Deployment issues of unmanned aerial vehicles in an urban environment

Goh, Boon Keat

2009

Goh, B. K. (2009). Deployment issues of unmanned aerial vehicles in an urban environment. Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/47223>

<https://doi.org/10.32657/10356/47223>

Nanyang Technological University

Downloaded on 26 Apr 2023 18:09:30 SGT

DEPLOYMENT ISSUES OF UNMANNED AERIAL VEHICLES IN AN URBAN ENVIRONMENT

GOH BOON KEAT

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Master of Engineering

2009

Acknowledgement

The author is glad to have completed this thesis for his Master of Engineering programme. He owes his highest gratitude to his supervisors, Associate Professor Seet Gim Lee, Gerald and Associate Professor Sim Siang Kok. Their constructive suggestions and advice have assisted the author in keeping the project on-track. Their patience and dedication in guiding the author throughout the learning journey is very much appreciated.

The author would also like to grab this opportunity to thank Dr. Ong Kai Wei, the Research Fellow of the team, for always sharing his experiences in the field and providing the author with many suggestions. The author also owes a credit to the research project collaborator, Mr. Jaroslaw Hajduk, an expert in Unmanned Aerial Vehicle systems, whom offered significant assistance during the flight tests for the SUAVs.

The project would not have been conducted so smoothly without the technical supports and administrative assistances from the helpful team of technicians in Robotic Research Centre. A big thank to Mr. Lim Eng Cheng, Madam Toh Siok Kuan Agnes, Mr. You Kim San and Ms. Toh Yen Mei. It also has been the author's pleasure to be able to work hand in hand with his fellow research mates, particularly Ms. Pang Wee Ching and Mr. Wong Choon Yue, and final year student mentees. Working with them along this journey has been very fulfilling. Discussions and opinions offered have assisted the author to shape this project well.

The author would also like to convey his thankfulness to his family, and his good friend, Ms. Chan See Yeu for always being supportive. Finally, the author would like to thank those who have assisted him in a way or another during the course of the research project.

Table of Contents

Acknowledgement i

Table of Contents ii

Abstract v

List of Figures vi

List of Tables ix

List of Abbreviations x

List of Symbols xi

CHAPTER 1: INTRODUCTION..... 1

1.1 Problem Statements3

1.1.1 Challenges in the Urban Setting4

1.1.2 Limitation of COTS autopilot4

1.2 Research Objective6

1.3 Research Scopes and Approach6

1.4 Report Outline.....8

CHAPTER 2: LITERATURE REVIEW 10

2.1 Human-UAV System10

2.1.1 Robot Control Architecture.....11

2.1.2 Review on the Autopilot Technology17

2.1.3 Issue in Designing the Low-Level Controller.....19

2.2 Deployment Issues in an Urban Environment22

2.2.1 Present of Obstacles in the Navigation Path22

2.2.2 Issue with Loss of Line-of-Sight.....27

2.3 Summary of the Findings.....29

CHAPTER 3: HUMAN-SUAV SYSTEM DEVELOPMENT 31

3.1 Aerial Robotic Platforms32

3.2 Avionic Subsystem33

3.2.1 Integration of Flight Control Subsystem.....33

3.2.2	Payload Subsystem	37
3.3	Ground Control Station Subsystem	39
3.3.1	Software Architecture	40
3.3.2	Graphical User Interface	44
3.4	Chapter Summary	47
CHAPTER 4: CONTROL ARCHITECTURE.....		49
4.1	Hardware Configurations.....	49
4.2	Control Architecture on the Flight Control Subsystem	53
4.2.1	Study on the Autopilot Control Architecture.....	54
4.2.2	Incorporation of Navigational Reactive Behavior Controller.....	57
4.3	Higher Level Reactive Behaviors	59
4.3.1	Collision Prevention and Obstacle Avoidance Behavior.....	60
4.3.2	Object Tracking based on Color Identification.....	61
4.4	Chapter Summary	64
CHAPTER 5: MOTION AND GUIDANCE CONTROL.....		65
5.1	Analytical Approach	65
5.1.1	Controller Design Methodology	65
5.1.2	Formulating the Mathematical Model for Rainbow SUAV	75
5.1.3	PID Gains Determination for Rainbow SUAV.....	79
5.1.4	Effects of CG Location on the SUAV Responses	87
5.2	Experimental Approach	90
5.3	Chapter Summary	96
CHAPTER 6: EXPERIMENT AND EVALUATION.....		97
6.1	Experiment E1: Rainbow SUAV Flight Test.....	97
6.1.1	Procedure of Experiment E1	97
6.1.2	Investigating Rainbow Waypoint Navigation Capability	101
6.1.3	Investigating the Motion and Guidance Controller	104
6.1.4	Verification of the Mathematical Model.....	112
6.2	Experiment E2: ECO-8 SUAV Flight Test.....	116
6.2.1	Procedure of Experiment E2.....	116
6.2.2	Investigating ECO-8Autonomous Flight Capability	117

6.2.3	Discussion	120
6.3	Experiment E3: Reactive Behavior Flight Test	120
6.3.1	Procedures of Experiment E3	121
6.3.2	Flight Test Results of Experiment E3	123
6.3.3	Discussion	126
6.4	Experiment E4: Object Tracking Behavior Ground Test	126
6.4.1	Procedure of Experiment E4	127
6.4.2	Ground Test Results of Experiment E4	128
6.4.3	Discussion	130
6.5	Experiment E5: Testing of the Graphical User Interface.....	131
6.5.1	Procedures of Experiment E5	132
6.5.2	Results of Experiment E5	133
6.5.3	Discussion	135
CHAPTER 7: DISCUSSION AND CONCLUSION		136
7.1	Discussion	136
7.1.1	Development of a Human-SUAV System Prototype.....	136
7.1.2	Implementation of the Control Architecture.....	139
7.1.3	Automation of the Low-Level Control	140
7.1.4	Achieving Autonomous Navigation	144
7.1.5	Achieving Higher-Level Reactive Behaviors	144
7.2	Concluding Remark	146
7.3	Recommendation of Future Work	148
7.3.1	Unaddressed Issue.....	148
7.3.2	Improvement on the Current Work.....	148
Reference.....		151
Appendix		160

Abstract

There has been an increase in interest in small unmanned aerial vehicles (SUAV). This is due to their high potential for urban deployments. Navigating SUAVs in a cluttered environment can be challenging, and at times hazardous as compared to flights in open spaces. This research aims to develop a Human-SUAV system using commercially-off-the-shelf (COTS) autopilot module for urban deployment study. Studies showed that the COTS autopilot modules are capable of delivering a deliberative control architecture for waypoint navigation. However, the capability of reactive behavior is absent from these modules. Thus, the research focuses on the implementation of the deliberative control architecture and to incorporate reactive behaviors to the autopilot control architecture.

In order to accomplish a high-level task, it is mandatory to first implement a lower-level controller for the motion and guidance control. This thesis presented the analytical and the “*test, observe and tune*” approaches in determining the PID gains. The mathematical models for the SUAVs were derived to facilitate the analytical approach. The Advanced Aircraft Analysis software was employed to estimate the aerodynamic coefficients of the SUAV. The PID gains for the *heading-following* and *airspeed-hold* behaviors were successfully determined using the analytical approach. In determining the gains for the *altitude-hold* behavior, the “*test, observe and tune*” approach was attempted.

A mechanism which permits reactive behaviors to interrupt the deliberative behaviors was proposed. Based on the mechanism, the *Detection Subsystem* and the *Navigational Reactive Behavior Controller* (NRBC) were incorporated into the original control architecture to execute the reactive behaviors. To demonstrate the capability of the controller in subsuming the commands from the autopilot, two simple behaviors, i.e. the *collision prevention & obstacle avoidance* and *object-tracking* were implemented on the NRBC.

The flight test results showed that the autopilot controller using the PID gains tuned by the analytical approach is capable of controlling the SUAV to perform the *heading following* behavior. Demonstration of the “*test, observe and tune*” approach has been carried out to achieve the *altitude hold* behavior. The experimental results also showed that the controller is able to detect obstacles and prevent the SUAV from collision. This demonstrated that the *Detection Subsystem* and the NRBC have been incorporated successfully into the autopilot control architecture.

List of Figures

Figure 1-1: Research scopes6

Figure 1-2: Research testbed.....7

Figure 2-1: Schematic of control architectures 12

Figure 2-2: Control architecture proposed by Boskovic et al. [42] 13

Figure 2-3: Control architecture proposed by Narayan et al. [45]..... 15

Figure 2-4: Original path and new path computed by the deliberative path-
planning and obstacle avoidance behavior (adapted from Griffiths
et al. [46])..... 16

Figure 2-5: Commercially off-the-shelf autopilot modules 17

Figure 2-6: Approaches to obtain the PID gains of a linear controller20

Figure 2-7: Typical terrain in Singapore (from Google Earth [78])22

Figure 2-8: Type of sensors used for objects detection24

Figure 2-9: UAV in University of California (adapted from Shim et al. [81]).....25

Figure 2-10: Flight test result using MPC algorithm (adopted from Shim et al.
[81]).....25

Figure 2-11: Flying-wing airframe in BYU (adapted from Griffiths et al. [46])...26

Figure 2-12: AVATAR helicopter with vision system (adopted from Hrabar et
al. [84]).....27

Figure 3-1: The overall SUAV system architecture developed31

Figure 3-2: Hardware integrated on the Rainbow avionic subsystem and GCS....33

Figure 3-3: Sensor suite and actuation modules on the SUAV avionics
subsystem.....34

Figure 3-4: Location of the hardware integrated on the Rainbow platform35

Figure 3-5: Hardware components integrated onto the ECO-8 helicotper36

Figure 3-6: Locations of the two video cameras on the Rainbow airframe.....37

Figure 3-7: Passive and active sensors used for object detection38

Figure 3-8: Concept in replicating the RC mode to a single user interface40

Figure 3-9: Framework of the GCS software architecture.....41

Figure 3-10: Details of the UAV module framework43

Figure 3-11: Automated ground tracking system44

Figure 3-12: Layout of the GCS graphical user interface.....46

Figure 3-13: Integration of avionics subsystem to the RC airframe	47
Figure 4-1: Possible hardware architectures for the flight control subsystem.....	52
Figure 4-2: Overview of the control architecture implemented without additional microprocessor	55
Figure 4-3: Activation of sub-behaviors to achieve waypoint-navigation behavior.....	56
Figure 4-4: Mechanism to append <i>UserCode</i>	58
Figure 4-5: Collision prevention behavior implemented on the ECO-8 SUAV	61
Figure 4-6: Axis-frames attached on the CMUCam3 and the SUAV	62
Figure 4-7: <i>Hover and track</i> behavior implemented on the ECO-8 SUAV	63
Figure 5-1: Methodology to design the <i>motion and guidance</i> controller for autopilot	65
Figure 5-2: SUAV deployment stages	66
Figure 5-3: Inputs and outputs from the AAA software module	70
Figure 5-4: Location of the control surfaces and the effects on the SUAV motion	72
Figure 5-5: Block diagram of the <i>airspeed-hold</i> and <i>heading-following</i> behaviors	73
Figure 5-6: Location of the center of gravity for the Rainbow SUAV	75
Figure 5-7: Geometry of the Rainbow horizontal tail drawn in AAA.....	77
Figure 5-8: Cascaded control loop for the <i>heading-following behavior</i>	81
Figure 5-9: System performance comparison among the P, PI and PID controllers	82
Figure 5-10: Unit step response for the <i>roll-hold</i> feedback loop.....	84
Figure 5-11: Unit step response for the heading hold behavior.....	85
Figure 5-12: Movement of the CG locatoin.....	87
Figure 5-13: Open loop step input response	88
Figure 5-14: Closed-loop response for the elevator from pitch feedback loop	89
Figure 5-15: <i>Throttle from altitude</i> feedback control loop	90
Figure 5-16: Flight tests results for Test #1 and Test #2	92
Figure 5-17: Flight tests results for Test #3 and Test #4	93
Figure 6-1: The Rainbow was hand-launched to take-off	99
Figure 6-2: Navigation path taken by the Rainbow SUAV in Experiment E1	103

Figure 6-3: Directional control flight test results.....	105
Figure 6-4: Airspeed control.....	109
Figure 6-5: Matlab/Simulink model to compute Model Desired Pitch.....	110
Figure 6-6: Investigation on the discrepancy in the desired pitch computed	111
Figure 6-7: Simulated heading angle response	112
Figure 6-8: Comparison between simulation results and flight results for directional control	113
Figure 6-9: Comparison between simulation results and flight results for airspeed control.....	114
Figure 6-10: ECO-8 autonomous take-off	118
Figure 6-11: Navigation path taken by the ECO-8 SUAV in Experiment E2	119
Figure 6-12: Flying site for Experiment E3	121
Figure 6-13: Navigation path taken by the ECO-8 SUAV in Experiment E3	123
Figure 6-14: Distance reading from the laser sensor	124
Figure 6-15: Reactive behavior altitude control	125
Figure 6-16: Reactive behavior heading control.....	125
Figure 6-17: Ground test results of the object tracking using the CMUCam3	128
Figure 6-18: Images captured by the CMUCam3 vision sensor.....	128
Figure 6-19: Heading control when <i>tracking & hover</i> behavior was enabled.....	129
Figure 6-20: Camera configuration in flight test	130
Figure 6-21: Images captured by the CMUCam3 from flight test.....	131
Figure 6-22: Experiment setup for Experiment E5.....	132
Figure 6-23: GUI display after the communication was established	133
Figure 6-24: GUI display during the navigation.....	134
Figure 7-1: Corrupted video signals	137
Figure 7-2: Collaboration among expertises to develop a suitable system for urban application.....	149

List of Tables

Table 1-1: Potential applications of UAV in an urban setting 1

Table 1-2: Classification of SUAV and MAV 2

Table 2-1: UAV Modes of Control 28

Table 3-1: UAV airframes developed 32

Table 3-2: Telemetry received from the UAV 42

Table 3-3: Information required for different deployment mode 45

Table 4-1: Comparison among the three hardware configurations 53

Table 4-2: Failure detection subsystem 54

Table 4-3: Reactive behaviors implemented 59

Table 5-1: Fixed-wing SUAV moments of inertia 76

Table 5-2: Flight conditions used to estimate the aerodynamic coefficients 76

Table 5-3: Input parameters for the horizontal tail geometry 77

Table 5-4: Rainbow aerodynamic coefficients obtained from AAA 78

Table 5-5: MicroPilot autopilot behaviors in level-flight condition 79

Table 5-6: Transfer functions for the Rainbow platform in level flight
condition 80

Table 5-7: Effect of the increase in the gain on the output response 81

Table 5-8: PID gains determined using the analytical approach 86

Table 5-9: Longitudinal stability coefficients for different CG location 89

Table 5-10: PID gains used in the flight test attempts for the *throttle from
altitude* feedback control loop 91

Table 6-1: Flight mission for E1 experiment 98

Table 6-2: Flight data required for Experiment E1 100

Table 6-3: Summary of the data plotted for directional control results 104

Table 6-4: Summary of the data plotted for airspeed control results 108

Table 6-5: Control inputs and output responses from each feedback loop 115

Table 6-6: Flight mission for Experiment E2 116

Table 6-7: Flight data required for E2 experiment 117

Table 6-8: Flight mission for Experiment E3 121

Table 6-9: Flight data required for the E3 experiment 122

Table 6-10: Data required for the E4 experiment 127

List of Abbreviations

μ -Processor	-	Additional microprocessor
AGL	-	Above ground level
AMP	-	Autopilot Mission Manager
ATS	-	Automated tracking system
BYU	-	Brigham Young University
CA	-	Center of aerodynamic
CAD	-	Computer aided drawing
CCD	-	Charged-coupled device
CG	-	Center of gravity
COTS	-	Commercial off-the-shelf
DARPA	-	Defense Advanced Research Projects Agency
DOF	-	Degree of freedom
FCS	-	Flight control system
GCS	-	Ground control station
GPM	-	Geo-pointing module
GPS	-	Global positioning system
GUI	-	Graphical user interface
ISR	-	Intelligent surveillance and reconnaissance
MAV	-	Micro unmanned aerial vehicle
MFC	-	Microsoft Foundation Class
MIMO	-	Multi-input multi-output
NBC	-	Navigational behavioral controller
NBRC	-	Navigational reactive behavioral controller
OAB	-	Obstacle avoidance behavior
PTU	-	Pan-tilt unit
RC	-	Radio-controlled
RF	-	Radio frequency
SA	-	Situational awareness
SBB	-	Skill-based behavior
SCD	-	Stability and control coefficients
SISO	-	Single-input single-output
SUAV	-	Small unmanned aerial vehicle
UAV	-	Unmanned aerial vehicle

List of Symbols

<u>Symbol</u>	<u>Definition</u>	<u>Unit</u>
U	: Body-frame X -axis translational velocity	ms^{-1}
V	: Body-frame Y -axis translational velocity	ms^{-1}
W	: Body-frame Z -axis translational velocity	ms^{-1}
p	: Body-frame X -axis rotational angular velocity	rads^{-1}
q	: Body-frame Y -axis rotational angular velocity	rads^{-1}
r	: Body-frame Z -axis rotational angular velocity	rads^{-1}
θ	: Inertia-frame pitch angle	rad
ϕ	: Inertia-frame roll angle	rad
ψ	: Inertia-frame yaw angle	rad
Ψ	: Heading angle	Degree
a_{b-x}	: Body-frame x -axis translational acceleration	ms^{-2}
a_{b-y}	: Body-frame y -axis translational acceleration	ms^{-2}
a_{b-z}	: Body-frame z -axis translational acceleration	ms^{-2}
u	: Ratio of change in airspeed over nominal airspeed	Dimensionless
α	: Ratio of change in angle of attack over nominal angle of attack	Dimensionless
β	: Ratio of change in sideslip angle over nominal angle of attack	Dimensionless
C_x	: X -axis aerodynamic coefficient	Dimensionless
C_y	: Y -axis aerodynamic coefficient	Dimensionless
C_z	: Z -axis aerodynamic coefficient	Dimensionless
C_D	: Drag coefficient	Dimensionless
C_L	: Lift coefficient	Dimensionless
C_W	: Side coefficient	Dimensionless
C_l	: X -axis moment coefficient	Dimensionless
C_m	: Y -axis moment coefficient	Dimensionless
C_n	: Z -axis moment coefficient	Dimensionless
F_x	: Force exerted along the X -axis	N
F_y	: Force exerted along the Y -axis	N
F_z	: Force exerted along the Z -axis	N

List of Symbols

L	: Moment exerted about the X-axis	Nm
M	: Moment exerted about the Y-axis	Nm
N	: Moment exerted about the Z-axis	Nm
T	: Thrust force	N
b	: Wing span	m
c	: Mean aerodynamic chord	m
S	: Wing-span surface area	m ²
I_{xx}	: Moment of inertia along X-axis	kgm ²
I_{yy}	: Moment of inertia along Y-axis	kgm ²
I_{zz}	: Moment of inertia along Z-axis	kgm ²
I_{xz}	: Cross-moment of inertia between X and Z axes	kgm ²
I_{xy}	: Cross-moment of inertia between X and Y axes	kgm ²
I_{yz}	: Cross-moment of inertia between Y and Z axes	kgm ²
δ_{th}	: Throttle setting	%
δ_a	: Aileron control surface angle	rad
δ_e	: Elevator control surface angle	rad
δ_r	: Rudder control surface angle	rad
m	: SUAV mass	kg
ρ	: Air density	kgm ⁻³
g	: Acceleration due to gravity	ms ⁻²
d_{laser}	: Laser sensor reading	m
Γ_{laser}	: Laser flag for reactive behavior	Dimensionless
Γ_{cam}	: CMUCam3 flag for reactive behavior	Dimensionless

CHAPTER 1: INTRODUCTION

As highlighted by the Defense Advanced Research Projects Agency (DARPA) in United States, unmanned aerial vehicles (UAVs) are identified to be potentially deployable in ‘dirty, dull and dangerous’ missions [1]. Without a pilot onboard in a UAV deployment, adverse risks to the pilot in highly hazardous missions such as performing surveillance in battlefield or collecting environmental information from nuclear site are eliminated. Unmanned aerial vehicles (UAVs) were initially deployed by National Security agencies for surveillance, provision of ad hoc communication infrastructure and ‘active’ intervention. These were demonstrated during the Vietnam War in the 1960s and early 1970s. Most of the military roles are still very relevant to the current context. For example, in recent years, Predator and Global Hawk UAVs were seen performing intelligent reconnaissance and surveillance (ISR) in Kosovo (1999), Afghanistan (2001) and Iraq (2003) [2].

Table 1-1: Potential applications of UAV in an urban setting

Military	Civilian	Commercial	Others
<ul style="list-style-type: none">•Reconnaissance•Surveillance•Battlefield assesment•Close air support•Precision strike	<ul style="list-style-type: none">•Search & Rescue•Road traffic monitoring•Homeland security•Environmental monitoring	<ul style="list-style-type: none">•Aerial photography•Agricultural monitoring•Pipeline monitoring•Forest imaging	<ul style="list-style-type: none">•Space exploration•Communication relay•Non-profit missions

Although, UAV deployment is still very much focused on military applications, the list of potential UAV roles is diversifying. New roles have been consistently emerging and they have made inroads to other areas; commercial, civilian services and non-profit missions, etc. Within the civilian realm, UAVs such as the Aerosonde [3, 4] and the RMax helicopter [5, 6] have been deployed for weather data collection and agricultural monitoring respectively. In 2003, the Bat-III UAV [7], which is equipped with high resolution imaging system, had successfully demonstrated the capabilities for forest imaging, traffic monitoring and powerline inspection [8]. Other civilian applications include environmental monitoring (e.g. [9]) and law enforcement. Table 1-1 provides a list of potential applications for UAVs in various fields.

Auxiliary advantage that can be derived by using an UAV, is the possibility of reducing the size of the aerial vehicle significantly. The trend of development shows that small UAV (SUAV) and micro UAV (MAV) are getting more attentions from researchers and developers globally [10]. The term ‘small’ is often loosely used along with ‘micro’ by many researchers. Professor Bekey G. A. from the University of Berkeley defines MAV based on the wing span size of the fixed-wing UAV [11]. The term ‘small’ on the other hand, has not been formally defined by researchers. In this thesis, SUAV is referring to as UAV weighting less than 10kg and with a wing span not larger than 2 meter. On the other hand, MAV has a wing-span less than 0.15 meter and weight less than 100g. Table 1-2 gives the summary of the UAV classification based on the size and weight.

Table 1-2: Classification of SUAV and MAV

Class of UAV	Wing Span	Weight	Example
Small UAV (SUAV)	< 2 m	< 10 kg	<ul style="list-style-type: none">• Dragon Eye [12]• BAT-III [7]
Micro UAV (MAV)	< 0.15 m	< 100 g	<ul style="list-style-type: none">• MicroStar [13]• Black Widow [14]

The boom in the research and development of smaller UAVs are motivated by their great prospect in the urban applications [15]. Recent statistic shows that by the end of year 2008, about 50 percent of the global population are living in urban areas and by the year 2030, the percentage is estimated to rise to 60 percent [16]. It is conjectured that urbanization will shift the current existing conflicts to the urban areas. Some of these conflicts over the last two decades took place in cities such as El Salvador, Peru, Egypt and Afghanistan took place in the cities [17]. In 1992, DARPA identified urban warfare as a new type of warfare in the coming future [18]. Also, the demand for civilian usages is expected to escalate with the growth of population in urban areas. Another factor that has made the development of SUAV and MAV possible is the advancement in miniaturization technology, especially in micro-electro-mechanical systems (MEMS) [15, 19]. Cost of MEMS sensors and controller are becoming more affordable and the performance has also improved.

SUAV and MAV can fly at lower attitudes and at lower speeds, as compared to large-scale UAVs such as the Predator. Being physically small allows them to navigate through narrow paths between buildings. Collateral damage imposed by SUAV and MAV are also relatively low and this is critical to ensure that the potential hazard to the highly dense urban environment is kept at minimum if not mitigated. The ease of deployment is another added advantage to smaller UAV. For example, Eagle UAV [1] can be launched by hand and are very portable. As the deployment become easier, a smaller operating crew is required.

1.1 Problem Statements

In today's context, deployment of UAV in an urban setting is still uncommon, especially for civilian applications. High failure rate of UAVs indicates that they are still in the research and development stage. Authorities [20] and researchers have raised their concerns over the UAV's reliability. Reports and statistics indicated that the accident rate for UAVs is significantly higher than manned aircraft [1, 21, 22]. Human errors accounted for 17 percent of the total failure rate [21]. UAVs are either remotely piloted by a land-based pilot or through a ground control system (GCS). As a result, deployment of UAVs is highly subjected to human errors, especially under a highly stressed situation, for example, in a combat mission. These methods of deployment showed that current UAV is lacking in autonomy to operate on its own.

The need for involvement of the operation crews during the entire mission is still very significant. The ratio of the number of personnel to one UAV platform is still relatively high. For example, currently, a Predator UAV requires at least three personnel during deployment; a ground-pilot, an air vehicle operator, and another personnel in charge of the mission payload [23]. In another example, in a proposal of flight tests for traffic monitoring by Western Michigan University, four personnel are required for full operations [24].

This reliability issue has to be addressed before UAVs can be accepted by the authorities and public to fly in civilian airspaces due to safety concern. In urban deployments and missions, the robustness of the UAV system is even more critical.

This is because UAVs flying in urban areas face challenging issues that usually do not exist in open spaces. Two major challenges will be further elaborated in Section 1.1.1.

1.1.1 Challenges in the Urban Setting

In an urban setting, man-made infrastructures such as buildings posed much challenges to UAV. Navigating an UAV in a cluttered and unknown or partially-known environment is extremely challenging. Narrow path between buildings makes the navigation of UAV difficult. In addition, a UAV may easily collide upon obstacles which are either stationary (e.g., pedestrian bridges, trees) or moving dynamically (e.g., a flock of birds). Collision is undesirable, raising much public and private safety concerns. Thus, the potential collateral damages that can be caused by UAV must be contained.

Deploying and monitoring an aircraft is a challenge in UAV deployment, since there is a high possibility of losing the aircraft from sight. In urban deployment, having the UAV to be deployed near to the GCS site is common; however obstacles such as buildings often obstruct the sight of the UAV. Every often, this created the off-the-sight situation. The ground pilot will lose the sight of the UAV and this result in the inability to observe the orientation of the vehicle. With the deprivation of visual cues, the effectiveness of direct control and monitoring will be drastically reduced. The ground pilot will have to rely on feedbacks from the instruments onboard to fly and monitor the UAV. Temporary lapse in situational awareness (SA) often occurred during the transition from one mode of control to other modes. For example, switching from the direct-control with line-of-sight to the instrument-aided control. Short moment of situational ‘unawareness’ could result in catastrophic consequences as UAV is flying at a relative high speed.

1.1.2 Limitation of COTS autopilot

Discussion in previous subsection highlights the need of a Human-SUAV system that addresses the two major issues. Firstly, human errors contributed to low reliability of SUAV. Some of the operations, for examples, taking-off and landing

of SUAV, and payload monitoring can be automated. Relocating the tasks from the humans to the SUAV control system will increase the vehicle level of autonomy and lessen its dependence on human inputs. The number of operators (or pilots) participate in an operation may also be reduced. For instance, if an UAV is capable of navigating autonomously, one ground-pilot may be able to perform flight control and payload monitoring, instead of two persons. It is argued that, when humans' involvements in the operation are reduced, the reliability of SUAV will improve [25]. Secondly, transitions between control modes may result in loss of situational awareness, which is not desirable. The ground control station (GCS) developed for the Human-SUAV system need to be designed and customized to address this issue.

Commercially off-the-shelf (COTS) autopilot modules are widely available, thus, they can be used to facilitate and accelerate the development of the Human-SUAV system. These modules provide technical solutions for avionics subsystem, communications and also GCS. However, current COTS autopilot technology has two major drawbacks. Firstly, the highest level of behavior that could be achieved by the COTS autopilot is the waypoint navigation behavior, which is a deliberative behavior. The deliberative behavior¹ has high-level of intelligence that will decide on the navigation paths based on the *a-prior* knowledge of the SUAV's flight envelope [26, 27]. The COTS autopilot does not include the reactive behavior², which is a control approach that reacts almost instantly to external stimulus without excessive planning [28]. Without reactive behavior, the SUAV will not be able to react to obstacles confronted, which is common in an urban setting. Secondly, the GCS provided by the autopilot manufacturer are fixed in the functional context and also in the graphical user interface display. This prevents customization for urban deployment.

¹ Detail explanation on the deliberative behavior is given in Section 2.1.1.

² Detail explanation on the reactive behaviors is given in Section 2.1.1.

1.2 Research Objective

The ultimate aim of this research is to develop a Human-SUAV system for urban deployment study. The focus is on implementing the control architecture on the SUAVs using COTS autopilot modules. Study is conducted to understand and determine how higher-level reactive behavior can be incorporated onto the control architecture. The intention of this study is to facilitate the implementation of higher-level reactive behaviors (e.g. obstacle avoidance behaviors) so that the UAV has the capability to navigate autonomously in an urban setting.

1.3 Research Scopes and Approach

Figure 1-1: Research scopes

Figure 1-1 shows an overview of the research scopes for this thesis. Generally, the scopes can be discussed in two phases. Firstly, in order to provide a realistic experimental setting for the study of SUAV deployment, having a physical Human-SUAV system as the research testbed is desirable. The development involved the building of the vehicle platforms and integration of avionics subsystem. A GCS subsystem would be designed and implemented. Figure 1-2 depicts the overall view of the system architecture developed.

Figure 1-2: Research testbed

Three vehicle platforms were built using radio-controlled (RC) airframe used by hobbyist. Two of them are fixed-wing aircrafts and the third is a rotating-wing vehicle. All the platforms' avionic subsystems were developed using the MicroPilot autopilot modules, which are available commercially. Additional payloads such as vision and object detection subsystems were integrated to the vehicle platform to provide additional functionalities and to increase the capability of the UAVs. Interactions between these UAVs with the land-based pilot are communicating through the GCS subsystem, that acts as medium of information exchange. The GCS was designed and customized to allow the land-based pilot to control the UAVs in multiple modes.

With the research testbed in place, the research moved on to the second phase. In this phase, the focus was on the study of the control architecture fundamental to robotic applications. The original control architecture of the autopilot module will be studied in details. This investigation would lead to the understanding on how reactive behaviors could be incorporated and called to interrupt the original control architecture. A mechanism for incorporating higher-level reactive behaviors was proposed to modify the original control architecture.

A low-level controller³ has to be designed for the UAV to be self-stabilizing about its own axes and guiding itself to a target location. The design of the controller

³ The term low-level controller will be further explained in Chapter 2.

required a mathematical model that represents the dynamic motion of the SUAV. However, the challenge faced in formulating the mathematical model was the lack of aerodynamic data for these RC platforms. Thus, a parametric software module, Advanced Aircraft Software (AAA), would be used to estimate the aerodynamic coefficients. The controller was designed based on classical control theory with the available mathematical model. Flight tests were conducted to investigate the suitability of the PID gains determined in controlling the SUAV.

Upon successful completing the low-level control, higher-level reactive behaviors would be imbued to the SUAV control architecture. Simple higher-level reactive behaviors such as *object tracking* and *collision prevention & obstacle avoidance* were developed and implemented. These behaviors were used to demonstrate the feasibility of incorporating the reactive behaviors into the COTS autopilot control architecture. Flight test were conducted to demonstrate the high-level behavior.

1.4 Report Outline

The thesis was organized as following. The second chapter presents a review of literature of the relevant works that had been conducted by other researchers. The review studies the control architecture for the SUAV implementation. A review on the limitation of the COTS autopilot module would also be presented. The two major issues faced by SUAV in an urban deployment as presented in Section 1.1.1 would also be reviewed in this chapter.

Development of the research testbed as depicted in Figure 1-2 will be discussed in the third chapter. This chapter discusses the integration of the hardware components onto the SUAVs avionics subsystems. The GCS that adopted the concept of multi-mode control is discussed. Using the same GCS, the ground-pilot could switch from one mode to the other easily. The software architecture implemented on the GCS and the design of the graphical user interface are also detailed in Chapter 3.

Chapter 4 discusses the overview of the control architecture implemented on the SUAV avionics subsystem. This chapter studied the autopilot original control architecture and proposed a mechanism to modify the architecture for incorporating higher-level reactive behaviors. Two reactive behaviors were presented to demonstrate the mechanism proposed. The algorithms for the *collision prevention & obstacle avoidance* and *object tracking* were also provided in this chapter.

