A Study of Urban UAV in A Simulation Environment

Aryo Wiman Nur Ibrahim

School of Mechanical and Aerospace Engineering

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

2007

Acknowledgements

I have many people to thank for their assistance. I have confronted many difficulties during my academic career. However, I could not have overcome such difficulties without their support and help. In particular I am greatly indebted to my supervisors, Associate Professor Gerald Seet and Associate Professor Michael Lau for providing me with the means and opportunity to work on this project. I wish to thank my colleagues who have worked with me: Pang Wee Ching, who always gave the best support and guide to *ROBOSIM* simulator; Goh Boon Keat and Suhartono Setiawan, who have shared mini UAV research enthusiasm; Anson Heryanto, who has spent considerable time assisting with thesis editing. I also would like to thank Mrs. Agnes Tan, Mr. Lim Eng Cheng, Mr. You Kim San, and Miss. Toh Yen Mei from Robotic Research Center, Nanyang Technological University, Singapore for providing technical and administrative support. To my family, I would like to say "I love you" rather than "thank you." I am glad to dedicate my thesis to them.

Table of Contents

Abstract

Unmanned Aerial Vehicles (UAVs) have been developed from remotely guided drones into experimental Unmanned Combat Aerial Vehicles (UCAVs) of today for combating terrorism. This technology was pioneered by the military and then expanded into civilian uses, e.g. Film production. The growth was endorsed by the advancement of computing and navigation technology, which leads to an important research issue in the design and development of Autonomous UAVs.

Recently, the demands for small platform of autonomous UAV have been increasing for urban operations, which require a mix of small, distributed air and ground based sensors to detect and track small, distributed targets in the urban clutter. However, current UAVs have numerous potential failure points. Doing an experiment to improve UAV system reliability is sometimes difficult due to many restrictions and limitations that prevent it from being done in the real world.

Simulation is therefore chosen as the approach of this project to study the autonomous behaviors of mini UAVs deployed for urban operation. The study includes: (i) defining the operational requirements of a modeled UAV deployed in urban environment, (ii) testing the conceptual design and operation of modeled UAV in a virtual environment using simulation platform, and (iii) refining the conceptual design and operation based on new defined operational requirements.

The development of the simulation platform is divided into two stages. The first stage is the formulation of a conceptual design for the mini UAV named *RRCUAV*, with an emphasis on the maneuverability and adaptability for urban operation. The second stage is the establishment of a UAV simulator, which is the product between the *RSuav* module that focuses on aerodynamic models and intelligent system, and *ROBOSIM* plug-ins, which specialize in 3D graphical drawing and the creation of virtual environment.

With the UAV simulator created, conceptual operation is iteratively improved and might be followed by re-designing the mini UAV to accommodate the requirements. A set of autonomous behaviors are developed and they include: (i) Automatic Takeoff, (ii) Go-to-waypoint operation followed by Automatic Ground Tracking (iii) Automatic Landing, and (iv) Obstacle Avoidance.

The result of study shows that *RRCUAV* is generally suitable for urban operation because it is able to: (i) perform take-off and landing at reasonable space, accuracy, and time, (ii) perform stable go-to-waypoint operation at limited waypoint separation distance, (iii) provide stable surveillance operation, (iv) provide safety procedure in the event of power shortage and loss of communication, and (v) perform obstacle avoidance over business district with nearly 100% robustness.

Glossary

AA	Adjustable Autonomy
AFRL	Air Force Research Laboratory
ALFUS	Autonomy Level for Unmanned Systems
ASATE	Aircraft Simulation and Testing Environment
AUVSI	Association for Unmanned Vehicle Systems International
BYU	Brigham Young University
CD	Collision Detection
COESA	Committee on Extension to the Standard Atmosphere
CS	Crystal Space
DT	Decision Tree
DTS	Dynamic Trajectory Smoother
FTS	Flight Termination System
GCS	Ground Control System
GPS	Global Positioning System
HRI	Human Robot Interaction
LADAR	Laser Radar
LOD	Level of Detail
OSG	Open Scene Graph
RMUS	Real Time Multi-UAV Simulator
SCM	Simulator Control Manager
SPOI	Sensor Point of Interest
STOMP	Simulation, Tactical Operations and Mission Planning
TT	Trajectory Tracker
UAVs	Unmanned Aerial Vehicles
UCAV	Unmanned Combat Aerial Vehicles
V&V	Validation and Verification
WPP	Waypoint Path Planner

List of Figures

List of Tables

Chapter 1

Introduction

This chapter is presented to provide a brief introduction of Unmanned Aerial Vehicle (UAV) technology by describing the chronology of UAV technology maturity over historical missions, which later strengthen the role of UAVs in replacing manned flight operations.

In the next section, UAV classification is provided to help the reader to be familiar with various kinds of UAVs operated in wide range of applications. Since the focus of this project is on the behavioral study, UAV autonomy classification is also described to establish a reference point for the development autonomous behaviors of modeled UAV used in this project.

At the end of this chapter, the objective and the approach of this project are explained. It is narrowed down to the selection of UAV platform to be modeled and then followed by defining UAV target application. Lastly, the role of the simulator is limited to the generation of time parameterized trajectories based on defined waypoints.

1.1. Unmanned Aerial Vehicle

1.1.1. History

The earliest UAVs were developed after World War I and were used during and after the Second World War as targets for anti-aircraft gunnery training as illustrated in Figure 1.1.

Figure 1.1: USAF *BQM-34A Firebee* Target (Ryan Aeronautical)

Early target drones were not much more sophisticated than radio controlled model airplanes. The only payload they could handle was a towed target sleeve. In time, target drones became more highly developed. They were used as scoring devices, radar enhancement devices, and to carry countermeasures.

The success of drones as targets led to their use for other missions. The well-proven Ryan Firebee was a good platform to evaluate it for the reconnaissance and Intelligence gathering missions. A series of reconnaissance drones derived from the *Firebee*, known generally as *Lightning Bugs*, were used by the US to spy on Vietnam, China, and North Korea in the 1960s and early 1970s.

While drones were evolving from simple targets to long-range reconnaissance platforms, they were also being developed as decoys. Figure 1.2 shows a decoy drone with sharp corners and concave body to enhance radar reflections so it can be easily detected.

Figure 1.2: *Quail (B-52) Decoy drone* (Davidwoolsey)

The idea of designing a UAV that could remain in the air for a long time has been around for decades. Endurance UAVs are useful to perform inspection and data collection, especially over vast area. This idea only became an operational reality in the late 90s when pre-programmed UAVs were in full service. Endurance UAVs, such as *Global Hawk* and *Predator* shown in Figure 1.3 are used for medium and high-altitude operations. To compensate the problem of high altitude operations, e.g. obscurant clouds and low temperature causing machine jam, both UAVs are equipped by anti icing technology and Synthetic Aperture Radar.

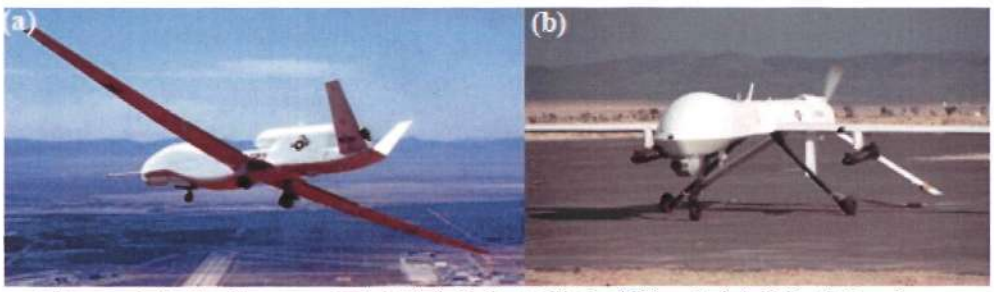

Figure 1.3: Endurance UAV (a) *RQ-4 Global Hawk* (b) *RQ-1 Predator*

If UAVs could be used for reconnaissance, they can be possibly extended to active combat missions. The objective of Unmanned Combat Aerial Vehicles (UCAV) is to destroy enemy Anti-Aircraft Artillery and Surface-to-Air Missile sites. These activities, in military term, are called as Suppression of Enemy Air Defenses. One of the newest UCAV is *X-45A* as shown in Figure 1.4.

Figure 1.4: A UCAV *X-45A* (NASA)

Lately, interest in UAVs has grown into civilian uses. The growth is driven by the advancement of technology, which offered a cheaper, simpler, and more effective way of assembling and operating a UAV. For example, the advancement of computational technology has increased the portion of decision-making process for computers and the revolutionary development of Global Positioning System (GPS) provides outstanding accuracy of navigation data for UAV system. Figure 1.5 shows *Aerosonde*, which has been frequently used for commercial and scientific applications.

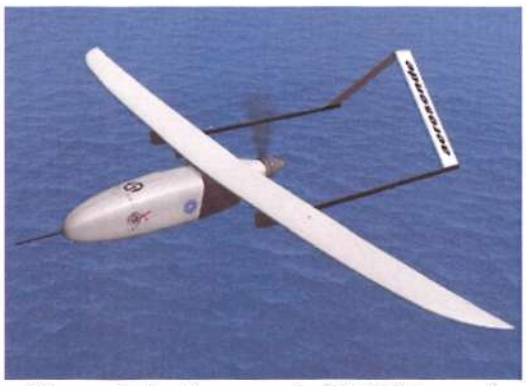

Figure 1.5: *Aerosonde* (UAV forum)

1.1.2. Motivation of UAV Operation in Replacing Manned Flight Operation

In highly dynamic and dangerous environment, a pilot normally encounters psychological intenseness during the mission. Such situations can be found in missions, for instance in urban warfare operations and volcano explorations. However, without a

constant state of environment awareness, the operation may not achieve its objective and may also end with the destruction of the aircraft.

If that happens, the mission will result in the loss of human lives, an invaluable factor and high political costs as in an operation like in Somalia, where the death of 18 U.S. troops in 1993 forced the U.S. government to withdraw its troops from the regions and led to a profound shift in American foreign policy.

In this context, UAVs may play an important role. Figure 1.6 shows *Aerosonde* in The Convection and Moisture Experiment (CAMEX-4) at Mayport Naval Station, Florida gathering data on tropical cyclones. CAMEX-4 was expected to improve hurricane forecasting. It is an example of perilous operations where the use of UAV is decisive and invaluable in many aspects.

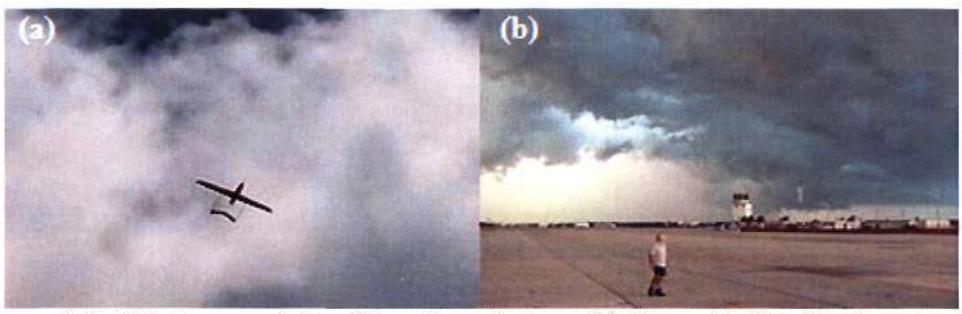

Figure 1.6: (a) *Aerosonde* is off on the mission, (b) Storm in developing stage at Mayport Naval Station, Florida

There are secondary benefits in using computer based control of aircraft. UAVs offer new design freedoms that can be exploited to produce a smaller and simpler aircraft. The life support system required by human pilots is no longer required by UAVs. This benefit can be clearly seen in military aircraft, where the life support system generally requires integrated components of clothing, protective gear, and aircraft equipment. A UCAV can also take advantage of higher maneuverability. Without an onboard pilot, G-force levels can be increased.

Additionally, High Endurance UAVs can be used for lengthy and mundane missions over a vast region. Some typical applications that require extensive surveillance are reconnaissance, port security, forest fire patrol, and disaster respond.

1.1.3. Classification

There are different ways of classifying UAVs. This thesis presents the classification of UAV as follows.

Applications

It is a good start to get familiar with UAV by knowing various applications of UAV. UAV operations are used in these three major areas:

- Military and National Defense – The major pioneering UAV operations are in intelligence gathering, mine detection, and border patrol.
- Civil Defense and Law Enforcement – Recently, the applications of UAVs have been expanded into civilian areas. They are frequently used for Search and Rescue, disaster relief operation, and forest fire inspection.
- Scientific, Commercial, and Civilian – Scientific research and commercial institutions have started using UAVs to support their works, e.g. volcano exploration, film production, and agricultural spraying.

Configurations

This section classifies UAVs based on the aerodynamic that governs the vehicle. It is mainly divided into:

- Aircraft/Fixed Wing – Most of endurance UAVs, e.g. *Global Hawk* and *Predator* are in this category. They have larger wingspan, which allows UAV to minimize drag and thus, fly for a longer operational time.

- Helicopter/Rotary Wing – It can fly in any direction with reference to the heading and to hover as well. *Yamaha Rmax* is an example of a UAV with rotary wings.
- Vertical Take Off and Landing (VTOL) – It refers to a vehicle that takes off and lands vertically and moves faster than rotary wing vehicle. *Eagle Eye* is an example equipped with the technology that allows it to takeoff and land vertically on coastguard ship.

Autonomy Level

The absence of an onboard pilot in the aircraft system defines *unmanned system* that brings about some issues, which people normally ignore. Cooke et al (2006) described the misconception of seeing unmanned system with no involvement of human as well as the negligence of the scores of human who operate, monitor, and coordinate this system.

Huang et al (2003) on Autonomy Level for Unmanned Systems (ALFUS) explained some research institutions that plan to fund development of new autonomous systems currently lack the means of specifying the level of autonomy required and validating that the delivered systems meet those specifications.

It would be, therefore, beneficial to have a set of widely recognized standard definitions for this project, which will propose a conceptual operation for modeled UAV. The conceptual operation will consist of a set of autonomous behaviors and the development of these behaviors will require a reference that assesses the portion of decision making between operator and UAV and UAV level of intelligence.

Bruce (2002) of Air Force Research Laboratory (AFRL) stated the metric must be easily visualized such that upper management could grasp the concepts, broad enough to measure past, present and future system development, and have enough resolution to

easily track impact of technological program. These two institutions proposed the classification of UAV autonomy level, which is later labeled as ALFUS-AFRL Autonomy Classification in this thesis. The most relevant autonomy level to the development of UAV autonomous behaviors in this project lies between level 1 and 4. They are described as follows.

- Level 1. Remotely Guided, which is the lowest autonomy level in which UAVs' perception and decision-making process fully rely on the ground control station. The UAV is manually controlled by an operator and the mission quality totally depends on the remote pilot's skill. This type of UAVs can be found in the earliest type, such as *Lightning Bug* and *Decoy drone*.
- Level 2. Real Time-Health Diagnosis and Level 3. Adapt to Failure & Flight Conditions – *Pioneer* and *Predator* are positioned at these levels, where the ground control station is able to monitor UAVs' current conditions and to pre-program UAV actions, e.g. go-to-waypoints. *Predator* is equipped with multi-path redundant avionics architecture, which is capable of preventing internal management failure. However, they can only adapt in a static environment, which preloaded mission data is dedicated for.
- Level 4. Onboard Route Re-plan – There is on-going research to develop onboard planning, such as obstacle avoidance technology and algorithms. This capability has greater situational awareness than pre-programmed behaviors of level 2 and 3. This is because UAV is equipped with a sensor that enables it to respond and perform decisions based on changing environmental status. Some testbeds has successfully demonstrated such capabilities and there has been a great indication that these technologies will be available in the market in the

near future. The current difficulty lies in the computational limitation that restricts the capability of self-resource management, which constructs situational awareness and provides decision making capabilities for UAV in real time.

1.2. Motivation of Work

1.2.1. Benefits of Higher Level Autonomy

The development of autonomous guided vehicle, including UAV has become one of the major trends in robotic technology in the past decades. The ultimate objective is to eliminate major problems described as follows.

Highly dependent on Human Operator

Current UAVs still rely much on a decision maker at the ground control station due to their limitation to absorb and process information. This phenomenon can be found especially in endurance UAVs for the purpose of maintaining the reliability of operation. *Global Hawk* requires a crew of 28 personnel in the theatre of operation. On the other hand, *Predator* needs approximately 55 members, including communication, pilot, and sensor operators for 24-hour operations.

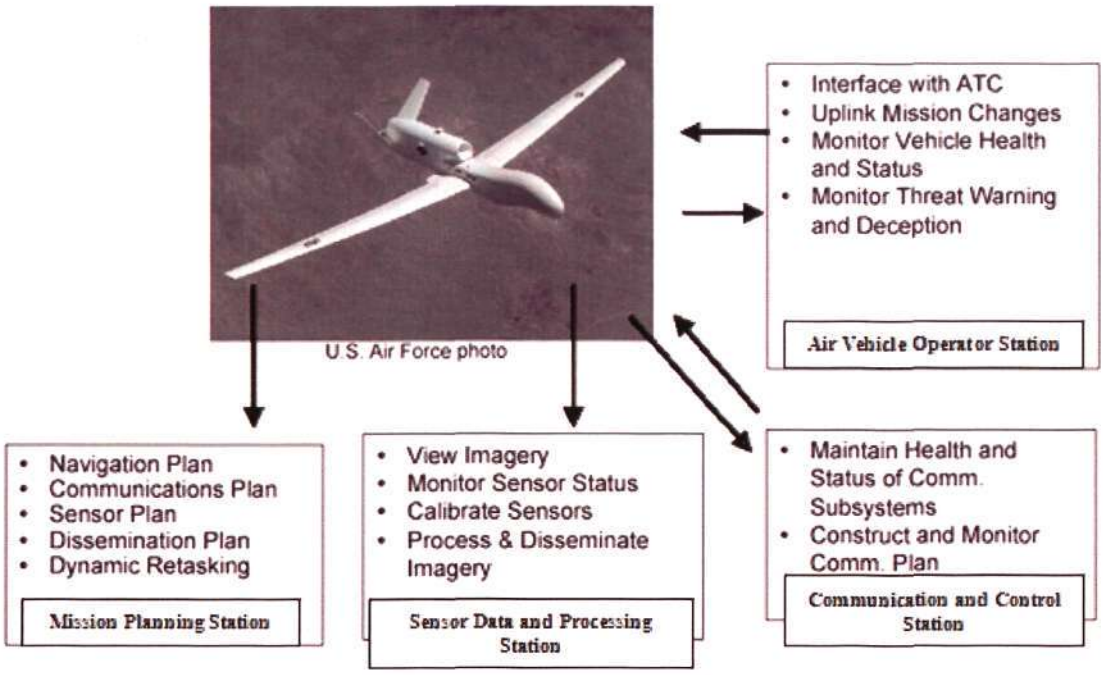

Figure1.7: *Global Hawk* Mission Control Elements (Hansman and Weibel, 2004)

Therefore, operators' training and the operational cost are high with platoon of operators to monitor different aspects of UAV conditions and mission progress. Diagram shown in Figure 1.7 illustrates operators' tasks in s *Global Hawk* operation.

Mishaps Due to Human Error

The need of manpower also highlights other issues, such as crew selection and training. Crew selection and training are needed to ensure operators are qualified and proficient to handle critical issue of tele-flight operations. Failure in this effort has brought several cases of UAV accidents. Some of the factors described by Leduc et al (2005) are: (i) Training failure, when the operator is not trained to a required standard (e.g., insufficient, incorrect or no training on the task); (ii) Standards failure, when standards/procedures are not clear or practical, or do not exist.

Ferguson (1999) studied the causes of UAV mishaps based on various sources. Based on statistics, he found that almost half of UAV mishaps are due to human factors. There are several reasons why operators made mistakes during UAV operations.

Piloting a UAV is different from piloting a manned aircraft since it involves studies on depth perception, which the followings are some of the examples.

- Maintaining Spatial Orientation and knowledge of local to global image – Due to the limitation of Situational Awareness, UAVs do not have the capability of global motion estimation, which means constructing global picture of the environment from individual image frame of camera. Therefore, an operator is constantly required to maintain spatial orientation and knowledge of local image to global situation of environment.
- Target Identification – Current UAV technology cannot support Automatic Target Identification since it requires extensive image data processing. This task is therefore manually performed by operators.
- Health monitoring, in which the operator is required to supervise internal conditions of a UAV, such as fuel management.

Gugert (2004) mentioned operators sometimes reported difficulty in maintaining spatial orientation, such as telling the sensor operator which way to move the camera in order to view a target or finding cardinal direction to identify target locations and determining which location was east of a landmark. Target identification is not an easy task either. This task requires broad knowledge in aerial imagery since target identification must be done in nearly real time, especially for high risk operations.

1.2.2. More Demands on Smaller and Mixed Agents for Urban Operation

Urban operations, especially military operations will become more frequent in the future. By 2025, two-thirds of the earth's population is projected to live in urban areas and 90 percent of the growth will be in the developing world (World Resources, 1998). Some observers believe that, as populations shift from rural to urban areas, the focus of

existing conflicts whether tribal, ethnic, religious, or ideological will shift to urban areas as well.

In a place like Singapore, the presence of urban surveillance technology will give great benefits, such as traffic monitoring and MRT railway inspection. However, Vick et al (2002) stated that urban surveillance and reconnaissance are more demanding than surveillance and reconnaissance of most other types of terrain because of poor Line of Sight, intense clutter and potential for intermingling of civilians, friendly forces, and adversaries in an area of military operation.

Vick et al (2002) added that instead of a single large, long-range sensor platform, urban settings require a mix of small, distributed air and ground based sensors to detect and track small, distributed targets in the urban clutter. For example, *Global Hawk* operating at an altitude of 9 km can detect moving vehicles 185.2 km away on flat terrain but in an area of widely spaced apartment buildings, it is only able to monitor within a radius of 12.6 km only. On the other hand, a combination of small, distributed ground and air based sensors, e.g. mini UAVs is more effective to detect and track small, distributed targets in the urban clutter.

An example of small aerial agent system deployed for urban operation is the traffic management system. Coifman et al (2006) stated that there is an alternative to the current technology of traffic monitoring, which involves an inflexible fixed network of highway cameras, is labor-intensive, and potentially slow in the deployment of personnel during traffic accident. While most traditional traffic monitoring is spatially myopic, only capturing conditions at discrete locations, mini UAVs, for example *Bat-3* have the ability to cover a larger area, focuses resources on the current problems, and give faster assessments and responses to incidents. Faster response can lead to reduced

traveler delay, as well as improved health status of injured victims through faster medical attention.

1.2.3. Predominance of Simulation

Current UAVs have numerous potential failure points. Table 1.2 was taken from Peck (2003), which compared the number of accidents between UAVs and manned aircrafts per 100,000 flight hours. There are sharp differences that sufficiently prove that the reliability of current UAVs is much lower than manned aircrafts.

Table 1.1: UAV Reliability Comparison (Peck, 2003)

Unmanned Aircraft Accidents/100,000 hr	
Global Hawk (4 accidents)	168
RQ- 2A Pioneer	363
RQ-2B Pioneer	139
Predator RQ-1A	43
Predator RQ-1B	31
Manned Military Aircraft	
F-16	3.5
Manned Civil / Commercial Aircraft Accident /100,000 hr (2003)	
General Aviation (Fatal)	1.4
Part 121 Scheduled &Unscheduled	0.313
Part 121 Scheduled &Unscheduled (Fatal)	0.012

Doing an experiment to improve UAV system reliability is sometimes difficult. There are many restrictions and limitations that prevent experiments from being done in the real world. These restrictions include cost constraints, safety considerations, logistical challenges, and a host of variables that make use of advanced technology a necessity.

Simulation might be one good approach to solve this kind of situation. It has been part of major process and become the prerequisites to physical implementation of many

engineering product. McGarity (2005) stated that simulation mitigates the risk associated with UAV selection acquisition and operation.

One of the most important issues in UAV simulation is the understanding of its environment. McManus et al (2003) stated that testing aircraft systems in a realistic synthetic environment is an important process to create situational awareness for UAV. Their *Aircraft Simulation And Testing Environment (ASATE)* simulator provides a suitable simulation environment as a means for designers to construct mission and control algorithms for avionics hardware and real time software systems.

Another version of simulator with synthetic environment is CAE (2005) with integrated *live* (real people and real equipment), *virtual* (real people operating simulated systems) and *constructive* (simulated people operating simulated system with real people making inputs) environments.

Lastly, Johnson and Fontaine (2001) focused more on the simulation of low level UAV system. They performed tests in range of UAV systems, such as guidance, navigation, and control by taking account on model error and environmental disturbances.

1.3. Objective

The aim of this project is to create a simulation platform to study the autonomous behaviors of mini UAV deployed for urban operation. The study approach, which is highly inspired from Weibel and Hansman (2005), is an iterative process that follows a prescribed set of activities, which can be illustrated by a closed loop control system as shown in Figure 1.8.

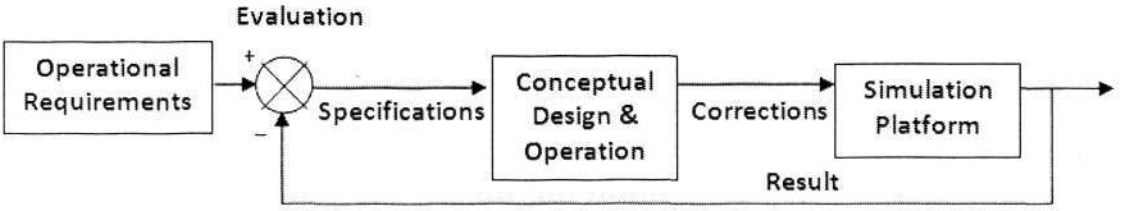

Figure 1.8: UAV Autonomous Behavior Study

These activities include:

i) Defining the Operational Requirements of modeled UAV deployed in urban environment

ii) Testing the conceptual design and operation of modeled UAV in a virtual environment using simulation platform.

iii) Refining the conceptual design and operation arising from new operational needs, such as payload requirements.

During a simulation test on the conceptual operation, modeled UAV will be equipped with a set of autonomous behaviors, e.g. pre-programmed go-to-waypoints and onboard planner on obstacle avoidance. Conceptual operation can be refined through an iterative process by for example, finding the most appropriate capability versus affordability tradeoff.

1.4. Scope

1.4.1. Target Application

The modeled UAV in this project is selected for the Singapore environment, which is completely an urban area. UAV technology gives promising advantage in military applications, such as Urban Warfare Operation and Mine Detection and in numbers of civilian uses, for example Coastal Patrol, Search and Rescue, Fire and Chemical Hazard

Monitoring in Industrial Areas, Crime Scene Mapping, Perpetrator Tracking, City Mapping, Traffic Monitoring, MRT Railway Inspection, and many more. These become the reasons why urban operation is chosen as the target application of UAV in this project.

1.4.2. UAV

To meet the objective, a UAV that is maneuverable enough to move among buildings and to takeoff and land in urban area is required. It is expected to be maneuverable as well as stable in proceeding surveillance operation. There are only two kinds of UAV with reasonable size and turning radius. They are mini and micro UAV. Mini UAV generally has wingspan of less than 2 m, weight of less than 10 kg, and turning radius of less than 100 m. On the other hand, micro UAV generally has wingspan of less than 50 cm, weight of less than 500 gram and turning radius of less than 10 m.

However, micro UAV has too many restrictions, especially the size and payload constraints. Unlike the mini UAV that can fly from one to four hours, micro UAV can last up to 30 minutes only. These problems limit the use of micro UAV in higher level of autonomous behaviors and in the possibility of accomplishing urban operation successfully. Although mini UAV has larger turning radius, flying in commercial area with widely spaced building seems feasible. Therefore, based on these reasons, a mini UAV named the *RRCUAV* is created for this project.

1.4.3. Simulator

The simulator will be equipped with the model of current technologically developed mini UAV systems and environments to facilitate the study of autonomous UAV

operations. The development of the autonomous behaviors will be based on ALFUS-AFRL autonomy classification. It will start from the development of pre-programmed behaviors that lies between level 2 and 3. It will then be developed to level 4, which is based on onboard planner. Lastly, the accomplishment of urban operation in three modes: takeoff, cruise, and landing will be demonstrated as well.

Beard and McLain (2003) and Kingston (2004) illustrated a comprehensive architecture of UAV system as shown in Figure 1.9.

