, handling temporary command errors and monitoring the progress of the drone, as shown in *Figure 3.8*. It is responsible for sending the planned trajectory path to the drone, in addition to the landing target feedback that is provided by the Visual Guidance for landing segment. However, the decision to transition between states is handled by the Top Level Mission Planning section.

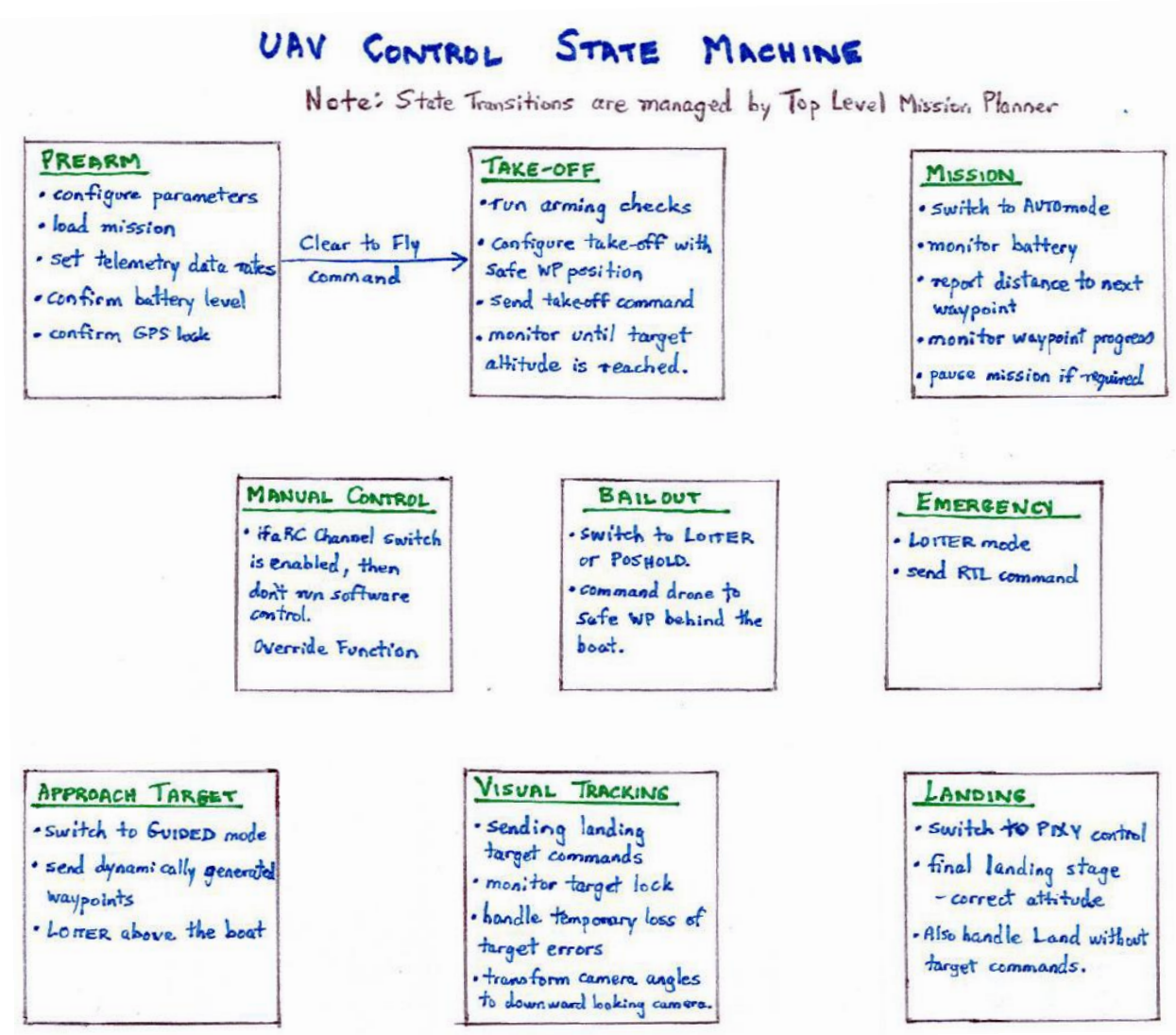


Figure 3.8: Quadrotor Control Software States and their component tasks. Note, the decisions governing transitions between the states is handled by the Mission Planner.

The supplementary HOVERING and LAND states were implemented for quick recovery of the quadcopter via software if needed during field testing.

The States shown above have the following features:

- Only one of the states can be active at a time.
- Each state has an associated status that is published, indicating the progress and whether any errors were encountered.
- If a persistent error is discovered, then the state will flag the error and leave the state, waiting for commands from the Top Level Mission Planner.

To develop this state machine, the C++ programming language was selected due to the finer low-level control and the superior error checking compared with Python. Additionally, the author's familiarity and group's adoption of the C++ language made development easier. The techniques used in the DroneKit Python examples were transferable.

Furthermore, one key consideration involved selecting what MAVLink command(s) the software would use to guide the drone through the precision landing process. One factor governing the command selection was the preference to specify an approach trajectory rather than repetitive positional corrections as it is smoother and more easily corrected given wind disturbances. Commands considered from the MAVLink specification (Mavlink, 2017) include:

- × MAV_CMD_DO_FOLLOW - lacks precision due to GPS uncertainty,
- × HOME_POSITION – specifying a 3D approach vector but based on GPS co-ordinates,
- × SET_POSITION_TARGET_LOCAL_NED – can specify position/velocity trajectory
- ✓ LANDING_TARGET - specifies the location of the landing target as seen from a downward facing camera fixed to the drone.

The LANDING_TARGET command was selected since it utilizes the relative target position derived from the image to update the quadcopter's descent trajectory and has been successfully used by IR Lock and numerous developers including Quilter (2017). Additionally, this command does not rely on altitude readings to determine the target angle.

3.3.2. ROS to Flight Controller Linking Software

The following explains the evaluation of options for the ROS to MAVLink communications link, and provides justification for choosing the MAVROS package.

- × MAVPROXY – Using a single python script, this command-line tool is effectively a ground control station (GCS), useful for live control but not programmatic control.
- × TopCat's existing MAVLINK communications –The low level package would require significant work to handle the strict MAVLink protocol enforced by the ArduPilot firmware and map each message to topics, which is unworkable in the allotted timeframe.
- × MAVLINK_ROS – A well-documented serial, middleware and top-level node, developed by the PIXHAWK/ETHZ team, has been superseded by,
- ✓ MAVROS – The popular package, proven by Chandra (2016) and regularly updated with new features, manages the low level packaging of ROS messages into MAVLink packets and provides a suite of ROS topics from the FCU telemetry. It is fully compatible with QGroundControl and ArduPilot software, and has strong online support (ROS, 2017).

3.3.3. UAV Control Software Testing Procedure

Essential to the project's success, is the ability to be confident in the performance of the drone throughout all stages, both during a successful operation and when a fault occurs. To build this trust, the software was regularly tested during the staged development, with necessary improvements made and documented.

Initial software testing comprised validating subroutines developed for routine commands such as changing the Flight mode, on a standalone PIXHAWK, and then in the field. As the control states were developed, the basic functionality was tested in the field, with the overriding Manual Flight control as the immediate fail-safe option. Console output logs were used for diagnosis.

Where possible, the component algorithms were validated first using pre-recorded data-log files, and then in the field using LOITER mode in the case of Landing Target corrections, before committing to the completely software controlled landing. Integrating the component software elements is made easier through the ROS topic interface, where the topic messages can be defined initially and incorporated into the individual programs, enabling test code to publish to these topics, validating the logic of the control sequence before introducing the complexities of the natural environment.

4. RESULTS & ANALYSIS

The following sections outline the outcomes achieved, covering the Drone Construction, UAV Control Software Development and the Overall System Integration.

4.1. Drone Construction

4.1.1. Final Drone Components & Accessories List

The following table lists the final components selected for the drone, including their corresponding weight. The QAV400 frame with carbon-fibre arms (G10) was chosen over the aluminium arms' option due better price and lead-times, with the advantage that these arms can be swapped with the QAV500 arms, if 10" propellers are required in the future. Care was taken to ensure that with selected components shown below, the propeller hubs matched the motor shaft diameters as well as the 3.5mm banana plug leads on the ESCs with the motor leads.

TABLE 4.1: Final Drone Components Purchased.

Quadrotor Item	Qty	Cost	Weight (gms)	Supplier
Essential Parts:				
Frame (QAV400 w/ G10 arms)	1	$ 200.00	375	DesertAircraft
Flight Controller Kit (HKPilot, GPS, 915MHz telemetry, wiring)	1	$ 255.40	140	HobbyKing
Turnigy L2215J-900 Brushless Motor (200w)	4	$ 74.44	304	HobbyKing
ESCs (Alfro 20A w/BEC)+SimonK,Oneshot125	4	$ 51.88	91.2	HobbyKing
Propellers (8x4.5) pairs, 6mm hub	2	$ 11.38	15.4	HobbyKing
R/C Radio Receiver - 2.4GHz	1	$ 43.14	16.8	HobbyKing
Velcro Strap for Battery	1	$ 2.43	9	HobbyKing
LiPo Low Voltage Alarm	1	$ 4.01	9	HobbyKing
Battery - 3S LiPo (3000mAh, 20C 136x44x18mm)	1	$ 21.66	224	HobbyKing
Extra Components:				
Laser Range Sensor - LedderOne Optical (3.3V UART)	1	$ 151.00	14	RobotShop
FPV camera + wiring - RunCam Swift 2	1	$ 54.27	14	HobbyKing
PIXY IR-LOCK Sensor - Pixhawk Kit	1	$ 135.00	27	AdaFruit
5.8GHz Video Transmitter	1	$ 60.94	18	HobbyKing
5.8GHz CP Antenna Set	1	$ 40.00	11.5	HobbyKing
On-Screen-Display (OSD) Link	1	$ 25.44	2	HobbyKing
Fr-Sky Telemetry overlay to the Taranis RC transmitter screen	1	$ 20.01	5.2	HobbyKing
Camera Power Filter- L-C, 1.7A	1	$ 5.23	6.4	HobbyKing
4x Landing Gear & soft pads	1	$ 4.95	5	DesertAircraft
Total		$ 1,161.18	1,287.50	
		Parts procured from the US.		

In total, the essential drone components cost approximately $665, with the completed drone weighing 1.10kg.

Thus, the specification regarding allowable payload based on the battery fitted was derived using the Thrust to Weight ratio of 2:1 as shown in *TABLE 4.2*, to provide ample thrust for more aggressive flying or larger payloads than the approximately 100gm used for this project. As will be discussed later, a 5200mAh battery was also used during testing, weighing 325gms, increasing the AUW to 1.43kg, with minimal detrimental impact on flight performance.

TABLE 4.2: Final Drone Payload Specifications.

Design Components	Formula	Calc.	Units
Individual Propeller Thrust		820	gms
Thrust to Weight Ratio	2 is to 1	2	
Desired Payload		150	gms
Actual Battery Weight (3000mAh)		224	gms
Total Thrust	4 x THRUST	3280	gms
Maximum allowable AUW	THRUST/2	1640	gms
Actual Weight w/o battery	actual weights w/o battery	960.4	gms
Max Additional Weight allowed	AUW - Actual Weight	679.6	gms
Maximum Battery Weight allowed	Addit. Weight - Des.Payload	529.6	gms
Payload already used		103.1	gms
Allowable Payload		**352.5**	**gms**
	User Configurable Items		

TABLE 4.3: Additional Components Procured.

Quadrotor Item	Qty	Cost	Supplier
Additional Parts:			
Universal FPV Monitor To Transmitter Mounting Bracket	1	$ 14.77	HobbyKing
FPV LCD Monitor Boscam - 7 inch 800 x 480 5.8GHz	1	$ 188.15	HobbyKing
LiPo battery fire safe bag (230mm x 140mm)	1	$ 3.68	HobbyKing
LiPo battery charger	1	$ 43.67	HobbyKing
Female SMA-jack to Female RP-SMA plug	2	$ 16.96	DigiKey
TH_LEDs - 950nm, 10degree divergence	25	$ 12.57	DigiKey
Breadboard for LED PCB development	1	$ 11.00	DigiKey
Zener Diode (5.6V, 5W) - PIXHAWK servo rail protection	1	$ 0.80	Element14
MOSFET Transistor, N Channel, 2 A, 30 V, Vgs(on) = 4.5 V	2	$ 0.37	Element14
Carbon fibre Rod - 0.5m, 8mm diameter	1	$ 22.50	HobbyHabit
Parts available internally:			
Upward looking camera - using existing obital Logitech Webcam	0	$ -	Flinders
Taranis X9R Plus R/C Transmitter	0	$ -	Flinders

P.T.O for continuation of this table.

Spares:			
Propellers (8x3.8) pairs, 6mm hub, 5,4,3&2.5mm adapters	4	$ 12.00	HobbyKing
Propellers (6x3) pairs, 5mm hub with 4,3 & 2.5mm adapters	2	$ 4.68	HobbyKing
Folding Propellers (2x 8045 CW & CCW) for QAV 500 & 450	2	$ 21.20	eBay
Folding Propellers (2x 7045 CW & CCW) for QAV 400 & 450	1	$ 9.20	eBay
Simple Prop Balancer - shaft only	1	$ 2.21	HobbyKing
Turnigy L2215J-900 Brushless Motor (200w)	2	$ 37.22	HobbyKing
ESCs (Alfro 20A w/BEC)+SimonK,Oneshot125	2	$ 25.94	HobbyKing
Wattmeter and Voltage Analyzer - for Motors	1	$ 27.50	HobbyKing
Wire - 18AWG (1m Red & black)	2	$ 2.16	HobbyKing
Wire - 14AWG (1m Red & black)	2	$ 4.22	HobbyKing
XT60 connectors - 5 pairs, genuine	1	$ 6.60	HobbyKing
Battery - 3S LiPo (3000mAh, 20C 136x44x18mm)	2	$ 43.32	HobbyKing
Battery (Zippy Flightmax) - 3S, 5000mAh, 25C	1	$ 32.17	HobbyKing
Spare Telemetry radios	1	$ 42.71	HobbyKing
Total		$ 585.60	
	Remaining Budget =	$ 53.22	

The additional items shown above were used for constructing the drone, features of the landing pad and providing spares and related equipment. One key consideration was R/C equipment and power supplies needed to be procured from Australian suppliers, in order to comply with Australian EM spectrum & electricity regulation. The combined purchases were within the allotted budget.

4.1.2. Drone Assembly

The assembly process of the drone went to plan, with the selected components assembled in a similar manner to that of *Figure 4.1*. The main differences were the 4 in 1 ESC shown was substituted for 4 individual ESCs and the PPM Sum Receiver was omitted, as the FrSky receiver chosen connects directly to the PIXHAWK through SBUS. No gimbal was fitted, but a first-person-view (FPV) camera with 5.8GHz transmitter with On-Scree-Display (OSD) with MAVLink telemetry overlayed was used.

The LeddarOne rangefinder was connected to the Serial4/5 port, while the PIXY was connected through an I^2C bridge to the PIXHAWK, requiring minimal configuration. The propellers were orientated according to the Quad-X formation, with flight tests confirming acceptable stability, so the H-frame configuration was not needed.

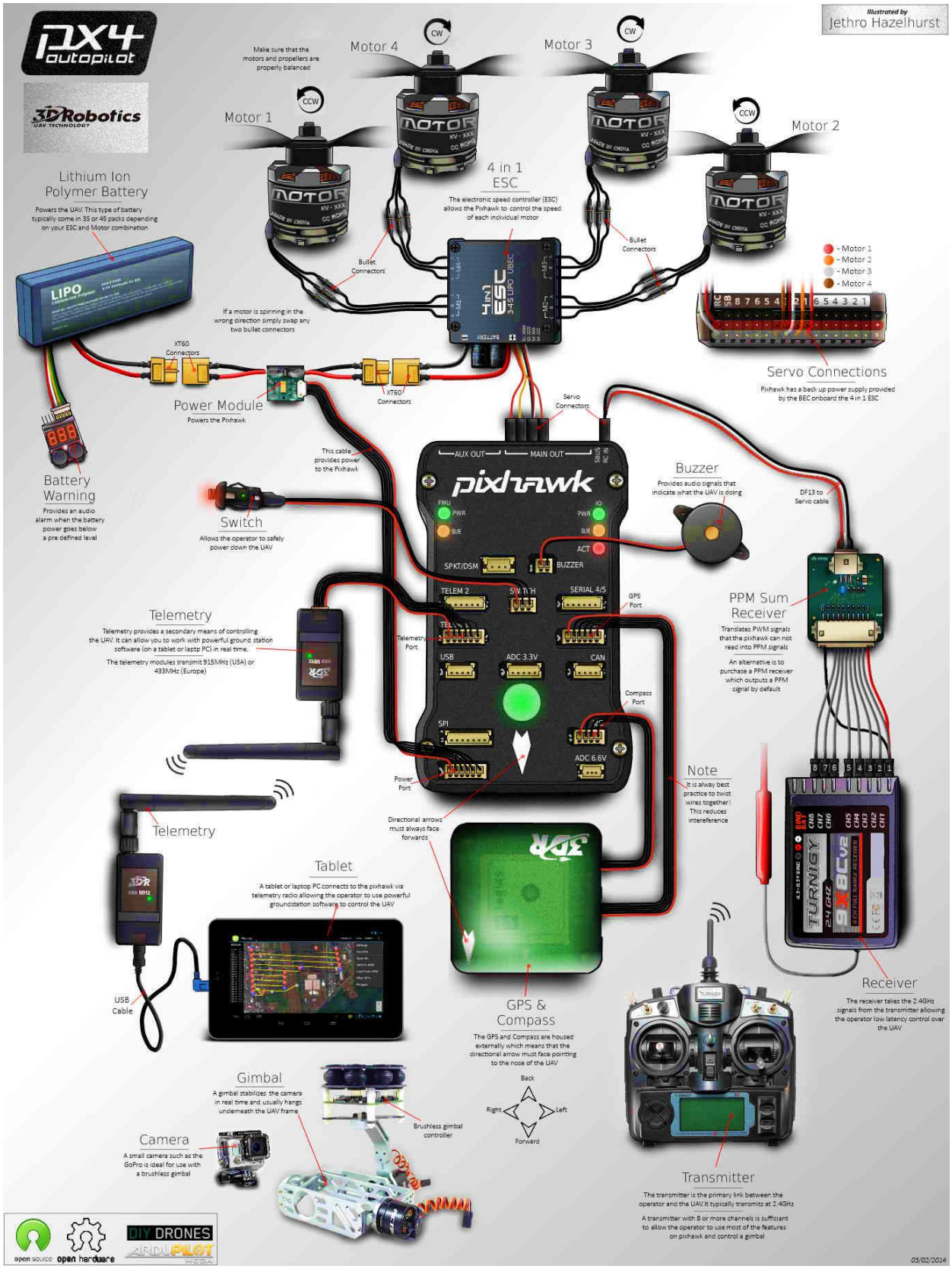


Figure 4.1: Diagram showing the typical PIXHAWK wiring and companion sensors (ArduCopter, 2017).

