Autonomous operation and control of a Multirotor Unmanned Aerial Vehicle through 4G LTE using onboard GPS and image processing

Mechatronics Engineering Final Year Project Report

Faculty of Engineering, Computing and Mathematics

The University of Western Australia

Semester Two, October, 2014

Alexander Mazur, 20516281
alexander.mazur@uwa.edu.au

Supervisors: Thomas Braunl and Christopher Croft

ABSTRACT

Through utilising modern telecommunication networks, we present a 'hexacopter' capable of intelligent remote waypoint navigation and image processing. Through the use of a web interface employing mapping software, users are able to specify remote waypoints for the hexacopter to navigate. The hexacopter maintains an internet connection through 4G LTE (or 3G if necessary), allowing monitoring and control within regions of no WiFi or radio reception. The hexacopter is controlled through a Raspberry Pi, employing GPS and 4G modules, along with a suite of sensors including a 5 megapixel camera. Utilising various image processing algorithms for object detection, the hexacopter can coordinate flight patterns to perform tasks such as object tracking and surveillance. Through utilising GPS and image processing techniques in tandem, the hexacopter can perform intelligent searching operations.

Acknowledgements

"If you can't take a little bloody nose, maybe you oughtta go back home and crawl under your bed. It's not safe out here. It's wondrous, with treasures to satiate desires both subtle and gross; but it's not for the timid."

– Q, from *Star Trek: The Next Generation*

I'd like to thank my supervisors and team mates for accompanying me throughout this project; my team mates (Omid Targhagh, Michael Baxter and Merrick Cloete) especially for being excellent sounding boards for whatever (possibly awful!) hexacopter-related ideas I may have had.

Gratification also goes out to Isaac Hochberg; who frequently helped me to test the hexacopter when nobody else was available, and who did his best to convince me to not procrastinate writing my report.

I extend further gratitude to Laura Pitts, who helped me greatly in editing this report into something comprehensible, and for her encouragement throughout the year.

Last, but not least, I'd like to thank *you*, dear reader, for reading these words. I hope the next 8000 hold your attention just as dearly!

CONTENTS

1. INTRODUCTION

Unmanned aerial vehicles (UAVs) have drawn significant levels of interest and research in the past decade. Over the years, electronic components have undergone a steady decrease in size and price, whilst levels of standardisation have increased. This has allowed the development of relatively cheap and lightweight small helicopter 'drones', or multirotor UAVs. These UAVs can be fitted with a variety of sensors and equipment to perform a multitude of tasks - from aerial photography [17] to disaster search and rescue [78].

A multirotor UAV (MUAV) traditionally comprises four rotors (a quadcopter, exemplified in Figure 1). This configuration allows similar flight capabilities to a regular helicopter: movement in three dimensions is possible, along with yaw and hovering in place. The MUAV physically consists of a central enclosure, housing the majority of the electronics, with four fixed radial arms with a motor and propeller on the end of each. The MUAV achieves movement through varying the speed of these rotors. For example, by reducing the speed of two rotors on one side, the MUAV looses lift on this side, causing it to angle slightly. This angle, in turn, creates horizontal thrust (much like a helicopter angling forwards to move) [38].

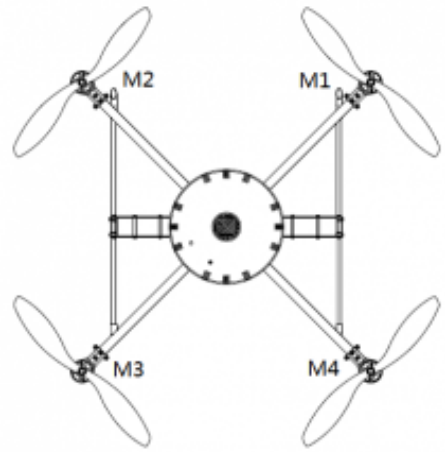

Figure 1: A multirotor UAV with four rotor sets - a 'quadcopter'. (XAircraft, 2014 [83])

UAVs can be controlled directly by an operator (much like a remote-controlled aeroplane), or they can be controlled by a pre-programmed computer. UAVs controlled by a computer with the ability to complete tasks with no human interaction are known as autonomous UAVs. The computer can be located on the ground, from where it can send commands wirelessly which will be processed and carried out by the UAV. Alternatively, a computer can be installed on the UAV, allowing the UAV to act completely independently.

Modern single-board microcontroller designs are sufficiently small and lightweight to be installed directly on a UAV. The *Raspberry Pi* is a popular single-board computer built around a 32-bit Atmel ARM microprocessor [60, 8]. The Raspberry Pi includes a 700MHz ARM processor [7], a GPU, and 512MB of RAM, and runs variants of the Linux operating system [62]. This combination of computing power and an operating system allow the Raspberry Pi to perform a diverse spectrum of tasks, from image processing to network connectivity.

Communication with an autonomous UAV can be achieved through a variety of means, such as analogue radio control (for example, with a remote control), or digitally through a WiFi network utilising 2.4 GHz UHF radio waves. In both cases, a wireless radio receiver is installed on the UAV, requiring the user to be within a fixed distance of the UAV to maintain communications. Alternatively, mobile communications networks can also be utilised through the use of 4G LTE or 3G modules, allowing communication with the UAV from any location with network coverage.

A fully autonomous UAV is able to utilise a variety of onboard sensors to complete tasks. Two common sensors are global positioning system (GPS) devices and cameras. Through the use of a GPS device, the UAV is able to locate itself in terms of latitude, longitude and altitude [32]. This system can be used to allow the UAV to follow a set of predefined waypoints. An onboard camera can be used to allow the UAV to perform tasks based on what is immediately visible. For example, the UAV can be programmed to recognise certain objects and attempt to 'follow' them.

An autonomous UAV must be supplied directives to complete; this can either occur during construction ('pre-loaded' instructions), or in real time through communicating with an operator. There are a variety of methods of communication, ranging from specially constructed remote control devices to common smart phones. The latter option involves several further possibilities: communication through a smart phone 'app' or through a web interface. This report elaborates on methods of control and applications for an autonomous UAV and presents a workable UAV control solution.

1.1 Applications

The global market share of autonomous UAVs in the civilian sector is expected by the EU to increase to up to 10% of aviation in the next 10 years [20]. This forecast is given in light of recent developments in the private UAV sector, where several major multinational corporations have launched research and development operations into potential applications for autonomous UAVs.

1.1.1 Package Delivery

In December 2013, electronic commerce company Amazon.com announced plans to deliver packages utilising autonomous drones capable of carrying 5 pound packages, at speeds of 50 miles per hour [65, 2]. In the same month, Deutsche Post (the world's largest courier company) delivered a sub-kilogram quantity of medicine using their own autonomous UAV, raising speculation about potential uses for disaster relief efforts [25, 30]. In February 2014, the United Arab Emirates announced plans to launch a fleet of UAVs for civilian purposes such as the delivery of official documents, utilising sensors for fingerprint and iris-recognition systems [42]. Furthermore, Google revealed in August 2014 that they had been developing UAVs for delivery purposes in Australia for the past two years [47].

1.1.2 Agriculture

Several companies offer UAV-based imaging solutions for crop fields [45, 55]. UAVs are used to capture visual, thermal, LiDAR or multi/hyperspectral images of crop fields. These images can be used to map the terrain of the crops, giving plant heights and counts, along with weed detection. Canopy cover can be determined, along with various metrics for judging crop 'health'. Intelligent solutions can inform farmers of regions that need to be harvested or sprayed, and can help fix problems before they occur [41, 57, 66].

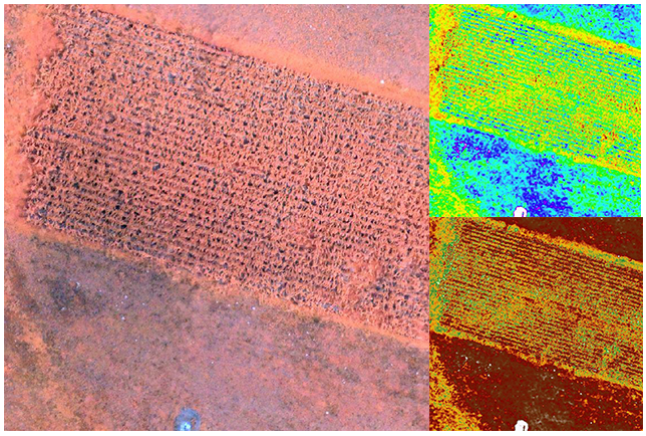

Figure 2: A combination of multispectral images taken of a crop, used to assess crop 'health' through a variety of metrics (Precision Hawk USA Inc., 2014 [55])

1.1.3 Filming and Surveillance

Today, a many commercial UAVs are available for both hobbyist and commercial applications, such as the popular Parrot AR.Drone [54]. Drones specifically designed with high quality cameras for filming purposes are abudant in the private sector. Examples include the Coptercam [58] and SteadiDrone [68].

1.2 Project Scope

1.2.1 Previous Work

The purpose of the autonomous UAV project is to integrate a Raspberry Pi with a six-rotor UAV (hexacopter) and build autonomous functionality utilising only onboard sensors and onboard processing. This project commenced in 2013, when students purchased a DJI F550 UAV chassis, along with a DJI NAZA M flight controller and all additional components required for user-controlled flight [77, 53] . This 'hexacopter' (illustrated in Figure 3) could be operated utilising a Futaba T14SG 14 channel 2.4GHz remote control, allowing flight in a fashion much like a toy helicopter. Students then spliced a Raspberry Pi between the Futaba R7008SB receiver and the flight controller on the UAV, allowing the Raspberry Pi to simulate the output from the remote control device. Combined with a Qstarz GPS module and a Raspberry Pi Camera module, they were able to achieve autonomous flight of the UAV.

1.2.2 Current Work

The project this year was handled by a team of four students. As such, four additional broad tasks were identified:

Figure 3: The hexacopter, a DJI F550 chassis modified to include a Raspberry Pi, Raspberry Pi Camera and Qstarz GPS.

- To develop a suite of adaptable software for basic hexacopter control.
- To develop advanced flight algorithms for GPS waypoint navigation.
- To develop advanced image processing functionality for tasks such as object tracking, whilst providing dynamic environment awareness.
- To develop a user interface for high-level hexacopter control.

