Universal Control Methodology Design and Implementation for Unmanned Vehicles

Submitted by
Phang Swee King

Department of Electrical & Computer Engineering

In partial fulfillment of the requirements for the Degree of Bachelor of Engineering
National University of Singapore
Abstract

This project aims to develop a universal control methodology which can be universally applied to different type of unmanned vehicles. As a starting point, an unmanned aerial vehicle (UAV) is to be constructed as the test bed for this controller. A co-axial radio-controlled (RC) helicopter is chosen as the basic helicopter. A simple onboard avionics system is designed. The avionics system includes two Gumstix Linux computer systems as the onboard processors and an inertial measurement unit MNAV100CA for attitude measurement. A pair of XBee wireless modules is used for communication data link between the onboard system and the ground control system. A camera is used to capture real time video during the flight for onboard image processing. Important data such as attitude, velocity and acceleration of the UAV, together with the real time video will be sent to the ground station via communication links for monitoring purposes. Flight tests have been carried out to verify the results of the vision processing system and to ensure the robustness of the controller in the UAV.
# Contents

1 Introduction .................................................. 1

2 Platform Design ............................................... 5

2.1 Basic Helicopter ........................................... 7

2.1.1 Parts Upgrade ....................................... 8

2.1.2 Dynamic Analysis ................................... 10

2.1.3 Takeoff Weight Test ................................ 13

2.2 Avionics System ........................................... 14

2.2.1 Onboard Processors .................................. 15

2.2.2 Inertia Measurement Unit ............................. 19

2.2.3 Ultrasonic Sonar Sensor ............................... 21

2.2.4 Onboard Camera ....................................... 22

2.2.5 Wireless Communication Modules ................. 22

2.2.6 Safety Switch ......................................... 23

2.2.7 Power Supply ......................................... 25

2.2.8 Virtual Design and Assembling of the Avionics System . 26
3 Sensors Improvement

3.1 Sensors Calibration

3.2 Orientation Measurements

3.3 Position Measurements

3.3.1 Height

3.3.2 Lateral Position

3.4 Accuracy Tests

3.4.1 Yaw, Pitch and Roll Measurements

3.4.2 Height Measurement

3.4.3 Lateral Position Measurement based on Vision

4 Control Methodologies

4.1 PID Controller

4.1.1 Transfer Function

4.1.2 Performances

4.2 Low Level Controls

4.2.1 Pitch (θ) and Roll (ϕ) Control

4.2.2 Yaw (ψ) Control

4.3 High Level Controls

4.3.1 Lateral Position (y) Control

4.3.2 Heading Angle (ψ_{ref}) Control
List of Figures

1.1 KingLion in display. .............................................. 4
1.2 Colored tracks on the ground. The ordering of colors determine the direction of travel. .............................................. 4
2.1 An example of fixed wings UAV —RQ-4 Global Hawk surveillance aircraft. .............................................. 5
2.2 An example of rotorcraft UAV —Bell Eagle Eye for US Coast Guard. .............................................. 6
2.3 A type of airship UAV. .............................................. 6
2.4 ESky Big Lama Co-Axial helicopter .............................................. 7
2.5 Esky brushless motor 3900KV .............................................. 8
2.6 3DPro Hard DuraBlade .............................................. 9
2.7 Metal shelf and swashplate .............................................. 9
2.8 Euler angles assigned to the frame of helicopter .............................................. 11
2.9 Main components in the basic helicopter —ESky Big Lama Co-Axial .............................................. 12
2.10 Co-axial helicopter at three different states: not turning, right turn, left turn. The net lifting forces for all three states are identical. .............................................. 13
2.11 Manual Takeoff Weight Test ............................................. 14
2.12 Onboard Avionics System Overview ................................. 15
2.13 Gumstix Verdex Pro (Left) and Gumstix Overo (Right) ........... 16
2.14 MNAV100CA unit ....................................................... 19
2.15 Telebee GR302-AD unit .................................................. 20
2.16 MaxSonar EZ4 unit ......................................................... 21
2.17 Onboard camera used in UAV ........................................... 22
2.18 XBee-Pro Wireless Module .............................................. 23
2.19 Failsafe multiplexer ......................................................... 24
2.20 Battery for the UAV ........................................................ 26
2.21 Avionics components and its virtual designs using SolidWorks ... 28
2.22 Physical design of the avionics system in UAV ....................... 29
2.23 Side view of UAV design with labels .................................. 30
2.24 Front view of UAV design ................................................ 30
3.1 Main windows of Micro-View software ................................. 33
3.2 Servo interface of Micro-View ............................................ 34
3.3 PPM GUI of Micro-View ................................................... 34
3.4 Orientation prediction algorithm ........................................ 35
3.5 Yaw control using headlock gyroscope .................................. 36
3.6 Acceleration measured by the MNAV100CA unit ..................... 37
3.7 Position calculated by integrate measured acceleration twice ... 38
3.8 The error in measuring actual height ........................................ 39
3.9 Retrieving actual height ....................................................... 40
3.10 Position measured by ultrasonic sonar sensor ......................... 42
3.11 Up: One of the picture captured by the camera mounted below the
UAV. Down: The assigned frames for easy calculation of heading
angle and lateral position. ..................................................... 44
3.12 Left: Without roll tilt. Right: With roll tilt. .......................... 45
3.13 Euler angles of the UAV under oscillation motions at three different
axes. ................................................................................. 47
3.14 Up: Height measured by the sonar range finder under oscillation
motion. Down: Lateral position calculated by the vision algo-
rithms under oscillation motion. .......................................... 48
4.1 Assignment of parameters to the UAV’s frame ......................... 50
4.2 PID controller .................................................................... 52
4.3 The Overall Control Blocks Diagram .................................... 53
4.4 Pitch/Roll control ................................................................. 55
4.5 Yaw control ....................................................................... 55
4.6 Lateral position control ........................................................ 56
4.7 Heading angle control .......................................................... 57
4.8 Height control .................................................................... 58
4.9 Polling method used in Extended MicroGear algorithm ............ 62
5.1 The set up of the colored track and obstacles ........................... 64
5.2 Pitch angle of the UAV ................................................. 66
5.3 Roll angle of the UAV .................................................. 67
5.4 Yaw angle of the UAV .................................................. 67
5.5 Height of the UAV ....................................................... 68
5.6 The UAV is released from hands ....................................... 69
5.7 The UAV passing through the doorway ............................... 69
5.8 KingLion made a smooth right turn at the first corner .......... 70
5.9 The UAV is raised to the height of 2.5 meters right above the symbol .......................... 70
5.10 KingLion is lowered to 1.5 meters height after the second turn . 71
5.11 The only left turn of the circuit ....................................... 71
5.12 The UAV preparing to land at the landing area .................... 72
5.13 The camera view seen from the ground station for object identifying .......... 72
5.14 Awards won in Singapore Amazing Flying Machine Competition 2010 .......................... 74
List of Tables

2.1 Important specifications of ESky Big Lama Co-Axial Helicopter . 10
2.2 Important specifications of Gumstix Verdex Pro and Gumstix Overo 18
2.3 Important specifications of XBee-Pro wireless module . . . . . 24
2.4 Power consumptions of the main components . . . . . . . . . . . 25
4.1 Control parameters for KingLion . . . . . . . . . . . . . . . . . . 50
4.2 Characteristics of P, I and D controllers . . . . . . . . . . . . . 54
4.3 Requirements to trigger an event . . . . . . . . . . . . . . . . . . 60
4.4 Possible control in an event . . . . . . . . . . . . . . . . . . . . . 60
List of Symbols and Abbreviations

\( \theta \) pitch angle
\( \phi \) roll angle
\( \psi \) yaw angle
\( \omega_x \) roll rate
\( \omega_y \) pitch rate
\( \omega_z \) yaw rate
\( x \) \( x \)-position aircraft body frame
\( y \) \( y \)-position aircraft body frame
\( z \) \( z \)-position aircraft body frame
\( a_x \) linear acceleration at \( x \)-axis
\( a_y \) linear acceleration at \( y \)-axis
\( a_z \) linear acceleration at \( z \)-axis
$m_x$  magnetometer reading at $x$-axis

$y_m$  magnetometer reading at $y$-axis

$z_m$  magnetometer reading at $z$-axis

bps  bit-per-second

CMOS  complementary metal-oxide-semiconductor

CNF  composite non-linear feedback

FPS  frame-per-second

GHz  gigahertz

GPS  global positioning system

GUI  graphical user interface

IMU  inertia measurement unit

OEM  original equipment manufacturer

PCB  printed circuit board

PID  proportional-integral-derivative

PWM  pulse width modulation

RC  radio-controlled

RF  radio frequency

UAV  unmanned aerial vehicle

VTOL  vertical take-off and landing

xi
Chapter 1

Introduction

Robust and efficient automatic control is one of the most challenging issues for unmanned vehicle development. It is also time- and labor-cost considering the different vehicles dynamics. As such, implementing a universal robust control methodology to diverse unmanned vehicles could minimize the difficulty for control law design and implementation in the future.