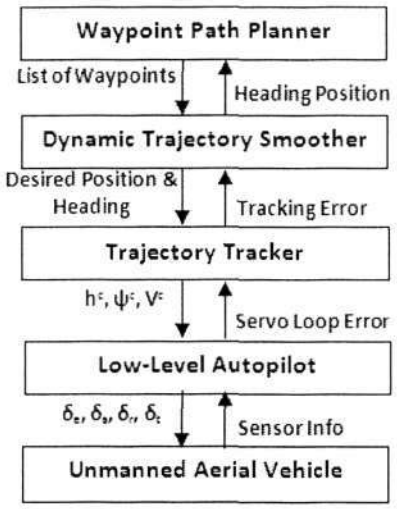

Figure 1.9: UAV Architecture

Waypoint Path Planner (WPP) is the highest planner that produces waypoint paths for the UAV, which satisfy operational constraints, e.g. fuel limit and/or the cooperation constraints if it is formed as group of UAVs. Based on given waypoints, Dynamic Trajectory Smoother (DTS) is the next level that smoothes out the waypoint paths producing time parameterized trajectories, which maintain the operational constraints and also satisfy the kinematics constraints of the UAV. Trajectory Tracker (TT) is then used to track the desired heading and position produced by the DTS and outputs altitude h^c, velocity V^c, and heading ψ^c commands. Lastly, the physical UAV is controlled by a low-level autopilot commands for: elevator δ_e, aileron δ_a, rudder δ_r deflection angle, and

throttle δ_t. The commands of each level will be refined based on the information given by lower level. For example, the DTS will correct the heading and position values if there is any tracking error caused by wind disturbance.

The *ROBOSIM* simulator used in this project acts as a DTS that can import WPP and generate desired heading based on the generated waypoints from WPP. However, evaluating the mission performance of mini UAV, which is operated by particular WPP, is not an interest of this project. This is simply because WPPs were created by other researchers, who had performed the evaluation study in their project.

1.4.4. Human Robot Interaction

Human Robot Interaction (HRI) is a branch of study on the development of autonomous behaviors. It is about optimizing the performance of mission by managing the portion of human and robot involvement in the process of decision making.

One of fields focusing on this approach is Adjustable Autonomy (AA), which refers to the agent *dynamically* varying its share of the decision to control vis-à-vis the human operation (Scerri et al, 2002). Another branch study is the Mixed Initiative Planning (MIP) systems, where human and machines collaborate in the development and management of plans (Burstein and McDermott, 1994). AA concentrates more on the autonomy level, while MIP focuses more on intelligence.

Since this project only focuses on the study of deployment of autonomous UAV for urban operation, HRI modeling is out of the project scope. It is also important to note that increasing autonomy in the context of this project is to minimize human's supervision in terms of number of operators and supervising time. Human involvement characteristically makes a robot system more reliable. Even at the highest level of UAV autonomy, human cannot be totally out of the loop since human have the abilities that

are difficult to duplicate synthetically, for example interpreting rare or unforeseen events and presenting novel solutions on creativity.

1.5. Structures of Report

The structure of report is organized as follows.

i) Chapter 1. Introduction – The overview of UAV technology is described in this chapter and it is followed by the motivation of work and the objective of project. At the end of this chapter, the scopes of project are defined.

ii) Chapter 2. Conceptual Design of *RRCUAV* – It describes the conceptual design of *RRCUAV*. This includes the engineering specifications of *RRCUAV* and its characteristics, such as capabilities, e.g. real time surveillance and operational constraints, e.g. minimum turning radius.

iii) Chapter 3. Simulator – In this chapter, the architecture of *ROBOSIM* are described. Then, the model of *RRCUAV* systems and environments are created on this platform and followed by the discussion on the assumptions, weaknesses and strengths of each model.

iv) Chapter 4. Model Validation – In the early part of this chapter, the fundamental concept of modeling are described. It is then followed by the model validations of UAV simulator and it is divided into: Conceptual Validation, Verification, and Credibility.

v) Chapter 5. Study on Simulated Behaviors – At the beginning of this chapter, the formulation of conceptual operation comprising a set of autonomous behaviors is presented. The behaviors will be tested and may generate new requirements, which are later used to refine the conceptual design and operation of *RRCUAV*.

vi) Chapter 6. Conclusion and Future Works – At the end of this thesis, the conclusion of works is drawn and the potential future works of the project are listed.

1.6. Chapter Summary

A brief history of UAV technology is important to create the awareness on how and why the current UAVs are operated to replace manned flight missions, especially for high risk and lengthy operations.

The development of *RRCUAV* autonomous behaviors motivates the conceptualization of UAV autonomy classification to assess not only the degree of UAV independence to human intervention but also UAV intelligence level. The developed autonomous behaviors will be useful to reduce high dependency on human operators and UAV mishaps due to human errors.

Simulation is chosen as the approach for the behavioral study and the development of *RRCUAV* autonomous behaviors because it is simpler, less risky, and cheaper than real experiment. The study approach is formulated like a closed loop control system, which conceptual operation and conceptual design of *RRCUAV* are improved based on the requirements generated from simulation tests.

Lastly, the scope of project is defined into three major areas: (i) Class of UAV, which mini UAV is selected, (ii) Target application, which urban operation is chosen based on the nature of Singapore environment and the current trends (iii) Level of Simulator, which acts as a DTS that generates flight trajectory based on given waypoints.

Chapter 2

Conceptual Design of *RRCUAV*

Developed as an engineering product, *RRCUAV* follows a standardized design process and this chapter is provided to describe the conceptual state of its product design. This might include the motivation of adopting the technology and the basis of determining appropriate parameterized value as tradeoff is usually one of the major challenges of engineering design.

In view of this, both engineering specification and operational characteristics of *RRCUAV* needs to be provided. This is to know the technology underlying the product and its operational capabilities as well as the constraints. Categorized as a mini UAV, the *RRCUAV* is then conceptually designed based on the general specifications and characteristics of mini UAVs that are commercially available or used in different applications and competitions in the world.

2.1. General Specifications and Characteristics

Based on the classification given by Blyenburgh (2006) of Unmanned Vehicle Systems (UVS) International and Weibel and Hansman (2005) of MIT International Center for Air Transportation (ICAT), the characteristic of a mini UAV can be summarized as shown in Table 2.1.

Table 2.1: General Characteristics of Mini UAV

Category	Value
Altitude	Less than 5 km
Endurance	Less than 5 hours
Coverage	Less than 200 km
Wingspan	Less than 2 m
Weight	Less than 10 kg

Figure 2.1 shows some commercial mini UAVs in the world. The general specifications and performances of these UAVs are presented in Table 2.1.

Figure 2.1: Mini UAVs: (a) Aladin, (b) Bayraktar, (c) Bat-3, (d) Dragon Eye, (e) Carolo T140, and (f) Orbiter

Table 2.2: The General Specifications of Some Commercial Mini UAVs in the World and *RRCUAV*

Mini UAV	Aladin	Baryaktar	Bat 3	Carolo T140	Dragon Eye	Orbiter	RRCUAV
Manufacturer	EMT, Germany	Baykarmakina, Turkey	MLB Company, USA	Mavionics GmbH, Germany	AV Aerovironment, USA	Aeronautics Defense System Ltd, Israel	Conceptual Design, yet to be manufactured
Wingspan (m)	1.5	1.6	1.8	1.4	1.1	2.2	1.8
Axis (m)	1.6	1.2	1.4	1	0.9	1	1.3
Gross Weight (kg)	4	5	11	3	2.7	6.5	9.0
Payload Weight (kg)	N/A	1.5	1.8	0.3	0.5	1.2	2.4
Operational Altitude (m)	30-3000	600	< 3050	N/A	30-150	150-3000	<3000
Endurance (hr)	>0.5	1	4	0.3	1	1.5	1.3
Comm. Range (km)	10	20	32	6	5	15	32
Max. Speed (m/s)	25	25.8	33.3	18	9.7	38.6	39.6
Cruise Speed (m/s)	12.5	15.3	15.6	13	N/A	12.8	16.1
Engine	N/A	Brushless Electric Motors (Double)	26cc, 1xZenoah G-26, 2 internal combustion	Brushless Electric Motors (Double)	Single Battery	Brushless Electric Engine	Brushless Electric Motors (Single): Xtra 25-14
Fuel Type	N/A	Li-Pol Battery	Gasoline+Oil (40:1)	N/A	N/A	N/A	Li-Pol Battery

As mentioned earlier in section 1.4.2. UAV, a mini UAV named *RRCUAV* is created for this project. To show that *RRCUAV* is designed following the general specification and characteristics of Mini UAVs, the specification of *RRCUAV* is put together with other mini UAVs in Table 2.2.

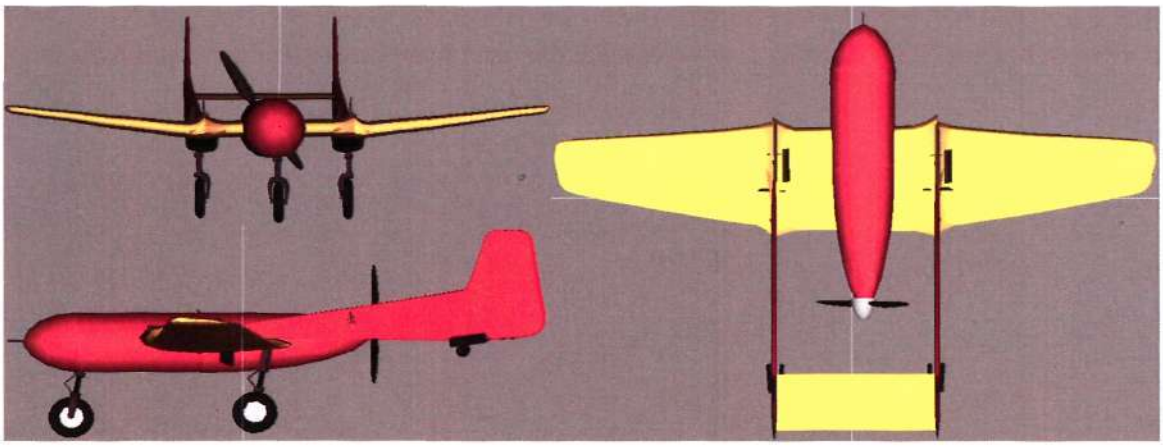

Figure 2.2: *RRCUAV* in three different views

Most of the current mini UAVs are made from composite materials, such as honeycomb, reinforced carbon, foam, and Kevlar, which provides sufficient strength when they execute maneuver up to a wing loading of 3 G. The weight of composite materials is much lighter than metal and thus, the airframe of *RRCUAV* would be only 30% of the total weight.

Based on the comparison with other mini UAVs shown in Table 2.2, *RRCUAV* is relatively heavier and this is because of the allocation of payload with about 2.5 kg for obstacle avoidance sensor, gimbaled camera, and other potential sensors, e.g. infrared, and Synthetic Aperture Radar.

In terms of propulsion location, *RRCUAV* is categorized as a pusher as shown in Figure 2.2 since the engine is at the rear part of aircraft. This is because obstacle avoidance sensor needs to be located at the front part of fuselage. The advantage of pusher propeller is that it can reduce the fuselage skin friction drag but on the other

hand, it causes a reduction in propeller efficiency. This configuration also requires a larger tail and makes climbing at take-off more difficult.

The propulsion system, including electrical motor, propeller, and battery takes about 30 % of the total weight of aircraft. This 3 kg of propulsion system gives *RRCUAV* up to 1.3 hours of operation and to travel for almost 75 km. It can be achieved only if *RRCUAV* flies near to stall speed, e.g. 16.1 m/s to optimize the power consumption.

2.2. System Specifications and Characteristics

This section will describe the specification of part or technology adapted for *RRCUAV* systems. Catalog study is therefore carried out to explain why corresponding part or technology is suitable for *RRCUAV* that is deployed for urban operation.

2.2.1. Propulsion

Most mini UAVs use either internal combustion engine or electrical motor. Mini UAVs using electrical propulsion, e.g. *Bayraktar*, *Dragon Eye*, and *Carolo T140* last for about an hour with maximum traveling distance of 50 km and an operational ceiling of 1 km. On the other hand, the Internal Combustion Engine based mini UAV, e.g. *Bat-3* can be flown for almost 4 hours with maximum traveling distance of 300 km and ceiling of 3 km.

Internal combustion engine is more durable and creates higher speed but it is bulkier and heavier than electrical propulsion. Electrical propulsion is chosen based on the relevance of urban operation that requires lighter plane in order to perform higher maneuverability. The proposed specifications for the electrical propulsion system are given by Table 2.3.

Table 2.3: Specification of Electrical Propulsion System

Category	Value
Engine type	Outrunner Brushless Motor, *Xtra 25-14* (Plettenberg)
Energy source	Battery 8000 mAh, 18.5V
Engine I/P & O/P power	1451.9 Watt & 1219.6 Watt
Engine dimension	57 mm of diameter & 59.5 mm of length
Engine weight	450 gram
Propeller type	3 blade-pusher, 4.6 x 2.5 dm
Battery	LiPo 8000mAh, 18.5 V (Advance Energy Tech)

Plettenberg developed a series of large outrunner motor, called *Xtra* for sport scale and large aerobatics planes, e.g. the 8-kg *Telemaster*. The internal efficiency of *Xtra 25-14* to be used in the *RRCUAV* is 84 % and the losses include heat loss, friction, and etc. The propeller efficiency generally lies between 50% and 70% (Kotwani et al, 2004).

The combination of *Xtra 25-14* outrunner (Plettenberg Elektromotoren, 2005) shown in Figure 2.3 and Thunder power Batteries (Advanced Energy Tech, 2006) gives enormous thrust to minimize takeoff distance and to minimize stall from happening. Low takeoff distance is important to provide reasonable space for the *RRCUAV* to take off in urban area. On the other hand, stalling is an event that must be avoided during climbing to provide safety and stability for *RRCUAV* operation. The consumption of three Thunder power LiPo 8000mAh-18.5 V batteries maintains the flight for 1.3 hours with operational speed of 16.1 m/s and at 100 m altitude.

Figure 2.3: Xtra 25-14 Outrunner (Plettenberg)

2.2.2. Aerodynamic

The aerodynamic characteristics shown in Table 2.4 are proposed for the *RRCUAV*.

Table 2.4: Aerodynamic Characteristics

Category	Value
Wing	
Aspect Ratio *AR*	7.2
Parasite Drag Coefficient *Cdo*	0.04
Induced Drag Coefficient *k*	0.07
Lift Coefficient *Clmax* & *Clmin*	1.4 & -0.5
Takeoff/Landing	
Takeoff & Landing Distance	37.6 m & 52.0 m
Max. Climbing rate	5.0 m/s
Landing Speed	11.2 m/s
Cruise	
Stall Speed	15.1 m/s
Maximum Speed	39.6 m/s
Min. turning radius	25.2 m

Wing

The *RRCUAV* is to be modeled after major mini UAVs, which have been used in different applications and competitions in the world. In particulars, the wing properties of *RRCUAV* are based on *Athena* (Yeung, 2006), which was used for the Association for Unmanned Vehicle Systems International (AUVSI) competition by New York Polytechnic University. These are shown in Table 2.5.

Table 2.5: Wing Properties of Mini UAVs

Small UAV	Wing	C_{do}	Cl_{max}	k	C_d^e	AR
RRCUAV	N/A	0.040	1.40	0.070	0.177	7.20
Athena (Yeung, 2006)	N/A	0.037	1.38	0.073	0.176	~7.75[d]
WSUAV (Kyuho Lee, 2004)	LKH2411	~0.005[a]	~1.50[a]	~0.047[b]	~0.111	8.55
Tornado	HQ3.0/14	0.017	1.20	~0.020[b]	~0.046	21.00

(Puscov, 2002)						
CUAV (Buretta et al, 2003)	Midnight	~0.040[a]	1.60	~0.076[c]	~0.235	8.00
Slug (Eyster, 2006)	SHaWn001 XLSE	0.030	1.31	~0.059[c]	~0.131	8.00

[a] Estimated from the given graph
[b] Estimated based on Lift-to-Drag *L/D* equation
[c] Estimated based on Oswald span efficiency method
[d] Estimated based on given reference area with assumed wingspan of 2.5 m
[e] Estimated from total drag coefficient *Cd* equation

Aspect ratio *AR* is an airplane wing span divided by its standard mean chord. It can be calculated more easily as span squared divided by wing area. A high aspect ratio indicates long and narrow wings, whereas a low one indicates short and wide wings. The aspect ratio of general aviation aircraft lies between 7 and 9, probably averaging around 7.5. With 1.8 x 0.25 m^2 wing area, the aspect ratio of *RRCUAV* is 7.2. The aspect ratio of some mini UAVs can be very high, e.g. *Tornado* reaches up to 21. Higher aspect ratio reduces the induced drag of the wing, gives better lift-to-drag characteristics but poorer stall and spins performance and more likely suffers wingtip damage on landing.

In order to raise the maximum lift coefficient Cl_{max} and to decrease the forward speed during takeoff and landing, aerodynamic devices are affixed to the wing. The most usual devices are trailing-edge flaps with the increase of Cl_{max} obtained by a change in airfoil shape and/or by increased camber. Cl_{max} of wing with flaps down generally reaches 2.2 in sub-sonic airplane. With $Cl_{max}=2.2$, the total drag coefficient C_d of *RRCUAV* becomes 0.38.

As the *RRCUAV* touches the ground, to slow down the speed, it is more effective to have braking mechanism than other recovery methods, such as nets or parachute. One notable braking technologies is the disc brake system (see Figure 2.4) manufactured by BAI Aerosystems and has been applied in *Tern*.

Figure 2.4: Main Gear Disc Brakes on *Tern* Front Wheel (BAI Aerosystems)

Performance

Figure 2.5 gives an indication of the performance of the proposed *RRCUAV* accelerating to maximum speed at 30 m altitude with initial speed V_{init} of 16.6 m/s. This graph is plotted by a simple simulation, which runs based on the specifications given in section 2.2.1. Propulsion and section 2.2.2. Aerodynamic. Required power P_{req} is the power to compensate air drag D, while available power P_{avb} is that generated by engine, which creates thrust T. The graph shows that both available P_{avb} and required power P_{req} meet at maximum output power P_{max} of 731.8 Watt. At this point, UAV reaches maximum speed V_{max} of 39.6 m/s.

Figure 2.5 also indicates that the minimum required power occurs at stall speed. To minimize power consumption, it is better to have an operational speed nearer to stall speed. However, it is important to note that at least 10% of margin of stall speed is required as a safety factor.

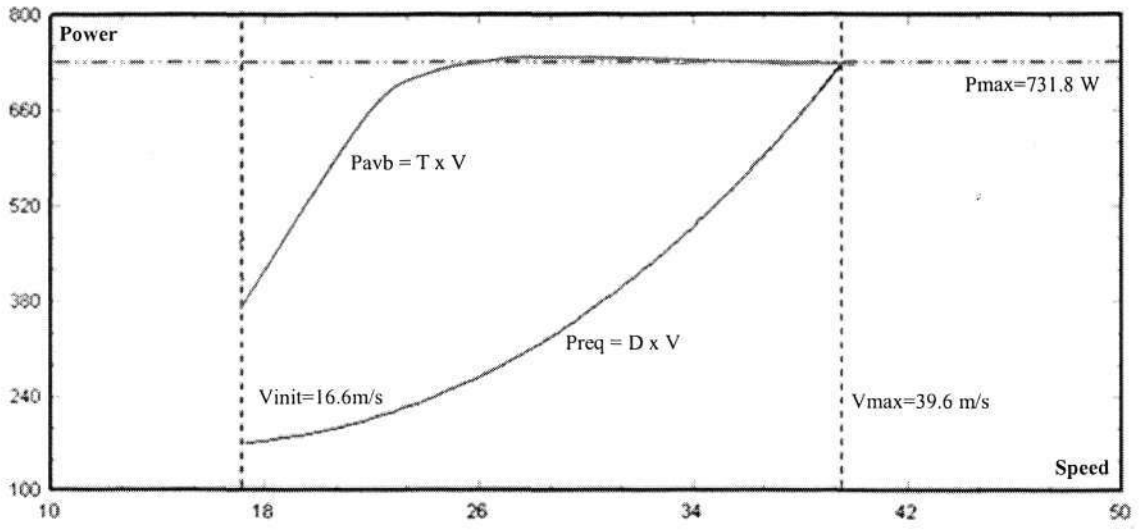

Figure 2.5: Acceleration Performance of *RRCUAV*

Obester (2005) illustrates a comprehensive classification of fixed wing aerial vehicles in Figure 2.6. The classification is made based on three parameters. The first parameter is power-to-weight ratio, which is calculated as the ratio between maximum output power and total mass of the aircraft. Power-to-weight ratio of *RRCUAV* is about 0.08 kW/kg. The second one is wing loading, which is determined as the weight of the aircraft divided by wing surface. *RRCUAV* has relatively higher wing loading (nearly 200 N/m^2), which makes *RRCUAV* located slightly outside the region of "Light Aircraft and UAVs" in Figure 2.6. The last parameter is rate of climb, which indicates the maximum vertical speed of aircraft. Based on Figure 2.6, mini UAVs generally has climbing rate of 5 m/s or less, while, on the other hand, some light passenger aircrafts may have climbing rate up to 12 m/s. The maximum climbing rate of *RRCUAV* is also around 5.0 m/s, which is resulted from the maximum pitching angle γ_{max} of 10^o.

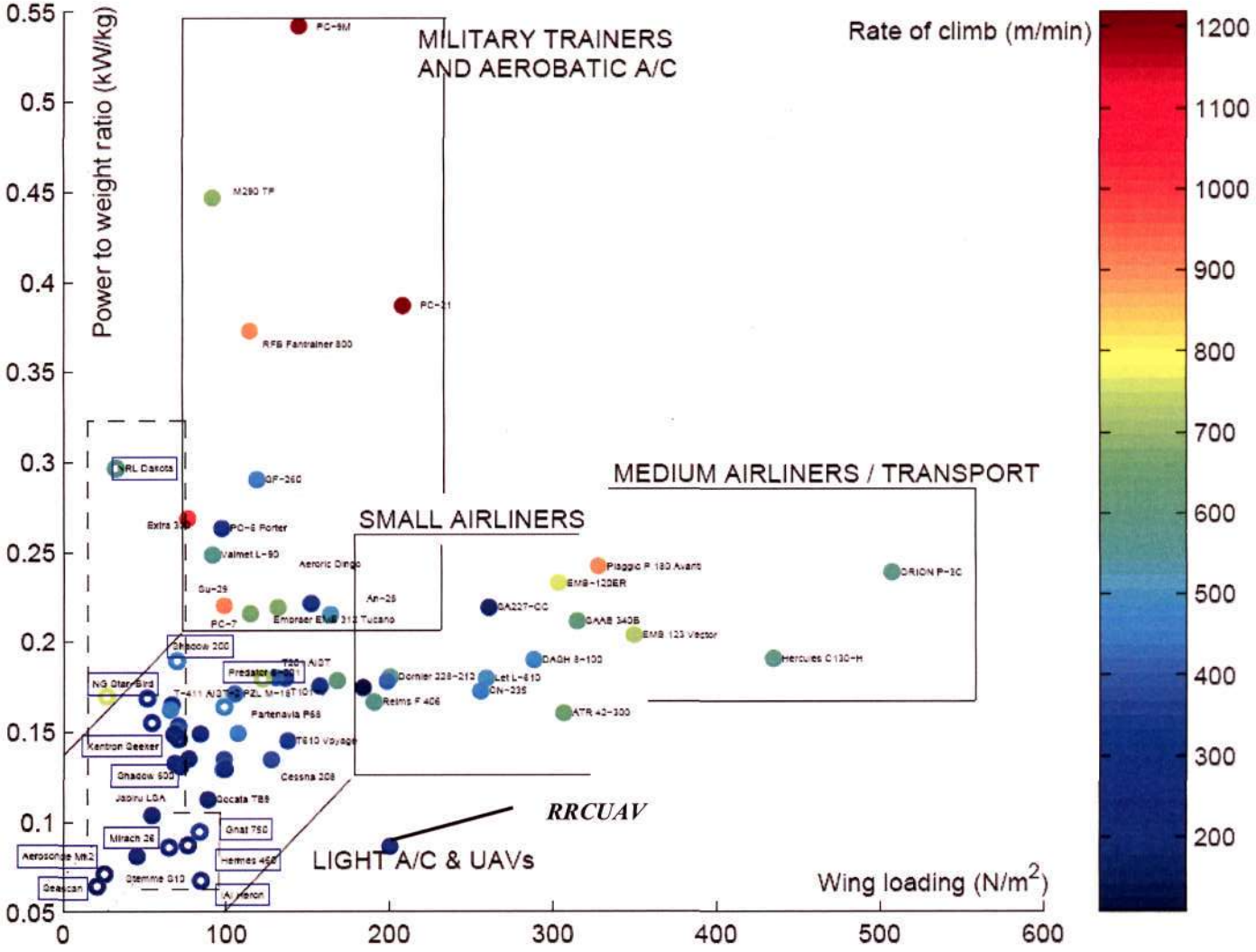

Figure 2.6: Graph of Aircraft Performance (Obester, 2005)

Maneuverability

To be able to fly among buildings, the *RRCUAV* has to be very maneuverable with suitable turning radius. According to the basic principles of aerodynamics, turning is affected by speed and bank angle. It is important to find the right proportion of speed and bank angle to ensure the stability, which is defined as the capability of minimizing the altitude change during turning. Figure 2.7 shows the correlation of speed V and bank angle ϕ to turning radius R in stable condition. This graph is created from a series of data collected from the simulation, which runs based on the specifications given in section 2.2.1. Propulsion and section 2.2.2. Aerodynamic.

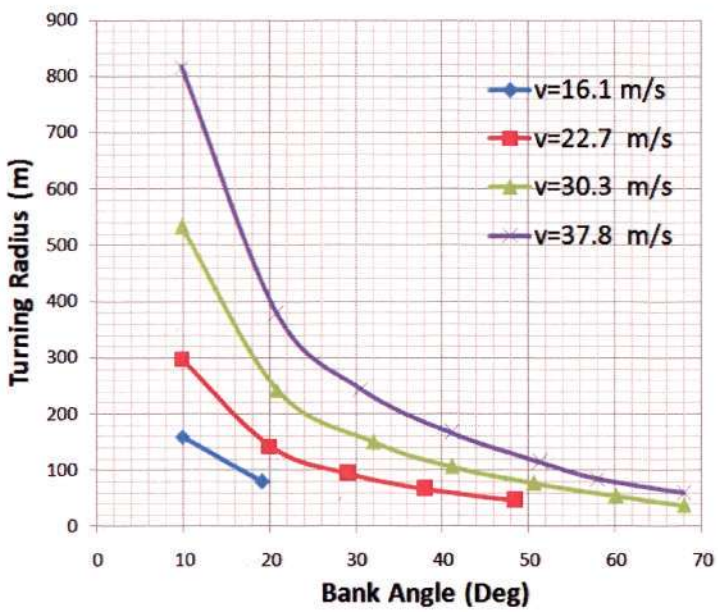

Figure 2.7: Bank Angle and Speed vs. Turning Radius

Figure 2.7 illustrates that lower speed only allows for lower maximum bank angle to ensure stability and vice versa. For example, at speed of 16.1 m/s, *RRCUAV* should be able roll at a maximum angle of 20°. Higher speed, such as 22.7 m/s, on the other hand, will result in a maximum bank angle of 48.3°. If bank angle is larger than the maximum value, the *RRCUAV* will lose its altitude and eventually crash due to insufficient lift.

At turning speed of more than 30.3 m/s, the *RRCUAV* bank angle will hit a limit of 67.9°, and it will slowdown to 24.7 m/s, move in spirals, and will settle at a turning radius of 25.2 m as shown by the flight trajectory plotted in Figure 2.8. Another interesting point is that the minimum radius cannot be achieved by setting the initial speed at 24.7 m/s. In conclusion, the minimum turning radius can be only obtained at speed, which is higher than merely 30 m/s.

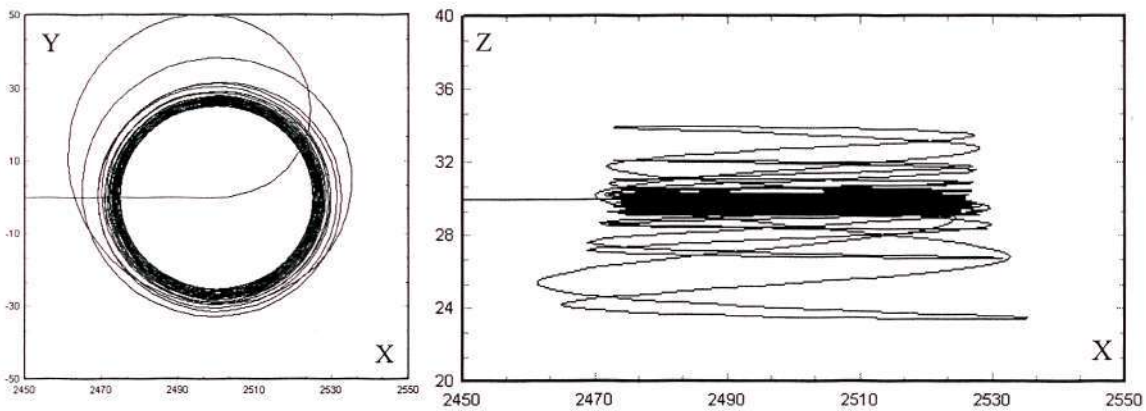

Figure 2.8: *RRCUAV* Flight Trajectory on performing minimum turning radius

Besides stability, wing loading is another operational constraint on turning. As stated earlier, wing loading is determined as the weight of the aircraft divided by wing surface. This definition is true if aircraft flies in a straight path but when it turns, wing loading will be multiplied by a factor called as load factor.