This report focuses on the latter task; presenting a method of control for a Raspberry Pi controlled autonomous UAV through a web interface, operable from any modern web-enabled device (smart phone, tablet, laptop computer, etc). This is presented in combination with a suite of 'back end' software for UAV control, along with methods of communication between the user and all presented layers of software.

2. LITERATURE REVIEW

2.1 Hexacopter UAVs

The family of vehicles falling under the category 'UAV' is large. Fixed wing aircraft (such as the infamous 'Predator drone' [22]) and weather balloons can be classified as UAVs. The UAV used throughout this project is a 'Vertical Take Off and Landing' UAV (VTOL UAV), capable of taking off and landing without need of a runway.

A hexacopter features six rotor sets, each comprising of two identical aerofoils, known as blades. These are attached from one end to a common rotating shaft, which generates lift upon rotation. The aerofoils generate a pressure gradient between the regions above and below them when spun in the correct direction through a fluid. The effect of this is a higher pressure on the bottom side of the aerofoil than on the top side, causing the aerofoil to experience a net vertical force known as aerodynamic lift [5]. In a hexacopter configuration, the six rotors must each rotate counter to the

rotors directly adjacent to them in order to counteract the torque experienced by the UAV, so as to generate lift [4].

Rotation of the hexacopter is achieved by speeding up individual motors relative to the others, causing an imbalance resulting in a yaw rotation. Vertical thrust is generated by spinning all rotors faster, and horizontal travel is generated by decreasing the thrust of rotors on a certain side of the hexacopter, causing it to move in that direction [24].

The extra rotors of a hexacopter offer several additional capabilities when compared to a helicopter (two rotors) or quadcopter (four rotors) . A hexacopter features greater lifting strength and stability, along with the ability to lose one motor and still have sufficient power to control the UAV [13].

2.2 UAV Control Methodologies

In order for a UAV to be autonomous, it must be able to complete tasks without human interaction. Such control is achieved through the use of computer systems.

2.2.1 External Control

An autonomous UAV can be controlled through a ground control station. This is similar to an operator manually controlling the UAV, except a computer is 'using the remote control'. The UAV can communicate information wirelessly to the ground control station, such as camera and GPS data. Such a setup exploits the power of conventional computer systems to process commands for the UAV. Image processing can be performed rapidly and efficiently, and the ground computer may have access to extensive databases of information useful in determining the UAVs next course of action (for example, map data). This control scheme is limited, however, by the range of wireless communications between the UAV and the ground control station, and the bandwidth of these communications. Bandwidth limitations are common, and as such smooth, high resolution video is not always possible. An externally controlled UAV is 'tethered', and can never move out of the range of the control station [34].

2.2.2 Onboard Control

A 'truly' autonomous UAV is able to control itself without need for external systems. One step towards this goal is achieved by placing a computing system onboard the UAV, allowing it to process all commands itself. Such a computer system must be compact and lightweight; the most popular systems meeting this criteria are the *Arduino* family of microcontrollers and the *Raspberry Pi*.

Arduino. The Arduino [6] is a popular single-board microcontroller designed around an 8-bit Atmel AVR microcontroller [9], or a 32-bit Atmel ARM [8]. The Arduino was introduced in 2005 as an inexpensive and simple way to allow the creation of devices that can interact with their environment. As a relatively low-powered microcontroller (the Arduino Uno has a maximum CPU clock speed of 20 MHz), the Arduino is designed to be used to solve specific application problems; it is not intended to run an operating system. This processing power limitation makes the Arduino unsuitable for computationally intensive tasks, such as image processing. As Arduino's are not intended to be used as general computers, they do not come with several features such as LAN (Local Area Networking) and A/V (Audio/Video) output. This functionality must be added separately. [15]

Raspberry Pi. The Raspberry Pi [60] (shown in Figure 4) includes a 700MHz ARM processor [7], a GPU, and 512MB of RAM, significantly outperforming the Arduino. Instead of executing code directly, the Raspberry Pi can run variants of the Linux operating system, allowing a greater potential of tasks to be completed. Due to its increased power over the Arduino, the Raspberry Pi can more easily handle computationally intensive tasks such as image processing. Additionally, due to running the Linux operating system, the Raspberry Pi is able to handle additional tasks such as wireless communication and interfacing, allowing for a user to input directives to an autonomous UAV system. [75]

Figure 4: The Raspberry Pi Model B+ is a credit card sized computer capable of running the Linux operating system. (The Register, 2014 [64])

2.2.3 Sensors

An autonomous UAV cannot operate meaningfully with a computer alone; it requires information from the world around it. Common sensors used for flight include GPS devices for latitude and longtitude positioning, Inertial Measurement Units (IMUs) for compass and accelerometer data, and cameras for visual awareness [37].

GPS Device. A GPS device can be used to provide positional data for a UAV in terms of latitude, longitude and altitude [32]. Common GPS devices can provide horizontal positional accuracy to within 3 metres, with a 'worst case' pseudorange accuracy of 7.8 metres at a 95% confidence level [51]. Higher accuracy can be achieved through higher quality GPS devices, or augmentations such as the International GNSS Service [40] and Global Differential GPS [50].

Commercially available GPS devices suitable for connection to an Arduino or Raspberry Pi include the QStarz BT-Q818X GPS Module [59] and the Swift Navigation Piksi GPS receiver [71]. The QStarz GPS Module is a standard accuracy GPS device, with an operational accuracy of 100m

always, whilst the Piksi GPS Module is an experimental device featuring centimetre level accuracy.

Raspberry Pi Camera. Visual cameras are a common data rich sensor used to provide environmental awareness to autonomous systems. A camera can allow a UAV to be aware of obstacles or to track objects [74]. The Raspberry Pi Camera [61] is an inexpensive five megapixel camera directly compatible with the Raspberry Pi. Capable of both 1080p and 720p images and video, this camera allows the Raspberry Pi to locate objects by applying appropriate image processing algorithms.

2.2.4 Communications

An autonomous UAV must be supplied directives to complete; this can either occur during construction or operation. In order to send the UAV directives, a wireless communication protocol must be chosen. There exist several options utilising a range of wireless spectra and standards.

Radio Remote Control. Radio remote control devices can be used to send flight instructions directly the UAV. These devices generally send analogue yaw, pitch and roll signals directly to a receiver on the UAV, allowing it to respond in near real-time. Most modern remote control systems utilise a spread spectrum technique over the 2.4 GHz band. This technique spreads the bandwidth of the transmitted signal over the frequency domain, allowing for a more stable connection with increased immunity to interference and noise [1]. The 2.4 GHz band is chosen as this is part of the industrial, scientific and medical (ISM) portion of the radio spectrum; utilising this band (usually) does not require any licensing or approval worldwide [10]. Depending on the receiver/transmitter used, ranges of several kilometres can be achieved utilising the 2.4 GHz band. The Futaba T14SG 14 channel 2.4GHz remote control is currently used to control the hexacopter directly [31]. It is possible to utilise a modified remote control to directly transmit telemetry to the UAV; however, this is contrary to the goals of constructing an autonomous UAV.

IEEE 802.15.4. The IEEE 802.15.4 standard specifies the requirements for low-rate wireless personal area networks (LR-WPANs) [85]. These networks focus on low-cost, low-speed communication between devices. The standard is the basis for several standards such as the ZigBee and XBee specifications [27]. These specifications are used to produce inexpensive, compact radio modules allowing digital communications. These can operate on the 868 MHz, 902 MHz or 2.4 GHz ISM bands. Whilst the range of the 2.4 GHz band is limited, utilising the lower frequency ISM bands can allow for effective ranges of up to 20 km. However, utilising this standard requires specialised hardware on the receiving end, and restricts communications to low bandwidths (making applications like video streaming impractical) [14].

Wi-Fi. Wi-Fi is a local area network wireless technology based on the IEEE 802.11 standards [73]. The name is derived from a 'play on words' of the term *Hi-Fi*. Wi-Fi is a ubiquitous technology, with the majority of modern smart phones, tablets, and laptop computers coming with Wi-Fi capabilities. Wi-Fi operates over the 2.4 GHz ISM and 5 GHz radio bands, however traditionally have very limited range compared to that possible utilising a 2.4 GHz analogue remote control. This is due to a plethora of factors, such as limited transmission power, lower fault tolerances than an analogue signal, and the congestion of the Wi-Fi wireless radio band. Outdoors, ranges of 50-100 metres can be expected [16, 84].

4G LTE. 4G LTE is a marketing term for a standard of wireless communication for cellular networks called Long-Term Evolution, or LTE. LTE does not meet the technical requirements specified by the 3GPP consortium for a new generation of cellular wireless communication, and hence the term '4G' is a misnomer.

LTE is based on the previous '3G' network standards (GSM / EDGE and UMTS / HSPA). LTE increases the capacity and speed of the previous standards by using different radio interface combined with core network improvements. From a practical perspective, both LTE and '3G' networks allow data to be transferred through a global cellular network, removing the need to set up a local wireless network through Wi-Fi or the IEEE 802.15.4 standards. A device controlled through LTE or '3G' can be operated from across the world, if deemed necessary.

LTE features peak download rates up to approximately 300 Mb/s, an improvement over 3G by roughly a factor of ten. Furthermore, LTE provides low latency communication (sub 5 ms), making it suitable for quick communication to a UAV [21].

2.3 Human-Computer Interface

A human-computer interface, or *user interface*, is required to supply directives to a UAV. As a common task in UAV control is GPS navigation, a graphical user interface capable of displaying a map is required. In designing such a user interface, there are several considerations to take into account. These include general user interface design principles, along with what hardware and software the interface will be designed for. In the case of the UAV, appropriate hardware and software must be chosen to operate in a predominantly outdoors environment, and the interface must provide clarity in bright environments with potentially small screens.

2.3.1 Graphical User Interface Design

As a general consensus from a multitude of various sources, the following elements are considered to be essential to any well-designed user interface:

- **Clarity.** The interface must avoid ambiguity through using clear language and visual elements [56].
- **Concision.** The interface must not be tedious to use, or 'bloated'. This criteria often must be balanced with clarity [63].