The following wiring modifications were made in order to reduce weight and match cable lengths:

- The ESCs' output leads were extended to better suit the QAV400 arms, and input leads cut to length for soldering to the Power Distribution Board (PDB) that constituted part of the quad's sub frame.
- The servo connections for the motors were shortened and only one BEC output wired to power the servo rail and provide a backup power supply.
- A custom FPV wiring harness created, that used the servo rail as power.
- The Power Module input leads were extended to allow greater free play in adjusting the battery's position to equalize the drone's CoG.

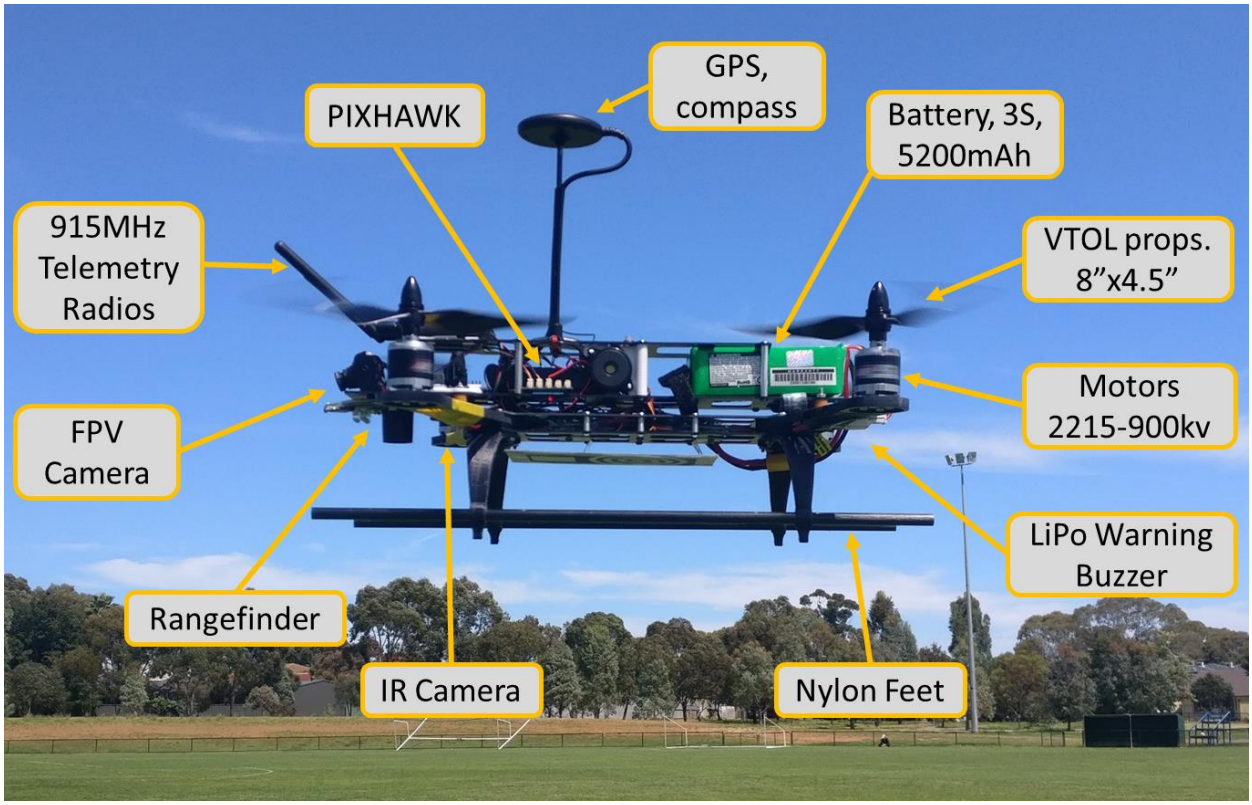


Figure 4.2: The completely assembled quadrotor in flight.

Some custom brackets were designed in addition to conventional fasteners, which gave greater control over the design features. The buzzer bracket shown in *Figure 4.3a* was essential to ensuring the safety switch could be pressed without danger from the propellers, while the flexible landing feet (nylon) was designed to absorb impact from hard landing, while allowing the fitment of a rod to enable landing on a net.

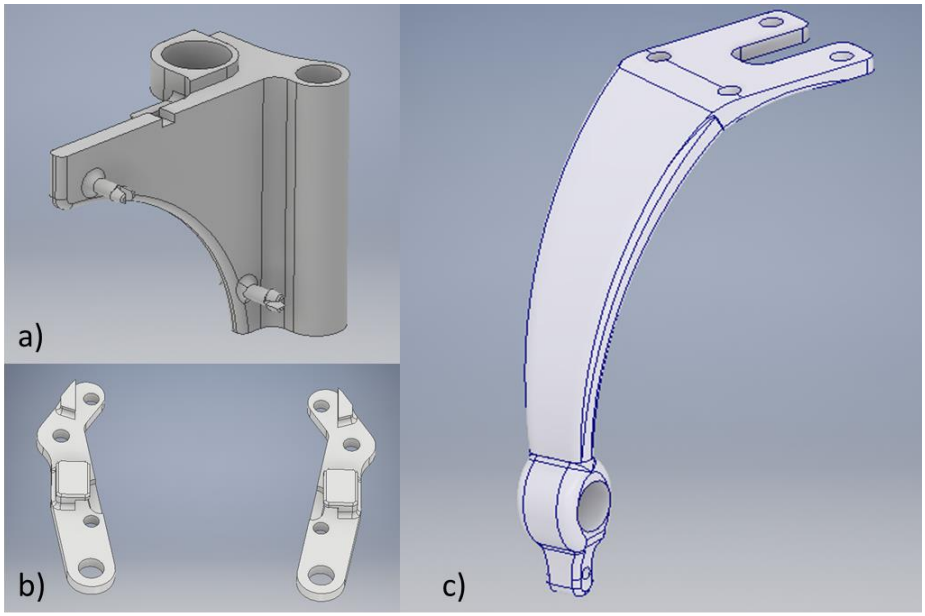


Figure 4.3: CAD model representations of custom brackets designed, a) Buzzer & Safety Switch mount, b) PIXY camera mount, c) Nylon landing feet suitable for landing on a net.

The PIXY mounting bracket ensured the downward facing IR camera was positioned such that its FOV was unobstructed by the drone, as illustrated in *Figure 4.4*.

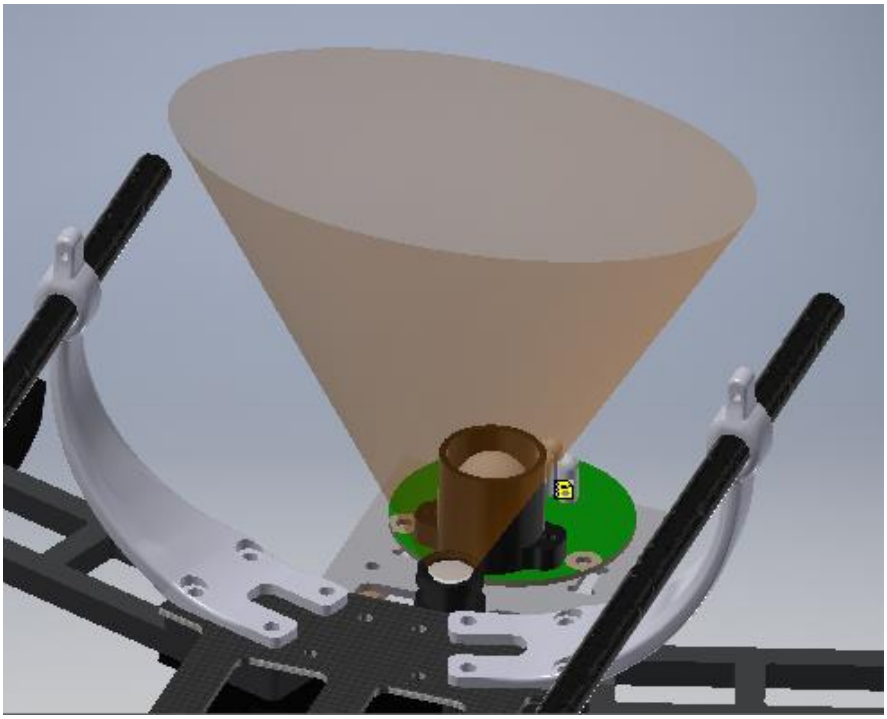


Figure 4.4: Snapshot of the Quadrotor's CAD model showing the PIXY camera's field of view unobstructed by the LeddarOne rangefinder (green), body frame of landing feet.

4.1.3. Drone Flight Tests

Before commencing the flight tests, extensive training was performed using the Mode 2 transmitter with both a Flight trainer as well as some mini drones.

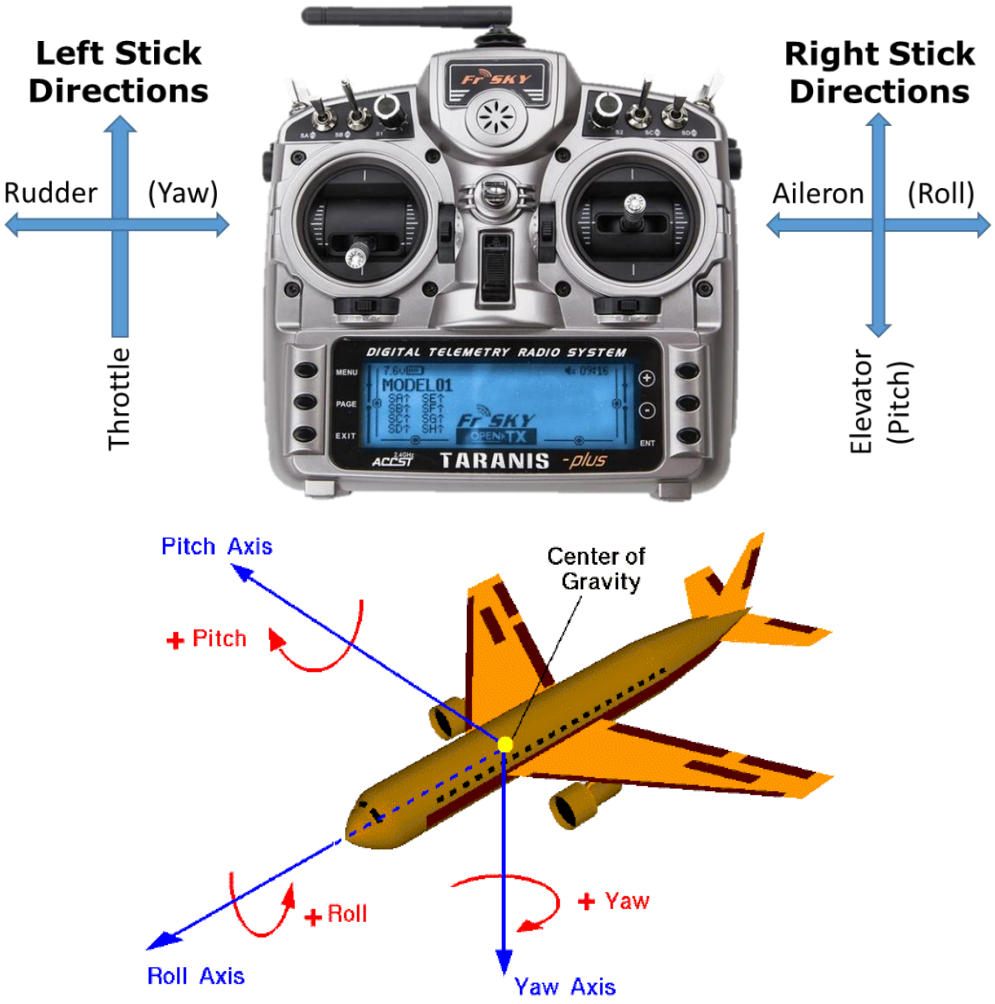


Figure 4.5: a) Mode 2 R/C Transmitter stick configuration and b) the corresponding aircraft rotations (Benson, 2017).

Bench Tests:

The basic power-up tests worked seamlessly, as did the pairing of the R/C transmitter and initial configuration of the flight controller. It was discovered though, that the 600mW, 5.8GHz transmitter used would heat up excessively, and was causing significant load on the BEC that was powering it. Consequently, a 100mW transmitter was employed that provided vision up to 100 meters, which was suitable for this application. Additionally, *Figure 4.6* shows the PIXHAWK FCU was mounted a sufficient distance from the electrically noisy speed controllers, since the overall interference of 9% is well below the 30% limit.

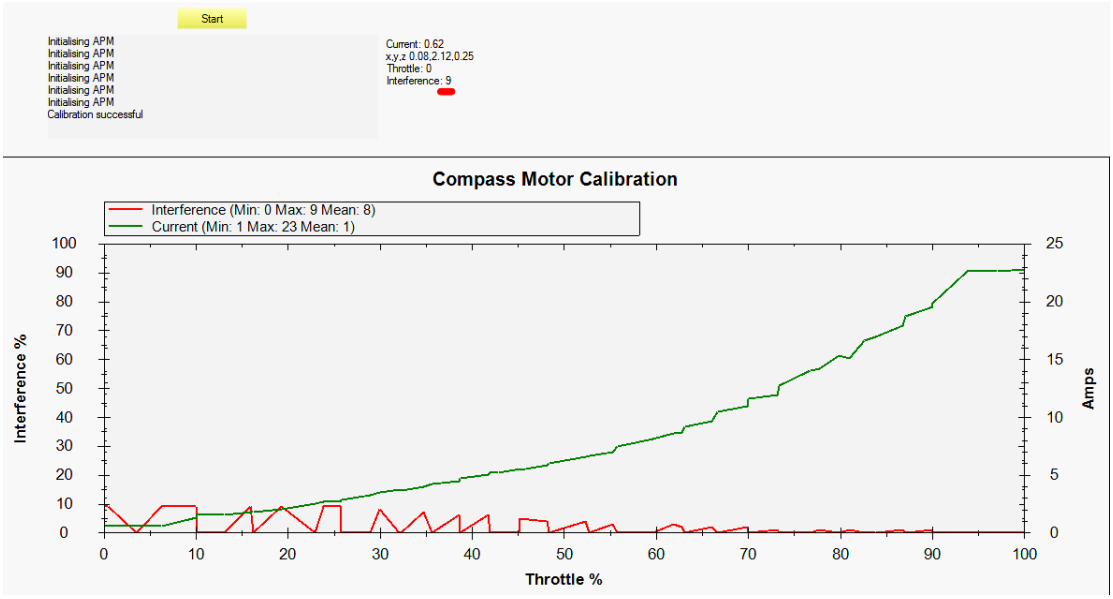


Figure 4.6: Compass/Motor Interference test showing the motor current (green) causes minimal interference (red) to the compass.

Manual Flight Tests:

The maiden flight was successfully performed late July, demonstrating the quadcopter was able to hover easily around 50% throttle using the Gemfan 8x3.8 propellers, which reinforced the calculations shown in *TABLE 4.2*. However, the default yaw PID values were too low, causing continual twisting during flight. Once outdoors, the auto tuning of the Roll, Pitch and Yaw PID controllers was performed, taking approximately 7-9 minutes on each axis, with the final results shown in *Figure 4.7*, and the improvement of the flight response shown in *Figure 4.8*.

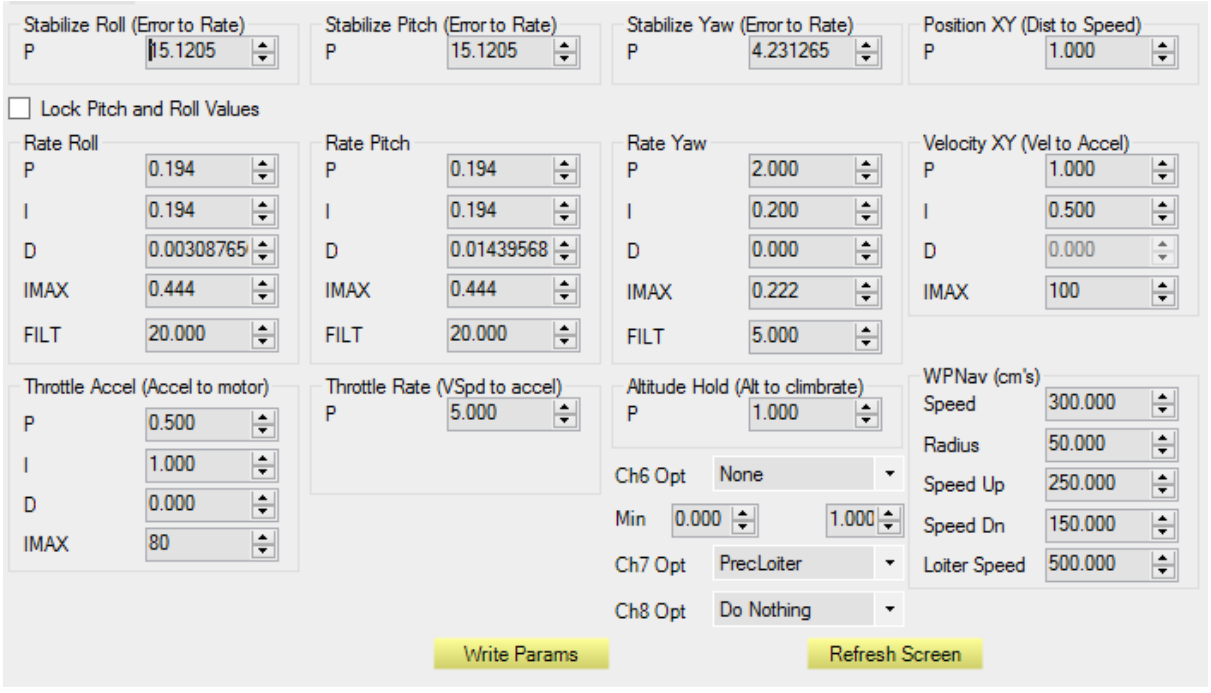


Figure 4.7: PID values after minor adjustments of the Auto tuned quadcopter.