Chapter 5 presents the approaches attempted to design the low-level controller implemented on the fixed-wing SUAVs for motion and guidance control. Detailed methodology for the analytical approach employed for tuning the gains will also be discussed. This chapter also presented the experimental tuning approach. Discussion based on the flight test data and analysis in determining the PID gains will be presented as well.

In Chapter 6, experiments that had been conducted to verify the implementation were presented. Five main experiments were conducted. The first two tests demonstrated the capability of the SUAVs in performing the deliberative behavior, i.e. the *go-to waypoint* behavior. The third test was designed to show the *collision prevention & obstacle avoidance* behavior. A ground test conducted to show the capability of tracking an object is also included in the chapter. The final experiment was carried out to validate the functionalities of the GCS developed.

The final chapter summarizes the thesis by providing a discussion on the implementation and results obtained in the thesis. It was then concluded by emphasizing the major works that have been carried out. Finally, recommendations of future work will be proposed.

CHAPTER 2: LITERATURE REVIEW

In the previous chapter, the need for a Human-UAV system which addresses the challenges of urban environment deployment has been presented. The aim of this chapter is to further ascertain this need. This chapter begins with Section 2.1 by reviewing the importance of autonomy and the fundamental requirements in developing an autonomous Human-SUAV system. In this section, the robot control architecture, particularly for UAV will be reviewed. As this research is motivated by the advantage of using commercially-available products to speed up the development, the current autopilot technology and the limitation of it, will also be studied. The review affirms that these COTS autopilot are only capable of delivering deliberative behavior and it is not sufficient for urban navigation, which needs higher level reactive behaviors. In Section 2.2, the two major challenges in urban deployment will be reviewed.

2.1 Human-UAV System

In order to improve the reliability, human errors have to be significantly reduced. As discussed by Clough [29], UAV reliability can be improved by increasing the level of autonomy in an UAV system. The term UAV autonomy is defined as “*the ability of an agent (in this context, an UAV) to act efficiently without any human’s intervention*” (adapted from Braynov and Hexmoor [30]). In order for the UAV to be autonomous, it has to be intelligent, which is the ability “*to perceive dynamically changing conditions in its environment, to reasons and to interpret perceived information, and to acts appropriately to affects conditions in its environment*” (adapted from Kim et al. [31]).

When an UAV is deployed autonomously, it does not imply that throughout the operation, the UAV is thoroughly self-governing and is totally self-planning and self-control [26]. However, during the deployment, the UAV is capable of performing a given task on its own without the need of being supervised for a defined period of time [32]. Thus, when an UAV becomes more autonomous, the operator or land-based pilot will not need to continuously control the UAV. As a result, the workload of the pilot will be reduced significantly [33, 34] and studies

show that this will release the pilot cognitive resources [35], which will effectively reduce the possibilities of the pilot making mistakes. Release of cognitive resource also allows the ground personnel to perform other tasks, such as to control and monitor multiple vehicles.

Although, human's intervention reduces as the level of autonomy of the UAV increases, the role of humans in the entire deployment is still very significant, as the human focus on less hectic roles such as to continuously plan missions or to perform decision making tasks. The aim of collaboration between the human and UAV is to exploits each other's unique strength and to achieve mutual compensation of each other weaknesses [26, 36, 37]. For instance, UAV which is equipped with embedded microprocessors can synthesize sensors reading at an extremely fast rate (in term of milliseconds), thus giving them an advantage to react to a sudden danger instantaneously. A human-agent, good in decision making, may be tasked to control the camera payload and to analyze the video image received in an intelligent reconnaissance and surveillance (ISR) mission.

2.1.1 Robot Control Architecture

After establishing the need of developing an intelligent autonomous UAV platform, it will be essential to discuss on how autonomy can be implemented on the UAV system. This requires an understanding of the robot control architecture, which is a principled way of organizing a robot control system [38]. There can be identified three classes of control architecture, i.e. deliberative architecture [39, 40], reactive architecture [40] and hybrid control architecture, which is a combination of both the deliberative and reactive architectures.

The deliberative control architecture is hierarchical and it follows a top-down approach (sees Figure 2-1a). When a goal is defined, the map planner will synthesize a plan based on the knowledge of the world (or the environment) that it is working in. A sequence of actions to be executed that will lead to the successful accomplishment of the goal, will then be computed. Traditionally, the planning phase is performed offline. Offline planning is suitable for applications in

structured and known environment, for example in automotive industrial applications. However, in an unknown or with limited knowledge of the working theater, online planning is deemed to be necessary [41]. The idea of online planning is the capability to continuously sense and update the environment model, which is usually imperfect. The map planner will re-plan the plan dynamically based on the updated environmental model. The challenge of the online planning is that it has to be carried out at real time rate and, thus computationally intensive. This may be a burden to SUAV and MAV which have limited and smaller processing capabilities due to its small payload available, as compared to large UAV or manned aircraft. Furthermore, higher computational delays the response of the UAV that makes them slow in responding to instantaneous threat.

Figure 2-1: Schematic of control architectures

In contrast to the deliberative control architecture, the reactive control architecture (sees Figure 2-1b) does not involve excessive planning or reasoning. This architecture couples the perception and action tightly, which gives a fast reflective response to external stimuli. The advantage of this response is that it has a higher response rate and does not require the knowledge of the world model. Thus, the computation requirement is lower and easier to be implemented. Behavior-based control architecture (or subsumption) is a subset of the reactive control architecture, proposed by Brook [28]. *Behavior* is “a set of purposive perception-action units” (adapted from Shi [41]) and each behavior produces commands to control the robot

based on selected sensory information. Thus, in this control architecture, intelligent action is accomplished by managing and executing a set of behaviors, that is defined to achieve an overall mission. Commands to the robot are generated by each behavior by processing selected sensory information. The challenge of this behavioral architecture is to design a set of behaviors that can accomplish the mission goal.

Figure 2-2: Control architecture proposed by Boskovic et al. [42]

The discussion this far focuses on the common control architecture implemented on robotic system generally. Specifically for UAV, Boskovic et al. [42] presented an architecture for UAV navigation, as illustrated in Figure 2-2. Four layers of hierarchical architecture has been formalized, consisting of (i) the inner-loop control layer, (ii) the trajectory generation layer, (iii) the path planning layer, and (iv) the decision making layer. This control architecture will be discussed based on the example where an UAV is required to perform reconnaissance in a defined area. Assuming that the area for reconnaissance has been identified by the first control layer, a series of waypoints that form the path for navigation will be generated by the path-planning control layer. In the trajectory generation layer, a trajectory is fitted to the waypoints by computing the desired states (i.e. the airspeed, the altitude and the heading), based on the generated waypoints and also on the constraints imposed (such as aerodynamic constraints and environment constraints). The trajectory generation layer is also often termed as the navigation loop of control [43, 44]. Finally, the computed desired states will be used as the control inputs to the inner-loop (also known as the *guidance and motion* control loop) that will eventually produce the control signals that actuate the propulsion and control surfaces of the UAV.

The control architecture proposed by Boskovic et al, lacks a subsystem that deals with reactive behavior [45]. Having a reactive behavior subsystem will enhance the robustness of the UAV system to react to instantaneous threat swiftly. To address the weaknesses in the Boskovic's control architecture, Narayan et al. [45] improves the control architecture by including a sense-and-detect subsystem for static and dynamic obstacles detection. Collision free path segments will be generated based on the data collected from the detection subsystem to perform emergency collision avoidance maneuver. In this architecture, a communication module is also included where the land-based human operator can interact with the onboard virtual operator. The virtual operator will then delegate the mission objectives to the lower layer and also perform monitoring tasks to ensure that all the layers are working towards the overall goal. Figure 2-3 shows the improved control architecture proposed by Narayan et al. [45].

Figure 2-3: Control architecture proposed by Narayan et al. [45]

The control architectures discussed by both Boskovic and Narayan provide a framework towards the development of a full autonomous UAV system. Although the implementation of these control architecture is still in the development stage, parts of the control architecture framework had been demonstrated by researchers. For example, Griffiths et al. [46] and Lucas et al. [47] demonstrated it in real flight tests and also on simulation (e.g. Aryo [48]). Figure 2-4 shows the results of the work that was conducted in Brigham Young University (BYU) [46]. The green line indicates the original path planned by the deliberative path planner to the target destination. The blue-dot shows the position of the SUAV during deployment and the red square shows the obstacle. It was observed that the reactive behavior was triggered when the obstacle was detected and the SUAV navigated away from the obstacle.

Figure 2-4: Original path and new path computed by the deliberative path-planning and obstacle avoidance behavior (adapted from Griffiths et al. [46])

All layers of the control depicted in the control architecture as shown in Figure 2-2 and Figure 2-3 can either be automated or performed by the human agent. Ideally, in a full autonomous UAV system, it is desirable for the UAV to be given an abstract mission commands and the flight control system is capable of refining the commands to real time control signals. However, this could only be achieved if the most fundamental issue which is the automation of the inner-loop is accomplished.

The inner-loop of control is the most critical loop in ensuring that the UAV is stable and remain airborne. The success of the outer-loops of control is dependent on the robustness of this inner-loop. Failure of this loop will fail the entire system. Without automation, land-based pilot who interacts with the UAV in the inner-loop level is basically flying the aircraft by manipulating the control surfaces directly. This activity is a skill-based behavior (SBB) because the interaction demands continuous inputs from the human-agent who is assisted by sensory perception [49, 50]. SBB activity requires significant cognitive resource from the human-agent. Automating this loop will effectively decrease the workload of the land-based pilot [50]. Automation of this loop can be achieved by implementing the autopilot module. In the next section, the current COTS autopilot system will be investigated and the issues on its control architecture limitation will also be presented.

2.1.2 Review on the Autopilot Technology

The technology of small autopilot modules has advanced rapidly in recent years. Such commercially available modules have increased in quantity significantly. Some of the popular commercially off-the-shelf (COTS) autopilot modules are from MicroPilot [51], Piccolo [52], and Kestrel [53] (as depicted in Figure 2-5). Many researchers have begun exploring the use of COTS autopilot modules for their UAV systems, for examples; Low et al. [54], and Platanitis et al. [55]. Using COTS autopilot modules has the advantage of accelerating the development cycle. Also, these modules are presumable to be more reliable since they have gone through several tests before they are finally released to the end-users. Lower cost has also become an attractive feature of these modules, (considering the significant reduction of development and testing costs). Researchers believe that these COTS autopilot modules will increase the cost effectiveness of UAV development in future, and it is envisaged that UAVs will be affordable enough to be disposable [56].

Figure 2-5: Commercially off-the-shelf autopilot modules

The number of researchers using COTS autopilot has been increasing, which suggests that it will be a usual practice for UAV developers to incorporate these modules into their UAV systems for real world applications in the near future. However, current COTS autopilot module has a few drawbacks. With most of the autopilots, the flight plan definition is basically a collection of waypoints which are pre-defined offline and uploaded to the autopilot modules. During flight, the waypoint locations are hand-manipulated by the operator via a GCS. The highest

capability of these autopilot modules is the *go-to waypoint* behavior [57, 58]. However, when the missions become more complex, these autopilot modules may not be capable of accomplishing the task required. Also, the option for the end-users to specify the trajectory, on the fly, is usually not available. The inflexibility of the waypoint-manipulation and the obscure of user's defined trajectory on the autopilot module, have made them (SUAV controlled by autopilot module), unable to accommodate to the changing mission circumstances [58], which is often the case, in dynamic real-world applications.

Higher-level reactive behavior has to be imbued onto the SUAV control architecture to equip the SUAV with the capability of reacting to instantaneous threat or stimulus (as suggested by Narayan et. al [45]). The main hurdle is that usually COTS autopilot uses proprietary technology that is undisclosed (or partially-disclosed) to the end-users, in order to protect the manufacturer's intellectual property. Limited knowledge on the autopilot control architecture poses a difficulty to the researchers in modifying the original architecture to imbue higher-level algorithms.

For illustration, consider the following example. An UAV is tasked to navigate from its current location to a destination. Assuming that an obstacle has been detected and the UAV has to be directed away to prevent collision. In such a situation, the current autopilot algorithm has to be interrupted by providing the autopilot a new waypoint and trajectory. However, from the implementation perspective, questions often arises is on how to interrupt the autopilot algorithm and thus, redirect the UAV For example; *"How to provide a new waypoints to the autopilot?"*, *"How to set a new desired altitude and heading on the autopilot?"*, etc. These issues highlight the lack of a mechanism to bridge the interaction between the higher-level of control algorithm (which is developed by end-user) and the lower-level of control (which is developed by the autopilot manufacturers).

Researchers in BYU have successfully demonstrated higher level reactive behaviors using the Kestrel COTS autopilot module [46, 59-61]. Figure 2-4 shows the flight test result of the UAV deployed. However, there is difference between

their research and the work being investigated here. This thesis focuses on the investigation using MicroPilot autopilot module. The study will lead to a solution in modifying the autopilot control architecture for both the fixed-wing and rotating-wing UAVs. Whereas, the Kestrel autopilot module is only limited to fixed-wing UAV application. Furthermore, the Kestrel autopilot module was originally developed by the researchers in BYU, which was then commercialized. As a result, researchers in BYU will not face the proprietary technology issues as encountered by other researchers.

2.1.3 Issue in Designing the Low-Level Controller

In order to automate the inner-loop, a low-level controller has to be designed. Modern control strategies are being researched actively, for controlling UAV, such as linear-quadratic regulator controller (for hovering a small helicopter) [62] and fuzzy-based controller (for Kite UAV) [63]. However, classical linear controller using the proportional, integral and differential gains controller (PID) [43, 64] is the most widely implemented control algorithm on COTS autopilot modules [57]. The popularity of the PID controller algorithm is attributed to its simplicity and ease of implementation. Tuning of the gains can be easily performed and thus, provides flexibility to the developer to adjust the controller to their specific UAVs.

Although the PID algorithm may be simple, obtaining a suitable set of gains for the controller feedback loops is not a straightforward task. Many researchers who developed their own UAV using COTS autopilot module faced the same challenge in determining the PID gains [54, 55, 65, 66]. This is because there is no one universal set of gains that fit all type of UAVs. The gains to be implemented are system and operating conditions dependent [43, 64]. From the aircraft system point of view, the gains used are functions of the UAV aerodynamic characteristics and flight conditions.

Figure 2-6: Approaches to obtain the PID gains of a linear controller

Generally, there are two approaches to obtain the tuning of the gains, either by using the experimental or the analytical approach. Figure 2-6 provides the overview and examples of the two approaches available. The former approach is basically a ‘test, observe and tune’ approach of obtaining the controller gains. Flight tests are conducted using a set of gains selected (usually arbitrarily or based on the experience of the researchers) and responses of the UAV during the tests are observed. A new set of gains will be obtained by adjusting the previous gain values, based on responses during the flight test and the effect of the proportional, integral and differentiator gains to the response. Flight tests will be conducted again and the same ‘test, observe and tune’ approach will be repeated until the researcher concludes that the set of gains are “optimum” for their UAV. The analytical approach, in contrast, is a more systematic approach. This method requires a derivation of a mathematical model of the UAV dynamic motion. With the dynamic model, the PID gains will then be designed based on the classical feedback theory [43, 64].

The ‘test, observe and tune’ approach is considerably less tedious, however it is not an easy approach. Since the gains were selected without (or with limited) knowledge of the UAV dynamic, allowing the UAV to self-control using this set of gains may prove to be hazardous. Incompatible PID gains may induce system instability or drastic motions during the flight tests. Usually, in this situation, an experience and skillful pilot is required to take over the control from the autopilot,

when erratic responses are observed, to prevent it from crashing. The overall experience could be very stressful and taxing as, the risk of losing the UAV is tremendously high, not to mention the catastrophic consequences and damages that may occur. For example, assuming the UAV has a descent rate of 10ms^{-1} and is flying at an altitude of 120m, approximately 12 seconds are what it takes for the UAV to crash onto the ground. Tuning based on this method is tedious as more than one test are usually conducted in order to obtain suitable gains for one feedback loop. Despite the potential risk and the tediousness of this method, it is recommended by a number of autopilot developers [51, 52]. This is mainly due to the fact that it does not require theoretical modeling, which usually consumes a lot of man-hours to obtain the aerodynamic coefficients. This approach has been successfully demonstrated by Yeung et al. [67] using the Piccolo autopilot module successfully.

The analytical approach is a more explored approach in the literature and is preferred by researchers. There are four categories of methods under this approach, namely;

- (i) the parametric method (e.g., Data Compendium, DatCom [68] and Advanced Aircraft Analysis software [69, 70]),
- (ii) the computational method (e.g., Tornado [71], Computational Fluid Dynamics software such as Fluent [72]),
- (iii) the system identification method [73, 74], and
- (iv) the wind tunnel tests method.

These four methods are used to estimate the aerodynamic coefficients or the UAV *stability and control derivatives* (SCD) [43, 64], which form the mathematical model of the UAV. The analytical approach is a more time-consuming approach as estimating all the coefficients and validating the reliability of the coefficients estimated can be tedious. Even, a simplified linear model for the lateral and longitudinal motions of a fixed-wing aircraft consists of more than 20 SCDs to be estimated. Although time-consuming, the analytical approach is always preferred over the experimental because “*acquisition of high-fidelity models of the target UAVs is a critical step in the design of a high-performance flight management*

system”, as quoted from Shim et al. [75]. Examples of researchers using the analytical methods, according to the methods are; Platanitis et al. [55, 76] and Jodeh et al. [66] applied the parametric methods, Low et al. [54] used the computational methods, and Lee et al. [65] and Manaï et al. [77] used the system identification method.

2.2 Deployment Issues in an Urban Environment

In this section, the review focuses on two major issues pertaining to urban navigation. One of the main concerns in urban navigation is the presence of obstacles in the navigation path. Section 2.2.1 looks into the problem of the cluttered environment in the urban terrain and discusses on the work that had been done to address this issue. In Section 2.2.2, the importance of line-of-sight flying for UAV deployment will be presented. The review highlights the need of a solution to assist the pilot in navigating the UAV.

2.2.1 Present of Obstacles in the Navigation Path

Figure 2-7: Typical terrain in Singapore (from Google Earth [78])

The cluttered and high dense environment is a common sight in most of the global cities such as Singapore. Figure 2-7 shows a satellite image of a typical urban terrain in Singapore. Although it is easier to deploy the UAV to navigate at higher

altitude using waypoint navigation, it is desired to have the UAV capable of navigating nearer to the ground for missions such as search and rescue. It is a tough task for the UAV to navigate in such environment as finding a collision free path in a three-dimensional space is a challenging task. In spite of a widespread ongoing effort in solving this problem, there is currently, no UAV that could fly and operate in a real world urban terrain. Navigating while avoiding obstacles still remains a big challenge to the researchers.

Generally, if the environment that the UAV is navigating at, is known (i.e. the world model is available to the UAV system), online path planning capabilities can be implemented on the UAV to dynamically and continuously replan and search for a collision free path, as discussed in Section 2.1.1. However, in real world applications, the environment is usually unknown or just partially known. In such a situation, reactive behavior, i.e. reactive obstacle avoidance behavior (OAB) that responses instantaneously to the obstacles confronted will have to be incorporated in the flight control of the UAV.

Although, OAB has been widely implemented on land vehicles (for example, on the All-Terrain-Remote-Vehicle in Robotic Research Centre [26, 37]), the complexity of implementing of OAB on the UAV increases tremendously. Considering the speed that a UAV is flying at, fast and reliable decision making process is required. However, ironically, the microprocessor on UAV, typically SUAV and MAV is not as powerful as those on land vehicles. Also, in comparison with land vehicles, the motion for UAV flying in a three-dimensional space, which is rather more complicated and complex. Secondly, during the development stage, testing of any algorithm on the platform has to be done outdoor and is not easy to conduct. It is usually very costly and involving significant logistics arrangements. In addition, flight test has to be conducted in the presence of an experience ground-pilot, to perform recovery task should the UAV perform erratically during testing. These difficulties cause a great limitation to the number of tests that could be conducted. Furthermore, trial and error approach may not be feasible as failure of the algorithm or system may result in lost of the plane. In addition, UAV particularly SUAV and MAV, has a very limited payload. This significantly

reduced the number of on-board sensors and the types of sensor that it can carry. For example, the AVATAR helicopter platform from University of Southern California can only carry approximately 0.5 kg payload [79].

Fundamentally, the concept of OAB is to ‘see and avoid’. Thus, the OAB algorithm is very much dependent on the sensors used for object detection. Generally, there are two types of sensors used for object detection; the active and passive sensors [80], as shown in Figure 2-8. Active sensor emits energy to the environment in order to either modify the energy or enhance it. Sonar, laser, radar and ultrasounds are some of the examples of active sensor or also known as range sensor. The weight of a range sensor increases with the maximum distance a range-sensor can detect. As a result, laser and radar range sensors are either too heavy or the range of detection is too short for SUAV application. Besides, infrared and acoustic sensors are highly sensitive to noise and disturbance. In contrast to active sensor, passive sensor does not emit energy to the environment but require the environment to provide the medium for observation. Post-processing usually has to be carried out to detect the object. Vision systems have widely used on SUAV for object detection due to its light weight and low power consumptions. The following discussion in this section will provide review on the techniques researchers used to implement OAB on SUAV.

Figure 2-8: Type of sensors used for objects detection

Range-Based Sensors

Figure 2-9: UAV in University of California (adapted from Shim et al. [81])

Not many OAB algorithms have been developed using range-based sensor for UAV. Platforms with higher payloads are capable of carrying a higher resolution and further range sensor such as the rotary wing platform (Figure 2-9a) from the University of California, Berkeley [81]. A laser range finder (Figure 2-9b) is used to build a dynamic online map around the vehicle. Online re-planning of a collision-free path is computed based on the Model-Predictive-Control (MPC). This map is used together with the Model-Predictive-Control (MPC) algorithm to compute a collision-free trajectory based on real-time gradient-search based optimization. Figure 2-10 shows the result of the experiment conducted in validating the algorithm. The demonstration only shows the algorithm for a planar avoidance and the 3D navigation is not shown. The MPC algorithm requires reliable range data and the algorithm is computationally expensive. Thus, it may not be suitable for most of the SUAV and MAV.

Figure 2-10: Flight test result using MPC algorithm (adapted from Shim et al. [81])

Vision-Based Sensors

Passive sensors using vision system are more widely explored by researchers. The optical flow technique has been explored by Muratet et. al. [82, 83], Griffiths et al. [46], and S. Hrabar et al. [84, 85]. This technique compares the images captured sequentially and output the magnitude and direction of the changes in the pixel intensity [86]. As such, when the UAV moves relative to the obstacles, the outputs of the optical flow sensors could be used to estimate the distance of the obstacle from the UAV [46]. The vision method is found to be the most suitable technique to be used in SUAV in the current context. These methods are usually demonstrated using rotary-wing SUAV. This is because mounting a vision-system on a fixed-wing platform would be a big challenge. Usually a fixed-wing platform is a puller-type platform, i.e. the propulsion system is mounted at the nose the fuselage. As such, the motion of the propeller may effectively affect the accuracy of the image processing, when the camera is mounted to look forward. OAB using a fixed-wing has been demonstrated by researchers in BYU [46]. The researchers use a pusher-type airframe, as shown in Figure 2-11, where the propulsion system is installed at the tail of the fuselage. Two cameras were installed at an angle of 30^0 from body-fixed x -axis for the terrain avoidance using optical flow.

Figure 2-11: Flying-wing airframe in BYU (adapted from Griffiths et al. [46])

Optical technique using a single camera is limited to frontal obstacle avoidance only. Thus, the team from University of Southern California pursued a novel technique which uses a pair of forward facing fisheye cameras on the AVATAR helicopter platform as shown in Figure 2-12 [84, 85]. The researchers combine the stereo-flow and optic-flow approaches. The optical flow technique is similar to the

one discussed above and is used for lateral obstacle avoidance in urban canyon. The stereo-flow technique is used to avoid frontal collision. The stereo image captured will be processed by applying distance threshold and region growing. Features detected that are within a threshold will be taken as an obstacle and the helicopter will either stop or initiate a turn. This combination method is shown to be more effective for navigation in the urban setting than either of the two individual approaches (optical flow and stereo-flow).

Figure 2-12: AVATAR helicopter with vision system (adopted from Hrabar et al. [84])

2.2.2 Issue with Loss of Line-of-Sight

Table 2-1 shows three UAV deployment modes commonly used in UAV application. In autonomous mode deployment, the land-based pilot will perform monitoring and mission planning through the GCS. This mode does not require continuous input from the pilot. The instrument mode is a mode where the land-based pilot controls the UAV remotely, with the assistance from the display of the states feedback from the airborne UAV. In this mode, continuous control inputs are required. The ground-pilot will provide the desired guidance states inputs, i.e. the airspeed, altitude and heading to the UAV. In the last mode, which is the direct line of sight teleoperation mode using a RC transmitter (or RC mode), the land-based pilot basically manipulates the actuators directly in response to the pilot's visual feedbacks.

Table 2-1: UAV Modes of Control

Autonomous mode	Instrumental Mode	RC Mode
(with or without line of sight)	(without line of sight)	(with line of sight)
<ul style="list-style-type: none">• UAV operates autonomously after receiving the goal of the mission.• The human agent monitors the states and the success of the mission via the GCS. The human agent may also modify the mission.	<ul style="list-style-type: none">• UAV requires continuous control inputs from the GCS.• The human agent will fly the UAV using joystick (or transmitter) based on the states feedbacks and display on the GCS.	<ul style="list-style-type: none">• UAV requires continuous input from the human agent.• Human agent controls the UAV by maneuvering the control surfaces and throttle directly. Human agent depends on his direct visual feedbacks.

Direct line of sight teleoperation control is the most common deployment method used by UAV pilots. However, in an urban environment, visual obstruction by obstacles reduces the effectiveness of direct control due to the deprivation of visual cues. As a result, the pilot is unable to see the orientation of the vehicle and eventually may lose control of the aircraft. In practice, when a UAV is ‘lost’ behind a tree or building, the land-based pilot will switch from direct line of sight teleoperation mode to instrument control mode and vice versa.

The process of switching from one mode of control to another is difficult for most pilots, as they have to assimilate real world (direct line of sight) sensory data with visual sensor information provided through onboard sensors on the aircraft. It has been widely reported that operators experience the ‘soda straw effect’, where the operators assume that the limited field of view they are observing represents the entire flight surrounding. Often a temporary lapse in situational awareness (SA) could result in the loss of the UAV.

2.3 Summary of the Findings

Review of the control architecture affirms that the hybrid control architecture is essential to ensure a safe and successful mission. A hierarchical architecture is required to perform mission planning by accepting a high-level command and transferring it into a single action, which generates the input signals to the actuators. Reactive behaviors equip the SUAV with behaviors that can react instantaneously to external stimuli or danger. In urban deployment, where the environment is usually cluttered and unknown, reactive *collision prevention* and *obstacle avoidance behavior* is crucial to prevent the SUAV from colliding with the man-made structures.

Boskovic et al. [42] proposed a hierarchical control architecture for SUAV navigation, that provides four layers of control, i.e. from decision-making to waypoint navigation and trajectory generation, and finally to actuation control. Narayan et al. [45] improved the architecture by incorporating a reactive behavior subsystem. The control architecture by Boskovic et al. [42] and Narayan et al. [45] are applicable to full autonomous system as required by this research application.

The thesis implements the control architecture proposed by Boskovic et al. [42] and Narayan et al. [45] using COTS autopilot module. The study shows that the highest capability of the autopilot is limited to *go-to waypoint* deliberative behavior. This capability indicates that the autopilot is able to perform high-level task. However, reactive behavior, such as collision prevention is not available on the autopilot modules. Thus, in order to incorporate the reactive behavior subsystem, investigation will be carried out to determine on a mechanism that facilitate the integration of the reactive behavior subsystem into the original control architecture.

Further review on the autopilot module showed that, to accomplish high-level behaviors, the low-level controller that governing the motion and guidance control of the SUAV has to be achieved. Although, simple PID controller was implemented on the autopilot for the low-level controller, researchers are facing challenges in obtaining a suitable set of gain for their UAVs developed. The task

of designing the PID gains becomes more challenging when the aerodynamic data of the SUAV airframes are not available. As a result, the SUAV aerodynamic coefficients have to be estimated first, in order to derive the mathematical model that represent the dynamic motion of the SUAV. The mathematical model will be used to facilitate the design of the controller.

In the final section of the chapter, the review discusses on three deployment methods, i.e. the RC mode, the instrumental mode and the autonomous mode, a ground-pilot use when flying the SUAV. When the SUAV is flying within line-of-sight, RC mode is usually engaged to deploy the SUAV. However, in a situation where the SUAV flies behind an object (e.g. building), the ground-pilot has to shift to instrumental mode. Transition between control modes often results in losing of situational awareness. To address this issue, a graphical user interface on the GCS subsystem that incorporates all the three modes of control is desired. The motivation is to allow the ground-pilot to control the SUAV using a single interface.

CHAPTER 3: HUMAN-SUAV SYSTEM DEVELOPMENT

This chapter presents the Human-UAV system that has been developed for the research investigation. Figure 3-1 shows the overview of the SUAV system architecture. The aerial platforms are developed from RC airframes and each of them is integrated with a micro avionics subsystem. A ground control subsystem, together with the graphical user interface (GUI) was designed and developed to facilitate the communications and exchange of critical information between the SUAV and the ground-pilot.

Figure 3-1: The overall SUAV system architecture developed

The following discussion is structured as following; In Section 3.1, the aerial robotics platforms that have been developed will be presented and the general characteristics of each of them will be discussed. The avionic subsystem and the integration will be detailed in Section 3.2. In Section 3.3, the GCS subsystem and the software architecture developed will be discussed. Details of the software and control architecture for the avionics subsystem will be presented in Chapter 4. Finally, this chapter is summarized in Section 3.4.

3.1 Aerial Robotic Platforms

Current UAVs that are available commercially such as the Aerosonde, RMax and BAT-III are far too costly for the research study. RC airframes, on the other hand, which are inexpensive and can be easily purchased, are more practical to be used for this research investigation. A fleet of heterogeneous SUAVs developed is given in Table 3-1 and the general characteristics are presented.

Table 3-1: UAV airframes developed

Platform	General Characteristics
M36	Wing configuration: Fixed-wing – low wing Wing span: 1.20 meter Chord length: 0.20 meter Weight: 1.50 kilogram Operational time: Approximately 10 minute Power Supply: LiPo 11.1V 1500mAh
Rainbow	Wing configuration: Fixed-wing – high wing Wing span: 1.60 meter Chord length: 0.26 meter Weight: 2.72 kilogram Operational time: Approximately 10 minute Power Supply: LiPo 15.6V 2200mAh (2 in parallel)
ECO-8	Wing configuration: Rotary-wing Blade length: 1.00 meter Weight: 2.32 kilogram Operational time: Approximately 8 minute Power Supply: LiPo 11.1V 1120mAh (3 in parallel)

Both the M36 and Rainbow are fixed-wing airframes, while on the other hand the ECO-8 is a rotary-wing platform. The selections of these platforms are based on the smallness in size and inherent ease of deployment. They are less than two meter in wing length (for fixed-wing) with a weight of less than 10kg and are powered by lithium-polymer batteries. Although, an internal combustion engine type platform has a higher propulsion power, electrically-powered airframes are preferred for a number of reasons. First of all, an electrically-powered platform can be operated easily without much hassle as compared to engine-operated platform that requires refilling of fuel and ensuring no fuel leakage. Besides, electrically-powered platforms have lower acoustic signature, thus low noise pollution.

3.2 Avionic Subsystem

The hardware components used to build the Human-UAV system are mainly COTS products. The overview of the hardware components for the avionic (integrated into the Rainbow airframe) and GCS subsystems, is given in Figure 3-2. More details of the airframes and hardware development for the M36 and Rainbow vehicle platforms could be found in the reports by Hu [87] and Nguyen [88] respectively. The integration of the major hardware components will be presented in Section 3.2.1.

Figure 3-2: Hardware integrated on the Rainbow avionic subsystem and GCS

3.2.1 Integration of Flight Control Subsystem

The avionic subsystem (or the flight control subsystem, FCS [89]) consists of the flight controller, central navigation sensor-suite, propulsion/actuation subsystem, and payload sensors. The MicroPilot MP2128 autopilot module [51] was integrated to the UAVs as the flight controller for flight stability and navigation control. The main task of the flight controller is to perform data readings from the navigation sensor-suite. The sensor feedbacks will be used by the control algorithm to compute the control signals to the propulsion/actuation subsystem. Details of the control algorithm implemented will be further discussed in Chapter 5.