- **Familiarity.** Even if an interface is used for the first time, it can be made to feel familiar to the user through elements such as real-life metaphors [35].
- **Responsiveness.** The interface should provide timely feedback to the user without causing frustration. This criterion is especially important for a UAV; the user must be able to tell the UAV to stop before something goes wrong [23].
- **Consistency.** Allowing a user to recognise usage patterns across the interface aids the speed and understanding with which a user can operate the interface [67].
- **Aesthetics.** A 'good' looking interface helps users to enjoy using the application, which assists in their use of the interface [18].
- **Forgiveness.** The interface should assist users in operating it, without causing unnecessary errors [44].

2.3.2 Hardware

Smart Phones and Tablets. Smart phones and tablets have become ubiquitous devices across the world in the past decade [12]. These devices typically come with a touch screen and GPS and Wi-Fi modules, allowing a user to access the internet and third party apps (including mapping software). Modern smart phones and tablets have access to a similar level of processing power as a Raspberry Pi, making them suitable for many levels of control. However, most modern smart phones and tablets are not able to communicate through more specialised protocols without external hardware, such as 802.11.4 compliant protocols or analogue radio transmission.

Laptop Computers. Modern laptop computers are generally more powerful than smart phones or tablets, and as such are capable of more computationally intensive tasks, such as image and video processing. However, they suffer the same limitations in that they require extra hardware for specific applications such as GPS tracking or communication over 802.11.4 protocols.

2.3.3 Software

Mobile Apps. Three primary operating systems are installed on modern smart phones and tablets: Microsoft Windows Phone, Google Android, and Apple iOS. Each of these distributors maintains their own 'app' stores, allowing developers to create and distribute software to users. Developing apps for each platform is a mutually exclusive task; an app developed for one platform will not operate on another. This is due, in part, to each platform executing code in different languages: Objective C for iOS, Java for Android, and various C derivatives for Windows Phone. As such, developing an app-based user interface would require supporting three different code bases for three different platforms. Furthermore, development of mobile apps excludes operation of the user interface on other platforms, such as laptop computers [19].

Web Interface. A web interface is a user interface that operates through a internet browser. Modern internet browsers do not suffer the cross-compatibility problem to the same extent as mobile apps; code written for one internet browser will operate similarly in another without any modification. Furthermore, all modern hardware identified in Section 2.3.2 is capable of running a variety of internet browsers. As such, a web interface is a more broadly compatible solution for hexacopter control than a suite of mobile apps. However, due to their generic nature, web interfaces can often suffer from poor, unresponsive design compared to mobile apps, which are specifically tailored for their devices.

2.4 Web Interface Technologies

A modern web solution utilises a standard paradigm: a *web browser* is used by a user to display web pages. These web pages are constructed from the *HTML* and *CSS* languages, describing the static layout of the web page; dynamic content is provided by a language called *JavaScript*. The web page is constructed on a server through a language called *PHP*, and then communicated to the user through the use of a *web server*. Due to the dynamic nature of web pages, the user always receives the most up-to-date version of a web page at all times, without the need to manually update [81, 49].

2.4.1 Browser

Popular modern web browsers include Google Chrome, Mozilla Firefox, Microsoft Internet Explorer and Apple Safari. Perhaps unfortunately, no modern web browser complies to the standards of website rendering dictated by the World Wide Web Consortium (W3C), the main international standards organisation for the World Wide Web. This means that a web interface must be adapted (if necessary) to render similarly on each of these browsers [48].

HTML/CSS. A web browser is designed to display websites written in HyperText Markup Language (HTML). HTML is intended to define data and the structure in which it is to be displayed. In order to change the look and layout of this displayed data, Cascading Style Sheets (CSS) are used. Together, these languages are used to describe a *static* (non-changing) web page [36].

JavaScript and AJAX. In order for a web page to be interactive, web browsers are capable of executing a scripting language called JavaScript. JavaScript can dynamically alter the HTML and CSS content of a webpage, allowing for the webpage to interact with the user. Furthermore, JavaScript can be used for asynchronous communication; that is, allowing the webpage to communicate with a server in the background, without refreshing the page. This method of communication is known as Asynchronous JavaScript + XML (AJAX), with the XML component describing the contents of the data packets sent to and from the server [33].

2.4.2 Web Server

A web server is required to send webpages written in HTML/CSS/JavaScript to a user. Web pages can be initially written in a scripting language called PHP, which is then

processed and constructed, with the output send to the web server [11]. The final product is sent to the user through TCP/IP network communications. There are several web servers available to choose from, each offering a different set of features.

Apache HTTP Server. The *Apache HTTP Server* [72], colloquially called *Apache*, is the most popular web server used on the internet, estimated to serve 54.2% of all active web sites as of June 2013 [80]. Apache is a full featured web server, providing extensive functionality and methods of configuration. As such, Apache can be rather resource intensive compared to other alternative web servers [29].

Nginx. Nginx (pronounced 'engine-x') is a suite of software containing a web server. Nginx is estimated to serve 14.56% of websites on the internet as of June 2013, and is rapidly growing as a popular alternative to Apache [80]. Nginx offers performance improvements over Apache, but requires more in-depth configuration. As such, incorrectly configured Nginx servers can open up the potential for security holes and errors [52].

Lighttpd. Lighttpd (pronounced 'lighty') is a lightweight web server, offering small CPU load and a low memory footprint when compared to Apache and Nginx, whilst still maintaining the majority of features common to the others. Lighttpd is an excellent choice for operation on a Raspberry Pi, owing to the limited computing power of the Raspberry Pi [43].

2.4.3 Inter-Process Communication

A web server/PHP configuration is suitable for generating web sites, but not for controlling the flight or hardware of a UAV. This is due, in part, to PHP being a scripting language; it is not designed to run in the background. Rather, it is designed to run as required, for example when a user requests a web page. A flight control program must run continuously to ensure correct UAV operation. As such, a method of communication between PHP and a flight control program is required. There are several options, ranging from manual network socket communication to higher levels of abstraction in the form of interface description languages.

Network socket communication. Network sockets are the endpoint of inter-process communications across a computer network. Through the use of protocols such as TCP or UDP, programs can send and receive data. In order for this to occur, a program can either transmit or listen for data on a network *port*. A port is software construct, existing to isolate multiple streams of network communication from each other. For example, web pages from the internet are served over port 80.

Utilising network socket communication requires the writing of a *server* (to listen for and respond to requests), and a *client* to send out requests. This is a low-level method of communication, and a lot of effort must be expended to write a reliable interface [26].

Representational State Transfer. As an alternative to utilising network sockets directly, there exist implementations at a higher level of abstraction. For example, a Representational State Transfer (REST) interface abstracts network communication to a *uniform resource identifier* (URI) and standard HTTP methods (GET, PUT, POST, DELETE). An example of a 'REST-ful' interface is an address `'http://example.com/gps/location'`; navigating to this address (or URI) may return the current location of some device.

A REST-ful implementation does come with downsides; as each URI is accessible through a web browser, it can be simple for malicious users to exploit commands; this is especially a concern for a UAV. Furthermore, a REST-ful implementation is not the fastest method of communication, and does not provide any facility for handling errors in commands [28].

Binary Communication Protocol. A binary communication protocol is a protocol intended to be read by a machine rather than a human being. This comes with advantages in terms of speed; but can add complexity in design. A binary communication protocol utilises network socket communication, but provides a fast and ready made interface with facilities for error management.

Several implementations exist, such as Apache Thrift and Google Protocol Buffers. Apache Thrift is a protocol originally developed by Facebook for internal use in 2007, but since released as an open source project to the Apache Software Foundation (the maintainers of the Apache HTTP Server). Google Protocol Buffers was released a year later, and is used internally by Google [69].

2.5 Mapping Software

A common requirement for an autonomous UAV is mapping software, to display where the UAV is and any routes it may take. There exist several web-based solutions providing this functionality.

2.5.1 Google Maps

The most popular and well-known mapping software, Google Maps provides an extensive developer Application Programming Interface (API) allowing, among other things, the display of custom routes and markers. Google Maps operates by presenting the user with a rectangular region defined by two latitude and longitude pairs. This region is divided into a grid based on the physical size of the users viewport (screen). For each grid square, Google sends a small image (a *tile*) mapping to that specific coordinate for a particular zoom level. The application tracks the movement and zoom of the users viewport, and sends new tile information appropriately.

Google stores this tile data on its servers, and does not provide functionality for downloading these tiles for permanent storage on a device. As such, Google Maps requires an active internet connection to operate, which may limit its use

for UAV position marking [70].

2.5.2 *Leaflet*

Leaflet is a JavaScript library providing similar functionality to Google Maps; however it is open source and usable offline [3]. In order to present a map image to the user, tiles corresponding to certain coordinates need to be saved locally. It is against Google's terms of service to download and use their tiles in this way, however several alternative tile sources exist, such as the *OpenStreetMap* project [39].

3. PROCESS

3.1 Project Management

The hexacopter project is a team project with four individuals contributing software. Each team member focuses on distinct areas; one on writing fundamental control and interface systems for the hexacopter, GPS, and IMU; another on intelligent searching algorithms; and another on image processing. Much of this work is interdependent, for example, the intelligent searching algorithms depend on being able to access the hexacopter flight systems and GPS. The user interface sits at the top level, and must access and control the searching and image processing functions. In order to construct a user interface, work must be done in integrating systems designed by other team members. As such, it was necessary to establish a regime for collaboration.

3.1.1 *Team collaboration*

To facilitate multiple team members working on the same code base, Git was chosen. Git is a distributed revision control and source code management system, originally released by Linus Torvalds, the namesake of the Linux operating system [46]. A Git system allows multiple users to modify the same code base simultaneously, with each change (or *commit*) logged and saved, allowing users to compare and/or revert changes. The hexacopter Git system was set up on GitHub, a public, free Git repository. [1]

3.2 Code Base Design

To facilitate the software from different team members, the code base was heavily modularised, allowing individual algorithms and methods of control to be easily adapted to different situations. The code base was split into several levels; *base*, *modules* and *apps*. *Base* holds all the lower level control code for hexacopter flight, the GPS, IMU and buzzer. *Modules* holds higher level algorithms for flight maneuvouring, waypoint tracking, intelligent search patterns, and image processing functionality. *Apps* holds high level applications for controlling the hexacopter, utilising combinations of various *modules* and *base* functions to perform defined tasks.