Load factor is calculated as the ratio between lifting force and aircraft weight. As shown in Figure 2.9, load factor is only dependent to bank angle and this is theoretically true. This result therefore eliminates the misconception that claims speed contributes to the wing loading.

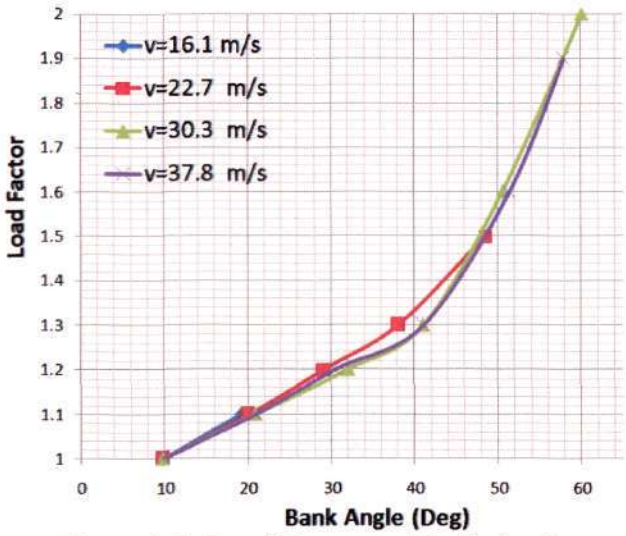

Figure 2.9: Load Factor vs. Bank Angle

During the transition period of turning, the *RRCUAV* generally overshoots about 5° before settling down to its steady state bank angle. For example, at 16.1 m/s, before *RRCUAV* settles at bank angle of 20 °, it overshoots at merely 25°. Figure 2.10 shows the load factor profile at minimum turning radius of 25.2 m. The *RRCUAV* overshoots to 72° with load factor of 4.1 before settling down to bank angle of 67.9°. At load factor of 4.1, wing loading achieves the highest value of 831.28 N/m^2, which is beyond the strength limit of material composite. In *RRCUAV* conceptual design, bank angle is therefore restricted within 70 ° only, which allows maximum load factor of 3.

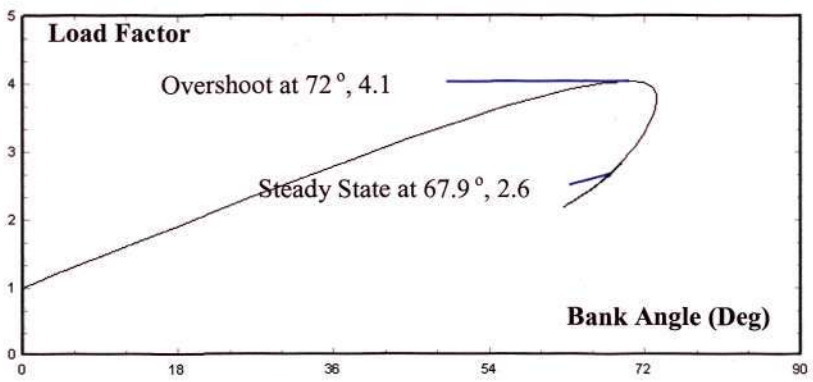

Figure 2.10: Load Factor vs Bank Angle Profiles as *RRCUAV* starts turning to minimum radius

2.2.3. Avionics

Autopilot

The desired autonomous behaviors of the *RRCUAV* will be based on ALFUS-AFRL autonomy classification. To achieve pre-programmed behaviors, an autopilot is required. Boryz and Colgren (2005) discussed three commercially available advanced autopilots that can be used for mini UAVs. They are *Piccolo* manufactured by Cloud Cap Technology, Inc., *MicroPilot* fabricated by MicroPilot, Inc., and *EZI-NAV* by AUAV, Inc.

The *RRCUAV* mimics the general capabilities of these autopilots and the capabilities can be summarized as follows:

- Manual and Half Automatic Control through Joystick
- Autonomous Landing and Take-Off
- Stability Control
- 3D flight path waypoints (up to 1000 waypoints); Interactive waypoint modification and mini UAV redirection
- Programmable error handlers for: loss of GPS signal; loss of RC signal; engine failure, loss of data link; and low battery voltage

The above capabilities are categorized as level 2 and 3 under the ALFUS-AFRL autonomy classification since all of them have a pre-programmed planner.

Many papers describing the testbeds of their UAVs demonstrated some of these capabilities. Quigley et al (2005) and Riseborough (2004) conducted an experiment on auto-landing and take off with data on landing accuracy and time. King (2004), on the other hand, studied the stability control and trajectory optimizer of *Aerosonde* that uses *Piccolo* Autopilot. Lastly, Go-to-Waypoint is the most favored capability in UAV

operations that can be implemented for different applications and also extended higher level behaviors, such as line sweeping demonstrated by Altair Integrated System Flight Demonstration Project (2005).

Some universities and research institutions are now trying to develop future technologies, such as obstacle avoidance system. Saunders et al (2005) tested real time obstacle avoidance over urban area and terrain following using a 1.5 m wingspan UAV. The sensors measure distances and output them to an RS-232 compatible port that can be interfaced to autopilot, such as *Piccolo* or *Kestrel*. Their research has inspired the use of laser range-finder, which will be explained in section 2.3.2. Laser Rangefinder.

Navigation, Aircraft State Sensor, Communication

To facilitate the overall aspects of UAV operation, CloudCap Technology, Inc. has also provided numbers of accessories as follows.

- Model aircraft servo drivers with gears and driven by pulse-width modulated signals with a 4-6 volt power range
- Inertial sensors *Crista IMU* and *Crista EOM*, which provide high resolution angular rates and accelerations
- *Motorola M12* GPS that provides UAV with basic groundspeed and position; and Pitot and Static tube as part of air data system to measure UAV current altitude and speed
- Data link, which is built on the *MHX 910/2400* radio modem from Microhard Systems Inc. The data link has up to 40K baud of throughput and is used for command and control, autopilot telemetry, payload data transfer functions, differential GPS corrections uplink, and pilot-in-the-loop modes

Table 2.6 provides an illustration of the weight and power composition that *RRCUAV* might have in this project. It makes use of most of the accessories of CloudCap Technology mentioned above.

Table 2.6: Illustration of weight and power decomposition of *RRCUAV*

Category	Type	Unit	Unit Weight	Unit Power	Weight (gr)	Power (W)
Aircraft Structure	Ultra-light Kevlar airframe (wing and fuselage)	1	2905	0	2905	0
	Landing Gear & Frame	1	500	0	500	0
Sub total					**3405**	**0**
Propulsion	Brushless Motor: Xtra 25-14 (Plettenberg)	1	450	1451.88	450	1451.88
	4.6x2.5 dm, 3 blade pusher propeller	1	0	0	0	0
	Battery: LiPo 8000mAh, 18.5 V (Advance Energy Tech)	3	790	148	2370	444
Sub total					**2820**	**1451.88**
Avionics	Piccolo Autopilot	1	212	3.6	212	3.6
	Crista IMU	1	37	0.72	37	0.72
	Motorola M12 GPS	1	65	0.23	65	0.23
	MHX 910/2400 Radio Modem	1	75	3	75	3
	Video Transmitter	1	0	1	0	1
	Model Aircraft Servos	5	60.98	1.84	304.9	9.2
Sub total					**693.9**	**17.75**
Payload	Sony FCB Ex-980S	1	230	3.3	230	3.3
	TASE Mechanical Camera Gimbal	1	670	12	670	12
	Opti-Logic RS400 Laser RangeFinder	2	226.8	1.8	453.6	3.6
	Extra Sensors depending on missions, i.e. IR, SAR	1	~250	~5	~250	~5
	Battery: TP 2200-3SX; 11.1V, 2.2Ah (Advance Energy Tech)	2	170	24.42	340	48.84
Sub total					**~1943.6**	**~23.9**
TOTAL					**~8903.5**	**~1493.53**

Table 2.6 shows that the total of maximum power consumption (flying at maximum speed, intensive use of gimbaled camera, and etc.) is about 1.49 kW. It is important to note that with 0.49 kW available from the battery (indicated as green columns) and at the possible maximum power consumption, a battery set can last for 19.8 minutes only.

2.3. Payload

Unlike propulsion, autopilot, navigation system, and data link, payload is anything beyond what is required for its basic operation during flight. It is normally carried by UAV to perform specific mission.

The *RRCUAV* payload will consist of three external devices comprising: (i) surveillance camera, which provides video telemetry data, (ii) laser rangefinder, which provides sensor information of nearby obstacles and other potential payloads considering the mission objective.

2.3.1. Gimbaled Camera TASE

Gimbaled Camera *TASE: CCT part number 900-90012-00* is to be mounted at the lower part of fuselage to allow operator to manually direct the camera view in real time and to execute Sensor Point of Interest (SPOI) mode that enables *RRCUAV* to autonomously track the position of ground objects. This technology works by defining the coordinates of a ground object and geometrically calculates the orientation of the camera angles so that the ground object of interest is always at the center of camera view. SPOI mode is very useful to provide stable surveillance operation regardless of the lateral and longitudinal oscillation and the limitation of turning radius as the aircraft cannot hover and stay exactly in one place in the air.

Table 2.7 shows the general specifications of the Gimbaled Camera *TASE: CCT part number–900-90012-00*, which is shown in Figure 2.11. It uses Sony FCB series that are also present in *Bat-3* and *Coyote* for their surveillance camera.

Table 2.7: General Specifications of Gimbaled Camera *TASE*

Category	Value
Diameter turret	11.2 cm
Overall package size	12.7 x 11.2 x 17.8 cm
Weight	900 g (including Sony daylight camera)
Camera	Sony FCB-EX980S
Zooming Ratio	300 x for 25 x (Optical) and 12 x (Digital)
Field of view	Continuous pan
Tilt	+23° / -203°
Slew rate	200°/sec
Pointing resolution	0.05°
Power Consumption	15.3 W

Figure 2.11: Gimbaled Camera TASE: CCT part number 900-90012-00 (CloudCap Technology)

2.3.2. Laser RangeFinder

Sensor Selection and Specification

In 2003, the first successful UAV testbed (NASA Dryden, 2003) on obstacle avoidance was conducted by NASA. It used a Medium Altitude Endurance UAV named *Proteus* with both autonomous and cooperative obstacle avoidance system. *Proteus* autonomous obstacle sensor is based on the Forward Looking Radar – *Ampitech*

OASYS. Griffiths, S. et al (reviewed) stated that scanning sensors such as Laser Radar (LADAR) and RADAR are typically too large and heavy for mini UAVs. For example, *OASYS*, which has been deployed on the *Proteus*, weighs about 10 kg.

There are three types of sensor that fulfill the design requirement of mini UAVs. They are stereo camera, optical flow sensor, and laser rangefinder. Most mini UAV obstacle avoidance systems use an optical flow sensor that principally estimates the distance of an object based on optical displacement of the image. According to this principle, if the rate of displacement is getting higher then the object is getting closer. Optical flow sensor is a passive sensor and thus, consumes low power (max. 0.165 W) but it is only applicable during day time and requires relatively higher computation for image processing.

On the hand, laser rangefinder providing higher resolution and accuracy can be used any time regardless of weather condition and the time of the day. Excluding the image processing of optical displacements on every pixel, pattern matching, and translational and rotational filtering, laser rangefinder is much lower in computational cost.

Although it is heavier and more power consumed (maximum of 1.8 W), both the weight and the power consumption are still within the tolerance of *RRCUAV* design. Therefore, laser rangefinder is being proposed as part of the external sensor system. The laser rangefinder used for the *RRCUAV* is inspired by testbeds conducted by Brigham Young University (BYU). It is an *Opti-Logic RS400* Laser RangeFinder*: Product Code OL-RF-RS400* as shown in Figure 2.12. This sensor can be interfaced to a RS-232 Serial Computer Port for connecting to autopilots, such as the *Piccolo* and the Kestrel. The General specifications for *Opti-Logic RS400* are described in Table 2.8.

Table 2.8: General Specifications of Opti-Logic RS400

Category	Value
Dimensions	32x78x84 mm
Weight	226.8 g
Range	3.7 to 365.8 m
Data Rate	10Hz for calibrated operation
Laser	Class I (eye-safe), 905nm +/- 10nm
Power	1.8 W
Accuracy	+/-1m on 1x1m^2 diffuse target with 50% reflectivity
Resolution	0.2 m

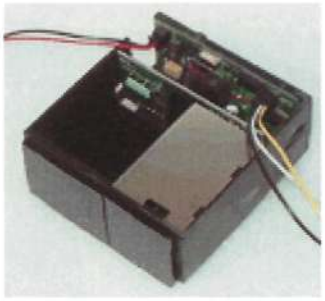

Figure 2.12: *Opti-Logic RS400* Laser Rangefinder: *Product Code OL-RF-RS400* (Opti-Logic Corp.)

Sensor Configuration and Limitation

Laser rangefinder has no scanning capability and thus, the Field of View (FOV) is very limited. A laser rangefinder is not enough to deal with information on building surface in order for the *RRCUAV* to decide whether to turn right or left. This is because the information on building geometry cannot be inferred from a single distance measurement. To compensate for this deficiency, multiple sensors are deployed. The *RRCUAV* uses two laser rangefinders with opening angle θ on the left and right side of fuselage nose.

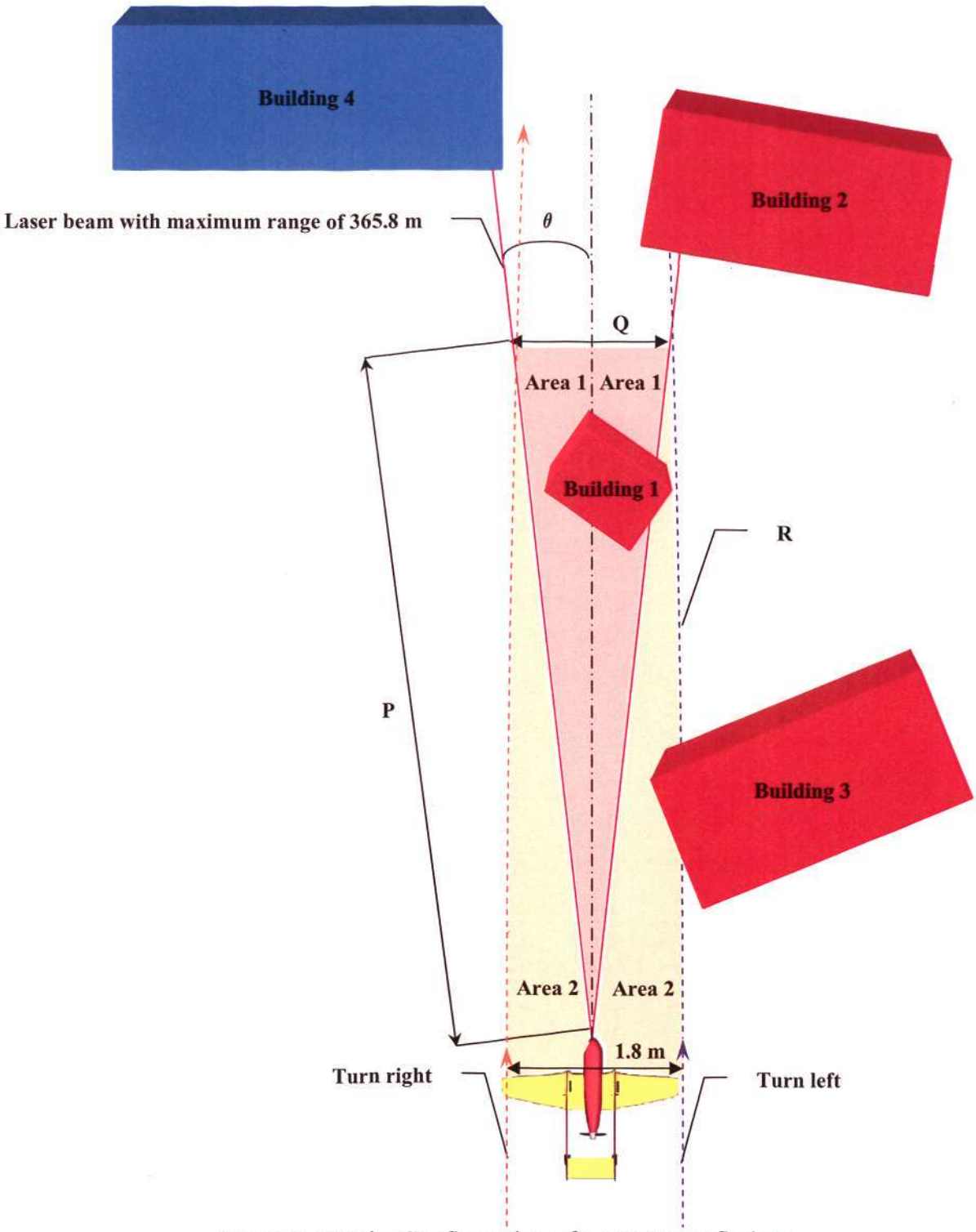

Figure 2.13: The Configuration of Laser Rangefinders

Based on the geometry of turning radius R and laser beams as shown in Figure 2.13, there are three main conditions that may cause the failure of obstacle avoidance. The

first one is when a building is too small (less than defined width Q) to detect and it becomes too late for *RRCUAV* to escape. This condition applies in area 1 as the laser distance is less than defined length P, which is taken from the intersection between laser beam and *RRCUAV* turning radius R.

The mathematical relationships of P, Q, turning radius R, and laser opening angle θ shown in Figure 2.13 are derived as follows.

$$P = \sqrt{R^2 + (R-0.9)^2 + 2R.(R-0.9).\cos\alpha} \quad (2.1)$$

$$Q = 2.P.\sin\theta \quad (2.2)$$

, where $\alpha = 90^\circ - \theta - \sin^{-1}\left(\frac{\cos\theta.(R-0.9)}{R}\right)$ (2.3)

The second condition is when a building is large enough and can be detected earlier but *RRCUAV* is still unable to avoid the obstacle shown as building 2, which intersects one of the *RRCUAV* wingtip trajectories. Chronologically, the *RRCUAV* firstly detects the presence of building 2 and then makes a left turn but due to limited turning radius, it is still possible to crash into the building.

The last condition is when the *RRCUAV* does not detect the presence of any obstacle but then, there is part of object (shown as building 3) entering area 2 that collides with the wing of aircraft. Second condition will be the most likely cause of a crash because the detection area outside area 1 and 2 is much bigger. The first and the last conditions might be the least possible since area 1 and 2 are relatively much smaller.

2.3.3. Potential Payloads

According to Table 2.6, an extra sensor is allocated with maximum weight of 250 g and maximum power of 5 W to enable *RRCUAV* to perform specific mission. Some possible sensors that can be deployed are described as follows.

Infra Red thermal imaging video camera

Infra Red (IR) is the most matured of the UAV sensor technologies. The sensor uses an array of IR detectors scanning across the target area and creates an image from the emitted thermal radiation. Therefore, this sensor is frequently used for forest fire inspection and border patrol that requires night vision capability to monitor for incoming illegal immigrants.

As the sensor is a passive system, it cannot detect targets obscured by foliage, fog, thick haze, dust, and smoke. Water vapor, in particular, can dramatically reduce its useful range due to integrated effects of scattering along sight line.

Miricle 110K as shown in Figure 2.14 manufactured by Thermoteknik Systems Ltd (2006) is one of the leading technology of Infra Red thermal imaging video camera. It is 42 x 40 x 40 mm and has a maximum power consumption of 3.2 W and weighs 86 g. It gives a 384 x 288 pixels of image.

Figure 2.14: Miricle 110KS (Thermoteknik Systems Ltd, 2006)

Together with Electro Optical (EO), IR helps UAV to upgrade its image quality, especially mobile targets at relatively low cost. It is capable of night vision. *EO/IR Payload Series 66 Miniature* fabricated by *BAE Aerosystem* is one example of the EO-IR combined sensors that can be deployed in Mini UAVs, e.g. *Evolution-XTS.* However,

the use of this sensor will be only possible for the *RRCUAV* if TASE gimbaled camera is removed.

Synthetic Aperture Radar

Synthetic Aperture Radar (SAR) is a high resolution ground mapping technique which takes advantage of the forward motion of UAV that is carrying pulsed radar to synthesize the effect of a large antenna aperture. SAR has been used in many military and civilian applications. It can be used to determine the regions that contain land mines. It is useful for missions, such as pipeline inspection performed to determine source of the leaks. SAR is also powerful tool in geology and geography research.

Unlike the IR, the SAR is largely unaffected by weather or time of the day. This sensor is also generally characterized by the accuracy of angular location being much poorer than its range location.

There have been various efforts to miniaturize SAR so it can be installed into Mini UAV. Zaugg et al (2006) developed a *microSAR* for a Mini UAV with 1.8 m wingspan. Their design requirement is to build an approximately cigar box size SAR designed for low-altitude (300m) operation. imSAR (2006) on the other hand, has successfully provided a commercially available SAR – *NanoSAR* shown in Figure 2.15, weighs only 1 kg and has dimension of 20.3x12.7x12.7 cm^3. However, the deployment of *NanoSAR* consuming maximum power of 25 W requires the exclusion of TASE gimbaled camera and Opti-Logic RS400 Laser Rangefinder for *RRCUAV*. This is because of limited power available in *RRCUAV*.

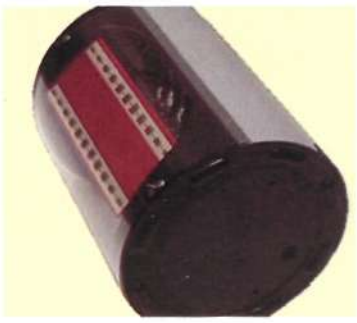

Figure 2.15: Insitu A-20 SAR Module (imSAR, 2006)

2.3. Chapter Summary

The engineering specifications and characteristics of commercial mini UAVs in the world have been used as the guideline of designing the *RRCUAV* as a model to be used in the project. Most of the technologies adopted for the proposed *RRCUAV* are inspired by CloudCap products that range from Autopilot to Surveillance Camera.

The main aspects of the *RRCUAV* conceptual design are the maneuverability and adaptability for urban operation. Turning radius therefore becomes the major concern as the *RRCUAV* is expected to fly among tall buildings. In addition, the deployment of flaps and the selection of appropriate engine result in better takeoff and landing performance in urban area, which are relatively limited in space.

The major challenges lie with the design tradeoff as the operational constraints on weight, size, and available power are tight. To reach an ALFUS-AFRL autonomous level for onboard route re-plan, the *RRCUAV* has to gain 9 kg and, this is relatively heavier than other mini UAVs powered by electrical propulsion.

Having the conceptual design of *RRCUAV* completed, the most valid components of design are the ones equipped with the most mature technology, which has been proven in many applications. Those components are present as the support systems of pre-programmed behaviors, such as autopilot executing go-to-waypoint operation or

gimbaled camera *TASE* that supports SPOI mode. In contrast, the support system for online behaviors, e.g. Obstacle Avoidance is still immature as it is yet to be commercially available and currently regarded as on-going research.

Chapter 3

Simulator

As mentioned earlier in Chapter 1, simulation was chosen as the means of studying the autonomous behaviors of the proposed *RRCUAV*. Hence, *ROBOSIM* simulator is provided in this project and acts as a real time simulator that maintains the continuity of simulation, state of entities (e.g. ground and aerial vehicles) and the generation of graphical display of entities and virtual environments. At the beginning of this chapter, most of the discussions will take place around the functions of *ROBOSIM* as a simulation platform. It is then followed by the description on *ROBOSIM* structure and how UAV module is interfaced into *ROBOSIM*.

The next section will describe the architecture of UAV module and the inter-correlation among UAV systems in this module. Each model implemented in this module is explained by providing the information on assumptions, theoretical foundations used, and its strengths and weaknesses. Lastly, it is followed by the discussion on the decision flow of autonomous behaviors implemented in intelligent system sub module.

3.1. Structure of *ROBOSIM*

ROBOSIM is an existing simulator that has been used for the development of autonomous behaviors of ground vehicles, e.g. *ATRV* and *Pioneer* (Wee Ching, 2006). By implementing the UAV module and relevant environmental models in *ROBOSIM*, the proposed *RRCUAV* design and autonomous behaviors can be tested and consecutively refined in this platform.

ROBOSIM is generally characterized as follows.

- Programming Language : *C++, C*
- Operating System : *Linux (Redhat Fedora Core 3 or above)*
- Environment : *X windows*

3.1.1. General Architecture

The system architecture of *ROBOSIM* is illustrated in Figure 3.1.

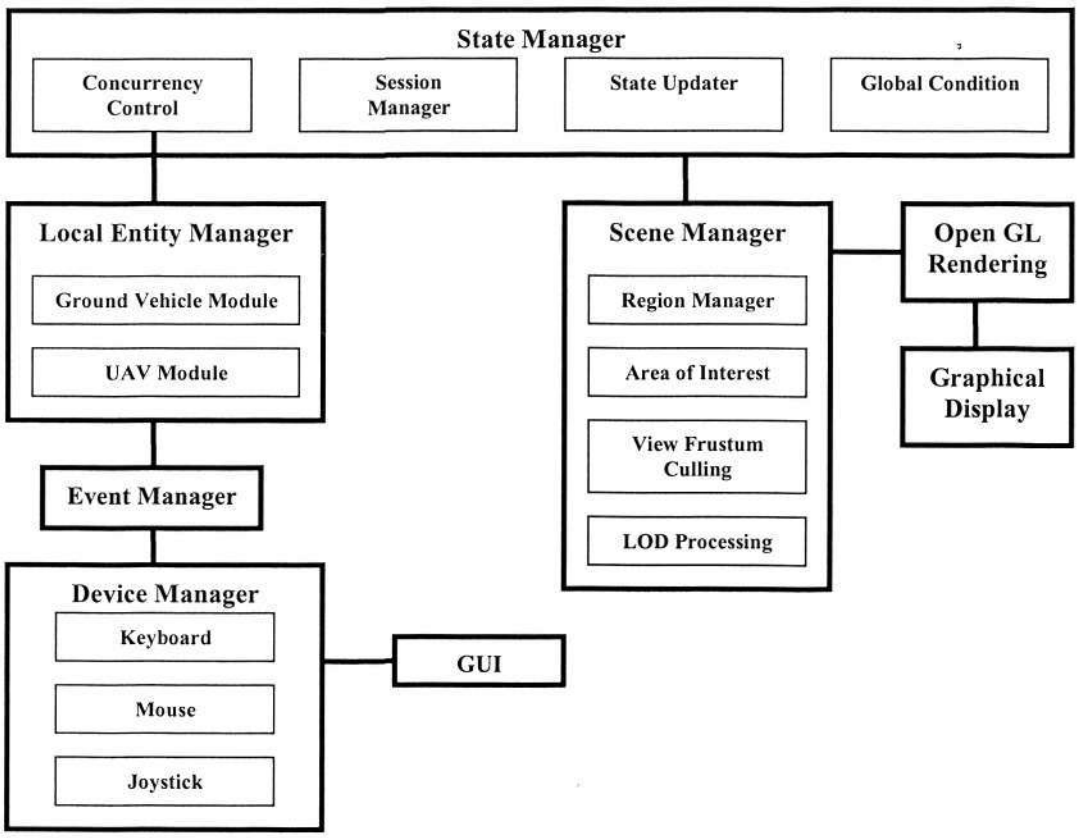

Figure 3.1: *ROBOSIM* System Architecture (Wee Ching, 2006)

There are a total of 7 managers in the ROBOSIM architecture and the followings are the more relevant ones to the implementation of UAV simulation in *ROBOSIM*.

- State Manager

State manager is responsible for maintaining the states of *various* entities, e.g. ground vehicles and UAV in the simulator. It consists of four subsystems: Concurrency

Control, Session Manager, State Updater, and Global Condition Manager. Their tasks include:

- providing an interface for dynamic creation and deletion of entities
- providing an interface for the various entitles to update their states in the simulator
- initializing the environment database during setup and modifying the environment database during execution
- parsing the command line input and configuration file, (un) load plug-ins
- providing the necessary interactive and real time rendering performance as every independent module can access shared information in the simulator.

The idea of the state manager is also present in various simulators, such as the Simulator Control Manager (SCM) created by Myeong-Wuk et al (2003) that manages virtual times of simulator components and is similar to the function of concurrency control of *ROBOSIM*. Other typical state manager is Event Controller in Simulation, Tactical Operations and Mission Planning (STOMP), which initializes simulation and provides facilities for scripting scenarios (Jones, 2003). A simulator built by Chang and Lindsay (2005) generates virtual airspace, and is equipped with control mechanism to (de)activate UAVs based on mission objectives and status of completion.