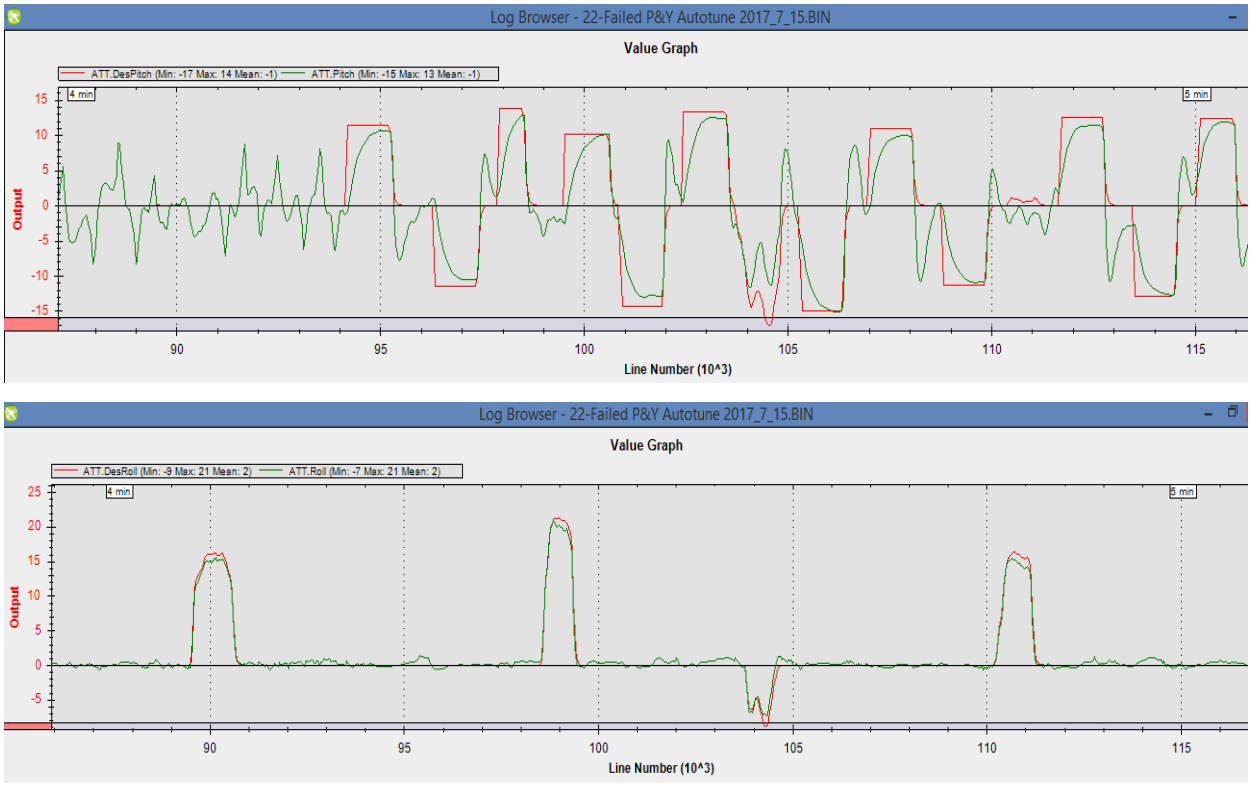


Figure 4.8: Plots showing the effect of the PID tuning. Top, the underdamped pitch angles have not been tuned yet, while bottom, the tuned roll angles closely track the desired roll.

However, after the auto-tune, the flight dynamics were initially observed to be very twitchy, due to the excessively high Pitch P & I gains of 0.69 that should range between 0.08 & 0.20. These were adjusted to the Roll gains. Early flight tests confirmed excellent position hold ability in LOITER mode in windy conditions up to 25kphr, and a flight time around 9 minutes with the 3000mAh batteries. Although the Gemfan propellers were functional, on a few occasions when the quadcopter was flying forward around $1ms^{-1}$, the drone would start a quick, controlled descent, which could not be countered even when the throttle was held at its maximum for over two seconds. However, when swapping to the Master Airscrew propellers specifically designed for VTOL aircraft, the flight response was more immediate, controllable and quieter, and consequently were used for the remainder of the testing.

The current load during flying during four consecutive flights is shown in *Figure 4.9*, revealing the highest current draw is during manual take-off, with the average current during flight between 16-17Amps. This test was using the 5200mAh battery, and demonstrates the maximum current draw of 24A is well within the battery's maximum discharge capacity of 52A (Eq. 1). Also, the average current draw per motor is approximately 4A as the FCU current draw is miniscule in comparison, well within the 20A capacity of the motor drivers (ESCs).

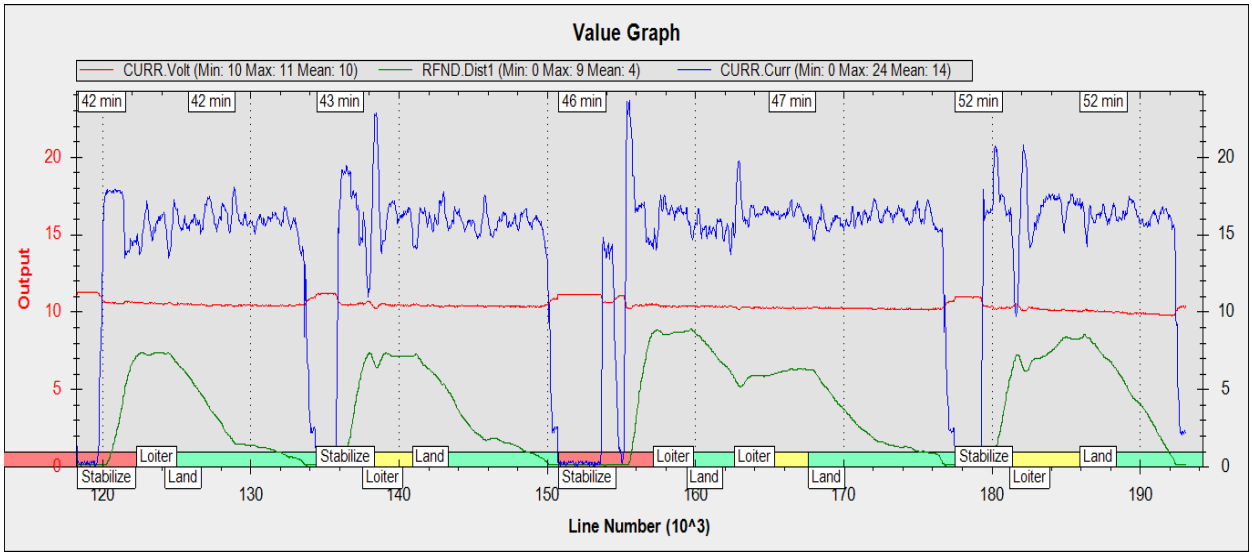


Figure 4.9: Total Current draw during four consecutive flights.

Autonomous Flight Features Validation:

The more advanced flight modes including ALTHOLD (indoors), Auto LAND, RTL and GUIDED modes worked very well using the Mission Planner software, demonstrating a controlled descent during LAND mode and a positional accuracy around 2m from the Home position in RTL mode. The AUTO mission mode of the PIXHAWK also worked well, using a Mission file loaded by the base station, Mission Planner, incorporating auto take-off, waypoint following and RTL. Additional details can be found in Swincer (2017)'s report.

When integrating the additional sensors chosen for this project, a few issues were encountered. Initial problems proving the LeddarOne sensor were overcome by using more recent firmware than ArduCopter 3.4.2 which included an updated driver. Even though the Pixy IR camera worked flawlessly using USB and the PixyMon configuration package, it failed to work using the I^2C interface with the PIXHAWK. Extensive testing included voltage level checks, debugging the I^2C interface and checking Bus ID configurations without success. The cause of the "*Bad Vision Position*" error was believed to be the Pixy firmware not recognizing the LeddarOne rangefinder that is essential for its operation. This meant that the final stage of landing would potentially stray from the target, due to the drone smothered of the upward looking camera's FOV.

4.1.4. Actual Specifications from the Drone's performance

The following table summarizes the actual specifications based on the drone's performance. It must be noted that the flight times listed are based on the battery being rundown to 10.0V (each cell voltage drops below 3.3V) equivalent to 30% remaining charge, but for normal flights, the pilot should consider landing when total voltage remaining is 10.4V. Additionally, the propellers' combined thrust is only based on specification only. It is proposed that this be the subject of a future study to quantitatively determine the best motor/propeller combination.

TABLE 4.4: Actual Specifications of the Drone.

SPECIFICATION	MEASUREMENT
TOTAL WEIGHT	1.425kg (w/ 5200mAh batt)
DIAGONAL SIZE	400mm
RECOMMENDED BATTERY	5200mAh, 3S LiPo, weighing 325gms
FLIGHT TIME (w/ 5200mAh batt.)	~13 minutes
FLIGHT TIME (w/ 3000mAh batt.)	~8 minutes
TOTAL THRUST	3.28kg
MAXIMUM PAYLOAD	(570gms – Battery Weight)
PAYLOAD USED	100gms

4.1.5. Landing Pad Design

The development of the landing platform design had the following features, to incorporate the vision system and enable landing on the TopCat vessel. A net (coated metal 15x15mm grid mesh) was planned to be offset from the payload tray of the vessel by 0.2m as shown in *Figure 4.10* as it would minimize the ground effect and ease construction. The camera is designed to be mounted under the centre of the platform on a gimbal to dampen rotational oscillation, and observe the quadcopter through an aperture in the netting. Its camera will be angled toward the rear of the vessel to ensure the quadcopter descent trajectory is furthest from the costly vessel's sensor array. For field deployment, angled guard rails should be fitted such as done by the RIVERWATCH team (Pinto et al., 2014).

For preliminary testing purposes, a second PIXHAWK was used to replicate the boat's position and heading, with a camera embedded in a flat board attached to a moving platform.

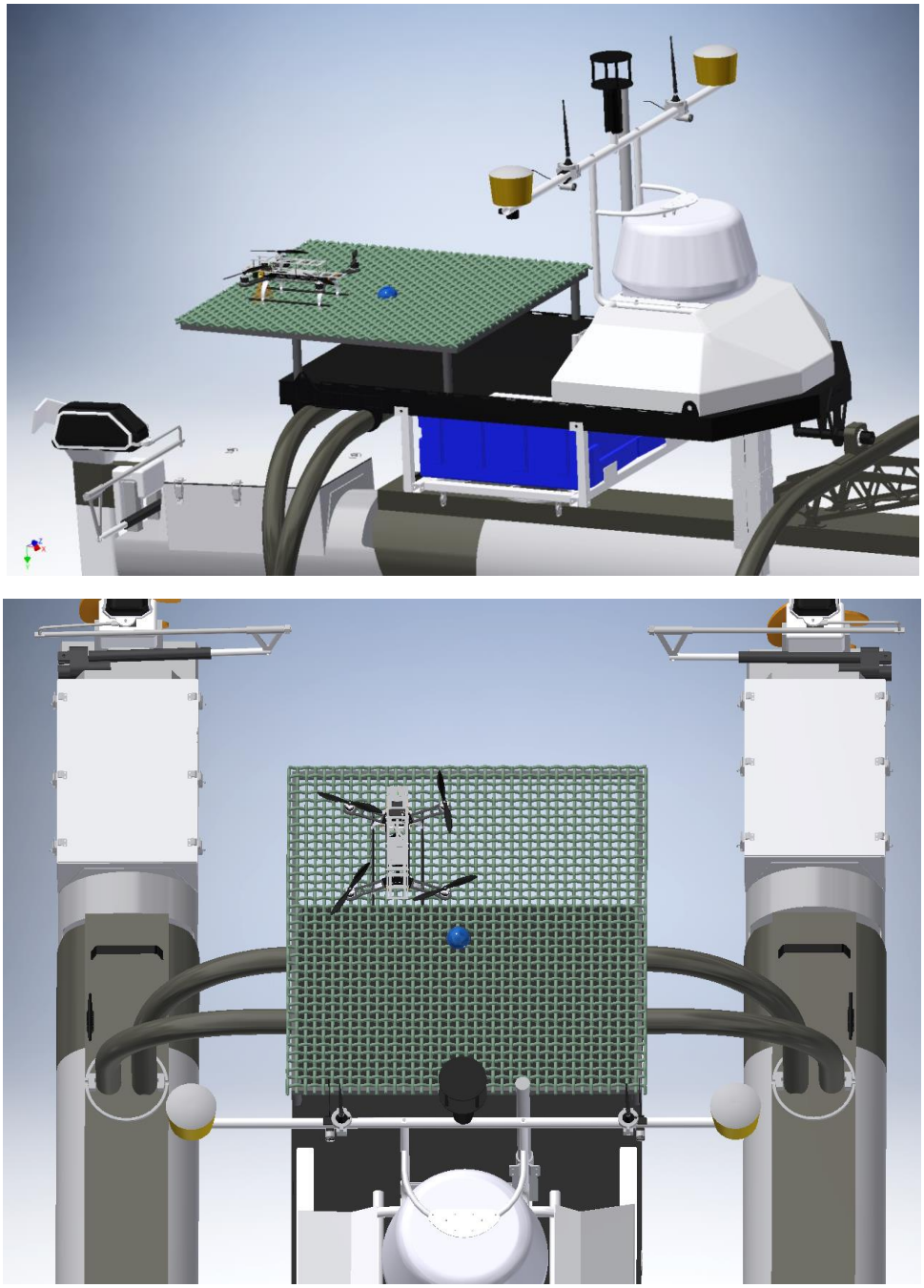


Figure 4.10. Proposed design of the 1m^2 Landing Pad on the TopCat vessel, showing the view from the side and from on top.

4.2. Software Development – UAV Control Implementation System

Given the functioning drone platform, the second phase to command the drone through software could be validated. The following sections outline the implementation process, the difficulties encountered and their resolution.

4.2.1. ROS to Flight Controller Link

The MAVROS package required minimal configuration using a COM port connection to the radio, providing the topics shown in *Figure 4.11*, in addition to a range of services and configurable parameters. Importantly, telemetry data stream rates had to be configured on start-up to ensure all the available flight data was available through the MAVROS node. MAVROS was installed from source rather than by binary installation, to improve portability and provide flexibility in customizing the package (see APPENDIX C**: Required Pre-requisites for Running the Software**). This was vital since the latest MAVROS package did not support the MAVLink Landing Target command, so a custom plugin was written for the MAVROS_EXTRAS package. This package proved to be a reliable interface with the flight controller, providing timely command feedback and the toolset needed to control the drone through software.

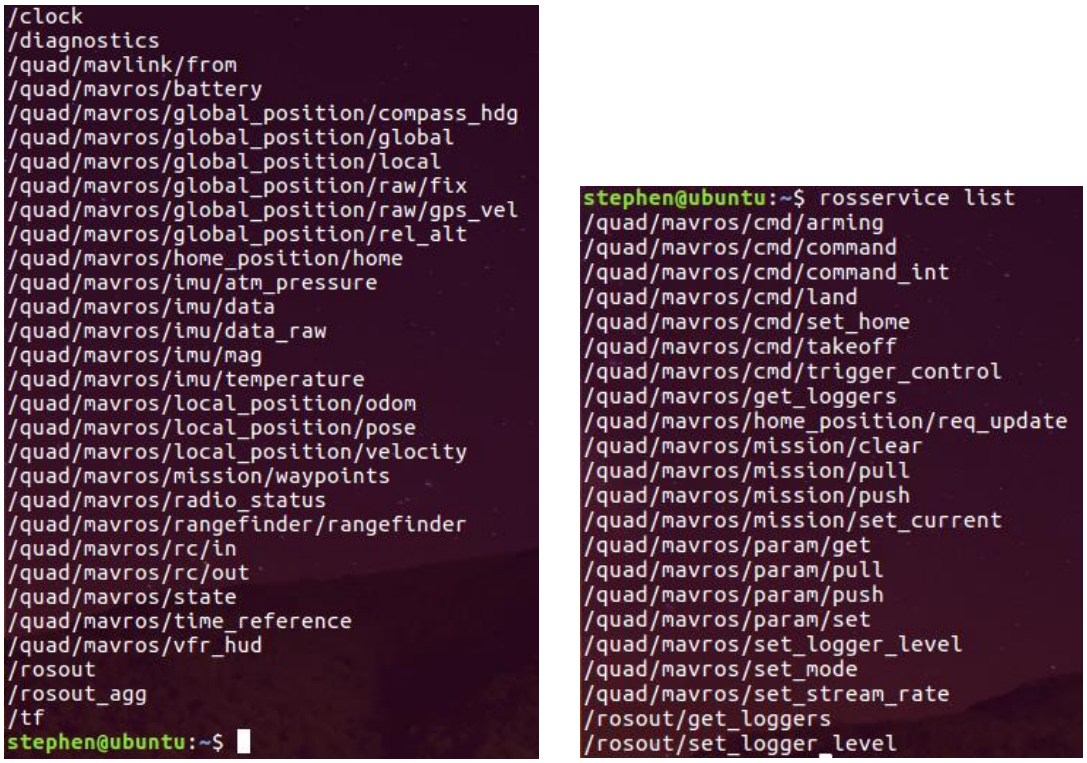


Figure 4.11: A Selection of the ROS Topics and Services available from the MAVROS package.

4.2.2. UAV Control Software State Machine

The state machine control method efficiently implemented the UAV Control software, providing convenient subdivisions for testing, while managing the sequential nature of the code. However, field testing revealed the necessity for the following modifications.

Firstly, closed-loop feedback in a number of subroutines was needed after it was discovered that a response of "success" from the execution of a MAVROS service merely indicated the command was sent, rather than the action completed. Consequently, subroutines were reconfigured to monitor the corresponding MAVROS topic and indicate successful if the desired change was seen within a given timeframe. From field testing, this delay was observed to vary between 1.5-2.8 seconds depending on the distance to the aircraft affecting signal latency, in addition to the telemetry data update rate.

Secondly, the following changes were made to the state machine, in order to remove unnecessary delays and improve integration with the other group members' work.

- A status indicator was used within each state to ensure completed functions were not repeated, whilst providing feedback regarding the progress throughout each state.
- The TAKEOFF state changed to include the ARMING command, as the take-off command needs to be sent within 5 seconds of the ARMING command.
- A WAITING_FOR_MISSION_CMD state was added, in order to handle the situation where a state was finished but no decision regarding the next state had been made. This was crucial for software control of the UAV Control node.
- The VISUAL TRACKING state was changed to PRECISION LANDING, so that the final landing stage could be run immediately without the need to wait for Top Level Mission Commands. Consequently, the LAND state was changed to EMERGENCY, in the event the guided landing failed and a conventional landing at a specified GPS location was required.

The most difficult state, Precision Landing, was achieved through the use of the LANDING_TARGET command, which is accepts target offsets from a drone mounted camera. However, in order to use this command with an upward looking camera, a transformation was necessary to correct the target angles from the vessel's camera into the corresponding target offset angles as seen by the drone's downward looking camera. Full details on how the target offset angles were computed from the captured image are covered in the Visual Guidance report (Prasetyono, 2017). However, this transformation can be simply understood when considering the

surface vessel and drone are heading in the same direction. Thus, when the vessel's camera observes the drone behind and to the left of the vessel, the corresponding camera on the drone would observe the vessel is ahead and to its right, as shown in *Figure 4.12a*. Therefore, the transformation can be simply defined in Eq. 2, assuming the boat and quadcopter are aligned.

$$Basic\ Transformation: \quad \begin{bmatrix} x_{quad} \\ y_{quad} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_{boat} \\ y_{boat} \end{bmatrix} \qquad Eq.2$$

In reality, the boat and quadcopter oscillates independently in 6 degrees of freedom (DOF) and consequently the yaw, roll, pitch variations should be fused with the above transformation. This was achieved using the ROS transformations library, *tf2*, that manages the relationships between multiple coordinate frames (TeamUSRG, 2016). Although a powerful tool, this method was not reliable for precise landing on a target, due to the compounding uncertainty between two 2.5m accurate GPS sensors.