Under this scheme, illustrated in Figure 5, the user interface is an *app*; a piece of software capable of instructing the hexacopter to perform various tasks as dictated through *modules* functions.

[1]The hexacopter Git can be seen at https://github.com/crazyoldmans/picopter

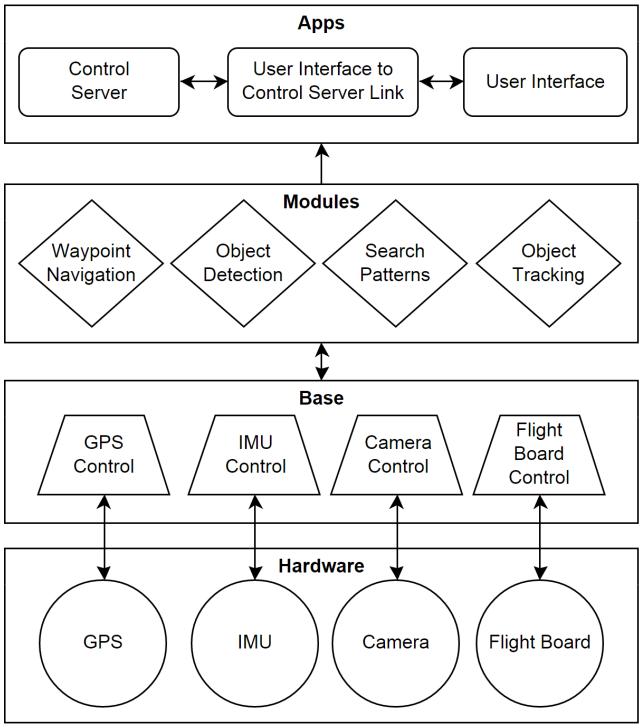

Figure 5: The structure of the collaborative code base used by the hexacopter team.

3.2.1 *Languages*

Due to the limited processing capabilities of the Raspberry Pi, and the hexacopter requirements for speed and responsiveness, C++ was chosen as the primary language for flight control programs. Compared to languages such as Java and Python, C++ is relatively low level, allowing for faster execution times and less overhead [76]. As the *base* system development was completed in C++, other *modules* and *apps* were also written in C++.

3.3 User Interface

3.3.1 *Platform*

In order to build a functional, reliable and understandable user interface for the hexacopter, several steps were taken. Firstly, a method of interface had to be chosen. As waypoint navigation is one of the objectives of the project, a visual interface capable of displaying mapping information must be used. This excluded the possibility of using the radio remote control, or some other button-based interface without visual feedback. As the hexacopter will generally be used in a variety of different environments, the device used for control must also be easily portable. This leads to several classes of portable LCD screen based devices: smart phones, tablets, and laptops.

Depending on the situation, a user may prefer to control the hexacopter 'on the go' with a smart phone or tablet, or they may prefer a more immobile setup with a larger screen through a laptop. As such, the user interface application must be cross-platform, that is, capable of running on common smart phone, tablet, and laptop computer operating systems. This requires the development of an application

to run on Microsoft Windows and Windows Phone, Apple Macintosh and iOS, and Google Android and Chrome OS. There are a variety of cross platform languages available to accomplish this task, such as Java. However, due to the restrictive nature of application development on mobile devices (see Section 2.3.3), the possibility of using an app was discounted. As such, a web interface was chosen - this will work on all required devices without the need to maintain separate code bases for each platform.

In order for a user to connect to the interface, the hexacopter was connected to a 4G LTE device, allowing the web interface to be published to the internet. Due to the inherent insecurities faced here, the web interface was locked with a password, rendering it inoperable to a malicious user. A user would be able to access the interface utilising either a Wi-Fi network or their own mobile broadband connection. Additionally, a Wi-Fi device was also installed on the hexacopter, allowing the hexacopter to broadcast its own secure 'hot spot'. A user would be able to connect to this, then navigate to the appropriate web page to control the hexacopter.

3.3.2 Graphical User Interface Design Requirements

In addition to the usability guidelines listed in Section 2.3.1, the web interface must accomplish several tasks mandated by the overall project scope:

- Display the location of the hexacopter (and the user, if possible) on a map.
- Display the hexacopter's camera feed in real time.
- Display the status of the hexacopter. This details what the hexacopter is doing, whether that be standing by, flying to a location, etc...
- Track the movement of the hexacopter on the map through a path.
- Send the user's location to the hexacopter, allowing the hexacopter to track the user.
- Allow the user to specify GPS waypoints for the hexacopter to traverse to.
- Allow the user to specify a region for the hexacopter to 'scan' over. The hexacopter will perform a 'lawnmower' pattern over this region, using the camera to search for objects of interest.
- Allow the user to tell the hexacopter to commence flight operations (either waypoint traversal, region scanning, or user tracking), and allow the user to stop the hexacopter moving at any time.

Furthermore, the web interface must be designed to be scalable, allowing detection of various screen sizes and orientations to tailor the interface to the device being used. Code must be written to give a consistent experience across the major browsers.

Table 1: User Acceptance Testing Questionnaire

Hexacopter User Experience Questionnaire Please rate the user interface in each of the following categories by circling the appropriate star.	
Clarity	☆ ☆ ☆ ☆ ☆
Concision	☆ ☆ ☆ ☆ ☆
Familiarity	☆ ☆ ☆ ☆ ☆
Responsiveness	☆ ☆ ☆ ☆ ☆
Consistency	☆ ☆ ☆ ☆ ☆
Aesthetics	☆ ☆ ☆ ☆ ☆
Forgiveness	☆ ☆ ☆ ☆ ☆
Comments and Suggestions	

3.3.3 User Acceptance Testing

To ensure the web interface operates as expected and to ascertain whether it meets the criteria in Section 2.3.1, user acceptance testing was performed. This testing involves allowing a user to control the hexacopter through the web interface, and then asking them a series of questions designed to help improve the design of the interface. These tests were performed as the the interface was developed, allowing it to develop 'organically', meeting the needs and requirements of its user base.

The 'questionnaire' given to each user (illustrated in Table 3.3.3) asked them to rate how the interface performed in each of the design criteria given in Section 2.3.1. The users were also asked to rate their experience with operating the hexacopter for each of the primary functions listed in Section 3.3.2. Furthermore, general comments, feedback and suggestions were also requested. An analysis of user responses is provided in Section 4.2.

3.3.4 Web Interface Implementation

The web interface was implemented using the broad spectrum of technologies described in Section 2.4, in combination with a server to control the functionality described in Section 3.2. This flight control server was written in C++ in order to interact with and control the functionality written by other team members for purposes such as region scanning and image processing. The overall implementation is illustrated in Figure 6.

The Lighthttpd web server was chosen over alternatives for several reasons: Lighttpd is simple to set up and operate, requiring minimal configuration compared to other web servers. Furthermore, Lighttpd is a lightweight web server, and when used in conjunction with PHP, it was found to use no more than 5-10% of the processing power of the Raspberry Pi for a short period of time (> 1 second), per request.

As the primary code base for hexacopter control is written in C++ (see Section 3.2.1) and the web interface server-side control achieved through PHP, the web interface is not natively able to communicate with the hexacopter control system. This obstacle was overcome through the use of Apache Thrift (Section 2.4.3). Apache Thrift was chosen for interprocess communication over alternatives due to its binary nature; it is fast, secure, and reliable.

In order to facilitate a mapping interface, the Leaflet JavaScript mapping library (see Section 2.5.2) was used in conjunction with tiles acquired from the OpenStreetMap project. This was chosen over alternatives such as Google Maps as the hexacopter may be operating offline (if controlled through Wi-Fi). A map of UWA was downloaded, given that the hexacopter was primarily tested on campus.

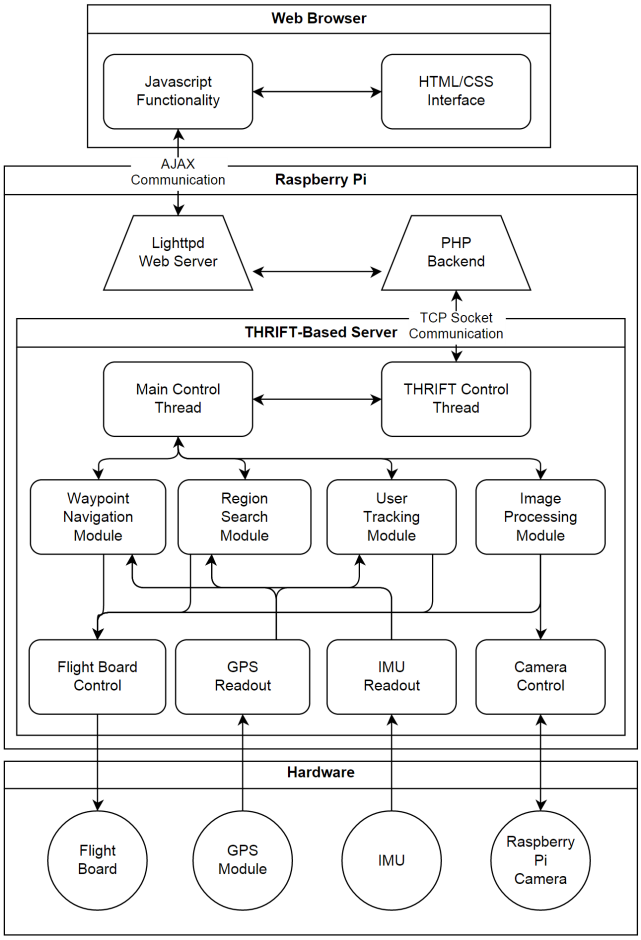

Figure 6: The structure of software and languages used in the final design, illustrating methods and directions of communication between different layers.

4. RESULTS AND DISCUSSION

4.1 Final Design

The final design of the web interface is presented in Figures 7 and 8, with a User Manual provided in Appendix A. The interface provides the functionality required by Section 3.3.2 and meets the design guidelines presented in Section 2.3.1, with supporting evidence given through user acceptance testing results in Section 4.2.