❑ Local Entity Manager

Similar to Ground Vehicles module, the local entity manager is where the UAV module is implemented. It is an entity simulator, which is responsible for maintaining the state of *individual* entity and simulates the result of its actions.

❑ Scene Manager

The scene manager is intended to provide advanced visual effect at the lowest possible processing cost. It comprises Region Manager, Area of Interest Manager, View Frustum Culling, and Level of Detail (LOD) processor. Their tasks can be summarized as follows.

- to preprocess complex scene data for real time use by the visibility, area of interest, and collision detection processor
- to optimize the computational load on the simulator by rendering the area of interest only
- to determine whether or not objects are visible in particular area
- to locally adapt surface geometry complexity to changing view parameters

Like the scene manager of *ROBOSIM,* some other advanced graphic visualizations are found in X-Plane of ASATE by McManus et al (2003) and the Flight Simulator that provides open GL based high resolution for Real Time Multi-UAV Simulator (RMUS) made by Goktogan et al (2003).

❑ External Resources

ROBOSIM uses many of freely available tools. Some of these tools and their applications in the simulator are described as follows.

- *OpenGL*, which is used as the Application Programming Interface (API)
- *Open Scene Graph (OSG)*, an open source, cross platform graphics toolkit that is used as the rendering engine
- *GLGooey*, an open source GUI toolkit, which was chosen as the GUI development tool

- *OPCODE* that specializes on Collision Detection (CD) module development. This library is taken from *CRYSTALSPACE (CS)* as a free and portable 3D Game Development Kit written in C++

❑ Device and Event Manager

Both device and event managers are used especially for manual operation. Device manager would read and preprocess data from attached input peripherals, e.g. Keyboard, Mouse, Joystick, and Steering wheel. The preprocessed data will then be passed to the Event Manager for further processing e.g. event queue to manage the event system.

3.1.2. Library Dependency Tree

A modular design concept has been employed in *ROBOSIM* to break up a large program into manageable units that make complex programs easier to understand, and to create code that can be easily re-used for program development, test, and upgrade.

A modular program generally consists of a main module, which is statically linked and compiled as static libraries of executable file, and more than one auxiliary module that is dynamically linked. Library consists of classes, which program errors will cause the simulation to crash during the compilation. On the other hand, auxiliary modules or plug-ins are implemented in *ROBOSIM* as Shared Class Facility, which the malfunctions will cause the models to work inappropriately in the simulation. The approximate library dependency tree of *ROBOSIM* is shown in Figure 3.2.

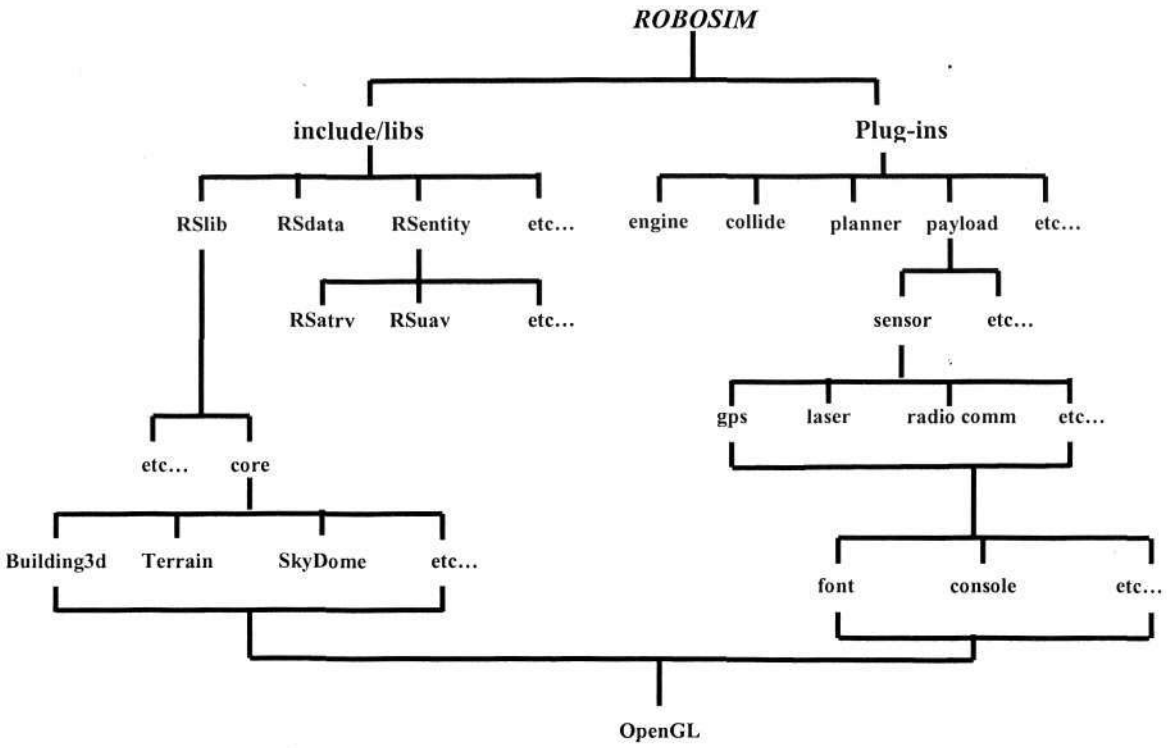

Figure 3.2: Library Dependency Tree (Wee Ching, 2006)

Some major components of *ROBOSIM* library are *RSlib*, *RSdata*, and *RSentity*. *RSlib* is responsible for the construction of environmental models, such as building, terrain, and sky dome. *RSdata*, on the other hand, provides the collection of environmental data, such as networks topology of roads and coordinate of plants. Lastly, *RSentity* is used to provide a base entity from which all the other entities can be inherited. Each entity must implement its own virtual function to:

- load entity model: graphical (3D model) and physical (specification data)
- create and initialize sensor attached to the entity
- process command and request
- update kinetic and kinematic behaviors

In real time simulators, entity class is generally required to simulate different types of agents. In STOMP built by Jones (2003), Simulated Object class provides a common

base containing relevant information of corresponding entities. On the other hand, entity based simulator created by Myeong-Wuk et al (2003) perceives UAV as an actor, which is a self-contained active object that has its own control thread and communicates with other actors through asynchronous message passing.

Each module of the simulator is written as a plug-in as far as possible. It is done independently as well as to ease the software test, maintenance, and upgrading. Some notable plug-ins are *RSengine* and *RSplanner*. *RSengine* represents an engine to encapsulate an application behavior, which occurs every frame. For example, there is a node in the scene graph, which represents a fish. An Engine class, e.g. *FishEngine*, subclass of *RSEngine* can be created to simulate the behavior of a fish. On the other hand, *RSplanner* provides the basic navigation capability to the simulated entity and serves as a foundation for higher level Artificial Intelligence process. More details about *ROBOSIM* architecture and classes can be found at Wee Ching (2006).

3.2. UAV module

Based on Figure 3.1 and Figure 3.2, *RSuav* is a module within local entity manager and inherited from *RSentity* in *ROBOSIM*. The model is based on that by Robbins and Anderson (1998) and later adapted to model *RRCUAV* and part of its environment. The followings are detailed descriptions of the conceptual models built in this module.

3.2.1. UAV Systems

Robbins and Anderson (1998) built a simple but sophisticated architecture that well separates Intelligent System, Control System, and Actuator as illustrated in Figure 3.3.

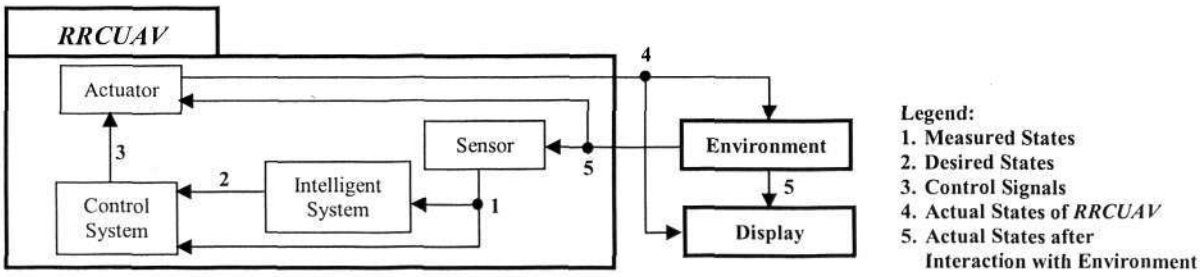

Figure 3.3: Architecture of *RSuav* Module

The architecture is basically a closed loop system, in which *Desired States* are generated from the Intelligent System module in terms of a 3D vector specifying the intended direction. The Control system module then minimizes the differences between *Actual States* and *Desired States* by sending *Control Signals* to the Actuator. They consist of commanded thrust T_c, commanded load factor n_c, and commanded bank angle ϕ_c. Based on this information, the Actuator module executes an action, which is represented by 9 *Actual States*: coordinates x, y, and z, orientations: pitch γ, yaw ϕ, and roll χ, thrust T, speed V, and load factor n. Some actions might also interact with the environment, for instance *RRCUAV* altitude z is restricted to ground level during takeoff and landing. Detailed modeling of each system is described as follows.

Propulsion

In basic control engineering, an actuator is defined as a power mechanism to activate a movement or process. In low level control, it might be a device, e.g. motor that converts an electrical control signal to a physical action. Such definition is applicable to the UAV, e.g. the elevator, aileron, rudder, and throttle receive control signals from the autopilot (see again Figure 1.9) and cause a resultant motion for the UAV.

The actuators of UAV belong to two major systems. Elevator, aileron, and rudder are categorized as aerodynamics and throttle is grouped as propulsion. Figure 3.4 illustrates the structure of Actuator module comprising aerodynamic and propulsion systems.

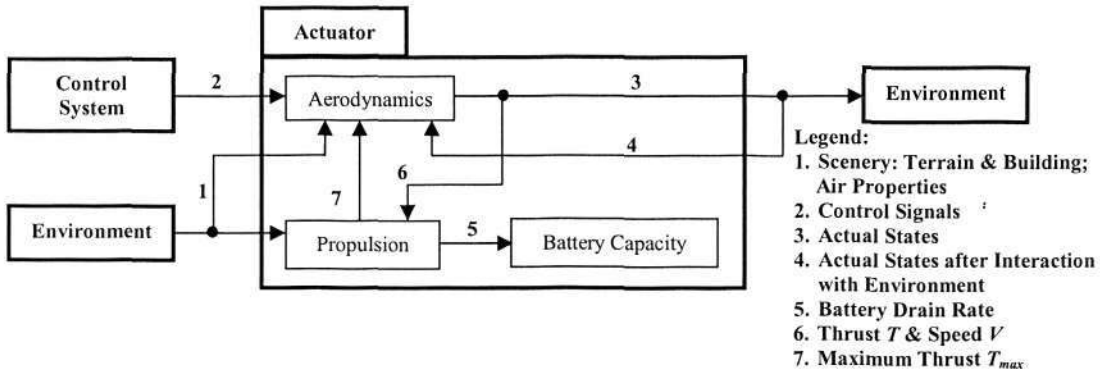

Figure 3.4: Diagram of Actuator

Some operational constraints, such as ceiling (maximum operational altitude) and takeoff distance are affected by the thrust T that is generated from the propulsion system. However, the original model made by Robbins and Anderson (1998) only defined maximum thrust T_{max} as a constant value used to restrict commanded thrust T_c. Additionally, fuel constraint was neglected in the initial model so that the operational period of the propulsion unit had never been calculated.

To refine this model, power consumption P, which is defined as battery drain-rate, is formulated as the product of generated thrust T and current speed V. With the addition of battery capacity as the parameters in *RSuav* module, operational time of *RRCUAV* can be therefore determined.

Instead of predefining maximum thrust T_{max}, this value was re-formulated as maximum engine output power P_{max_engine} divided by both current speed V and overall efficiency η, e.g. propeller and engine. Propeller efficiency is assumed constant at 60%

based on Kotwani et al (2004) study. Details on propulsion models are given in section B1.5. Propulsion.

Aerodynamics

Based on its configuration, the *RRCUAV* is categorized as a fixed wing UAV and unlike a ground vehicle, the aircraft is a typical Dubin's Vehicle that moves forward along paths of bounded curvature in 3D space and cannot reverse.

- Point Mass Model

The aircraft model is assumed to be point mass where all external forces e.g. weight, lifting force, thrust, and drag are applied at single point – the central of gravity. Therefore, the calculation of moment is simplified in this model.

In conventional model, the elevators are turned down to create a moment that will increase the attack angle of the wing in order to climb as illustrated in Figure 3.5. In point mass model, pitching is simplified by assuming the rate of pitching $d\gamma/dt$ to be positively related to commanded load factor n_c as formulated in Equation b.15 (see Appendix B. Mathematical Models of RSuav Module and Environment).

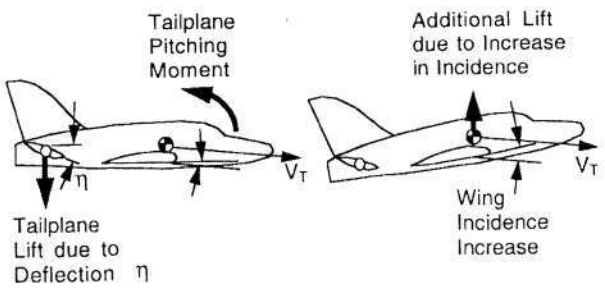

Figure 3.5: Maneuvering in the Pitch Plane (Collinson, 1996)

Yawing results from the combination of rolling and lifting force. Hence, the ailerons are given the opposing angular rate as illustrated in Figure 3.6 and the elevators are turned down. This maneuver is simplified in a point mass model as conceptualized in

Equation b.16 assuming the rate of yawing $d\chi/dt$ is positively related to commanded lifting force n_c and commanded bank angle ϕ_c.

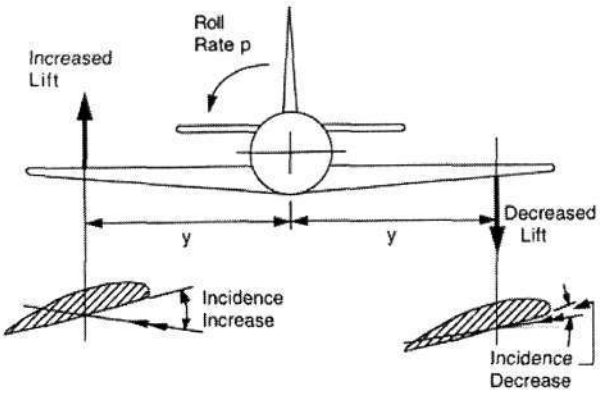

Figure 3.6: Rolling moment due to roll rate (Collinson, 1996)

Last but not least, the most observable difference between conventional and point mass model is that in point mass model, flight direction is always inline with body axis. This may not be true in real application, in which body axis is normally interfered by the moments generated by the actuators, e.g. elevators, ailerons from different parts of aircraft. The instability problem gets worse with wind disturbance. That is why a rudder, which is excluded in point mass model, is present in real application to keep the nose of aircraft pointed along the direction of travel.

❑ Wing Model

The parameters of a wing constitute two basic aerodynamic components, which are Lift and Drag. As specified earlier in section 2.2.2. Aerodynamic, the *RRCUAV* lifting coefficient is ranged between the minimum Cl_{min}=*-0.5* and maximum lifting coefficient Cl_{max}=*1.4*. This means that that lifting force works within this range as well. Flaps are deployed only during landing to cut stall speed as well as to increase drag. It is modeled that by having flaps with Cl_{max} up to 2.2, stall speed can be reduced to 11.2 m/s.

On the other hand, Drag is divided into surface/parasite and polar/induced drag. Surface drag is calculated using boundary layer theory. To reduce the complexity of problem, all parts of the aircraft is projected onto a thin plate, which the thickness is assumed zero. The thin plate area exposed to airflow is then called as reference area *S*. It is also regarded as the wing area of modeled aircraft since wing is the most significant part that creates lift and drag.

Induced drag is caused by wingtip vortices due to the pressure difference between lower and upper part of wing. As conceptualized in Equation b.13, this drag is proportionally to load factor *n* and inversely proportional to speed *V*.

It is only during takeoff and landing that the ground partially blocks the trailing vortices and thus, decreases the amount of downwash generated by the wing. For simplicity, such drag is modeled between stall speed of 15.1 m/s and maximum speed of 39.6 m/s since both takeoff and landing occur below stall speed. Figure 3.7 shows the profile of surface, induced and total drag over speed *V* within this range.

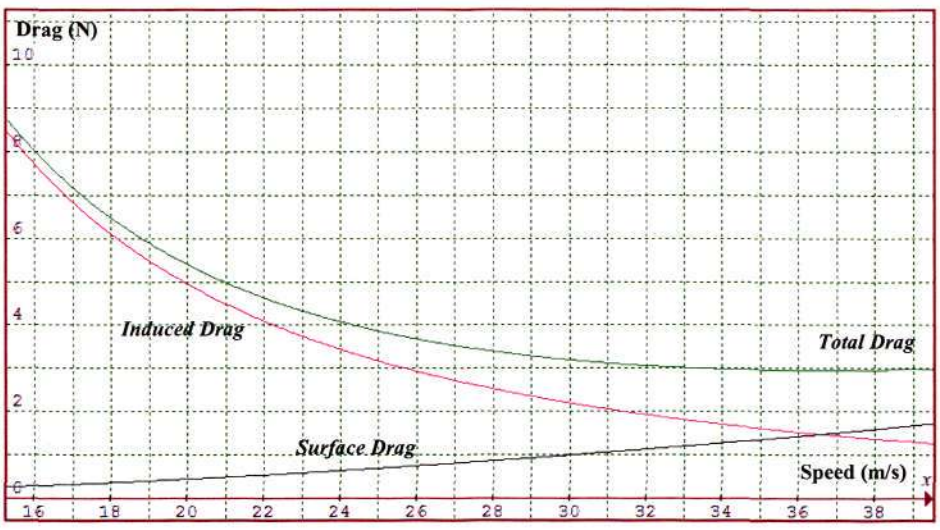

Figure 3.7: Drag profiles over aircraft speed V

The complete mathematical formulations of 9 *Actual States* are available in section B1.4. Aerodynamic. The information about RRCUAV Parameters in *RSuav* module can be found in section B2. RRCUAV Parameters.

Sensor

The sensor model is divided into the aircraft state and external sensors module. Aircraft state sensor, including the gyroscope that measures the aircraft orientation and position is useful for UAV internal management. Some aircraft state sensors also require information from the outside environment, such as pitot tube, which provides information of air properties to determine aircraft speed.

This project does not put too much emphasis on UAV internal management. It is assumed that that actual aircraft states are similar to the measured ones, and do not consider problems of signal errors and delays.

Figure 3.8 illustrates the sensor module that provides the information of aircraft states: speed V, orientation χ γ ϕ, load factor n, and battery level to the Control System and the information of external world (e.g. measured distance to detected object) to Intelligent System respectively.

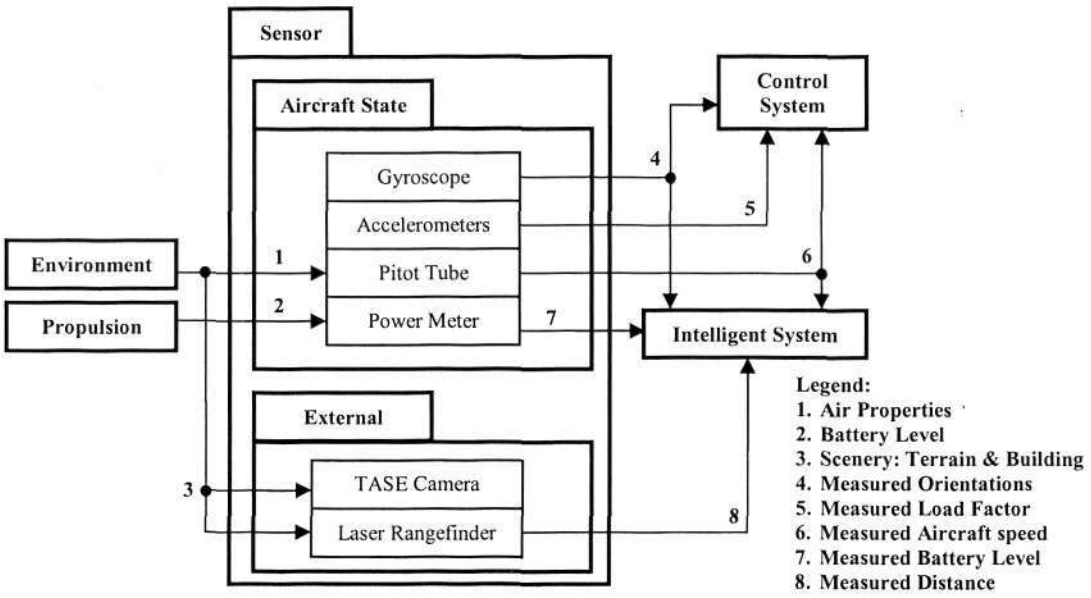

Figure 3.8: Diagram of Sensor

External sensor, on the other hand, deals more on the sensing capability of outside world in order to construct the situational awareness for UAV. External sensor is modeled using the specification of the sensor, e.g. type of information, such as distance and angular location, accuracy, resolution, and data rate. The followings are the modeled external sensors.

- TASE Gimbaled Camera

TASE Gimbaled Camera is provided as a plug-in in *ROBOSIM*. A Pan-Tilt-Zoom mechanism is modeled with pan angle of ±180°, tilt angle of +23°/-203° and pointing resolution of 0.05°. The captured image will be then displayed on the camera window as shown in Figure 3.9.

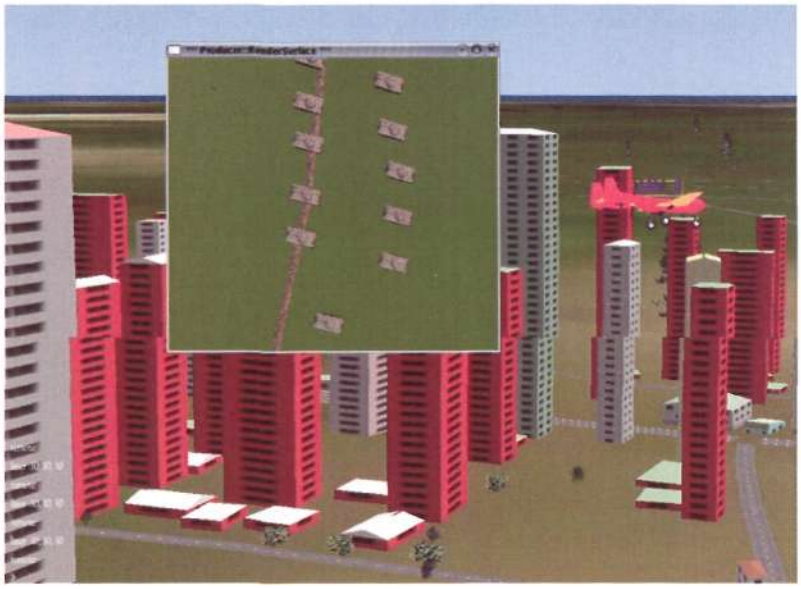

Figure 3.9: Camera View *Producer::RenderSurface* in *ROBOSIM* Graphic Display

Since there is no current technology that supports automatic object recognition for mini UAV in real time, this task is fulfilled by the operator. That is why, automatic target tracking performed by TASE Gimbaled Camera as SPOI mode is based on pre-programmed behavior, which requires operator or Waypoint Path Planner (WPP) to define target location. By adopting a geometrical-calculation based planner that directs the camera pan and tilt, the camera view will always point at the target regardless of *RRCUAV* position and orientation.

- Laser Rangefinder

The model of a laser beam is based on CD that reports and simulates all geometric contacts between two objects. *rayProximity* class then returns the distance between the intersection point and the origin of laser beam. This distance is then used for the calculations of sensor accuracy, resolution, and loop rate and then regarded as the measure of distance.

The range of laser rangefinder is known to be 365.76 m, which is modeled as the maximum distance of the laser beam. With a defined resolution of 0.2 m, measured distance is a multiple of 0.2 m. For instance, true distance of 120.34 m will be represented as measured distance of 120.20 m. Having an accuracy of 1 m meaning measured distance is then modeled as the +/-1 m random deviation of the true distance. Measured distance of 70.80 m with 0.2 m resolution and 1 m accuracy might represent true distance of 70.13 m. Laser rangefinder updates the measured distance every 0.1 second to Intelligent System. This is according to the specification of sensor data rate of 10 Hz (see again section 2.3.2. Laser Rangefinder).

Intelligent System

Intelligent System module is a typical Dynamic trajectory Smoother (DTS) that works based on given waypoints from either WPP or operator via GUI. It generates *Desired States*, which are originated from a unit vector $\vec{V}_T'$ indicating the intended direction in 3D space. Similar to DTS that produces desired heading and position, *Desired States* consist of desired speed V_d, pitching γ_d, and yawing χ_d. The complete mathematical formulation can be found in section B1.2. Intelligent System.

In the Intelligent System module, simple WPPs are constructed from four major autonomous behaviors. They are set based on the decision flow illustrated in Figure 3.10, in which the blue rectangular illustrates each autonomous behavior and it is described as follows.

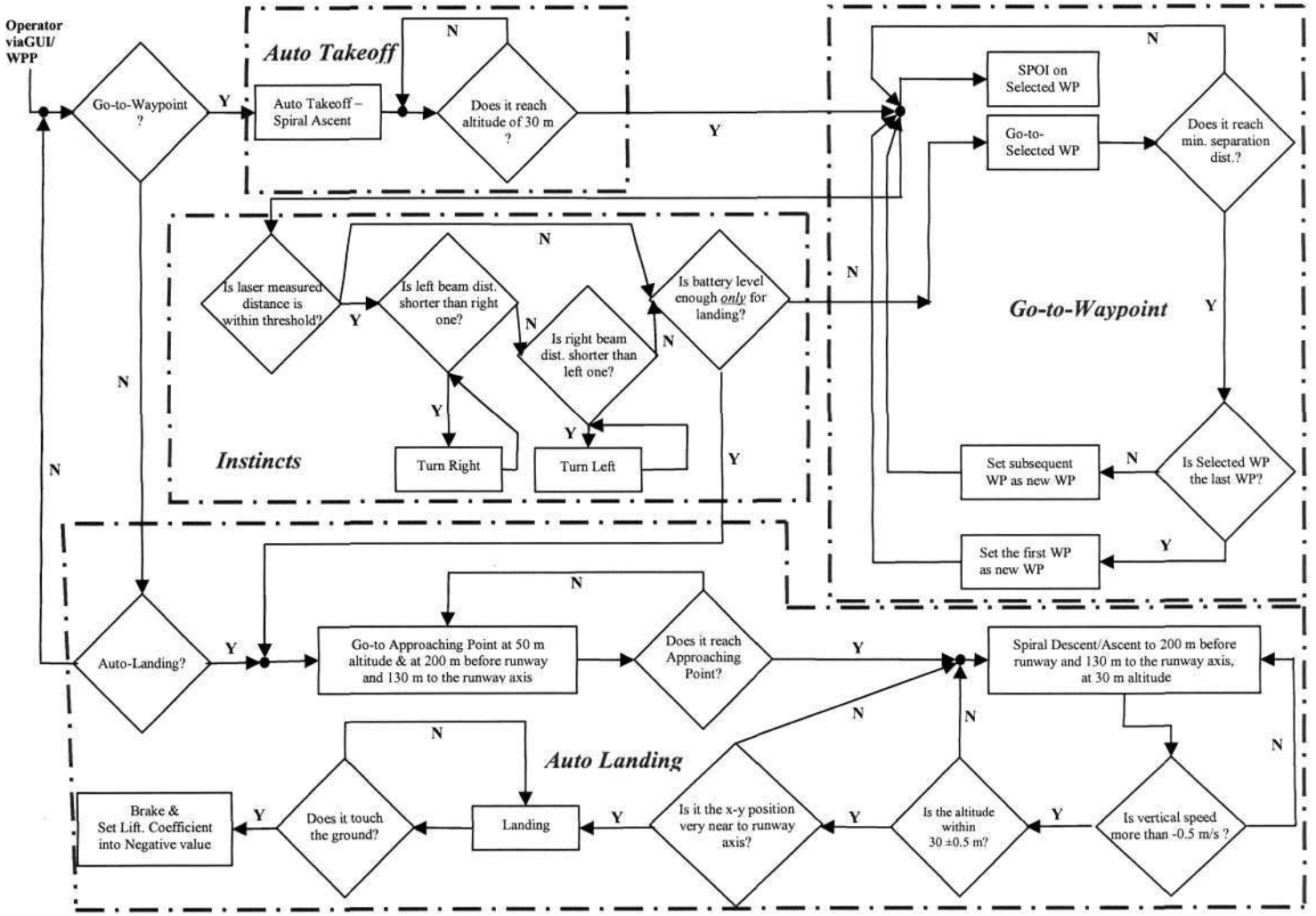

Figure 3.10: Decision Flow of *RRCUAV* Autonomous Behaviors

❑ Automatic Takeoff

As go-to-waypoint mode is selected, the number and location of waypoints are defined. *RRCUAV* initially accelerates along the runway and starts climbing as stall speed is reached. Due to limited space offered in urban area, *RRCUAV* performs automatic spiral ascent to 30 m altitude just after the lift-off.