A mini Unmanned Aerial Vehicle (UAV) will be used as a test platform for this universal controller. Driven by the rapid development of microprocessors, sensors and actuators, unmanned aerial vehicle (UAV) are getting smaller and lighter but with more sophisticated functions. It’s tiny size realized the ability to navigate in the indoor environment. It can be widely used in remote observation of hazardous environment which is not accessible by other unmanned vehicles [1]. Besides obvious military applications, mini UAVs also serve as an excellent platform for researchers to investigate robust control theories [2] [3] [4].
The conventional control methodology involving the usage of GPS signal can no longer be realized in an indoor environment. Hence, a vision based navigation system is proposed to replace the conventional GPS navigation system. It involves the usage of vision sensors such as camera and the implementation of onboard vision processing algorithms.

Vision processing is typically a costly operation that requires large amounts of computing power. This computational requirement usually translates to bulky processing units that can be carried only by large UAVs \[5\]. A common attempt to vision-enable mini UAVs is to include a control station whereby the video feed is sent to a fixed control station and processed results are sent back to the robots \[6\] \[7\]. However, the first approach using a ground control station is tested with limited success due to the interferences and delays in communication between control station and the UAV.

To overcome this problem, the use of onboard vision processing technology with simplified vision algorithms is proposed. The solution was to put together an embedded vision system that is more powerful than the current state of art and is still small and lightweight enough for the mini-UAV.

A typical UAV should consist of the following essential parts \[8\]:

1. Physical aircraft with engines

2. An onboard avionics system to realize automatic flight control, which include

   (a) Onboard processor to collect data, implement control laws, drive actu-
ators and communication with ground stations

(b) An inertia measurement unit (IMU) to measure the altitudes of the vehicles

(c) A good communication system to provide communication link with ground stations

(d) Power supply system

3. A ground control station to monitor and schedule flight courses and collect in-flight data.

In this project, the UAV rotorcraft platform, KingLion (Fig. 1.1) is designed using these guidelines. The avionics system will take care of low-level attitude control of the UAV. It includes two processors, an inertia measurement unit, an ultrasonic sonar range finder, modems for communication purposes and a power supply system which last up to ten minutes flying time.

The ground station consists of a personal computer operating in Linux Ubuntu system. This ground station is mainly for monitoring purposes and to collect in-flight data. There will be both data link and video link between the onboard processors and the ground supporting station.

To test the feasibility of the vision based navigation and the control methodology, KingLion is required to follow a colored track which has been painted on the ground as shown in Fig. 1.2.
Figure 1.1: KingLion in display.

Figure 1.2: Colored tracks on the ground. The ordering of colors determine the direction of travel.
Chapter 2

Platform Design

UAVs can be generalized into a few categories as follows

1. Fixed Wings UAV (Fig. 2.1)

![Fixed Wings UAV](image)

**Figure 2.1:** An example of fixed wings UAV —RQ-4 Global Hawk surveillance aircraft.

2. Rotorcraft UAV (Fig. 2.2)
Figure 2.2: An example of rotorcraft UAV —Bell Eagle Eye for US Coast Guard.

3. Airship UAV (Fig. 2.3)

Figure 2.3: A type of airship UAV.

Among these three types of UAVs, rotorcraft UAV is chosen as the basic platform in this project. Its ability of vertical take off and landing (VTOL) and hovering in a fixed position allows it to perform more complicated tasks in urban
environments. Also, due to the challenging nature of helicopter control, rotorcraft UAVs serve as an excellent platform for researchers to investigate robust control theories such as non-linear control or H-infinity control \[2 \ 3 \ 4\].

2.1 Basic Helicopter

There are a few types of rotorcraft available: single rotor helicopter, co-axial helicopter, and multi-rotors helicopter. A co-axial helicopter is chosen to be the development platform for the universal controller on top of other rotorcrafts. Co-axial helicopters are generally smaller than single rotor helicopters due to the lack of a tail rotor. On the other hand, the double main rotors result in a higher up thrust force to lift the helicopter compare to the conventional single rotor
helicopters. This is preferred in the project, where the weight of the onboard controller needs to be taken into account. In this project, ESky Big Lama Co-Axial helicopter is chosen to be the basic helicopter (Fig. 2.4).

2.1.1 Parts Upgrade

To further increase the takeoff weight and to improve the performance of the helicopter, the following changes were made:

1. Motors were upgraded to 3900KV brushless motors (Fig. 2.5).

![Figure 2.5: Esky brushless motor 3900KV](image)

2. Blades were replaced by longer and stronger blades (Fig. 2.6).
3. Metal shaft and swashplate were used to replace the original plastic shaft and swashplate (Fig. 2.7).

Figure 2.7: Metal shaft and swashplate

Some important specifications of the helicopter after upgrading are shown in Table 2.1.
<table>
<thead>
<tr>
<th>Main rotor diameter</th>
<th>460mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>160mm (without tail)</td>
</tr>
<tr>
<td>Height</td>
<td>260mm</td>
</tr>
<tr>
<td>Width</td>
<td>110mm</td>
</tr>
<tr>
<td>Weight</td>
<td>352g</td>
</tr>
<tr>
<td>Power system</td>
<td>3900KV Brushless Motor ×2</td>
</tr>
</tbody>
</table>

Table 2.1: Important specifications of ESky Big Lama Co-Axial Helicopter

2.1.2 Dynamic Analysis

In order to facilitate the design of the avionics system, investigations are carried out to study the control and the dynamics of the co-axial helicopter. Before analyse the dynamic of the co-axial helicopter, some terminologies will be defined first. Fig. 2.8 shows the assignments of linear positions and Euler angles of the helicopter.

There are a sequence of Euler angles used frequently to describe the orientation of an aircraft [9]. In this project, for the co-axial helicopter, the roll angle (\( \phi \)) is defined by the rotation about the longitudinal axis; the pitch angle (\( \theta \)) is defined by the rotation about the lateral axis; while the yaw angle (\( \psi \)) is defined by the rotation about the vertical axis, in other words, the heading of the helicopter.

Beside the Euler angles, there are four commonly used servo channels in aircraft control. In general, for co-axial helicopter, the aileron channel controls the
Figure 2.8: Euler angles assigned to the frame of helicopter

roll angle and the lateral position of the aircraft; the elevator channel controls the pitch angle and the forward speed of the aircraft; the throttle channel controls the lifting force of the aircraft; and the rudder channel controls the heading or yaw angle of the helicopter. Besides the four main channels described above, the radio transceiver of ESky Big Lama Co-Axial has additional two channels for other purposes. They will be helpful in the design of the autonomous avionics system.

As shown in Fig. 2.9, the top rotors of the co-axial helicopter is attached to a stabilizer bar, while the bottom rotors are connected to a swash plate. The elevator and aileron channels are connected to two servo motors directly. The left servo is connected by the elevator channel to control the pitch angle of the helicopter, while the right servo is connected by the aileron channel to control
Figure 2.9: Main components in the basic helicopter —ESky Big Lama Co-Axial

the roll angle of the helicopter. For a co-axial helicopter, the yaw angle or the heading of the helicopter is controlled by varying the angular velocities of both upper and lower rotors. An electronic mixer is embedded with the motors speed controller such that the lifting force will not be affected when the yaw angle of the helicopter is varied.