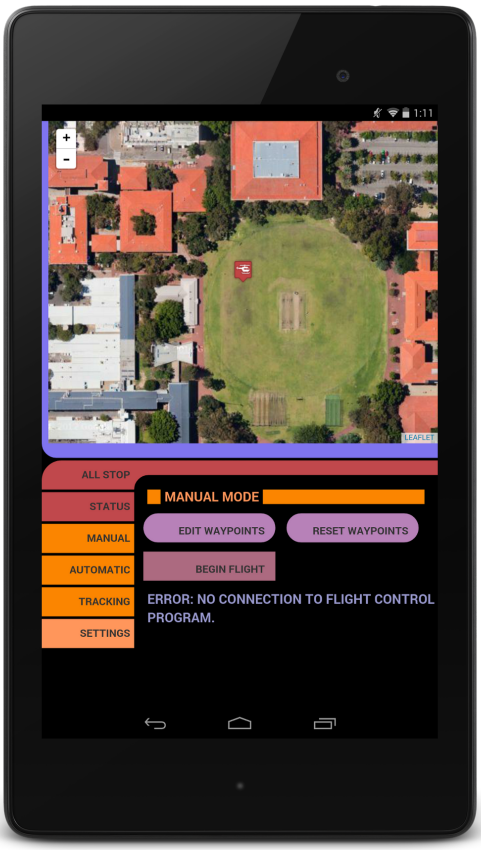

Figure 7: The final user interface, rendered on a Nexus 7 Android tablet. The user interface is shown waiting in 'Manual Mode', waiting for a connection to the hexacopter.

The interface is segmented into two main areas; a viewing window and an options menu. The orientation and further segmentation of these areas depends on the screen size and orientation with which the interface is accessed. As seen in Figure 7, the interface provides only a map and options menu in a vertical fashion on a portrait-orientated tablet or smart phone. In this scenario, the option is given to toggle the map view with a camera feed (with an overlay showing detected objects of interest). If the tablet or smart phone is rotated, the interface will adapt itself to a landscape orientation appropriately.

If used on a laptop computer, or any device with a large enough screen, the interface will detect unused screen area and attempt to fill it intelligently. For example, as shown in Figure 8, the interface has filled the bottom right area with a camera feed, meaning there is no need to toggle the map.

The options menu of the interface follows a *tabbed* design; that is, presenting the user with a series of 'modes' for overall

Figure 8: The final user interface, rendered on a Windows 8 machine through Google Chrome. The interface is shown waiting in 'Manual Mode' for user instructions. A camera feed (showing the research lab) is shown.

hexacopter functionality (waypoint traversal, region scanning, user tracking), and only allowing them to interact with one of these modes at one time. This streamlines the interface, and reduces the potential for error or confusion. However, a button to stop the hexacopter is always available on the screen, ensuring the user can stop the hexacopter regardless of the mode they're viewing.

The web interface is kept constantly informed by the hexacopter as to what it is doing, for example, traversing to a waypoint, scanning a region, or standing by. This information is displayed to the user at all times as the current 'status' of the hexacopter, adding to the dynamic nature of the web interface. Furthermore, the current location of the hexacopter and user are continuously displayed on the map, along with a path showing the previous movements of the hexacopter. The user may choose to toggle and reset the display of this path through an options tab.

The user is able to specify waypoints for traversal, or a region for scanning over. This is implemented through a 'lock-down' technique. The user specifies they wish to enter waypoints by pressing a button, causing all buttons (except the 'edit' and 'stop' buttons) to disable themselves on the interface. Any touch on the map after this point will add a new waypoint. Waypoints can also be dragged around the map to have their location changed, or tapped again to be deleted. This functionality is illustrated in Figure 12, Appendix A. A region is added in much the same way, except only two waypoints can be specified and changed. The web interface will automatically draw a 'box', showing the region to be scanned (Figure 13, Appendix A).

When the hexacopter traverses between waypoints, the next waypoint is indicated through a different colour and status message, with already 'completed' waypoints indicated through a different colour change (to red). This functionality is illustrated in Figure 9.

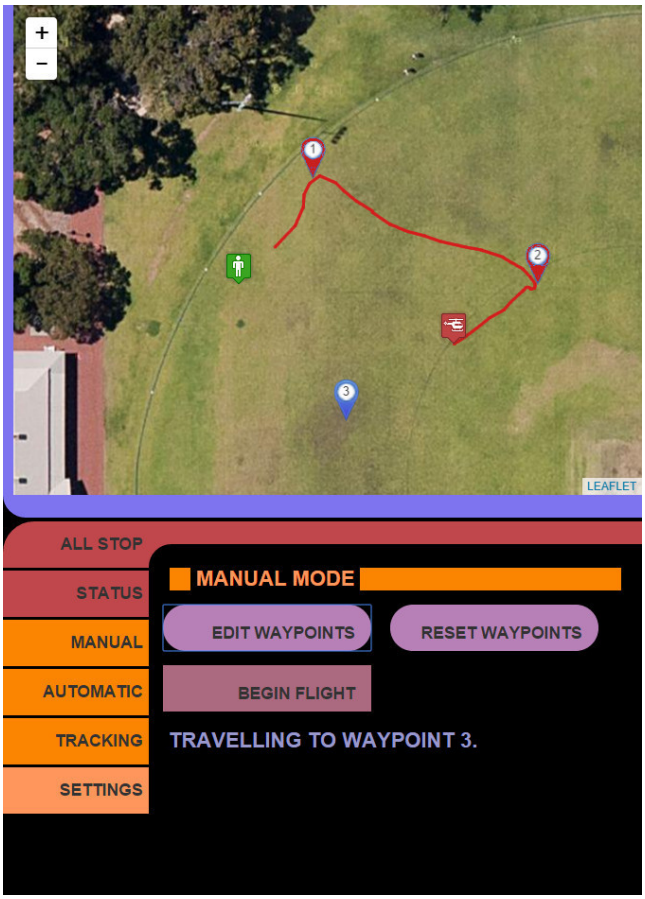

Figure 9: Screenshot illustrating the hexacopter traversing waypoints.

4.2 User Acceptance Testing Results

As described in Section 3.3.3, user acceptance testing was performed throughout the development of the interface. Users were requested to complete a questionnaire, illustrated in Table 3.3.3. The users chosen for testing were selected from a diverse a sample as possible; however this was restricted by factors such as the battery life of the hexacopter (10-15 minutes flight time per battery) coupled with the availability of candidates at times when the hexacopter was able to be flown (and not undergoing software or hardware changes).

A spread of 15 different users trialled the interface throughout its development. Trials were conducted over ten weeks (weeks 32 to 41 of the year), with at least two trials occurring per week. A total of 22 trials were conducted, with some of the 15 users trialling the interface twice. As the study continued, suggestions were taken into account and developed, resulting in a general trend for mean scores to increase. A graph of the mean trial scores from each week is shown in Figure 10.

The early iterations of the interface were simple; with the major overhaul to resemble the final interface occurring by Week 36. The results of this overhaul are evident in Figure 10; a significant improvement. It was found that, on average,

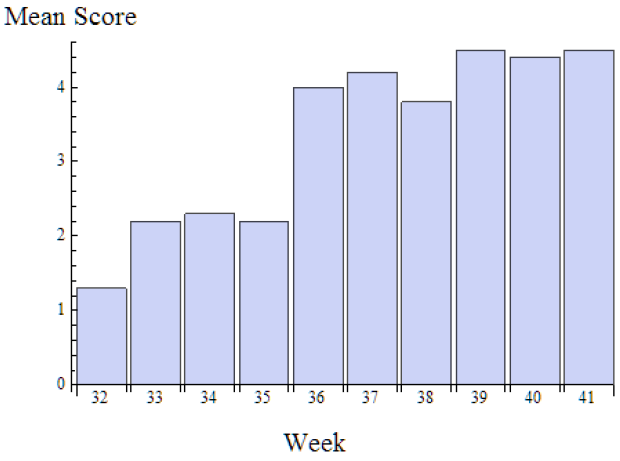

Figure 10: Mean user rating of the web interface out of five as it was developed over ten weeks.

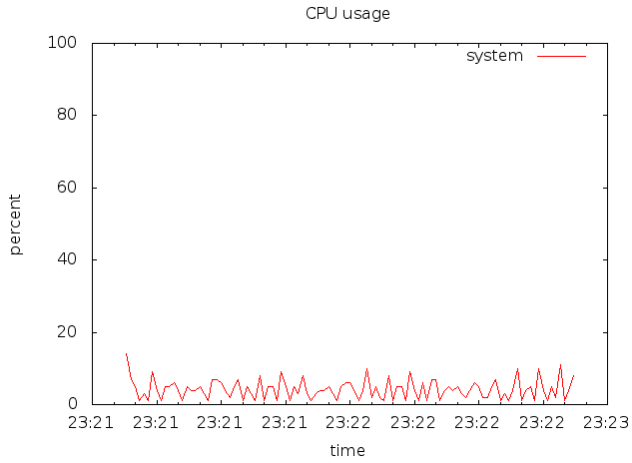

Figure 11: Raspberry Pi CPU load over a period of two minutes when running the flight control server with a user connected to the web interface.

scores for clarity and concision remained high throughout development, whilst responsiveness, aesthetics and consistency improved as the study progressed.

Through utilising this method of progressive user acceptance testing, the web interface design was able to progress into a highly polished and proven form.

4.3 Response Times

In keeping with the design requirements of the web interface, it is essential that the interface control the hexacopter rapidly; especially in dangerous scenarios (for example, sending a sudden stop command to avoid a crash). Furthermore, the resource usage of the web interface and control server in terms of CPU and memory usage were to be kept to a minimum, to allow the Raspberry Pi ample computational power to smoothly calculate flight plans and perform image processing.

In addition to concerns over computational power, network latency must also be taken into account. This concern is especially valid for when the hexacopter is controlled through 4G network; cellular networks have the potential to be unreliable as compared to conventional Wi-Fi networks.

To achieve these ends, lightweight design choices were made throughout the construction of the web interface, for example, the use of the Lighttpd web server, the binary communication protocol Apache Thrift, and the use of the relatively low level language C++ for core flight control (see Sections 3.2.1 and 3.3.4).