❑ Go-to-waypoints

At an altitude of 30 m, spiral ascent is terminated and *RRCUAV* then moves towards the first waypoint. If there is only one waypoint defined, *RRCUAV* will encircle it but if there are more, it will move in a closed loop according to defined sequence of waypoints. It is then followed up by activating SPOI mode, which camera will automatically track the current waypoint.

The algorithm of go-to-waypoint is developed from target acquisition instinct, which was originally designed by Robbins and Anderson (1998) as one of collaborative instincts. Target acquisition instinct directs *RRCUAV* to move towards defined waypoint by using the virtual attraction force of potential field planner described in details in Section A1. Prototype of Simulator.

By the addition of a waypoint scheduler, target acquisition instinct will turn into pre-programmed behavior, which can be further developed for the optimization of mission performance. This can be realized by adopting any competent WPPs, such as Savla et al (2005) and McLain et al (2001).

❑ Instinctive Behaviors

Instinct, by definition, is a behavior that is mediated by reactions, which are principally the identification of a circumstance with near-term implications and the

implementation of a pre-established response to rapidly mitigate the situation. Generally, instinct is less systematic, unplanned, and more reactive than proactive due to limited time to plan. This is because instinct generally reacts to spontaneous stimuli, e.g. threats to preserve individual survivability.

Robbins and Anderson (1998) constructed intended direction $\vec{V}_T'$ as a linear combination of virtual repulsion and attraction force. Unlike attraction force that works for target acquisition instinct, repulsion is employed to prevent UAV approaching any threats, e.g. potential collision to nearby object.

Modified from its original model, intended direction $\vec{V}_T'$ is then formulated in Equation 3.1 as a weighted average of collision avoidance and target acquisition vectors.

$$\vec{V}_T' = w_{ca} \times \vec{V}_{ca}' + w_{ta} \times \vec{V}_{ta}' \qquad (3.1)$$

where $w_{ca} + w_{ta} = 1$ and both w_{ca} and w_{ta} are ranged between 0 and 1. The scalar scale factors of collision avoidance w_{ca} and target acquisition w_{ta} are used to represent the strength of instinct at a given time. For example, when the *RRCUAV* is about to collide with an object or the ground, the scale factor for collision avoidance w_{ca} will have the largest value of 1 at that time and thus, it may cancel the current operations, e.g. go-to-waypoint, to perform an escape maneuver.

Collision avoidance is programmed as the highest hierarchical instinct in order to prevent any damage to the *RRCUAV* and the fatality caused by the collision. Therefore, scale factor of collision avoidance w_{ca} is firstly assigned and thus, target acquisition w_{ta} is taken as the remaining portion in the overall intended direction $\vec{V}_T'$.

In *RRCUAV*, collision avoidance instinct is implemented to avoid the collision of any nearby static objects in urban area. The presence of nearby object is indicated if one of the measured distances of the laser beam used as obstacle avoidance sensor is lower than its maximum range and if the value is within defined threshold. In this case, obstacle avoidance will be then activated by setting the scale factor w_{ca} to 1.0. If the right beam is shorter than the left beam, Intelligent System module then instructs the Control System module to change the current heading to 90° to the left to execute the maneuver of obstacle avoidance and vice versa.

❑ Automatic Landing

a. Procedures

An operation inspired by Riseborough (2004), allows the operator to select and to cancel auto-landing mode via GUI. If it is selected, automatic landing begins with *RRCUAV* moving towards the approaching point, which is at 50 m altitude, and 200 m before the runway and 130 m to the runway axis.

As it is near the approaching point, the *RRCUAV* will carry out a spiral descent/ascent, which is the second phase motivated by Quigley et al (2005). Spiral descent/ascent is useful to ensure landing accuracy and safety but it lengthens the landing time. This phase is terminated if: (i) the *RRCUAV* is positioned near to the runway axis, (ii) its height is within 30 ±0.5 m, and (iv) its vertical speed is more than -0.5 m/s.

With these conditions fulfilled, the *RRCUAV* will gradually descend at gliding slope of less than 10°. Flaps are then activated by setting Cl_{max} into 2.2 to decrease stall and forward speed. As the *RRCUAV* touches the ground, it sets the lifting coefficient *Cl* into negative value to cut the lifting force and lastly, the brake is applied to slow the UAV.

b. Emergency Landing

Emergency landing is another version of automatic landing that can be executed if it is estimated that battery capacity is enough *only* to perform landing. Although it is to maintain the survivability of *RRCUAV*, automatic landing is a pre-programmed behavior and is different from instinctive behavior.

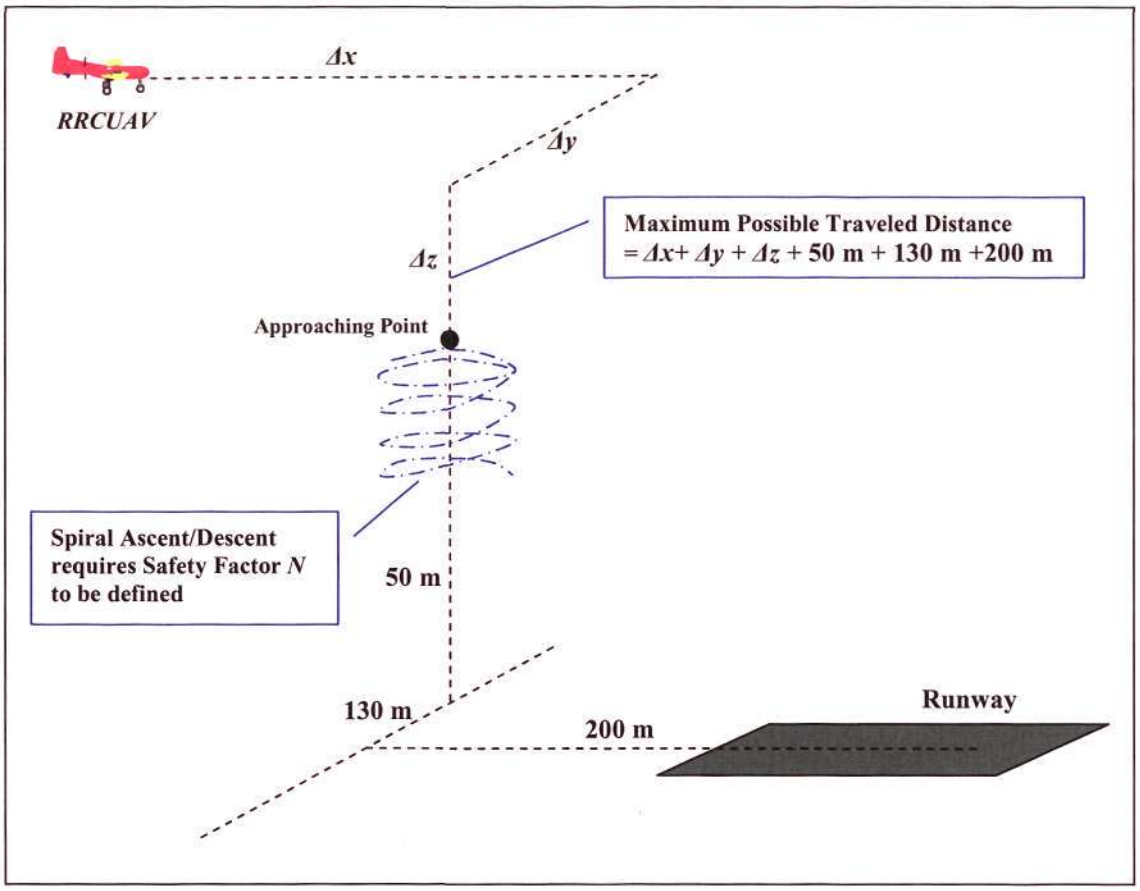

Figure 3.12: Maximum Possible Distance for Landing

The estimation is based on the maximum possible flying time derived from the maximum possible traveled distance divided by stall speed. This distance is the sum of x, y, and z coordinates between current position of the *RRCUAV* and the runway as illustrated in Figure 3.12. However, due to the need to carry out spiral descent/ascent, there must be a defined safety factor *N* added to the calculation. Based on a number of tests, N needs to be at least 1.5 to ensure safe emergency landing.

During automatic landing, the power consumption P is estimated to be around 500 W, in which 400 W is allocated for the *RRCUAV* flying 10% margin to the stall speed and another 100 W is for spiral ascent that is executed in the second phase.

If the product of safety factor N, maximum possible flying time, and estimated power consumption P is equal or bigger than current battery level, then emergency landing is activated. The complete pseudo-code of these four autonomous behaviors is provided in *Appendix C. Pseudocodes of RRCUAV Autonomous Behaviors.*

Control System

Control system module is a high level controller used to minimize the difference between *Actual States* and *Desired States*. It converts the kinetic states of the *Desired States* into kinematic states that are represented by three commanded values: thrust T_c, load factor n_c, and bank angle ϕ_c according to the equations formulated in section B1.3. Dynamic Inversion Control. This is typical to Trajectory Tracker (TT) that receives desired heading from DTS and generates a set of commands to UAV. The only difference is the commands are represented in kinematic states in Control system module. The low level controllers for lateral and longitudinal stability are excluded due to the simplification of the aerodynamic model.

Communication

In this project, wireless communication between *RRCUAV* and ground control station is modeled to determine the communication range. It is done by defining the maximum distance where information is received at minimum acceptable level by both parties.

Goldsmith (2005) stated there are three methods of modeling communication system. The most accurate one is by modeling 3D environment to simulate the signal reflection, refraction, absorption, and interference. The second most accurate technique is through the collection of statistical data on the signal propagation in particular area. Lastly, the least accurate method is by simply adopting the theoretical model involving propagation loss that is calculated from transmitting power, antenna gain, terrain, weather condition, distance, and interference.

However, none of these methods is adopted as it will increase the complexity of the problem and thus, distract us from the focus of this project, which is to study the autonomous behaviors of *RRCUAV*. For this reason, communication range is defined at fixed value of 32 km. If the distance exceeds the predefined threshold value, the communication signal will be considered lost. Communication range of 32 km is chosen based on the major trend in mini UAVs. For example, *Baryaktar*, *Tern*, and *Silver fox* have a communication range of 20 km, 50 km, and 36 km respectively.

3.2.2. Environment

Environmental modeling is another issue in UAV simulation to facilitate the construction of Situational Awareness for a UAV. The followings are environment models built in the *ROBOSIM*.

Atmosphere

Troposphere and lower Stratosphere lying between 0 and 18 km are the only Atmosphere layers that an aircraft can operate in. This is because at these layers, air density is sufficient for the operation of the propeller based engine to produce thrust.

Troposphere needs to be modeled by the variation of air pressure P, temperature T_h, and density ρ respect to altitude z. This is because the variation affects aircraft characteristic in certain aspects. For example, higher altitude results in lower air density ρ and thus, the aircraft needs to increase speed V to maintain its altitude.

The atmosphere is modeled as a function in the *RSuav* module. It is a mathematical representation of the 1976 Committee on Extension to the Standard Atmosphere (COESA) of United States. Details can be seen at section B3. COESA Atmosphere.

Sky Dome

Based on Figure 3.2., sky dome is a class inherited from *RSlib*. It is the top part of a sphere, where both ground vehicle and UAV are located inside the dome seeing the inner surface. As shown in Figure 3.13, sky dome is basically a series of triangles that reasonably approximate the shape of the sphere. To add sense of reality, clouds are drawn onto the dome by blending a luminosity texture with required texture color.

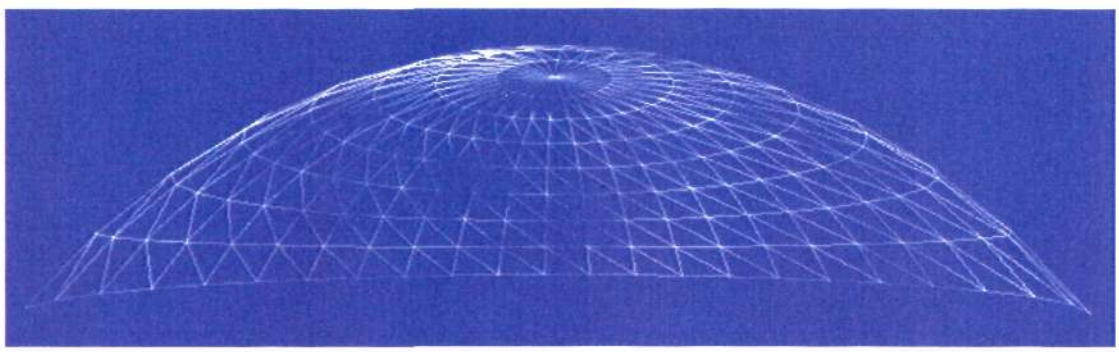

Figure 3.13: Sky dome drawn as lines (Wee Ching, 2006)

Terrain

RSTerrain class, which is inherited from *RSlib*, represents a terrain that is a part of the of the earth surface described by a set of parameters, e.g. elevation, vegetation, and time of day. It implements the terrain-rendering engine based on the McNally (1999) algorithm and the algorithm for regular-grid terrain Level of Detail (LOD) that

effectively renders in long viewing distance and an unconstrained viewpoint. Height Fields, on the other hand, is used as the solution to store the features inherent in a landscape. This function helps the height adjustment of *RRCUAV* during takeoff and landing.

Building

RSBuilding class is inherited from *RSlib* and used as part of the environment model in *RRCUAV* obstacle avoidance. It consists of *RSBuilding3d* that implements the procedural construction of 3D geometry using the *RSBuilding* properties. On the other hand, *RSStructure* acts as a container for *RSBuilding* objects and includes the information on location, footprint, type and color of walls/trim/roof, and number of stories/doors/windows. Figure 3.14 shows blocks of buildings developed in simulated urban area in *ROBOSIM*.

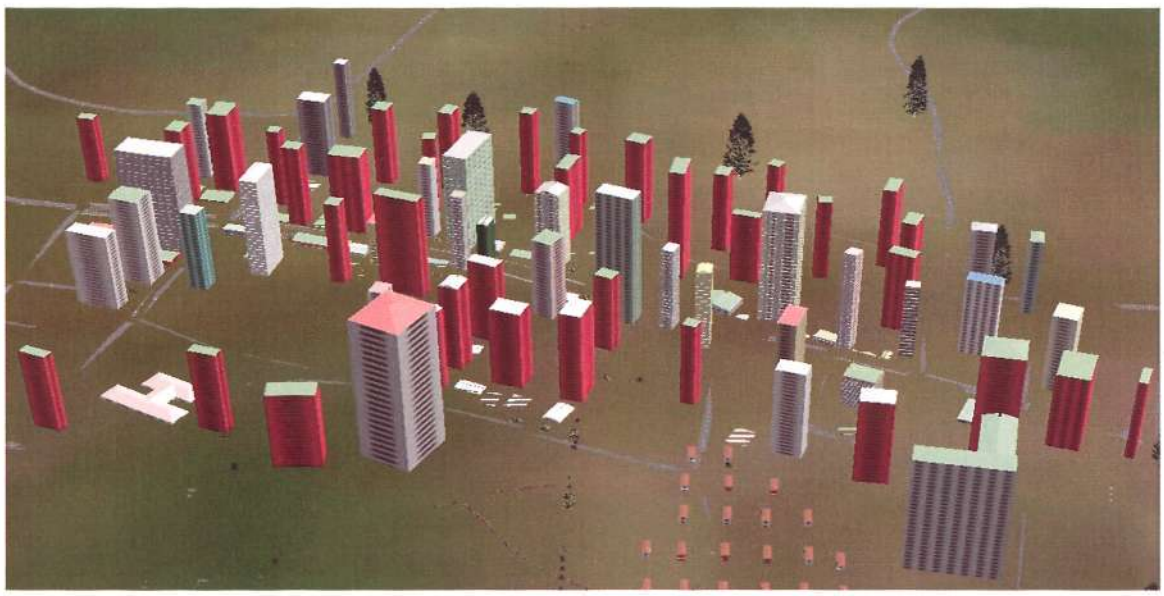

Figure 3.14: Blocks of Building in Synthetic Urban Area

3.3. Chapter Summary

ROBOSIM is principally the same with other UAV simulators, which perform numerical solution of differential equations, periodically updates the states of entity, and outputs of continuous dynamic simulation. The role of *RSuav* module in *ROBOSIM* is just like the existing *ATRV* or *Pioneer* class, which is an extension of robot entity class – *Rsentity*. In this project, UAV simulator is the product between *ROBOSIM* plug-ins, which specialize in 3D graphical drawing and the creation of virtual environment, and *RSuav* module that focuses on aerodynamic models and intelligent system.

Basically, UAV is a multidisciplinary system that mainly consists of Intelligent, Control, and Aerodynamic Systems as what Robbins and Anderson (1998) have represented in their architecture. In *RSuav* module, each system has been modeled in the simplest way yet noteworthy to address the issues of establishing a platform to study the autonomous behaviors of *RRCUAV*. Intelligent System module acts as a DTS, which receives a list of waypoints and generates desired headings. On the other hand, Control System module, which is a high level controller, functions as a TT receiving desired heading from Intelligent System module and generating commands to Actuator module, which models the kinematics of the mini UAV as a mass point with simplified equation of moments.

In the Intelligent System module, simple WPPs are constructed from a set of four major autonomous behaviors, which are pre-programmed and onboard planner based and then developed into three flight modes. Both automatic takeoff and landing are basically the modification of go-to-waypoint operation. On the other hand, obstacle avoidance is an instinctive behavior developed from the algorithm of virtual repulsion force.

Chapter 4

Model Validation

It is important to emphasize the understanding about the modeling process to know how to build a simulator run as a model of reality and relevant to the objective of project. These issues will be addressed in the beginning part of Chapter 4 by describing the comprehensive concept of modeling and simulation.

Above all, the credibility of a simulation is always about the overall quality of models involved in the simulator and a credible simulation will build the confidence about the usefulness of the simulator. That is why, the validation of UAV models described in Chapter 3 will be performed at the end of this chapter in three different areas: the formulation of theoretical models, the transformation from theoretical to computerized models, and lastly, simulated behaviors.

The result of model validations made in this chapter later shows that the basic features of simulation, e.g. gravity, aerodynamic, and power consumption worked according to reality and no anomalies were observed. The implemented models are qualified to be used as a simulation platform to study autonomous behaviors of mini UAV.

4.1. Basic Concepts of Model and Simulation

According to Abt (1964), a model is an actual or theoretical representation of the structure or dynamics of a thing or process. As the theoretical representation, Schlesinger (1979) then outlined the modeling process, which is shown in Figure 4.1.

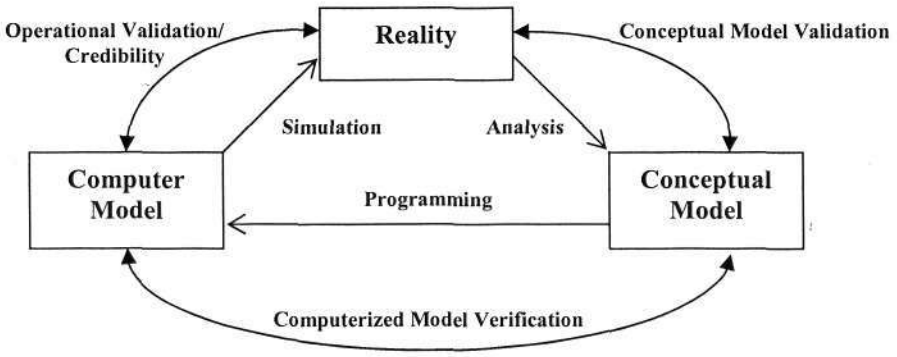

Figure 4.1: Simulation Space and Model Definition (Schlesinger, 1979)

The modeling process encompasses three phases as follows.

i) Analysis, which formulates the conceptual model from reality. This part is presented earlier in section 3.2. UAV module.

ii) Programming, which is the process of writing a source code based on the defined conceptual model, and it is described in more details in Appendix A. Simulator Development.

iii) Simulation, which demonstrates the imitated behaviors of reality and it is built from computerized models. Chapter 5 will perform the study on simulated behaviors.

Garzia et al (1990) mentioned that the completion of these three phases must be followed by three steps in deciding if a simulation is an accurate representation of the actual system considered, namely validation, verification, and credibility. Each of them is described as follows.

4.1.1. Conceptual model validation – Did I build the right thing?

Conceptual model validation is the process of determining that the theories and assumptions underlying the conceptual model are correct. A conceptual model is described via equations or other governing relationships and analyzed from reality.

Kleijnen (1999) and Sterman (1984) gave insight on the validation of a conceptual model using statistical techniques and reasoned that techniques applied would depend on the availability of data in real system.

Nevertheless, Shannon (1975) observed a phenomenon that a modeler nearly always tends to simulate too much detail rather than too little. The tendency among system analysts has too often been to transfer all the detailed difficulties in the real situation into the model, hoping that the computer will solve their problems. He explained that this approach is unsatisfactory not only because of the increased difficulty of programming the behavior of the model and the additional cost of computer experimentals, but also because the truly significant aspects may be lost amongst trivial details.

Therefore, systems, which may be extremely complex, should be simplified in the interest of obtaining practical solutions and yet still retain sufficient realism at hand. Martis (2006) added generally, the most powerful model is the simplest model, which expresses a valid relation.

4.1.2. Computerized Model Verification – Did I build the thing right?

Computerized Model Verification is the process of determining that the model implementation in the program accurately represents the developers' conceptual description of the model and the solution to the model (AIAA, 1998).

Whitner and Balci (1989) illustrated Programmed Model Verification (PMV) taxonomy that consist of 6 techniques, which are grouped based on categories of level of formality, complexity, human resource allocation, computational cost, effectiveness, and instrumentation. Some of them are described in the followings.

- Informal Analysis, which analyzes computerized model through the employment of informal design and deployment activities. This technique relies heavily on human reasoning and subjectivity by mentally exercising the model. It means evaluating the model using human mind, e.g. reviewing the logic behind algorithms and decisions, examining the effect of each implementation to the overall outcome of the model. Since it involves less mathematical formalism and is very intuitive, the quality depends on the level of knowledge and expertise of the individuals.
- Static Analysis that analyzes computerized model via characters of the static source code. It is used to assure the syntax programming language is being applied correctly as well as to monitor the behavior of pre-programmed model with respect to its use of model variables, e.g. when variable space is accessed and (de)allocated. The strength of this technique lies in the well known methods supported by a variety of commercially available automatic tools.
- Dynamic Analysis, which is accomplished by evaluating the model during its execution. Some of the examples are *black* and *white box*. *Black box* is accomplished by feeding inputs to the model and verifying the corresponding outputs. That means it concerns more what is produced by the "box" rather than what is inside the "box". In contrast, *white box* tests the model based on its internal structure, e.g. uses data flow and control flow graphs to verify the logic and data representation in the model. Although it is more effective having dynamic analysis compared to informal and static analysis, it requires the program to be executed with certain computational cost.

4.1.3. Operational Validation/Credibility – Should results from the model be believed?

Operational validation/credibility demonstrates that the computer model possesses a satisfactory range of accuracy in comparison with reality and consistent with its intended application (Giannasi et al, 2001). Credibility is another term used by Pace (2004) to define the overall quality of model or simulator and the function of available information of conceptual Validation and Verification (V&V), the reputation of those who developed, and its history of use.

Coyle (1977) also pointed out that operational validation is the process of establishing confidence in the usefulness of a model. Forrester and Senge (1980) then added that the level of confidence in the model can be increased gradually by passing more tests. Sterman (2000) controversially argued that V&V are impossible, the emphasis should be more on model testing to build a confidence that a model is appropriate for the purpose. There is no absolutely valid model since there could always be identified a counter test to which the model did not conform to completely.

A wide range of tests to build confidence in a model have been developed by authors, such as Forreseter and Senge (1980) who mentioned model validation can be done by inspecting the model structures and behaviors. It is presented as lists of questions, which some of them are illustrated as follows.

a. Validating Model Structure:

- Is the complexity, richness of details appropriate for the audience to study?
- Do modeled parameters correspond numerically to real life?
- Is model structure appropriate for the model purpose?

b. Validating Model Behaviors:

- How well do the modeled behaviors match observed behaviors of the real system, in terms of symptom, multiple mode, and behavior characters?
- Does the model under some tested circumstances produces dramatically unexpected behaviors, not observed in real system?
- Can plausible shift in parameters cause model to fail behavior test previously passed?

Khazanchi (1996) proposed another scheme, which some of the examples are described as questions (Q) and answers (A) in the followings.

- Q: Is it plausible? A: Demonstrated by deduction from past research and theories
- Q: Is it feasible? A: Open to mathematical, illustrative, and graphical characters
- Q: Is it pragmatic? A: Have some degree of logical self-consistency with other concepts in the discipline
- Q: Is it empirical/experimental? A: Must have empirical/experimental testability

4.2. Evaluation of Models

Having described the architecture of *ROBOSIM* and models in the *RSuav* module, the following section explains the validation of models implemented in Chapter 3.

4.2.1. Conceptual Validation

One technique of the conceptual validation in this project is done by clarifying the assumptions and theoretical foundation of the models. Assumption is made to limit the degree of model details as well as to define its relevance to the project. The formulation of assumption and theory underlying the models has been discussed earlier at section 3.2. UAV module.

Using different versions of calculation to validate the models in the *RSuav* module is another validation method employed in this project. Such calculations are governed by high level equations, which are derived from basic aerodynamic parameters and described in the followings.

Takeoff Distance

Takeoff distance can be derived from Hale (1984), who correlates the distance to be proportionally related to kinetic energy and inversely to available thrust as represented by Equation 4.1.

$$L_{to} = \frac{V_{to}^3}{2.g.\sigma.\eta_{total}.\left(\frac{P_{motor}}{W}\right)} \qquad (4.1)$$

Based on Equation 4.1, at given input power P_{motor} of 1451.9 watt, overall efficiency η_{total} of 50% (motor and propeller), takeoff speed V_{to} of 18.2 m/s (1.2 times of stall speed), and by assuming the ratio density over sea level density σ of 1, the calculated takeoff distance L_{to} is 36.8 m.

The simulation shows that *RRCUAV* needs 37.6 m to lift off. The distance is measured from the initial position to the first point, where *RRCUAV* altitude is more than the terrain height (see the video clip at CD attached in this report for more details). This result is pretty close to the mathematical calculation given in Equation 4.1 and therefore, it enhances the confidence of the implemented model in the aspect of takeoff.

Minimum Turning Radius

Based on the theoretical equations from Hale (1984), minimum turning radius can be determined by following the mathematical procedures shown from Equation 4.2 to

4.5. Firstly, minimum speed $V_{stall,turn}$ is defined in Equation 4.2 to ensure stability during turning.

$$V_{stall,turn} = \left(\frac{2.\eta_{total}.P_{motor}}{\rho.S.\sigma.\left(C_{DO} + k.Cl_{\max}^2\right)} \right)^{1/3} = 24.65m/s \qquad (4.2)$$

In Equation 4.3, $V_{stall,turn}$ is then used to calculate loading factor $n_{stall,turn}$, which is the ratio between lifting force and weight.

$$n_{stall,turn} = \frac{\rho.\sigma.V_{stall,turn}^2.Cl_{\max}}{2W/S} = 2.66 \qquad (4.3)$$

By using geometric calculation, maximum bank angle for turning is determined in Equation 4.4.

$$\varphi_{stall,turn} = \cos^{-1}\left(\frac{1}{n_{stall,turn}}\right) = 67.88^{o} \qquad (4.4)$$

Lastly, minimum turning is determined from Equation 4.5.

$$R_{stall,turn} = \frac{V_{stall,turn}^2}{g.\tan(\varphi_{stall,turn})} = 25.18m \qquad (4.5)$$

Interestingly, the simulation on minimum turning radius illustrated in Figure 2.6 gives exactly the same result as the ones calculated from Equation 4.2 to 4.5. It therefore validates the conceptual models of equation b.16 (see Appendix B) that organizes aircraft turning in *RSuav* module.

4.2.2. Verification

Various verification methods have been addressed in this project. In section 3.1 and 3.2, the architecture of *ROBOSIM* and the structure of *RSuav* module have been described in the decision flow diagram and pseudo-codes as part of author's Informal

Analysis. It is not only to evaluate the resulting model for completeness and consistency but also to seek justification for the various design and development decisions made.