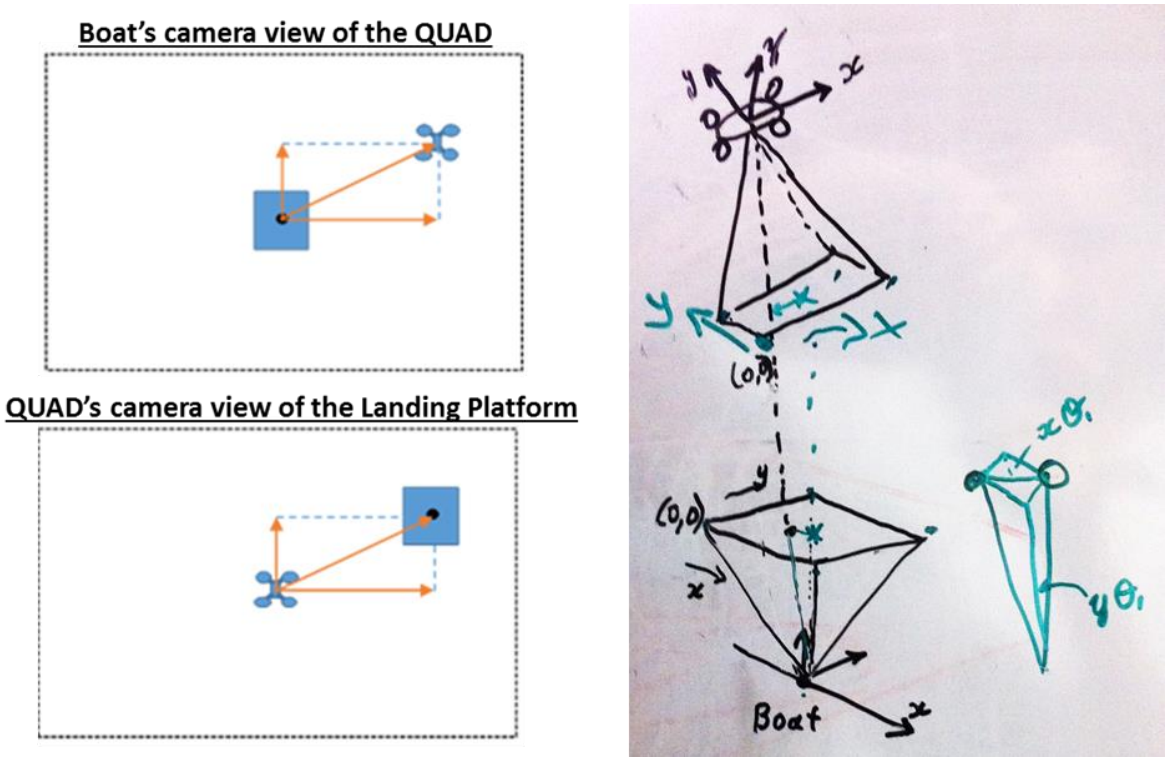


Figure 4.12: Simplified Transformation (left) between the vessel's camera frame and the corresponding downward looking camera frame fixed to the drone's body (Prasetyono, 2017). Right, the complex transformation possible in 6DOF.

However, due to the advanced stabilization control of the drone, the roll and pitch of the quadcopter is minimal, while a passive gimbal can minimize the disturbance to the upward looking camera from the vessel's rocking. Therefore, Eq. 3 shows the practical transformation that

accounts for the difference in heading between the quadcopter and the vessel, converting the upward looking camera angle offsets to the correct offsets as seen from the drone's camera.

$$Yaw\ Compensation: \quad \begin{bmatrix} x_{quad} \\ y_{quad} \end{bmatrix} = \begin{bmatrix} \cos\psi & \sin\psi \\ \sin\psi & -\cos\psi \end{bmatrix} \begin{bmatrix} x_{boat} \\ y_{boat} \end{bmatrix} \qquad Eq.3$$

$$where\ \psi = Heading_{quad} - Heading_{boat}$$

This transformation relies on a sufficiently accurate heading estimate, which was calculated using the GPS heading in this project (see the next section's end), but future work should explore determining the relative heading using the camera data for increased accuracy.

4.2.3. Test Results of Key States of the UAV Control

Before commencing field trials, the core functionality of the code was proven using a standalone PIXHAWK as Hardware-in-the-Loop Simulation. More rigorous simulation using the UAV RotorS Simulator in Gazebo was explored, but discarded due to the long learning curve which did not fit the project goals and timeframe. The positive results already achieved gave confidence to proceed. Subsequent field trials were performed using the R/C transmitter as the manual override, with the key learnings from each state presented below.

PREARM:

The configuration of the parameters (see APPENDIX A**: Configuration Parameters File for PIXHAWK**) worked well, however, the writing of the Mission Waypoint file was significantly affected by the distance between the quadrotor and the commanding computer due to the large transmit bandwidth required. Therefore this file was written while the quadcopter was on the landing platform and before the flight data stream rates were set, which were offset to reduce the peak load on the telemetry bandwidth.

TAKEOFF:

Early tests revealed great difficulty in commanding the drone to both arm and take-off. This was because the same telemetry channel was being used for both feedback to the base station and the MAVROS node control. Separating these into two independent channels from the flight controller greatly improved the software command success rate and the operating range.

The software controlled take-off, although working, displayed some interesting characteristics that should be considered when implementing this on the vessel. Firstly, the aggressiveness of the take-off is in proportion to the target altitude and is inversely proportional to the magnitude of drift

observed. Additionally, setting the WP_NAVALT_MIN parameter to 0.5m reduced the drifting from the wind by allowing navigational correction from 0.5m rather than once at the target altitude, reducing the earlier sharp jerky correction observed earlier, as shown below. Therefore, the vessel needed to be pointed into the wind during take-off to ensure the quadcopter ascended away from the expensive sensor array on the vessel, while the take-off command would direct the drone to a safe location behind the vessel. In total, six successful automatic take-offs were achieved from over ten tests, with the main errors being random communication errors that prevented take-off but once in the air, the flight controller consistently commanded the drone to the target altitude, even recovering from wind gusts during the process, as shown below. It must be noted, that the R/C throttle should be set to 50% before take-off in order to avoid rapid loss in altitude once the drone switches to LOITER mode.

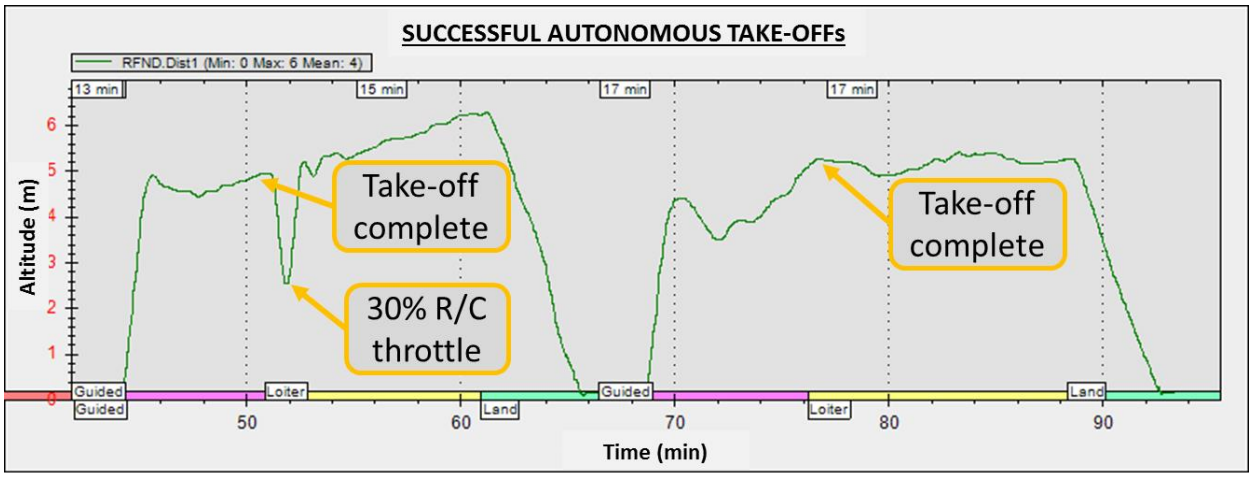


Figure 4.13: Two autonomous take-offs in mild winds, demonstrating self-recovery of altitude.

MISSION & APPROACH LANDING:

As the PIXHAWK efficiently managed the navigation of the quadcopter through a series of waypoints, the software control simply monitored the progress of the drone through computing the relative distance of the drone from the target waypoints using the Proj4 Library. However, when approaching the landing target, the set of precomputed waypoints from the Top Level Mission Planner node would be dynamically loaded, to account for changes in the vessel's orientation. Difficulties trying to cause the flight controller to respond to the sent waypoints have not been solved to date, and will require further investigation.

PRECISION LANDING:

The use of the LANDING_TARGET effectively implemented the precision landing state as described below, but several enhancements are required needed to consistently land within one

square meter before deploying on the vessel. Preliminary testing of the landing commands was performed using the Precision Loiter capability, available in the latest release of ArduCopter v3.5.x, allowing the offset corrections' transformation to be verified before commencing the landing. Testing demonstrated the camera feedback successfully guided the drone toward the centre of the camera's FOV, even in mild winds (<10kphr) and when the camera was angled, only when switched to LAND mode as shown in *Figure 4.14*. The graphs have been aligned for comparison, with the blue line indicating when the target was target was visible and the third graph showing the drone's corrections to minimize the angular offsets to the target only when in LAND mode.

However, initially it was observed that the quadcopter would consistently halt its steady descent around an altitude of 1 metre, performing minor adjustments while hovering up to 20 seconds, before completing the landing process. Adjustments were made to limit the smallest angle correction to more than 5° and to stop sending correction commands during the last meter without avail. Eventually, it was discovered that leaving the R/C throttle at 50% during the landing process was contributing to the delayed descent, so by gradually lowering it to zero through the last few meters minimized the delay, as shown in the second graph in *Figure 4.14*.

Additionally, the landing guidance experienced drifting during the final meter descent due to the ground effect and the prolonged camera occlusion when the drone was very close to the camera as indicated in *Figure 4.14c*. This was expected due to the PIXY camera not working, and was the primary cause for the large number of close landings shown in *TABLE 4.5* and should be rectified in future by tracking a smaller target on the underneath of the drone (refer to Prasetyono (2017)'s report) or resolving the issues with the IR Lock hardware. It was observed that by changing the landing platform to a net, offset 0.5m from the ground dramatically improved the success rate, as discussed in Section 4.3.

Nonetheless, *TABLE 4.5* demonstrates the work to date has developed a proof of concept precision landing sequence that has a 30% success rate in calm weather, and a potential 80% success rate when the issues with the last meter are resolved.

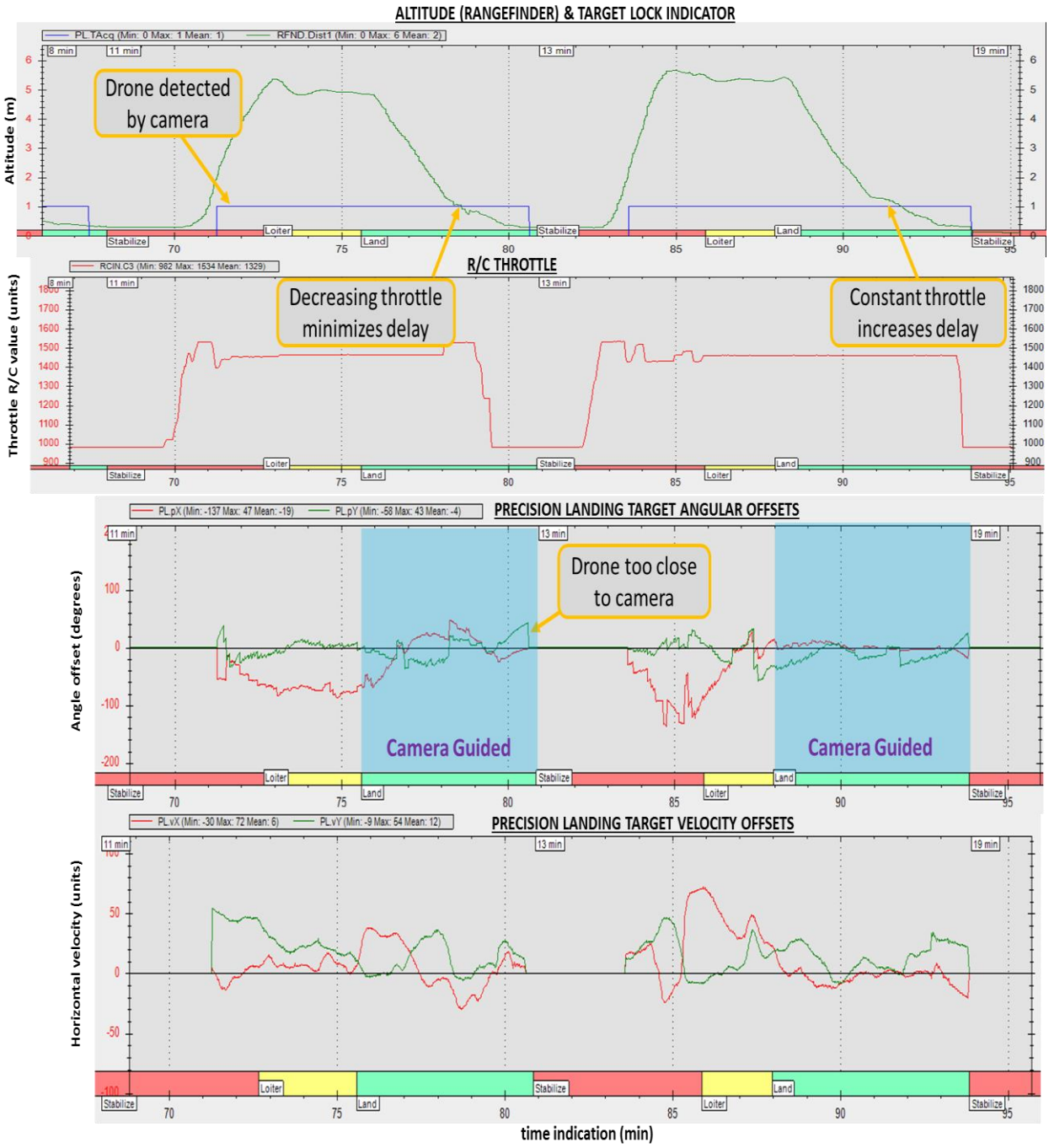


Figure 4.14: Flight data showing two consecutive precision landings on a one square meter target. The altitude during flight and indication of whether the target can be seen (top), the manual throttle input (top-middle), the angular offsets during LOITER and LAND mode (bottom-middle), and bottom, the horizontal velocities in the drone's camera frame due to the angular corrections.

TABLE 4.5: Summary of Precision Landing Flight tests.

FLIGHT TEST OUTCOME	QUANTITY
SUCCESSFUL LANDINGS (< 1m from camera)	9
CLOSE LANDINGS (< 2m from camera)	15
LANDINGS FURTHER THAN 2m	2
ABORTED LANDINGS	4
SUCCESSFUL LANDING RETRIES	5
MAXIMUM DESCENT TIME	~ 60sec
MINIMUM DESCENT TIME	~ 20sec
TOTAL LANDING TESTS	30

During subsequent tests, the following observations were made:

- When the quadcopter commenced its descent approximately centred in the camera's FOV and maintained this relative position throughout the duration of the descent, then minimal delay was noticed at the 1 meter mark.
- When the confidence level of the target lock was low either due to temporary occlusion of the drone or it being outside the camera's FOV, the drone recovered better when the landing target commands continued to be sent, as this continued to command the drone in the direction the target was last seen. However, if the target lock was lost for more than two seconds, then the software was configured to abort the landing and reattempt the landing from the Bailout waypoint.
- While descending into medium strength winds, the attitude lean due to the correction commands was sufficient to maintain a steady position within the FOV, but not reduce the correction angles to enough to permit further descent, therefore hovering indefinitely.

Further testing was performed to explore the effect of increasing the frequency of angle corrections from 5Hz to 10Hz. Even only sending the latest correction from the Visual Guidance node resulted in oversensitive drone's behaviour, which was irrecoverable when moving downwind. A suitable compromise was reached with 7Hz sending rate, with future work to explore if not sending altitude readings would dampen the responsiveness at higher altitudes.

Furthermore, the capability of the current system to track and land on a moving target was demonstrated by the quadcopter following a walking target in the Precision Loiter mode, as shown

in *Figure 4.15*. Although a suitable moving platform has not been developed yet, this test indicates the system is capable of handling to slow moving targets.

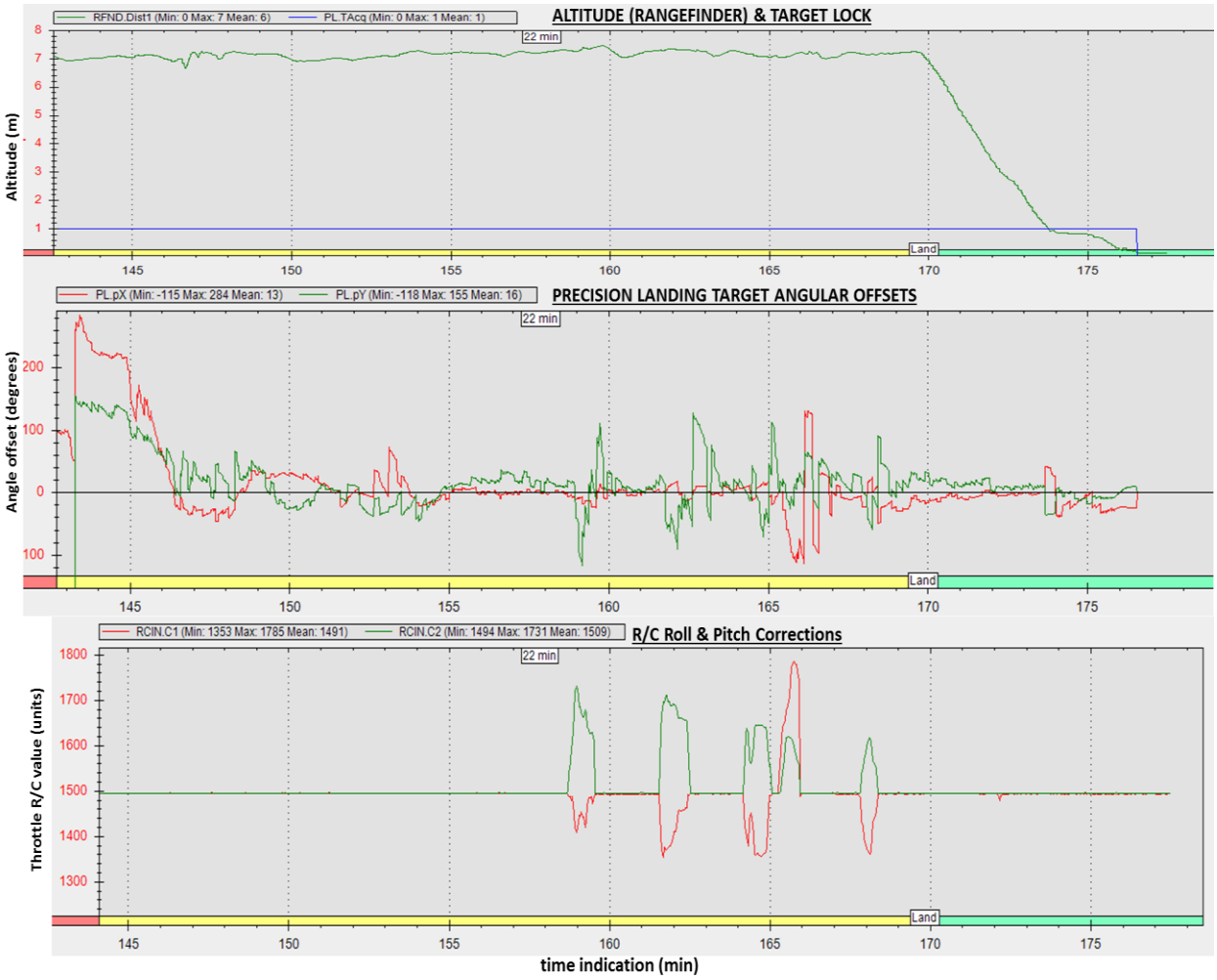


Figure 4.15: Quadcopter following a target moving at walking pace in Precision Loiter mode using the upward-looking camera and control software. Note, the minimal R/C roll and pitch corrections (bottom) and the stable altitude hold in Loiter mode (top).