As illustrated in Fig. 2.10 when the helicopter is turning, one of the rotor will spin faster while another spins slower which result in a net torque acting on the fuselage while maintaining its total lifting force.
Figure 2.10: Co-axial helicopter at three different states: not turning, right turn, left turn. The net lifting forces for all three states are identical.

2.1.3 Takeoff Weight Test

In the manual takeoff weight test of the helicopter (Fig. 2.11), the maximum takeoff weight of the bare helicopter (including battery) is approximately 975g. Thus the limitation on the avionics system is a maximum weight of 623g.
2.2 Avionics System

Weight and size of the onboard components are significant in the avionics system design, given the limited size and payload of the rotorcraft platform. On the other hand, performances of the system need to be guaranteed —processing speed and sampling rate of the components must be fast enough. A comprehensive survey is done on the state-of-the-art technologies, components are selected based primarily on their weight, size and performance, as well as availability and reliability.

The avionics system consists of two onboard embedded microprocessors, an
inertial measurement unit embedded with 8 channels servo driver, an ultrasonic sonar range finder, motor speed controllers, a wireless module, and a camera. Fig. 2.12 shows a general view of the avionics system.

2.2.1 Onboard Processors

The onboard embedded microprocessor is the most important component in the avionics system. It acts as a brain of the whole system — collect flight data such as height and angular rate from the sensors, process the data, then send information to the servo driver to execute appropriate control actions. Thus selecting suit-
able processors out of the available products in the industry is crucial to ensure the successful implementation of the UAV. Weight, size and performance of the processors are taken into account in choosing the onboard processors.

**Gumstix Verdex Pro, Gumstix Overo**

After much consideration, Gumstix Verdex Pro (Fig. 2.13) is chosen to be the onboard processor to realize control algorithms. Operating at a speed of 600MHz, Gumstix Verdex Pro together with the extension board — Gumstix console vx only weighs 23 grams. The extension board has three Mini-Din 8 RS-232 ports for interfacing with peripheral components. The RS-232 ports are used to interface with the IMU, the second onboard processor, and the modem for communication with the ground station. Gumstix Buildroot built from Linux 2.6 kernel is installed as the operating system. It provides a cross-compilation tool-chain to compile and generate binary executable files which is compatible to the Gumstix processors.
A 2GB Multimedia Card (MMC) is used to transfer files and to store flight data.

To realize vision based navigation, a high performance microprocessor is needed to process images captured by the camera. The main processing unit for the embedded vision platform is Gumstix Overo Fire coupled with its Overo Summit expansion board (Fig. 2.13). The attracting features of this unit include a 600 MHz processor, DSP coprocessor and Wifi connectivity. At the core of this system is a Texas Instruments OMAP3530 ARM processor, and is one of the fastest low powered embedded systems as of writing. This unit is small in physical dimensions and weighs 18 grams in total. To improve the system for vision processing task, the operating system provided by the manufacturer of Gumstix Overo has been replaced by a custom built version of the GNU/Linux operating system. Embedded GLIBC (EGLIBC) is used as the core system library instead of the conventional GNU C Library (GLIBC) which is more suited for workstations. In addition, this custom built system provides a cleaner cross compiling environment for ease of software development.

A camera will be connected to Gumstix Overo via the mini USB port from the extension board. Data obtained from the vision processing algorithms will be sent to the control platforms onboard processor via the UART port.

Table 2.2 shows important specifications of each Gumstix. Both Gumstix uses Linux Kernel 2.6 which is available for free in http://www.gumstix.net. In addition, X-loader and U-Boot files can also be downloaded in the same link. The Kernel, X-loader and U-Boot files are essential to start up the Gumstix.
<table>
<thead>
<tr>
<th></th>
<th>Gumstix Verdex Pro</th>
<th>Gumstix Overo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
<td>Marvell PXA270 with XScale</td>
<td>OMAP 3503</td>
</tr>
<tr>
<td>Speed</td>
<td>600MHz</td>
<td>600MHz</td>
</tr>
<tr>
<td>Memory</td>
<td>128MB RAM, 32MB Flash</td>
<td>256MB RAM, 256MB Flash</td>
</tr>
<tr>
<td>Software</td>
<td>Linux 2.6.21 Gumstix, OpenEmbedded</td>
<td>Linux 2.6.21 Gumstix, OpenEmbedded</td>
</tr>
<tr>
<td>Power</td>
<td>3.6–5V DC</td>
<td>3.3–5V DC</td>
</tr>
<tr>
<td>Physical Size</td>
<td>80mm × 60mm × 6.3mm</td>
<td>17mm × 58mm × 4.2mm</td>
</tr>
<tr>
<td>Weight</td>
<td>8g</td>
<td>6g</td>
</tr>
</tbody>
</table>

**Table 2.2:** Important specifications of Gumstix Verdex Pro and Gumstix Overo
2.2.2 Inertia Measurement Unit

An inertia measurement unit is an essential part for a UAV. It gives UAV the feedback of position, velocity and Euler angles. In this project, the MNAV100CA unit is chosen to be the inertia measurement unit of the UAV. In addition, a head-lock gyroscope, Telebee GR302-AD is chosen to provide more accurate heading measurement.

MNAV100CA

The MNAV100CA (Fig. 2.14) is a digital sensor system integrated with servos controller. This module has a compact dimension of $5.72 \times 4.57 \times 2.54$ millimeters with the weight of 33 grams. It contains all necessary sensors and drivers required to control a UAV despite its small size. The MNAV100CA module includes tri-axis accelerometers, tri-axis gyroscopes, tri-axis magnetometers and a GPS receiver module.

MNAV100CA is also embedded with a servo controller. It is able to drive
Telebee GR302-AD unit

Figure 2.15: Telebee GR302-AD unit

up to 8 RC servos via PWM channels. It has a 3-pin RS-232 serial port which allows the communication with the onboard processor. Performance wise, it has a switchable output rate of 50 or 100Hz which is sufficient to control our UAV.

In the absence of GPS signal in the indoor environment which is the focus of this project, the accelerometers and gyroscopes of MNAV100CA provides the output of 3-axis accelerations and angular rates for velocity and position control. Due to the nature of integration, the positions calculated using accelerometers might not be accurate. Thus, vision based system is needed to correct the $x$- and $y$-axis positions, while an ultrasonic sonar sensor MaxSonar EZ4 is used to correct the height (position in $z$-axis) measurement.

**Telebee GR302-AD**

Telebee GR302-AD is the smallest and lightest dual rate gyro in the industry as of writing. It provides both head lock and standard mode, which support digital servo with the addition of ATV tuning. It has a size of $21 \times 22 \times 15$ millimeters.
and weight of 13 grams. Despite its small size, it provides high accuracy in the UAV yaw angle control.

### 2.2.3 Ultrasonic Sonar Sensor

Ultrasonic sonar sensor works on a principle similar to radar which evaluate the distance of a target by interpreting the echoes from radio waves. It generates high frequency sound waves and evaluate the echo received back by the sensor itself. Time interval between sending the signal and receiving the echo is then calculated to determine the distance to the target. In this project, ultrasonic sonar sensor will be used to measure the height of the UAV with reference to the ground.

**MaxSonar EZ4**

The MaxSonar EZ4 (Fig. 3.10) offers range detection and ranging in a small package of 19.9 × 21.1 × 16.4 millimeters and 4.3 grams. It is capable to detect objects and provides sonar range information ranging up to 645 centimeters with the sensitivity of 2.5 centimeters. It has three interface output formats — pulse
width output, analog voltage output, and asynchronous serial digital output. To optimize the port usage of the onboard processor, analog voltage output is used to transmit data to the processor.

2.2.4 Onboard Camera

A camera is used to capture real-time onboard video for image processing purpose. A webcam module taken from a Hp pavilion DV5 series laptop (Fig. 2.17) is chosen to be mounted in the UAV. This color CMOS camera has a tiny dimension of 8 × 80 × 6 millimeters and weight of 3 grams. Despite its small size, it is capable for providing 30 FPS at 320 × 240 resolutions. It is mounted below the platform of the UAV and is face vertically downwards to capture the colored tracks on the ground.