As the compiler is written for C++, the syntax of *RSuav* module need only a static analysis (see again section 4.1.2. Computerized Model Verification). Finally, black box tests were conducted to study how close the computed model behaves based on the formulation of conceptual model. The result of each test is summarized in Table 4.1 and described as follows.

Table 4.1: Black Box Testing

No	Conceptual Model	Simulated Behaviors of Computerized Model
1	Equation b.15 and b.12 (see Appendix B) describes the gravitational effect on *RRCUAV* pitching rate and acceleration.	If *RRCUAV* descents too long or if *RRCUAV* stalls, the speed is multiplied due to free fall.
		As *RRCUAV* climbs, engine power is mostly allocated to create thrust compensating gravity so it is difficult to accelerate.
2	Equation b.10 correlates the effect of maximum attack angle on *RRCUAV* lifting force.	Due to difficulty of acceleration, *RRCUAV* climbs at merely constant speed. However, higher altitude results in lower air density and to maintain lifting force, attack angle needs to be increased. As it reaches maximum angle, *RRCUAV* stalls.
3	Equation b.16 includes the effect of *RRCUAV* bank angle and speed on yawing rate	Figure 4.4 shows at higher bank angle and lower speed, turning radius becomes smaller
4	Effect of thrust and speed on power consumption (Equation b.25)	As *RRCUAV* flies 10% above stall speed, it can stand for 1.3 hours but when it flies at maximum speed or climbs continuously, it only stands from around 20 to 30 minutes only.

Gravitational effect on *RRCUAV* pitching rate and acceleration

Figure 4.2 is presented to verify the gravitational effect on aircraft pitch rate and climbing speed. It shows the trajectory plot (x-z coordinate/side view) of *RRCUAV* climbing. During this event, the engine power is mostly allocated to create thrust compensating gravity so it is difficult to accelerate. The aircraft speed then saturates at 95 km/h and spends almost 100% of maximum thrust (as indicated in speedometer and thrust-meter display).

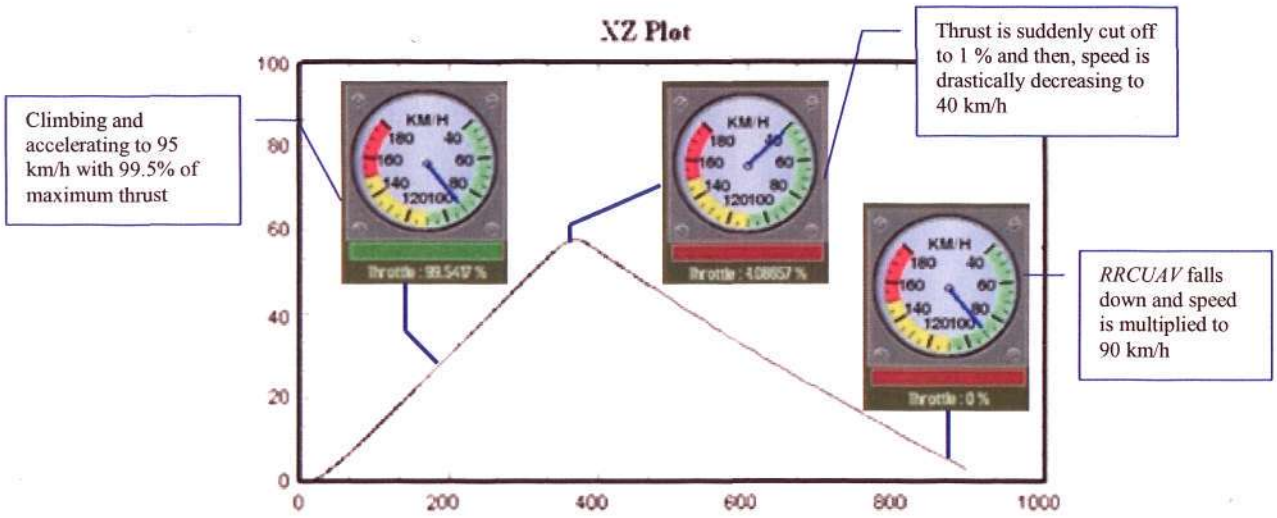

Figure 4.2: Black box test on gravitational effect

At merely 60 m altitude, operator then purposely reduces aircraft speed below stall speed by cutting the thrust and thus, *RRCUAV* falls down. Based on recorded flight data in the simulation, the speed is known to be multiplied due to gravitational effect.

Effect of maximum attack angle on *RRCUAV* lifting force

In the test shown in Figure 4.3, *RRCUAV* climbs at constant speed of 55 km/h and spends from 65% to 100% of maximum thrust as it flies higher. Although both tests in Figure 4.3 and Figure 4.2 experience stalling, the causes of stalling are different. The *RRCUAV* stalling in Figure 4.2 is due to engine shut down which then decelerates aircraft speed below stall speed. On the other hand, the *RRCUAV* stalling in Figure 4.3 is caused by low air density, which provides insufficient lifting force.

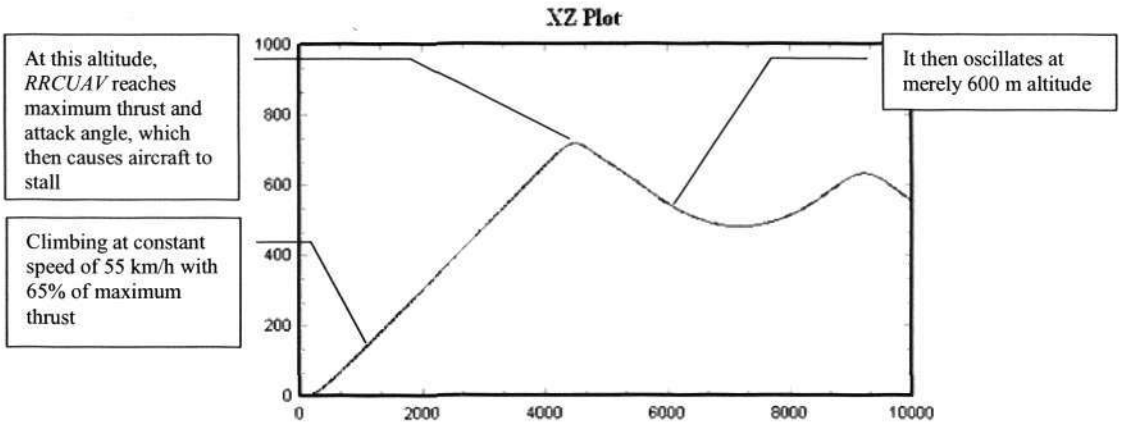

Figure 4.3: Black box test on the effect of maximum attack angle

Table 4.1 mentioning that having constant climbing speed (in this case 55 km/h), the *RRCUAV* needs to maintain lifting force by increasing attack angle to compensate lower air density. However, it is limited by maximum attack angle, which causes aircraft to stall. That is why in this test, *RRCUAV* could not go beyond 800 m altitude and keeps oscillating.

Effect of *RRCUAV* bank angle and speed on yawing rate

Three scenarios with different combinations of bank angle and speed are presented in Figure 4.4 with *RRCUAV* performing go-to-waypoint over a coordinate of x=2500 m, y=0 m, and z=30 m.

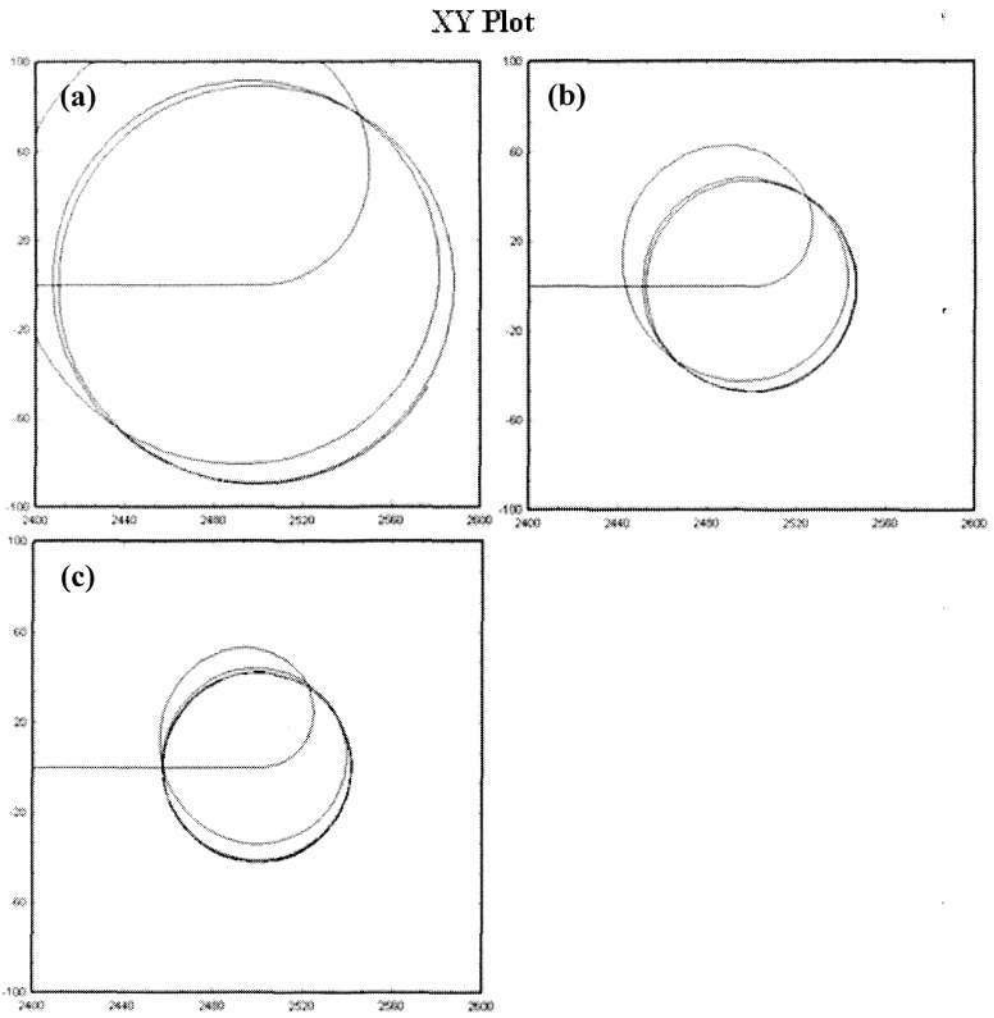

Figure 4.4: Black box tests on the effect of *RRCUAV* bank angle ϕ and speed V: (a) ϕ=16.5°& V=16.1m/s, (b) ϕ=48.2°& V=22.7m/s, (c) ϕ= 65.9°& V=30.3m/s

The first scenario is set as *RRCUAV* flying near to stall speed, 16.1 m/s and maximum bank angle of 16.5° and it results in steady state turning radius of 90 m. The second one is set at moderate speed of 22.7 m/s and maximum bank angle of 48.2°, and it results in 45 m turning radius. Lastly, the most maneuverable scenario is set to be 30.3 m/s with maximum bank angle of 65.9 ° and it results in turning radius of 40 m. By using this method, various combinations of bank angle and speed are simulated and the results are then summarized in Figure 2.7.

Effect of thrust and speed on power consumption

The last verification test is performed to study the effect of operational speed and thrust on *RRCUAV* endurance. As shown in Figure 4.5, *RRCUAV* performs horizontal flight at 100 m altitude in two different speeds.

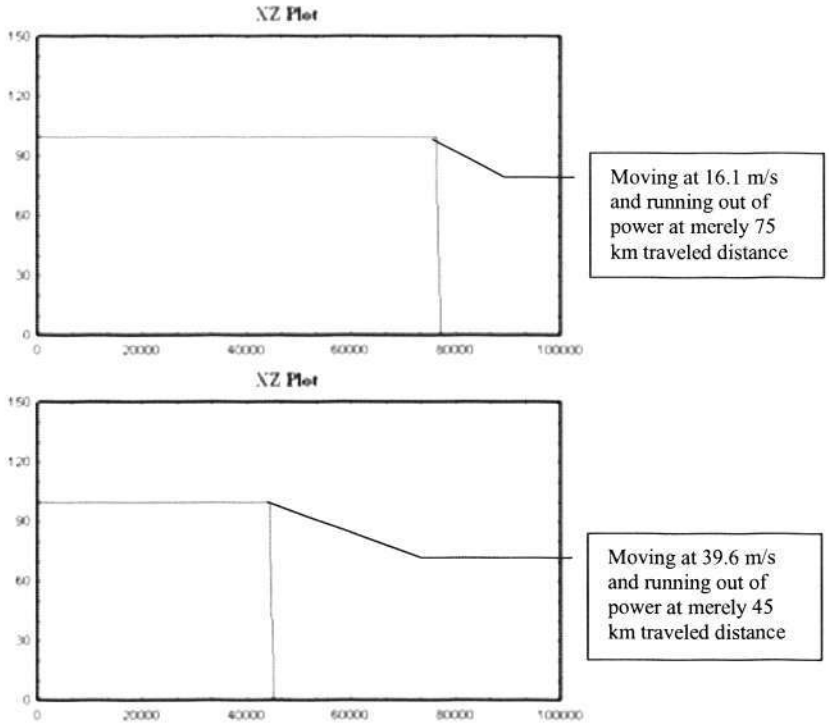

Figure 4.5: Black box test on the effect of speed V (or thrust) on power consumption: (a) V=16.1m/s, (b) V=39.6m/s

The first scenario is set at near stall speed, 16.1 m/s and the second one is at maximum speed of 39.6 m/s. It shows that flying near to stall speed will give 1.3 hour endurance and 75 km traveled distance. On the other hand, flying at maximum speed will give 0.3 hour endurance and 45 km traveled distance only. The movie demonstration of these black box tests can be seen at CD attached in this report.

4.2.3. Credibility

In this project, all the information collected to support the credibility of *RRCUAV* model came from literature resources, e.g. academic journals, papers, and commercial brochures.

The basis of *RRCUAV* simulator is the architectural models introduced by Robbins and Anderson (1998). Their architecture is suitable to the objective of this project since it is built in closed loop control system allowing operator and/or WPP to input high level instructions and let the controller interpret these instructions into low level commands, which are then followed by aerodynamic motions. This architecture is also open, flexible, and expandable for future developments, such as the study of UAV autonomous behaviors since the separation of intelligent system, controller, and physical agent opens the opportunity of providing the simulator with more detail models (see Appendix A. Simulator Development for details).

To ensure that the *RRCUAV* model remains relevant to the modern trend, numerical parameters of *RRCUAV* e.g. turning radius, stall speed, endurance, and etc. are defined within the class of mini UAV as the catalog study on different systems of *RRCUAV* has been carried on in Chapter 2. This makes the simulator capable of providing a satisfactory range of accuracy in comparison with reality, which constitutes a credible simulator (Giannasi et al, 2001).

As stated by Khazanchi (1996), a credible simulator must be open to mathematical, illustrative, and graphical characters and must have experimental testability (see again section 4.1.3. Operational Validation/Credibility).The results of the implemented models will be shown in Chapter 5. Study on Simulated Behaviors with the illustration of trajectory plot and performance graph. In Chapter 5, experimental tests on Obstacle

Avoidance are conducted. There have been three thousands of experimental tests are run but no anomalies are observed.

4.3. Chapter Summary

Robbins and Anderson (1998) architecture adopted in *RSuav* module is appropriate to the purpose of simulating autonomous behaviors since it is oriented to the high level instructions coming from WPP. The control and kinematic models are built simple yet noteworthy to simulate the motion of fixed wing aerial vehicle. This effort was discussed in Chapter 3 and becomes part of Conceptual Validation. Another method of Conceptual Validation is by using different versions of aerodynamic calculations to assess the consistency of *RRCUAV* simulated performance. The result shows that both horizontal flight, .e.g. turning at constant altitude and vertical flight, e.g. takeoff and climbing are consistent with other concept of disciplines.

On the other hand, the Verification of *RSuav* module is dominated by author's informal analysis, which was done in Chapter 3. Black Box testing is another method used to demonstrate that simulated behaviors work according to the expected behaviors of theoretical models.

To make a credible simulator, *RRCUAV* model is formulated based on the general specifications and characteristics of mini UAVs that are commercially available or used in different applications and competitions in the world (see again Chapter 2). In Chapter 5, it will be shown that the developed UAV simulator is experimental testable over three thousands of scenarios to study obstacle avoidance and no anomalies are observed during the tests.

Chapter 5

Study on Simulated Behaviors

In chapter 2 and 3, the creation of both conceptual design of *RRCUAV* and UAV simulator results in the simulated behaviors of modern mini UAVs. This chapter is presented to carry out the study of *RRCUAV* autonomous behaviors, which will then lead to the formulation of conceptual operation of *RRCUAV* in urban area.

The conceptual operation is designed based on the most appropriate tradeoff between its capability and affordability. To show how far it can satisfy the defined operational requirements, the performance of developed behaviors is presented in this chapter. However, the proposed conceptual operation covers limited areas only. Some aspects of UAV operation, such as communication system and human factor are excluded in this chapter.

The conceptual operation is based on pre-programmed and onboard flight plan. Compared to pre-programmed behaviors, obstacle avoidance is relatively new and still immature as it is yet to be commercially available and currently regarded as on-going research. At the end of this chapter, several experimental tests are conducted to find the feasibility of this technology applied to *RRCUAV* performing urban operation.

5.1. Conceptual Operation

Each operation has different objectives depending on the nature of its mission. However, the conceptual operations discussed in this chapter are not dedicated for any specific missions, such as path following operation in traffic monitoring or MRT railway inspection. Instead, the conceptual operation only includes the general mission profiles that mainly consist of three flight modes: takeoff, cruising, and landing. Go-to-

waypoint operation will be then introduced in cruising mode and it can be expanded for different applications by the users of this simulator. The conceptual operation consists of a set of autonomous behaviors, which are described as follows.

5.1.1. Automatic Take-off

False Alarm due to Ground Reflection

In most of urban areas, there is only limited space provided for the *RRCUAV* to takeoff. Although it only requires a minimum length of 37.6 m for the *RRCUAV* to lift off, there is a need to quickly reach to the minimum altitude, where obstacle avoidance sensor can work optimally. This is because at lower altitude, especially below 30 m, a false alarm due to laser reflection on the ground frequently occurs.

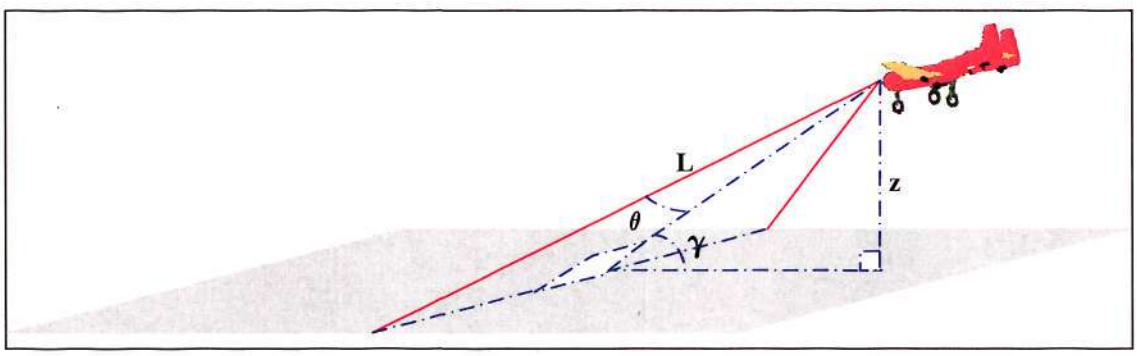

Figure 5.1: Length of laser beam touching the ground

As shown in Figure 5.1, the laser opening angle and *RRCUAV* pitch angle are defined as θ and γ respectively. Then, L is defined as the length of laser beam touching the ground and mathematically determined as:

$$L = \frac{z}{\sin|\gamma| \times \cos\theta} \qquad (5.1)$$

L is used to distinguish between obstacle and ground in order to prevent false alarm from happening. If the *RRCUAV* pitches down ($\gamma < 0$) and laser distance returns a value

smaller than L, the *RRCUAV* will conclude the existence of objects ahead. However, the altitude z in Equation 5.1 is measured from sea level yet the fact shows that terrain height is often irregular. Although such algorithm has been implemented, some parts of the ground may be perceived wrongly as obstacles. These phenomena often occur when the *RRCUAV* flies lower than 30 m.

Spiral Ascent

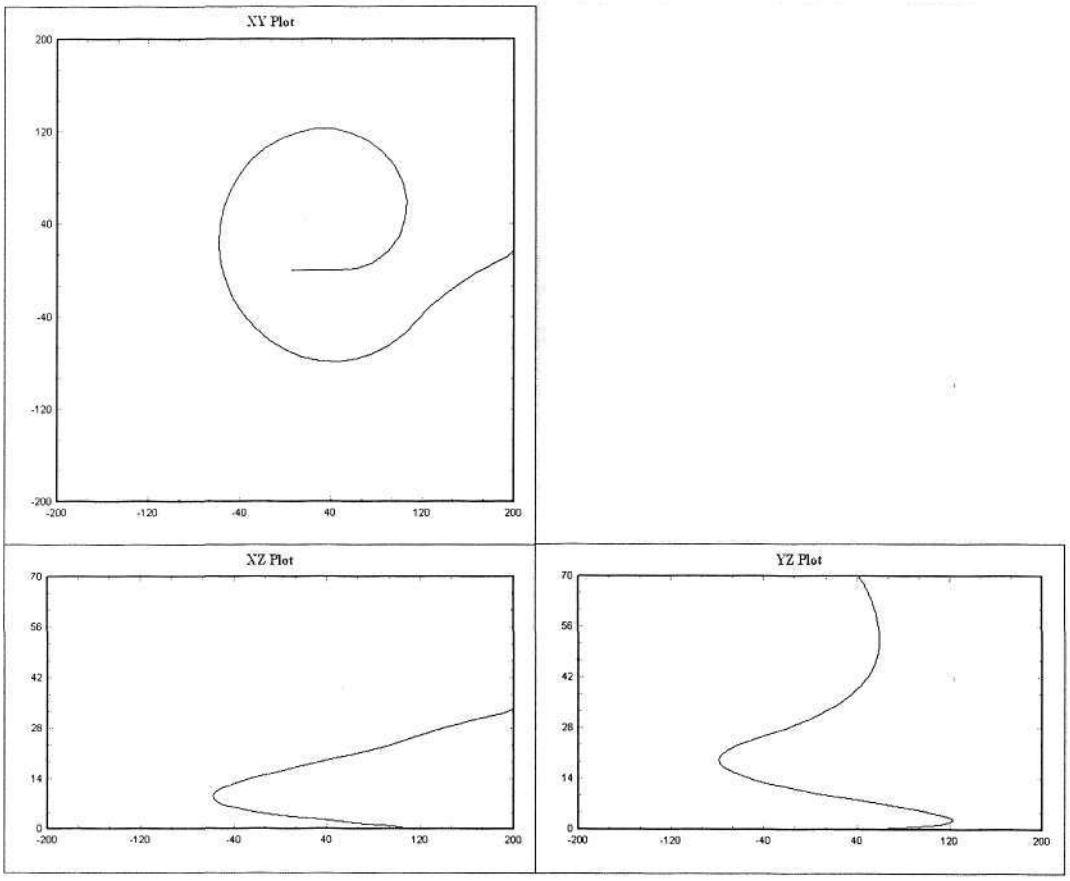

Figure 5.2: Trajectory Profile of Automatic Takeoff based on Spiral Ascent

To solve this problem, spiral ascent was designed with the trajectory profile shown in Figure 5.2. During this operation, it is important to keep clear from any object within a 130 m radius and a 30 m altitude from its initial position. Additionally, there is a need to gradually increase the bank angle during spiral ascent to prevent the wingtip from

being damaged by the ground. Therefore, *RRCUAV* progressively increase its bank angle to 53° at merely 1 m above the ground. The time taken for automatic take-off is around 35 seconds and after that, the *RRCUAV* will move towards the first waypoint.

5.1.2. Go-to-Waypoints

Accuracy and Stability

Go-to-waypoint operation is performed by Intelligent System module – Dynamic Trajectory Smoother (DTS), which generates desired heading based on the given waypoint(s), by accommodating the kinematics constraints of aircraft. To determine the quality of go-to-waypoint operation, there are two main criteria that are normally used by most of UAV testbeds, such as the one conducted by Pisanich and Morris (2001). These criteria are accuracy and stability. Accuracy determines how close UAV can reach the corresponding waypoint and on the other hand, stability is related to the ability to minimize flight disturbance and the altitude change on turning.

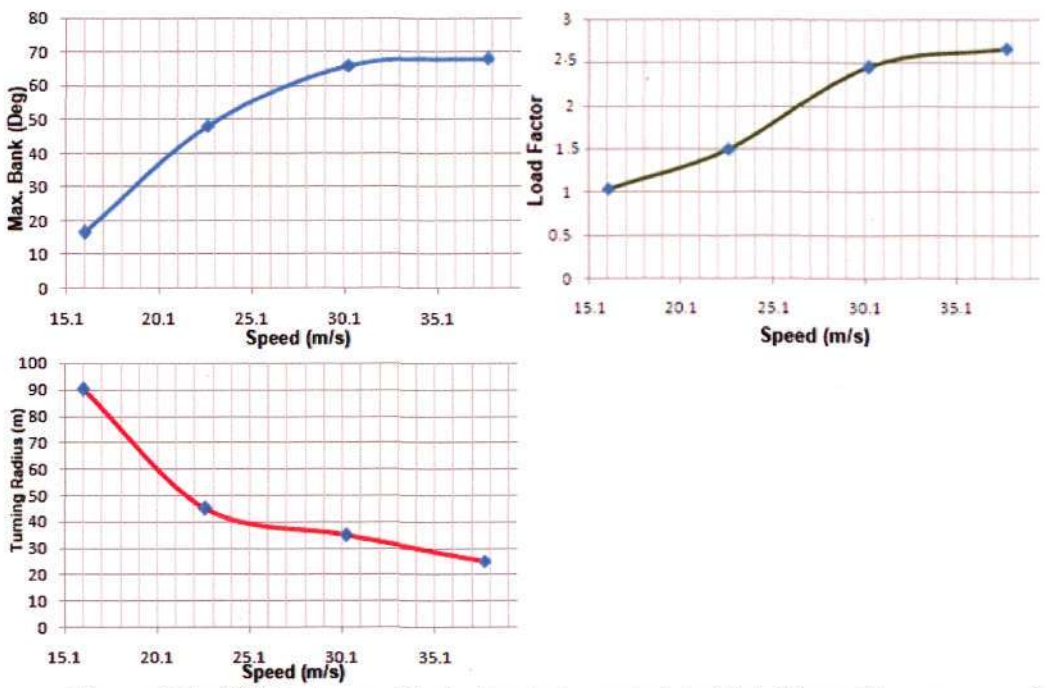

Figure 5.3: (i) Maximum Bank Angle to maintain Stability with corresponding (ii) Load Factor and (iii) Turning Radius

Since lateral and longitudinal stability control is not the interest of this project, the *RRCUAV* is assumed to have stable operation if the altitude change is within ±5 m on turning. Figure 5.3 depicts the maximum bank angle allowed (corresponding load factor and turning radius) for the *RRCUAV* to maintain stability.

Accuracy of go-to-waypoint is affected by three parameters: separation distance *d*, bank angle, and speed. Simulation results shown in Figure 5.4 is plotted with the *RRCUAV* moving with defined speed and average bank angle from the first to the second waypoint over a separation distance of *d*. The accuracy is then defined as the closest distance *RRCUAV* can go to the second waypoint.

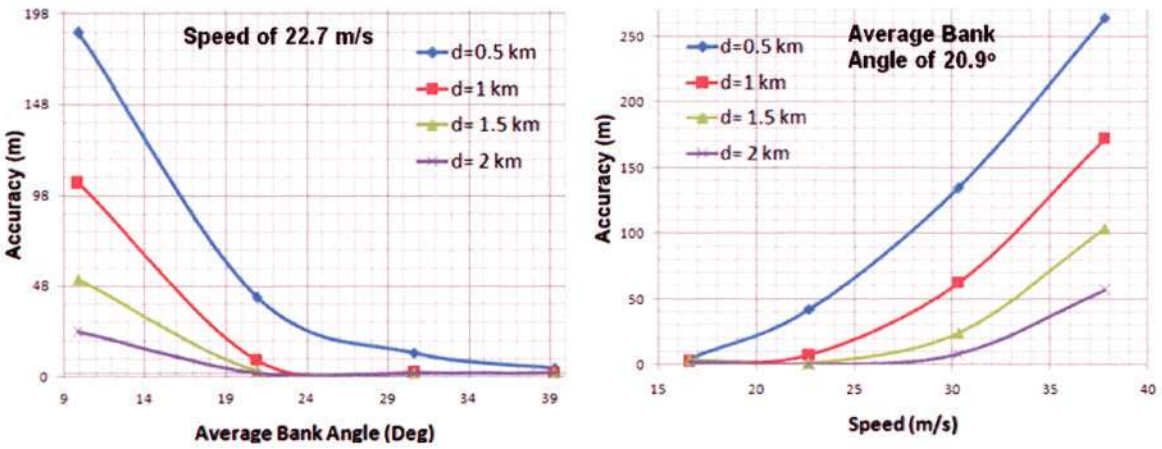

Figure 5.4: Speed, Bank Angle, and Separation Distance *d* vs. Accuracy

The graphs show that generally, higher bank angle and/or lower speed result in better accuracy. This is simply because by having this conditions, the turning radius is smaller and thus, the *RRCUAV* becomes more maneuverable. Additionally, greater distance also improves the accuracy of go-to-way points since the *RRCUAV* has more time to actuate its direction projected to the waypoint.