Figure 11 illustrates the CPU demand through normal usage of the web interface with the flight control server running. It is seen to be consistently at roughly 5-10% usage, leaving most of the CPU power available for other flight programs to utilise.

When Wi-Fi is used, hexacopter response times are found to be fairly constant at 200 milliseconds. When 4G LTE is used, response times are consistent at 250 milliseconds. These times are below half a second, and were found to give the user the impression of a responsive system through the user acceptance testing survey described in Section 4.2.

5. CONCLUSIONS AND FUTURE WORK

This report presents a highly polished and tested user interface and control server software package enabling autonomous control of a UAV. This software was applied to a DJI F550 + NAZA M hexacopter package which was modified to enable control through a Raspberry Pi computer. This software package allows a user to interface with a UAV through a variety of input devices, such as smart phones, tablets, and personal computers. The functionality to specify waypoints for UAV to traverse and regions for the UAV to scan over utilising a camera is provided through the interface.

The interface was developed utilising existing web technologies for a uniform experience across a multitude of devices and browsers. Capable of working offline, the interface provides offline mapping data and live video feeds from a connected UAV. UAVs may be connected to the interface through a Wi-Fi network, or through 4G LTE, allowing for control at great distances.

The interface is non-resource intensive; utilising resource-light technologies and techniques to operate such as *Lighttpd*, *Apache Thrift*, and *C++*, as demonstrated in Section 4.3. Furthermore, utilising the recommended methods of communication with the hexacopter results in sub 0.5 second response times.

5.1 Applications

As described in Section 1.1, there is a wide array of commercial and industrial applications for UAVs, including parcel delivery, agriculture, and filming and surveillance. Utilising the software package presented in this report, UAVs can be programmed to automatically scan over regions of land for objects of interest. This is especially applicable to agriculture, where UAVs can be fitted with specialised spectral cameras, allowing for the automated detection of irregulari-

ties within crop fields. Furthermore, utilising the navigation capabilities presented through this software, UAVs can be used for small manual package delivery and filming.

5.2 Future Work

The interface described in this report allows the user to specify UAV tasks for completion, along with informing the user about the current status of the UAV. This interface is expandable to accommodate additional flight algorithms as they are developed, for example, to allow the UAV to perform specialised aerial manoeuvres.

The installation of a lightweight longwave infrared sensor on UAV could be used to scan for regions of interest in temperature, leading to applications in the agricultural sector. An example of such a sensor is the FLIR Lepton compact sensor.

The QStarz GPS used on the hexacopter can be upgraded to the Swift Navigation Piksi GPS receiver. This will allow for centimetre level GPS accuracy, allowing the hexacopter to perform advanced, intricate flight manoeuvres.

Further interface improvements are possible; for example, an automated 'return to home' feature could be implemented, in the event that the UAV leaves the range of wireless reception. Furthermore, the possibility for further integration with various flight algorithms is possible. For example, the interface could offer the user the possibility to tune parameters used in an automated search operation.

A new and upcoming technology is the *Oculus Rift*, a virtual reality headset [79]. It would be possible to adapt the user interface to display through this, allowing a user full immersion with the activities of the UAV.

6. REFERENCES

[1] Proceedings of 2004 ieee international symposium on spread spectrum techniques and applications, sydney, australia. *IEEE International Symposium on Spread Spectrum Techniques and Applications*, September 2004.

[2] E-commerce giant amazon seeks faa nod for testing drones, July 2014. [Online; posted 12-July-2014] `http://www.seattlebulletin.com/index.php/sid/223727243`.

[3] V. Agafonkin. Leaflet, October 2014. [Online; retrieved 20-October-2014] `http://leafletjs.com/`.

[4] A. Alaimo, V. Artale, C. Milazzo, A. Ricciardello, and L. Trefiletti. Mathematical modeling and control of a hexacopter. *2013 International Conference on Unmanned Aircraft Systems*, pages 1043–1050, May 2013.

[5] J. D. Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill, 2007.

[6] Arduino. Arduino, 2014. [Online; retrieved 20-October-2014] `http://www.arduino.cc/`.

[7] ARM Limited. Arm11 mpcore processor, revision r1p0 - technical reference manual, 2008.

[8] Atmel Corperation. At91sam arm-based embbedded mpu - sam9260 datasheet, 2012.

[9] Atmel Corperation. Atmel 8-bit avr microcontroller with 2/4/8k bytes in-system programmable flash - attiny25/v / attiny45/v / attiny85/v datasheet, 2014.

[10] S. M. A. (Australia). *Australian Radiofrequency Spectrum Plan : including general information*. Canberra, ACT : Australian Govt. Pub. Service, 1997.

[11] S. S. Bakken, Z. Suraski, and E. Schmid. *PHP Manual: Volume 2*. iUniverse, Incorporated, 2000.

[12] R. Ballagas, J. Borchers, M. Rohs, and J. G. Sheridan. The smart phone: a ubiquitous input device. *Pervasive Computing, IEEE*, 5(1):70–77, 2006.

[13] R. Baranek and F. Solc. Modelling and control of a hexa-copter. *13th International Carpathian Control Conference (ICCC)*, 2012.

[14] P. Baronti, P. Pillai, V. W. Chook, S. Chessa, A. Gotta, and Y. F. Hu. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards. *Computer communications*, 30(7):1655–1695, 2007.

[15] S. F. Barrett. *Arduino Microcontroller Processing for Everyone! Third Edition*. Synthesis Lectures on Digital Circuits and Systems, August 2013.

[16] G. Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *Selected Areas in Communications, IEEE Journal on*, 18(3):535–547, 2000.

[17] M. Bryson, A. Reid, F. Ramos, and S. Sukkarieh. Airborne vision-based mapping and classification of large farmland environments. *Journal of Field Robotics*, 27:632–655, 2010.

[18] J. M. Carroll. Human–computer interaction. *Encyclopedia of Cognitive Science*, 2009.

[19] A. Charland and B. Leroux. Mobile application development: web vs. native. *Communications of the ACM*, 54(5):49–53, 2011.

[20] E. Commission. *European Commission calls for tough standards to regulate civil drones*. 2014. [Online; retrieved 20-October-2014] `http://europa.eu/rapid/press-release_IP-14-384_en.htm`.

[21] E. Dahlman, S. Parkvall, and J. Skold. *4G: LTE/LTE-advanced for mobile broadband*. Academic Press, 2013.

[22] Deagel.com. Mq-1 predator / mq-1b, mq-1l block 10, May 2014. [Online; retrieved 20-October-2014] `http://www.deagel.com/Unmanned-Combat-Air-Vehicles/MQ-1-Predator_a000517002.aspx`.

[23] A. Dix. *Human-computer interaction*. Springer, 2009.

[24] L. DongBin, T. C. Burg, X. Bin, and D. M. Dawson. Output feedback tracking control of an underactuated quad-rotor uav. *American Control Conference*, pages 1775–1780, 2007.

[25] D. Elliot. Dhl testing delivery drones, December 2013. [Online; posted 9-December-2013] `http://www.cbsnews.com/news/dhl-testing-delivery-drones/`.

[26] K. R. Fall and W. R. Stevens. *TCP/IP illustrated, volume 1: The protocols*. addison-Wesley, 2011.

[27] R. Faludi. *Building wireless sensor networks: with ZigBee, XBee, arduino, and processing.* " O'Reilly Media, Inc.", 2010.

[28] R. Fielding. Representational state transfer. *Architectural Styles and the Design of Netowork-based Software Architecture*, pages 76–85, 2000.

[29] R. T. Fielding and G. Kaiser. The apache http server project. *Internet Computing, IEEE*, 1(4):88–90, 1997.

[30] B. Fuest. Dhl testet erstmals paketlieferung per drohne, December 2013. [Online; posted 9-December-2013] `http://www.welt.de/wirtschaft/article122747484/DHL-testet-erstmals-Paketlieferung-per-Drohne.html`.

[31] Futaba. 14sg, October 2014. [Online; retrieved 20-October-2014] `http://www.futaba-rc.com/systems/futk9410-14sg/`.

[32] L. R. Garcia-Carrillo, E. Rondon, A. Dzul, A. Sanche, and R. Lozano. Hovering quad-rotor control: A comparison of nonlinear controllers using visual feedback. *49th IEEE Conference on Decision and Control (CDC)*, pages 1662–1667, 2010.

[33] J. J. Garrett et al. Ajax: A new approach to web applications. 2005.

[34] J. G. V. Ghadiok and W. Ren. On the design and development of attitude stabilization, vision-based navigation, and aerial gripping for a low-cost quadrotor. *Autonomous Robots*, 33:41–68, 2012.

[35] C. A. Gough, R. Green, and M. Billinghurst. Accounting for user familiarity in user interfaces. In *Proceedings of the 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI*, pages 137–138. ACM, 2006.

[36] I. S. Graham. *The HTML sourcebook*. John Wiley & Sons, Inc., 1995.

[37] C. Guowei and et al. Modeling and control of the yaw channel of a uav helicopter. *Industrial Electronics, IEEE Transactions*, 55(3):3426–3434, 2008.

[38] S. Gupte, P. I. T. Mohandas, and J. M. Conrad. A survey of quadrotor unmanned aerial vehicles. *Proceedings of IEEE Southeastcon*, pages 1–6, 2012.

[39] M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *Pervasive Computing, IEEE*, 7(4):12–18, 2008.

[40] IGS Central Bureau. International gnss service, 2014. [Online; retrieved 20-October-2014] `http://www.igs.org/`.

[41] T. Jensen, L. Zeller, and A. Apan. The use of an unmanned aerial vehicle as a remote sensing platform in agriculture. *Australian Journal of Multi-disciplinary Engineering*, 8(2):139–146, 2011.

[42] S. Kerr. Uae to develop fleet of drones to deliver public services, February 2014. [Online; posted 12-February-2014] `http://www.ft.com/cms/s/0/7f65fb32-9270-11e3-8018-00144feab7de.html#axzz2t3xx0VLS`.

[43] J. Kneschke. Lighttpd, 2003.

[44] B. Laurel and S. J. Mountford. *The art of human-computer interface design*. Addison-Wesley Longman Publishing Co., Inc., 1990.