2.2.5 Wireless Communication Modules

Wireless communication modules are needed to form communication link between the onboard system and the ground station. A pair of Xbee-Pro OEM RF modules is used to establish a data link between the onboard avionics system and the ground supporting system, while the built-in wireless module in Gumstix Overo
XBee-Pro OEM

XBee-Pro wireless module (Fig. 2.18) operates at 2.4GHz. Its indoor range of 100 meters and miniature size of 24.4 × 32.9 millimeters makes it an ideal wireless module to be implemented in the UAV. Besides, XBee-Pro can operate at the Transparent Mode, in which the module acts as a RS-232 serial cable replacement. The baud rate of the serial interface is set as 115200 bps in accordance with the Gumstix Verdex Pro computer console. Relevant specifications of XBee-Pro wireless module are given in Table 2.3.

2.2.6 Safety Switch

In the event of an emergency, such as components break down or unstable control during the flight test, it is important that a human pilot to take back manual
<table>
<thead>
<tr>
<th>Indoor Range</th>
<th>100m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Transmit Power</td>
<td>100mW</td>
</tr>
<tr>
<td>Serial Data Rate</td>
<td>1200–115200bps</td>
</tr>
<tr>
<td>Dimensions</td>
<td>24.4mm × 32.9mm</td>
</tr>
<tr>
<td>Weight</td>
<td>3g</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>2.8–3.4V</td>
</tr>
<tr>
<td>Operating Frequency</td>
<td>2.4GHz</td>
</tr>
</tbody>
</table>

**Table 2.3:** Important specifications of XBee-Pro wireless module

**Figure 2.19:** Failsafe multiplexer

control quickly and easily [5]. A simple failsafe board (Fig. 2.19) is used to trigger between automatic control and manual control. It has two 4 channels servo input ports and a 4 channels servo output port. This failsafe board operates like a relay, where there is an addition input channel to select which input port will be used as the output of the board. Since channel 1 to 4 are used for manual control the UAV, channel 5 of the receiver is used as the selector for this safety switch.
<table>
<thead>
<tr>
<th>Main components</th>
<th>Quantity</th>
<th>Current (mA)</th>
<th>Voltage (V)</th>
<th>Power consumption (mW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brushless motors</td>
<td>2</td>
<td>2000</td>
<td>11.1</td>
<td>29600</td>
</tr>
<tr>
<td>Main processors</td>
<td>2</td>
<td>500</td>
<td>5</td>
<td>5000</td>
</tr>
<tr>
<td>MNAV</td>
<td>1</td>
<td>70</td>
<td>5</td>
<td>350</td>
</tr>
<tr>
<td>Wireless data link</td>
<td>1</td>
<td>233</td>
<td>5</td>
<td>1165</td>
</tr>
<tr>
<td>Sonar</td>
<td>1</td>
<td>2</td>
<td>3.3</td>
<td>7</td>
</tr>
<tr>
<td>On-board camera</td>
<td>1</td>
<td>35</td>
<td>5</td>
<td>175</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
<td></td>
<td><strong>36297</strong></td>
</tr>
</tbody>
</table>

Table 2.4: Power consumptions of the main components

### 2.2.7 Power Supply

Power supply is needed to supply power to the bare helicopter and the avionics system. An 11.1V battery will be used as the power source of the UAV. The power consumption breakdown of the components are shown in Table 2.4.

The total power consumption is 36297 mW. Judging by 10 minutes flight time and 85% efficiency, Energy required = \(36297 \times \frac{10}{60} \times \frac{1}{0.85}\) = 7117 mWh. Using 11.1v battery, capacity of the battery = \(\frac{7117}{11.1}\) = 641 mAh.
Raiden 11.1V 900mAh 15C Li-polymer Battery

Based on the calculation above, Raiden 11.1v 900mAh 15C Li-polymer battery (Fig. 2.20) is chosen as the power supply of the UAV.

2.2.8 Virtual Design and Assembling of the Avionics System

The avionics system must be properly situated on the helicopter such that it will not affect its flying capability. It is especially significant since the avionics system is heavier than the bare helicopter. Because of the weight increase due to the addition of the avionics system, the original landing skid made by plastic is not strong enough to withstand the impact during landing. A revised landing skid is made of two layers of printed circuit boards (PCB) together with a light yet strong aluminum landing skid.
The avionics system must be designed to minimize changes on the original dynamics of the helicopter. In order to measure accurately for effective control actions, the following requirements should be met in its physical design [8]:

1. The centre of gravity of the helicopter with the avionics system should lie on the axis of the rotors.

2. The IMU should be aligned as closely to the centre of gravity of the helicopter as possible, so to minimize the error in the measurements of the angular rates and positions.

3. The IMU should be place parallel to the rotors such that the frame of the helicopter is coincide with the frame of the IMU to minimize cross coupling in the attitude measurements.

To meet the requirements above, virtual design using SolidWorks is carried out. Individual components are created as different objects in SolidWorks, and then assembled in a simulated virtual environment to ensure the feasibility of the design. Weight distribution can be calculated in SolidWorks and hence centre of gravity can be obtained. Virtual design greatly reduces effort, time and budget of the hardware assembling. Fig. 2.21 shows the SolidWorks designs of the individual components together with the real entity.

Fig. 2.22 shows the physical design of the avionics system in the UAV. The position of the components in the avionics system is shown in Fig. 2.23 (side view) and Fig. 2.24 (front view).
Figure 2.21: Avionics components and its virtual designs using SolidWorks
Figure 2.22: Physical design of the avionics system in UAV
Figure 2.23: Side view of UAV design with labels

Figure 2.24: Front view of UAV design
2.3 Ground Supporting System

The ground supporting system incorporates a ground computer running in Linux operating system to give command and collect flight data from the onboard system for monitoring. Live video streaming from the camera on the UAV will be displayed in a customized software for observation. Development of the control laws and the image processing algorithms of the UAV are also done in the ground station before transferring to the onboard microprocessors.
Chapter 3

Sensors Improvement

Hardware improvement is essential for good control actions. In order to increase the performance of the UAV, the measurement of the sensors must meet certain requirements. Performance of sensors such as IMU must be revised and improved.

3.1 Sensors Calibration

To obtain accurate measurement, precise calibrations are done to the IMU sensor. MNAV100CA unit comes with its own software —Micro-View, which enables the communication between the MNAV100CA unit and the computer. Micro-View contains a GUI which can be viewed in desktop for calibration and debugging purposes. Fig. 3.1 shows the main windows of the Micro-View software. Beside the calibrations of the gyroscopes, accelerometers, magnetometers and pressure sensor, this Micro-View software also provides calibration and trimming for servo
 motors that connect to the servo controller output pins of the MNAV100CA unit. Fig. 3.2 and 3.3 shows the interface for servo motors calibration and trimming using Micro-View software.

3.2 Orientation Measurements

The MNAV100CA provides measurements on $\omega_x$, $\omega_y$, and $\omega_z$, the 3-axis angular velocities of the UAV. In order to obtain the yaw ($\theta$), pitch ($\phi$) and roll angle ($\psi$) more accurately, a gyroscopic earth-stabilized orientation estimation algorithm is
Figure 3.2: Servo interface of Micro-View

Figure 3.3: PPM GUI of Micro-View
Fig. 3.4 shows the flow chart of the process used to compute the orientation using this sensors configuration. The gyroscopic angular rates \((\omega_x, \omega_y, \omega_z)\) will provide a rapid dynamic response with high resolution to compute the uncorrected orientation of the system. The 3-axis accelerometer and 3-axis magnetometer (see dotted arrow in Fig. 3.4) embedded in the same unit are used to stabilize the orientation of the UAV to the earth’s gravitational and magnetic fields, thus eliminating the potential unbounded accumulation of gyroscopic drift errors. An extended Kalman filter is used to discard the measurement errors and to provide the orientation prediction as close to the real orientation as possible.

Experimental data shows that the yaw value will drift a few degrees per minute. Although this drift is too slow to notice while it is happening, but the accumulated yaw error may eventually become significant in long run. Thus, a headlock gyroscope, Telebee GR302-AD is installed to correct the yaw angle of the UAV.
directly. When error occurs in the yaw angle of the UAV, Telebee GR302-AD controls the rotors speed directly such that the heading of the UAV will be corrected. Fig. 3.5 shows the block diagram of the installation of Telebee GR302-AD headlock gyroscope.