However, go-to-waypoint operation is only useful if the separation distance *d* is equal or more than the turning diameter. Figure 5.5 shows an example of separation distance *d* less than its turning diameter: the *RRCUAV* only encircles both waypoints:

(0, 0, 50) and (50, 0, 50) without making any significant change of minimizing the gap between *RRCUAV* and the waypoint. This is because the separation distance *d* is 50 m, while the turning diameter of *RRCUAV* is nearly 120 m.

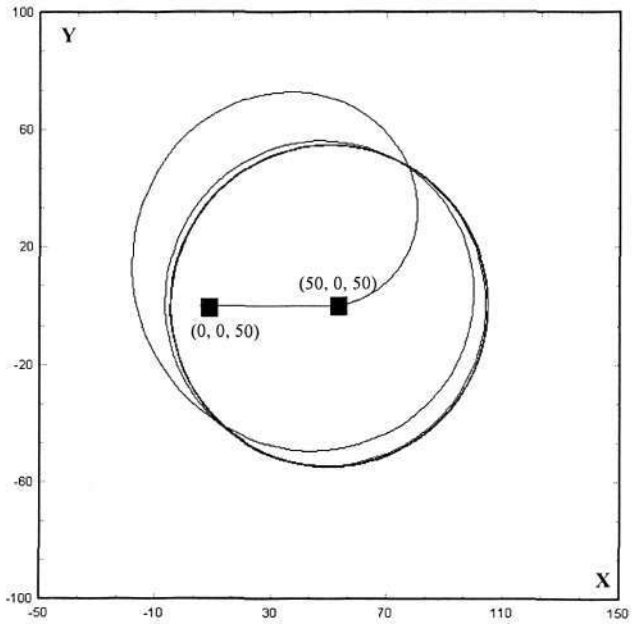

Figure 5.5: Case when separation distance *d* is less than turning diameter

If the number of defined waypoint is only one, the *RRCUAV* will encircle it but if there are more than one waypoint, it will move in a closed loop trajectory based on the defined sequence. Figure 5.6 shows the trajectory profile of the *RRCUAV* that flies at 16.1 m/s to perform go-to-waypoints operation over 8 waypoints. These waypoints are located at different altitudes with the average separation distance of around 2 km.

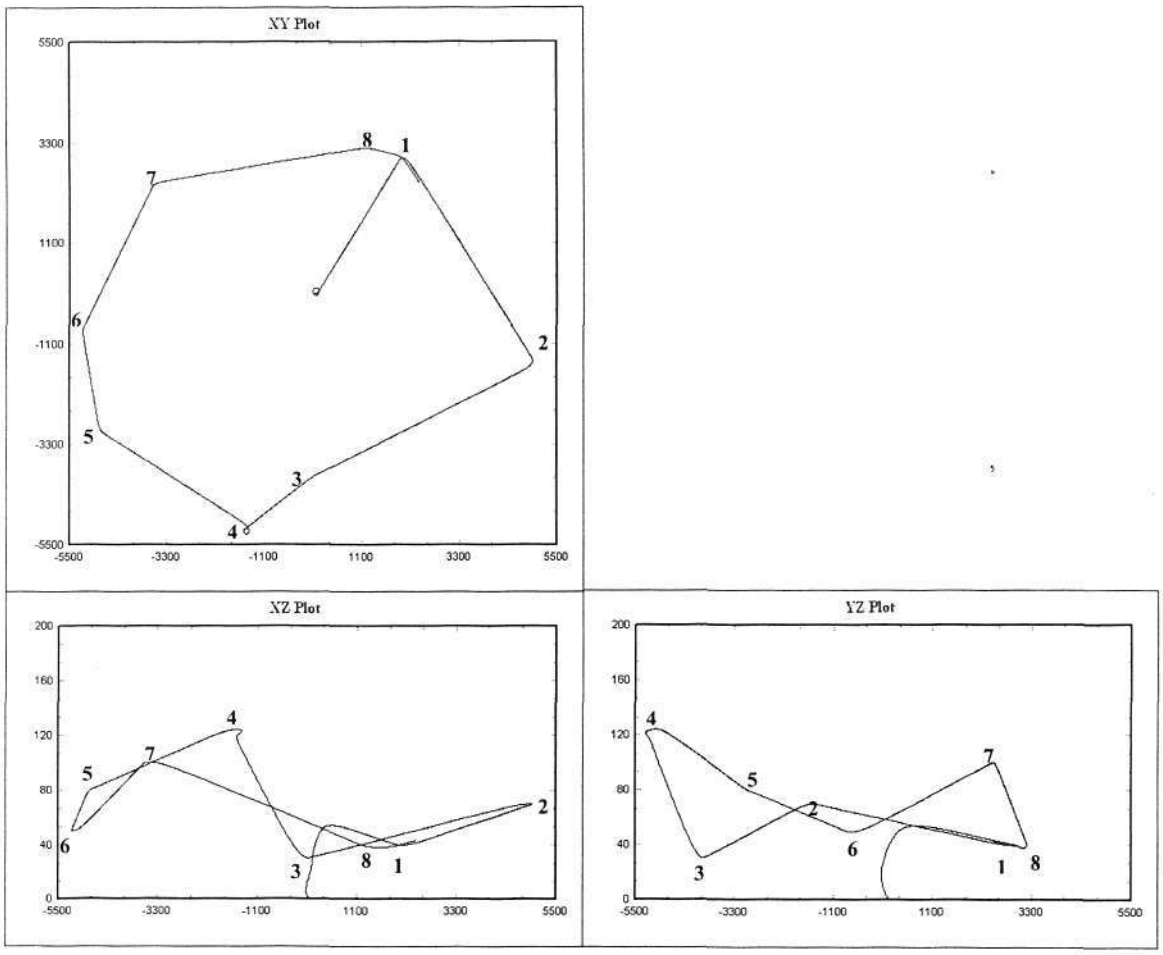

Figure 5.6: Trajectory Profile of Go-to-Waypoint over 8 defined Waypoints

SPOI Mode

As stated earlier in Chapter 2, to provide stable surveillance operation, Sensor Point of Interest (SPOI) mode is activated. This capability enables *RRCUAV* to autonomously track the ground objects, which positions have been defined in a go-to-way operation. Figure 5.7 shows the SPOI mode activated to perform surveillance over building ruins on finding any survivors or potential threats. A video clip of SPOI mode can be seen in the CD attached in this report.

Figure 5.7: Surveillance based on SPOI mode over building ruins

5.1.4. Automatic Landing

Landing Performance

Automatic Landing is not a simple task since it requires greater control capability. The quality of a landing will depend on the vertical speed on ground touching, the accuracy of landing point, and the required time. Having three phases on approaching, spiral ascent/descent, and landing designed in the automatic landing operation, 24 scenarios as illustrated in Figure 5.8 are performed to study its performance. These simulation tests were conducted with *RRCUAV* flying from different initial altitudes and incoming angles to perform auto-landing. The results are presented as follows:

- Landing Accuracy: average (x coordinate): 2.43 m, standard deviation (x coordinate): 1.82 m; average (y coordinate): 0.44 m, standard deviation (y coordinate): 1.03 m
- Vertical speed on ground touching: average: -2.0 m/s, standard deviation: 0. 00 m/s

- Time taken: average: 2.70 min, standard deviation: 0.46 min
- Landing Distance: 52.03 m (bouncing once), standard deviation: 0.07 m

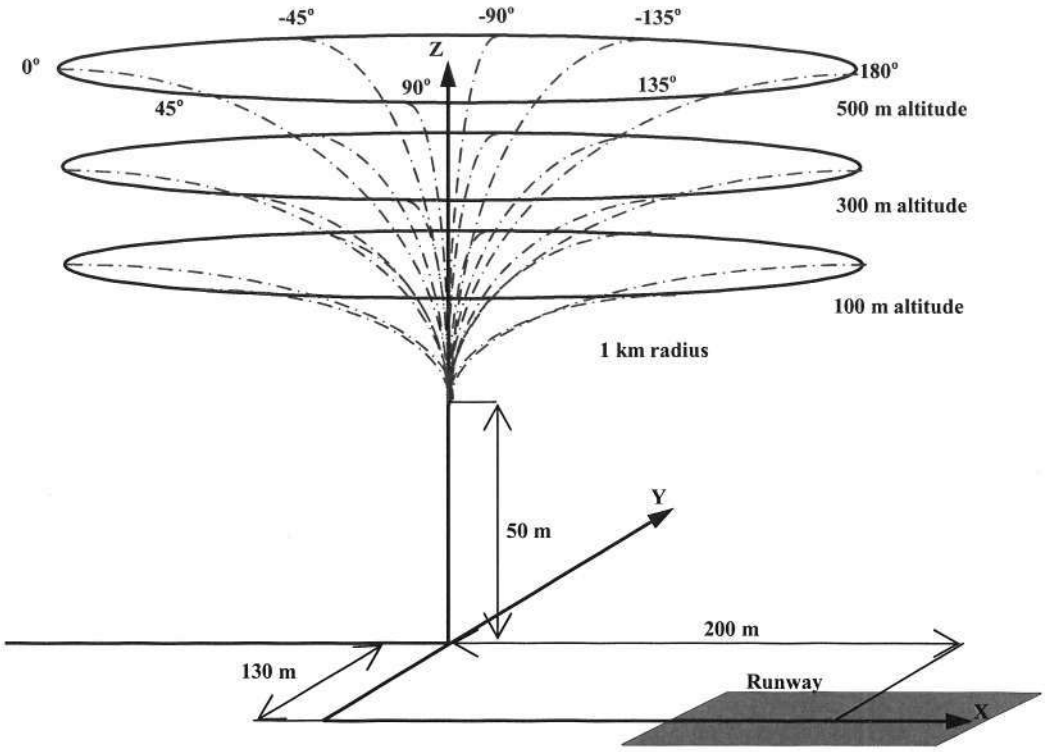

Figure 5.8: 24 Simulations Tests conducted to asses the safety, accuracy and time for landing

Detailed results of landing performance can be seen at *Appendix D. Experimental Test and Result.*

Emergency Landing

According to Civil Aviation Safety Authority Australia (2002), UAV system should incorporate a fail-safe Flight Termination System (FTS), which provides recovery to a predetermined recovery area, in the event of loss of data link, failure of navigation, and airframe damage.

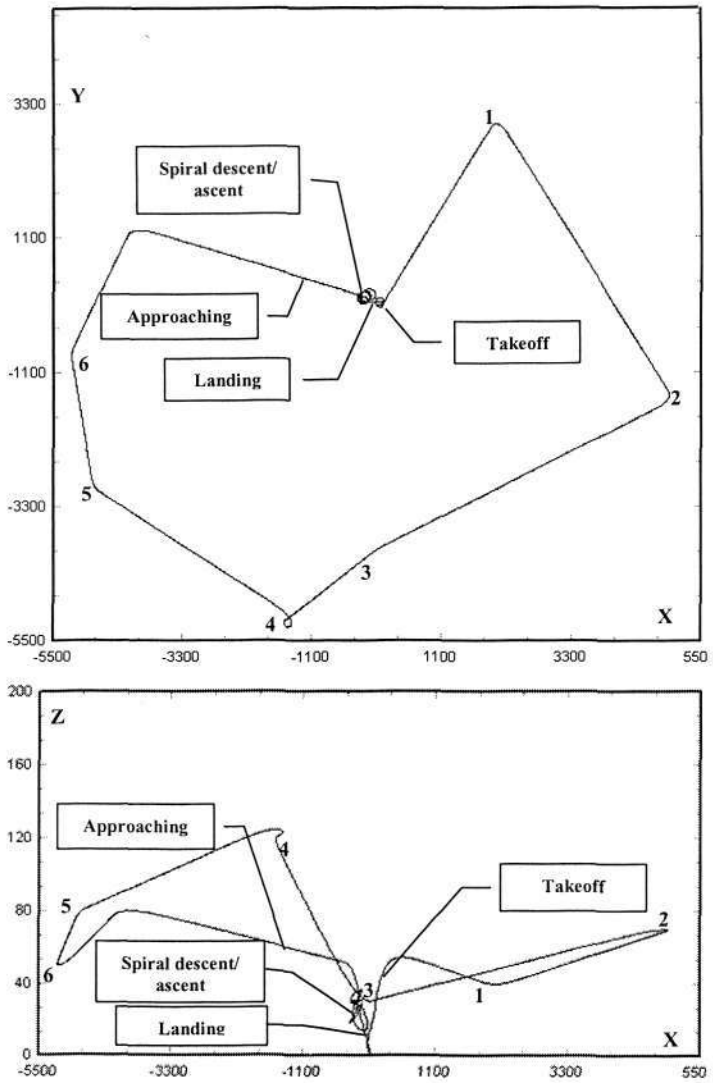

Figure 5.9: Trajectory Profile of Automatic Landing in Three Phases

Automatic landing is therefore taken as part of FTS and will be executed if any of the events mentioned earlier occur. Figure 5.9 shows the trajectory profile of *RRCUAV*, which flies at 30.3 m/s and actively uses its camera gimbals. Due to power shortage, emergency landing is then executed into three phases: approaching, spiral ascent/descent, and landing after passing waypoint 6.

Operational Requirement

The last phase of automatic landing is performed with initial altitude of 30 m at a distance of 200 m from the runway and 130 m left to the runway axis. Figure 5.10 is the

zoomed picture of the landing trajectory from different views. On touching the ground, the UAV bounces once for about 40 cm height at landing speed of 11.2 m/s, which creates slight lift on the aircraft. This makes the landing distance almost one and a half times longer than the takeoff distance. After landing, the brake is applied to slow the UAV down and it takes around 20 m for *RRCUAV* to come to a complete stop.

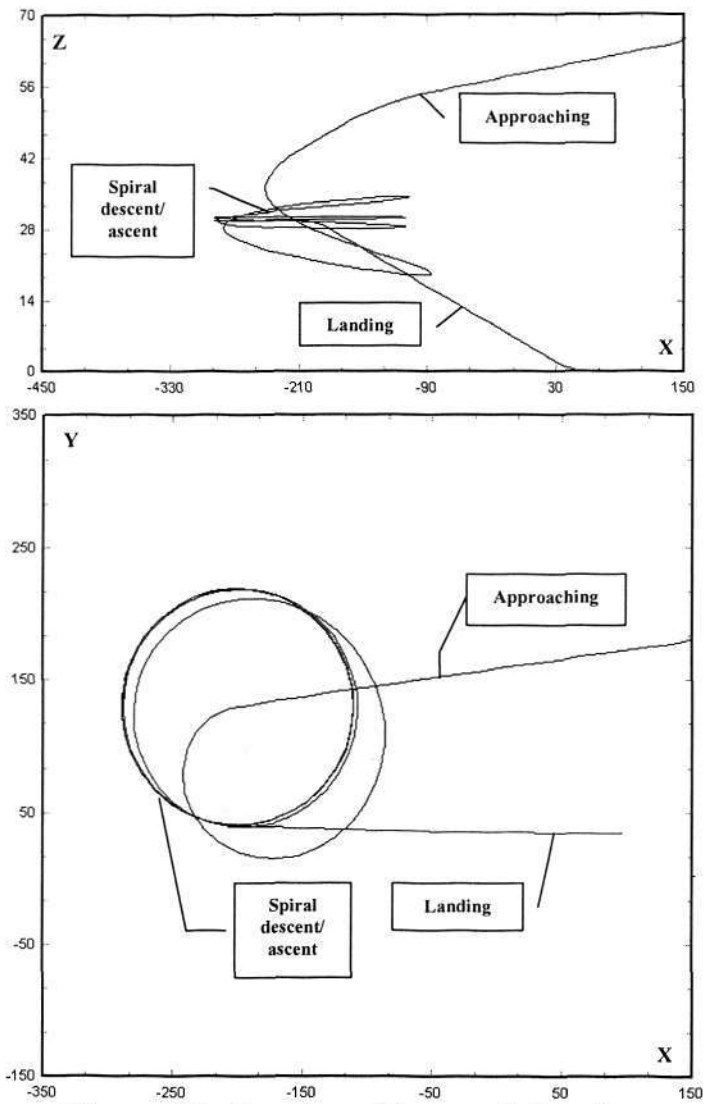

Figure 5.10: Zooming of Automatic Landing

Figure 5.10 shows that the proposed automatic landing requires at least a vacant area of 400 m^2. This is because: (i) the *RRCUAV* needs to perform spiral ascents/descents

with merely 75 m turning radius and at 30 m altitude, which is lower than the average height of residential buildings, e.g. HDB flats, and (ii) it needs another 320 m to land.

5.1.5. Obstacle Avoidance

In robotics, obstacle avoidance is the task of satisfying some control objective subject to non-intersection or non-collision position constraints. Normally, obstacle avoidance is considered to be distinct from path planning in that one is usually implemented as a reactive control law while the other involves the preprogrammed of an obstacle-free path which will then guide a robot along. Based on this definition, *RRCUAV* obstacle avoidance always refers to the use of information coming from the sensor.

Motivation

A number of testbeds on obstacle avoidance have been demonstrated by other universities and research institutions for both rotary wing UAV, e.g. Shim et al (2005), Hrabar and Sukhatme (2004), Wagter and Mulder (2005), Muratet et al (2005) and fixed wing UAV, e.g. Saunders et al (2005), Griffiths et al (reviewed), which combines both obstacle avoidance and path planning.

Most of the testbeds were dominated by the use of rotary wing UAV. This is because rotary wing UAV can hover and thus, it gives greater flexibility to move along sharp turn and to reverse as well. Additionally, hovering provides more time for the autopilot of rotary wing UAV to detect and identify the object in order to select the best escaping maneuver.

Although the maneuverability of fixed wing UAV is restricted by its turning radius, it offers some prominent advantages, such as greater speed and lesser noise than rotary

wing UAV. Compared to rotary wing UAV, e.g. *Rmax* with a maximum speed of 6 m/s, the maximum speed of fixed wing mini UAV averages around 20 m/s. These features are useful for military applications or any other applications that require immediate response, such as search and rescue operation, and ground troop escort.

Obstacle avoidance is different from any other pre-programmed behaviors. This behavior is new and still immature as it is yet to be commercially available and currently regarded as on-going research. This project therefore conducts a study on the feasibility of *RRCUAV* obstacle avoidance for urban operation.

Urban Area

As stated in section 1.4.1. Target Application and 1.6. Chapter Summary, urban operation is chosen based on the nature of Singapore environment. Hence, it is important to first identify the characteristic of Singapore environment that will affect the performance of *RRCUAV* obstacle avoidance.

According to the study on some surveyed cities by Vick et al (2002), urban area is generally formed by the composition of 70% of residential area, 25% of business district, and the remaining 5% of industrial zone. In both residential area and business district, buildings are high enough to affect the operational flight of *RRCUAV*. Table 5.1 illustrates the characteristics of a typical Singapore residential area (HDB flats), where the separation distances are generally narrower than building lengths and widths. It also shows that the lengths of residential buildings are generally about three to five times longer than its width.

Table 5.1: Characteristics of Some Residential Areas in Singapore

Residential Area	Avg. Build Length (m)	Avg. Build Width (m)	Avg. Build. Sep. Dist. (m)	Avg. Dist. btw Precincts (m)
Outram	80	23	14	37
Queenstown	65	19	19	28
Clementi	85	22	28	51

Based on observations on some residential areas in Singapore, residential buildings are usually parallel or perpendicular patterned to other buildings. The orientation of buildings normally depends on the landscape and/or the orientation of nearest roads. As shown in Figure 5.11, residential buildings are divided into different precincts (illustrated by dashed white lines) by main roads. The distance between precincts is almost two times longer than the building separation distance within a precinct.

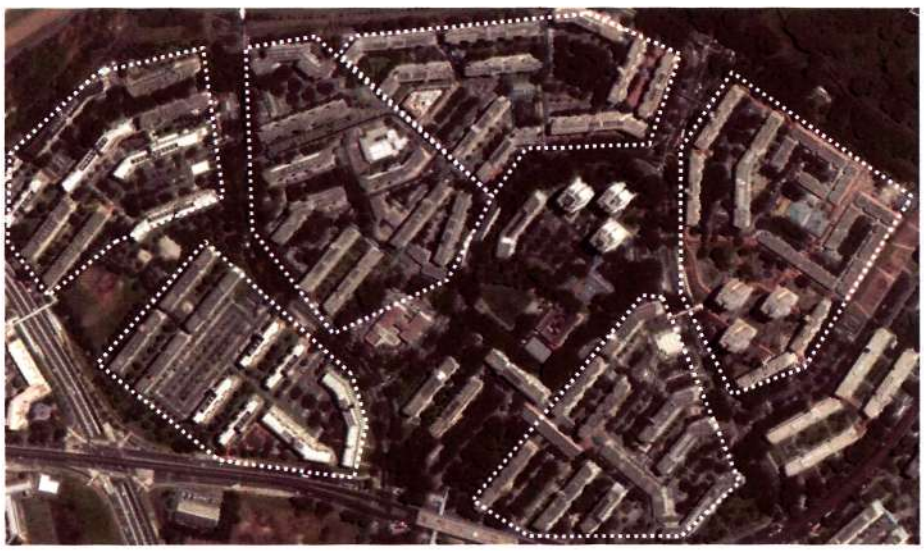

Figure 5.11: Top View of HDB Flats (Google Earth)

Distinct from residential area, the separation distances in business district are generally wider than building lengths and widths. Table 5.2 shows the characteristics of one of a typical business district in Singapore. In this area, the separation distances nearly equal to building lengths and widths. It is dominated by skyscrapers with heights ranging from 100 m to 200 m.

Table 5.2: Characteristics of a Residential Area in Singapore

Business District	Avg. Build Length	Avg. Build Width	Avg. Build. Sep. Dist.
Raffles	40	35	35

In general, the buildings in business district are less cluttered and more randomly orientated as shown in Figure 5.12. They are not uniform in design and the height varies

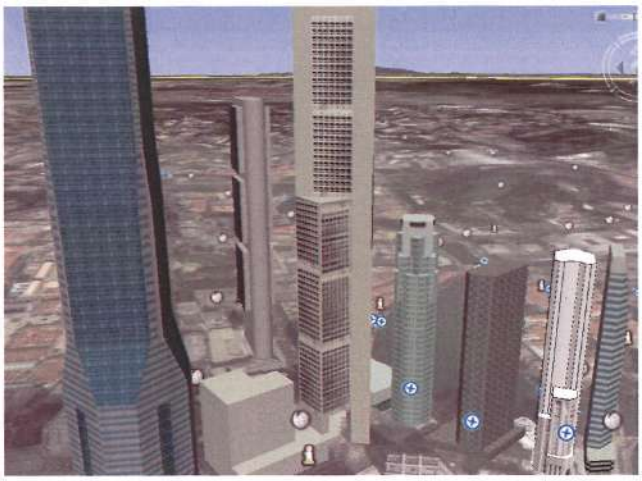

Figure 5.12: Business District in Raffles (Google Earth)

Experimental Test

There are two experimental tests on *RRCUAV* Obstacle Avoidance. They are as follows.

- Experimental Tests I. Finding the Significance of Parameters

At the beginning of this section, the prediction is made by listing the potential parameters that may likely affect the performance of obstacle avoidance. These parameters are tested and then analyzed by software named REG that adapts a classification method based on decision tree to find the significance of each parameter.

a. Tuning Parameters

The first experiments are conducted by having *RRCUAV* treated under different scenarios. In each scenario, the parameters, which are believed to affect the performance of *RRCUAV* obstacle avoidance, are systematically varied. The variation is made within reasonable range, which is believed to give more meaningful result. Those parameters include:

1) Environment

Environment is predicted to be the most significant factor determining the performance of obstacle avoidance. This is due to the fact that there is a sharp difference between residential area and business district in degree of constraint (see again Table 5.1. and Table 5.2). Such difference is most likely to give the most significant effect on the performance of *RRCUAV* in obstacle avoidance.

Both business district and residential area are set up for the first experiment in circular shape with radius of 1.6 km (see section D2.1. Experimental Tests I. Finding the Significance of Parameters for details). Building width and separation distance are then defined as the environmental parameters in these experimental tests.

2) Maneuverability

Maneuverability is more likely to be the second most significant parameter. The safety of the *RRCUAV* flying over buildings will depend on how tight the turning radius is. The parameter of maneuverability includes aircraft speed and bank angle. Speed will be divided into three schemes as follows. The first is when the *RRCUAV* is flying near at stall speed to minimize power consumption. The second is the aircraft speed, which is set to be moderately high, around 20 m/s and the last is when the *RRCUAV* flies at a

high speed of around 30 m/s to create the tightest turning radius among three schemes. Higher speed will result in higher maneuverability since the it is followed by higher bank angle. In each scheme, bank angle will be set according to Figure 5.3, which ensures *RRCUAV* stability.

3) Sensor Configuration

Based on Figure 2.13, obstacles can be divided into frontal and lateral obstacles depending on the *RRCUAV* flight direction. Frontal obstacles contribute to the potential damage of the *RRCUAV* nose or fuselage, while lateral ones might damage the *RRCUAV* wing. It seems that sensor configuration affects the performance of obstacle avoidance over frontal and lateral obstacles. The bigger the laser opening angle θ from aircraft nose, the more likely the *RRCUAV* will be responsive to the presence of lateral obstacles and vice versa. In view to this, experimental tests are conducted based on sensor configuration as well, which laser opening angle θ is varied into: i) nearly parallel, ii) 45^{o}, and iii) nearly perpendicular to the body axis.

4) Sensor Threshold

Sensor threshold is defined as the limit of the measured distance that *RRCUAV* activates its obstacle avoidance routine. The maximum range of the laser rangefinder is 365.8 m but it is not necessary that obstacle avoidance is executed at this distance if any obstacle is detected. Threshold with lower value is useful to minimize the traveled distance of obstacle avoidance but if it is too low, it might be too late for *RRCUAV* to avoid collision. It can not be too big either since it will cause the *RRCUAV* to prematurely react when it is far away from the obstacle.

There is a lack of knowledge on how to find optimum value of threshold. It seems that threshold must not be far from the average of building separation distance of particular area. That is why, threshold will be set into five values, which are: bigger than, smaller than, and equal to average building separation distance in the first experiment.

5) Sensor loop rate

Sensor loop rate constitutes the updated rate of information on measured distance. It measures the sensitivity of obstacle avoidance. Intuitively, higher sensor loop rate will give better performance of obstacle avoidance. The rate to be tested is divided into: 3 Hz, 6 Hz, and 10 Hz, which is the maximum sensor loop rate.

b. Experimental Procedures

The first experiment is conducted by varying each parameter alternatively. In every combination of five parameters, *RRCUAV* flies in eight different trajectories. There are total of 2160 scenarios conducted and the details of experimental procedures and its results can be seen at Appendix D. Experimental Test and Result.

c. Experimental Result and Interpretation

To assess the significance of each parameter on obstacle avoidance performance, the Decision Tree (DT) method, which is applied in software named DTREG, is used. DTREG analyzes the data and generates a model showing how best to predict the values of *target variable* based on the values of *predictor variables*. Details on how to apply experimental data on DTREG can be read from section D.2.3. Applying Experimental Data on DTREG Software. Five parameters described earlier are categorized as predictor variables. On the other hand, traveled distance becomes the target variable. It

is defined as the total distance of *RRCUAV* passing through the buildings and indicates the effectiveness of obstacle avoidance.

DT is a predictive model that classifies the value of target variable based on given predictive variables. It is appropriate for the evaluation of the result of first experiment since:

i) DT provides a clear, logical representation of the data model and can be understood and used by people who are not mathematically inclined

ii) DT can handle both continuous and categorical variables. This is useful since some continuous variables, e.g. sensor threshold and categorical variable, e.g. environment type are used in the first experiment.

iii) DT identifies important variables. By examining which variables are used to split nodes near the top of the tree, the most important variables can be quickly determined.

Since traveled distance is continuous, a regression model is therefore generated. The rectangular boxes shown in the tree in Figure 5.13 are called *nodes*. Each node represents a set of records (rows) from the original dataset.