[45] P. D. LLC. Drones for agricultural crop surveillance, October 2014. [Online; retrieved 20-October-2014] `http://precisiondrone.com/drones-for-agriculture.html`.

[46] J. Loeliger and M. McCullough. *Version Control with Git: Powerful tools and techniques for collaborative software development*. " O'Reilly Media, Inc.", 2012.

[47] A. C. Madrigal. Inside google's secret drone-delivery program, August 2014. [Online; posted 14-August-2014] `http://www.theatlantic.com/technology/archive/2014/08/inside-googles-secret-drone-delivery-program/379306/?single_page=true`.

[48] A. Mesbah and M. R. Prasad. Automated cross-browser compatibility testing. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 561–570. ACM, 2011.

[49] S. Murugesan, Y. Deshpande, S. Hansen, and A. Ginige. Web engineering: A new discipline for development of web-based systems. In *Web Engineering*, pages 3–13. Springer, 2001.

[50] NASA Jet Propulsion Laboratory. Global differential gps (gdgps) system, 2014. [Online; retrieved 20-October-2014] `http://www.gdgps.net/`.

[51] National Coordination Office for Space-Based Positioning, Navigation, and Timing. Gps accuracy, 2014. [Online; retrieved 20-October-2014] `http://www.gps.gov/systems/gps/performance/accuracy/`.

[52] C. Nedelcu. *Nginx HTTP Server: Adopt Nginx for Your Web Applications to Make the Most of Your Infrastructure and Serve Pages Faster Than Ever*. Packt Publishing Ltd, 2010.

[53] R. O'Conner. Developing a multicopter uav platform to carry out research into autonomous behaviours, using on-board image processing techniques. Final year project thesis, The University of Western Australia, November 2013.

[54] Parrot SA. Ar.drone 2.0, May 2014. [Online; retrieved 20-October-2014] `http://http://ardrone2.parrot.com/`.

[55] PrecisionHawk. Lancaster platform, October 2014. [Online; retrieved 20-October-2014] `http://precisionhawk.com/`.

[56] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey. *Human-computer interaction*. Addison-Wesley Longman Ltd., 1994.

[57] J. Primicerio et al. A flexible unmanned aerial vehicle for precision agriculture. *Precision Agriculture*, 13(4):517, August 2012.

[58] C. Pty.Ltd. Coptercam, October 2014. [Online; retrieved 20-October-2014] `http://www.coptercam.com.au/`.

[59] Qstarz International Co. Ltd. Qstarz bt-q818x, 2014. [Online; retrieved 20-October-2014] `http://www.qstarz.com/Products/GPS%20Products/BT-Q818X-F.htm`.

[60] Raspberry Pi Foundation. Raspberry pi, 2013. [Online; retrieved 20-October-2014] `http://www.raspberrypi.org/`.

[61] Raspberry Pi Foundation. Camera module, 2014. [Online; retrieved 20-October-2014] `http://www.raspberrypi.org/products/camera-module/`.

[62] Raspbian. Raspbian, 2014. [Online; retrieved 20-October-2014] `http://www.raspbian.org/`.

[63] E. S. Raymond. *The art of Unix programming*. Addison-Wesley Professional, 2003.

[64] T. Register. New raspberry pi b+, October 2014.

[Online; retrieved 20-October-2014] http://www.theregister.co.uk/2014/07/14/raspberry_pi_b_debuts_with_four_usb_ports/.
[65] K. Robillard and A. Byers. Amazon drones: Obstacles to the bezos dream, December 2013. [Online; posted 2-December-2013] http://www.politico.com/story/2013/12/obstacles-to-the-jeff-bezos-drone-dream-100536.html.
[66] P. E. Ross. Chris anderson's expanding drone empire, February 2014. [Online; posted 27-February-2014] http://spectrum.ieee.org/aerospace/aviation/chris-andersons-expanding-drone-empire.
[67] J. W. Satzinger and L. Olfman. User interface consistency across end-user applications: the effects on mental models. *Journal of Management Information Systems*, pages 167–193, 1998.
[68] SteadiDrone. Steadidrone, October 2014. [Online; retrieved 20-October-2014] http://www.steadidrone.com/.
[69] A. Sumaray and S. K. Makki. A comparison of data serialization formats for optimal efficiency on a mobile platform. In *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, page 48. ACM, 2012.
[70] G. Svennerberg, M. Wade, C. Andres, S. Anglin, M. Beckner, E. Buckingham, G. Cornell, J. Gennick, J. Hassell, M. Lowman, et al. *Beginning Google Maps API 3*. Springer, 2010.
[71] Swift Navigation Inc. Piksi, 2014. [Online; retrieved 20-October-2014] http://swiftnav.com/piksi.html.
[72] The Apache Software Foundation. Apache http server project, 2014. [Online; retrieved 20-October-2014] http://httpd.apache.org/.
[73] The Wi-Fi Alliance. Wi-fi, 2014. [Online; retrieved 20-October-2014] http://www.wi-fi.org/.
[74] Z. Tianguang, K. Ye, M. Achtelik, K. Kuhnlenz, and M. Buss. Autonomous hovering of a vision/imu guided quadrotor. *International Conference on Mechatronics and Automation (ICMA)*, pages 2870–2875, 2009.
[75] E. Upton and G. Halfacree. *Raspberry Pi User Guide Second Edition*. Wiley, August 2013.
[76] T. L. Veldhuizen and M. E. Jernigan. Will c++ be faster than fortran? In *Scientific Computing in Object-Oriented Parallel Environments*, pages 49–56. Springer, 1997.
[77] C. Venables. Multirotor unmanned aerial vehicle autonomous operation in an industrial environment using on-board image processing. Final year project thesis, The University of Western Australia, November 2013.
[78] R. Ventura and P. U. Lima. Search and rescue robots: The civil protection teams of the future. *Third International Conference on Emerging Security Technologies (EST)*, pages 12–19, 2012.
[79] O. VR. Oculus rift, October 2014. [Online; retrieved 20-October-2014] http://www.oculus.com/rift/.
[80] W3Techs. Usage of web servers for websites, October 2014. [Online; retrieved 20-October-2014] http://w3techs.com/technologies/overview/web_server/all.
[81] B. Ware et al. *Open Source Development with LAMP: Using Linux, Apache, MySQL and PHP*. Addison-Wesley Longman Publishing Co., Inc., 2002.
[82] Wikipedia. File:enterprise-e lcars.jpg, October 2014. [Online; retrieved 20-October-2014] http://en.wikipedia.org/wiki/File:Enterprise-E_LCARS.jpg.
[83] XAircraft. X650 flight style/en, October 2014. [Online; retrieved 20-October-2014] http://wiki.xaircraft.com/en-us/X650_Flight_Style/en.
[84] Y. Xiao. Ieee 802.11 n: enhancements for higher throughput in wireless lans. *Wireless Communications, IEEE*, 12(6):82–91, 2005.
[85] J. Zheng and M. J. Lee. A comprehensive performance study of ieee 802.15. 4, 2004.

APPENDIX

A. WEB INTERFACE USER MANUAL

This manual describes the operation of the DJI F550 + NAZA M hexacopter package modified for autonomous operation as described this report. The hexacopter is shown pictured in Figure 3, Section 1.2.1.

A.1 Setting Up

To power on the hexacopter, attach a charged battery to the underside of the base and connect the battery leads. Then, flick the switch atop the hexacopter. If powered, the hexacopter will play a short startup melody and prepare itself for operation. Additionally, ensure that the Qstarz GPS module is powered on. Once the hexacopter autonomous control software has started, the hexacopter will beep once. At this point, you may connect to the hexacopter using a web enabled device.

A.2 Connecting to the Hexacopter

There are two methods of connection to the hexacopter; either through the 4G LTE module or through local Wi-Fi.

A.2.1 4G LTE Connection

Once connected to the cellular network, the hexacopter web interface will be accessible from any internet connected device through the following URL:

http://picopter.mooo.com

If using 4G LTE, for security reasons, the interface will prompt for a password. By default, this is `qwerty12345`.

A.2.2 Wi-Fi Connection

The hexacopter will broadcast a WPA encrypted wireless network hotspot called `picopter`. The default password is `qwerty12345`. Once connected to this network, the web interface will be accessible from the following URL:

http://10.5.5.1

A.3 Manual Hexacopter Operation

For safety purposes, the hexacopter will not take off or change altitude by itself. This functionality must be controlled manually through the Futaba T14SG remote control.

To launch the hexacopter, place it on flat ground with no obstructions within a 2 metre radius. Tilt both joysticks of the remote control inwards and downwards to start the motors, then release and tilt the left joystick up slowly to gain altitude. The hexacopter will hover in place unless instructed to move.

The hexacopter has two modes defined by the remote control; manual and automatic, corresponding to the top right switch being down and up. In manual mode, the hexacopter will not control itself autonomously, and will allow the remote control to send telemetry. In automatic mode, the remote control will have no control except for altitude, whilst the hexacopter controls yaw and pitch.

A.4 Device Orientation and Screen Size

The web interface will automatically scale itself to suit the screen size and orientation of the device you are using. If used on a portrait orientated device, the display will consist of a map and an options menu, vertically aligned. To toggle the map with the camera in this orientation, select the `Options` tab, and select `Toggle Camera`.

If the interface is used on a landscape orientated device with sufficient resolution, the camera feed will display in the bottom right corner of the screen automatically.

A.5 Operating Modes

To change between different operating modes of the interface, select one of the tabs listed on the side of the overall menu structure.

A.5.1 All Stop

This button will stop the hexacopter moving, and cause it to hover in place. This button is always accessible.

A.5.2 Status

This tab details the detailed status of the hexacopter. Information such as bearing, state, GPS location, and latency are given.

A.5.3 Manual

This tab enables manual waypoint traversal mode. To add waypoints, tap `Edit Waypoints`. The rest of the buttons in the options menu will disable, except for the `All Stop`, `Edit Waypoints` and `Reset Waypoints` buttons. At this point, tap anywhere on the map to add a new waypoint. This is illustrated in Figure 12. Tapping again on a waypoint will delete it, and dragging a waypoint will move relocate it. Once all waypoints have been configured, tap `Edit Waypoints` once more to leave the editing mode.