Figure 3-3: Sensor suite and actuation modules on the SUAV avionics subsystem

Figure 3-3 depicts the sensor suite on the avionics subsystem for measuring the UAV states. The autopilot module has built-in gyroscopes, accelerometers, and pressure sensors for measuring the UAV attitudes, altitude and airspeeds. These states are critical for the control of the stability of the UAV and to ensure that it is still airborne, for example in pitch, airspeed and altitude hold. The autopilot uses GPS for localization and digital compass for heading to allow it to navigate the

UAV from one location to another. An ultrasound based altimeter is employed to determine the altitude of the UAV from the ground to facilitate autonomous take-off and landing. The pictures of the physical components for the MP2128, digital compass and the ultrasonic altimeter can be identified from Figure 3-2. The digital compass is a magnetic compass and careful calibration, as instructed by the MicroPilot developer, has to be performed during the setup before the flight tests can be carried out.

Figure 3-4: Location of the hardware integrated on the Rainbow platform

Figure 3-4 shows the hardware components integrated onto the Rainbow SUAV. Most of the items such as the power supply and controller board (for fixed wing, MP2128^g was integrated) are located in the fuselage of the SUAV. The ultrasonic altimeter sensor was mounted on the bottom surface of the fuselage. The GPS antenna was mounted on a piece of copper plate to improve the reception of the GPS signal. The antenna was mounted on the fuselage, away from the autopilot board, to prevent interference. The hardware components integrated onto the ECO-8 helicopter, on the other hand, is as shown in the Figure 3-5.

Figure 3-5: Hardware components integrated onto the ECO-8 helicopter

The autopilot module (for rotating-wing, MP2128^{HELI} was integrated) was connected to the propulsion/actuation subsystem via the servo board. This board decodes the signal from the autopilot module into pulse-width modulation (PWM) signals for actuating the servo motors and to control the speed of the propulsion subsystem. Both the Rainbow and M36 platforms are puller-type aircrafts where the motors was installed at the nose of their fuselages and 11×7 propellers are used to propel the SUAVs.

Continuous and reliable communication links to SUAV is more critical than in conventional FCS for manned vehicles [75], as the ground-pilot can only interact with the SUAV remotely. Thus, to facilitate interaction, a remote RF modem was installed on the avionics subsystem for a two-way communication via 2.4GHz radio frequency (RF) wave. The remote modem communicates with the MicroPilot module via RS232 serial connection at a baud rate of 56 kbps and the telemetry

from the module is feedback to the GCS at a rate of 5Hz. Commands from the ground-pilot are also sent over to the module via this channel. A RC receiver (as shown as RC Rx in Figure 3-2) was also installed to allow the ground-pilot to transmit signal directly from the RC transmitter.

The discussion so far provided the common hardware component on all the three platforms. The payload sensors integrated onto the avionics subsystem vary with the vehicle platform. A camera subsystem with two video cameras was installed on the Rainbow platform and an object detection subsystem was integrated onto the ECO-8 helicopter. Since the M36 platform is relatively smaller, no payload sensor was integrated on it.

3.2.2 Payload Subsystem

Figure 3-6: Locations of the two video cameras on the Rainbow airframe

The difference between UAV and RC aircraft is the existence of payload subsystem on the former vehicle that allows it to perform task-related missions such as surveillance, object tracking and traffic monitoring. Two CCD video cameras (model: KPC-S226CB) with 1/4" lens and image resolution of 537(H) × 505(V) were installed on the Rainbow SUAV. The first camera was mounted on the nose of the fuselage as shown in Figure 3-6, to provide a forward looking view. The second camera was installed at the bottom of the fuselage to serve as a mission camera. Both video images are transmitted through individual independence transmission channels via 2.4GHz RF and are transmitted directly to the GCS.

The CMUCam3 is an embedded vision sensor, which is fully programmable using GNU GCC compiler [90]. The camera is used for color tracking using simple RGB threshold color model. The OmniVision OV6620 camera installed on the CMUCam3 outputs an image with a resolution of 356 x 292 pixels. Figure 3-7a depicts the CMUCam3 mounted on a Pandora Pan-Tilt unit and Figure 3-7b shows the CMUCam3 integrated onto the ECO-8 helicopter. In addition to the vision system, an active sensor is also being used for the investigation. A single-beam laser range sensor, OptiRange-400 [91], as shown in Figure 3-7c is used as the active sensor for object detection. The laser was mounted on the frontal of the helicopter as depicted in Figure 3-7d. The laser is capable of detecting obstacle as far as 400m [91], as specified by the manufacturer, and is used for forward object detection. Calibration of the laser sensor has been performed in an outdoor environment to ensure the reliability of the range detected. Both the CMUCam3 and laser sensor are interchangeable.

Figure 3-7: Passive and active sensors used for object detection

3.3 Ground Control Station Subsystem

In order to facilitate a seamless transition between control modes, the GUI on the GCS was designed to allow multi-mode deployments. All the three control modes as described in Table 2-1 would be integrated onto a single GUI. The major challenge in the integration was to incorporate the RC control mode into the GUI. In the RC mode (as shown in Figure 3-8a), the land-based pilot depends on his/her own visual feedbacks to control the SUAV and this requires the pilot to look up and search for the UAV. In order to lessen human fatigue and discomfort during the operation, the amount of time that a pilot needs to look up into the sky, searching for the plane, shall be reduced.

An automated tracking subsystem (ATS) was implemented to track the UAV and the video from captured from the video camera in the ATS will be displayed on the GUI. A zoom lens is attached to the video camera to provide a larger effective range of detection. Preliminary calculation showed that the ATS has an approximate effective range of 200m. The ATS is cable-routed to the GCS and is located within a few meters from the control station.

The video from ATS allows the land-based pilot to continuously observe the orientation of the aircraft, via the GUI without requiring him/her to look up for the plane. In short, the ATS video will surrogate the vision of the pilot following the aircraft and this concept is illustrated schematically in Figure 3-8b. Thus, by eliminating the need of the pilot to look up for the UAV, temporary lost in situational awareness during mode transition will be minimized.

A software system for the GCS was developed on a ruggedized laptop. The software system runs on the Windows XP Professional in the laptop with a 1.6GHz Intel Centrino Duo processor and 1GB RAM memory. The software system was written in C++ programming language under the Visual Studio 2007 environment based on the Microsoft Foundation Class (MFC) framework. The software architecture was developed for creation of the GUI and also to facilitate communications among the GCS, SUAV and ATS. The modules developed within the architecture will be further discussed in Section 3.3.1. The automated tracking

subsystem will also be highlighted along with the software module developed to control it. After understanding the framework of the software architecture, the GUI developed will be presented in Section 3.3.2 to describe its usage for multi-mode of control.

Figure 3-8: Concept in replicating the RC mode to a single user interface

3.3.1 Software Architecture

The software architecture implemented on the GCS subsystem is as shown in Figure 3-9, and is termed as the RRC-GCS. The three main modules, i.e. the *UAV Module*, the *Tracking Module* and the *Video Module* will be discussed as following.

UAV Module

The details of the UAV module frameworks are as illustrated in Figure 3-9. The *UAV Module* is an interface module between the RRC-GCS and the SUAV avionics system. The RRC-GCS comprises of two main layers, i.e. the console-to-SUAV layer and human-to-console layer. The former layer handles the communication between the UAV and RRC-GCS and is running in the background. The latter layer, on the other hand, runs on the foreground. This layer forms the interface that displays the feedback from the UAV in a readable form and captures the inputs or commands from the ground-pilot (or flight crew).

Figure 3-9: Framework of the GCS software architecture

In the background, a thread (sees the “Communication” module on Figure 3-9) was created to constantly read the telemetry data from the serial port which was connected to the ground modem. The telemetry can be grouped into four categories; (i) flight monitoring, (ii) navigation, (iii) status alarm, and (iv) flight state management [58], as tabulated in Table 3-2. The telemetry data received will be

converted to SI units and are dynamically updated to the shared global variables, which are accessible by other threads/modules. A dialog-based (based on the *CDialog* class) GUI, which will be discussed in Section 3.3.2, is created for ease of usage and data representation. The GUI will periodically read the global variables and update the digital cockpit and navigation map. Buttons, *drag-and-release* function, sliders are created to capture commands from the ground-pilot. The ground-pilot may also control the SUAV by connecting a joystick to the GCS. The inputs/commands captured are converted to the autopilot defined units before they are sent to the avionic subsystem.

Table 3-2: Telemetry received from the UAV

Telemetry Data Type	Telemetry Data
Flight monitoring	<ul style="list-style-type: none">• Desired/actual attitude angles• Desired/actual altitude and airspeed
Navigation	<ul style="list-style-type: none">• GPS Easting and Northing locations, GPS ground speed, GPS altitude and GPS heading.• Desired heading and actual compass heading
Flight state management	<ul style="list-style-type: none">• Flight command in execution.• Flight condition – takeoff, level, on ground• Mode of control
Status alarm	<ul style="list-style-type: none">• Battery voltages• GPS status• Failure states• RC transmitter status

The “*Communication*” module was built using the XTender^{MP} SDK [92] from the autopilot manufacturer. This console-to-SUAV module was developed in such a way that this layer is abstracted away from the details implementation of the human-to-console layer. Furthermore, this layer is the only layer that is dependent on the autopilot module. The advantage of using an abstraction layer is the possibility of reusing most of the software modules developed when the autopilot is changed [58]. The developed software architecture is also scalable to multiple UAVs control, for future development. The console-to-UAV layer was developed within a single *class* and this *class* can be duplicated to control more than one UAV. Details of the UAV module are as illustrated in Figure 3-10.

Figure 3-10: Details of the UAV module framework

Tracking Module

The implementation concept and the related hardware for the automated tracking subsystem will also be discussed here. The physical hardware components for the ATS are as depicted in Figure 3-11. A video camera (model: DSP 301-01) attached to a zoom lens (model: Fujinon H22x11.5R2D-ZP1) was mounted on a pan-tilt unit (model: PTU-47). The *Tracking Module* acts as an intermediate medium that retrieves the GPS locations from the data received from the autopilot and channels it to the Geo-Pointing Module or GPM, as depicted in Figure 3-9 . The latitude, longitude and altitude positions will be sent over to the GPM via RS232 communication protocol. Based on the GPS inputs, the GPM computes (the computation algorithm is hardcoded by the manufacturer) the corresponding pan and tilt signals to the pan-tilt unit (PTU), to direct the camera to the SUAV. The

task of searching for the UAV will then be replicated by the ATS. Video from the ATS will be rendered on the GUI for the view of the ground-pilot. The *Tracking Module* also computes the zoom magnitude required to the zoom lens based on the distance of the UAV from the location of the GCS.

Figure 3-11: Automated ground tracking system

Video Module

Video signals received from the onboard subsystem and from the ATS will be channeled to the RRC-GCS. The *Video Module* was developed to capture the video streams and render them on the window widget. Three videos will be displayed simultaneously on the GUI (refers to Figure 3-12 in next subsection).

3.3.2 Graphical User Interface

The discussion in the previous section mainly focused on the background modules in the GCS. In this section, the GUI, which runs in the foreground will be presented to explain its usage for multi-mode deployment. The layout of the GUI designed is given in Figure 3-12. The information and sensory cues on the GUI was customized for the three modes and Table 3-3 provides a summary of the information required for each of the deployment mode

Table 3-3: Information required for different deployment mode

Deployment Mode	Information Required
RC Mode	<ul style="list-style-type: none">• Video received from the ground tracking system
Internal Mode	<ul style="list-style-type: none">• UAV attitude (orientation), airspeed and altitude states• Forward looking video
Autonomous Mode	<ul style="list-style-type: none">• Navigation map with waypoint manipulation

From Figure 3-12, most of the display area on the interface was occupied by the three videos received from the onboard SUAV and ATS. The window widget on the upper left, middle and right of the GUI display the videos from the ATS, forward-looking camera and downward-looking video respectively. In the RC mode deployment, the ground-pilot will rely on the ATS video.

When the UAV is deployed in internal mode, the pilot will rely on the “*Digital Cockpit*” and the forward looking view video. This video assists the operator in navigating the vehicle and video information has been found to be more effective in assisting human to perform the navigation control [93]. The UAV states will be updated graphically on the “*Digital Cockpit*”. The cockpit shows three dials which depict the speed, orientation (roll, pitch and yaw angles), and altitude of the UAV respectively. Critical status such as the communication status and GPS signal status are also included on the digital cockpit to provide cue to the ground-pilot on the SUAV subsystem health. A joystick is connected to the GCS (via an USB hub) to provide direct teleoperation of the SUAV, using the internal mode.

In the autonomous mode of control, the SUAV navigates to waypoint location autonomously. An interactive navigation map at the bottom right of the GUI shows the designated waypoint locations. The latitude and longitude that defined the area of the picture map are keyed in from the GUI. From these inputs, the GCS is able to identify the two-dimensional flying area. The GCS computes the SUAV location in Cartesian coordinates based on the GPS location received from the SUAV, and display them on the “*Navigation Map*” display. The *drag and release* function was incorporated into the “*Navigation Map*” to allow the ground-pilot to manipulate and modify the waypoint locations.

Figure 3-12: Layout of the GCS graphical user interface

Besides providing the required information, the GUI also includes a “*Connection and Flight Log Display*” to facilitate the ground-pilot to select the SUAV to be deployed. In the same area, the “*Flight Log Display*” displays messages on the main events that are happening to the SUAV, for example, messages such as “*Rainbow is connected*” will be displayed when the SUAV is successfully connected. The log history allows the ground-pilot to revisit and check the status of the SUAVs.

3.4 Chapter Summary

Figure 3-13: Integration of avionics subsystem to the RC airframe

The development of the Human-SUAV system for this research has been presented in this chapter. Three heterogeneous SUAV platforms as discussed in Section 3.1 have been built and an avionics subsystem was integrated to each of the platform. Details of the avionics subsystem, together with the integration of the hardware into the airframes, were given in Section 3.2. The SUAV avionics subsystem was centered on the MP2128 autopilot module from MicroPilot. A video camera

system and an object detection subsystem were also integrated onto the Rainbow and ECO-8 SUAV respectively. The aim of the payload system was to prepare the SUAV for mission-based tasks and to detect obstacles. Figure 3-13 summarizes the main functions of the components integrated into the avionics subsystem.

The ground-pilot communicates with the SUAV using the GUI developed on the GCS detailed in Section 3.3. The GCS was designed based on the concept of integrating three main modes of control into a single GUI. The concept of replicating the RC mode was presented, where the ATS is used to track the UAV and the video from the ATS is rendered on the GUI. This video is aimed to assimilate the view of the pilot following the SUAV. Besides, the ground-pilot may control the SUAV in the internal mode assisted by the digital cockpit and forward-looking video. A two-dimensional picture map enhanced with waypoint manipulation allows the pilot to deploy the SUAV autonomously and change the mission profile accordingly.

CHAPTER 4: CONTROL ARCHITECTURE

The aim of this chapter is to provide a discussion on the control architecture proposed and the implementation of the architecture using the COTS autopilot module. This chapter is organized as following; Section 4.1 provides a general discussion on the possible hardware architecture that could be employed to implement the control architecture. In Section 4.2, an overview of the control architecture proposed will be presented. The discussions focus on the original autopilot control architecture and the modification that need to be carried out to facilitate higher-level reactive behaviors integration. Section 4.3 discusses the higher-level reactive behaviors algorithms implemented. Finally, a summary of the chapter is given in Section 4.4.

4.1 Hardware Configurations

Implementation of the control architecture is very much dependent on the availability of the computational resources on the avionics subsystem. In this section, three possible hardware configurations of the avionics subsystem are discussed. Figure 4-1 illustrates these configurations schematically. The configurations consider four main components, i.e. (i) additional microprocessor to increase the onboard computational resources, (ii) autopilot module, (iii) detection sensor for detecting external stimuli, and (iv) payload subsystem.

Figure 4-1a shows the architecture that employs an additional microprocessor (μ -Processor) as the main controller for the FCS. The μ -Processor interfaces with the autopilot module, detection sensor and payload subsystem. Communication is established between the μ -Processor and the GCS. The configuration was attempted by Jodeh et al. [66] on their fixed-wing SUAV. A Piccolo autopilot module was integrated to a PC-104 microprocessor. Both the higher-level deliberative (e.g. autonomous path planning) and reactive behaviors (e.g. obstacle avoidance) were implemented on the μ -Processor.

The second configuration is as illustrated in Figure 4-1b. The hardware configuration is similar to the first architecture presented in Figure 4-1a, with an exception that the detection sensor is connected to the autopilot instead of the μ -Processor. This configuration allows the reactive behaviors algorithm to be implemented on the autopilot module. The μ -Processor, on the other hand, focuses on implementing deliberative higher-level control.

Both the configurations discussed require an additional microprocessor onboard. The final configuration, on the other hand, excludes the μ -Processor from the avionics subsystem. As depicted in Figure 4-1c, the autopilot is the sole and main controller of the FCS. The detection sensor and the payload subsystem are connected directly to the autopilot module. The GCS sends commands and receives telemetry from the autopilot. Since, there is only one controller onboard, all both the deliberative and reactive behaviors have to be performed by the autopilot.

The first configuration eliminates the need of using the autopilot module to perform additional tasks that may increase the autopilot computational workload. In this configuration, the autopilot is employed mainly to perform low-level control, i.e. to control the actuation and propulsion subsystem. However, implementation of the reactive behaviors using this configuration has a disadvantage. Since the reactive behaviors are residing in the μ -Processor, the *time-to-execution* is inevitably longer. The *time-to-execution* is the duration taken, when the external stimulus was first detected to the time the SUAV begins to response to the behavior. This is due to the additional communication interfacing required between the μ -Processor and autopilot module.

Comparing the second with the first configuration, the *time-to-execution* for the second configuration will definitely be shorter. The reactive behaviors are implemented on the autopilot. Once the external stimulus is detected, the autopilot responses instantaneously to the stimulus and sends the control signals directly to the actuation and propulsion subsystem. Thus, the second configuration is a better option for implementing reactive behaviors that require very quick reactions.

However, for the second configuration, the implementation will be constrained by the autopilot module. The amount of the computational resources available on the autopilot is one of the constraints. Usually, the autopilot has very limited computational power, in order to reduce the power consumption. Most of the available resources may already be used by the manufacturer to implement the autopilot algorithm. Thus, the room for additional implementation is limited. The limitation disallows implementation of complicated or sophisticated behavior.

The first two configurations may not be feasible for smaller size UAVs. Having an additional microprocessor onboard will add extra weight to the vehicle. In addition, the microprocessor will increase the demand of the power requirements. Small payload available on the SUAV constraints the number of equipments and batteries the vehicle can carry. Besides, limited space available on the small platform increases the difficulty to incorporate more components to the already packed avionic subsystem. Due to these restrictions, it is desirable to have the lowest number of components possible on the onboard subsystem. Thus, the third configuration may offer a possible solution to SUAV.

Since, reactive behaviors require instantaneous response, the behavior will be implemented on the autopilot. However, as there is no additional μ -Processor onboard, deliberative behaviors that requires high computational such as autonomous path-planning may not be possible to be implemented onboard. In this situation, the computational burden of performing the higher-level deliberative behavior has to be shifted to the GCS. In the work by Griffiths et al. [46], the third configuration was adapted. Obstacle avoidance behavior (reactive behavior) was implemented on the Kestrel autopilot module and the autonomous path-planning runs from the GCS. Waypoints generated from the autonomous path-planning were sent to the airborne SUAV via the RF communication. Since the autonomous path-planning algorithm will be running externally, the update rate from the algorithm will be slower, as compared to having the algorithm running onboard.

Figure 4-1: Possible hardware architectures for the flight control subsystem

Table 4-1 summarizes the advantages and disadvantages of the three hardware configurations illustrated in Figure 4-1. As the vehicles used in this research are small in size and are modified from RC platforms, they are not designed to carry high payloads. Based on the discussion, the third configuration was a feasible solution and was adapted for the SUAV avionics subsystem for this research.

Table 4-1: Comparison among the three hardware configurations

Configuration	Feature	Advantage	Disadvantage
With Additional Processor #1 (Figure 4-1a)	<ul style="list-style-type: none">• Reactive and deliberative behaviors are implemented on the μ-Processor	<ul style="list-style-type: none">• Sophisticated reactive behavior.	<ul style="list-style-type: none">• Slower reactive response• Additional weight.• Increase power demand
With Additional Processor #2 (Figure 4-1b)	<ul style="list-style-type: none">• Reactive behavior is implemented on the autopilot.• Deliberative behavior is implemented on the μ-Processor	<ul style="list-style-type: none">• Faster response time to external stimulus.	<ul style="list-style-type: none">• Simple reactive behavior• Additional weight.• Increase power demand
Without Additional Processor (Figure 4-1c)	<ul style="list-style-type: none">• Reactive behavior is implemented on the autopilot module.• Deliberative behavior can be implemented on the GCS.• Commands are sent wireless from the GCS to the SUAV.	<ul style="list-style-type: none">• Smallest payload.• Faster response time to external stimulus.	<ul style="list-style-type: none">• Dependent on the GCS for high-level deliberative control.• Update rate from the GCS is slow.• Implementation of non-sophisticated reactive behavior algorithm.

4.2 Control Architecture on the Flight Control Subsystem

The control architecture implemented on the avionics subsystem developed was based on the architecture proposed by Boskovic et al. [42] and Narayan et al. [45]. The focus is on the accomplishment of the *go-to waypoint* behavior by automating the *motion and guidance* control and the integration of reactive behaviors to the FCS. The high-level decision making (e.g. pursuing a goal) and automation of navigation behavior (e.g. searching for a feasible optimum navigation path) are performed by the ground-pilot on the GCS.

4.2.1 Study on the Autopilot Control Architecture

Figure 4-2 shows the overview of the control architecture proposed and implemented. From the figure, the control architecture implemented by the autopilot manufacturer is basically a hierarchical architecture. The mission profile is planned offline (i.e. before the flight), where the commands and waypoint locations are uploaded to the autopilot module through a flight mission file (with *.fly extension). The autopilot supports four parallel processing threads that can run simultaneously. However, all the navigational commands such as *takeoff* and *go-to waypoint* can only be executed in the first thread, as restricted by the manufacturer. In this case, the other three threads can only used for non-navigational tasks such as performing arithmetic computations and payload control.

Table 4-2: Failure detection subsystem

Failure Detection	Failure Detection Behavior Flag
Lost in GPS	GPS signal not detected for more than 5 seconds
Lost in RC transmitter signal	RC signal not received for more than 5 seconds
Lost in communication with GCS	The telemetry was not successfully sent to the GCS for a period of 5 seconds
Low battery voltage	Battery voltage is lower than 10.0V

The executions of the threads and pattern commands as described are managed by the *Autopilot Mission Manager* (AMP), as shown in Figure 4-2. The AMP plays a role in monitoring the states and statues of the SUAV and accessing if the goals have been achieved. The AMP also monitors the *Failure Detection Module*, which is developed by the manufacturer, to detect if any failure occurs in the FCS. For example in an occasion if there is a loss of GPS signals. Table 4-2 lists the failure detections that are available. As tabulated in the table, the current remedy available is only to instruct the SUAV to return to its original take-off position and perform landing.

Figure 4-2: Overview of the control architecture implemented without additional microprocessor

The *Navigational Behavioral Controller* (NBC) receives a goal-behavior (e.g. *autonomous take-off*, *go-to waypoint* or *autonomous landing*) from the AMP. The NBC decomposes the goal-behavior into a set of sub-behaviors. The selection of the sub-behaviors to be activated will be conducted by the NBC to achieve the overall goal-behavior. For instance, as illustrated in Figure 4-3, to accomplish a *go-to waypoint* behavior, the *altitude hold*, *airspeed hold* and *heading-following* sub-behaviors will be called. In some of the sub-behavior, there is more than one way to achieve it, for example, the *altitude-hold* sub-behavior. Referring to Figure 4-3, this sub-behavior can be achieved either by adjusting the throttle (thrust power) or the SUAV pitch angle. In such situation, the NBC will activate only one of the sub-behaviors, based on the flight settings preset by the ground-pilot.

Figure 4-3: Activation of sub-behaviors to achieve waypoint-navigation behavior

Ideally the *Path Generator* should be automated, where given a final destination, the *Path Generator* will compute the navigation path to travel at autonomously [42, 45]. However, currently, the path generation is performed by the ground-pilot, where the waypoints are loaded onto the autopilot using the mission flight file. Based on the waypoints pre-loaded, the *Trajectory Generator* synthesises the navigation trajectory by computing the trajectory states, i.e. the desired airspeed, u_d desired altitude, h_d and desired heading, Ψ_d . The trajectory states will be used as the control inputs to the *Motion and Guidance Controller* and the control signals to the actuation/propulsion subsystem will be computed. The *Motion and Guidance Controller* implemented by the manufacturer is based on the proportional, integral and derivation (PID) controller.

The *HumanControl* module included in the control architecture as depicted in Figure 4-2 is basically the GUI developed on the GCS, which has been earlier discussed in Chapter 3. The *Online Planning* module allows the ground-pilot to manipulate the waypoints. The manipulation can be carried out in two manners; Firstly, the existing waypoints can be moved or changed to a new location using the GUI developed (see Section 3.3.2). Secondly, the entire navigation path could be re-planned by uploading a new flight mission file (using *.fly) to the autopilot while the SUAV is on flight.

The discussion so far highlighted the structure of the autopilot original control architecture. The autopilot manufacturer allows a certain degree of flexibility in amending the original control architecture. The XTender^{MP} SDK [92] is used to develop the *UserCode*, which consists of the algorithm developed by the end users to implement a certain algorithm. The *UserCode* is written in C programming language. As the control architecture is not fully open, the manufacturer forbids the access to the autopilot code. The *UserCode* developed (in *.bin extension) is appended to the original autopilot code (also in *.bin extension) to form a new code. The new code is then cross-compiled to the autopilot microprocessor using *cgywin* [94]. Only a maximum of 64K memory is allocated for the *UserCode*. Such limitation suggests that the processing power of the autopilot module is extremely limited and as a result, the room for development of higher-level behaviors is also highly constrained.

4.2.2 Incorporation of Navigational Reactive Behavior Controller

Figure 4-4 shows a schematic drawing showing the use of the *UserCode* to modify the original autopilot control architecture. From the discussion on the original autopilot control architecture in Section 4.2.1, it was found that, each control layers receives a set of control inputs and the control layer will compute the control outputs that will be passed on to the next control layer. Thus, *UserCode* can be developed to overwrite the control inputs, at the moment before they are passed on to the subsequent control layer. Effectively, the commands from the algorithm developed on the *UserCode* will replace the commands computed by the original algorithm implemented by the autopilot developer.

Figure 4-4: Mechanism to append *UserCode*

To illustrate, take the *Trajectory Generation* as an example. This controller forms the second control layer (sees Layer-2 in Figure 4-4). The layer receives the waypoint locations from the *Path Generator* (sees Layer-3 in Figure 4-4) as inputs and computes the desired airspeed, altitude and heading (i.e. u_d, ψ_d, h_d) as outputs. In order to overwrite the original algorithm implemented on the *Trajectory Generation*, high-level reactive behaviors can be implemented on the *UserCode* (in this example, *UserCode-B*) to compute and replace the desired airspeed, altitude and heading computed by the *Trajectory Generation*. The commands from the *UserCode-B* effectively subsume the commands from the *Trajectory Generation*.

The mechanism discussed was used to incorporate the reactive behaviors into the FCS. *UserCode* was used to develop the *Detection Subsystem* and the *Navigational Reactive Behavior Controller*, NRBC (sees Figure 4-2). The former module is an interface layer that constantly queries sensors reading from the external sensors integrated into the avionics subsystem. In this context, the external sensors are the laser range sensor and the CMUCam3 (sees Section 3.2.2). The sensor readings will be stored in the shared memory location, which can be easily accessed by the *UserCode* in all the control layers. The NRBC plays the role to select the reactive behaviors to be activated based on the behavior flags as given in Table 4-3. Currently two reactive behaviors have been developed and they are the *collision prevention & obstacle avoidance behavior* and the *hover & track*. Detail discussions on these behaviors are presented in Section 4.3.

Table 4-3: Reactive behaviors implemented

Behaviors	Description	Behavior Flag
Collision prevention & obstacle avoidance	The SUAV will hover at its current location when the distance between the SUAV and the obstacle is less than 150 meter. The SUAV will then wait for 4 seconds before making a 90° turn.	Laser Flag
Hover and track	The SUAV will hover at its current location. The heading of the SUAV will be adjusted so that the SUAV point to the target.	CMUCam 3 Flag

To summarize this section, two major controllers have to be implemented on the SUAV, i.e. the *Motion and Guidance Controller* and the *Navigational Reactive Behavior Controller* (NRBC). The former controller has to be firstly achieved to allow the SUAV to be capable of tracking a trajectory and maintain its stability. Details of implementing this controller are given in Chapter 5. The reactive behavior algorithms implemented on the NRBC will be discussed in the next section.

4.3 Higher Level Reactive Behaviors

Demonstration of the reactive behaviors will be conducted using the ECO-8 rotating-wing SUAV. Rotating-wing SUAV is selected as the higher-level reactive behavior testbed because of its capability of hovering. This capability allows it to

be treated like a land-vehicle moving in a planar motion at a constant altitude. Another advantage of using a rotating-wing SUAV is that, it does not require a runway for taking-off and landing. Furthermore, the ECO-8 SUAV can be commanded to fly at a slower speed and lower altitude.

The reactive behaviors can only be engaged after the ECO-8 SUAV has taken-off and climbed to a given altitude. The constraint imposed, prevents the behaviors being triggered during take-off or landing since during these phases, the SUAV may be too near to the ground.

Section 4.3.1 presents the *collision prevention & obstacle avoidance* behavior and the *hover & tracking* behavior will be discussed in Section 4.3.2. For both behaviors, the algorithms were developed on *UserCode-B* (see Level-2 on Figure 4-4).

4.3.1 Collision Prevention and Obstacle Avoidance Behavior

Figure 4-5 depicts the flow of the algorithm implemented for the *collision prevention & obstacle avoidance* behavior. The single beam laser range finder is used to detect the presence of a frontal obstacle that will return the distance of the object from the SUAV. The autopilot module reads the distance from the laser sensor at a rate of 10Hz. When the distance of the frontal object is below the threshold distance preset, the Laser Flag (see Table 4-3) will be enabled and the reactive behavior will be engaged.

After the behavior is called, the NRBC will hover the ECO-8 SUAV at its current location and altitude. A four seconds delay period is imposed intentionally to allow the SUAV to stabilize itself after the reactive behavior is called. After four seconds, the SUAV makes a 90° turn to the left and this is done by gradually reducing the desired heading of the ECO-8. Since, no modification was performed on the *Motion and Guidance Controller*, this controller will receive the desired heading computed by the NRBC and generate the control signals to the tail rotor control surface in order to track the commanded desired heading. Execution of this behavior will prevent the SUAV from continuing to move forward and collide with the obstacle.

Figure 4-5: Collision prevention behavior implemented on the ECO-8 SUAV

4.3.2 Object Tracking based on Color Identification

The tracking behavior is demonstrated using the CMUCam3 as the passive sensor for RGB color detection. An orange object (with RGB color code of 143, 25, and 18) is used as the tracking target. The aim of having this behavior is to allow the ECO-8 SUAV to track an object and capture the image of the object for post-processing purpose.

Figure 4-6: Axis-frames attached on the CMUCam3 and the SUAV

A *search object* sub-behavior is programmed into the CMUCam3. The CMUCam3, which is mounted on a pan-tilt unit, will continuously search for an orange object. If an orange object is detected, the behavior estimates the centroid of the object. The aim of this sub-behavior is to ensure that the centroid of the object is always at the center of the image. If the distance between the centroid and the image center, $\bar{x}$ is not zero, the CMUCam3 will command the pan-tilt unit to pan to the direction that could reduce $\bar{x}$. When the CMUCam3 was panned from its original direction, the camera heading will not be aligning with the SUAV heading as depicted in Figure 4-6. The algorithm will then estimate the angle difference, ϵ between the camera heading, Ψ_{cam} and the ECO-8 SUAV heading, Ψ_{heli} , as given in the equation (4-1).

$$\epsilon = \Psi_{heli} - \Psi_{cam} \quad (4-1)$$

When an orange object was successfully detected, the CMUCam3 Flag (sees Table 4-3) will be activated and the angle difference, ϵ will be written to the shared-memory in the autopilot. When the NRBC detects that the CMUCam3 Flag has been turned on, the *hover and track* behavior will be engaged. The NRBC hovers the ECO-8 SUAV at its current location and begin to save the images captured into the SD card. For the SUAV to point to the object detected, the SUAV heading shall be aligning with the camera heading. To achieve this, the NRBC will adjust the SUAV heading so that the angle difference is minimized (i.e. $\epsilon = 0$). The algorithm discussed in this section is illustrated with a flowchart as shown in Figure 4-7.