This project used the heading from the GPS to test the transformation shown in Eq. 3 for compensating for the heading difference between the quadcopter and the boat, which was simulated using a second PIXHAWK FCU, shown in *Figure 4.16*. The initial errors between the corresponding GPS headings were quite significant (>30°), requiring calibration when the physical alignment was known. The drift, however, was found to be less than 8 degrees within a typical flight time of 10 minutes, which was deemed sufficient for this application. Preliminary field tests however, were unsuccessful, with the copter flying off in the wrong direction. However, the transformation in Eq. 2 was robust to heading offsets up to 10°, as was the case in the final landing shown in *Figure 4.14*.

4.3. Overall Software System integration

Having developed the component systems, this section outlines the results of the combined system. Firstly, the combination of the Visual Guidance for Landing with the UAV Control software functioned very well, as described in the Precision Landing section above. However, the integration with the Top Level Mission Planner is still in development, requiring the addition of a Waiting state to allow programmatic control of the UAV control state machine. Further work will be required to fully implement and test the Top-Level Mission Planner's control, involving programmatically handling any errors that are reported in real-time.

Field trials to date have confirmed correct operation of initialization, take-off, mission and precision landing states, with the remaining states requiring additional work before working towards complete autonomy. The field setup of the landing deck is shown below, with several captures of the landing process shown in *Figure 4.17*.

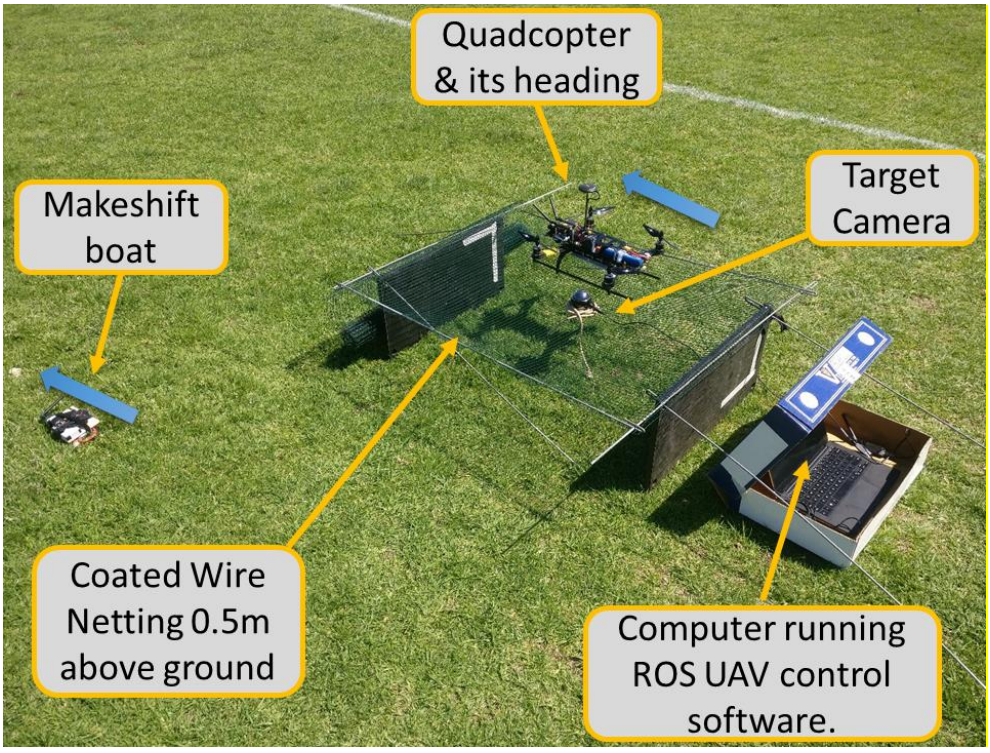


Figure 4.16: The field trial landing deck of dimensions 1.1m by 0.9m.

Further testing is required for landing on dynamically moving platforms similar to marine environments.

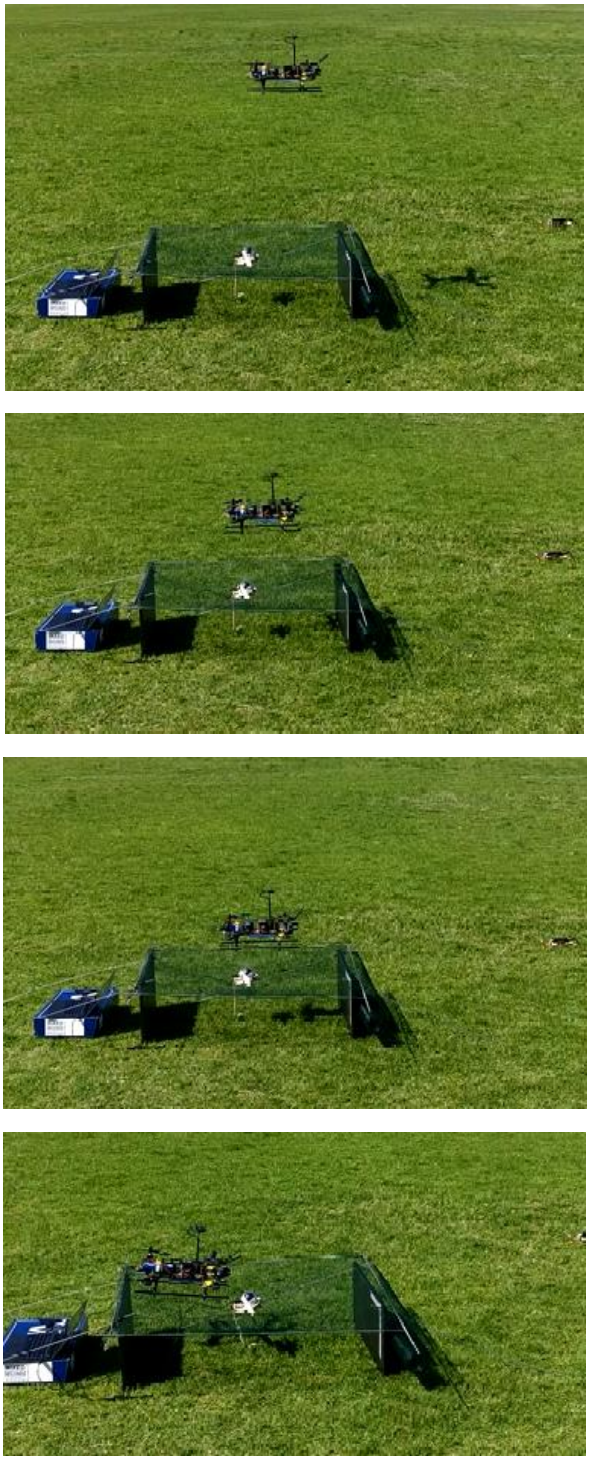


Figure 4.17: Precision landing on a stationary net platform in mild winds.

The effect of the final stage drifting is seen in *Figure 4.17*, caused by the chaotic winds generated by the quadcopter descending into its own wash close to ground and called the ground effect. However, the drift seen here is much reduced compared with landing on a solid platform, due to the open platform design enabling the easy escape of pressurized air.

4.3.1. Handling Emergency & Failsafe Conditions

The largest barrier to achieving full autonomous deployment and recovery of the quadcopter is the handling of field-based errors, including the effects of environmental disturbances. Although semi-automatic functionality has been achieved in select stages of this project, extensive thought and testing is required to handle safely the possible errors.

However, much of detection and response to quadcopter faults is capably managed by the PIXHAWK flight controller, including fail-safe actions, unstable flight position recovery and communication loss mitigation strategies. This limits the scope to responding to the reported fault and overseeing the quadcopter response and informing the autonomous vessel. This is beyond the scope of this report, but in the event of a failure or persistent error detected, the following states have been included and do require field testing in future.

BAILOUT:

If an unsafe condition was detected or errors experienced during the landing process, this state would command the drone to a safe waypoint behind the vessel, as specified by the Mission Planning node.

EMERGENCY:

This state used the dynamically calculated closest safe bailout position from the Mission Planning node to update the saved home position of the flight controller during flight. This meant the RTL mode could be used to redirect the drone to land on the nearest shoreline if required. Only preliminary testing of this functionality has been performed indicating decent RTL performance, but dynamic home updates has not yet been implemented.

5. DISCUSSION

Having described the individual results in detail, the following section provides a high level assessment of the work performed, including the opportunities for future enhancements.

5.1. Evaluation of the Drone Design

The development of a drone platform proved to be a success for the following reasons. Firstly, the time invested in selecting components proved invaluable as the drone assembly worked as planned and the drone performed very well in field trials. The resultant payload and capabilities of the drone matched the expected specifications used during the design phase, leaving sufficient capacity for future expansion. Except for the PIXY IR camera, the additional systems fitted worked as expected, providing extra functionality such as FPV monitoring for future applications. Secondly, the design maximized the use of existing technologies to develop a sturdy platform within the budget constraints of $1800AUD, with the complete drone costing under $1200 and related equipment and spares costing $550. Thirdly, the quadcopter had excellent flight performance, demonstrated by its ability to hold position in medium gusty winds (<20kphr), rapid response to pilot control and stable maneuverers. Overall, the decision to design and build a custom drone was vindicated by the robust, modular, platform developed that not only formed a reliable test bed for this project but will provide a good basis for future aerial system research.

5.2. Evaluation of UAV Control Software

The development of the software control, although not finished, has successfully demonstrated the ability for the full autonomous control of the drone. The challenging task of interfacing external software with the existing on-board flight controller was implemented using the effective state-machine software structure, enabling component wise testing and streamlined integration. However, the major lesson discovered was that extensive software compensation for temporary faults in the Vision Guidance system for example, was not constructive and even unsafe at times, since it caused jerky flight corrections that were unpredictable. Thus, implementing software monitoring with a slower refresh rate (<1Hz) was far more effective, since the PIXHAWK flight controller capably managed the low-level control.

A large portion of the system has been developed and verified using field testing, as seen in *Figure 5.1*, but a couple of states require additional work. Chiefly, the Approach Target state is not fully functional due to difficulties encountered with sending dynamically generated waypoints. The drone failed to change its position after successfully receiving a waypoint over MAVLink,

although the GUIDED mode functionality had been proven using the Mission Planner software. This hampered the progress of the BAILOUT state that relied on a dynamic waypoint to specify a safe recovery position for the drone. The remaining work for the Precision Landing state includes resolving the drifting during the final meter descent and verifying the performance on a moving platform before integration with the TopCat ASV. To date, the EMERGENCY state has not been completed, though its implementation is straightforward.

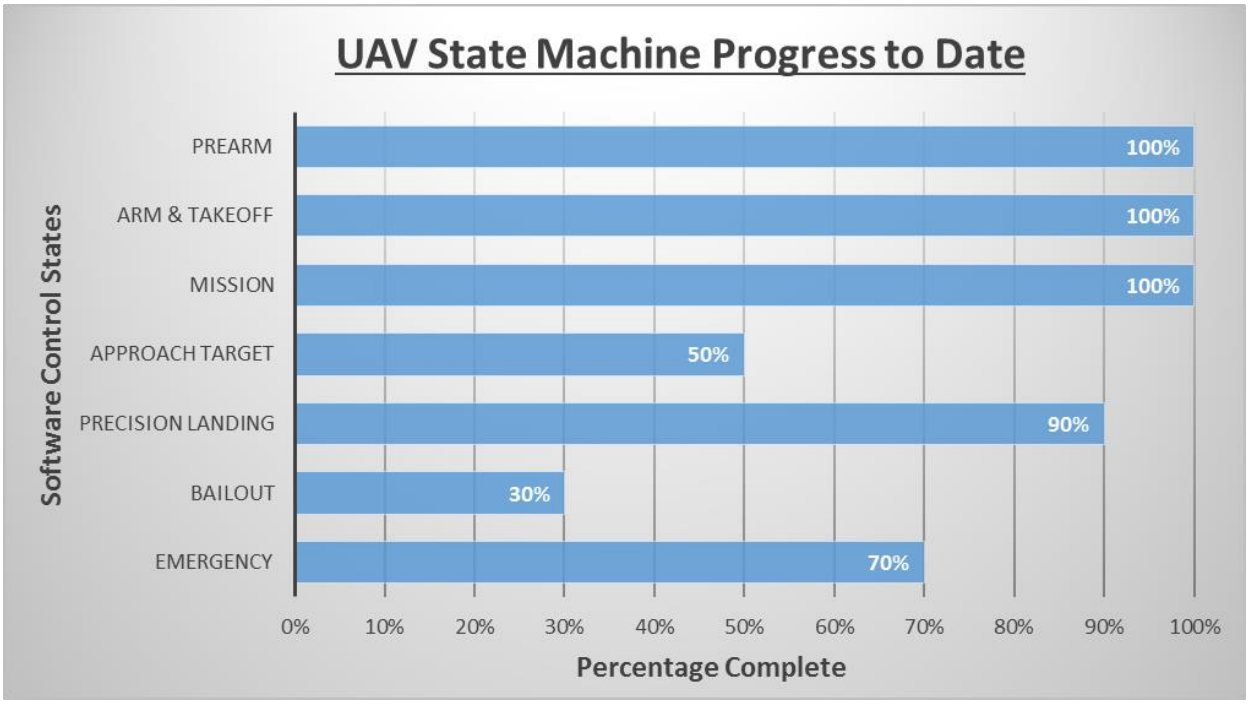


Figure 5.1: Progress update for the Software Control of the Quadcopter.

TABLE 5.1: Estimated work status given the relative complexity of each state.

Software Control States	Complexity Weighting	Weighted Average
EMERGENCY	10%	7.00%
BAILOUT	5%	1.50%
PRECISION LANDING	40%	36.00%
APPROACH TARGET	20%	10.00%
MISSION	10%	10.00%
ARM & TAKEOFF	10%	10.00%
PREARM	5%	5.00%
		80%

5.3. Evaluation of Overall System Performance & Significance

Much of the individual modules of the software control have been field tested, with successful integration of the Visual Guidance and UAV Control components (see Section 4.3), but significant work remains to integrate and fully test the Top-Level Mission Planner with the UAV Control in order to achieve entirely autonomous control of the take-off and landing of the quadcopter. Firstly, some states still require proving in field trials, while others occasionally experience errors from communication faults or "PIXHAWK Busy" feedback. Therefore, the programmatic response to persistent errors from the flight controller needs to be incorporated in future. Secondly, the state-machine transition logic needs to be comprehensively validated in real-time whilst controlling the UAV Control node before fully autonomous missions can be commenced.

Despite not attaining the ultimate goal of landing the quadcopter on the autonomous vessel, significant progress has been made towards this end. Specifically this project, in conjunction with the companion projects, has developed a reliable drone platform that is capable of autonomous take-off, mission following and landing on a stationary target in various lighting conditions. The current configuration is expected to work unmodified on a horizontal moving platform, supported by positive precision loiter tests, and doubtless, with some enhancements, the system presented above will achieve autonomous landing on a dynamically oscillating platform. But first, the following limitations of the existing solution should be understood, with recommended improvements listed in the ensuing section:

- The landing accuracy is currently within 2 meters, but can be improved,
- The current system was tested with a single drone in the camera's vision,
- The landing system is reactive rather than predictive of the target's motion, and
- The flexibility of the quadcopter is restricted, given the control software runs on the vessel.

This work has laid the foundation for the UAV & ASV to cooperatively complete missions, whilst providing a launching pad for future aerial vehicle research.

5.4. Recommendations for Future Research

The following recommendations have been identified to address the identified limitations of the presented solution and provide direction for future research. With regard to the constructed drone platform, the following improvements are recommended:

- Arm mounted Beacons (LEDs) be attached for visual heading indication, ideally pulsing when the battery voltage is low.
- Design a waterproof shell for the drone and coat the electronics with conformal coating.
- Incorporate obstacle avoidance sensors for dynamic path planning modifications.
- Verify the thrust & efficiency of the propellers with the chosen motors, thereby quantitatively determining the best combination.
- Explore linking the MAVROS package directly to a GCS on Linux, in order to eliminate one of the radios on the Base station computer.
- Lock down mechanism to secure the drone to the net when landed and before take-off.
- Hot swapping batteries – enabling repeated deployment of the quadcopter, possibly using a similar setup to LEE, D., (2015).

The recommendations to develop the Software Control include:

- The separation of the TopCat's and Quadcopter's software into two independent ROS masters, utilizing ROS bridge for communication, giving much greater redundancy in the event of failures. This enables the fitment of a companion computer to the Quadcopter, to process the on-board vision, reducing latency, and enabling standalone operations. See TeamUSRG (2016) for a recent example of this implementation.
- The current software can be improved by developing a class-based structure from the subroutines developed. Additionally, the code should attempt repeated attempts at sending FCU commands if errors encountered, before reporting back to the Mission Planner Node, as a first step in programmatically handling errors.
- Integration with the TopCat's system, particularly enhancing the vessel's Mission Planner to manage the quadcopter, whilst handling emergencies (e.g. E-Stop button pressed).

Future work can explore determining the relative heading of the drone with respect to the vessel by extracting the drone's orientation from the camera, thereby removing the need for compass offset calibration and drift mitigation. This could be done by detecting an AR Tag on the underside of the drone to correct the GPS heading offset error, but this would be difficult to distinguish at

7m altitude, when the landing commences. A better estimate of the relative heading offset would be gained by fusing this data with that of the GPS sensors, and the recorded heading of the quadcopter as it is aligned with the boat during the approach path.

Additionally, the cyber security of the overall control system should be considered, either using encryption on communications between two ROS masters, or configuring a MAVLink message to decrypt other standard message types.