3.3 Position Measurements

3.3.1 Height

The MNAV100CA unit includes a 3-axis linear accelerometer to detect acceleration at $x$, $y$ and $z$-axis. However, it does not provide accurate acceleration measurements due to the sensor limits. In addition, gravitational force affects all these axes when the IMU is not perfectly level. In real application, the acceleration measurements are noisy due to disturbances. It is impossible to obtain the actual position of the UAV by integrating the acceleration measurements.
An experiment is conducted to test the accuracy of the position measurements using the accelerometers in MNAV100CA. The IMU is maintained at a constant height for 20 seconds and measurement of the $z$-axis acceleration is obtained. After subtracting the gravitational offset, the $z$-axis acceleration is plotted as shown in Fig. 3.6.

By integrating the $z$-axis acceleration twice, the position of the UAV is calculated (Fig. 3.7). The actual height in this experiment was held constant, however, as observed in Fig. 3.7, the calculated height is drifting away from the set point due to the noise in accelerometer measurements. A total drifting of 25cm away from set point is obtained in 20 second time.

It can be concluded that the MNAV100CA alone cannot provide accurate position measurements. Therefore, ultrasonic sonar sensor is required to obtain
Figure 3.7: Position calculated by integrate measured acceleration twice

reasonable measurements on the z-axis (height) position of the UAV. A simple
low pass filter is also implemented to eliminate the sensor noise as follows.

\[
\frac{Y(s)}{U(s)} = \frac{1}{\tau s + 1} \tag{3.1}
\]

where the cut off frequency, \( \omega_C = \frac{1}{\tau} \) is chosen to be 5Hz to eliminate noise with
frequency 5Hz or higher.

\[
Y(s)(0.2s + 1) = U(s) \tag{3.2}
\]

Taking inverse Laplace Transform,

\[
0.2y'(t) + y(t) = u(t) \tag{3.3}
\]
Discretize it with sampling interval, $h = 0.02$,

\[ 0.2 \frac{y(t) - y(t - 1)}{0.02} + y(t) = u(t) \]  
(3.4)

\[ 11y(t) - 10y(t - 1) = u(t) \]  
(3.5)

\[ y(t) = 0.9y(t - 1) + 0.1u(t) \]  
(3.6)

Beside the low pass filter, a compensator is implemented to overcome the measurement error caused by the orientation of the UAV. As shown in Fig. 3.8 when the UAV is tilted, ultrasonic sonar sensor only managed to detect the distance of the ground perpendicular to the base of the UAV. In this case, the detected height is larger than the actual height of the UAV. The angles that affect the
height measurement are pitch (θ) and roll (ϕ) angles. According to Fig. 3.9 the height of the UAV can be compensated using the following calculations.

Supposed $h$ is the actual height and $h_{pr}$ is the measured height, with pitch deviation, $\theta = \angle BAC$ and roll deviation, $\phi = \angle DAB$. Thus,

$$\tan \theta = \frac{BC}{h} \quad (3.7)$$

$$\tan \phi = \frac{BD}{h} \quad (3.8)$$

\textbf{Figure 3.9:} Retrieving actual height
Using Pythagoras theorem,

\[(h_{pr})^2 = h^2 + BE^2 \] (3.9)

\[= h^2 + BC^2 + BD^2 \] (3.10)

Thus,

\[(h_{pr})^2 = h^2 + h^2 \tan^2 \theta + h^2 \tan^2 \phi \] (3.11)

\[= h^2(1 + \tan^2 \theta + \tan^2 \phi) \] (3.12)

Actual height,

\[h = \frac{h_{pr}}{\sqrt{1 + \tan^2 \theta + \tan^2 \phi}} \] (3.13)

After the implementation of the low pass filter together with the compensator, the integrated IMU-sonar system is placed at the same position as the previous experiment. The graph of the height obtained is plotted as shown in Fig. 3.10.

As observed, it shown a much better performance comparing with the result obtained using MNAV100CA alone. The general trend of graph is maintained at constant at all time using the sonar sensor. Hence, with the introduction of ultrasonic sonar sensor to the IMU, it is able to obtain measurements of height at a much higher accuracy.
3.3.2 Lateral Position

Vision sensor\textsuperscript{1} provides measurement on the lateral position of the UAV with reference to the colored track. In order to improve the update rates, a *Fast Path Detection* algorithm is proposed.\textsuperscript{1} on top of that, three different color detection methods are used to improve the accuracy of the sensor.

After vision processing is done, the coordinates of the mid points of the colored track at the upper portion and lower portion will be sent to the onboard avionics controller. Once the coordinates of these two points are obtained, the heading angle and the distance to the colored track can be calculated as follows.

Two frames are assigned to the system (Fig. 3.11). Frame A represents the

\textsuperscript{1}Vision processing is done by another FYP student, Ong Jun Jie. Hence only brief summary is given here. Please refer to Ong Jun Jie’s thesis for detail implementation.
original frame of the UAV with \( x_A \)-axis pointed to the right and \( y_A \)-axis pointed to the front of the UAV. For simplicity, origin of frame \( B \) are assigned to be coincide with the origin of frame \( A \), with \( y_B \)-axis parallel with the colored track.

Using these frame assignments, heading angle,

\[
\psi = \text{atan}2(Ax_1 - Ax_2, Ay_1 - Ay_2)
\]

where positive angle corresponds to turning right and negative angle corresponds to turning left.

Coordinates of midpoint \( M \) in frame \( B \),

\[
BM = BR_A \times M = AR_B^{-1} \times M \equiv \begin{pmatrix} Bx \\ By \end{pmatrix}
\]

where midpoint

\[
M = \begin{pmatrix} A_{x1} + A_{x2} \\ A_{y1} + A_{y2} \end{pmatrix} / 2
\]

and

\[
AR_B = \begin{pmatrix} \cos \psi & \sin \psi \\ -\sin \psi & \cos \psi \end{pmatrix}
\]

is the clockwise rotational matrix from frame \( A \) to frame \( B \).

Refering to Fig. 3.11, the distance to the colored tracks, \( d \) is the \( x \)-coordinate of point \( M \) in frame \( B \), in this case, \( d = Bx \). Similar to the height measurement, compensator is implemented to reduce the error in the measurement. For lateral position measurement, the position of the colored track is heavily affected by the roll angle (\( \phi \)) tilt, as shown in Fig. 3.12. In the case where the roll angle of the
Figure 3.11: **Up:** One of the pictures captured by the camera mounted below the UAV. **Down:** The assigned frames for easy calculation of heading angle and lateral position.
UAV is zero, i.e. $\phi = 0$, the lateral position measurement is correct. However when $\phi \neq 0$, the lateral position measurement is none zero even if the UAV is directly above the colored track.

The error caused by roll tilt, $e$, can be calculated as

$$e = h \times \tan \phi$$

where $h$ is the current height of the UAV.

The actual position of the colored track is then obtained by substracting the error caused by roll tilt, $e$, by the measured lateral position. By using this method, the heading angle and lateral position from the colored track are accurately calculated.
3.4 Accuracy Tests

3.4.1 Yaw, Pitch and Roll Measurements

A simple experiment is carried out to verify the yaw, pitch and roll measurements by the IMU. It is done by manually tilting the UAV in oscillation motion with respect to the three axes. The Euler angles measurements from the IMU are plotted in Fig. 3.13. It reflects that the measurement result are generally acceptable since there is no unexpected changes between any consecutive data point on the curve.

3.4.2 Height Measurement

A simple experiment is carried out to verify the height measurement by the ultrasonic sonar sensor. It is done by manually oscillating the UAV position with respect to the ground. Upper part of Fig. 3.14 reflects that the measurement result are generally accurate since there is no abrupt changes between any consecutive data point on the curve.