Figure 4-7: Hover and track behavior implemented on the ECO-8 SUAV

4.4 Chapter Summary

The chapter began by discussing the three possible hardware configurations for the avionics subsystem. Based on the discussion presented in Section 4.1, the configuration that requires the smallest payload, i.e. without additional microprocessor onboard, was found to be suitable for this research purpose. The control architecture based on the one proposed by Boskovic et al. [42] and Narayan et al. [45] has been implemented on the SUAV avionics subsystem. Section 4.2 investigated the control architecture implemented on the COTS autopilot module. In order to facilitate incorporation of reactive behaviors, a mechanism to modify the original control architecture was discussed. The mechanism was used to implement the *Navigational Reactive Behavioral Controller*. In Section 4.3, the algorithms for the *collision prevention & obstacle avoidance behavior* and *object tracking behavior* implemented on the *Navigational Reactive Behavioral Controller* were presented.

CHAPTER 5: MOTION AND GUIDANCE CONTROL

In the control architecture presented in previous chapter, the discussion showed how motion and guidance control (also known as the low-level control) is vital to facilitate the implementation of higher-level behaviors. In this chapter, details of the approaches employed to design or to obtain the motion and guidance controller will be elaborated. Both the analytical and experimental approaches discussed earlier in Section 2.1.3, are presented in Section 5.1 and Section 5.2 respectively.

5.1 Analytical Approach

The concept of the analytical approach is to first obtain a mathematical expression describing the motion of the SUAV when it is subjected to a control input. Based on the expression, a controller is designed to generate the control input required for achieving the desired motion response. Section 5.1.1 presents the methodology for the analytical approach.

5.1.1 Controller Design Methodology

Figure 5-1: Methodology to design the *motion and guidance* controller for autopilot

The analytical approach presented in this section is based on a similar work carried out by Platanitis et al. [55]. In this subsection, the key steps of the methodology will be discussed. Figure 5-1 shows three main phases of the approach, i.e. the *modeling phase*, the *controller design phase*, and, the *verification phase*.

Modeling Phase

Generally, the motions of the SUAV can be modeled in three stages as shown in Figure 5-2. A SUAV will have to first take-off to transit from the ground to the air and begin to climb to a target altitude. Then, the SUAV will navigate to a desired location in level-flight conditions (either straight and level, and level and coordinated turn). In the final stage, the SUAV will descend to land.

Figure 5-2: SUAV deployment stages

In the *modeling phase*, the mathematical model derived and the investigation is performed based on the level-flight conditions. The UAV motions are firstly modeled by applying Newton’s Second Law. Since the PID gains for the ECO-8 SUAV *motion and guidance controller* had been determined by the autopilot manufacturer (as included in Appendix E), the discussion in this chapter will focus only on fixed-wing SUAV. This is to facilitate the determination of PID gains for the M36 and Rainbow SUAVs. The details derivation of the equations of motion (EOM) for a fixed-wing aircraft is given in Appendix B and the meaning of the symbols used in this chapter onwards are listed in the *List of Symbol*.

In this subsection, the EOM will be presented briefly. An aircraft is basically a 6-DOF vehicle that translates in forward, backward or side-way directions and also to rotate in the roll, pitch and yaw directions. The longitudinal direction equations consist of the longitudinal force (i.e. the force along the X and Z axes of the stability coordinate system) and pitching moment as depicted in the following equations:

$$F_{a_x} = C_x \bar{q} S \quad F_{a_z} = C_z \bar{q} S \quad M = C_m \bar{q} S \bar{c} \quad (5-1)$$

The C_x and C_z coefficients can be related to the drag and lift coefficients using the following relationships, assuming small angle of attack (less than 15°) [95].

$$\begin{Bmatrix} C_D \\ C_L \end{Bmatrix} = \begin{bmatrix} -\cos \alpha & -\sin \alpha \\ \sin \alpha & -\cos \alpha \end{bmatrix} \begin{Bmatrix} C_x \\ C_z \end{Bmatrix} \quad (5-2)$$

The lateral-directional force and moments are the side-force, rolling moment and yawing moment. There are as depicted in the following equations:

$$F_{a_y} = C_y \bar{q} S \quad L = C_l \bar{q} S b \quad N = C_n \bar{q} S \bar{c} \quad (5-3)$$

The complex and coupled non-linear EOM for an aircraft can be simplified into linear and decoupled EOMs by applying small perturbation theory [43, 64]. This can be performed by analyzing the flight mechanics at trim conditions, when the aircraft is in its stable operating point. The trim conditions are achieved when the aircraft is in equilibrium (i.e. the summation of total forces and total moments are zero), for examples, during straight and level flight. The linear EOM is obtained by expanding the equations using Taylor's series. The dimensionless variables for the longitudinal and lateral directions are $\left[\frac{u}{U_0}, \alpha, \frac{\dot{\alpha} \bar{c}}{2U_0}, \frac{q \bar{c}}{2U_0}, \delta_e \right]$ and $\left[\beta, \frac{\dot{\beta} b}{2U_0}, \frac{p b}{2U_0}, \frac{r b}{2U_0}, \delta_a, \delta_r \right]$ respectively. The values of the partial differentiations with respect to the dimensionless variables are the SCD coefficients.

In practice, the linear differentiation equations obtained can be conveniently represented in a state-space form as depicted in equation (5-4). The states vector, X for the longitudinal direction and lateral direction are $[u, \alpha, q, \theta]$ and $[\beta, p, r, \phi, \psi]$ respectively. Y is the output matrix and U is the input matrix. The A and B

matrixes are formed by the SCD and physical properties of the SUAV. Matrix **C** is the output matrix and **D** is the direct transmission matrix, which can be assumed to be zero [64], in this analysis. Details of the longitudinal and lateral linear EOM can be found in Appendix B or from the “*Performance, Stability, Dynamics, and Control of Airplanes*” text reference by Pamadi [64].

$$\begin{aligned}\dot{X} &= \mathbf{A}X + \mathbf{B}U \\ Y &= \mathbf{C}X + \mathbf{D}U\end{aligned}\tag{5-4}$$

Although the EOM for an aircraft is generic, the physical and aerodynamic properties are unique to each vehicle. These properties have to be estimated to complete the mathematical model.

The vehicle physical properties required are the total mass, the three-dimensional center of gravity (CG) and the moments of inertia (i.e., I_{xx} , I_{yy} , I_{zz} , I_{xz} , I_{xy} , and I_{yz}). These properties can be directly estimated from the physical platform [66]. The CG location can be estimated by finding the point where the weight can be balanced with a single point support. The moments of inertia can be estimated indirectly by hanging the SUAV at its CG location and measuring the damped oscillation periods. However, measurement of a 3-axis body-fixed coordinate of the CG location and the cross-moments of inertia (i.e., I_{xz} , I_{xy} , I_{yz}) are difficult.

Alternatively, the physical properties can be estimated using computer-aided-drawing (CAD) software [55, 76]. Some of the common commercial CAD software modules are CATIATM, SolidWorksTM, and ProEngineeringTM. The SUAV and the avionics subsystem have to be reconstructed into a CAD model. The arrangement of the components on the avionics subsystem had to be drawn as close as possible to the actual arrangement. This is because the arrangement will change the weight distribution, thus will affect the physical properties estimated.

The methods for estimating SCD has been earlier presented in Section 2.1.3. Therefore, they will not be iterated here. Different methods will require different sets of input. Thus, it is not of the interest of this thesis to cover all the methods.

However, when considering only the linear motion, the SCD estimated by any methods has to be based on the trim condition. The SCD needed for the model are given in equation (5-5). The flight altitude, h_0 and the SUAV airspeed, U_0 at the trim points have to be identified *a priori* to the SCD estimation. Flight test in open loop (i.e. the flight controller is not engaged) can be carried out by the ground-pilot to estimate these two flight conditions.

$$\Theta = \begin{Bmatrix} C_{D_0}, C_{D_\alpha}, C_{D_{\delta e}}, C_{L_0}, C_{L_\alpha}, C_{L_{\delta e}}, C_{Y_0}, C_{Y_\beta}, C_{Y_{\delta a}}, C_{Y_{\delta r}}, C_{l_0}, \\ C_{l_\beta}, C_{l_{\delta a}}, C_{l_{\delta r}}, C_{m_0}, C_{m_\alpha}, C_{m_{\delta e}}, C_{n_0}, C_{n_\beta}, C_{n_{\delta a}}, C_{n_{\delta r}} \end{Bmatrix} \quad (5-5)$$

The parametric method using Advanced Aircraft Analysis (AAA) [69, 70] software was employed to estimate the SCD coefficients. The AAA software was developed based on the aircraft design books written by Roskam [70]. It uses classical aerodynamic theory to predict and estimate the coefficients of conventional aircraft design from a database of wind-tunnel tests conducted by NASA. Among the various modules available for estimating the SCD, the AAA module is used because it has a GUI to facilitate the data inputs to the module and is more accurate [96]. The original purpose of the software is to assist an airplane designer to design an airplane from scratch. However, it could also be used to estimate SCD coefficients if the aircraft dimensions are available. The dimensions serve as the inputs to the AAA software module for computing the corresponding SCDs. Figure 5-3 shows the inputs required and outputs generated from the AAA software module. The SCDS values generated by the AAA software provide an initial estimate of the aircraft coefficients. These values, however, should not be taken as the final accurate SCDs. Other methods (e.g., wind tunnel test and CFD) shall be used to validate these values. However, in this thesis, since the objective of obtaining the mathematical model was to facilitate the initial PID gains tuning for the first flight test, the SCDs estimated by AAA is used as the final values.

Figure 5-3: Inputs and outputs from the AAA software module

Controller Design Phase

The automation of a flight controller is basically to implement a control algorithm that will compute the control signals to the control surfaces (i.e., the aileron and elevator) and the throttle of the SUAV automatically based on the states feedbacks from the sensor-suite. This second phase is dependent on the types of controller implemented on the autopilot module and the configuration of the SUAVs.

The main task is to derive the single input single output (SISO) transfer function from the mathematical model. The SISO transfer functions are dependent on the configuration of the SUAV. As an illustration, the control of the aileron and elevator of a SUAV are discussed. In order to facilitate the explanation of the derivation of the SISO transfer function engaged for the flight control automation, the effect of the control surfaces deflection on the SUAV motions are firstly presented.

Effects of the Control Surfaces Deflections on the SUAV Motions

The motions of the Rainbow SUAV are controlled using the elevator and aileron, as shown in Figure 5-4a. The decoupled EOMs simplify the analysis, as the motions in the longitudinal and lateral directions are independent of each other. The deflection of the elevator control surfaces can be assumed to have no impact on the lateral motions. Similarly the deflection of the aileron will have negligible effect on the motion in the longitudinal direction.

The elevator is a small flap attached on the horizontal tail of the SUAV (see Figure 5-4a). A deflection of this control surface will change the net pitching moment, which effectively change the SUAV pitch angle. For example, when the elevator trailing edge is deflected downwards (positive deflection angle), a negative pitching moment is created and as a result, the SUAV pitches down as illustrated in Figure 5-4b. The aileron deflection will change the rolling moment, as depicted in Figure 5-4c. When the trailing edge of the left aileron is deflected downwards, the SUAV will roll to the right about its x -axis direction.

The dominant effect of one control surface deflection to one of the rotation movement allows the assumption of analyzing the motion using single-input-single-output (SISO) system. In this case, the elevator and aileron can be used to control the angle of pitch and roll respectively

Figure 5-4: Location of the control surfaces and the effects on the SUAV motion

Feedback Control Loops

In level flight, the aircraft is expected to navigate from one location to another while maintaining the flight at the target altitude and cruise speed. To maintain a constant altitude, h_0 and airspeed, u_0 , the *altitude-hold* and *airspeed-hold* behaviors are engaged. These two behaviors are achievable by controlling the throttle setting, δ_{th} and the pitch angle, θ respectively. In a case there is a drop in the altitude, the throttle will be increased to provide a higher thrust to the SUAV. An increase in the thrust force will generate a higher lift force (and vice-versa), which will eventually lift the SUAV to nearer to the targeted altitude.

To achieve *airspeed-hold* behavior, the airspeed is kept constant by adjusting the SUAV pitch angle. For example, when the SUAV airspeed is lower than the cruise speed, the SUAV pitch angle will be decreased (so that the SUAV dives) in order to increase the airspeed back to the cruise speed.

The *heading-hold* behavior is engaged when the SUAV is directed to travel to a given location. This behavior ensures that the SUAV is navigating at the targeted heading, which will guide it to the desired location. The *heading-hold* behavior is achieved by controlling the roll angle, ϕ of the SUAV. For example, by rolling the SUAV to the left will effectively change the SUAV heading towards the left of its original heading.

Figure 5-5: Block diagram of the *airspeed-hold* and *heading-following* behaviors

The *altitude-hold*, *airspeed-hold* and *heading-following* behaviors are also known as the *guidance and motion control* loops [43, 64]. Each of the latter two behaviors consists of two cascaded feedback loops of control. Figure 5-5a illustrates the cascaded feedback loops engaged to achieve the *airspeed-hold* behavior. The outer loop generates the desired pitch angle, θ_{des} based on the error between the targeted airspeed, u_{des} and the actual airspeed, u_{act} . The inner loop, in this case, is the *pitch-hold* control loop. This loop controls the SUAV by generating the control signal to the elevator control surfaces, δ_e . The control aim of this inner-loop is to minimize the error between the desired pitch angle and the actual pitch angle, θ_{act} .

Similarly, in the *heading-hold* cascaded feedback loops of control as depicted in Figure 5-5b, the outer loop computes the desired roll angle, ϕ_{des} by minimizing the error between the desired heading, Ψ_{des} and the actual heading, Ψ_{act} . The *roll-hold* loop will then generate the control signal, δ_a to deflect the ailerons so as to minimize the error between the desired roll angle, ϕ_{des} and the actual roll angle, ϕ_{act} .

After identifying all the various SISO loops of interest (i.e. by knowing the control input and output response of the loop), the transfer functions are derived by applying Laplace Transformation to the time-domain equations. In the Laplace domain, the linear controller can be designed using methods of the classical control theory [43, 64] (e.g. the root-locus method and the bode-plot method). Alternatively, software tool such as SISOTOOL from Matlab [97] can be employed to graphically tune the controller gains (i.e. by varying the gain values) and analyze the control system response. SISOTOOL is a toolbox that facilitates the designing of the controller for SISO systems. The design tools available in the toolbox are *step response optimization* and *root placement* on the root locus plot, to cite a few.

Verification Phase

Flight tests are conducted to collect flight data for verification purpose. The autopilot is engaged to perform autonomous flight in the level flight conditions used to derive the mathematical model. Data collected will be used to verify the

suitability of the PID controllers designed for tracking the commanded states. The same set of data can also be used to verify the accuracy of the mathematical model.

5.1.2 Formulating the Mathematical Model for Rainbow UAV

The methodology was applied to both the M36 and the Rainbow fixed-wing platforms. The following will discuss the results obtained for the Rainbow UAV and that of the M36 were appended in Appendix D. In this subsection, the *modeling phase* of the Rainbow platform is presented. The methods employed to estimate the physical and aerodynamic properties to form the linear mathematical model will be detailed. Section 5.1.3 then elaborates on the *controller design phase*. The *verification phase* is included in Chapter 6 together with the experiment results.

Center of Gravity Location and Moments of Inertia

The Rainbow UAV was drawn using the SolidWorks™ CAD system. The drawing included the entire airframe and the electrical and mechanical components of the avionics subsystem. The placements of the hardware components in the drawing assimilate the physical locations. This was critical in providing an accurate estimation of the CG and moment of inertias. The physical properties were then estimated using the CAD software.

Figure 5-6: Location of the center of gravity for the Rainbow UAV

The mass estimated by the software was 2.62kg which is 0.06kg or 2.2% less than the actual mass of the UAV. This is because cables and other small components such as linkages were not modeled in the CAD drawing. The CAD drawing of the Rainbow together with the location of the CG estimated is depicted in Figure 5-6a.

Comparison between the estimated and the actual CG locations (see in Figure 5-6b) along the x-axis showed that the estimated CG was very close to the actual value. The actual CG is located 10mm aft the CG estimated from using SolidWorks™ software. The comparison shows that the CAD model is accurate. The moments of inertia were also estimated from the SolidWorks™ and Table 5-1 tabulates the results. Since, the aircraft was assumed to be symmetrical along the *xz-plane*, the I_{xy} and I_{yz} cross moment of inertias were assumed to be negligible [64]. The values of the moment of inertias are found to be very small (sees Table 5-1) and this indicates that SUAV has a higher tendency to oscillate at its own axes compared to the actual size aircraft [98].

Table 5-1: Fixed-wing SUAV moments of inertia

Moment of Inertia	Value	Moment of Inertia	Value
$I_{xx} (\times 10^2 \text{ kgm}^2)$	8.0577	$I_{xz} (\times 10^2 \text{ kgm}^2)$	0.5573
$I_{yy} (\times 10^2 \text{ kgm}^2)$	1.3881	$I_{xy} (\times 10^2 \text{ kgm}^2)$	0.0000
$I_{zz} (\times 10^2 \text{ kgm}^2)$	1.9832	$I_{yz} (\times 10^2 \text{ kgm}^2)$	0.0000

Aerodynamic Coefficients Estimation

The SCD were computed based on the trim conditions given in Table 5-2, where the SUAV is assumed to be flying at level conditions. The altitude and speed trim values were obtained from an open-loop flight test (i.e. no flight controller is engaged), where the Rainbow cruise speed, u_0 was approximately 15.0 ms^{-1} at 120m operating altitude, h_0 . Since the flight envelope for the Rainbow SUAV is not large, it is assumed that the changes in the flight condition will not significantly affect the aerodynamic coefficients [99]. The trim angle of attack, α_0 was estimated from the AAA software module.

Table 5-2: Flight conditions used to estimate the aerodynamic coefficients

Flight Condition	Value
Altitude, h_0	120.00 m
Speed, u_0	15.00 ms^{-1}
Angle of attack, α_0	2.09°

The detailed geometry information required are the fuselage, wing, horizontal wing, vertical wing, and the control surfaces geometries. These detailed dimensions of the RC airframes used in this research project were not provided by the manufacturer. As a result, the only way to obtain the dimensions was to measure the dimensions directly from the physical airframe [100]. Since the number of inputs required is too large, it is impossible to describe them entirely in this section.

Table 5-3: Input parameters for the horizontal tail geometry

Trim Condition	Value
Horizontal tail span, b_h	120.00 m
Horizontal tail chord at the root of the tail, c_{rh}	15.00 ms ⁻¹
Horizontal tail chord at the tip of the tail, c_{th}	2.09°
Horizontal tail quarter-chord sweep angle, $\Lambda_{c/4h}$	-11.75°

The inputs for the horizontal tail geometries are taken as an example. Table 5-3 shows the geometry inputs required for the horizontal tail. Four input parameters, i.e. the horizontal tail span (b_h), the horizontal tail chord at the root and the tip of the tail (c_{rh} and c_{th}), and finally the horizontal tail quarter-chord sweep angle ($\Lambda_{c/4h}$) had to be keyed in. By entering these four values, the AAA software computed the relevant output parameters. Figure 5-7 depicts the horizontal tail plotted in the AAA software.

Figure 5-7: Geometry of the Rainbow horizontal tail drawn in AAA

The Rainbow SCD coefficients obtained from the AAA software module for the longitudinal and lateral motion directions respectively are tabulated in Table 5-4. The table shows the forces (C_D, C_L, C_Y) and moments (C_l, C_m, C_n) coefficients with respects to the steady states; longitudinal non-dimensional states ($u, \alpha, \dot{\alpha}, q$, and δ_e) and lateral non-dimensional states ($\beta, \dot{\beta}, p, r, \delta_a$ and δ_r). For example, the partial derivative of the drag coefficients with respect to the elevator deflection, $c_{D\delta_e}$ ($= \frac{\partial C_D}{\partial \delta_e}$) is 0.0281. This value signifies that an increase of 1° in the elevator deflection will increase the total drag coefficient by 0.00040437 ($= 0.0281 \times \frac{1^\circ \times \pi}{180^\circ}$) and vice-versa.

Table 5-4: Rainbow aerodynamic coefficients obtained from AAA

Longitudinal Direction				Lateral Direction			
States	C_D	C_L	C_m	States	C_Y	C_l	C_n
SS*	0.0471	0.5819	-0.0112	β	-0.2314	-0.0707	0.0204
u	0.0883	0.0011	0.0005	$\dot{\beta}$	-0.0331	0.0001	-0.0136
α	0.2947	5.1909	-2.6345	p	-0.0442	-0.4738	-0.0710
$\dot{\alpha}$	0.0000	2.0504	-6.7518	r	0.1118	0.1770	-0.0616
q	0.0000	10.0121	-21.7260	δ_a	0.0000	0.0910	-0.0082
δ_e	0.0281	0.4864	-1.6017	δ_r	0.0212	0.0007	-0.0099

Notation: * SS indicates coefficients evaluated at the steady state

Linear Equations of Motion

The SCD coefficients obtained in Table 5-4 are substituted into the linearised state-space equation of motions for the decoupled longitudinal and lateral motion as given in equations (B-14) and (B-15) (sees Appendix B). The followings show the state-space equations for the longitudinal and lateral directions respectively.

$$\begin{pmatrix} \dot{u} \\ \dot{\Delta\alpha} \\ \dot{q} \\ \dot{\Delta\theta} \end{pmatrix} = \begin{bmatrix} -0.2003 & 0.3153 & 0 & -0.6388 \\ -1.2595 & -5.6636 & 2.8454 & 0 \\ 3.8461 & -155.8465 & -12.5090 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} u \\ \Delta\alpha \\ q \\ \Delta\theta \end{pmatrix} + \begin{bmatrix} -0.0309 \\ -0.5259 \\ -103.5875 \\ 0 \end{bmatrix} \Delta\delta_e \tag{5-6}$$

where, $u = \frac{\Delta U}{U_0}, \quad q = \frac{d\theta}{dt}$

$$\begin{pmatrix} \dot{\Delta\beta} \\ \dot{\Delta\phi} \\ \dot{p} \\ \dot{\Delta\psi} \\ \dot{r} \end{pmatrix} = \begin{bmatrix} -0.2536 & 0.6376 & -0.0026 & 0 & -0.9915 \\ 0 & 0 & 1 & 0 & 0 \\ -62.0071 & -0.0077 & -22.4487 & 0 & 8.2832 \\ 0 & 0 & 0 & 0 & 1 \\ 5.6349 & -0.1665 & -1.9884 & 0 & -0.6881 \end{bmatrix} \begin{pmatrix} \Delta\beta \\ \Delta\phi \\ p \\ \Delta\psi \\ r \end{pmatrix} + \begin{bmatrix} 0 & 0.0233 \\ 0 & 0 \\ 80.2642 & 0.3642 \\ 0 & 0 \\ -0.6815 & -3.5561 \end{bmatrix} \begin{pmatrix} \Delta\delta_a \\ \Delta\delta_r \end{pmatrix} \tag{5-7}$$

where, $p = \frac{d\phi}{dt}, \quad r = \frac{d\psi}{dt}$

5.1.3 PID Gains Determination for Rainbow SUAV

The *motion and guidance control* loops described earlier in Section 5.1.1 are used in the level flight condition and they are available in the MicroPilot autopilot module. In Table 5-5, a summary of these behaviors are given together with the corresponding name of the feedback loops implemented in the MicroPilot autopilot module. The controllers were designed based on single-input-single-output (SISO) methods, also known as the classical control methods [43, 64].

Table 5-5: MicroPilot autopilot behaviors in level-flight condition

Behavior	Description	Loops in MicroPilot
Altitude hold	Throttle setting is used to control the altitude by minimizing the error between the actual altitude and the desired altitude.	Throttle from altitude
Airspeed hold	Pitch angle is used to control the airspeed by minimizing the error between the actual airspeed and the desired airspeed.	Pitch from airspeed
Heading following	Roll angle is used to control the heading by minimizing the error between the actual heading and the desired heading.	Roll from heading
Pitch-hold	Elevator is used to control the pitch angle by minimizing the error between the actual pitch and the desired pitch. The trailing edge of the elevator will be deflected upwards when the error is positive and vice-versa.	Elevator from pitch
Roll-hold	Aileron is used to control the roll angle by minimizing the error between the actual roll and the desired roll. The left aileron will be deflected downwards when the error is positive and vice-versa.	Aileron from roll

The transfer functions corresponding to each of the SISO feedback control loops discussed in Table 5-5 are derived by applying Laplace Transformation to the state-space equations (5-6) and (5-7). The state space mathematical model for the Rainbow SUAV was written in MatlabTM M-files and the transfer functions were obtained from the MatlabTM software. Table 5-6 shows the transfer functions derived for the *airspeed hold*, *heading following*, *pitch hold* and *roll hold* feedback

loops. The *altitude hold* transfer function was not derived as the dynamic thrust data for the Rainbow propulsion subsystem was not available. The PID gains implemented on the *altitude hold* control loop were tuned using experimental approach, which will be discussed later in Section 5.2.

Table 5-6: Transfer functions for the Rainbow platform in level flight condition

Control loop	Transfer Function
Airspeed due to pitch angle	$\frac{u(s)}{\Delta\theta(s)} = \frac{0.0309s^3 + 0.7273s^2 + 44.71s - 322.4}{103.6s^2 + 525.6s + 149.6} \quad (5-8)$
Heading due to roll angle	$\frac{\psi(s)}{\Delta\phi(s)} = \frac{0.654}{s} \quad (5-9)$
Pitch due to elevator	$\frac{\Delta\theta(s)}{\Delta\delta_e(s)} = \frac{-(103.6s^2 \mp 525.6s \mp 149.6)}{s^4 + 18.37s^3 + 518.3s^2 + 107s + 139.3} \quad (5-10)$
Roll due to aileron	$\frac{\Delta\phi(s)}{\Delta\delta_a(s)} = \frac{80.26s^2 + 69.94s + 419.1}{s^4 + 23.39s^3 + 43.22s^2 + 296.7s + 8.074} \quad (5-11)$

The control algorithm used to compute the output signal of the feedback loops implemented in the autopilot module is based on the linear PID (proportional, integral and derivative) controller, $C(s)$ as given in the equation below (in Laplace's domain).

$$C(s) = K_P + \frac{K_I}{s} + K_D s \quad (5-12)$$

The aim of designing the controller is to obtain a bulk quantitative values of these three gains of the controller, i.e. the proportional gain, K_P , the integral gain, K_I and the derivative gain, K_D . The bulk value provide a good starting point for the flight test to avoid instability and unpredicted performances, such as large oscillation. The effects of the gains to the output response are summarized in Table 5-7.

Table 5-7: Effect of the increase in the gain on the output response

Gains	Effect on the output response when the gain is increased
Proportional, K_P	Accelerate the response. However, high proportional gain may cause overshoot due to over correction.
Integral, K_I	Accelerate the response and reduce the steady-state error. However, a high integral gain will result in instability.
Derivative, K_D	Reduce overshoot and oscillation. However, high derivative gain will cause the response to becomes sluggish.

SISOTOOL [97], is used to assist in the PID gains tuning process for the four feedback loops given in Table 5-6. The *aileron from roll* and *roll from heading* loops are used as examples to illustrate the process to tune the controller in achieving *heading-following* behavior. Figure 5-8a shows the schematic feedback loops for the *aileron from roll*, which is also the inner loop of the heading hold behavior. Figure 5-8b shows that the *closed-loop of the roll-hold* is cascaded into the outer loop of the *heading-hold* behavior.

Figure 5-8: Cascaded control loop for the heading-following behavior

Tuning the Aileron from Roll Feedback Loop

The default PID gains provided by the manufacturer for the *aileron from roll* loop ($K_P = 1.557$, $K_I = 0.006$, $K_D = 0.575$) were firstly investigated by analyzing the performance of the closed-loop step input response [44, 55, 101]. Three controllers, the P, PI and PID controllers were investigated as shown in Figure 5-9. Three parameters, i.e. the *settling time*, *rise time* and *steady-state error*, which are commonly used for evaluating step response, were also included in the same figure.

Figure 5-9: System performance comparison among the P, PI and PID controllers

The results showed that feedback loops with P (red line) and PI (turquoise line) controllers have similar responses. This is due to the small integral gain implemented. However, the *steady-state error* of the unit step response with PI controller was reduced. Adding a derivative gain to the feedback loop slows the response as both the *rise time* and *settling time* increased 46% and 12% respectively.

$$\phi(t) = \phi_1(t) + \phi_2(t) + \phi_3(t)$$

Where,

$$\begin{aligned}\phi_1(t) &= e^{-10.411t}[-1.6147 \cos(0.516t) - 27.654 \sin(0.516t)] \\ \phi_2(t) &= e^{-1.284t}[0.6270 \cos(2.105t) - 0.8318 \sin(2.105t)] \\ \phi_3(t) &= 0.9878\end{aligned}\tag{5-13}$$

Oscillation was observed from the responses of the loop with P and PI controllers between $t=0s$ and $t=2s$. By taking the inverse Laplace transformation of the closed-loop feedback with a step input, the equation for the roll response in time domain was derived as shown in equation (5-13). The equation consists of three main components, $\phi_1(t)$, $\phi_2(t)$ and $\phi_3(t)$. The first two components are due to the fourth order transfer function and the third component is due to the step input. Since, the first component is always negative and the last component is a constant, the oscillation was caused by the second component. By taking the first derivative of the second component, a local minimum can be found at time near 0.7s. This local minimum contributed by the second component caused the total response to reduce instead of increase.

The oscillation observed could be eliminated by adding a derivative controller. The response of the loop with PID controller, as depicted with purple line showed that the oscillation was removed. However, as derivative controller increase the response time, the response was observed to be more sluggish.

The results as depicted in Figure 5-9 show that the PID gains investigated were suitable for the *aileron from roll* feedback loop as the controller guarantees stability and provide quick response with small *steady state error*. Thus, no further gains tuning were required.

Investigating the Effect of the PID Gains

Figure 5-10: Unit step response for the *roll-hold* feedback loop

To understand the effects of the PID gains, following study was performed using the *aileron from roll* feedback loop. Figure 5-10 shows the unit step responses of the loop with four controllers using different sets of PID gains. The red line shows the response of the loop with the same PID gains tabulated in Table 5-8 for the *aileron from roll* loop. It could be observed from the responses drawn with purple and turquoise lines that the responses improved or became more sluggish when the proportional gain was increased or decreased respectively. The investigation also showed that by doubling or halving the proportional gains, the responses still guarantees stability. Referring to the same figure, the blue line provides an example of inappropriate gains selection, as instability (oscillation) was induced. The proportional gain was intentionally increased by tenfold and the integral gain was doubled. The response showed that the SUAV drastically rolled to left and back to right and back to left before settling down.

Tuning the Roll from Heading Feedback Loop

Figure 5-11: Unit step response for the heading hold behavior

Next, the response of the outer loop of the heading-hold behavior was investigated. Similarly, the default gain for the *roll from heading* feedback loop ($K_p = 0.171$, $K_i = 0$, $K_d = 1.639$) was studied first. The step responses of the *heading-hold* behavior are plotted in Figure 5-11. The red line shows the response of the feedback loop with the default PID gains. The response was a little sluggish as it took approximately 70 seconds to settle down. The proportional gain was firstly increased by a small percentage i.e. by 20% and it was observed that the response (in turquoise line) improved a little. Then, the proportional gain was doubled and the response observed was still sluggish as depicted in purple line. The proportional gain was further increased by 50% and the performance was now acceptable (as shown in blue line). The response overshoots when the proportional gain was further doubled as indicated with pink line. Based on the results, the PID gains for the blue line response were selected as the final results for the implementation. The results showed that the time response was approximately 13 seconds and no steady state error or overshooting was observed. This is acceptable for a fixed-wing SUAV as a drastic change in the heading direction is not desirable.