Tap `Reset Waypoints` at any time to remove all waypoints on the screen.

Tap `Begin Flight` to start the hexacopter traversing the waypoints.

A.5.4 Automatic

This tab enables automatic region scanning mode. To add a region, tap `Edit Boundaries`. The rest of the buttons in the options menu will disable, except for the `All Stop`, `Edit Boundaries` and `Reset Boundaries` buttons. At this point, tap anywhere on the map to add the first corner an area, and tap again to add the second. The web interface will draw a box between these two points, showing the region to be scanned over. This is illustrated in Figure 13. Tapping again on a corner of the box will delete it, and dragging a corner will resize the box. Once the boundaries have been configured, tap `Edit Boundaries` once more to leave the editing mode.

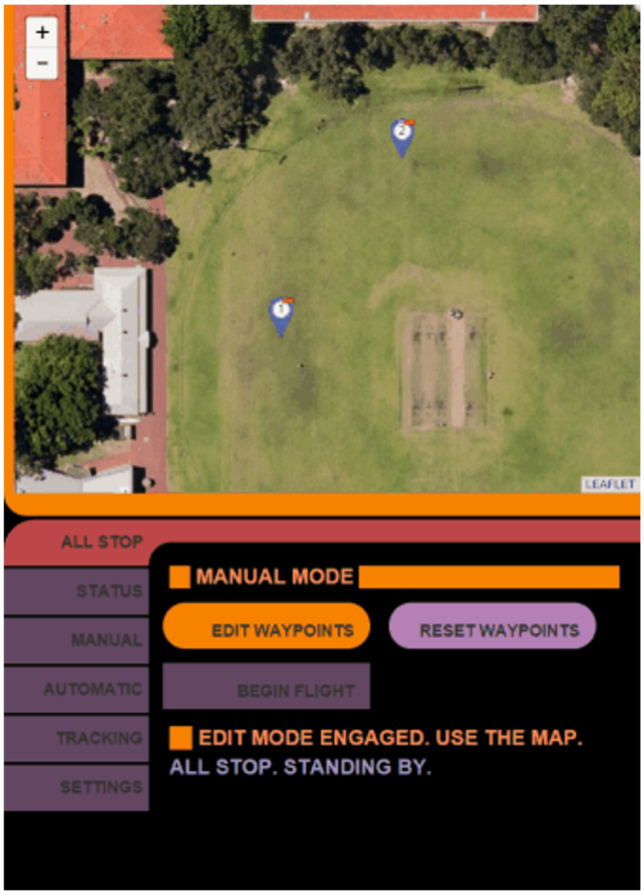

Figure 12: Screenshot illustrating waypoints being added to the web interface.

Tap `Reset Boundaries` at any time to remove the boundaries on the screen.

Tap `Begin Flight` to start the hexacopter scanning the specified region.

A.5.5 Tracking

This tab enables automatic user tracking mode. This mode will only function if the hexacopter is controlled from a device with an inbuilt GPS. Tap `Begin Tracking` to have the hexacopter move towards the control device. The hexacopter will then follow the device at a safe distance.

A.5.6 Settings

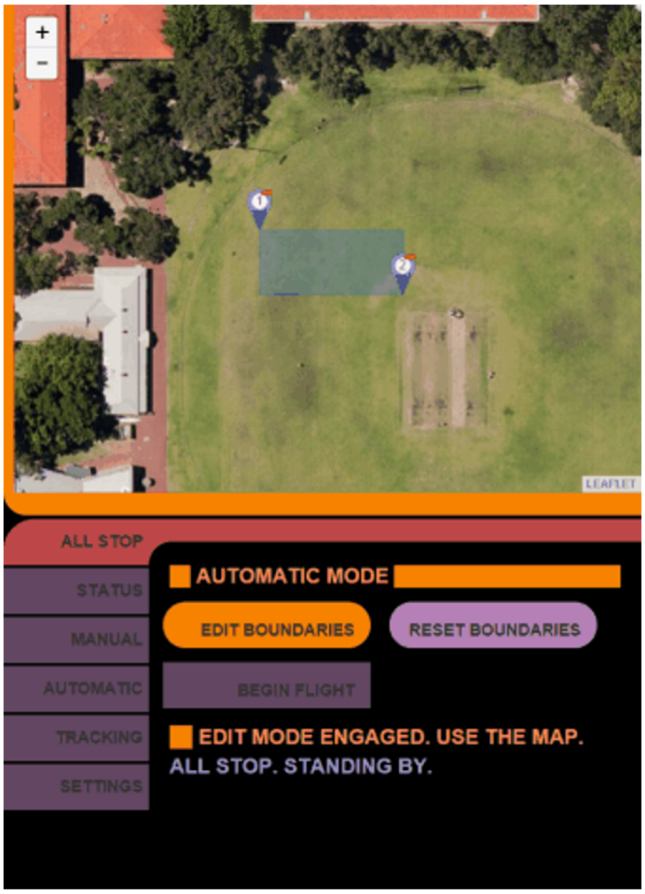

Figure 13: Screenshot illustrating a bound region being added to the web interface.

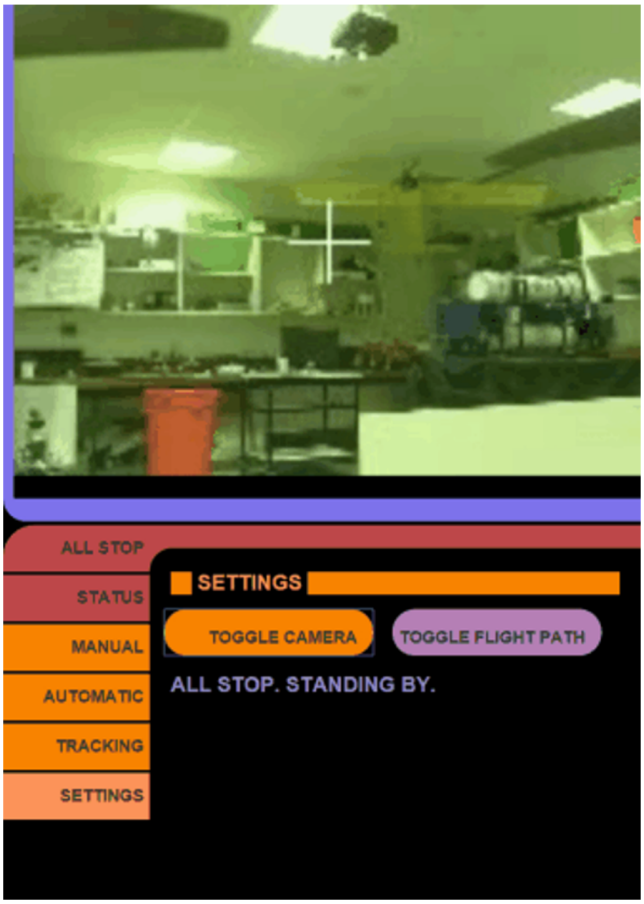

Figure 14: Screenshot illustrating toggling the camera display in the portrait orientation.

This tab allows the adjustment of settings for the web interface. If using the interface from a lower resolution device, tap `Toggle Camera` to toggle the map view with a camera feed from the hexacopter, illustrated in Figure 14. Tap `Toggle Flight Path` to toggle display of the red flight path of the hexacopter.

A.6 Flight Safety Procedures

The hexacopter will **not** move unless the remote control is switched to automatic mode, as described in Section A.3. If the hexacopter is requested to start moving, and the remote control is in manual mode, the interface will display a prompt. The hexacopter will then wait for automatic mode to engage before moving.

If an IMU (Inertial Measurement Unit) is not installed on the hexacopter, it will perform a bearing test before commencing any flight manoeuvres. The bearing test involves moving in a straight line for 5 seconds. The interface will warn before the hexacopter performs this test. Ensure that the front of the hexacopter (denoted by two red coloured arms) is facing away from any obstructions before starting this test.

A.7 Shutting Down

To cease flight, ensure that the hexacopter is not moving by pressing the `All Stop` button on the interface. Then, using the remote control, lower the altitude slowly until the hexacopter touches the ground. Move the altitude control stick directly downwards to disengage the motors. The hexacopter can now be safely powered off.

A.8 Troubleshooting

A.8.1 The web interface doesn't display correctly

Ensure that you are using a modern browser such as Google Chrome, Mozilla Firefox, Apple Safari or Microsoft Internet Explorer, and that it is up to date. If this does not resolve the problem; refresh the webpage manually.

A.8.2 No connection can be made through 4G LTE

The domain name is configured to redirect to the public IP address of the hexacopter, which can change occasionally. Wait for the IP address change to propagate to DNS servers (a few minutes), then retry connection.

A.8.3 The hexacopter beeps three times instead of flying

Ensure that the Qstarz GPS module is switched on. If this is the case, the module has not aquired a GPS lock, and the

hexacopter cannot perform navigation. Wait for the GPS module to aquire a lock.

A.8.4 The hexacopter doesn't take off

The DJI NAZA-M flight controller comes with its own integrated GPS, used as a safety feature. The hexacopter will not take off unless this achieves a GPS lock. Wait for this GPS module to aquire a lock.

A.8.5 The hexacopter does not hover in place, but moves erratically

Land the hexacopter immediately. Perform an IMU calibration using the DJI NAZA-M flight software. Ensure that the hexacopter IMU is aligned correctly; it may need to be rotated.

B. UAV CONTROL SOFTWARE

The complete suite of UAV control software is available at:

`https://github.com/crazyoldmans/picopter`

The web interface described in this report is available from:

`https://github.com/crazyoldmans/picopter/tree/master/www`

The control server software is available from:

`https://github.com/crazyoldmans/picopter/tree/master/www-waypoints`

C. WEB INTERFACE DESIGN INSPIRATION

The web interface design is inspired by a fictional computer system called *LCARS*, based on the *Star Trek* entertainment franchise. This design was chosen as it is a fluid, simple and visually appealing design, and will hold value to certain users of the system. Aesthetics, as mentioned in Section 3.3.2, are a major design consideration and feature. An image, for comparison, is given in Figure 15.

Figure 15: A screenshot from *Star Trek Nemesis*, illustrating the LCARS computer display system the web interface in this report is inspired by. (Wikipedia, 2014 [82])