3.4.3 Lateral Position Measurement based on Vision

A similar experiment is done to verify the accuracy of vision based positioning. The UAV is manually oscillate laterally above the colored track, and the vision based calculated distance from the colored track is plotted as shown in lower part of Fig. 3.14. The result generally shows that the real time lateral position
Figure 3.13: Euler angles of the UAV under oscillation motions at three different axes. Calculated by our image processing algorithm is accurate without any unbearable noise. Note that the minor edges on the smooth sinusoidal curve are due to the failure of path detection at that instants. The previous stored value of lateral distance will be used when the path is not detected to preserve the stability of the UAV.
Figure 3.14: 

**Up:** Height measured by the sonar range finder under oscillation motion.

**Down:** Lateral position calculated by the vision algorithms under oscillation motion.
Chapter 4

Control Methodologies

Once the sensor measurements can be obtained accurately, an avionics control system can be implemented. The mini-UAV is designed to fly in a near-hovering condition with minimal pitch and roll deviation during the navigation. Thus the yaw, pitch and roll angle of the UAV can be assumed to be decoupled to simplify the control tasks \[9\]. These three angles will be control independently by individual controllers. Table 4.1 and Fig. 4.1 shows the sensor parameters that are used to control the UAV. These sensor measurements are assumed to be accurately measured.

In this chapter, PID controller is proposed to be implemented to control the orientation and position of the mini-UAV —KingLion. Individual controllers will also be discussed in detail including the values of each control parameters. Lastly, the control algorithms is translated to a software to be executed in the onboard processor.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\theta$</td>
<td>Pitch angle</td>
</tr>
<tr>
<td>$\phi$</td>
<td>Roll angle</td>
</tr>
<tr>
<td>$\psi$</td>
<td>Yaw angle</td>
</tr>
<tr>
<td>$x$</td>
<td>$x$-position</td>
</tr>
<tr>
<td>$y$</td>
<td>$y$-position</td>
</tr>
<tr>
<td>$z$</td>
<td>$z$-position</td>
</tr>
</tbody>
</table>

**Table 4.1:** Control parameters for KingLion

**Figure 4.1:** Assignment of parameters to the UAV’s frame
4.1 PID Controller

A proportional–integral–derivative controller (PID controller) is a generic closed-loop controller widely used in industrial control systems. In a closed-loop control system, the error defines as the difference between a measured process variable and a desired reference point of that variable. The major working principle of this PID controller is to attempt to minimize the error by adjusting the process control inputs. Usually, the PID parameters must be tuned according to the nature of the system.

The PID control algorithm involves three separate parameters — the proportional, the integral and the derivative parameters. In general, the proportional value determines the reaction to the current error, the integral value determines the reaction based on the summation of previous errors, and the derivative value determines the reaction based on the rate at which the error has been changing. The weighted summation of these three actions (Fig. 4.2) is used to control the process such that the system will respond in a desire way.

4.1.1 Transfer Function

Transfer function of the PID controllers can be obtained rather straight forward based on the block diagram shows in Fig. 4.2 From the block diagram, the control signal output,

\[ u(t) = K_p e(t) + K_i \int e(\tau)d\tau + K_d \frac{de(t)}{dt} \]  

(4.1)
Taking Laplace Transform,

\[ U(s) = K_p E(s) + \frac{K_i}{s} E(s) + K_d s E(s) \]  \hspace{1cm} (4.2)

Transfer function,

\[ G_c(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s \]  \hspace{1cm} (4.3)

\[ = \frac{K_d s^2 + K_p s + K_i}{s} \]  \hspace{1cm} (4.4)

### 4.1.2 Performances

Specific process performance can be achieved by fine tuning the three constants \( -K_p, K_i, \) and \( K_d \). The performance of the controller can be described in terms of the overshoots, steady state errors, rise time, settling time, and the degree of system oscillation.
The flight control algorithm of KingLion comprising three portions: the inner loop control, outer loop control as well as path planning. Each feedback loop will be controlled by a specified PID controller for each control parameter. IMU sensors will provide feedback for the inner loop controller to control the stability of the UAV, such as the pitch and roll angles. Vision measurements provide feedback to the outer loop controller, which will control the heading angle and the position of the UAV. Beside the outer loop controller, path planning algorithm will generate position set point to drive the UAV. The general view of the control system of the UAV are summarized in the block diagram shown in Fig. 4.3.

Some channel of KingLion may require using only one or two modes to provide the appropriate system control. This can be achieved by setting the gain of
<table>
<thead>
<tr>
<th>Increase</th>
<th>Rise Time</th>
<th>Overshoot</th>
<th>Settling Time</th>
<th>Steady-state error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_p$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Small Change</td>
<td>Decrease</td>
</tr>
<tr>
<td>$K_i$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Increase</td>
<td>Eliminate</td>
</tr>
<tr>
<td>$K_d$</td>
<td>Small Change</td>
<td>Decrease</td>
<td>Decrease</td>
<td>Small Change</td>
</tr>
</tbody>
</table>

Table 4.2: Characteristics of P, I and D controllers

undesired control outputs to zero. In this way, they are called a PI, PD, P or I controller in the absence of the respective control actions. In general, the guidelines in Table 4.2 are used to tune the controllers of KingLion.

4.2 Low Level Controls

The low level controls defines the inner most control loops of KingLion. This level of closed-loop controls has the fastest dynamic and response in order to maintain the stability of the UAV. It consists of the control of three Euler angles — $\theta$, $\phi$, and $\psi$.

4.2.1 Pitch ($\theta$) and Roll ($\phi$) Control

Pitch and roll angles of KingLion are controlled by a same proportional controller as shown in Fig. 4.4. For any co-axial helicopter, the stabilizer bar acts as mechanical passive controller for both pitch and roll angles. Thus it is sufficient to control these two angles using a simple proportional controller with small effort.
After fine tuning and testing with actual flight tests, the proportional gain is chosen to be

\[ K_p = 0.013 \]

In order to achieve desired stability, the update rate of the control loops for pitch and roll angles control is 50 Hz.

4.2.2 Yaw (\(\psi\)) Control

For yaw angle control, as mentioned previously, by using IMU alone will results
in the drifting of heading angle in long run. A solution to it is to use an analog headlock gyroscope to maintain the heading angle of the UAV. This headlock gyroscopes incorporate a built in PID controller as shown in Fig. 4.5.

### 4.3 High Level Controls

High level controls take place as the outer loop controllers for a control system. Generally it has slower dynamic compare to the inner loop controllers, for it will work well only if the inner loop control outputs are stable. For KingLion's controller design, the high level controls are the control of the lateral position of the UAV to the colored track, the heading angle setpoint, and the height of the UAV with reference to the ground.

#### 4.3.1 Lateral Position (y) Control

![Figure 4.6: Lateral position control](image.png)

In lateral position control, a single PID controller is used to provide control.
signal for pitch ($\theta$) and roll ($\phi$) angle setpoint. Vision processing algorithms are assigned to measure and calculate the current lateral position of the UAV to the colored track and provide the feedback to the PID controller. The PID control parameters are chosen as follows:

\[
K_p = 0.4 \\
K_i = 0.0033 \\
K_d = 0.12
\]

In order to work well with the inner loop controller, the dynamic of this controller is set as three times as slow as the inner loop controller, i.e. 17 Hz.

### 4.3.2 Heading Angle ($\psi_{ref}$) Control

![Diagram of Heading Angle Control](image)

**Figure 4.7:** Heading angle control

Beside having a stable yaw control, KingLion is required to control its heading
direction parallel to the direction of the colored track, as a mission to navigate following the track. As mentioned in the previous chapter, vision processing algorithms are able to provide a heading angle feedback to the controller. Manual control of the UAV shows that the heading of the UAV is too sensitive to the rudder signal. Thus, in order to perform smooth turning, the control gain is lowered. Derivative gain is insignificantly small in the case, only PI controller is used. The PI control parameters for this heading control are tuned based on flight tests. The final control parameters are

\[
K_p = 0.001 \\
K_i = 0.00033
\]

4.3.3 Height (z) Control

A PID controller is used to control the height (z-position) of the UAV. Ultrasonic sonar sensor provides accurate height measurements to the controller in
the rate of 25 Hz due to sensor limit. The controller is tested and fine tuned, the parameters chosen are

\[
K_p = 0.085 \\
K_i = 0.00283 \\
K_d = 0.04675
\]

4.4 Path Planning

A simple path planning algorithm is implemented on top of the low and high level controls. Instead of using the tedious process of automaton learning, an off-line path planning algorithm is developed to improve the accuracy and efficiency of the UAV.