The discussion on the tuning of the PID gains using the *heading-following* and *roll-hold* behaviors as examples, showed how the PID gains were selected by analyzing the step input response. In general, the step response of the feedback with the controller (P, PI or PID) is investigated by firstly using the default gains provided by the manufacturer. If the response is not satisfactory, the proportional gain will be firstly adjusted to improve the response time. This is followed by adding integral gain and derivative gain to reduce the *steady state error* and oscillation respectively. The default gains provide a base for the tuning of the gains. By using the default gains as the initial gains for investigation, the possibility of the final gains selected is out of the implementation range (either too large or too small), can be avoided.

A similar approach was taken for the *elevator from pitch* and *airspeed from pitch* feedback loops and Table 5-8 tabulates all the PID gains tuned and selected for all the respective loops. The PID gains were determined in SI units and were then converted to the units defined by the MicroPilot manufacturer. The conversion ratios are given in Appendix F.

Table 5-8: PID gains determined using the analytical approach

Feedback Loops	SI Unit			MicroPilot Unit		
	K_P	K_I	K_D	K_P	K_I	K_D
Aileron from roll	1.557	0.006	0.575	-130000	-128	-3000
Elevator from pitch	-2.492	-0.880	-0.213	26000	9800	8900
Pitch from airspeed	-0.043	-3.755×10^{-4}	-2.428×10^{-3}	13756	24	194
Roll from heading	0.615	0.000	1.639	-900	0	-300

5.1.4 Effects of CG Location on the SUAV Responses

The CG of the airframe affects the static stability and dynamic performance of the fixed-wing SUAV [64]. This section studies the sensitivity of the SUAV performance to the change in the CG location. The horizontal CG (along the x -axis) location of the Rainbow SUAV were virtually moved 5% towards the head, as depicted in Figure 5-12. Based on the new CG location, the SCD were re-evaluated to obtain a new mathematical model. The same was repeated by virtually moving the CG towards the tail from the original location by 5%. The estimated SCDs are tabulated in Table C-1 and Table C-2 in Appendix C.

The following terms are defined to differentiate the three mathematical models derived based on the three CG locations:

- i. CG_Original refers to the mathematical model without shifting the CG.
- ii. CG_Front refers to the mathematical model with the new CG location moved towards the SUAV nose.
- iii. CG_Behind refers to the mathematical model with the new CG location moved towards the SUAV tail.

The longitudinal stability coefficients obtained for the above three different configurations (refers to Table 5-9) are negative in values. The Rainbow SUAV is statically stable in the longitudinal direction for all the three configurations.

Figure 5-12: Movement of the CG locatoin

Firstly, the open loop response of the SUAV for the *elevator from pitch* loop was investigated. Figure 5-13 shows the unit step response of the loop. In the response, one degree of elevator deflection in the positive direction was simulated as the input and the output response, i.e. the pitch angle was plotted against time.

The response for the CG_Original is plotted in green. The red line shows the response for the CG_Behind and the blue line depicts the response for the CG_InFront. The results showed that a positive elevator deflection resulted in a negative pitching moment, consistent with the discussion presented in Figure 5-4.

Figure 5-13: Open loop step input response

Comparing the response of the CG_Behind (red line) and CG_InFront (blue line) with the CG_Original (green line), similar pattern of response was observed but with different steady state values. When the CG was moved towards the fore, the SUAV pitch angle becomes less sensitive to the elevator deflection. This was because the steady state value of the CG_InFront was lower than CG_Original. When the CG location was nearer to the nose, the SUAV has a higher tendency to pitch down. This could be explained using the longitudinal stability coefficients, C_{m_α} as given in Table 5-9. The coefficient became more negative when the CG was nearer to the fore. As a result, a larger elevator deflection will be required by the CG_InFront to create the same pitching moment generated by the CG_Original, since the elevator control derivative, $C_{m_{\delta_e}}$, for CG_InFront is smaller than CG_Original. Having the CG too near to the head could result in dangerous consequence, especially during the take-off phase. The deflection of the elevator may not be sufficient to create a counter moment against the pitching moment generated by the CG, for it to pitch up.

Table 5-9: Longitudinal stability coefficients for different CG location

Coefficient	CG_InFront (5% In Front)	CG_Original	CG_Behind (5% Behind)
$C_{m_{\delta e}}$	-0.9988	-1.6017	-1.5649
$C_{m_{\alpha}}$	-3.0276	-2.6345	-2.2415

The investigation revealed that the open loop response for SUAV was very sensitive to the CG location. The sensitivity of the closed loop response will be now investigated. The same PID gains given in Table 5-8, were applied to the controller for all the three models. Step inputs were simulated and the step responses were shown in Figure 5-14. The responses for the CG_Behind (red line) and CG_Original (green line) were almost identical. Looking at the CG_InFront response (blue line), the same controller took a longer time to react to the given input, when the CG was shifted nearer to the nose. The settling time for the CG_InFront response is the longest among the three responses. Generally, the results showed that the closed-loop response was less sensitive to the CG location.

Figure 5-14: Closed-loop response for the elevator from pitch feedback loop

5.2 Experimental Approach

The experimental or the ‘*test, tune, and observe*’ approach to tune the controller gain will be discussed in this subsection. The feedback control loop of interest is the *throttle from altitude* loop. Figure 5-15 depicts the *throttle from altitude* closed-loop. Referring to the figure, the loop takes the desired altitude as the control input and uses the actual altitude as the negative feedback. Based on the error between the desired altitude and actual altitude, the PID controller computes the throttle power to the propulsion subsystem for correcting the error detected.

Figure 5-15: *Throttle from altitude* feedback control loop

The flight ‘*test*’ was conducted as following:

- i. The Rainbow SUAV was hand-launched to take off.
- ii. In RC mode deployment, the ground-pilot climbed the SUAV to an altitude near 120m, which was the cruise altitude.
- iii. The autonomous mode was then triggered and the SUAV climbed to the desired altitude.
- iv. When it reached the desired altitude, the SUAV would then travel in a level flight condition.
- v. After sufficient data was collected, the SUAV would land in RC mode.
- vi. The flight data were retrieved for analysis.
- vii. Based on the flight data, the gains were then adjusted.

Four flight tests were conducted to tune the PID gains. Table 5-10 tabulates the gains (in units defined by the MicroPilot manufacturer) used for the tests. The PID gains used in the first flight test (Test #1) were the default gains recommended by the autopilot manufacturer.

Table 5-10: PID gains used in the flight test attempts for the *throttle from altitude* feedback control loop

Flight Test Attempt	PID Gains (in MicroPilot defined unit)		
	K_P	K_I	K_D
Test #1	2600	73	32
Test #2	1300	73	32
Test #3	650	73	32
Test #4	650	20	32

Figure 5-16 and Figure 5-17 shows the flight test results collected from the four tests. The x -axis indicates the time horizon and the left-hand side y -axis depicts the altitude axis in meter. Finally the right-hand side y -axis depicts the throttle setting in term of percentage. The desired altitude, actual altitude and throttle setting were plotted on the graph for analysis. The blue line shows the desired altitude, the green line shows the actual altitude and the red line shows the throttle setting. The plot was ‘*observed*’ if the result showed any of the possible responses; instability (oscillation), sluggishness of response, and overshooting.

The results from the first flight as shown in Figure 5-16a were investigated. Referring to the figure, within the timeframe of 110 seconds to 116 seconds, the SUAV altitude was below the desired altitude. A negative discrepancy between these two altitude values triggered the autopilot control loop to increase the throttle setting, which will increase the thrust to the SUAV. As a result, the SUAV started to climb to a higher altitude and overshoot to approximately 140 meter in 120 seconds. When the SUAV altitude overshoot 120m and continued to climb to a higher altitude, there was a positive difference between the actual and desired altitude. This situation was observed in the timeframe from 116 seconds to 126 seconds. During this time frame, the controller reversed the initial response, i.e. the throttle setting was decreased to reduce the SUAV altitude.

Figure 5-16: Flight tests results for Test #1 and Test #2

Figure 5-17: Flight tests results for Test #3 and Test #4

By increasing and decreasing the throttle setting, the controller will be capable of maintaining the SUAV at a certain altitude. From the same figure (see Figure 5-16a), the throttle setting was increased to the maximum (100%) and saturation was observed at this value before decreasing back to the minimum (0%), when the SUAV altitude was below and above the desired altitude respectively. The oscillation was observed for a few cycles. The response gave a cue that the proportional gain assigned was too high. A small error between the desired altitude and actual altitude yielded a large increase or decrease in the throttle setting. The drastic oscillation of the throttle setting resulted in an oscillation of the SUAV altitude around the desired altitude (within ± 20 meter from 120 meter).

The first flight test result showed that if a set of gains was arbitrary selected, the gains may be too large and unsuitable to the SUAV. For example, should the similar oscillation of the throttle response be observed on the aileron response, the SUAV may have sudden roll which may end up flipping over the SUAV. Instability may occur and if the land-based pilot is inexperienced to handle the situation, the situation may be hazardous. Secondly, such oscillation may cause damages to the equipments. For example, having the throttle setting at its maximum for too long would burn the electronics speed controller that regulates the current from the battery to the motor.

The PID gains were '*tuned*' before the second flight test. Since oscillation in the SUAV altitude was observed, the proportional gain would be adjusted. The proportional gain was reduced by half from 2600 to 1300, as recommended by the manufacturer [51]. The integral and derivative gains were kept the same since it is advisable to only adjust one gain at one time. This is to ease the analysis of the response from the post-flight data. After tuning the gain, the same flight test was conducted and the results were plotted on Figure 5-16b. Analyzing the plot, similar response, i.e. oscillation was still observed. This indicated that, despite the 50% reduction in the proportional gain, the gain was still very high.

Before the third test, the proportional gain was further reduced by half to 650. Similarly, the derivative and integral gains were not changed. The flight results

were plotted in Figure 5-17a. The throttle settings and the altitude responses were still oscillating. However, the response had improved as the throttle setting did not escalate to the maximum value or plunge to its minimum value. Saturation of the throttle setting observed in the first two flight tests results did not occur in the third flight. This showed that by reducing the proportional gain, the oscillation in the throttle setting can be reduced or eliminated. Although the response was improved, the gains were still ‘unacceptable’ as oscillation was still observed.

Based on the observation from the third flight test results, the proportional gain was supposed to be further reduced by half for the fourth flight test. If the proportional gain was further reduced, the oscillation would be expected to be reduced. By reducing the proportional gain further, the throttle signals computed would be smaller for the same error detected.

However, due to the limited tests available, the proportional gain was not adjusted for the flight test. Instead, the integral gain was tuned for investigating the effect of the integral gain on the response. Since, the increase in the integral gain accelerates the response, an integral gain that is too large, may induce overshooting and cause oscillation. Thus, for the fourth flight test, the integral gain was reduced by one-third from 73 to 20. The flight results are plotted on Figure 5-17b. The figure shows that the throttle setting no longer oscillating and saturation of the throttle setting was also not observed. The SUAV was also observed to be flying at the altitude very close to the desired altitude. Although, discrepancy of the actual altitude from the desired altitude was observed from time 150s to 170s, the discrepancy might be caused by the wind condition during flight test.

Based on the last flight test, it was observed that the gains implemented on the fourth test were capable of controlling the throttle power to maintain the SUAV at a given altitude. Thus, it was concluded that the PID gains are now ‘acceptable’ and will be used for further flights.

5.3 Chapter Summary

This chapter discussed the approaches that had been taken to determine the PID gains for *the motion and guidance* controller. Considering the level flight condition, the controller for five feedback loops of control, i.e. the *roll from heading*, the *pitch from airspeed*, the *aileron from roll*, the *elevator from pitch*, and the *throttle from heading*, were obtained. In Section 5.1, a methodology to design the controller analytically was detailed. The methodology was demonstrated using the Rainbow SUAV to analytically determine all the feedback loops except for the *throttle from altitude*. This loop was tuned using the experimental approach, as presented in Section 5.2.

CHAPTER 6: EXPERIMENT AND EVALUATION

This chapter presents the experiments conducted. The deliberative behavior, i.e. the *go-to waypoint* behavior, was firstly investigated in Section 6.1 and Section 6.2. In Section 6.3, the *collision prevention & obstacle avoidance* reactive behavior was investigated in a flight test using the ECO-8 SUAV. A ground test was conducted, as presented in Section 6.4 to demonstrate the *object tracking behavior*. Finally, an experiment was conducted to validate the functionality of the graphical user interface and this experiment is detailed in Section 6.5.

6.1 Experiment E1: Rainbow SUAV Flight Test

This section explained the flight test conducted to demonstrate the capability of the SUAV to perform the deliberative *go-to waypoint* behavior using the Rainbow SUAV. Three objectives were aimed to be accomplished from this experiment. They were listed as following:

- E1-A. To demonstrate the capability of the Rainbow SUAV to navigate autonomously to designated waypoint locations as planned in the flight mission.
- E1-B. To verify the capability of the *motion and guidance* controller in tracking the commanded inputs. The aim was to investigate the successfulness of the autopilot in performing *heading-following*, *roll-hold*, *airspeed-hold*, and *pitch-hold* controls.
- E1-C. To verify the mathematical model of the Rainbow SUAV.

6.1.1 Procedure of Experiment E1

The followings describe the procedures for Experiment E1. The flight data collected from the experiment were used to evaluate objectives E1-A to E1-C.

Pre-Flight Session

Step 1: The cruise speed of the Rainbow SUAV was set to be 15 ms^{-1} in the parameters file (*.vrs).

- Step 2:* The PID gains for the feedback control loops given in Table 5-8 were set in the same parameters file.
- Step 3:* The autopilot was turned on, and the parameters file was uploaded to the autopilot via the GCS.
- Step 4:* The flight mission was planned using the *.fly file. Table 6-1 shows the commands in the file. Comments were included to explain the commands.

Table 6-1: Flight mission for E1 experiment

i.	<i>takeoff</i>	// To initiate take-off
ii.	<i>climb 120</i>	// To climb to an altitude of 120m
iii.	<i>flyTo (-200, 0)</i>	// To navigate to waypoint (-200,0)
iv.	<i>flyTo (-200, 250)</i>	// To navigate to waypoint (-200, 250)
v.	<i>flyTo (200, 250)</i>	// To navigate to waypoint (200, 250)
vi.	<i>flyTo (200, 0)</i>	// To navigate to waypoint (200, 0)
vii.	<i>repeat -4</i>	// To go 4 steps backwards (i.e. to step iii)
viii.	<i>circuit</i>	// To initiate an autonomous landing

Referring to the flight mission, eight commands were planned as shown in Table 6-1. These commands would be expected to run in a top-down approach, i.e. from command (i) to command (viii) sequentially. As shown in command (i) and command (ii), the UAV was commanded to take-off and climbed to an altitude of 120m. Upon reaching the desired altitude, the UAV would navigate to the desired waypoint locations at a cruise speed of 15ms^{-1} . In this test, four waypoint locations were pre-loaded at point (-200,0), (-200, 250), (200, 250), and (200,0), as shown in command (iii) to command (vi). The waypoint location is set in the (x, y) coordinate point. The 'x' is referring to the relative distance of the waypoint to the north of the origin point and 'y' is the relative distance of the waypoint to the east of the original point. The original point will be the location where the UAV successfully received the GPS signals. The *repeat* command as shown in command (vii), would instruct the autopilot to repeat the commands from command (iii) onwards. Thus,

the *circuit* command for autonomous landing, as given in command (viii) would never be executed.

Step 5: The flight mission file was then uploaded to the autopilot via the GCS.

Step 6: After that, the SUAV was placed at the location, where it was intended to be taken-off.

Step 7: The autopilot was then restarted.

Step 8: When the autopilot was turned on, the module will automatically begin to receive signals from the GPS.

Step 9: The status of the GPS signals was monitored via the GCS.

Step 10: After the GPS signals were successfully locked by the autopilot module, a pre-flight check was conducted. The pre-flight checklist is included in Appendix F. After passing the pre-flight check, the SUAV was ready for the flight test.

In-Flight Session

Step 11: The Rainbow SUAV was hand-launched to take off, as shown in Figure 6-1. The ground-pilot controlled the SUAV in the RC mode.

Figure 6-1: The Rainbow was hand-launched to take-off

- Step 12:* After the SUAV took-off, the ground-pilot continued to command the SUAV to climb to a safe altitude, at approximately 100m above the ground.
- Step 13:* Then, the autonomous mode was triggered by toggling one of the switches on the RC transmitter. For the experiment, Switch-5 had been pre-assigned as the toggle switch.
- Step 14:* After the autonomous mode was triggered, commands from the RC transmitter from the RC transmitter would not be executed. The autopilot would be in control and the SUAV would perform self-correct and self-navigation to the desired waypoint locations.
- Step 15:* During the flight, the autopilot logged the flight states, sensor data and control inputs, at a rate of 5Hz. Table 6-2 tabulates the flight data that was logged and required for Experiment E1

Table 6-2: Flight data required for Experiment E1

States	Symbols
SUAV Relative Locations	X_{act}, Y_{act}
Actual Attitude (roll and pitch)	ϕ_{act}, θ_{act}
Actual Airspeed	u_{act}
Actual Heading	ψ_{act}
Actual Altitude	h_{act}
Desired Attitude	ϕ_{des}, θ_{des}
Desired Airspeed	u_{des}
Desired Altitude	h_{des}
Desired Heading	ψ_{des}

- Step 16:* The ground-pilot continued to monitor the SUAV during the autonomous navigation. Monitoring was performed by observing the physical platform responses. This was done by observing if any of the followings occur: (a) oscillation about the SUAV body-frame axis, (b) a sudden drop in the altitude, (c) the SUAV flight path was not within the anticipated flight envelope, or (d) the SUAV was not performing the commands as planned in Table 6-1.

Step 17: If any of the responses mentioned in *Step 16* was observed, the ground-pilot would take-over the control and performs a landing.

Step 18: Otherwise, the SUAV would be expected to navigate to each of the four waypoints. After the SUAV navigated to all the four waypoints for two laps, the ground-pilot would begin to land the SUAV.

Step 19: Using the same toggle switch, i.e. the Switch-5, the control mode was toggled from the autonomous mode to the RC mode.

Step 20: The ground-pilot then descended the SUAV and landed it near the origin location.

Post-Flight Session

Step 21: After the flight, the data log saved in the autopilot was retrieved for evaluation. The results will be presented and discussed in Section 6.1.2, Section 6.1.3, and Section 6.1.4.

Step 22: Post-flight check was conducted to ensure that no damage was occurred during the flight test. The checklist for the post-flight check is given in Appendix G.

6.1.2 Investigating Rainbow Waypoint Navigation Capability

This section discussed the capability of the autopilot in controlling the Rainbow SUAV for autonomous waypoint navigation, i.e. to evaluate the E1-A objective. The relative location, (X_{act}, Y_{act}) and the actual altitude, h_{act} flight data collected from Experiment E1 were used for the evaluation.

The results were presented in plots as given in Figure 6-2. The two-dimensional navigation path, i.e. the path taken by the SUAV during the flight test, is given in Figure 6-2a. Figure 6-2b includes the SUAV altitude to the navigation path to form a three-dimensional plot. The blue circles drawn on the Figure 6-2a indicate the waypoint locations pre-loaded into the autopilot.

Referring to Figure 6-2b, the SUAV took-off at the origin point, i.e. at (0, 0) coordinate. Then, the SUAV was maneuvered by the ground-pilot to climb to an

altitude of 100m in the RC mode. This is shown with a blue-line on the plots. Subsequently, the autonomous mode was triggered and the red line on the figure traced the path the SUAV travelled in this mode. After the autonomous mode was triggered, the autopilot detected the altitude of the SUAV was approximately 100m above ground. This can be observed from Figure 6-2b, by looking at the point where the RC take-off path (blue line) and the autonomous path (red line) met at Point-A. As the SUAV altitude was below the desired altitude of 120m, the autopilot controlled the SUAV to continue climbing, while navigating to the first waypoint at (-200, 0) location. After the SUAV reached the desired altitude at 120m, it can be observed from Figure 6-2b that the autopilot was successfully maintaining the SUAV at this targeted altitude, while travelling to different waypoint locations.

From Figure 6-2a, it can be observed from the autonomous path (red line) that the SUAV travelled to all the four waypoint locations. This showed that the SUAV has successfully navigated to each of the four waypoints autonomously and in the sequential order as discussed in the flight mission presented in Table 6-1. The result also showed that the SUAV repeated the navigation pattern after the SUAV reached the fourth waypoint for the first time. This can be observed from the plot that the autonomous path (red-line) formed two loops of navigation path to the same waypoint locations.

After reaching the fourth waypoint for the second time, the ground-pilot engaged the RC mode and landed the SUAV. The green line shows the descending path of the SUAV. It can be observed from Figure 6-2b that the SUAV altitude decreased after reaching the fourth waypoint. The SUAV was then landed near the original point.

The result presented in this subsection had shown that the high-level deliberative behavior was successfully accomplished. In the next subsection, detail investigation on the *motion and guidance* controller implemented to achieve this high-level task will be presented.

Figure 6-2: Navigation path taken by the Rainbow UAV in Experiment E1

6.1.3 Investigating the Motion and Guidance Controller

This subsection evaluates the EI-B objective. The discussion will be on the suitability of the PID gains implemented on the autopilot module in controlling the SUAV to track the desired states. Four feedback loops, i.e. the *aileron from roll*, the *roll from heading*, the *elevator from pitch*, and the *pitch from airspeed* feedback loops will be discussed.

Evaluating the Roll from Heading and the Aileron from Roll feedback loops

In order to evaluate the suitability of the PID gains implemented on the motion and guidance controller, the desired state, i.e. the control input to the feedback loop and the SUAV actual responses were compared.

The *roll from heading* and the *aileron from roll* feedback loops were first studied. These two loops form a nested control loop, as discussed earlier in Section 5.1.1. For the former feedback loop, the desired heading and the actual heading results were compared and plotted in Figure 6-3a. The blue line showed the desired heading, λ_{des} and the red line traced the actual heading λ_{act} of the SUAV measured from the digital compass. On the other hand, Figure 6-3b plots the time history of the SUAV roll angle during the flight. The blue line showed the desired roll angle, ϕ_{des} and the red line traced the actual roll angle, ϕ_{act} . Table 6-3 summarizes the data plotted on the figures.

Table 6-3: Summary of the data plotted for directional control results

Control Loops	Control Inputs (Blue Line)	Output Responses (Red Line)	Figure
<i>Heading-following</i>	Desired heading	Actual heading	Figure 6-3a
<i>Roll-hold</i>	Desired roll angle	Actual roll angle	Figure 6-3b

Results plotted on Figure 6-2a, Figure 6-3a and Figure 6-3b would be used for the discussion. The discussion was zoomed into the time frame from $t=5s$ to $t=90s$, where the SUAV was in the first lap of the autonomous waypoint navigation. The results were segmented and a diamond-icon was used to mark the beginning of each segment.

Figure 6-3: Directional control flight test results

Navigating from Point-A to Point-B

After the autonomous mode was triggered, the autopilot was responsible to navigate the Rainbow SUAV to the first waypoint at (-200, 0) location. In Figure 6-3a, Point A on plot marked the instance when the autonomous mode was

triggered by the ground-pilot. At this moment, the SUAV heading was pointing at 272° (refers to the actual heading at Point-A in Figure 6-3a) and the first waypoint was located exactly in front of the SUAV (refers to Figure 6-2a). Thus, the desired heading was set to be approximately 270° . However, from Point A to Point B, instead of navigating towards the first waypoint, the SUAV was observed to be navigating away from it (sees Figure 6-2a). The inability of the heading-hold controller to track the desired heading once the autonomous mode was triggered may be due to the sluggishness in the inner-loop control, i.e. the *roll-hold* control loop. Examining Figure 6-3b, from Point-A to Point-B, the SUAV actual roll angle was lagging behind the desired roll commanded. The delay in the roll-hold control loop caused the delay in the outer loop that tracked the desired heading commanded.

Navigating from Point B to Point-C

However, after reaching Point-B (sees Figure 6-3b), the SUAV began to response to the commanded roll angle and rolled to the right, in order to direct the SUAV to the intended heading. As a result, the desired heading during this duration was increased in order to command the SUAV to navigate towards the first waypoint (sees Point-B in Figure 6-2a and Figure 6-3a). After Point-B, the SUAV travelled towards the intended direction and reach the first waypoint at Point-C (sees Figure 6-2a).

Navigating from Point-C to Point-D

After reaching the first waypoint, the SUAV was commanded to navigate to the next waypoint at (-200, 300). At this moment, the second waypoint was located to the north of the SUAV. The desired heading now was increased to 360° (sees Figure 6-3a). When traveling to the second waypoint, the SUAV was oscillating along the straight path with 0° heading. This can be seen on the navigation path as shown in Figure 6-2a, when the SUAV was navigating from the first waypoint to the second waypoint.

Navigating from Point-D to Point-H

After reaching the second waypoint at Point-E (sees Figure 6-2a), the desired heading was changed to 90° (sees Point-E in Figure 6-3a) and the SUAV continued to navigate to the third waypoint at Point-F. Observation on Figure 6-3a between Point-E and Point-F, showed that the actual heading was lagging the desired heading. When the actual heading was lagging behind the desired heading, it can be observed from Figure 6-2a that, the path taken by the SUAV was longer in distance. After reaching third waypoint at Point-F, the SUAV continued to navigate to the fourth waypoint at Point-G and the entire navigation pattern was repeated for the second lap and the autonomous navigation ended at Point-H. After Point-H, The ground-pilot took over the control and landed the SUAV.

Based on the result from Figure 6-3a, it can be observed that the actual heading did not follow the commanded heading accurately. The desired heading computed was aimed to command the SUAV to travel in the direction that will reach the waypoint in the shortest path. Consequently, the path taken by the SUAV during the test was relatively longer in distance. Nevertheless, the results had successfully showed that the *heading-following* behavior has been achieved with small bound of tracking error.

From Figure 6-3b, the desired roll computed was actually moving cyclically. Since the *roll-hold* is the inner loop of the nested loop for the directional control, the desired roll was computed by the outer loop, i.e. the *heading-following*. Thus, when the actual heading was lagging the desired heading, the SUAV would be commanded to roll to the right (i.e. positive angle). For example, this was observed during the navigation from Point-A to Point-B. In contrast, when the actual heading was leading the desired heading, the SUAV would be commanded to roll to the left (i.e. negative angle). The plot shows that, in most of the time, the SUAV was commanded to roll to the right. This was because the SUAV was completing the four waypoints navigation in Figure 6-3a clockwise direction. In conclusion, the results had successfully demonstrated the *roll-hold* behavior, where the SUAV managed to track the pattern of the commanded roll angle.

Evaluating Pitch from Airspeed and Elevator from Pitch

The capability of the autopilot to maintain the cruise speed at the desired speed will now be discussed. The airspeed control engaged the *airspeed-hold* and *pitch-hold* feedback loops. Figure 6-4a shows the time history of the airspeed. The blue line shows the desired airspeed, u_{des} and the red line shows the actual airspeed, u_{act} . Figure 6-4b shows the time history of the pitch angle. Similarly, blue line traces the desired pitch attitude, θ_{des} and red line traces the actual pitch, θ_{act} attitude. Table 6-4 summarizes the flight data plotted on Figure 6-4.

Table 6-4: Summary of the data plotted for airspeed control results

Control Loops	Control Inputs (Blue Line)	Output Responses (Red Line)	Figure
<i>Airspeed hold</i>	Desired heading	Actual heading	Figure 6-4a
<i>Pitch hold</i>	Desired roll angle	Actual roll angle	Figure 6-4b

From Figure 6-4a, the desired airspeed was initially set to 17ms^{-1} at Point-P. At this point, the SUAV was still climbing, thus a higher than the desired cruise speed of 15ms^{-1} was commanded. After reaching Point-Q, the level flight condition was achieved and the desired airspeed was then fixed at 15ms^{-1} , throughout the level flight as shown with a blue straight line on Figure 6-4a.

Results on Figure 6-4a shows that the airspeed of the SUAV did not successfully track the commanded airspeed. Based on the control loop, 15ms^{-1} was the reference airspeed and the SUAV was expected to travel within this airspeed. If the airspeed was above 15ms^{-1} , a positive desired pitch would be expected and if the airspeed was below 15ms^{-1} , negative pitch angle would be expected. However, the result depicted on the graph shows that desired pitch computed by the autopilot was not consistent with the expectation.

Figure 6-4: Airspeed control

To illustrate, refers to Figure 6-4 for the time frame between $t=20s$ and $t=40s$. Looking at Point-R in Figure 6-4a, the actual airspeed was approximately $9ms^{-1}$ and was far below the desired $15ms^{-1}$. Thus, the desired pitch was computed to be -7° , which was a negative value as shown in Figure 6-4b. When the SUAV pitch angle was reduced, the kinetic energy will be increased, thus elevated the airspeed, which was observed from Point-R to Point-S (sees Figure 6-4a). At Point-S, the

airspeed was 12ms^{-1} , which was still below the desired 15ms^{-1} . Thus, it was expected, the desired pitch angle computed should continue to be a negative value, in order to increase the airspeed. However, the desired pitch angle became positive (i.e. pitching upwards) after Point-S. As a result, instead of increasing the airspeed from 12ms^{-1} to 15ms^{-1} , the airspeed began to reduce after Point-S (see Figure 6-4a).

Investigating the Pitch from Airspeed feedback loop

To investigate further, the desired airspeed, u_{des} and actual airspeed, u_{act} data collected from the flight test were used to compute the desired pitch angle. The difference between the desired and actual airspeed was used as the control input to the outer loop PID controller to compute the desired pitch angle. Matlab/Simulink was used to facilitate the computational, as depicted in Figure 6-5. To avoid confusion, the following definitions are adopted;

- The desired pitch angle computed by the Matlab/Simulink model is termed as “Model Desired Pitch”
- The desired pitch angle computed by the autopilot and was retrieved from the flight test data is termed as “Actual Desired Pitch”.

Figure 6-5: Matlab/Simulink model to compute Model Desired Pitch

The desired pitch angle computed from the Matlab/Simulink model was plotted with blue line in Figure 6-6. The desired pitch angle computed by the autopilot and was retrieved from the flight test data was plotted with red line. Comparing the Model Desired Pitch with the Actual Desired Pitch, it was observed that both lines share the same trend. However, the Actual Desired Pitch was shifted upwards for 10° . One possible explanation to this condition is that the SUAV was not initialized properly. It might have been initialized in a nose up position and the initial 10° pitch angle was taken by the autopilot module as the level flight reference angle.

Figure 6-6: Investigation on the discrepancy in the desired pitch computed

A positive 10° shift caused the Actual Desired Pitch to be higher than the expected value. Referring back to the discussion at Point-S, the expected desired pitch should be -10° (refers to Point-S on Figure 6-6). However, due to the positive shift, the commanded desired pitch became 0° and this explained the scenario where the SUAV was commanded to pitch up even though the SUAV airspeed was still below the desired airspeed.

Results had shown that the *airspeed-hold behavior* was not successfully achieved since the airspeed was not maintained at the desired 15ms^{-1} airspeed due to the initialization issue. However, examining the result as shown in Figure 6-4b, the *pitch-hold* behavior was successfully accomplished. It can be observed that the actual pitch track the commanded pitch accurately.

6.1.4 Verification of the Mathematical Model

In this subsection, the E1-C objective will be evaluated. Flight test data collected were used to verify the mathematical models of the feedback loops. The desired states data collected from the flight test were used as the control inputs to the feedback control loops derived and given in Table 5-6 (see Section 5.1.3). The output responses of each of the feedback loops were simulated. The numerical simulation responses were then compared to the SUAV actual responses.

Figure 6-7: Simulated heading angle response

For example, referring to Figure 6-7, the desired heading from the flight data was the input to the directional-control nested loops. The output response, i.e. the heading angle, was simulated using the mathematical model derived in equation (5-9). The simulated heading angle response is plotted in green line on Figure 6-8a. The actual heading angle reading collected from the flight data is plotted on the same figure in red line, to facilitate comparison.

Similar tasks were performed to the other three feedback loops, i.e. the *roll-hold*, the *airspeed-hold* and the *pitch-hold*, in order to obtain the respective simulated responses. Table 6-5 tabulates the control input that was used for each of the feedback loop and the corresponding output responses. The results were then plotted on Figure 6-8 and Figure 6-9. In Figure 6-8b, the comparison between the simulated *roll hold* response and actual response is presented. The *airspeed hold* responses are depicted in Figure 6-9a and the *pitch hold* responses are as shown in Figure 6-9b.