Lastly, the current landing routine tracks the platform, but superior implementations would predict the trajectory of the target and navigate to the expected location before arrival. See TeamCTU-UPENN-UoL (2016) for further details. Such implementations also incorporate the disturbances due to the wind conditions in their landing control system, and the deck's vertical oscillations Yang et al. (2008). This would be highly recommended as the current system can only tolerate mild wind conditions during landing.

6. CONCLUSION

The foundational work toward delivering fully autonomous take-off and landing system for an unmanned marine vessel's aerial platform has been presented in this report. Specifically, this report described the development of the aerial platform capable of autonomy and the software control integration, in particular, the element responsible for sending commands to the quadcopter. The project utilized the learnings from the growing research field, providing a scalable drone design that maximized the use of existing technologies, and implemented a modular software control that has undergone extensive component-wise field testing. Although the ultimate goal of autonomous landing on a moving vessel has not been reached, this and the companion projects' prototype system has achieved the following within the budget constraints:

- Designed a reliable drone platform that is capable of supporting future research needs,
- Developed a working prototype control software for the autonomous deployment and recovery of the quadcopter, intended for marine applications,
- Demonstrated autonomous take-off, mission following and target tracking in practical flight trial, and
- Delivered a vision based Precision Landing algorithm capable of landing on a one square meter target in both fair and overcast conditions.

Immediate future works involve proving followed by refining every state of the software control and completing the system integration and field testing of the Top Level Mission Planner. Also, the incorporation of the target motion prediction and environmental disturbance into the precision landing control loops is essential for robustness. Further research areas include utilizing this platform to classify the observed environment by the quadcopter for improved mission planning of the autonomous marine vessel.

Together, the body of work presented in these three complementary reports has brought together the knowledge surrounding autonomous systems, the extensive work performed by the ArduPilot developers and the powerful ROS platform, to present a proof of concept system for the autonomous launch and recovery of UAVs. This provides a firm basis for the heterogeneous cooperation of UAVs and ASVs to complete surveillance missions, ushering in numerous opportunities future autonomous aerial vehicle research.

References

Ajmera, J, Siddharthan, PR, Ramaravind, KM, Vasan, G, Balaji, N & Sankaranarayanan, V 2015, 'Autonomous visual tracking and landing of a quadrotor on a moving platform', in *2015 Third International Conference on Image Information Processing (ICIIP)*, pp. 342-7.

ArduCopter 2017, *ArduCopter Documentation - Home*, ArduPilot, viewed 10th Mar 2017, <http://ardupilot.org/copter/>.

Autopilot, P 2017, *Pixhawk Flight Controller Hardware Project*, Pixhawk Autopilot, viewed 9th Mar 2017, <https://pixhawk.org/modules/pixhawk>.

Benson, T 2017, *Aircraft Rotations*, NASA -Glenn Research Center, viewed 29th Sept 2017, <https://www.grc.nasa.gov/www/k-12/VirtualAero/BottleRocket/airplane/rotations.html>.

Boyd, DS & Foody, GM 2011, 'An overview of recent remote sensing and GIS based research in ecological informatics', *Ecological Informatics*, vol. 6, no. 1, pp. 25-36.

Cai, G, Dias, J & Seneviratne, L 2014, 'A Survey of Small-Scale Unmanned Aerial Vehicles: Recent Advances and Future Development Trends', *Unmanned Systems*, vol. 02, no. 02, pp. 175-99.

Chandra, S, Chapman, R, DiVerdi, R, Preston, V, Woo, J, Bennett, A & Barrett, D 2016, 'Protocol for autonomous landing of unmanned air vehicles on research vessels', in *OCEANS 2016 MTS/IEEE Monterey*, pp. 1-5.

Chartier, B. 2017 Participating Industry Representative and Expert on Autonomous Systems, Nova Systems, Mile End, SA 2017

CMECI 2016. Hawaii. [ONLINE] Available: http://csem.flinders.edu.au/competitions/maritime_robotx/hawaii.html. [Accessed 13 June 2017].

Djapic, V, Prijic, C & Bogartz, F 2015, 'Autonomous takeoff & landing of small UAS from the USV', in *OCEANS'15 MTS/IEEE Washington*, pp. 1-8.

Drone_Insider 2017, *How do Brushless Motors Work?*, Drone Insider, viewed 19th Sept 2017, <http://droneinsider.org/brushless-motor-work/>.

Elkins, L, Sellers, D & Monach, WR 2010, 'The Autonomous Maritime Navigation (AMN) project: Field tests, autonomous and cooperative behaviors, data fusion, sensors, and vehicles', *Journal of Field Robotics*, vol. 27, no. 6, pp. 790-818.

FPV_Drone, R 2017, *Motor Basic Knowledge Introduction*, RC FPV Plane, viewed 23rd Sept 2017, <http://rcfpvplane.com/motor-basic-knowledge-introduction/>.

Gautam, A, Sujit, PB & Saripalli, S 2014, 'A survey of autonomous landing techniques for UAVs', in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1210-8.

getFPV 2017, *QAV400 FPV Quadcopter RTF (Pre-built and Tuned)*, getFPV, viewed 17th Apr 2017, <http://www.getfpv.com/ready-to-fly-quadcopters/6-rtf-quadcopters/qav400-fpv-quadcopter-rtf-pre-built-and-tuned.html>.

Helipal 2017, *STORM Drone AntiGravity GPS Flying Platform *RTF/NAZA V2)*, HeliPal, viewed 29th Mar 2017, <http://www.helipal.com/storm-drone-antigravity-gps-flying-platform-rtf-naza-v2.html>.

Herissé, B, Hamel, T, Mahony, R & Russotto, FX 2012, 'Landing a VTOL Unmanned Aerial Vehicle on a Moving Platform Using Optical Flow', *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 77-89.

Hobbyking 2017, *Turnigy SK450 Quad Copter Powered By Multistar. A Plug And Fly Quadcopter Set (PNF)*, HobbyKing, viewed 20th Apr 2017, <https://hobbyking.com/en_us/turnigy-sk450-quad-copter-powered-by-multistar-a-plug-and-fly-quadcopter-set-pnf.html>.

Kanellakis, C & Nikolakopoulos, G 2017, 'Survey on Computer Vision for UAVs: Current Developments and Trends', *Journal of Intelligent & Robotic Systems*, pp. 1-28.

Kendoul, F, Fantoni, I & Nonami, K 2009, 'Optic flow-based vision system for autonomous 3D localization and control of small aerial vehicles', *Robotics and Autonomous Systems*, vol. 57, no. 6–7, pp. 591-602.

Kim, J, Jung, Y, Lee, D & Shim, DH 2014, 'Outdoor autonomous landing on a moving platform for quadrotors using an omnidirectional camera', in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 1243-52.

Leddartech 2017, *Leddar Solid-State LiDAR for Drones*, UAS, viewed 2nd May 2017, <http://leddartech.com/drones-uas/>.

Lee, D, Ryan, T & Kim, HJ 2012, 'Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing', in *2012 IEEE International Conference on Robotics and Automation*, pp. 971-6.

Lee, D, Zhou, J & Lin, WT 2015, 'Autonomous battery swapping system for quadcopter', in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 118-24.

Liang, O 2017, *How to choose LiPo Battery Beginner Guide for Mini Quad, Drones and Quadcopters* OscarLiang, viewed 23rd Sept 2017, <https://oscarliang.com/lipo-battery-guide/>.

Liu, Z, Zhang, Y, Yu, X & Yuan, C 2016, 'Unmanned surface vehicles: An overview of developments and challenges', *Annual Reviews in Control*, vol. 41, pp. 71-93.

Mathews, N, Christensen, AL, O'Grady, R & Dorigo, M 2010, 'Cooperation in a heterogeneous robot swarm through spatially targeted communication', in *International Conference on Swarm Intelligence*, pp. 400-7.

Mavlink 2017, *MAVLINK Common Message Set*, Mavlink, viewed 19th Jul 2017, <http://mavlink.org/messages/common>.

MAVLink Micro Air Vehicle Communication Protocol - QGroundControl GCS. 2017. *MAVLink Micro Air Vehicle Communication Protocol - QGroundControl GCS*. [ONLINE] Available at: http://qgroundcontrol.org/mavlink/start. [Accessed 14 June 2017].

Meier, L, Tanskanen, P, Heng, L, Lee, GH, Fraundorfer, F & Pollefeys, M 2012, 'PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision', *Autonomous Robots*, vol. 33, no. 1, pp. 21-39.

Pinto, E, Marques, F, Mendonça, R, Lourenço, A, Santana, P & Barata, J 2014, 'An autonomous surface-aerial marsupial robotic team for riverine environmental monitoring: Benefiting from coordinated aerial, underwater, and surface level perception', in *Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference on*, pp. 443-50.

Prasetyono, W 2017, 'Autonomous Takeoff and Landing of a Small Drone from the TopCat ASV: Drone Platform Development and Software Control Integration', Honours thesis, Bachelor of Electronic Engineering (Honours), Flinders University (CSEM).

Quilter, S 2017, *S. Quilter/target-land: land a 3DR Solo on a colorful target*, Github, viewed 16th Jul 2017, <https://github.com/squilter/target-land>.

Ribeiro, D 2017, *ESC Firmwares and Protocols*, FPV Sampa, viewed 23rd Sept 2017, <https://fpvsampa.com/esc-firmwares-and-protocolos/>.

ROS, W 2017, *Mavros Wiki*, ROS Wiki, viewed 19th Jul 2017, <http://wiki.ros.org/mavros>.

Sammut, K, Lammas, A, Wheare, J, Webb, A, Donnelly, B, Stewart, J, Kossatz, M, Hutchinson, S, Geyer, S & Crouch, T 2016, 'Development and Testing of the TopCat Autonomous Surface Vessel for the Maritime RobotX Challenge 2016'.

SiK Radio. 2017. *SiK Radio — Advanced Configuration — Copter documentation*. [ONLINE] Available at: http://ardupilot.org/copter/docs/common-3dr-radio-advanced-configuration-and-technical-information.html. [Accessed 16 June 2017].

Swincer, D 2017, 'Autonomous Takeoff and Landing of a Small Drone from the TopCat ASV: Top Level Mission Planning', Honours thesis, Bachelor of Electronic Engineering (Honours), Flinders University (CSEM).

TeamCTU-UPENN-UoL, M 2016, *[MBZIRC] Challenge 1. UAV Landing on a Moving Vehicle (Second video submission, TeamUSRG at KAIST)*, Multi-robot Systems Group at FEE-CTU in Prague, viewed 29th Sept 2017, <https://www.youtube.com/watch?v=e1JEMmaZmDU>.

TeamUSRG, M 2016, *[MBZIRC] Challenge 1. UAV Landing on a Moving Vehicle (Second video submission, TeamUSRG at KAIST)*, ROS Wiki, viewed 29th Sept 2017, <https://www.youtube.com/watch?v=tN5D1UAgINg>.

The Three Laws. 2017. *RIVERWATCH: A Marsupial Surface-Aerial Robotic Team for Riverine Environmental Monitoring*. [ONLINE] Available at: http://thethreelaws.org/2014/02/01/riverwatch-a-marsupial-surface-aerial-robotic-team-for-riverine-environmental-monitoring/. [Accessed 15 June 2017].

Trimble. 2017. *Trimble - Precision GNSS + Inertial - BX982*. [ONLINE] Available at: http://www.trimble.com/gnss-inertial/bx982.aspx?dtID=overview. [Accessed 14 June 2017].

uBlox 2017, *NEO-7 Series Product Information*, uBlox, viewed 17th Apr 2017, <https://www.u-blox.com/en/product/neo-7-series#product-information >.

von Frankenbeg, F 2016, 'Development of an Autonomous OmniCopter Aerial Vehicle', Masters thesis, Mechanical Engineering, University of Ontario Institute of Technology (Canada).

WAM-V Marine Advanced Research Inc. 2017. *Marine Advanced Research.* [ONLINE] Available: https://www.wam-v.com/. [Accessed 13 June 2017].

Weaver, JN, Frank, DZ, Schwartz, EM & Arroyo, AA 2013, 'UAV performing autonomous landing on USV utilizing the robot operating system', in *Proc. of the ASME District F-Early Career Technical Conference*.

Wenzel, KE, Masselli, A & Zell, A 2011, 'Automatic Take Off, Tracking and Landing of a Miniature UAV on a Moving Carrier Vehicle', *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1, pp. 221-38.

Yang, X, Pota, H, Garratt, M & Ugrinovskii, V 2008, 'Prediction of vertical motions for landing operations of UAVs', in *2008 47th IEEE Conference on Decision and Control*, pp. 5048-53.