KingLion is designed to navigate following a colored track that is placed on the ground. Thus, external control of heading angle is not needed for this UAV. Height of the UAV and the forward speed of the UAV will be controlled by the path planning algorithm.

Due to inaccurate speed estimation, time-based path planning algorithm does not guarantee a good response. Therefore, an orientation-plus-time-based path planning algorithm is implemented. An event function is created in order to achieve the goal. The whole circuit of flight consists of multiple events. Each event will be triggered when the yaw angle of the UAV met certain requirements for two seconds continuously. Each event consists of the target height and pitch.
angle for the UAV to move forward or stop moving. Table 4.3 shows the possible requirements to trigger the event while Table 4.4 shows the possible control for each event.

<table>
<thead>
<tr>
<th>Command</th>
<th>Requirement to trigger event</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT</td>
<td>Made a &gt;60 degree left turn which last for two seconds continously</td>
</tr>
<tr>
<td>RIGHT</td>
<td>Made a &gt;60 degree right turn which last for two seconds continously</td>
</tr>
<tr>
<td>STRAIGHT</td>
<td>Heading deviation &lt;30 degree which last for two seconds continously</td>
</tr>
</tbody>
</table>

Table 4.3: Requirements to trigger an event

<table>
<thead>
<tr>
<th>Control</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEIGHT</td>
<td>Height can be arbitrary chosen from 0 to 6 meters</td>
</tr>
<tr>
<td>PITCH</td>
<td>Only two modes are available.</td>
</tr>
<tr>
<td></td>
<td>FORWARD will set the pitch target to 3 degree</td>
</tr>
<tr>
<td></td>
<td>STOP will set the pitch target to 0 degree</td>
</tr>
</tbody>
</table>

Table 4.4: Possible control in an event
4.5 Software Implementation

Once the control law has been implemented, it has to be translated into software language to run in the onboard processor. An open source autopilot software—MicroGear is used as the platform to develop the KingLion’s control software. MicroGear is an aerial robotics application that is robust, highly capable, highly configurable, and interfaceable. Along with high levels of capability, MicroGear focuses on robustness, high performance, and minimal memory and disk footprints, which is an obvious advantage over the other autopilot softwares.

MicroGear comes only with the low level control algorithm. Major modification of MicroGear is done by implementing the outer loop controllers and the path planning algorithms. Besides, drivers for analog-to-digital port and serial ports are written to interface with the ultrasonic sonar sensor and the vision processor. Data obtained from the sonar sensor and vision processor, together with the data streams from the IMU are integrated into MicroGear to realize the autopilot system of KingLion.

The customized software—Extended MicroGear uses polling method to obtain data from individual components. Fig. 4.9 shows the sequence of polling of the Extended MicroGear algorithm. Each loop of the main code is running precisely at 50 Hz rate. The code will access data received from the IMU at every loop at 50 Hz, access data received from ultrasonic sonar sensor at every two loops, and access data received from the vision processor at every three loops. As for
path planning *event*, it will only be accessible when it is triggered as discussed in previous section.

For the ease of tuning the PID controllers, an XML file in the MicroGear package is modified to include the parameters of individual PID controllers. When the Extended MicroGear program is starting up, it will read in the parameters from the XML file and assign them to each individual controllers. Thus, it saves the effort of re-compiling of the whole program by simply modifying the XML file. Refer to Appendix A for the layout of the modified XML file.
Chapter 5

Flight Tests and Results

Flight tests are carried out in the indoor environment to verify the feasibility of the controllers. In this chapter, a flight test result will be shown. Before the flight test, the arrangement of the obstacles and colored track, together with the planned path envelope of the UAV will be pre-determined.

5.1 Ground Set Up

Setting up of the colored track and obstacles are done and is illustrated in Fig. 5.1. The total distance of the whole flight is 42 meters. Beside the hand launching area and the landing area, there are a total of two obstacles on the colored track.

The first obstacle is a doorway of height 2.5 meters and width 2 meters situated at the first stretch of colored track. The UAV is required to navigate through the doorway without crashing. The second obstacle is a cardboard of 0.9 × 1.2
Figure 5.1: The set up of the colored track and obstacles
meters dimension. Beside having a huge symbol printed on it, small objects such as paper cutter, measuring tape and a small box are placed on it. The mission for this task is to identify the objects placed on the cardboard by looking at the video feedback of the UAV from the ground station.

Finally, a landing area of $3 \times 4$ meters is placed at the end of the colored track for automatic landing.

5.2 Missions

In general, the colored track will lead the UAV to complete the whole circuit. Throughout the process, the UAV is required to perform the following tasks:

1. **Launching**: The UAV will be launched (released) from hands at the Hand Launching Area after starting up. The initial height of the UAV is set as 1.5 meters above the track.

2. **Enter Doorway**: The UAV is required to fly through a doorway placed 10 meters after the Hand Launching Area.

3. **Object Identifying**: The UAV is required to identify objects placed on the $0.9 \times 1.2$ meters cardboard. In this section, the UAV is required to navigate at the height of 2.5 meters above the track.

4. **Height Recovering**: At the third stretch of colored track, a step change to the height reference of the UAV from 2.5 meters back to the initial starting
5. Automatic Landing: After the last corner of the circuit, the UAV is required to land autonomously at the Landing Area with the dimension of $3 \times 4$ meters.

5.3 Results

The Euler angles and the height of the UAV during the flight test are logged. Fig. 5.2, 5.3, and 5.4 shows the set of Euler angles of the UAV for the flight test, whereas Fig. 5.5 shows the height of the UAV plotting together with the reference points for the same flight test.

In this flight test, the UAV is released at $t = 10$ seconds. In Fig. 5.5, the red
Figure 5.3: Roll angle of the UAV

Figure 5.4: Yaw angle of the UAV
Figure 5.5: Height of the UAV

dotted line corresponds to the height set point of the UAV, while the blue solid line shows the actual height of the UAV at that instant. Note that (Fig. 5.4) the UAV made a right turn at $t = 44$ seconds and at $t = 67$ seconds, while the final left turn is at $t = 115$ seconds. Based on the plotted graphs, the performance of the UAV is acceptable. Fig. 5.6 to 5.12 show the pictures taken during the flight test while the camera view seen from the ground station for object identifying task is shown in Fig. 5.13. The complete flight test video can be viewed at

http://www.youtube.com/watch?v=fLRWcgA1ffg
Figure 5.6: The UAV is released from hands

Figure 5.7: The UAV passing through the doorway
Figure 5.8: KingLion made a smooth right turn at the first corner

Figure 5.9: The UAV is raised to the height of 2.5 meters right above the symbol
Figure 5.10: KingLion is lowered to 1.5 meters height after the second turn

Figure 5.11: The only left turn of the circuit
Figure 5.12: The UAV preparing to land at the landing area

Figure 5.13: The camera view seen from the ground station for object identifying
5.4 Awards

With this mini-UAV we developed, our team took part in the Singapore Amazing Flying Machine Competition (SAFMC) 2010 organized by DSO National Laboratories and Science Centre Singapore. KingLion has achieved the following awards in this competition:

1. Best Performance Award - Gold Medal
2. Championship Award - Silver Medal
3. Theory of Flight Award - Silver Medal
4. Most Creative Award - Bronze Medal
Figure 5.14: Awards won in Singapore Amazing Flying Machine Competition 2010
Chapter 6

Future Works

Although KingLion is able to perform simple indoor navigation such as following the colored track and height control, there are still areas to be improved. Some possible future improvements are given in this chapter.

6.1 Sensors Improvement

The current component used to detect the height of the UAV — ultrasonic sonar sensor is unable to give a precise reading depending on the material of the ground surface. For instance, if KingLion is to navigate above a carpet floor, the carpet might absorb part of the ultrasonic wave emitted by the sensor, thus results in inaccuracy in height measurement. A laser range finder can be used to replace the ultrasonic sonar sensor. However, after extensive research on the industrial product of laser range finder, the lightest good performance range finder weighs
50 grams. Thus payload issue need to be reconsidered.