Figure 6-8: Comparison between simulation results and flight results for directional control

Figure 6-9: Comparison between simulation results and flight results for airspeed control

Table 6-5: Control inputs and output responses from each feedback loop

Control Loops	Control Inputs (Flight data)	Output Responses		Figure
		Green Line (From Model)	Red Line (Flight Data)	
Heading hold	Desired heading	Simulated heading response	Actual heading	Figure 6-8a
Roll hold	Desired roll	Simulated roll response	Actual roll	Figure 6-8b
Airspeed hold	Desired airspeed	Simulated airspeed response	Actual airspeed	Figure 6-9a
Pitch hold	Desired pitch	Simulated pitch response	Actual pitch	Figure 6-9b

The results as depicted in Figure 6-8 and Figure 6-9 showed that the simulated responses are close to the actual responses for four of the feedback loops. Both the simulated and actual responses share the same trend with small discrepancy between the two responses. Among the four loops, the *pitch-hold* loop simulated pitch angle was the closest to the actual pitch angle response. Consistency of the mathematical model responses with the actual responses showed that in a closed-loop system, the mathematical model derived could be used to facilitate offline tuning to determine a set of gains that is closed enough to the actual system.

6.2 Experiment E2: ECO-8 SUAV Flight Test

This section presented the flight test conducted for the ECO-8 rotating-wing SUAV. The objective of this Experiment was to demonstrate the deliberative *go-to waypoint* behavior using a rotating-wing SUAV. The autonomous take-off, waypoint navigation and autonomous landing capabilities would also be demonstrated.

6.2.1 Procedure of Experiment E2

The followings outline the procedures for Experiment E2.

Pre-Flight Session

- Step 1: Repeat Step 1 described in Section 6.1.1, but the cruise speed was set to be 5ms^{-1} .
- Step 2: Repeat Step 2 described in Section 6.1.1 but using the PID gains in Table E-1 (sees Appendix E)
- Step 3: Repeat Step 3 described in Section 6.1.1.
- Step 4: The flight mission planned in the *.fly file was given in Table 6-6.

Table 6-6: Flight mission for Experiment E2

i.	takeoff	// To initiate take-off
ii.	climb 15	// To climb to an altitude of 15m
iii.	flyTo (0,90)	// To navigate to waypoint (0, 90)
iv.	flyTo (-90, 90)	// To navigate to waypoint (-90, 90)
v.	flyTo (-90, 0)	// To navigate to waypoint (-90, 0)
vi.	flyTo (0, 0)	// To navigate to waypoint (0, 0)
vii.	circuit	// To initiate an autonomous landing

As shown in Table 6-6, seven commands were uploaded to the autopilot. As shown in command (i) and command (ii), the SUAV was commanded to take-off and climbed to an altitude of 15m. Four waypoints were planned sequentially, at (0, 90), (-90, 90), (-90, 0) and finally at (0, 0) as shown in command (iii) to command (vi). At the final waypoint, an autonomous landing command would be initiated as shown in command (vii).

Step 5 to Step 10 described in Section 6.1.1 were then repeated.

In-Flight Session

Step 11: Autonomous take-off commands were initiated by the ground-pilot via the GCS. After that, the SUAV would be expected to continue to perform the flight mission in Table 6-6 autonomously.

Step 12: During the flight, the autopilot logged the flight states, sensor data and control inputs, at a rate of 5 Hz. Table 6-7 tabulates the data that was logged and required for the E2 experiment.

Step 13:

Table 6-7: Flight data required for E2 experiment

States	Symbols
SUAV Relative Locations	X_{act}, Y_{act}
Actual Altitude	h_{act}
Desired Altitude	h_{des}

Step 14: Repeat Step 16 described in Section 6.1.1.

Step 15: Repeat Step 17 described in Section 6.1.1. If the flight test was performing as the intended flight mission, the SUAV should land at the take-off point autonomously.

Post-Flight Session

Step 16: Repeat Step 21 described in Section 6.1.1.

Step 17: Repeat Step 22 described in Section 6.1.1.

6.2.2 Investigating ECO-8Autonomous Flight Capability

The blade of the ECO-8 SUAV began to spin after the “take-off” command was initiated by the ground-pilot via the GCS. The autopilot increased the throttle power to almost maximum at the point when the SUAV was almost taken off. Figure 6-10 shows the pictures of the SUAV when it was taking off autonomously.

Figure 6-10: ECO-8 autonomous take-off

The SUAV relative locations and the altitude data collected from the flight test were plotted on Figure 6-11. The two-dimensional navigation plot is shown in Figure 6-11a and the three-dimensional plot is given in Figure 6-11b. The waypoints locations were indicated with blue-circles (sees Figure 6-11a).

The SUAV climbed vertically upwards to an altitude of 15m after it took-off from the ground (Figure 6-11b). After reaching 15m, the SUAV began to navigate to the first waypoint at (0, 90). After reaching the first waypoint, the SUAV was observed hovering at the first waypoint, as intended in the flight plan. The SUAV heading was changed to 270° to face the second waypoint. After that, the SUAV continued to navigate to the second waypoint in a straight path. Similar response was observed where the SUAV hovered at the second waypoint location while changing its heading to face the third waypoint. Then it began to navigate to the third waypoint. The same responses were observed when the SUAV reached the third and fourth waypoints. After reaching the fourth waypoint, the SUAV began to perform autonomous landing. The altitude was observed to be decreasing over time as shown on Figure 6-11b. When the SUAV reached the ground, the throttle was automatically cut-off.

Figure 6-11: Navigation path taken by the ECO-8 SUAV in Experiment E2

6.2.3 Discussion

The results showed that the autopilot had executed the commands planned in the flight mission in the intended sequence. The ECO-8 SUAV took-off autonomously and navigated to all the pre-loaded waypoints. Finally, the SUAV landed at the designated location autonomously. The entire flight test did not require any intervention from the ground-pilot.

6.3 Experiment E3: Reactive Behavior Flight Test

This section discusses the flight test conducted to demonstrate the capability of the SUAV to perform higher-level reactive behaviors. In this test, the *collision prevention and obstacle avoidance* behavior would be demonstrated. The objectives of the flight test were given as following:

- E3-A. To demonstrate that the detection subsystem (i.e. the laser sensor) is capable of querying the external sensor data and stored it into the shared memory.
- E3-B. To demonstrate that the reactive behavior (the *collision prevention and obstacle avoidance*) could be triggered by external stimuli.
- E3-C. To validate that the *Navigational Reactive Behavior Controller* implemented is capable of controlling the SUAV when the reactive behavior has been triggered.
- E3-D. To demonstrate that the SUAV with limited processing power is capable of performing frontal *collision prevention and obstacle avoidance* behavior.

In order to demonstrate the *collision prevention and obstacle avoidance* behavior, buildings were used as the obstacles. Figure 6-12 shows the flight test location and the buildings. The distance of the SUAV take-off point to the building was estimated to be approximately 220m. For safety concern, the SUAV should not be commanded to navigate too close to the buildings. Thus, a single waypoint would be planned to be 80m away from the building and 150m away from the take-off point (sees Figure 6-12).

Figure 6-12: Flying site for Experiment E3

6.3.1 Procedures of Experiment E3

The followings outline the procedures for Experiment E3.

Pre-Flight Session

- Step 1:* Repeat *Step 1* described in Section 6.1.1, but the cruise speed was set to be 2.5ms^{-1} .
- Step 2:* Repeat *Step 2* described in Section 6.1.1 but using the PID gains in Table E-1 (in Appendix E).
- Step 3:* Repeat *Step 3* described in Section 6.1.1
- Step 4:* The flight mission was planned as shown in Table 6-8.

Table 6-8: Flight mission for Experiment E3

i.	<i>takeoff</i>	// To initiate take-off
ii.	<i>climb 40</i>	// To climb to an altitude of 40m
iii.	<i>flyTo (-150,-20)</i>	// To navigate to waypoint (-150, -20)
iv.	<i>circuit</i>	// To initiate autonomous landing

Four commands were pre-loaded to the autopilot, as listed in Table 6-8. The first two commands, instruct the SUAV to take-off and climb to an altitude of 40m. On waypoint was planned at (-150, -20) as shown in

command (iii). After reaching the final waypoint, an autonomous landing command would be initiated as shown in command (vii).

Step 5 to Step 10 presented described in Section 6.1.1 were then repeated.

In-Flight Session

Step 11: Repeat Step 11 described in Section 6.2.1

Step 12: Repeat Step 15 described in Section 6.2.1, but the data collected for Experiment E3 are tabulated in Table 6-9.

Table 6-9: Flight data required for the E3 experiment

States / Sensor Data	Symbols
SUAV Relative Locations	X_{act}, Y_{act}
Actual Altitude	h_{act}
Desired Altitude	h_{des}
Actual Heading	ψ_{act}
Desired Heading	ψ_{des}
Laser Reading	d_{laser}
Laser Flag	Γ_{laser}

Step 13: The ground-pilot continued to monitor the SUAV during the autonomous navigation. When the SUAV is travelling nearer to the waypoint, the laser readings would be expected to decrease with time. When the laser readings went below 150m, the distance threshold, the reactive behavior would be expected to be turned on and the SUAV would perform the *collision prevention & obstacle avoidance behavior*.

Step 14: After the SUAV finished performing the behavior, the ground-pilot would engaged the RC mode and take over the control from the autopilot. Then, the SUAV was descended and landed.

Post-Flight Session

Step 15: Repeat Step 21 described in Section 6.1.1.

Step 16: Repeat Step 22 described in Section 6.1.1.

6.3.2 Flight Test Results of Experiment E3

The relative locations of the SUAV collected from Experiment E3 are plotted on a two-dimensional navigation plot, as shown in Figure 6-13. The navigation path is superimposed on the terrain map of the flying site.

From Figure 6-13, looking at the navigation path plotted in blue, it was observed that the SUAV was navigating towards the first waypoint autonomously. Based on the flight mission as planned and discussed in *Step-4* in Section 6.3.1, the SUAV was expected to travel to the first waypoint and after reaching the waypoint, the SUAV would began autonomous landing. However, looking at Figure 6-13, when the SUAV reached Point-Z, instead of moving towards the designated waypoint, the SUAV was hovering at this point. The initial mission was interrupted when the SUAV reached Point-Z.

Figure 6-13: Navigation path taken by the ECO-8 SUAV in Experiment E3

The interruption of the flight mission showed that the *collision prevention and obstacle avoidance* reactive behavior had been activation. Examining the status of the reactive behavior flag will further affirm the activation.

Figure 6-14 shows the time history of the status of the Laser Flag (the reactive behavior flag) in magenta line. The blue line shows the distance of the obstacle detected by the laser sensor in blue line. As the laser was rigidly mounted on the rotary-wing SUAV, the pointing direction of the laser beam was easily affected by the SUAV vibration. As a result, the laser reading moved up and down drastically as shown on the figure from 200s to 220s. To prevent false-triggered, reading below 100 meter (as shown in green line) was filtered out and ignored during the flight test.

Referring to Figure 6-14 again, if the fluctuation of the readings was ignored, the distance detected was reducing in a linear manner, as shown with a brown line. By taking two points on the brown line and perform simple calculation, the brown line has a slope with magnitude 2.5ms^{-1} . The slope was consistent with the forward speed of the SUAV. When the distance detected decreased gradually below 150m at $t=210\text{s}$, the Laser Flag was turned. The flag changed from zero (OFF) to one (ON).

Figure 6-14: Distance reading from the laser sensor

After the flag was turned on, the *Navigational Reactive Behavior Controller* (NRBC) commanded the SUAV to perform the *collision prevention & obstacle avoidance* behavior. The NRBC commanded the SUAV to hover at its current location and altitude. The laser reading unchanged for 4 seconds showed that the SUAV was hovering (sees Figure 6-14 right after the behavior flag was turned on).

Figure 6-15: Reactive behavior altitude control

The time history of the desired and the actual altitudes of the SUAV are plotted in Figure 6-15. The desired altitude is plotted with blue line and the actual altitude is plotted with red line. At $t=210s$, it can be observed that when the behavior was enabled, the NRBC fixed the desired altitude at value equivalent to the SUAV altitude. From $t=210s$ to $t=300s$, it was observed that the SUAV was hovering at the altitude (red line) near the targeted altitude.

Figure 6-16: Reactive behavior heading control

After the reactive behavior was enabled for 4s, the NRBC began to control the SUAV heading. Figure 6-16 illustrates the results collected for the heading control. Blue line shows the time history of the desired heading and the red line shows the actual heading. At the moment when the reactive behavior was enabled, the desired heading was 250° . It can be observed from the figure that, the desired heading was

gradually decreased by from 250° to 160° . When the desired heading was changed by NRBC, the low-level controller adjusted the tail-rotor to adjust the SUAV heading, in order to track the commanded heading. The actual heading of the SUAV was observed to be following the desired heading closely as shown in Figure 6-16.

After the SUAV completed a 90° turning, the SUAV would continue to hover at the current location and waited for the ground-pilot command. At $t=330s$, the ground-pilot engaged the RC mode and landed the SUAV. The green line as depicted in Figure 6-13, shows the descending path taken by the SUAV during landing.

6.3.3 Discussion

The results from Experiment E3 show that the laser sensor integrated on the ECO-8 SUAV was functioning properly. The reading queried from the laser sensor was written to the shared memory of the autopilot and was accessible by the NRBC (thus, achieving objective E3-A). When, the laser reading went below the distance threshold, the *collision prevention & obstacle avoidance* behavior was triggered, as shown in Figure 6-15. Analysis on the altitude control and the heading control after the reactive flag was triggered validated that the NRBC implemented was capable of controlling the SUAV to execute the higher-level reactive behavior (i.e. the objective E3-B was achieved.). Success in executing the reactive behavior also showed that SUAV with limited processing power is capable of performing frontal *collision prevention and obstacle avoidance* behavior, accomplishing objective E3-C.

6.4 Experiment E4: Object Tracking Behavior Ground Test

Ground test that was conducted to investigate the object tracking behavior implemented using the CMUCam3, will be presented in this section. The objective of this test was to demonstrate the capability of the CMUCam3 in tracking a specific color coded object. The test is also aimed to demonstrate the use of the CMUCam3 to enable the reactive behavior flag.

The *object tracking* behavior discussed in Section 4.3.2, consists of two sub-behaviors, i.e. *search for object* (which is implemented on the CMUCam3 and *hover & track*. As the experiment conducted is a ground test, only the *search for object* behavior could be demonstrated.

6.4.1 Procedure of Experiment E4

The followings describe the experiment procedures:

- Step 1:* The ECO-8 SUAV was placed on a table with the CMUCam3 initially facing forwards.
- Step 2:* The tilt-motion of the pan-tilt unit was disabled. Thus, the CMUCam3 could only pan about the horizontal direction.
- Step 3:* The autopilot and the CMUCam3 were turned on, as in the SUAV was in real flight test.
- Step 4:* A “Fake GPS” command, i.e. a command to force the autopilot to initialize, was sent to the autopilot.
- Step 5:* After the autopilot was initialized, an orange cardboard was initially placed in front of the camera.
- Step 6:* Then, the cardboard was moved to the right and back to the original location. The intention of moving the cardboard was to allow the CMUCam3 to pan and follow the cardboard motion.
- Step 7:* The angle difference between the SUAV heading and the camera heading, ϵ , sent from the CMUCam3 to the autopilot will be logged into the data log file in the autopilot. The following table tabulates all the data required for this experiment.

Table 6-10: Data required for the E4 experiment

States / Sensor Data	Symbols
Angle Difference	ϵ
Actual Heading	ψ_{act}
Desired Heading	ψ_{des}

- Step 8:* Repeat from *Step-6*, but this time, the card board was moved to the left.

6.4.2 Ground Test Results of Experiment E4

Figure 6-17 shows the images obtained from a video recording the *search for object* sub-behaviors. The images showed that the CMUCam3 was detecting and moving with the orange cardboard. When the board was shifted away from its original location, the CMUCam3 detected that the centroid of the orange blob (or the cardboard object) was moved from its original location. For example, referring to Image-#01, the CMUCam3 was initially pointing forward. When the board was shifted towards the right, as shown on Image-#02, CMUCam3 was panned to the right (negative direction). Similarly, when the board was shifted to the left (positive direction) the camera was panned to the left, as shown on Image-#03.

Figure 6-17: Ground test results of the object tracking using the CMUCam3

Once the object was successfully tracked, the image captured by the CMUCam3 would be saved into the SD flash memory installed on the camera. Figure 6-18 shows the images retrieved from the SD flash memory card. From the figure, the CMUCam3 tried to track the object so that the centroid of the object coincided with the centroid of the image.

Figure 6-18: Images captured by the CMUCam3 vision sensor

After the orange object was successfully detected, the CMUCam3 Flag was enabled and the *hover & track* behavior was engaged. When this behavior was

activated, the NRBC controlled the altitude and the heading of the SUAV. Since, the SUAV was on ground, the altitude would not be studied.

Figure 6-19 shows the heading control when the *object tracking* reactive behavior has been triggered. The blue line indicates the angle difference between the helicopter heading and the CMUCam3 heading. As the helicopter heading in the experiment was fixed, the angle difference, ϵ , was equivalent to the camera pan angle. The green line shows the current heading of the SUAV and the red line shows the desired heading commanded by the NRBC. The algorithm to compute the pan angle is relatively short and simple, thus the computational delay was assumed to be insignificant.

Figure 6-19: Heading control when *tracking & hover* behavior was enabled

Referring to Figure 6-19, after the CMUCam3 Flag was enabled, the NRBC controlled the SUAV by adjusting the desired heading. When the *object tracking behavior* was activated, the desired heading was the sum of the current heading and the angle difference, as given in equation (4-1). Since the SUAV was not moving, the actual heading of the SUAV remain unchanged throughout the experiment (sees the green line on Figure 6-19). Thus, when the angle difference changed, the desired heading would also change. As a results, the desired heading (red line) shared a similar trend with the angle difference.

6.4.3 Discussion

In summary, the results from the ground test demonstrated the following:

- The *search object* sub-behavior implemented on the CMUCam3 was capable of tracking an object with a specified color.
- When the object was detected, the CMUCam3 tracked and moved in the direction of the object.
- The reactive behavior flag (i.e. the CMUCam3 Flag) was enabled when the CMUCam3 successfully detected the object.
- The angle difference, ϵ , between the camera heading and the SUAV heading was estimated by the CMUCam3 and stored into the shared memory in the autopilot module.
- When the CMUCam3 Flag was enabled, the NRBC on the autopilot read the angle difference from the shared-memory, and changed the desired heading of the SUAV.

Although, the ground test conducted cannot represent the actual behavior, the ground test has successfully demonstrated that the behavior was called after receiving a trigger signal. The NRBC also successfully took over the control from the autopilot by changing the desired heading, in order to minimize the angle difference, ϵ . Also, from the flight test results collected from Experiment E3 (Section 6.4.3), it was observed that when the desired heading was changed by the NRBC, the SUAV tracked the commanded heading. Thus, the result suggested that if the object tracking behavior is triggered during flight, the SUAV shall be able to track the desired heading commanded by the NRBC.

Figure 6-20: Camera configuration in flight test

Attempt to demonstrate the behavior through flight test had been carried out. Figure 6-20 shows the configuration of the camera during the flight test. In the test, instead of having an object in front of the camera, the object was arranged on the ground. The CMUCam3 was tilted to face downwards, to search for the object.

The flight test was not successfully conducted. This was because the behavior was either false triggered or the camera failed to detect the object. Images retrieved from the CMUCam3 showed that the quality of the image in an outdoor operation was very much different from the indoor environment. Figure 6-21 shows the images captured from the outdoor test. The images were corrupted with noises. This may be possibly caused by high temperature and glaring effect. As a result, the orange object was difficult to be tracked. Since false-triggered may result in danger the flight test was not continued due to safety concern.

Figure 6-21: Images captured by the CMUCam3 from flight test

6.5 Experiment E5: Testing of the Graphical User Interface

This section presents the experiment conducted to validate the functionalities of the GUI developed. The objective of the test was to demonstrate that the GUI was working properly in receiving telemetry and sending commands to the SUAV. The telemetry received should be updated on the GUI.

The experiment employed the ECO-8 SUAV as the research testbed. A simple modification was performed on the ECO-8 platform. The laser sensor was taken out from the SUAV and the KPC-S226CB video camera (as presented in Section

3.2.2) was installed on the SUAV to provide a forward looking-view. A digital video camcorder was used to manually film the SUAV flight from the ground. Both video signals from the onboard camera and camcorder were channeled to the GCS. Figure 6-22 shows the setup of the experiment.

Figure 6-22: Experiment setup for Experiment E5

6.5.1 Procedures of Experiment E5

A flight test was conducted and the SUAV was deployed in the RC mode by the ground-pilot. The following outline the experiment procedures:

- Step 1:* The autopilot and the onboard video camera were turned on.
- Step 2:* Using the RRC-GCS graphical user interface, communication with the autopilot was initiated. This was performed by selecting the SUAV to be connected and clicking the “Connect” button on the GUI.
- Step 3:* After the communication was established, the GPS status was monitored. Telemetry as tabulated in Table 3-2 (sees Section 3.3.1) was received by the GCS. Video signals from the ground and onboard video would be displayed on the GUI.
- Step 4:* After the GPS was locked, the ground-pilot began the deployment.

- Step 5: The SUAV was taken-off and the ground-pilot maneuvered the SUAV around the flight site to capture different video images.
- Step 6: After approximately four minutes, the ground-pilot landed the SUAV.
- Step 7: The communication was closed by clicking on the “Disconnect” button on the GUI.

6.5.2 Results of Experiment E5

Figure 6-23: GUI display after the communication was established

During the deployment, the changes on the GUI were captured in a video file. Pictures extracted from the video will be discussed in this subsection. Figure 6-23 shows the display of the GUI after the communication was established between the GCS and the SUAV. This can be observed from the “Connection and Flight Log Display”, where the message “ECO-8 Helicopter is Connected” was displayed, showing that the SUAV had been successfully connected and the communication was established.

Figure 6-24: GUI display during the navigation

Figure 6-24 shows the displays of the GUI when the SUAV was airborne. It was observed that the telemetry received from the SUAV was successfully updated on the GUI. The “*Digital Cockpit*” was changing with time due to the changes in the SUAV orientation and altitude. From the display, it can also be observed that the video from the ground camcorder and the forward-looking view onboard camera were successfully received by the GCS and were rendered on the GUI. Since, the ECO-8 SUAV had only one forward-looking camera installed during the experiment, the “*Downward Looking View*” was blank as shown in Figure 6-24. The location and the heading of the SUAV were also updated on the “*Navigation Map*”.

6.5.3 Discussion

The results presented in Section 6.5.2 had demonstrated that communication between the SUAV and the GCS was successfully established. A two-way transmission between the SUAV and GCS was achieved where the telemetry was received by the GCS and commands captured by the GUI were transmitted to the SUAV.

The results also showed that the telemetry received was successfully updated on the “*Digital Cockpit*”, “*Connection and Flight Log Display*” and “*Navigation Map*”. Video footages received from the ground camcorder and the airborne camera were also successfully displayed on the GUI.

CHAPTER 7: DISCUSSION AND CONCLUSION

This thesis focuses on the development of prototype of a Human-SUAV system using COTS autopilot module. In the quest to design an autonomous SUAV system, the control architecture implemented in the early stage of the development was crucial in facilitating the development and incorporation of higher-level behaviors. Section 7.1 provides a discussion on the Human-SUAV prototype developed and the results obtained from the experiments. The contribution of this thesis will be summarized in Section 7.2. Finally, the last section provides recommendations of future work in improving the current research.

7.1 Discussion

7.1.1 Development of a Human-SUAV System Prototype

The development of a “physical” Human-SUAV system was necessary in order to provide a realistic experimental setting for the study of SUAV deployment. As presented in Chapter 3, a Human-SUAV system had been successfully developed.

Development of the SUAVs

Two fixed-wing, i.e. the Rainbow and the M36, and one rotating-wing SUAVs, i.e. the ECO-8 SUAVs were developed as the research testbeds. These vehicles were modified from the RC platforms used by hobbyists. The original RC platforms consist of only a propulsion and actuation subsystem. They can only be deployed manually by the ground-pilot in the RC mode. Without onboard sensor suite and controller, it will not be possible for the SUAV to achieve autonomous flight. Thus, a flight control subsystem (FCS) with a COTS autopilot module and a sensor suite were integrated to each vehicle.

Integration of a COTS autopilot module to a fixed-wing SUAV has been well publicised, for example by Krashanitsa et. al. [76], Platanitis et. al. [55], and Jodeh et. al. [66], but not for rotating-wing SUAV. Most of the available rotating-wing research platforms were integrated with self-developed autopilot modules. For example, the vehicles from Mierau et. al [102] and Saripalli et. al. [79]. This may be due to the limited variety of COTS autopilot module available commercially for

rotating-wing SUAV at the moment, except for the MP2128^{HELI} module from MicroPilot.

Successful flight tests conducted in Experiment E1, Experiment E2 and Experiment E3 (sees Section 6.1 to Section 6.3) showed that the FCSs built from the MicroPilot autopilot modules, were successfully integrated to the SUAVs. The results from Experiment E2 (sees Section 6.2.2) also demonstrated that the MP2128^{HELI} autopilot could be employed to control small rotating-wing SUAV.

Integration Issue

Figure 7-1: Corrupted video signals

The main challenge encountered during the development stage was the difficulty to incorporate various electrical and mechanical components into the limited and constrained space in the SUAVs. Although COTS products that are developed for SUAV are relatively small as compared to conventional sensors and controllers, their sizes are still considerably large for the research vehicle platforms. Insufficient space caused various practical issues. Short distances among the electronics components also posed a high possibility of interferences. During the development stage, interference was encountered with the telemetry data and video transmission. Video signals received by the ground receivers were corrupted with signal noises. As a result, the video displayed were not clear as shown in Figure 7-1. In a worse situation, the telemetry signal from the airborne vehicle could be interfered and communication between the SUAV and GCS was lost.

Limited space also posed a challenge in arranging the various components into the SUAV. Placement of additional components (i.e. sensors, controllers and payloads) will effectively moved the RC airframe CG location, which was designed by the manufacturer to be one-third from the airframe nose. Study in Section 5.1.4 showed that changes in the location of the CG will effectively affect the magnitude of the pitching moment generated by the elevator deflection. Thus, it is ideal to arrange the components so that the modified CG location coincides with the original CG. For example, in this context, the Rainbow platform has a larger fuselage area than the space near the nose. As a result, most of the components were installed and mounted on the fuselage. Consequently, the CG location was shifted aft the original CG location. In order to move the CG location forward, two battery supplies was placed on the nose instead of one in as intended in the initial plan.

Development of the Ground Control Station

The ground-pilot (or the flight crew) communicates with the airborne SUAV through the graphical user interface (GUI) developed on the ground control station (GCS). Section 3.3 presented a concept for integrating the RC mode, internal mode and the autonomous mode into a single GUI. Based on the concept, a multi-mode control GUI had been developed.

The GCS subsystem was not fully tested due to the absent of a professional UAV ground-pilot, who supposes to assist by conducting the flight test. A professional UAV ground-pilot is deemed to be necessary as the test would require the ground-pilot to switch between modes of control, without looking at the physical platform. Furthermore, deploying an UAV by using a joystick requires a different skill-set and is not a conventional deployment method for an ordinary RC ground-pilot. Besides, there is a lack in understanding on the suitability and effectiveness of the GUI in assisting the ground-pilot to deploy and monitor the SUAV for urban applications. Evaluation of the GUI would require expertise from the cognitive sciences field.

However, an experiment had been conducted to validate the functionalities of the GUI developed as presented in Experiment E5 (sees Section 6.5). The results from the experiment (sees Section 6.5.2) showed that the GUI was functionally operational.

Telemetry was successfully received and based on the telemetry, the map and the digital cockpit was updated. The results also showed that the video received by the GCS was successfully rendered on the GUI. Commands captured by the GUI were also sent to the SUAV successfully.

7.1.2 Implementation of the Control Architecture

The main focus of this thesis is on the implementation of a control architecture for the SUAV avionics subsystem. Review in Section 2.1.1 concluded that the control architecture proposed by Boskovic et. al. [42] and Narayan et. al. [45] could be employed for the research application. Based on the discussion presented in Section 4.1, for a SUAV with limited payload and limited power supply, the control architecture could be implemented using a sole autopilot controller.

Thus, this thesis implemented the architecture using the MicroPilot autopilot module. In Section 4.2.1, a detailed study on the autopilot control architecture leads the following major findings:

- i. First of all, the highest level capability implemented by the manufacturer is the *go-to waypoint* behavior. This is a deliberative behavior, i.e. the autopilot controls the SUAV to execute the navigation behavior based on a planned flight mission.
- ii. Secondly, the higher-level reactive behavior subsystem is absent from the original autopilot control architecture. For urban deployment, it is desirable to have behaviors that could react instantaneously to external stimuli. As a result, higher-level reactive behaviors incorporation to the architecture is needed for the SUAV.

In order to incorporate the higher-level reactive behavior, the waypoint navigation behavior has to be achieved first. This is because a reactive behavior reacts to external stimuli to interrupt a deliberative behavior. Thus, without an existing deliberative behavior, the reactive behavior will not be possible to be demonstrated.

7.1.3 Automation of the Low-Level Control

Automation of the low-level controller is mandatory to ensure the success of higher-level behaviors execution (both the deliberative and reactive). Automating the lowest-level control implies the implementation of a controller for the *motion and guidance* control [43, 64], and this has been carried out (sees Chapter 5).

The Analytical Approach

To design the controller for *motion and guidance* control, the required feedback control loops involved during level flight were: (i) the *heading from roll*, (ii) the *aileron from roll*, (iii) the *pitch from airspeed*, (iv) the *elevator from pitch*, and (v) the *throttle from altitude* (sees Section 5.1). An analytical approach to determine the PID gains to be implemented on these feedback loops (except for loop (v)) were outlined in Section 5.1.1. The methodology was demonstrated using the Rainbow fixed-wing UAV.

The challenge of this approach was the requirement of a mathematical representation that describes the motion of the UAV. Employing the Advanced Aircraft Analysis (AAA) and the SolidWorksTM software modules, the physical properties and the aerodynamic coefficients required for the mathematical representation was estimated (sees Section 5.1.2). Based on these estimations, the linear equation of motions for the longitudinal and lateral motions were successfully derived as given in equations (5-6) and (5-7) in Section 5.1.2.

From the equations of motion, the transfer functions for each of the feedback loops were derived, as stipulated in equations (5-8) to (5-12) (sees Section 5.1.3). The PID gains determined from analyzing the *step input response* of each of these loops were tabulated in Table 5-8 (in Section 5.1.3). Results of the Experiment E1

in Section 6.1.3 provided the evidences that the PID gains determined in Chapter 5 were suitable in controlling the SUAV for the feedback loops. The SUAV managed to navigate in the direction intended, and was able to track the commanded roll and pitch angles closely. However, the autopilot did not manage to maintain the SUAV in the desired airspeed (see Section 6.1.3). Further investigation of the *pitch from airspeed* loop showed that the inability of the controller to maintain the SUAV at the cruise speed was possibly due to an inappropriate initialization.

Besides validating the suitability of the PID gains for controlling the SUAV, the flight test results from Experiment E1 were also used to verify the mathematical model (refers to Section 6.1.4). It was found that the actual responses of the SUAV agreed well with the numerical simulation results computed from the mathematical model. This showed that the mathematical model for the Rainbow SUAV derived using the analytical approach was close to the actual responses of the SUAV in a closed-loop system. Small bound of discrepancy between the simulation and flight test results was observed. Measurement errors during the *modeling phase*, where the SUAV dimensions were physically measured, could be one of the possible contributors to the estimation error. The measurement errors may also be due to human's faults (e.g. parallax error) or apparatus errors. The imperfect model could have also contributed to the estimation error. For example, in the mathematical model derived in this thesis and by researchers who are doing similar works [66, 96, 100], disturbance due to wind were not taken into account, which in reality, the effect of wind on the performance of the SUAV is inevitable.

Usability of the Methodology for other SUAV Platforms

The analytical methodology could basically be applied on other fixed-wing SUAVs that use classical feedback control theory for flight automation. However, certain modifications have to be carried out to accommodate the different configurations. For example in the *modeling phase*, the mathematical model presented was derived based on the configuration where the SUAV has aileron, elevator and rudder control surfaces. Thus, if the SUAV control surfaces are

different from this configuration, the mathematical expressions have to be derived again with regards to the new SUAV configuration.