APPENDIX A: Configuration Parameters File for PIXHAWK

```
#NOTE: 16/10/2017 6:33:31 PM Frame : QAV400 Final Parameter
Configuration
ACCEL_Z_D,0
ACCEL_Z_FF,0
ACCEL_Z_FILT,20
ACCEL_Z_I,1
ACCEL_Z_IMAX,800
ACCEL_Z_P,0.5
ACRO_BAL_PITCH,1
ACRO_BAL_ROLL,1
ACRO_RP_EXPO,0.3
ACRO_RP_P,4.5
ACRO_THR_MID,0
ACRO_TRAINER,2
ACRO_Y_EXPO,0
ACRO_YAW_P,4.5
ADSB_ENABLE,0
AHRS_COMP_BETA,0.1
AHRS_EKF_TYPE,2
AHRS_GPS_GAIN,1
AHRS_GPS_MINSATS,6
AHRS_GPS_USE,1
AHRS_ORIENTATION,0
AHRS_RP_P,0.2
AHRS_TRIM_X,0.001541911
AHRS_TRIM_Y,0.02404268
AHRS_TRIM_Z,0
AHRS_WIND_MAX,0
AHRS_YAW_P,0.2
ANGLE_MAX,4500
ARMING_ACCTHRESH,0.75
ARMING_CHECK,2
ARMING_VOLT_MIN,0
ARMING_VOLT2_MIN,0
ATC_ACCEL_P_MAX,84766.1
ATC_ACCEL_R_MAX,143968.5
ATC_ACCEL_Y_MAX,16834.2
ATC_ANG_LIM_TC,1
ATC_ANG_PIT_P,15.1205
ATC_ANG_RLL_P,15.1205
ATC_ANG_YAW_P,4.231265
ATC_ANGLE_BOOST,1
ATC_RAT_PIT_D,0.01439568
ATC_RAT_PIT_FF,0
ATC_RAT_PIT_FILT,20
ATC_RAT_PIT_I,0.194
ATC_RAT_PIT_IMAX,0.444
ATC_RAT_PIT_P,0.194
ATC_RAT_RLL_D,0.003087656
ATC_RAT_RLL_FF,0
ATC_RAT_RLL_FILT,20
ATC_RAT_RLL_I,0.194
ATC_RAT_RLL_IMAX,0.444
```

```
ATC_RAT_RLL_P,0.194
ATC_RAT_YAW_D,0
ATC_RAT_YAW_FF,0
ATC_RAT_YAW_FILT,5
ATC_RAT_YAW_I,0.2
ATC_RAT_YAW_IMAX,0.222
ATC_RAT_YAW_P,2
ATC_RATE_FF_ENAB,1
ATC_SLEW_YAW,6000
ATC_THR_MIX_MAN,0.5
ATC_THR_MIX_MAX,0.5
ATC_THR_MIX_MIN,0.1
AUTOTUNE_AGGR,0.1
AUTOTUNE_AXES,4
AUTOTUNE_MIN_D,0.001
AVD_ENABLE,0
AVOID_ANGLE_MAX,1000
AVOID_DIST_MAX,10
AVOID_ENABLE,3
AVOID_MARGIN,2
BATT_AMP_OFFSET,0
BATT_AMP_PERVOLT,18.0018
BATT_CAPACITY,3000
BATT_CURR_PIN,3
BATT_MONITOR,4
BATT_SERIAL_NUM,-1
BATT_VOLT_MULT,10.10101
BATT_VOLT_PIN,2
BATT_VOLT_TIMER,10
BATT2_AMP_OFFSET,0
BATT2_AMP_PERVOL,17
BATT2_CAPACITY,3300
BATT2_CURR_PIN,3
BATT2_MONITOR,0
BATT2_SERIAL_NUM,-1
BATT2_VOLT_MULT,10.1
BATT2_VOLT_PIN,2
BCN_ALT,0
BCN_LATITUDE,0
BCN_LONGITUDE,0
BCN_ORIENT_YAW,0
BCN_TYPE,0
BRD_CAN_ENABLE,0
BRD_IMU_TARGTEMP,-1
BRD_IO_ENABLE,1
BRD_PWM_COUNT,4
BRD_SAFETY_MASK,16368
BRD_SAFETYENABLE,1
BRD_SBUS_OUT,0
BRD_SER1_RTSCTS,2
BRD_SER2_RTSCTS,2
BRD_SERIAL_NUM,0
BRD_TYPE,2
BTN_ENABLE,0
CAM_DURATION,10
CAM_FEEDBACK_PIN,-1
```

```
CAM_FEEDBACK_POL,1
CAM_MAX_ROLL,0
CAM_MIN_INTERVAL,0
CAM_RELAY_ON,1
CAM_SERVO_OFF,1100
CAM_SERVO_ON,1300
CAM_TRIGG_DIST,0
CAM_TRIGG_TYPE,0
CH10_OPT,0
CH11_OPT,0
CH12_OPT,0
CH7_OPT,39
CH8_OPT,0
CH9_OPT,0
CHUTE_ENABLED,0
CIRCLE_RADIUS,1000
CIRCLE_RATE,20
CLI_ENABLED,0
COMPASS_AUTODEC,1
COMPASS_CAL_FIT,16
COMPASS_DEC,0
COMPASS_DEV_ID,466441
COMPASS_DEV_ID2,131594
COMPASS_DEV_ID3,0
COMPASS_DIA_X,0.9780815
COMPASS_DIA_Y,1.106908
COMPASS_DIA_Z,1.019624
COMPASS_DIA2_X,0.8629416
COMPASS_DIA2_Y,0.9748495
COMPASS_DIA2_Z,1.015808
COMPASS_DIA3_X,0
COMPASS_DIA3_Y,0
COMPASS_DIA3_Z,0
COMPASS_EXTERN2,0
COMPASS_EXTERN3,0
COMPASS_EXTERNAL,1
COMPASS_LEARN,0
COMPASS_MOT_X,0.08372144
COMPASS_MOT_Y,2.115319
COMPASS_MOT_Z,0.2546179
COMPASS_MOT2_X,0.02525389
COMPASS_MOT2_Y,5.702569
COMPASS_MOT2_Z,0.2936423
COMPASS_MOT3_X,0
COMPASS_MOT3_Y,0
COMPASS_MOT3_Z,0
COMPASS_MOTCT,2
COMPASS_ODI_X,0.001049536
COMPASS_ODI_Y,0.03696046
COMPASS_ODI_Z,0.01198466
COMPASS_ODI2_X,-0.0185292
COMPASS_ODI2_Y,0.01887943
COMPASS_ODI2_Z,0.007927782
COMPASS_ODI3_X,0
COMPASS_ODI3_Y,0
COMPASS_ODI3_Z,0
```

```
COMPASS_OFFS_MAX,600
COMPASS_OFS_X,-272
COMPASS_OFS_Y,-46
COMPASS_OFS_Z,-181
COMPASS_OFS2_X,71
COMPASS_OFS2_Y,21
COMPASS_OFS2_Z,-68
COMPASS_OFS3_X,0
COMPASS_OFS3_Y,0
COMPASS_OFS3_Z,0
COMPASS_ORIENT,0
COMPASS_ORIENT2,0
COMPASS_ORIENT3,0
COMPASS_PRIMARY,0
COMPASS_USE,1
COMPASS_USE2,1
COMPASS_USE3,0
DEV_OPTIONS,0
DISARM_DELAY,10
EK2_ABIAS_P_NSE,0.005
EK2_ACC_P_NSE,0.6
EK2_ALT_M_NSE,3
EK2_ALT_SOURCE,0
EK2_BCN_DELAY,50
EK2_BCN_I_GTE,500
EK2_BCN_M_NSE,1
EK2_CHECK_SCALE,100
EK2_EAS_I_GATE,400
EK2_EAS_M_NSE,1.4
EK2_ENABLE,1
EK2_FLOW_DELAY,10
EK2_FLOW_I_GATE,300
EK2_FLOW_M_NSE,0.25
EK2_GBIAS_P_NSE,0.0001
EK2_GLITCH_RAD,25
EK2_GPS_CHECK,31
EK2_GPS_DELAY,220
EK2_GPS_TYPE,0
EK2_GSCL_P_NSE,0.0005
EK2_GYRO_P_NSE,0.03
EK2_HGT_DELAY,60
EK2_HGT_I_GATE,500
EK2_IMU_MASK,3
EK2_LOG_MASK,1
EK2_MAG_CAL,3
EK2_MAG_I_GATE,300
EK2_MAG_M_NSE,0.05
EK2_MAG_MASK,0
EK2_MAGB_P_NSE,0.0001
EK2_MAGE_P_NSE,0.001
EK2_MAX_FLOW,2.5
EK2_NOAID_M_NSE,10
EK2_POS_I_GATE,500
EK2_POSNE_M_NSE,1
EK2_RNG_I_GATE,500
EK2_RNG_M_NSE,0.5
```

```
EK2_RNG_USE_HGT,-1
EK2_RNG_USE_SPD,2
EK2_TAU_OUTPUT,25
EK2_TERR_GRAD,0.1
EK2_VEL_I_GATE,500
EK2_VELD_M_NSE,0.7
EK2_VELNE_M_NSE,0.5
EK2_WIND_P_NSE,0.1
EK2_WIND_PSCALE,0.5
EK2_YAW_I_GATE,300
EK2_YAW_M_NSE,0.5
EK3_ENABLE,0
ESC_CALIBRATION,0
FENCE_ACTION,1
FENCE_ALT_MAX,20
FENCE_ENABLE,0
FENCE_MARGIN,2
FENCE_RADIUS,300
FENCE_TOTAL,0
FENCE_TYPE,7
FLOW_BUS_ID,0
FLOW_ENABLE,0
FLOW_FXSCALER,0
FLOW_FYSCALER,0
FLOW_ORIENT_YAW,0
FLOW_POS_X,0
FLOW_POS_Y,0
FLOW_POS_Z,0
FLTMODE1,0
FLTMODE2,0
FLTMODE3,0
FLTMODE4,5
FLTMODE5,0
FLTMODE6,3
FRAME_CLASS,1
FRAME_TYPE,1
FS_BATT_ENABLE,0
FS_BATT_MAH,1000
FS_BATT_VOLTAGE,10
FS_CRASH_CHECK,1
FS_EKF_ACTION,1
FS_EKF_THRESH,0.8
FS_GCS_ENABLE,1
FS_THR_ENABLE,3
FS_THR_VALUE,970
GCS_PID_MASK,0
GND_ABS_PRESS,100878.8
GND_ABS_PRESS2,0
GND_ABS_PRESS3,0
GND_ALT_OFFSET,0
GND_EFFECT_COMP,1
GND_EXT_BUS,-1
GND_PRIMARY,0
GND_TEMP,0
GPS_AUTO_CONFIG,1
GPS_AUTO_SWITCH,1
```

```
GPS_BLEND_MASK,5
GPS_BLEND_TC,10
GPS_DELAY_MS,0
GPS_DELAY_MS2,0
GPS_GNSS_MODE,0
GPS_GNSS_MODE2,0
GPS_HDOP_GOOD,140
GPS_INJECT_TO,127
GPS_MIN_DGPS,100
GPS_MIN_ELEV,-100
GPS_NAVFILTER,8
GPS_POS1_X,0
GPS_POS1_Y,0
GPS_POS1_Z,0
GPS_POS2_X,0
GPS_POS2_Y,0
GPS_POS2_Z,0
GPS_RATE_MS,200
GPS_RATE_MS2,200
GPS_RAW_DATA,0
GPS_SAVE_CFG,0
GPS_SBAS_MODE,2
GPS_SBP_LOGMASK,-256
GPS_TYPE,1
GPS_TYPE2,0
GRIP_ENABLE,0
INS_ACC_BODYFIX,2
INS_ACC_ID,1246218
INS_ACC2_ID,1114634
INS_ACC2OFFS_X,0.3241463
INS_ACC2OFFS_Y,0.4001111
INS_ACC2OFFS_Z,1.772684
INS_ACC2SCAL_X,1.032945
INS_ACC2SCAL_Y,1.030075
INS_ACC2SCAL_Z,1.007484
INS_ACC3_ID,0
INS_ACC3OFFS_X,0
INS_ACC3OFFS_Y,0
INS_ACC3OFFS_Z,0
INS_ACC3SCAL_X,0
INS_ACC3SCAL_Y,0
INS_ACC3SCAL_Z,0
INS_ACCEL_FILTER,20
INS_ACCOFFS_X,-0.1988391
INS_ACCOFFS_Y,-0.3093835
INS_ACCOFFS_Z,0.3785743
INS_ACCSCAL_X,0.9915175
INS_ACCSCAL_Y,1.000806
INS_ACCSCAL_Z,0.9900876
INS_FAST_SAMPLE,0
INS_GYR_CAL,1
INS_GYR_ID,2163722
INS_GYR2_ID,2228490
INS_GYR2OFFS_X,-0.0001204061
INS_GYR2OFFS_Y,-0.002774837
INS_GYR2OFFS_Z,-0.02535386
```

```
INS_GYR3_ID,0
INS_GYR3OFFS_X,0
INS_GYR3OFFS_Y,0
INS_GYR3OFFS_Z,0
INS_GYRO_FILTER,20
INS_GYROFFS_X,0.005759267
INS_GYROFFS_Y,0.07848671
INS_GYROFFS_Z,0.003771209
INS_POS1_X,0
INS_POS1_Y,0
INS_POS1_Z,0
INS_POS2_X,0
INS_POS2_Y,0
INS_POS2_Z,0
INS_POS3_X,0
INS_POS3_Y,0
INS_POS3_Z,0
INS_PRODUCT_ID,5
INS_STILL_THRESH,2.5
INS_TRIM_OPTION,1
INS_USE,1
INS_USE2,1
INS_USE3,0
LAND_REPOSITION,1
LAND_SPEED,50
LAND_SPEED_HIGH,0
LGR_SERVO_DEPLOY,1750
LGR_SERVO_RTRACT,1250
LGR_STARTUP,0
LOG_BACKEND_TYPE,1
LOG_BITMASK,176126
LOG_DISARMED,0
LOG_FILE_BUFSIZE,16
LOG_FILE_DSRMROT,0
LOG_REPLAY,0
MAG_ENABLE,1
MIS_RESTART,0
MIS_TOTAL,26
MNT_ANGMAX_PAN,4500
MNT_ANGMAX_ROL,4500
MNT_ANGMAX_TIL,4500
MNT_ANGMIN_PAN,-4500
MNT_ANGMIN_ROL,-4500
MNT_ANGMIN_TIL,-4500
MNT_DEFLT_MODE,3
MNT_JSTICK_SPD,0
MNT_LEAD_PTCH,0
MNT_LEAD_RLL,0
MNT_NEUTRAL_X,0
MNT_NEUTRAL_Y,0
MNT_NEUTRAL_Z,0
MNT_RC_IN_PAN,0
MNT_RC_IN_ROLL,0
MNT_RC_IN_TILT,0
MNT_RETRACT_X,0
MNT_RETRACT_Y,0
```

```
MNT_RETRACT_Z,0
MNT_STAB_PAN,0
MNT_STAB_ROLL,0
MNT_STAB_TILT,0
MNT_TYPE,0
MOT_BAT_CURR_MAX,0
MOT_BAT_CURR_TC,5
MOT_BAT_VOLT_MAX,0
MOT_BAT_VOLT_MIN,0
MOT_HOVER_LEARN,2
MOT_PWM_MAX,0
MOT_PWM_MIN,0
MOT_PWM_TYPE,2
MOT_SAFE_DISARM,0
MOT_SPIN_ARM,0
MOT_SPIN_MAX,0.95
MOT_SPIN_MIN,0.14
MOT_SPOOL_TIME,0.5
MOT_THST_EXPO,0.65
MOT_THST_HOVER,0.6745227
MOT_YAW_HEADROOM,200
NTF_BUZZ_ENABLE,1
NTF_DISPLAY_TYPE,0
NTF_LED_BRIGHT,1
NTF_LED_OVERRIDE,0
NTF_OREO_THEME,0
PHLD_BRAKE_ANGLE,3000
PHLD_BRAKE_RATE,8
PILOT_ACCEL_Z,250
PILOT_THR_BHV,0
PILOT_THR_FILT,0
PILOT_TKOFF_ALT,200
PILOT_TKOFF_DZ,100
PILOT_VELZ_MAX,250
PLND_ACC_P_NSE,2.5
PLND_BUS,-1
PLND_CAM_POS_X,0
PLND_CAM_POS_Y,0
PLND_CAM_POS_Z,0
PLND_ENABLED,1
PLND_EST_TYPE,0
PLND_LAND_OFS_X,0
PLND_LAND_OFS_Y,0
PLND_TYPE,1
PLND_YAW_ALIGN,0
POS_XY_P,1
POS_Z_P,1
PRX_IGN_ANG1,0
PRX_IGN_ANG2,0
PRX_IGN_ANG3,0
PRX_IGN_ANG4,0
PRX_IGN_ANG5,0
PRX_IGN_ANG6,0
PRX_IGN_WID1,0
PRX_IGN_WID2,0
PRX_IGN_WID3,0
```

```
PRX_IGN_WID4,0
PRX_IGN_WID5,0
PRX_IGN_WID6,0
PRX_ORIENT,0
PRX_TYPE,0
PRX_YAW_CORR,22
PSC_ACC_XY_FILT,2
RALLY_INCL_HOME,1
RALLY_LIMIT_KM,0.3
RALLY_TOTAL,0
RC_FEEL_RP,50
RC_SPEED,490
RC1_DZ,20
RC1_MAX,2006
RC1_MIN,982
RC1_REVERSED,0
RC1_TRIM,1493
RC10_DZ,0
RC10_MAX,1900
RC10_MIN,1100
RC10_REVERSED,0
RC10_TRIM,874
RC11_DZ,0
RC11_MAX,1900
RC11_MIN,1100
RC11_REVERSED,0
RC11_TRIM,874
RC12_DZ,0
RC12_MAX,1900
RC12_MIN,1100
RC12_REVERSED,0
RC12_TRIM,874
RC13_DZ,0
RC13_MAX,1900
RC13_MIN,1100
RC13_REVERSED,0
RC13_TRIM,874
RC14_DZ,0
RC14_MAX,1900
RC14_MIN,1100
RC14_REVERSED,0
RC14_TRIM,874
RC15_DZ,0
RC15_MAX,1900
RC15_MIN,1100
RC15_REVERSED,0
RC15_TRIM,874
RC16_DZ,0
RC16_MAX,1900
RC16_MIN,1100
RC16_REVERSED,0
RC16_TRIM,874
RC2_DZ,20
RC2_MAX,2006
RC2_MIN,982
RC2_REVERSED,0
```

```
RC2_TRIM,1496
RC3_DZ,30
RC3_MAX,2006
RC3_MIN,982
RC3_REVERSED,0
RC3_TRIM,1233
RC4_DZ,20
RC4_MAX,2006
RC4_MIN,982
RC4_REVERSED,0
RC4_TRIM,1497
RC5_DZ,0
RC5_MAX,2006
RC5_MIN,982
RC5_REVERSED,0
RC5_TRIM,982
RC6_DZ,0
RC6_MAX,1997
RC6_MIN,982
RC6_REVERSED,0
RC6_TRIM,1494
RC7_DZ,0
RC7_MAX,2006
RC7_MIN,982
RC7_REVERSED,0
RC7_TRIM,982
RC8_DZ,0
RC8_MAX,1900
RC8_MIN,1100
RC8_REVERSED,0
RC8_TRIM,1494
RC9_DZ,0
RC9_MAX,1900
RC9_MIN,1100
RC9_REVERSED,0
RC9_TRIM,874
RCMAP_PITCH,2
RCMAP_ROLL,1
RCMAP_THROTTLE,3
RCMAP_YAW,4
RELAY_DEFAULT,0
RELAY_PIN,54
RELAY_PIN2,55
RELAY_PIN3,-1
RELAY_PIN4,-1
RNGFND_ADDR,0
RNGFND_FUNCTION,0
RNGFND_GAIN,0.8
RNGFND_GNDCLEAR,10
RNGFND_MAX_CM,4000
RNGFND_MIN_CM,20
RNGFND_OFFSET,0
RNGFND_ORIENT,25
RNGFND_PIN,-1
RNGFND_POS_X,0
RNGFND_POS_Y,0
```

```
RNGFND_POS_Z,0
RNGFND_PWRRNG,8
RNGFND_RMETRIC,1
RNGFND_SCALING,1
RNGFND_SETTLE,0
RNGFND_STOP_PIN,-1
RNGFND_TYPE,12
RNGFND2_ADDR,0
RNGFND2_FUNCTION,0
RNGFND2_GNDCLEAR,10
RNGFND2_MAX_CM,4000
RNGFND2_MIN_CM,5
RNGFND2_OFFSET,0
RNGFND2_ORIENT,25
RNGFND2_PIN,-1
RNGFND2_POS_X,0
RNGFND2_POS_Y,0
RNGFND2_POS_Z,0
RNGFND2_RMETRIC,1
RNGFND2_SCALING,3
RNGFND2_SETTLE,0
RNGFND2_STOP_PIN,-1
RNGFND2_TYPE,0
RPM_MAX,100000
RPM_MIN,10
RPM_MIN_QUAL,0.5
RPM_PIN,54
RPM_SCALING,1
RPM_TYPE,0
RPM2_PIN,-1
RPM2_SCALING,1
RPM2_TYPE,0
RSSI_ANA_PIN,0
RSSI_CHAN_HIGH,2000
RSSI_CHAN_LOW,1000
RSSI_CHANNEL,0
RSSI_PIN_HIGH,5
RSSI_PIN_LOW,0
RSSI_TYPE,0
RTL_ALT,500
RTL_ALT_FINAL,0
RTL_CLIMB_MIN,0
RTL_CONE_SLOPE,3
RTL_LOIT_TIME,5000
RTL_SPEED,100
SCHED_DEBUG,0
SCHED_LOOP_RATE,400
SERIAL0_BAUD,115
SERIAL0_PROTOCOL,1
SERIAL1_BAUD,57
SERIAL1_PROTOCOL,1
SERIAL2_BAUD,57
SERIAL2_PROTOCOL,1
SERIAL3_BAUD,38
SERIAL3_PROTOCOL,5
SERIAL4_BAUD,115
```

```
SERIAL4_PROTOCOL,9
SERIAL5_BAUD,57
SERIAL5_PROTOCOL,-1
SERVO_CANESC_BM,0
SERVO_CANSRV_BM,0
SERVO1_FUNCTION,33
SERVO1_MAX,2006
SERVO1_MIN,982
SERVO1_REVERSED,0
SERVO1_TRIM,1494
SERVO10_FUNCTION,0
SERVO10_MAX,1900
SERVO10_MIN,1100
SERVO10_REVERSED,0
SERVO10_TRIM,874
SERVO11_FUNCTION,0
SERVO11_MAX,1900
SERVO11_MIN,1100
SERVO11_REVERSED,0
SERVO11_TRIM,874
SERVO12_FUNCTION,0
SERVO12_MAX,1900
SERVO12_MIN,1100
SERVO12_REVERSED,0
SERVO12_TRIM,874
SERVO13_FUNCTION,0
SERVO13_MAX,1900
SERVO13_MIN,1100
SERVO13_REVERSED,0
SERVO13_TRIM,874
SERVO14_FUNCTION,0
SERVO14_MAX,1900
SERVO14_MIN,1100
SERVO14_REVERSED,0
SERVO14_TRIM,874
SERVO15_FUNCTION,0
SERVO15_MAX,1900
SERVO15_MIN,1100
SERVO15_REVERSED,0
SERVO15_TRIM,1500
SERVO16_FUNCTION,0
SERVO16_MAX,1900
SERVO16_MIN,1100
SERVO16_REVERSED,0
SERVO16_TRIM,1500
SERVO2_FUNCTION,34
SERVO2_MAX,2006
SERVO2_MIN,982
SERVO2_REVERSED,0
SERVO2_TRIM,1494
SERVO3_FUNCTION,35
SERVO3_MAX,2006
SERVO3_MIN,982
SERVO3_REVERSED,0
SERVO3_TRIM,982
SERVO4_FUNCTION,36
```

```
SERVO4_MAX,2006
SERVO4_MIN,982
SERVO4_REVERSED,0
SERVO4_TRIM,1497
SERVO5_FUNCTION,0
SERVO5_MAX,2006
SERVO5_MIN,982
SERVO5_REVERSED,0
SERVO5_TRIM,982
SERVO6_FUNCTION,0
SERVO6_MAX,1900
SERVO6_MIN,1100
SERVO6_REVERSED,0
SERVO6_TRIM,1494
SERVO7_FUNCTION,0
SERVO7_MAX,1900
SERVO7_MIN,1100
SERVO7_REVERSED,0
SERVO7_TRIM,1494
SERVO8_FUNCTION,0
SERVO8_MAX,1900
SERVO8_MIN,1100
SERVO8_REVERSED,0
SERVO8_TRIM,1494
SERVO9_FUNCTION,0
SERVO9_MAX,1900
SERVO9_MIN,1100
SERVO9_REVERSED,0
SERVO9_TRIM,874
SIMPLE,0
SPRAY_ENABLE,0
SR0_ADSB,5
SR0_EXT_STAT,2
SR0_EXTRA1,4
SR0_EXTRA2,4
SR0_EXTRA3,2
SR0_PARAMS,10
SR0_POSITION,2
SR0_RAW_CTRL,2
SR0_RAW_SENS,2
SR0_RC_CHAN,2
SR1_ADSB,5
SR1_EXT_STAT,2
SR1_EXTRA1,2
SR1_EXTRA2,2
SR1_EXTRA3,2
SR1_PARAMS,0
SR1_POSITION,2
SR1_RAW_CTRL,2
SR1_RAW_SENS,2
SR1_RC_CHAN,2
SR2_ADSB,5
SR2_EXT_STAT,2
SR2_EXTRA1,2
SR2_EXTRA2,2
SR2_EXTRA3,2
```

```
SR2_PARAMS,0
SR2_POSITION,2
SR2_RAW_CTRL,2
SR2_RAW_SENS,2
SR2_RC_CHAN,2
SR3_ADSB,5
SR3_EXT_STAT,2
SR3_EXTRA1,2
SR3_EXTRA2,2
SR3_EXTRA3,2
SR3_PARAMS,0
SR3_POSITION,2
SR3_RAW_CTRL,2
SR3_RAW_SENS,2
SR3_RC_CHAN,2
STAT_BOOTCNT,170
STAT_FLTTIME,10662
STAT_RESET,-504921500
STAT_RUNTIME,156472
SUPER_SIMPLE,0
SYSID_ENFORCE,0
SYSID_MYGCS,255
SYSID_SW_MREV,120
SYSID_SW_TYPE,10
SYSID_THISMAV,1
TELEM_DELAY,0
TERRAIN_ENABLE,1
TERRAIN_FOLLOW,0
TERRAIN_SPACING,100
THR_DZ,100
THROW_MOT_START,0
THROW_NEXTMODE,18
THROW_TYPE,0
TUNE,0
TUNE_HIGH,1000
TUNE_LOW,0
VEL_XY_FILT_HZ,5
VEL_XY_I,0.5
VEL_XY_IMAX,1000
VEL_XY_P,1
VEL_Z_P,5
VISO_ORIENT,0
VISO_POS_X,0
VISO_POS_Y,0
VISO_POS_Z,0
VISO_TYPE,0
WP_NAVALT_MIN,0.5
WP_YAW_BEHAVIOR,1
WPNAV_ACCEL,100
WPNAV_ACCEL_Z,100
WPNAV_LOIT_JERK,1000
WPNAV_LOIT_MAXA,250
WPNAV_LOIT_MINA,25
WPNAV_LOIT_SPEED,500
WPNAV_RADIUS,50
WPNAV_RFND_USE,1
```

```
WPNAV_SPEED,300
WPNAV_SPEED_DN,150
WPNAV_SPEED_UP,250
```