Beside height detection, speed control of KingLion is also not satisfying. Currently there is no device in KingLion to detect the forward speed directly, causing inaccuracy in speed control. One possible solution is to install a speed detection device onboard. However this would bring up the payload and cost issues again.

6.2 Obstacles Detection

Currently the mini-UAV only able to detect terrain changes on the ground. Thus it is unable to detect obstacles on its way, and is therefore prone to collision in complex environment. A possible improvement is to install a few sonar or infrared sensors at the side of the UAV. Alternately, a laser range scanner could be used.

6.3 Vision Processing Algorithm

In the current design of KingLion, to perform a high speed onboard vision processing, the Fast Path Detection method is used. However, this method might not be robust enough since the large part of the image is ignored. It is a trade off problem between having fast detection and having a more robust detection. It is possible to have a more robust detection while maintaining the processing speed in the future when new developments in embedded technology and image processing algorithms occurs.
6.4 Avionic Control Algorithm

The current avionic control algorithms mainly adopt PID schemes as discussed in the previous chapters. To implement PID controllers, the helicopter dynamics are assumed to be linear and decoupled. However, these assumptions are only valid at near-hovering conditions \[9\]. Thus, for aggressive maneuvers, the PID controllers are unable to control the UAV. A possible solution in the future is to implement more advanced control schemes such as H-infinity control and Composite Nonlinear Feedback (CNF) control methods developed by the NUS UAV Research Team \[8\].
Chapter 7

Conclusion

This thesis described the design and development of a universal controller for unmanned vehicles. A rotorcraft —ESky Big Lama Co-Axial helicopter is chosen as the development platform. A simple avionics control system is implemented, which include a Gumstix Verdex Pro microprocessor, a Gumstix Overo Fire microprocessor, an inertia measurement unit MNAV100CA as the primary sensor, an ultrasonic sonar sensor MaxSonar EZ4, a pair of XBee wireless modules, and a web based camera.

The algorithms run on the avionics control system are discussed in detail. Such mini-UAV is able to perform basic tasks such as indoor navigation following colored track. Also, this mini-UAV can serve as a test platform for advance control technologies in the future.

Finally, possible improvement of this mini-UAV is also discussed. Additional and improved sensors can be implemented to improve the performance and ro-
bustness of the controller. Besides, more advanced and robust control algorithm such as non-linear control can be designed to the mini-UAV to perform more complicated tasks.
Appendices

Appendix A

Modified XML file for the PID controllers

```xml
<PropertyList>

<pid-controller>
  <name>Vision distance</name>
  <debug>false</debug>
  <enable>
    <prop>/autopilot/locks/roll</prop>
    <value>dg-roll-hold</value>
  </enable>
  <input>
    <prop>/vision/distance</prop>
  </input>
  <reference>
    <value>0.0</value>
  </reference>
  <output>
    <prop>/autopilot/settings/roll-deg</prop>
  </output>

<config>
  <Kp>-0.4</Kp> <!-- proportional gain -->
  <beta>1.0</beta> <!-- input value weighing factor -->
  <alpha>0.1</alpha> <!-- low pass filter weighing factor -->
  <gamma>0.0</gamma> <!-- input value weighing factor for -->
  <Ti>120.0</Ti>  <!-- unfiltered derivative error -->
  <Td>0.3</Td>    <!-- integrator time -->
  <Td>0.3</Td>    <!-- derivator time -->
</config>
```
<pid-controller>
  <name>Vision heading</name>
  <debug>false</debug>
  <enable>
    <prop>/autopilot/locks/roll</prop>
    <value>dg-roll-hold</value>
  </enable>
  <input>
    <prop>/vision/heading</prop>
  </input>
  <reference>
    <value>0.0</value>
  </reference>
  <output>
    <prop>/controls/flight/rudder</prop>
  </output>
  <config>
    <Kp>-0.001</Kp> <!-- proportional gain -->
    <beta>1.0</beta>   <!-- input value weighing factor -->
    <alpha>0.1</alpha> <!-- low pass filter weighing factor -->
    <gamma>0.0</gamma> <!-- input value weighing factor for -->
    <!-- unfiltered derivative error -->
    <Ti>30.0</Ti>     <!-- integrator time -->
    <Td>0.0</Td>     <!-- derivator time -->
    <u_min>-5</u_min> <!-- minimum output clamp -->
    <u_max>5</u_max>  <!-- maximum output clamp -->
  </config>
</pid-controller>

<pid-controller>
  <name>Roll angle hold</name>
  <debug>false</debug>
  <enable>
    <prop>/autopilot/locks/roll</prop>
    <value>dg-roll-hold</value>
  </enable>
  <input>
    <prop>/orientation/roll-deg</prop>
  </input>
</pid-controller>
</input>
<reference>
  <prop>/autopilot/settings/roll-deg</prop>
</reference>
<output>
  <prop>/controls/flight/aileron</prop>
</output>
<config>
  <Kp>-0.013</Kp> <!-- proportional gain -->
  <beta>1.0</beta> <!-- input value weighing factor -->
  <alpha>0.1</alpha> <!-- low pass filter weighing factor -->
  <gamma>0.0</gamma> <!-- input value weighing factor for -->
  <u_min>-0.15</u_min> <!-- unfiltered derivative error -->
  <Ti>0.0</Ti> <!-- integrator time -->
  <Td>0.0</Td> <!-- derivator time -->
  <u_max>0.15</u_max> <!-- minimum output clamp -->
</config>
</pid-controller>

<pid-controller>
  <!-- Simple pitch angle hold -->
  <name>Pitch Hold</name>
  <debug>false</debug>
  <enable>
    <prop>/autopilot/locks/altitude</prop>
    <value>pitch-leveler</value>
  </enable>
  <input>
    <prop>/orientation/pitch-deg</prop>
  </input>
  <reference>
    <prop>/autopilot/settings/pitch-deg</prop>
  </reference>
  <output>
    <prop>/controls/flight/elevator</prop>
  </output>
  <config>
    <Kp>-0.013</Kp> <!-- proportional gain -->
    <beta>1.0</beta> <!-- input value weighing factor -->
    <alpha>0.1</alpha> <!-- low pass filter weighing factor -->
    <gamma>0.0</gamma> <!-- input value weighing factor for -->
    <Ti>0.0</Ti> <!-- integrator time -->
  </config>
</pid-controller>
<td>0.0</td>
<td>-0.25</td>
<td>0.25</td>
</config>
</pid-controller>

</!-- pid-controller>
<name>Heading Hold</name>
<debug>false</debug>
<enable>
  <prop>/autopilot/locks/heading</prop>
  <value>dg-heading-hold</value>
</enable>
<input>
  <prop>/orientation/heading-deg</prop>
</input>
<reference>
<prop>/autopilot/settings/yaw-deg</prop>
</reference>
<output>
  <prop>/controls/flight/rudder</prop>
</output>
<config>
 <Kp>0.00035</Kp>
 <beta>1.0</beta>
 <alpha>0.1</alpha>
 <gamma>0.0</gamma>
 <Ti>0.0</Ti>
 <Td>0.01</Td>
 <u_min>-0.1</u_min>
 <u_max>0.1</u_max>
</config>
</pid-controller -->

<pid-controller>
<name>Height Hold</name>
<debug>false</debug>
<enable>
  <prop>/autopilot/locks/height</prop>
  <value>dg-height-hold</value>
</enable>
<input>
  <prop>/position/height-m</prop>
</input>
<config>
 <Kp>0.00035</Kp>
 <beta>1.0</beta>
 <alpha>0.1</alpha>
 <gamma>0.0</gamma>
 <Ti>0.0</Ti>
 <Td>0.01</Td>
 <u_min>-0.1</u_min>
 <u_max>0.1</u_max>
</config>
</pid-controller>
</input>
<reference>
<prop>/autopilot/settings/target-height</prop>
</reference>
<output>
<prop>/engines/engine[0]/throttle</prop>
</output>
<config>
<Kp>0.085</Kp>
<beta>1.0</beta>
<alpha>0.1</alpha>
<gamma>0.0</gamma>
<Ti>30</Ti>
<Td>0.55</Td>
<u_min>-0.3</u_min>
<u_max>0.3</u_max>
</config>
</pid-controller>
</PropertyList>
Bibliography