The discussion of the methodology focuses on the fixed-wing SUAV. In practice, the three phases of the methodology could also be applied to the rotating-wing SUAV. Among all the three phases, a major modification has to be carried out in the *modeling phase*. The mathematical model presented in this thesis is not applicable to a rotating-wing SUAV. The direction of the thrust and the effect of the control surfaces on the motion are very different from the fixed-wing SUAV. The EOMs have to be derived from the first principal. Although, the mathematical expressions are different, a rotating-wing SUAV can still be analyzed in trim conditions [102-104]. This allows the EOMs to be expressed in linear decoupled equations, similar to analysis conducted to the fixed-wing SUAV. In the *controller design phase*, the SISO transfer functions can be derived by identifying the major effect of each control surface to the SUAV motion response. After obtaining the SISO transfer function, a similar approach of tuning could be employed to determine the PID gains.

The Experimental Approach

The experimental approach, (or the “*test, observe and tune*” approach), was employed to tune the PID gains for the *throttle from altitude* feedback loop (refers to Section 5.2). After attempting the “*test, observe and tune*” approach for four times, the PID gains used for the last attempt were deemed to be acceptable, and the gains were given in Table 5-10 (in Section 5.2). Although no analytical tuning was performed for this loop, results from Experiment E1 showed that the autopilot managed to control the SUAV to maintain at the targeted altitude, thus the *altitude hold* behavior was regarded as successfully tuned.

The experience gained using the experimental approach showed that, many of the decision made during the ‘*test*’, ‘*observe*’ and ‘*tune*’ processes were arbitrarily made. This highlighted that there is a lack in a proper guideline for the approach. For example, some of the question encountered as presented below may have no definite answer:

- i. *What are the initial PID gains that should be used for the first flight test?*
- ii. *When is it appropriate to adjust the proportional, derivative or the integral gain? How should it be adjusted?*
- iii. *When could the tuning be deemed to be successfully performed and the response was accepted?*

Although, the tuning approach as demonstrated in Section 5.2 could not be regarded to be a standard way of tuning the gains experimentally. The discussion provided one of the possible set of answers to the three questions presented above.

Comparison between the Analytical and Experimental Approaches

The analytical approach was found to be more cumbersome as compared to the experimental approach. Estimation of the physical and aerodynamic coefficients required the construction of the SUAV CAD models, measurements of the physical airframes geometries and inputting plenty of geometries data into the AAA parametric module.

On the other hand, the experimental approach, '*appears*' to be offering a quicker solution. The word '*appear*' is highlighted because quick successful tuning is achievable if the followings conditions are satisfied:

- i. The ground-pilot has the necessary experiences and skills to perform recovery should erratic responses were observed. Experimental results have shown that inappropriate implementation of the PID gains for the *throttle from altitude* feedback loops caused the SUAV to climb and descend periodically (sees Section 5.2). Failure in responding to the erratic response may result in damages to the SUAV components or airframe. In consequences, lengthening the development process.
- ii. The developer is capable of adjusting the PID gains based on the flight test results. This is to keep the number of tests required to the minimum level. In the approach demonstrated (sees Section 5.2), a total of four flight tests were carried out for tuning only a single feedback loop.

Although, both approaches have their own advantages and setbacks, it is undeniable that the experimental approach poses a certain risk of components or vehicle damages. Since safety outweighs any other factors, analytical approach should be a preferable option to begin with, in order to obtain a suitable initial PID gains offline.

7.1.4 Achieving Autonomous Navigation

The deliberative behavior, i.e. the *go-to waypoint* behavior, of the original autopilot control architecture was demonstrated in Experiment E1 (refers to Section 6.1) and Experiment E2 (refers to Section 6.2), with the Rainbow and ECO-8 UAVs respectively. For both experiments, mission flights to instruct the UAVs to perform navigation to four waypoints were planned. The experiment results showed that the Rainbow and ECO-8 UAVs successfully completed the mission (see Section 6.1.2 and Section 6.2.2). Results from Experiment E2 also demonstrated the capability of the ECO-8 rotating-wing UAV to vertically take-off and landing autonomously.

The demonstration conducted also showed that by defining the waypoint systematically, the UAVs could be commanded to perform a certain flight pattern. For instance, in Experiment E1, the four waypoints were defined to form a flight area of 200m by 250m. The results showed that by instructing the UAV to fly to the same waypoint locations again in the same sequential steps, the UAV was observed to be performing a holding pattern around the same area. This holding behavior will be useful for missions that require collecting information from the same area such as surveillance and reconnaissance.

7.1.5 Achieving Higher-Level Reactive Behaviors

To incorporate higher-level reactive behaviors to the autopilot control architecture, modifications to the architecture were carried out. The modified architecture has to satisfy two main requirements, i.e.: (i) the architecture has to allow the FCS to read external stimuli and (ii) the architecture shall permit the reactive behavior to interrupt the original deliberative behavior, when a specific stimulus was received.

Development of a Mechanism to Incorporate the Reactive Behaviors

Two subsystems were introduced to the original control architecture, i.e. the *Detection Subsystem* and the *Navigational Reactive Behavior Controller* (NRBC) to fulfill the above mentioned two requirements (sees Figure 4-2 in Section 4.2.1). The *Detection Subsystem* queries the sensory data from the external sensor and enables the behavior flag, when a designated signal or stimulus is detected.

The implementation of the NRBC was based on a mechanism as discussed in Section 4.2.2. In order to interrupt the original deliberative behavior, the NRBC commands overwrite the higher-level commands given by the autopilot module. By subsuming the autopilot original commands, the NRBC, thus, taking-over the control from the autopilot original algorithm. In this case, when the behavior flag was enabled, the NRBC could be employed to command the SUAV for executing a given reactive behavior.

Implementation of the Higher-Level Reactive Behaviors

Two reactive behaviors algorithm, i.e. the *collision prevention & obstacle avoidance* (sees Section 4.3.1) and *object tracking* (sees Section 4.3.2) were used to demonstrate the feasibility of the NRBC implementation. The former behavior uses the reading from the laser sensor as the external stimuli for triggering. The behavior prevented the SUAV from colliding with frontal obstacles. On the other hand, the *object tracking* behavior depends on the CMUCam3 vision sensor for triggering. This behavior allows the SUAV to track an object with a specific color code.

In Section 6.3, Experiment E3 was conducted using the ECO-8 SUAV to demonstrate the *collision prevention & obstacle avoidance* behavior. In the mission flight planned, the SUAV was instructed to take off autonomously, navigate to a waypoint and after reaching the waypoint, begin an autonomous landing. However, the results from the experiment (sees Section 6.3.2) showed that the SUAV did not follow the intended deliberative plan. The *collision prevention & obstacle avoidance* behavior interrupted the mission. The behavior was executed when the distance of the obstacle detected by the laser reading went below the pre-

set threshold. Results from Experiment E3 also showed that when the behavior was triggered, the SUAV hovered at its current location and altitude, preventing it from moving forward. This successfully demonstrated the algorithm implemented.

A ground test, i.e. the Experiment E4 was conducted to verify the implementation of the *object tracking* behavior (sees Section 6.4). The test was designed to show the capability of the CMUCam3 in tracking an orange object and to demonstrate the execution of the behavior to control the SUAV. The results from the experiment (sees Section 6.4.2), showed that the CMUCam3 was capable of detecting and tracking an orange object. The result also showed that when the orange object was, the CMUCam3 Flag was enabled. After the flag was turned on, the commands from the reactive behavior successfully overwrote the original commands from the autopilot deliberative behavior.

The results from Experiment E3 and Experiment E4 provided evidences that the autopilot original architecture had been successfully modified to incorporate higher-level behaviors. The implementation of the *Detection Subsystems* (the laser sensor and the CMUCam3 vision sensor) and the NRBC had been accomplished. The *Detection Subsystems* was capable of continuously querying readings from the external sensor that senses external stimulus, and the data was stored into the shared-memory location.

Also, the NRBC had also successfully executed the implemented reactive behavior to react instantaneously to the specific external stimulus.

7.2 Concluding Remark

This research contributes to the knowledge of developing a Human-SUAV system using COTS autopilot modules for urban deployment. The focus of the research was on implementing a control architecture on the avionics subsystem that could facilitate deliberative and reactive behaviors executions. The following summarizes the contributions of this thesis:

I. Development of a Human-SUAV System

A Human-SUAV system that encompasses three small aerial unmanned vehicles and a ground control station had been developed. The SUAVs were integrated with an avionics subsystem that consists of an autopilot module, a sensor suite, and a payload subsystem. The ground control station developed facilitates the ground-pilot in deploying the SUAVs in multiple modes of control.

II. Implementation of the Control Architecture

The control architecture based on Boskovic et al. [42] and Narayan et al. [45] architectures were implemented on the SUAV avionics subsystem. The original autopilot architecture provided the execution of deliberative behaviors, (i.e. the *go-to waypoint* behavior). For higher-level reactive behavior incorporation, the original autopilot control architecture was successfully modified and enhanced by including the *Detection Subsystem* and *Navigational Reactive Behavior Controller*. Incorporating these two modules into the original control architecture allows the autopilot to detect external stimuli and execute higher-level reactive behaviors.

III. Designing the Motion and Guidance Controllers

The PID gains for the *motion and guidance* controller had been successfully determined through analytical and experimental approaches. Experimental results had shown that the SUAV was capable of tracking the direction the SUAV is heading to while maintaining its altitude during level flight.

IV. Demonstration of the Deliberative Behavior (Waypoint Navigation)

Flight tests results showed that the Rainbow and the ECO-8 SUAVs had demonstrated the deliberative behavior successfully. The SUAVs followed the commands in the flight mission and completed the mission by navigating to the pre-loaded waypoints successfully without intervention from the ground-pilot.

V. Demonstration of the Higher-Level Reactive Behaviors.

Successful incorporation of the higher-level behaviors into the original autopilot control architecture had been demonstrated through flight test. From the experimental results, the *collision prevention & obstacle avoidance behavior* was successfully triggered and executed. It can be concluded from the results that it is feasible to modify the autopilot module control architecture for developing the required reactive behaviors for urban deployment.

7.3 Recommendation of Future Work

7.3.1 **Unaddressed Issue**

The ultimate aim of implementing the control architecture proposed by Boskovic et al. [42] and Narayan et al. [45] is to develop a SUAV that has the capability of making high-level decision autonomously. Currently, the research work assumes that the higher level behaviors, i.e. the *path-planning* and the *decision-making* control layers, are performed by human operator.

Future work should be focused on the automation of these two layers of control. Autonomous path planning algorithm, for example using the Rapidly-exploring Random Tree (RRT) algorithm can be implemented to automate the *path-planning* layer [61]. Although, the current processing power for the SUAV may be limited, the hardware configuration as discussed in Section 4.1 that uses the ground control station to compute the path could be employed. Similar work discussed by Saunders [61] has shown that this is possible.

7.3.2 **Improvement on the Current Work**

Current work has addressed various issues broadly. In depth investigation and implementation could be carried out to further enhance on the research. The following discusses the areas that need further attention.

Improvement on System Design

Further efforts in designing and improving the system should involve expertise from the cognitive sciences field and also from the practitioners, whom will be the end-users of the system eventually. Figure 7-2 shows the collaboration of the three areas.

Figure 7-2: Collaboration among expertises to develop a suitable system for urban application

Although the system is functionally operating, the suitability of the system for urban deployment needs to be evaluated. For example, the effectiveness of the graphical user interface in facilitating and assisting the ground-pilot to deploy the SUAV needs to be studied. Experts from the cognitive sciences will be able to offer solution in designing the evaluation and thus, provide suggestions in improving the design. On the other hand, the practitioners can assist in conducting the tests and provide feedbacks on the system requirements needed. For instance, the practitioner may provide valuable suggestions such as including certain crucial information to the GUI.

Improvement on the Reactive Behaviors Robustness

The behaviors implemented were to demonstrate the capability of the NRBC in controlling the SUAV. Although the flight test conducted had successfully demonstrated the *collision prevention & object tracking*, the followings could be

further improved. Firstly, a filtering algorithm shall be devised to capture only the usable information from the laser reading. Secondly, the robustness of the algorithm has to be improved. More flight tests can be conducted to evaluate the dynamic stability and performance of the SUAV when the reactive behaviors are triggered.

Improvement on the Current Analytical Tuning Methodology

The methodology discussed in tuning the PID gains can be improved to increase the accuracy of the mathematical model derived. Currently, the flight data collected were used only for verification purposes. The flight data can be employed to fine tune the aerodynamic coefficients using system identification methods [73, 74, 77].

Reference

- [1] Office of the Secretary of Defense, (2005), *Unmanned Aircraft System Roadmap 2005-2030*, Washington DC, USA.
- [2] J. A. Miller, P. D. Minear, and A. F. J. Niessner (2007), Intelligent Unmanned Air Vehicle Flight System, *Journal of Aerospace Computing, Information and Communication*, Vol. 4, No. 5, pp. 816-834.
- [3] Picture of Aerosonde UAV, Aerosonde Pte Ltd.
- [4] B. L. Mulac, G. J. Holland, and J. O. Pinto and J. A. Curry (2002), Applications of Aerosondes to Hurricane Research, *AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles*, Portsmouth, Virginia, 20-23 May.
- [5] Picture of Yamaha RMax UAV, Yamaha Motor Corporation.
- [6] B. Enderle (2002), Commercial Applications of UAV's In Japanese Agriculture, *AIAA's 1st UAV Conference*, Portsmouth, Virginia, 20-23 May.
- [7] MLB Bat UAV, MLB Company.
- [8] S. Morris and H. Jones (2004), Examples of Commercial Applications Using Small UAVs, *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit*, Chicago, Illinois, 20-23 September.
- [9] R. Pretolani, G. M. Saggiani, and B. Teodorani (2003) A Low Cost Unmanned Helicopter Platform for Geophysical and Environmental Applications, *Proceedings of the Workshop on Airborne Remote Sensing for Geophysical and Environmental Applications*, P.Le Aldo, Rome, April 14-16.
- [10] D. J. Pines and F. Bohorquez (2006), Challenges Facing Future Micro-Air-Vehicle Development, *Journal of Aircraft*, Vol. 43, No.2, pp 290-305.
- [11] G. A. Bekey (2005), *Autonomous Robots from Biological Inspiration to Implementation and Control*, MIT Press, Cambridge.
- [12] Dragon Eye, Aerovironment Inc.
- [13] M. Gad-el-Hak (2001), Micro Air Vehicles: Can They Be Controlled Better?, *Journal of Aircraft*, Vol. 38, No. 3, pp 419-429.
- [14] J. Grasmeyer and T. Keennon (2005), Development of the Black Widow Micro Air Vehicle, *39th Aerospace Sciences Meeting and Exhibit*, Reno, NV, 8-11 January.

-
- [15] R. K. Arning (2004), Flight Control of Micro Air Vehicles, in *AIAA Guidance, Navigation and Control Conference and Exhibit*, Providence, Rhode Island, 16-19 August.
 - [16] E. Carnevale, D. V. Cohn, M. M. Kent, S. Maki, Z. Malsawma, R. Skolnik, and S. Yin (2007), World Population Highlights: Key Findings from Population Reference Bureau's 2007 World Population Data Sheet, *Population Bulletin*, Vol. 62, No.3, pp 1-12.
 - [17] J. Taw and B. Hoffman (1995), The Urbanization of Insurgency: The Potential Challenge to U.S. Army Operations, *Small Wars & Insurgencies*, Vol. 6, No.1, pp. 68-87.
 - [18] R.O. Hundley and Eugene C. Gritton (2004), Future Technology-Driven Revolutions in Military Operations, National Defense Research Institute, RAND).
 - [19] S. Tsach, A. Peled, D. Penn, B. Keshales, and R. Guedj (2007), Development Trends For Next Generation UAV Systems, *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, Rohnert Park, California, 7-10 May.
 - [20] M. Peck (2003), Pentagon Unhappy About Drone Aircraft Reliability, *National Defense*, Vol 594, pp 36-37.
 - [21] Office of the Secretary of Defense, Washington DC, USA (2003), *Unmanned Aerial Vehicle Reliability Study*.
 - [22] K. W. Williams (2004), A Summary of Unmanned Aircraft Accident/Incident Data: Human Factors Implications, *DOT/FAA/AM-04/24*, Civil Aerospace Medical Institute, Federal Aviation Administration, Oklahoma City.
 - [23] M. J. Barness, B. G. Knapp, B. W. Tillman, B. A. Walters, and D. Veilcki (2000), Crew Systems Analysis of Unmanned Aerial Vehicle (UAV) Future Job and Tasking Environments, *ARL-TR-2081*, Army Research Lab, Aberdeen Proving Ground.
 - [24] K. Ro, J. Oh, and L. Dong (2007), Lessons Learned: Application of Small UAV for Urban Highway Traffic Monitoring, *45th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 8-11 January.
 - [25] B. T. Clough (2005), Unmanned Aerial Vehicles: Autonomous Control Challenges, a Researcher's Perspective, *Journal of Aerospace Computing, Information and Communication*, Vol. 2, No.8, pp. 327-347.
 - [26] K. W. Ong (2006), *Sharing and Trading in a Telerobotics System*, PhD Dissertation, Robotic Research Centre, Nanyang Technological University, Singapore.
-

-
- [27] H. Zhang, Z. Liu, and S. X. Yang, A Hybrid Robot Navigation Approach Based on Partial Planning and Emotion-Based Behavior Coordination, *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.
 - [28] R. C. Arkin (2008), *Behavior-Based Robotics*: MIT Press, Cambridge.
 - [29] B. T. Clough (2005), Unmanned Aerial Vehicles: Autonomous Control Challenges, a Researcher's Perspective, *Journal of Aerospace Computing, Information and Communication*, Vol. 2, No.8, pp. 327-347.
 - [30] S. Brainov and H. Hexmoor (2002), Quantifying Relative Autonomy in Multiagent Interaction, In: *Agent Autonomy*, C. C. H. Hexmoor, R. Falcone, (Eds), pp. 43-56, Kluwer Academic Publishers, Boston.
 - [31] H. J. Kim, D. H. Shim, and S. Sastry (2002), Flying Robots: Modeling, Control and Decision Making, *IEEE International Conference on Robotics and Automation*, pp. 66-71, Washington, DC, 2002, 11-15 May.
 - [32] E. Jackson, O. Williams, and K. Buchan (1991), Achieving Robot Autonomy, *Proceedings of the 3rd Conference on Military Robotic Applications*, DND, Canada, 1991.
 - [33] S. R. Dixon, C. D. Wickens, and D. Chang (2005), Mission Control of Multiple Unmanned Aerial Vehicles: A Workload Analysis, *Human Factors*, Vol. 47, No. 3, pp. 479-487.
 - [34] C. D. Wickens, S. R. Dixon, and D. Chang (2002), Workload Demands of Remotely Piloted Vehicle Supervision and Control: Single Vehicle Performance, *Technical Report*, Institute of Aviation, University of Illinois, Illinois, USA
 - [35] M. L. Cummings and P. J. Mitchel (2007), Operator Scheduling Strategies in Supervisory Control of Multiple UAVs, *Aerospace Science and Technology*, Vol. 11, No.4, pp. 339-348.
 - [36] T. B. Sheridan (1992), *Telerobotics, Automation, and Human Supervisory Control*. MIT Press, Cambridge.
 - [37] K. W. Ong, G. Seet, and S. K. Sim (2005), Sharing and Trading in a Human-Robot System, In: *Cutting Edge Robotics*, V. Kordic, A. Lazinica, and M. Merdan, (Eds), pp. 467-499. Pro Literatur Verlag, German.
 - [38] M. Mataric (1992), Behavior-Based Control: Main Properties and Implications, *Workshop on Intelligent Control Systems, International Conference on Robotics and Automation*, pp 46-54, Nice, France, May.
-

-
- [39] R. A. Brooks (1991), Intelligence without Reason, *International Joint Conference on Artificial Intelligence*, pp. 569-595, Sydney, Australia.
 - [40] R. A. Brooks (1986), A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation*, Vol. 2, No. 1, pp. 14-23.
 - [41] D. Shi (2006), *Aerial Robot Navigation in Cluttered Urban Environments*, PhD Dissertation, Department of Mechanical Engineering, The Florida State University, USA.
 - [42] J. D. Boskovic, R. Prasanth, and R. K. Mehra (2002), A Multi-Layer Control Architecture for Unmanned Aerial Vehicles, *Proceedings of the American Control Conference*, Anchorage, AK, 2002, 8-10 May.
 - [43] T. R. Yechout, S. L. Morris, D. E. Bossert, and W. F. Hallgren (2003), *Introduction to Aircraft Flight Mechanics: Performance, Static Stability, Dynamic Stability, and Classical Feedback Control*, American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia.
 - [44] J. Roskam (2003), *Airplane Flight Dynamics and Automatic Flight Controls*, Vol. 2, Analysis and Research Corporation (DARCorporation), Lawrence KS, USA.
 - [45] P. Narayan, P. Wu, D. Campbell, and R. Walker (2007), An Intelligent Control Architecture for Unmanned Aerial Systems (UAS) in the National Airspace System (NAS), *Proceedings 2nd International Unmanned Air Vehicle Systems Conference*, pp. 1-11, Grand Hyatt, Melbourne, Australia.
 - [46] S. Griffiths, J. Saunders, A. Curtis, B. Barber, T. McLain, and R. Beard (2007), Obstacle and Terrain Avoidance for Miniature Aerial Vehicles, In: *Advances in Unmanned Aerial Vehicles - State of the Art and the Road to Autonomy*, K. P. Valavanis, (Ed), pp. 213-244, Springer, Netherlands.
 - [47] A. Lucas, R. Rönquist, P. Rechter, C. Heinze, S. V. D. Velden, I. Powlesland, and S. Karim (2004), Development and Flight Testing Of an Intelligent, Autonomous UAV Capability, *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit*, Chicago, Illinois, 20-23 September.
 - [48] W. N. I. Aryo (2008), Deployment of Autonomous UAV by Means of Simulation, Master of Engineering (Research) Dissertation, Robotic Research Centre, Nanyang Technological University, Singapore.
 - [49] J. Rasmussen (1983), Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distractions in Human Performance Models, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 13, pp. 257-266.
 - [50] M. L. Cummings and P. J. Mitchell (2006), Automated Scheduling Decision Support for Supervisory Control of Multiple UAVs, *Journal of Aerospace Computing, Information, and Communication*, Vol. 3, No. 6, pp. 294-308.
-

-
- [51] MicroPilot (2004), MicroPilot Autopilot Installation and Operation, *Technical Manual*, Manitoba, Canada.
 - [52] Piccolo II Specification, Cloud Cap Technology.
 - [53] Kestrel Autopilot (Version 2.2) Specification, Procerus Technologies.
 - [54] J. H. Low, J. Navabalachandran, and G. Leng (2005), Autopilot Integration for a Flying Wing UAV, *Technical Report*, National University, Singapore.
 - [55] G. Platanitis and S. Shkarayev (2005), Integration of an Autopilot for a Micro Air Vehicle, *Infotech@Aerospace*, Arlington, Virginia, 26-29 September.
 - [56] T. Coffey and J. A. Montgomery (2002), The Emergence of mini UAVs for Military Applications, *Defense Horizon*, Vol. 22, Center for Technology and National Security Policy, National Defense University.
 - [57] H. Chao, Y. Cao, and Y. Chen (2007), Autopilots for Small Fixed-Wing Unmanned Air Vehicles: A Survey, *Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation*, Harbin, China, 5-8 August.
 - [58] E. Santamaria, P. Royo, J. Lopez, C. Barrado, E. Pastor, and X. Prats (2007), Increasing UAV Capabilities through Autopilot and Flight Plan Abstraction, *IEEE/AIAA 26th Digital Avionics Systems Conference, DASC 07*, pp. 5.B.5-1 to 5.B.5-10, Castelldefels (Barcelona), Spain, 21-25 October.
 - [59] S. Griffiths, J. Saunders, A. Curtis, B. Barber, T. McLain, and R. Beard (2006), Maximizing Miniature Aerial Vehicles, *Robotics & Automation Magazine*, Vol. 13, pp. 34-43, Institute of Electrical and Electronics Engineers, Inc.
 - [60] M. Quigley, M. A. Goodrich, S. Griffiths, M. Eldredge, and R. W. Beard (2005), Target Acquisition, Localization, and Surveillance using a Fixed-Wing Mini-UAV and Gimbaled Camera, *Proceedings of the IEEE International Conference on Robotics and Automation*, pp 2600 – 2605, Barcelona, Spain, 18-22 April.
 - [61] J. B. Saunders, B. Call, A. Curtis, R. W. Beard, and T. W. McLain (2005), Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles, *Infotech@Aerospace*, Arlington, Virginia, 2005, 26-29 September.
 - [62] J. Zhe, H. Jianda, W. Yuechao, and S. Qi (2006), Enhanced LQR Control for Unmanned Helicopter in Hover, *Systems and Control in Aerospace and Astronautics*, pp. 1438-1443, 19-21 January.
 - [63] M. Kumon, Y. Udo, H. Michihira, M. Nagata, I. Mizumoto, and Z. Iwai (2007), Autopilot System for Kiteplane, *IEEE/ASME Transactions and Mechatronics*, Vol. 11, pp. 615-624, October. .
-

-
- [64] B. N. Pamadi (1998), *Performance, Stability, Dynamics, and Control of Airplanes*, American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia.
 - [65] Y. Lee, S. Kim, J. Suk, H. Koo, and J. Kim (2002), System Identification of an Unmanned Aerial Vehicle from Automated Flight Tests, *AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles*, Portsmouth, Virginia, 20-23 May.
 - [66] N. M. Jodeh, P. A. Blue, and A. A. W. Waldron (2006), Development of Small Unmanned Aerial Vehicle Research Platform: Modeling and Simulating with Flight Test Validation, *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Keystone, Colorado, 1-24 August.
 - [67] W. S. Yeung, P. Placido Jr, E. Tang, S. Wasi, A. Genova, D. Wigley, and F. Khorrami (2006), The Anatomy of Athena: The Engineering Behind Polytechnic University's UAV, *Journal Paper for the 4th Annual Student UAV Competition 2006*, Polytechnic University, New York, USA.
 - [68] J. E. Williams and R. V. Steven (1979), The USAF Stability and Control Digital DATCOM – Implementation of Datcom Methods, Vol 2.
 - [69] Advanced Aircraft Analysis Software, Darcorporation.
 - [70] J. Roskam (1985), *Airplane Design*, Vol. 1-8, Roskam Aviation and Engineering Corporation.
 - [71] T. Melin (2000), *A Vortex Lattice MATLAB Implementation for Linear Aerodynamic Wing Applications*, Master Dissertation, Department of Aeronautics, Royal Institute of Technology, Sweeden.
 - [72] ANSYS Fluent Software, Fluent Inc. Joins ANSYS, Inc.
 - [73] V. Klein and E. A. Morelli (2006), *Aircraft System Identification: Theory and Practice*, American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia.
 - [74] R. V. Jategaonkar (2006), *Flight Vehicle System Identification: A Time Domain Methodology*, American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia.
 - [75] H. D. Shim, H. J. Kim, and S. Sastry (2003), A Flight Control System for Aerial Robots: Algorithms and Experiments, *Control Engineering Practice*, Vol. 11, No. 12, pp. 1389-1400.
 - [76] R. Krashanitsa, G. Platanitis, B. Silin, and S. Shkarayev (2006), Aerodynamics and Control Design for Autonomous Micro Air Vehicle, *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, Keystone, Colorado, 21-24 August.
-

-
- [77] M. Manaï and A. Desbiens (2005), Identification of a UAV and Design of a Hardware-in-the-Loop System for Nonlinear Control Purposes, *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, 15-18 August.
 - [78] Satellite image of urban terrain in Singapore, Google Earth.
 - [79] S. Saripalli, D. J. Naffin, and G. S. Sukhatme (2002), Autonomous Flying Vehicle Research at the University of Southern California, in *Multi-Robot Systems: From Swarms to Intelligent Automata, Proceedings of the First International Workshop on Multi-Robot Systems*, pp. 73-82.
 - [80] R. Murphy (2000), *Introduction to AI Robotics*, MIT Press, Cambridge.
 - [81] D. Shim, H. Chung, H. Kim, and S. Sastry (2005), Autonomous Exploration In Unknown Urban Environments For Unmanned Aerial Vehicles, *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, California, 15-18 August.
 - [82] L. Muratet, S. Doncieux, Y. Briere, and J. A. Meyer (2005), A Contribution to Vision-Based Autonomous Helicopter Flight in Urban Environments, In: *Robotics and Autonomous Systems*, Vol. 50, No.4, pp. 195-209.
 - [83] L. Muratet, S. Concieux, and J. A. Meyer (2004), A Biomimetic Reactive Navigation System Using the Optical Flow for a Rotary-Wing UAV in Urban Environment, *Proceedings of the International Session on Robotics*, March.
 - [84] S. Hrabar, G. S. Sukhatme, P. Corke, K. Usher, and J. Roberts (2005), Combined Optic-Flow and Stereo-Based Navigation of Urban Canyons for A UAV, *International Conference on Intelligent Robots and Systems*, pp. 3309-3316, Edmonton, Alberta, Canada, 2-6 August.
 - [85] S. Hrabar and G. S. Sukhatme (2004), A Comparison of Two Camera Configurations for Optic-flow Based Navigation of a UAV Through Urban Canyons, *International Conference on Intelligent Robots and Systems*, Vol.3, pp. 2673-2680, Sendai, Japan, 28 September - 2 October.
 - [86] T. C. Camus (1994), "Real-time Optical Flow," Brown University, Providence, RI, USA.
 - [87] D. Y. Hu (2007), *Integrating Autopilot Capability to the Kyosho M36 Spree Sports*, BEng Final Year Project Report, Robotic Research Centre, Nanyang Technological University, Singapore.
 - [88] T. C. Nguyen (2008), *Development of Unmanned Aerial Vehicle with Analytical Method Controller Design*, Final Year Project Report, Robotic Research Centre, Nanyang Technological University, Singapore.
-

-
- [89] R. W. Pratt (2000), *Flight Control Systems - Practical Issues in Design and Implementation*, Institute of Electrical and Electronics Engineers, Inc.
 - [90] A. Rowe, A. Goode, and D. Goel (2007), CMUcam3: An Open Programmable Embedded Vision Sensor, *Technical Report*, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.
 - [91] Opti-Logic RS400 Industrial Laser Range Finder, Optics Planet.
 - [92] MicroPilot (2004), XTenderTM Programmer User's Guide, *Technical Manual*, Manitoba, Canada.
 - [93] H. A. Yanco, M. Baker, R. Casey, B. Keyes, P. Thoren, J. Drury, D. Few, C. Nielsen, and D. Bruemmer (2006), Analysis of Human-Robot Interaction for Urban Search and Rescue, *Proceeding of the 2006 IEEE International Workshop on Safety, Security and Rescue Robotics*, Gaithersburg, MD.
 - [94] Red Hat, Inc (2003), Cygwin User's Guide, *Technical Manual*.
 - [95] D. Murali (2007), Aircraft Static Stability and Control - Static Directional and Lateral Stability of an Aircraft, *Lecture Notes of Flight Mechanics and Control*, Nanyang Technological University, Singapore.
 - [96] T. Wagner and J. Valasek (2005), Comparison of Computational Methods for Stability and Control Analysis, *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 10-13 January.
 - [97] The MathWorks Inc. (2007), User's Guide to Control System Toolbox, *Technical Manual*, United States.
 - [98] T. Foster and J. Bowman (2005), Dynamic Stability and Handling Qualities of Small Unmanned Aerial Vehicles, *43rd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 10-13 January.
 - [99] S. Y. Yoon, S. K. Lee, J. Y. K. Kong, and K. W. Name (2004), Development of a HLA-Based Simulation Game for RC Airplanes and UAVs, *AIAA Modeling and Simulation Technologies Conference and Exhibit*, Providence, Rhode Island, 16-19 August.
 - [100] D. Jung and P. Tsiotras (2007), Modeling and Hardware-in-the-Loop Simulation for a Small Unmanned Aerial Vehicle, *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, Rohnert Park, California, 7-10 May.
 - [101] J. H. Blakelock (1991), *Automatic Control of Aircraft and Missiles*, 2nd edition. John Wiley & Sons, Inc, United States.
-

- [102] M. Mierau, J. Tsai, and P. Haintz (2002), Helicopter Flight Automation, *Competition Paper Entry for International Aerial Robotics Competition*, Simon Fraser University, Canada.
- [103] G. D. PadField (2007), *Helicopter Flight Dynamics*, 2nd edition, American Institute of Aeronautics and Astronautics, Inc, North America.
- [104] I. H. Lee, D. J. L. Lee, and H. B. Ban (2007), A Small Scale Rotor UAV Autonomous Flight Control System Design and Verification, *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, Rohnert Park, California, 7-10 May.