APPENDIX B: Procured Parts List & Links

Item	Purchasing Link
Quadrotor Parts:	
Frame (QAV400 w/ G10 arms)	https://www.desertaircraft.com.au/shop/qav400-frame-only-with-g10-arms.html?category_id=377#product-details-tab-description
Flight Controller Kit (HKPilot, GPS, 915MHz telemetry, wiring)	http://www.hobbyking.com/hobbyking/store/__80555__HKPilot32_Autonomous_Vehicle_32Bit_Control_Set_with_Telemetry_and_GPS_915Mhz_.html
Turnigy L2215J-900 Brushless Motor (200w)	https://hobbyking.com/en_us/turnigy-l2215j-900-brushless-motor-200w.html
ESCs (Alfro 20A w/BEC)+Simon K,Oneshot125	http://www.hobbyking.com/hobbyking/store/__43709__Afro_ESC_20Amp_Multi_rotor_Motor_Speed_Controller_SimonK_Firmware_.html
Propellers (8x4.5) pairs, 6mm hub	https://hobbyking.com/en_us/catalog/product/view/_ignore_category/1/id/63661/s/master-airscrew-8-x-4-5-multi-rotor-propeller-set-jet-black-1-x-cw-1-x-ccw
R/C Radio Receiver - 2.4GHz	https://hobbyking.com/en_us/frsky-x8r-8-16ch-s-bus-accst-telemetry-receiver-w-smart-port.html
Fr-Sky Telemetry overlay to the Taranis RC transmitter screen	https://hobbyking.com/en_us/frsky-flvss-lipo-voltage-sensor-with-smart-port-1pc.html
Velcro Strap for Battery	https://hobbyking.com/en_us/turnigy-battery-strap-330mm.html
4x Landing Gear & soft pads	https://www.desertaircraft.com.au/shop/neoprene-landing-pads-4-pack-suits-raceblade-frames.html?category_id=404
LiPo Low Voltage Alarm	https://hobbyking.com/en_us/hobbykingtm-lipo-voltage-checker-2s-8s.html
Extras:	
FPV camera + wiring - RunCam Swift 2	https://hobbyking.com/en_us/catalog/product/view/id/64585/s/runcam-swift-2-pal-black

PIXY IR-LOCK Sensor - Pixhawk Kit	https://irlock.com/collections/ir-lock-pixy/products/ir-lock-sensor-precision-landing-kit
5.8GHz Video Transmitter	http://www.hobbyking.com/hobbyking/store/__17507__ImmersionRC_5_8Ghz_Audio_Video_Transmitter_FatShark_compatible_600mw_.html
5.8GHz CP Antenna Set	http://www.hobbyking.com/hobbyking/store/__49826__ImmersionRC_5_8GHz_Circular_Polarized_SpiroNet_Antenna_V2_RHCP_SMA_.html
On-Scree-Display (OSD) Link	https://hobbyking.com/en_us/micro-hkpilot-osd-mavlink-compatible-micro-on-screen-display.html
Camera Power Filter- L-C, 1.7A	https://hobbyking.com/en_us/l-c-power-filter-1-7a.html
Laser Range Sensor - LedderOne Optical (3.3V UART)	http://www.robotshop.com/en/leddartech-leddarone-optical-rangefinder-33v-uart.html
Battery - 3S LiPo (3000mAh, 20C 136x44x18mm)	https://hobbyking.com/en_us/zippy-flightmax-3000mah-3s1p-20c.html
Universal FPV Monitor To Transmitter Mounting Bracket	https://hobbyking.com/en_us/universal-carbon-fpv-monitor-to-transmitter-mount-system.html
FPV LCD Monitor Boscam - 7 inch 800 x 480 5.8GHz	https://hobbyking.com/en_us/7-inch-800-x-480-5-8ghz-diversity-fpv-lcd-monitor-boscam-galaxy-d2.html
LiPo battery fire safe bag (230mm x 140mm)	https://hobbyking.com/en_us/fire-retardant-lipo-battery-bag-230x140mm.html
LiPo battery charger	https://hobbyking.com/en_us/turnigy-b6-compact-50w-5a-automatic-balance-charger-2-6s-lipoly.html
TH_LEDs - 950nm, 10degree divergence	https://www.digikey.com.au/product-detail/en/osram-opto-semiconductors-inc/SFH-4545/475-2919-ND/2205955
Breadboard for LED PCB development	https://www.digikey.com.au/product-detail/en/vector-electronics/8016-1/V2012-ND/416001
Propellers (8x3.8) pairs,	https://hobbyking.com/en_us/gemfan-glow-in-the-dark-propeller-8x3-8-x-cw-ccw-2pcs.html

6mm hub, 5,4,3&2.5mm adapters	
Propellers (6x3) pairs, 5mm hub with 4,3 & 2.5mm adapters	https://hobbyking.com/en_us/gemfan-propeller-6x3-black-cw-ccw-2pcs.html
Wire - 18AWG (1m black)	https://hobbyking.com/en_us/turnigy-high-quality-18awg-silicone-wire-1m-black.html
Wire - 18AWG (1m red)	https://hobbyking.com/en_us/turnigy-high-quality-18awg-silicone-wire-1m-red.html
Wire - 14AWG (1m black)	https://hobbyking.com/en_us/turnigy-high-quality-14awg-silicone-wire-1m-black.html
Wire - 14AWG (1m red)	https://hobbyking.com/en_us/turnigy-high-quality-14awg-silicone-wire-1m-red.html
XT60 connectors - 5 pairs, genuine	https://hobbyking.com/en_us/nylon-xt60-connectors-male-female-5-pairs-genuine.html
Female SMA-jack to Female RP-SMA plug	https://www.digikey.com/product-detail/en/amphenol-rf-division/132171RP-10/ACX1248-ND/1011925

Second Order:

Item	Purchasing Link
Quadrotor Parts:	
Turnigy L2215J-900 Brushless Motor (200w)	https://hobbyking.com/en_us/turnigy-l2215j-900-brushless-motor-200w.html
ESCs (Alfro 20A w/BEC)+SimonK, Oneshot125	http://www.hobbyking.com/hobbyking/store/__43709__Afro_ESC_20Amp_Multi_rotor_Motor_Speed_Controller_SimonK_Firmware_.html
Folding Propellers (2x 8045 CW & CCW) for QAV 500 & 450	http://www.ebay.com/itm/-/132276074542?ssPageName=STRK:MESE:IT
Folding Propellers (2x 7045 CW & CCW) for QAV 400 & 450	http://www.ebay.com/itm/-/132276074541?ssPageName=STRK:MESE:IT
Simple Prop Balancer - shaft only	https://hobbyking.com/en_us/simple-prop-balancer.html

Wattmeter and Voltage Analyzer - for Motors	https://hobbyking.com/en_us/hobbyking-hk-010-wattmeter-voltage-analyzer.html
Battery (Zippy Flightmax) - 3S, 5000mAh, 25C	https://hobbyking.com/en_us/zippy-flightmax-5000mah-3s1p-25c.html
Spare Telemetry radios	https://hobbyking.com/en_us/hkpilot-transceiver-telemetry-radio-set-v2-915mhz.html?___store=en_us
Zener Diode (5.6V, 5W) - PIXHAWK servo rail protection	http://au.element14.com/on-semiconductor/1n5339brlg/diode-zener-5-6v-5w-do-41/dp/2101822
MOSFET Transistor, N Channel, 2 A, 30 V, Vgs(on) = 4.5 V	http://au.element14.com/nxp/pmv40un-215/mosfet-n-ch-30v-4-9a-sot-23/dp/1758117

APPENDIX C: Required Pre-requisites for Running the Software

Ubuntu 16.04

ROS Kinetic

MAVROS v0.20.1 or later

- gives rangefinder updated drivers
- uses “mode_sent” rather than “success” booleans

Source install of MAVROS required rather than plain binary installation.

- need to add the LANDING_TARGET custom plugin.

Installing MAVROS to control PIXHAWK over MAVLINK from ROS

To install, follow the instructions in the Installation section of the following README file.

https://github.com/mavlink/mavros/blob/master/mavros/README.md

Note: Step 5 will take some time to build the 6 packages. If using a VM to run Ubuntu, ensure that 4G RAM is allocated to Ubuntu on the VM.

Also, Steps 2 – 6 can be run from the ~/catkin_ws/ folder, not its /src/ folder.

Need to use catkin build command, - if catkin build command is not available.

```
sudo -su      // get into root directory
sudo apt-get install python-catkin_tools
exit          // switch back to user
```

Don’t forget to do the following:

Re-source environment to reflect new packages/build environment

```
echo "source ~/catkin_ws/devel/setup.bash" >> ~/.bashrc
source ~/.bashrc
```

https://dev.px4.io/en/setup/dev_env_linux_ubuntu.html -> MAVROS section

// Needed geographic-msgs package for the MAVROS_MSGS package to build.

```
sudo apt-get install ros-kinetic-geographic-msgs
sudo apt-get install ros-kinetic-control-toolbox
```

Upgrading to MAVROS v0.20.1.

Also need to install geographiclib_datasets, Why? = “mandatory to allow the conversion between heights in order to respoect ROS msg API”

```
sudo ./src/mavros/mavros/scripts/install_geographiclib_datasets.sh
```

New executables will be installed & run in /usr/sbin/ folder:

- installs egm96-5, magnetic and gravity data sets

Necessary for v0.20.1 onwards.

Found I had the libgeographic library installed, but not the source devel files, to get the cmake files.

```
sudo apt-get install libgeographic-dev
```

When running mavros launch file, if having issues “cannot open serial0...”

check the following permissions on the ports:

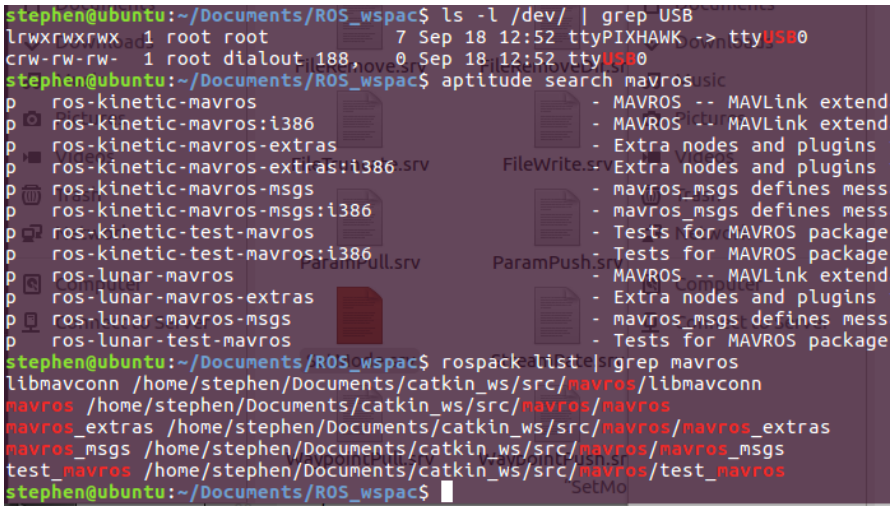
If permissions are “crw-rw----”, then run the following to change permissions, temporarily:

```
sudo chmod 666 /dev/ttyUSB0
```

OR permanently – develop a udev rule:

See: http://ask.xmodulo.com/change-usb-device-permission-linux.html

place in `/etc/udev/rules.d/`



Documentation describing each Node and function of the MAVROS package:

http://wiki.ros.org/mavros#Usage

APPENDIX D: Operational Issues & their Resolution

MAVROS radios:

This was because the same telemetry channel was being used for both feedback to the base station and the MAVROS node control. Separating these into two independent channels from the flight controller greatly improved the software command success rate and the operating range.

In future, MAVROS appears to have functionality to connect to a GCS as well as a serial port at the same time, so will need further investigation. This may help to reduce some of the operational errors that have been occurring during field testing.

RTK corrections:

These corrections are possible using the Mission Planner (RTCM v3.0) to forward these corrections to the FCU. Has not been tested in real flight yet.

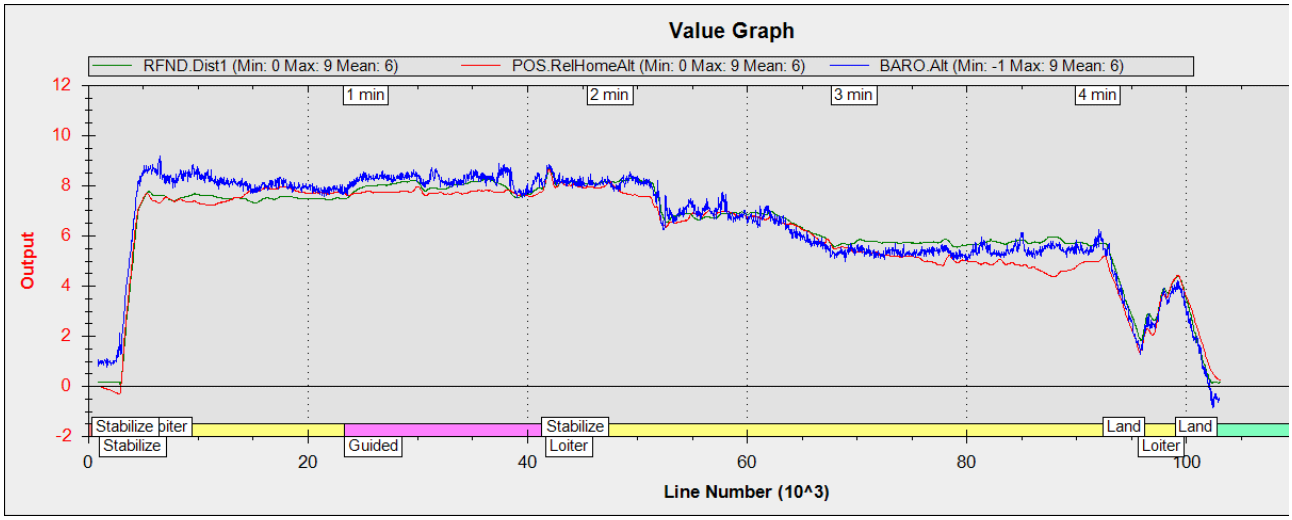
RC Transmitter:

Has been permitted to provide manual override during the landing process, so should always be used as a backup to the software during testing.

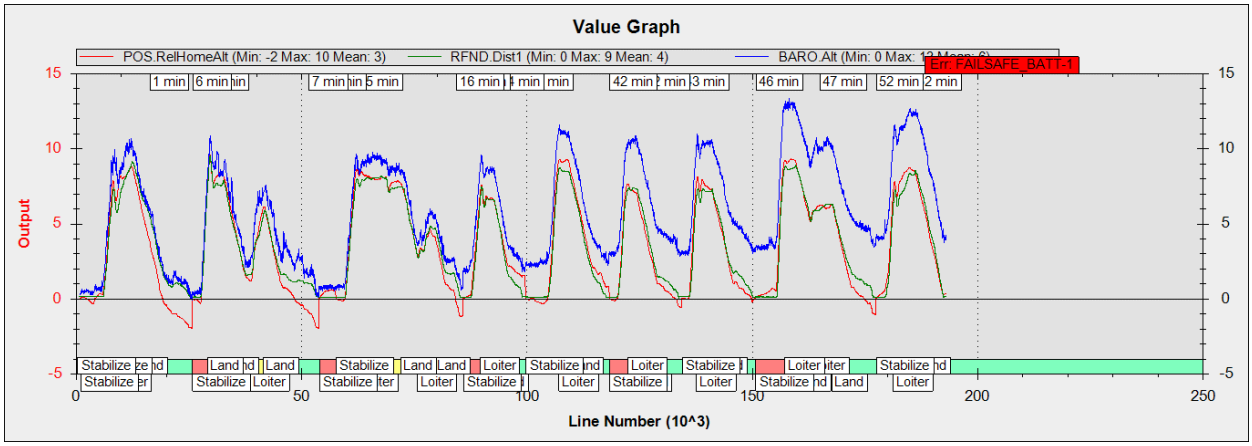
Ensure the throttle is <30% during the landing process, though, to ensure minimal hovering when close to the landing platform.

Barometer vs. Rangefinder:

Comparison of Drone's Actual Rangefinder readings (GRN) vs. Relative latitude – GPS (Red) and Barometer (Blue).



Log70



Log73 – 9 consecutive take-offs and lands showing the barometer drift over time.

The current PIXHAWK configuration prioritizes the Rangefinder reading over the GPS or barometer readings while this sensor is within range (i.e. < 40m). Higher than this, it automatically switches to the Barometer, which can give inaccurate readings particularly when hovering and experiencing the turbulence from the quad's propeller wash. Future implementations could consider fusing this data in a state predictor of choice.