Appendix

Appendix A: SUAV Technical Information

Appendix B: Fixed-Wing UAV Equations of Motion

Appendix C: Rainbow SUAV Results

Appendix D: M36 SUAV Results

Appendix E: ECO-8 SUAV Results

Appendix F: MicroPilot PID Gains Conversion Ratios

Appendix G: Pre-Flight and Post Flight Checklists

Appendix A: SUAV TECHNICAL INFORMATION

A.1 SUAVs Hardware Circuits Connection

This section includes the hardware circuit connections for the M36, the Rainbow and the ECO-8 SUAVs.

Figure A-1: M36 Hardware Connection

Figure A-2: Rainbow Hardware Connection

Figure A-3: ECO-8 Hardware Connection

A.2 Data Written in the Data Log File

The sensors, which are built-in the MP2128^g autopilot module, are given in the following table.

Table A-1: Built-In Sensors and Instrumentations in MP2128^g module

Components	Brand/Model	Basic Specifications
Microprocessor	Freescale MCF5233CVM	<ul style="list-style-type: none">150 MIPS
Global Positioning System (GPS)	Trimble	<ul style="list-style-type: none">Uses TSIP Protocol
Gyroscope	iMEMS ADXL203	<ul style="list-style-type: none">To measure the x, y, z direction angular acceleration rate.
Accelerometer	iMEMS ADXRS150	<ul style="list-style-type: none">To measure the x, y, z direction acceleration.
Static Pressure Sensor	Motorola MPX4115AP	<ul style="list-style-type: none">Altitude Measurement
Dynamic Pressure Sensor	Motorola MPXV5004DP	<ul style="list-style-type: none">Airspeed Measurement

During flight, the UAV states and health statuses are recorded in a datalog. The data is recorded in the rate of either 30Hz or 5Hz. Table A-2 tabulates the states and statues that are saved in the log file.

Table A-2: SUAV states available on SUAV MicroPilot

Attitude	Roll, pitch, yaw
Rate of Attitude	Roll rate, pitch rate, yaw rate
Altitude	GPS altitude, Altitude from pressure sensor, Above ground
Airspeed	Forward airspeed
Location	GPS easting, GPS northing, easting relative position, northing relative position

Appendix B: Fixed-Wing UAV Equations of Motion

This appendix explains the derivation of the fixed-wing aircraft equation of motion (EOM). The linearised EOM will be derived from first principal. More details could be found in the reference by Pamadi [61]. The derivations of the EOM are based on a puller-type UAV with the propulsion system mounted on the nose of the fuselage. The control surfaces available on the SUAV are the aileron, elevator and rudder.

B.1 Coordinate Systems

In flight mechanics, three coordinate systems are frequently used, i.e. the:

- (i) North-East-down axis (or the global reference frame)
- (ii) Body-attached coordinate system (relative to the SUAV body), and
- (iii) Wind-Axis frame (relative to the wind direction).

The north-east-down axis is a right-orthogonal axis system location on the surface of the earth. The X-axis is pointing to north, the Y-axis points towards east and the Z-axis is pointing downwards. This axis system is used to describe navigation and localization. The second and third coordinate systems are also right-orthogonal axis system. Both of the coordinate systems are illustrated in Figure B-1.

Figure B-1: Aircraft coordinate systems

B.2 Equations of Motion

Assuming that the airplane is a rigid body, the forces and moments acting on the aircraft are due to the aerodynamic forces and thrust force from the propulsion. By applying Newton's Second Law of Motion, the 6-DOF equation of motions can be derived by considering the forces and moment acting on the center of gravity of the aircraft. It is assumed that the propulsion force is acting along the body x-axis of the UAV, thus, the force only affects the forward motion and has no effect on other axes. The following six equations are the forces and moments equations. These equations are arranged so that the left-hand side of the equations is the either aerodynamic force or moment acting on the platform.

Force equations:

$$F_x = m\dot{u} - m(rv - qw) + mg \sin \theta - T \quad (\text{B-1})$$

$$F_y = m\dot{v} - m(pw - ru) - mg \cos \theta \sin \phi \quad (\text{B-2})$$

$$F_z = m\dot{w} - m(qu - pv) - mg \cos \theta \cos \phi \quad (\text{B-3})$$

Moment equations:

$$L = I_{xx}\dot{p} - I_{xz}(\dot{r} + pq) + (I_{xx} - I_{yy})qr \quad (\text{B-4})$$

$$M = I_{yy}\dot{q} + I_{xz}(p^2 - q^2) - (I_{xx} - I_{zz})pr \quad (\text{B-5})$$

$$N = I_{zz}\dot{r} - I_{xz}(\dot{p} - qr) + (I_{yy} - I_{zz})pq \quad (\text{B-6})$$

The aerodynamic forces and moments can be modeled using the dimensionless coefficients at a given flight condition. These forces and moments are proportional to the dynamic pressure generated due to the motion of the UAV. Thus, these six forces can be represented in the following equations:

Aerodynamic forces:

$$F_x = C_x \bar{q} S \quad F_y = C_y \bar{q} S \quad F_z = C_z \bar{q} S \quad (\text{B-7})$$

Aerodynamic Moments:

$$L = C_l \bar{q} S b \quad M = C_m \bar{q} S \bar{c} \quad N = C_n \bar{q} S \bar{c} \quad (\text{B-8})$$

From the above twelve equations, it can be seen that the aerodynamic forces and moments have a significant influence on the aircraft motion. Each of these dimensionless coefficients relating to these forces and moments will be investigated here.

The six dimensionless coefficients are i.e. C_x , C_y , C_z , C_m , C_n , and C_l . These coefficients are represented with respect with the body axis coordinate system. However, in aerodynamic analysis, it is easier to perform analysis in the wind-axis coordinate system. The relationship between the coefficients represented in body axis and wind axis are represented in (B-9). Only C_x and C_z coefficients are required for the transformation because the body and wind coordinate systems share the same axis for other coefficient.

$$\begin{Bmatrix} C_D \\ C_L \end{Bmatrix} = \begin{bmatrix} -\cos \alpha & -\sin \alpha \\ \sin \alpha & -\cos \alpha \end{bmatrix} \begin{Bmatrix} C_x \\ C_z \end{Bmatrix} \quad (\text{B-9})$$

In the forces components, the coefficients C_D , C_Y , and C_L are the drag, side-slip, lift coefficients. On the other hand, the coefficients C_m , C_n , and C_l are the longitudinal, cross-direction, and lateral stabilities coefficients respectively [1]. These coefficients are a function of a number of variables and they are related in a nonlinear correlation between each variable [2]. In practice, it is convenient to represent these coefficients as a sum of the contributing components [1, 2]. These contributing components are the stability and control derivatives (SCD) coefficients as depicted in the equations (B-10) and (B-11).

$$\begin{Bmatrix} C_D \\ C_L \\ C_m \end{Bmatrix} = \begin{Bmatrix} C_{D_0} \\ C_{L_0} \\ C_{m_0} \end{Bmatrix} + \begin{bmatrix} C_{D\alpha} & C_{D\delta e} \\ C_{L\alpha} & C_{L\delta e} \\ C_{m\alpha} & C_{m\delta e} \end{bmatrix} \begin{Bmatrix} \alpha \\ \delta_e \end{Bmatrix} \quad (\text{B-10})$$

$$\begin{Bmatrix} C_Y \\ C_l \\ C_n \end{Bmatrix} = \begin{Bmatrix} C_{Y_0} \\ C_{l_0} \\ C_{n_0} \end{Bmatrix} + \begin{bmatrix} C_{Y\beta} & C_{Y\delta a} & C_{Y\delta r} \\ C_{l\beta} & C_{l\delta a} & C_{l\delta r} \\ C_{n\beta} & C_{n\delta a} & C_{n\delta r} \end{bmatrix} \begin{Bmatrix} \beta \\ \delta_a \\ \delta_r \end{Bmatrix} \quad (\text{B-11})$$

From the above equation, it shows that the aerodynamic coefficients are affected by the control surfaces (the elevator, aileron, and rudder). Any changes of these control surfaces will induce a change in the coefficients, thus affecting the motion of the aircraft. The model of an aircraft is basically generic however; the stability and control derivatives (SCD) coefficients are not. They are dependent on the geometries of the aircraft and the flight conditions. Thus, these values have to be evaluated or quantified for each plane and at each flight condition. Equation (B-12) assembles the SCD into a matrix of unknown parameters, Θ that has to be quantified and the next section will present in more details of the quantification procedures.

$$\Theta = \begin{Bmatrix} C_{D_0}, C_{D_\alpha}, C_{D_{\delta e}}, C_{L_0}, C_{L_\alpha}, C_{L_{\delta e}}, C_{Y_0}, C_{Y_\beta}, C_{Y_{\delta a}}, C_{Y_{\delta r}}, C_{l_0}, \\ C_{l_\beta}, C_{l_{\delta a}}, C_{l_{\delta r}}, C_{m_0}, C_{m_\alpha}, C_{m_{\delta e}}, C_{n_0}, C_{n_\beta}, C_{n_{\delta a}}, C_{n_{\delta r}} \end{Bmatrix} \quad (\text{B-12})$$

B.3 Linear Equation of Motion

The equations here are adapted and modified from Pamadi [3]. State-space representation:

$$\dot{X} = AX + BU \quad (\text{B-13})$$

The state-space equations for the longitudinal and lateral are depicted in equations (B-14) and equation (B-15) respectively.

Longitudinal state-space representation:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{41} \end{bmatrix} \quad (B-14)$$

$$X = \begin{Bmatrix} u \\ \Delta\alpha \\ q \\ \Delta\theta \end{Bmatrix} \quad U = \Delta\delta_e$$

$$a_{11} = \frac{C_{xu} + \xi_1 C_{zu}}{m_1}$$

$$a_{32} = \frac{C_{m\alpha} + \xi_2 C_{z\alpha}}{I_{y1}}$$

$$a_{12} = \frac{C_{x\alpha} + \xi_1 C_{z\alpha}}{m_1}$$

$$a_{33} = \frac{c_1 + \xi_2(m_1 + C_{zq}c_1)}{I_{y1}}$$

$$a_{13} = \frac{C_{xq}c_1 + \xi_1(m_1 + C_{zq}c_1)}{m_1}$$

$$a_{34} = \frac{\xi_2 C_{z\theta}}{I_{y1}}$$

$$a_{14} = \frac{C_{x\theta} + \xi_1 C_{z\theta}}{m_1}$$

$$a_{41} = 0$$

$$a_{21} = \frac{C_{zu}}{m_1 - C_{z\dot{\alpha}}c_1}$$

$$a_{42} = 0$$

$$a_{43} = 1$$

$$a_{22} = \frac{C_{z\alpha}}{m_1 - C_{z\dot{\alpha}}c_1}$$

$$a_{44} = 0$$

$$a_{23} = \frac{m_1 + C_{zq}c_1}{m_1 - C_{z\dot{\alpha}}c_1}$$

$$b_{11} = \frac{C_{x\delta e} + \xi_1 C_{z\delta e}}{m_1}$$

$$a_{24} = \frac{C_{z\theta}}{m_1 - C_{z\dot{\alpha}}c_1}$$

$$b_{21} = \frac{C_{z\delta e}}{m_1 - C_{z\dot{\alpha}}c_1}$$

$$a_{31} = \frac{C_{mu} + \xi_2 C_{zu}}{I_{y1}}$$

$$b_{31} = \frac{C_{m\delta e} + \xi_2 C_{z\delta e}}{I_{y1}}$$

$$b_{41} = 0$$

Lateral state-space representation:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \\ b_{41} & b_{42} \\ b_{51} & b_{52} \end{bmatrix} \quad (B-15)$$

$$X = \begin{Bmatrix} \Delta\beta \\ \Delta\phi \\ \dot{p} \\ \Delta\psi \\ \dot{r} \end{Bmatrix} \quad U = \begin{Bmatrix} \Delta\delta_a \\ \Delta\delta_r \end{Bmatrix}$$

$$\begin{aligned} a_{11} &= \frac{C_{y\beta}}{m_1 - b_1 C_{y\dot{\beta}}} & a_{54} &= 0 \\ a_{12} &= \frac{C_{y\phi}}{m_1 - b_1 C_{y\dot{\beta}}} & a_{55} &= b_1(C_{nr}I'_{x1} + C_{lr}I'_{xz1} + \xi_2 a_{15}) \\ a_{13} &= \frac{C_{yp}b_1}{m_1 - b_1 C_{y\dot{\beta}}} & b_{11} &= \frac{C_{y\delta_a}}{m_1 - b_1 C_{y\dot{\beta}}} \\ a_{14} &= 0 & b_{12} &= \frac{C_{y\delta_r}}{m_1 - b_1 C_{y\dot{\beta}}} \\ a_{15} &= -\frac{(m_1 - b_1 C_{yr})}{(m_1 - b_1 C_{y\dot{\beta}})} & b_{21} &= 0 \\ a_{21} &= 0 & b_{22} &= 0 \\ a_{22} &= 0 & b_{31} &= C_{l\delta_a}I'_{z1} + C_{n\delta_a}I'_{xz1} + \xi_1 b_1 b_{11} \\ a_{23} &= 1 & b_{32} &= C_{l\delta_r}I'_{z1} + C_{n\delta_r}I'_{xz1} + \xi_1 b_1 b_{12} \\ a_{24} &= 0 & b_{41} &= 0 \\ a_{25} &= 0 & b_{42} &= 0 \\ a_{31} &= C_{l\beta}I'_{z1} + C_{n\beta}I'_{xz1} + \xi_1 b_1 a_{11} & b_{51} &= C_{n\delta_a}I'_{x1} + C_{l\delta_a}I'_{xz1} + \xi_2 b_1 b_{11} \\ a_{32} &= \xi_1 b_1 a_{12} & b_{52} &= C_{n\delta_r}I'_{x1} + C_{l\delta_r}I'_{xz1} + \xi_2 b_1 b_{12} \\ a_{33} &= C_{lp}b_1 I'_{z1} + C_{np}b_1 I'_{xz1} + \xi_1 b_1 a_{13} & \xi_1 &= C_{l\beta}I'_{z1} + C_{n\beta}I'_{xz1} \\ a_{34} &= 0 & \xi_2 &= C_{n\beta}I'_{x1} + C_{l\beta}I'_{xz1} \\ a_{35} &= C_{lr}b_1 I'_{z1} + C_{nr}b_1 I'_{xz1} + \xi_1 b_1 a_{15} & b_1 &= \frac{b}{2U_o} \\ a_{41} &= a_{42} = a_{43} = a_{44} = 0 & I'_{x1} &= \frac{I_{x1}}{I_{x1}I_{z1} - (I_{xz1})^2} \\ a_{45} &= 1 & I'_{z1} &= \frac{I_{z1}}{I_{x1}I_{z1} - (I_{xz1})^2} \\ a_{51} &= C_{n\beta}I'_{x1} + C_{l\beta}I'_{xz1} + \xi_2 b_1 a_{11} & I'_{xz1} &= \frac{I_{xz1}}{I_{x1}I_{z1} - (I_{xz1})^2} \\ a_{52} &= \xi_2 b_1 a_{12} \\ a_{53} &= b_1(C_{np}I'_{x1} + C_{lp}I'_{xz1} + \xi_2 a_{13}) \end{aligned}$$

Appendix C: Rainbow SUAV Results

The result for the Rainbow SUAV mathematical model was provided in Section 5.1.2 and Section 5.1.3. This appendix includes the aerodynamic coefficients for unit step responses for the feedback loops that were not included in the earlier section.

C.1 Unit Step Responses

This section includes the unit step responses for the airspeed hold and pitch hold.

Figure C-1: Unit Step Response for Pitch Hold

Figure C-2: Unit Step Response for Airspeed Hold

C.2 Aerodynamic Coefficients for Different CG Locations

Table C-1 tabulates the aerodynamic coefficients of the Rainbow SUAV for three different CG locations as discussed in Section 5.1.4 for the longitudinal direction.

Table C-1: Rainbow SUAV aerodynamic coefficients for the longitudinal direction

States	CG_Original			CG_Behind			CG_InFront		
	C_D	C_L	C_m	C_D	C_L	C_m	C_D	C_L	C_m
SS^*	0.0471	0.5819	-0.0112	0.0466	0.5819	-0.0112	0.0477	0.5819	-0.0112
u	0.0883	0.0011	0.0005	0.0883	0.0011	0.0005	0.0883	0.0011	0.0005
α	0.2947	5.1909	-2.6345	0.2947	5.1909	-2.2415	0.2947	5.1909	-3.0276
$\dot{\alpha}$	0.0000	2.0504	-6.7518	0.0000	2.0032	-6.4448	0.0000	2.0975	-7.0659
q	0.0000	10.0121	-21.726	0.0000	9.1792	-20.517	0.0000	10.8451	-23.039
δ_e	0.0281	0.4864	-1.6017	0.0281	0.4864	-1.5649	0.0269	0.2965	-0.9988

Notation: * SS indicates coefficients evaluated at the steady state

Table C-2 tabulates the aerodynamic coefficients of the Rainbow SUAV for three different CG locations as discussed in Section 5.1.4 for the lateral direction.

Table C-2: Rainbow SUAV aerodynamic coefficients for the lateral direction

States	CG_Original			CG_Behind			CG_InFront		
	C_Y	C_l	C_n	C_Y	C_l	C_n	C_Y	C_l	C_n
β	-0.2314	-0.0707	0.0204	-0.2314	-0.0708	0.0179	-0.2314	-0.0706	0.0229
$\dot{\beta}$	-0.0331	0.0001	-0.0136	-0.0331	0.0001	-0.0136	-0.0332	0.0002	-0.0136
p	-0.0442	-0.4738	-0.0710	-0.0446	-0.4737	-0.0692	-0.0438	-0.4740	-0.0728
r	0.1118	0.1770	-0.0616	0.1092	0.1733	-0.0583	0.1143	0.1808	-0.0652
δ_a	0.0000	0.0910	-0.0082	0.0000	0.0910	-0.0080	0.0000	0.0910	-0.0084
δ_r	0.0212	0.0007	-0.0099	0.0212	0.0007	-0.0097	0.0212	0.0007	-0.0101

Appendix D: M36 SUAV Results

This appendix includes the mathematical model and the PID gains obtained for the M36 SUAV results.

D.1 Physical Properties and Aerodynamic Coefficients

Figure D-1 shows the CG location for the M36 SUAV estimated using the SolidWorks™ software.

Figure D-1: M36 center of gravity location

The following table gives the moments of inertia estimated using the SolidWork™ module. Similar to the Rainbow SUAV platform, the I_{xy} and I_{yz} are assumed to be zero.

Table D-1: Fixed-wing SUAV moments of inertia

Moment of Inertia	Value	Moment of Inertia	Value
$I_{xx} (\times 10^2 \text{ kgm}^2)$	2.871	$I_{xz} (\times 10^2 \text{ kgm}^2)$	-0.096
$I_{yy} (\times 10^2 \text{ kgm}^2)$	4.114	$I_{xy} (\times 10^2 \text{ kgm}^2)$	0.000
$I_{zz} (\times 10^2 \text{ kgm}^2)$	6.780	$I_{yz} (\times 10^2 \text{ kgm}^2)$	0.000

Table D-2 tabulates the flight conditions for estimating the aerodynamic coefficients for M36 SUAV.

Table D-2: M36 Flight Conditions

Flight Condition	Values
Altitude, h_o	120 m
Airspeed, u_o	22 ms^{-1}
Angle of attack, α_o	-3.022°

Based on the flight conditions, the aerodynamic coefficients estimated from AAA software module are given in Table D-3.

Table D-3: M36 Aerodynamic Coefficients

Longitudinal Direction				Lateral Direction			
States	C_D	C_L	C_m	States	C_Y	C_l	C_n
SS^*	0.0245	0.1991	0.0000	β	-0.1890	0.2744	0.0460
u	0.0000	0.0008	0.0002	$\dot{\beta}$	0.0034	0.0004	0.0016
α	0.1196	4.7583	-0.1711	p	-0.0340	-0.4285	-0.0183
$\dot{\alpha}$	0.0000	1.2668	-3.1626	r	0.1112	0.0669	-0.0515
q	0.0000	3.9000	-9.3338	δ_a	0.0000	0.0671	-0.0024
δ_e	0.0113	0.2969	-0.7412	δ_r	0.0928	0.0085	-0.0442

Notation: * SS indicates coefficients evaluated at the steady state

Assembling the physical properties from Table D-1 and aerodynamic coefficients from Table D-3, the linear equations of motion for the M36 SUAV were derived as depicted in equations (D-1) and (D-2).

$$\begin{Bmatrix} \dot{u} \\ \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{Bmatrix} = \begin{bmatrix} -0.1093 & 0.1772 & 0 & -0.4437 \\ -0.8780 & -10.5262 & 3.1019 & 0 \\ 4.3670 & -6.7204 & -19.0721 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{Bmatrix} u \\ \alpha \\ q \\ \theta \end{Bmatrix} + \begin{bmatrix} -0.0252 \\ -0.6534 \\ -248.9834 \\ 0 \end{bmatrix} \Delta \delta_e \quad (\text{D-1})$$

$$\text{where, } u = \frac{\Delta U}{U_o}, \quad q = \frac{d\theta}{dt}$$

$$\begin{Bmatrix} \dot{\beta} \\ \dot{\phi} \\ \dot{p} \\ \dot{\psi} \\ \dot{r} \end{Bmatrix} = \begin{bmatrix} -0.4213 & 0.4438 & -0.0021 & 0 & -0.9934 \\ 0 & 0 & 1 & 0 & 0 \\ 801.1169 & 0.0128 & -34.1785 & 0 & 5.3695 \\ 0 & 0 & 0 & 0 & 1 \\ 45.6484 & 0.0231 & -0.1359 & 0 & -1.8687 \end{bmatrix} \begin{Bmatrix} \beta \\ \phi \\ p \\ \psi \\ r \end{Bmatrix} + \begin{bmatrix} 0 & 0.2069 \\ 0 & 0 \\ 196.3080 & 26.6382 \\ 0 & 0 \\ -5.7919 & -55.1088 \end{bmatrix} \begin{Bmatrix} \Delta \delta_a \\ \Delta \delta_r \end{Bmatrix} \quad (\text{D-2})$$

$$\text{where, } p = \frac{d\phi}{dt}, \quad r = \frac{d\psi}{dt}$$

D.2 Controller Gains

The transfer functions derived from the state-space equations (D-1) and (D-2) for the *pitch from airspeed*, *roll from heading*, *elevator from pitch* and *aileron from roll* loops are given in Table D-4.

Table D-4: Transfer Functions

Control loop	Transfer Function
Pitch from airspeed	$\frac{u(s)}{\Delta\theta(s)} = \frac{0.0252s^3 + 0.8611s^2 + 34.15s - 1161}{249s^2 + 2644s + 326.5}$ (D-3)
Roll from heading	$\frac{\gamma(s)}{\Delta\phi(s)} = \frac{0.4459}{s}$ (D-4)
Elevator from pitch	$\frac{\Delta\theta(s)}{\Delta\delta_e(s)} = \frac{-249s^2 - 2644s - 326.5}{s^4 + 29.71s^3 + 225s^2 + 26.73s + 23.02}$ (D-5)
Aileron from roll	$\frac{\Delta\phi(s)}{\Delta\delta_a(s)} = \frac{196.3s^2 + 418.4s + 13650}{s^4 + 36.47s^3 + 126.8s^2 + 1117s - 755.5}$ (D-6)

Based on the transfer functions derived and the similar approach presented in Section 5.1.3, was taken to tune the PID gains. The PID gains results are tabulated in Table D-5. The unit step responses for each of the loop will be presented in next section.

Table D-5: Results - feedback control PID gains for M36 SUAV

Feedback Loops	SI Unit			MicroPilot Unit		
	K_P	K_I	K_D	K_P	K_I	K_D
Aileron from roll	1.5572	0.0057	0.5745	-130000	-128	-3000
Elevator from pitch	-6.1330	-0.4402	-0.2132	78000	4600	8900
Pitch from airspeed	-0.0861	-7.509x10 ⁻⁴	0.0000	27500	48	0
Roll from heading	0.6830	0.0000	0.8196	-1000	0	-150

D.3 Unit Step Responses

This section includes the unit step responses for the four feedback loops. The gains tabulated in Table D-5 are used for the PID controllers.

Figure D-2: Unit step response for the *roll-hold*

Figure D-3: Unit step response for the *heading-hold*

Figure D-4: Unit step response for the *pitch*-hold

Figure D-5: Unit step response for the *airspeed*-hold

Appendix E: ECO-8 SUAV Results

The PID gains determined by the manufacturer for ECO-8 SUAV are tabulated in Table E-1.

Table E-1: Results - feedback control PID gains for M36 SUAV

Feedback Loops	MicroPilot Unit		
	K_P	K_I	K_D
Aileron from roll	-95000	-800	-275
Elevator from pitch	25000	600	1600
Pitch from airspeed	950	5	1200
Hover rudder from heading	-11500	-85	-1100
Hover roll from Y Velocity	1900	8	-96
Hover roll from X velocity	-1600	-8	96

Appendix F: MicroPilot PID Gains Conversion Ratios

The MicroPilot autopilot module uses unique units to implement the PID controller gains. This appendix provides the conversion ratio as following.

$$K_{MP} = \mathfrak{R} \cdot K_{SI} \quad (F-1)$$

Referring to equation (F-1), the followings are defined;

$$\begin{aligned} K_{MP} &= \text{Gains implemented by autopilot} \\ K_{SI} &= \text{Gains in SI unit} \\ \mathfrak{R} &= \text{Conversion Ratio} \end{aligned}$$

Based on the equation give in (F-1), the conversion ratios can be derived [4]. The equations used to derive the conversion ratios are given as followings:

Loop: Aileron from roll and elevator from pitch

$$\begin{aligned} K_{mp} &= \frac{K_{SI} \times \left(\frac{180}{\pi}\right)}{100} \times \left(\frac{2^a \times 32767}{1024}\right) \times \frac{1}{b} \\ [K_{SI}] &= \% \text{ deflection/deg} \\ [K_{MP}] &= \text{Fine servo unit/rad} \end{aligned} \quad (F-2)$$

Loop: Roll from heading

$$\begin{aligned} K_{mp} &= \left(K_{SI} \times \frac{\pi}{180}\right) \times \left(\frac{2^a \times 1024}{100}\right) \times \frac{1}{b} \\ [K_{SI}] &= \text{deg/deg} \\ [K_{MP}] &= \text{rad/deg} \end{aligned} \quad (F-3)$$

Loop: Pitch from Airspeed

$$\begin{aligned} K_{mp} &= \left(K_{SI} \times 0.3048 \times \frac{\pi}{180}\right) \times \left(\frac{2^a \times 1024}{100}\right) \times \frac{1}{b} \\ [K_{SI}] &= \text{deg/ms}^{-1} \\ [K_{MP}] &= \text{Fine servo unit/rad} \end{aligned} \quad (F-4)$$

Loop: Throttle from Altitude

$$K_{mp} = \left(\frac{K_{SI} \times 0.3048}{100} \right) \times \left(\frac{2^a \times 32767}{8} \right) \times \frac{1}{b}$$

(F-5)

$[K_{SI}] = \% \text{ deflection/meter}$

$[K_{MP}] = \text{Fine servo unit/meter}$

The value ‘a’ and ‘b’ values shown in equation (F-1) to (F-5) can be found from the following table:

Table F-1: Multiplication and divisor factors

Loops	Multiplication, <i>a</i>			Divisor, <i>b</i>		
	P Gain	I Gain	D Gain	P Gain	I Gain	D Gain
Aileron from Roll (30 Hz)	12	15	8	1	30	1
Elevator from Pitch (30 Hz)	9	14	11	1	30	1
Roll from Heading (5 Hz)	7	10	10	1	5	1
Pitch from Airspeed (5 Hz)	10	10	8	1	5	1
Throttle from Altitude (30 Hz)	5	5	5	1	5	5

Appendix G: Pre-Flight and Post Flight Checklists

G.1 Pre-Flight Checklist

Airframe: ECO-8

Date										
Engine Power pack is * DISCONNECTED *										
Balls Links secure										
Pushrods secure										
Control Horns secure										
Servos secure										
Swash Plate and control surfaces move w/ freedom										
Frame and main gear structure integrity										
Autopilot Box, brackets, anti-vibe isolation integrity										
Compass, GPS ant, AGL, Modem are secure										
Main & Tail rotor assemblies (incl. gezus bolt, pin, nut)										
Main and Tail rotor blade tightness										
Tail rotor belt and tail boom tightness										
Wiring (are all non-locking connectors secure?)										
Power wiring check										
Training Gear installed and secure										
Control surfaces deflections - directions										

Airframe: Rainbow and M36

Engine Power pack is * DISCONNECTED *										
Pitot tube secured										
Pitot tube plugged into the autopilot										
Camera transmitters secure										

Appendix G: Pre-Flight and Post-Flight Checklists

Camera connectors secure														
Camera lens removed														
Autopilot Box, brackets, anti-vibe isolation integrity														
Compass, GPS ant, AGL, Modem are secure														
Control surfaces deflections - directions														

Autopilot

Main Battery Voltage via voltmeter (autopilot/modem)														
Servo Battery Voltage via voltmeter (engine/servos)														
Fly and VRS (appropriate have been sent), compare														
Autopilot COM cabled connected back to modem														
PC to Base Modem connected, ON, Hor, Link green														
GPS Lock														
Flight Plan looks appropriate (use status monitor too)														
Pitch Response (artificial horizon and swash/servo)														
Roll Response (artificial horizon and swash/servo)														
Yaw Response (artificial horizon and tail servo)														
Compass readings (with pitch and roll)														
Collective Response (ENGINE DISCONNECTED!)														
AGL Transducer & readings														
Wind below < 35kmh gusts														

R/C Transmitter

Frequency Checker gone through 3 iterations - clear														
Vecfro Label appropriate for flight mode														
Model name match														
Transmitter Voltage														
Range Check														
ALL TX Trims at 0														
All switches in proper orientation (back and up)														
Control Deflections in PIC or CIC Attitude Arcade (direction and amplitude)														

Appendix G: Pre-Flight and Post-Flight Checklists

Motor

Motor leads connected (Motor Battery disconnected!)										
Secure to frame										
Vibration Test										
Propeller secure										
Propeller direction correct										

Initialization

Status Ok										
-----------	--	--	--	--	--	--	--	--	--	--

G.2 Post-Flight Checklist

Airframe: ECO-8

	Date																		
Engine Power pack is * DISCONNECTED *																			
Landing base instact																			
Control surfaces deflection - ok																			

Airframe: Rainbow and M36

Engine Power pack is * DISCONNECTED *																			
Pitot tube unplugged																			
Camera lens cover																			
Control surfaces deflection - ok																			

Autopilot

Main Battery Voltage via voltmeter (autopilot/modem)																			
Servo Battery Voltage via voltmeter (engine/servos)																			
Communication PC-Autopilot - OFF																			
Aileron control surface																			
All power supply - OFF																			

R/C Transmitter

Throttle setting at minimum																			
Transmitter Voltage																			

Motor

Propeller broken																			
Motor too hot																			

Reference in Appendix

- [1] T. R. Yechout, S. L. Morris, D. E. Bossert, and W. F. Hallgren (2003), *Introduction to Aircraft Flight Mechanics: Performance, Static Stability, Dynamic Stability, and Classical Feedback Control*, American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia.
 - [2] D. Jung and P. Tsiotras (2007), Modeling and Hardware-in-the-Loop Simulation for a Small Unmanned Aerial Vehicle, *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, Rohnert Park, California, 7-10 May.
 - [3] B. N. Pamadi (1998), *Performance, Stability, Dynamics, and Control of Airplanes*, American Institute of Aeronautics and Astronautics, Inc, Reston, Virginia.
 - [4] MicroPilot (2004), MicroPilot Autopilot Installation and Operation, *Technical Manual*, Manitoba, Canada.
-