To create node, a split is required. It is principally done by choosing the best predictor variable that favors the homogeneity *within* each child node and the heterogeneity *between* the child nodes (Sherrod, 2003). The heterogeneity – or dispersion – of target categories *within* a node is called the node impurity. The goal of splitting is to produce child nodes with minimum impurity. In regression trees, the variance splitting method is always used. It causes DTREG to use the split that minimizes the sum of variance, e.g. sum of squared errors in the child nodes.

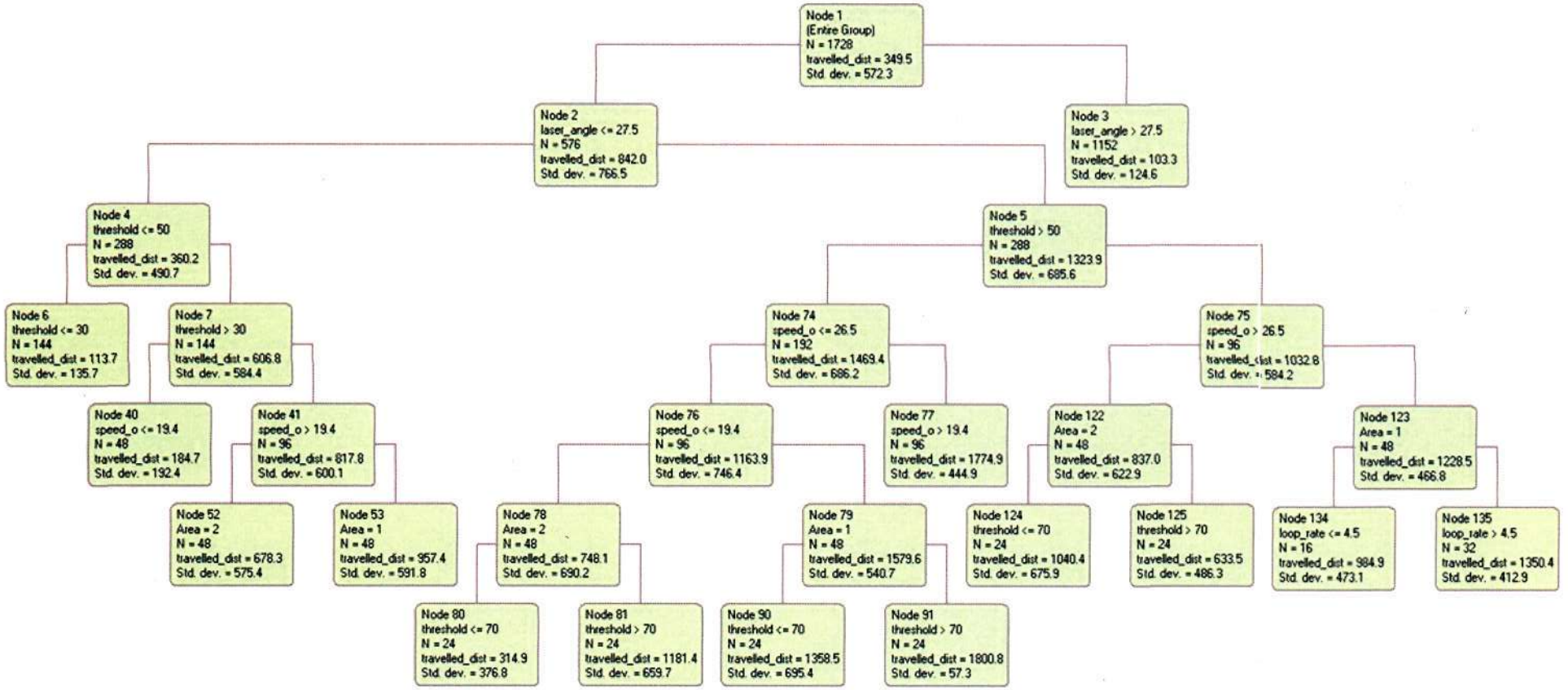

Figure 5.13: Decision Tree of the Result of First Experiment

Based on the result of 2160 scenarios, DTREG then generates a DT shown in Figure 5.13. Generally, predictor variable used in the earlier split gives greater significance to the target variable. In the first two splits, predictor variables of 27.5° laser angle of and 50 m threshold presents the best separation of traveled distance. It can be seen clearly that at (i) node 2, where the laser range finder is less than 27.5°, and at (ii) node 5, where the laser rangefinder is less than 27.5° and threshold is more than 50 m, the average traveled distance is between 800 and 1300 m. On the other hand, the negation of these conditions will result in the average traveled distance of between 100 m and 300 m only (see node 3 and 4). Large difference between the average traveled distances of both child nodes indicates the heterogeneity.

The homogeneity, on the other hand, is inversely related to standard deviation of target variable within one node. Bigger standard deviation constitutes bigger node impurity and it will more likely require split to reduce it.

However, DT has some limitations. DT created from numeric data sets can be quite complex since attribute splits for numeric data are binary. Additionally, DT algorithm is unstable. Slight variations in the training data can result it different attribute selections at each choice point within the tree. The effect can be significant since attribute choices affect all descendent sub trees.

Therefore, to minimize these disadvantages, DTREG has automatic pruning capability using V-fold cross-validation to determine the optimal tree size. Furthermore, instead of relying only on the resulted decision tree to know the significance of each predictor variable on target variable, DTREG calculates the importance of each variable by adding up the improvement in classification gained by each split using the predictor. The improvement is calculated as the decrement of misclassification cost (in this case,

sum of variance) before and after the split. The importance score for the most important predictor is scaled to a value of 100.00. Other predictors will have lower scores. The result is shown in Figure 5.14.

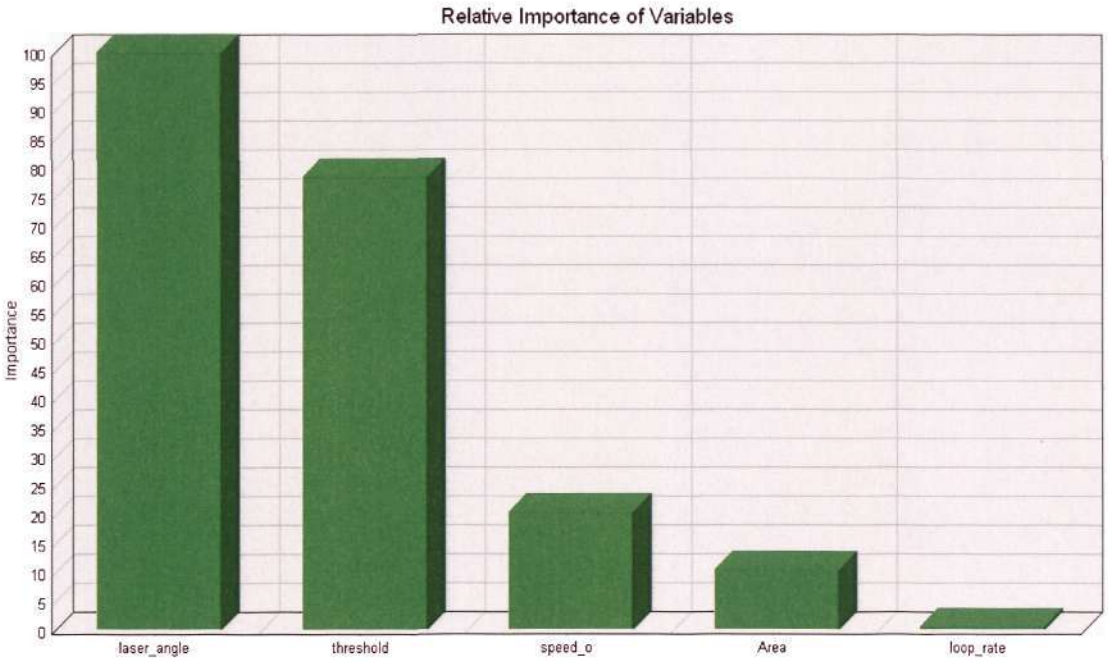

Figure 5.14: Variable Importance of *RRCUAV* Obstacle Avoidance

Surprisingly, sensor configuration, which is represented as laser angle, becomes the most significant factor determining the effectiveness of *RRCUAV* obstacle avoidance. In nearly all of the scenarios that *RRCUAV* survives when a 10^{o} laser angle, which is nearly parallel to body axis, is used. This implies that the effect of frontal obstacles is much more significant than lateral ones. This also explains why the second most significant parameter is the sensor threshold. Based on the experimental results, the robustness of the *RRCUAV* with laser angle set at 10^{o} drastically drops at a 20 m threshold. This is because *RRCUAV* has turning radius bigger than 20 m to avoid frontal obstacles.

Secondary factors that affect *RRCUAV* obstacle avoidance are maneuverability and environment. Although the differences of steady state turning radius among three

maneuver schemes are up to 50 m (see Section D2.1. Experimental Tests I. Finding the Significance of Parameters), it does not give the effect as much as laser angle and threshold. This is most likely because the maneuver of obstacle avoidance is executed in transition period, which adjusts the bank angle from zero to its steady state value. The effect of environmental constraint can be seen on the ratio of robustness of *RRCUAV* performed between business district and residential area. It is about 2.4:1. However, this effect is not as big as the ones caused by varying laser angle and sensor threshold.

❑ Experimental Tests II. Determining the Robustness

After knowing the importance of each parameter, the next question will be "Which scenario will give the most optimum robustness?" In the first experiment, among 2160 scenarios to be tested, the percentage of survived *RRCUAVs* is only 10.8%. The crash happens more frequently if the *RRCUAV*'s traveled distance is less than 1.63 km. This value nearly equals to the area diameter, which is set up for the first test.

a. Potential Scenarios

Table 5.3 lists the scenarios that survive during the first experiment. Based on the robustness and the diversity on speed and threshold, there are three scenarios in Table 5.3 selected for second experiment. They are as follows.

- Laser Angle=10°, Threshold=80m, V=22.7m/s, Loop Rate=6Hz
- Laser Angle=10°, Threshold=100m, V=16.1 m/s, Loop Rate=6Hz
- Laser Angle=10°, Threshold=60m, V=30.3 m/s, Loop Rate=6Hz

Table 5.3: Scenario with survived UAV

Laser Angle	Threshold	Maneuverability	Survived
10°	40 m	22.7 m/s	6
		30.3 m/s	11
	60 m	16.1 m/s	14
		22.7 m/s	30
		30.3 m/s	18
	80 m	16.1 m/s	28
		22.7 m/s	42
		30.3 m/s	9
	100 m	16.1 m/s	33
		22.7 m/s	38
		30.3 m/s	4
SUM			233

b. Experimental Procedures

Each scenario will be tested in both business district and residential areas. There are total of 30 areas, which diameters are varied into 1600 m, 1000 m, and 500 m. In each area, the *RRCUAV* flies in 20 different trajectories and thus, total of 1800 tests will be conducted in second experiment. Details of procedural test and its experimental result can be seen at section D2.2. Experimental Test II. Determining the Robustness.

c. Result and Interpretation

Table 5.4 shows the result of the second experiment. There are 9 rows that represent different combinations of aircraft properties and area diameters. In each row, 200 tests are conducted. 100 tests are applied over 5 different business districts and the remaining 100 are conducted over 5 different residential areas. *Total* robustness is the percentage of number of survived *RRCUAV* among the 200 tests. It can be divided into *Business* and *Resident* robustness, which respects to the type of area.

Table 5.4: Result of Second Experiment

Area Diameter	Aircraft Properties			Robustness		
	Laser Angle	Threshold	Maneuverability	Total	Business	Resident
1600 m	10°	80 m	22.7 m/s	63.9%	81.0%	45.1%
		100 m	16.1 m/s	64.0%	97.0%	29.9%
		60 m	30.3 m/s	47.5%	83.0%	12.0%
1000 m	10°	80 m	22.7 m/s	84.5%	97.0%	69.1%
		100 m	16.1 m/s	76.0%	98.0%	52.2%
		60 m	30.3 m/s	60.0%	86.0%	34.0%
500 m	10°	80 m	22.7 m/s	88.0%	98.0%	76.5%
		100 m	16.1 m/s	80.1%	99.0%	59.1%
		60 m	30.3 m/s	80.4%	96.0%	63.8%

The result shows that generally, the performance of *RRCUAV* Obstacle Avoidance for urban operation is not satisfactory. The results conducted in residential area are even worse – All the *Total* robustness rates are less than 80% and they encounter monotonically decrement at higher area diameter.

RRCUAV Obstacle Avoidance performs better in business district compared to residential area. If the RRCUAV deployed in business district is set at 16.1 m/s, 10° laser angle, and 100 m threshold, the robustness is nearer to 100%. The gap of robustness between business district and residential area gets bigger for higher area diameter. The *RRCUAV* set to 22.7 m/s, 10° laser angle, and 80 m threshold gives the lowest discrepancy.

When tests are conducted in residential area, sharp turns are frequently required to avoid crash. Therefore, the *RRCUAV* with higher speed and higher bank angle gives significant effect on robustness. This can be seen from the comparison of robustness between 16.1 m/s and 22.7 m/s. However, at 30.3 m/s, oscillations occur very

frequently, especially at higher threshold. It causes the *RRCUAV* to lose its altitude and thus, crash.

In conclusion, the Obstacle Avoidance designed for *RRCUAV* might give a chance of survival for urban operation if it is deployed only in business district with 16.1 m/s, 10^{o} laser angle, and 100 m threshold. However, the result shows that it does not guarantee 100% robustness. Therefore, the use of *RRCUAV* obstacle avoidance for civilian applications with routine might not be suitable. To compensate for this limitation, obstacle avoidance needs to be supported by intelligent WPP that can enhance robustness. This can be done by selecting more spacious areas, which fit to *RRCUAV* operational constraints.

Since the results shows that the robustness over residential areas are below 80%, residential area is proposed to be a restricted area for *RRCUAV* to fly. Therefore, the *RRCUAV* is recommended to fly above 60 m, which is the average height of residential building, over this area.

5.2. Chapter Summary

Conceptual Operation of *RRCUAV* is formulated into three modes: takeoff, cruise, and landing. It consists of a set of autonomous behaviors, which are developed to fulfill operational requirements in urban area. This includes: (i) The creation of stable flight with sufficient maneuverability so the *RRCUAV* can safely and accurately takeoff and land on urban area that offers limited vacant space, (ii) Spiral Ascent, which becomes part of automatic takeoff to eliminate false alarm on ground reflection,(iii) SPOI mode, which provides stable surveillance, and (iv) Emergency landing, which is executed in the event of loss of data link, navigation error, aircraft damage, and power shortage.

Based on the study of obstacle avoidance deployed for *RRCUAV*, sensor configuration and threshold play an important role on the effectiveness of obstacle avoidance. It is proven that by setting the laser angle nearly parallel to body axis, e.g. 10^{o} laser angle and defining threshold bigger than operational turning radius, increase the effectiveness of obstacle avoidance. The Obstacle Avoidance designed for *RRCUAV* might give a chance of survival for urban operation if it is deployed only in business district with 16.1 m/s, 10^{o} laser angle and 100 m threshold. The robustness can be increased by deploying intelligent WPPs that select more feasible areas for *RRCUAV* to perform obstacle avoidance.

Chapter 6

Conclusion and Future Works

6.1. Conclusion

Creating a simulation platform to study the autonomous behaviors of mini UAV deployed in urban area has been the objective of this project. This study is important as the demands for small platform of autonomous UAV has been increasing for both military and civilian applications. This research work contributes towards the knowledge of mini UAV operation and its performance in urban area.

As illustrated in Figure 6.1, the study is conducted by defining the operational requirements and then, improving the conceptual design and operation of *RRCUAV* based on the evaluation of operational requirements over a number of simulation tests.

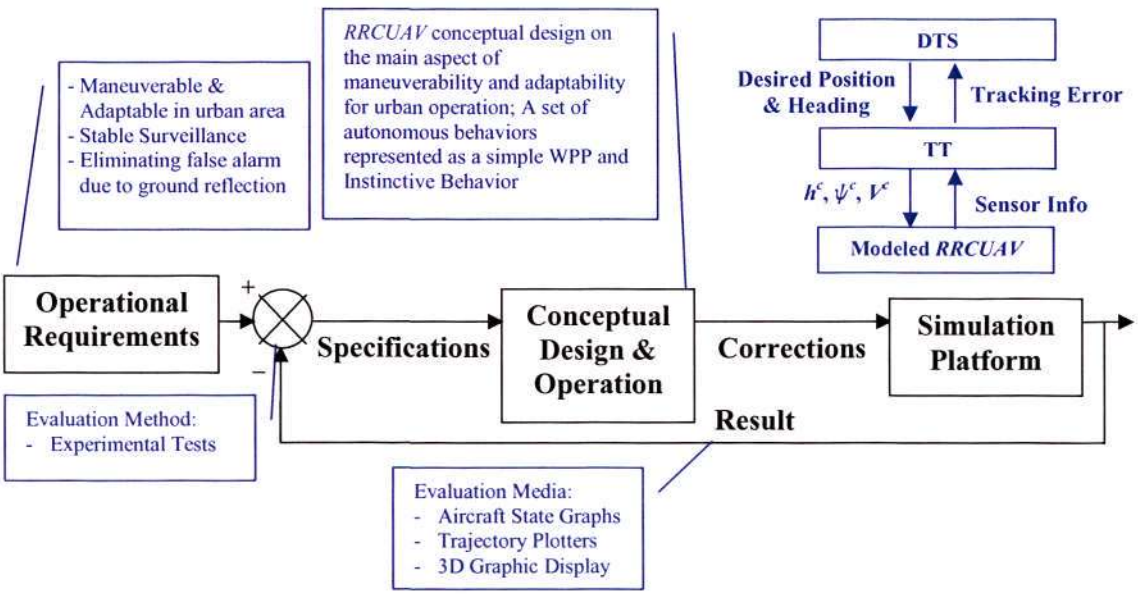

Figure 6.1: UAV Autonomous Behavior Study Approach

To achieve this objective, the conceptual design of *RRCUAV* needs first to be formulated. The engineering specifications and characteristics of commercial mini

UAVs in the world were used as the guideline of designing the *RRCUAV* as a model to be used in the project.

The main aspects of the *RRCUAV* conceptual design are the maneuverability and adaptability for urban operation. Turning radius therefore becomes the major concern as the *RRCUAV* is expected to fly among tall buildings. In addition, the deployment of flaps and the selection of appropriate engine result in better takeoff and landing performance in urban area, which are relatively limited in space.

The major challenges lie with the design tradeoff as the operational constraints on weight, size, and available power are tight. To reach an ALFUS-AFRL autonomous level for onboard route re-plan, the *RRCUAV* has to gain 9 kg and, this is relatively heavier than other mini UAVs.

Secondly, UAV simulator was then established from the combination between: (i) *ROBOSIM*, which is used as the simulator platform to periodically update the states of entity, output of continuous dynamic simulation, specialize in 3D graphical drawing, and create virtual environment, and (ii) *RSuav* module, which is an extension of robot entity class – *Rsentity* in *ROBOSIM*, and formulated based on Robbins and Anderson (1998) architecture.

RSuav module consists of intelligent system, control system, and actuator. The intelligent System acts as a Dynamic Trajectory Smoother (DTS), which receives a list of waypoints from Waypoint Path Planner (WPP) planning for mini UAV operation. On the other hand, Control System, which is a high level controller, functions as a Trajectory Tracker (TT) receiving desired heading from Intelligent System module and generating commands to Actuator, which models the kinematic of mini UAV as a mass point with simplified equation of moments.

To validate the conceptual models of *RSuav* module, different versions of aerodynamic calculations are used to assess the consistency of *RRCUAV* simulated performance. The result shows that both horizontal flight, .e.g. turning at constant altitude and vertical flight, e.g. takeoff and climbing are consistent with other concept of disciplines. On the other hand, the Verification of *RSuav* module is dominated by author's informal analysis, which mentally exercises the models. Black Box testing is another method used to demonstrate that simulated behaviors work according to the expected behaviors of theoretical models. Lastly, it was shown that the developed UAV simulator has been experimental testable over three thousands of scenarios to study obstacle avoidance and no anomalies are observed during the tests.

Several simulation tests of the conceptual operation are conducted by systematic manipulation of variables in an experiment. With the help of visualization tools, e.g. Internal State Graph, Trajectory Plotter, 3D Graphics Display, the evaluation of the tests is able to identify new operational requirements, for example i) a stable flight with sufficient maneuverability over urban area, ii) an ability to takeoff and land in urban area, and iii) stable surveillance operation.

The conceptual operation was iteratively improved based on these new generated requirements. This might be followed by re-designing the mini UAV to accommodate the requirements. In this project, the autonomous behaviors that have been developed includes: i) Automatic Takeoff with spiral descent that eliminates false alarm on ground reflection, ii) Go-to-waypoint operation, which is ranged from the lowest power consumption to the most maneuverable, iii) SPOI mode, which automatically tracks defined ground object to compensate for the limitation of turning radius iv) Automatic Landing with reasonable landing space, accuracy, and time, v) Emergency Landing,

which is a different version of automatic landing executed due to power shortage, and vi) Obstacle Avoidance.

Lastly, study of *RRCUAV* obstacle avoidance was performed. It was found that sensor configuration and threshold play an important role on the effectiveness of obstacle avoidance. It was proven that by setting the laser angle nearly parallel to body axis, e.g. 10^{o} laser angle and defining threshold bigger than operational turning radius, increase the effectiveness of obstacle avoidance. The Obstacle Avoidance designed for *RRCUAV* might give a chance of survival for urban operation if it is deployed only in business district with 16.1 m/s, 10^{o} laser angle and 100 m threshold.

6.2. Future Works

There are mainly two potential works that can be done on extending this project. The first one is to build more detail models of *RRCUAV* for the purpose of physical UAV implementation and the second one is to develop more intelligent behaviors for the optimization of UAV operations. Each of them is described as follows.

6.2.1. Flight Vehicle System Identification

Some inaccuracies of present mass point model are observed in several scenarios conducted in chapter 5. For example, flare maneuver (as illustrated in Figure 6.2), which is generally part of landing procedures, can not be executed in mass point model since the flight direction of modeled UAV is always in-line with body axis. Incapability of mass point model to perform flare maneuver causes the *RRCUAV* fail to reduce forward speed as well as descent rate in the simulation. This makes *RRCUAV* to bounce for 40 cm height and to spend another 30 m longer in the air before it is able to touch down.

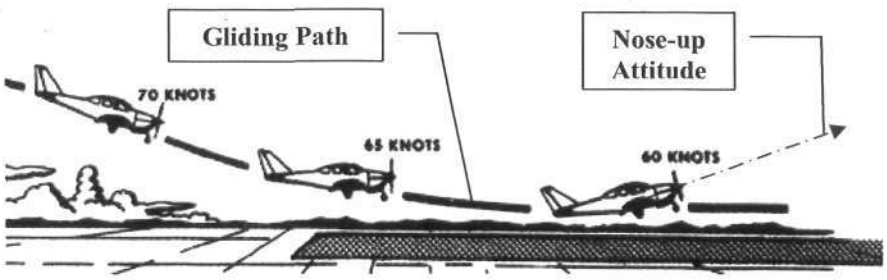

Figure 6.2: Flare Maneuver (Jonathan M. Stern)

To address such issue, Flight Vehicle System Identification needs to be done in the future. According to Jategaonkar (2006), System Identification is defined as the process of determining an adequate mathematical model with unknown parameters. He said that the process includes not only model postulating and determining parameters (which has been done in this project), but also performing suitable experiments, and gathering system inputs and responses.

Therefore, Flight Vehicle System Identification requires an experimented physical UAV to be used as a model reference. The process of identification is conducted by: (i) The creation of an accurate physical, inertial, and aerodynamic model based on tests conducted to determine several aircraft properties as illustrated in Figure 6.3; (ii) Estimation of basic stability and control derivatives by an Aircraft Analytical Software to give an initial look at the stability performance of an aircraft design; (iii)Development of a 6 Degree-of-Freedom simulation to establish the equation of moment, force, and motion of UAV. It can be incorporated with Hardware- in-the-Loop simulation to test new control algorithms with the UAV model before applying it to the real UAV; (iv) Lastly, Wind tunnel or Open Flight test is performed to validate the models that have been developed, for example to validate the stability and control derivatives determined from an Aircraft Analytical Software (Platanitis and Shkarayev, 2005) .

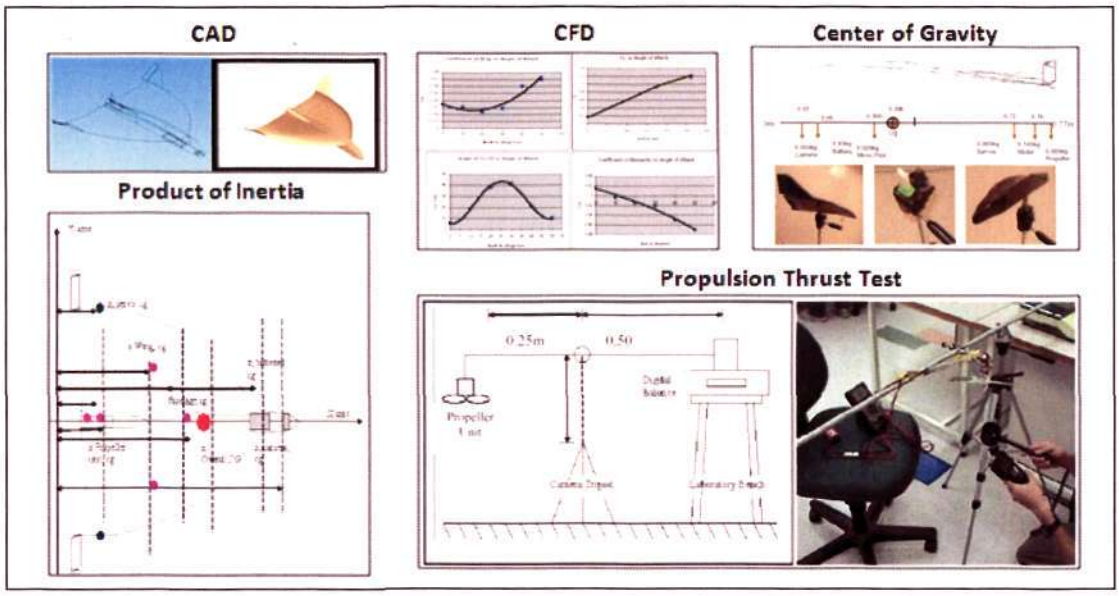

Figure 6.3: Flight Vehicle System Identification of *Golden UAV* (Jayabalan, 2005)

6.2.2. Adoption of Intelligent WPPs for Single and Multi UAV Operations

Another recommended future work is the optimization of UAV operation by adopting intelligent WPPs for the purpose of minimizing the operational costs, e.g. traveled distance. In chapter 5, the obstacle avoidance based on sensor information of laser rangefinders has been discussed. This capability is useful if mini UAV is deployed in unknown environment but it is not intelligent enough. In the future, a relevant WPP can be adopted into mini UAV system to increase the robustness of mini UAV flying among tall buildings and also to minimize UAV travelling distance.

Saunders et al (2005) adopts a priori WPP planning waypoint paths (see Figure 6.4) by exploring a terrain map using a Rapidly-Exploring Random Tree (RRT) algorithm. This algorithm is executed by considering the limitation of UAV turning radius and thus, it tends to find the more open areas to turn in.

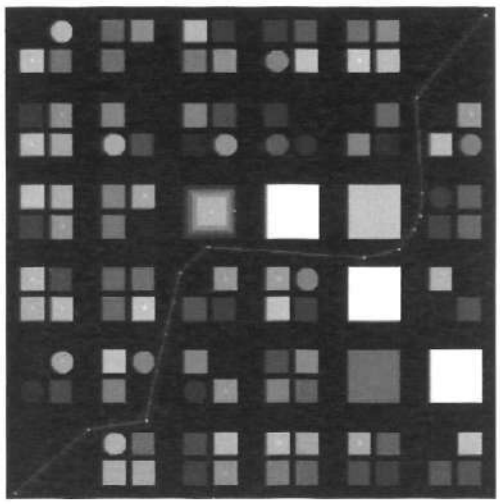

Figure 6.4: Rapidly-Exploring Random Tree (Saunders et al, 2005)

In near future, technology will allow mini UAV to process the computation of sensor information and decision making onboard. In this context, UAV simulation can be used as the prediction tool of future operations, such as Swarming UAVs, which offer a way of increasing reliability at low cost without relying much on technological support. Hart and Craig-Hart (2004) explores the opportunity of reducing robotic swarming theory into practice for UAVs. On the other hand, Parunak et al (2002) works on the development of swarming UAVs based on ant pheromone algorithm by using simulation as part of study methods.

To simulate multi-UAV behaviors, UAV modules must be developed as a multi thread architecture shown in Figure 6.5. Environment and Cooperative Planner are the only modules separated from the threads. Environment has the global information for every UAV and therefore, it is separated. On the other hand, Cooperative Planner is separately allocated to plan the best trajectory and workloads for each UAVs.

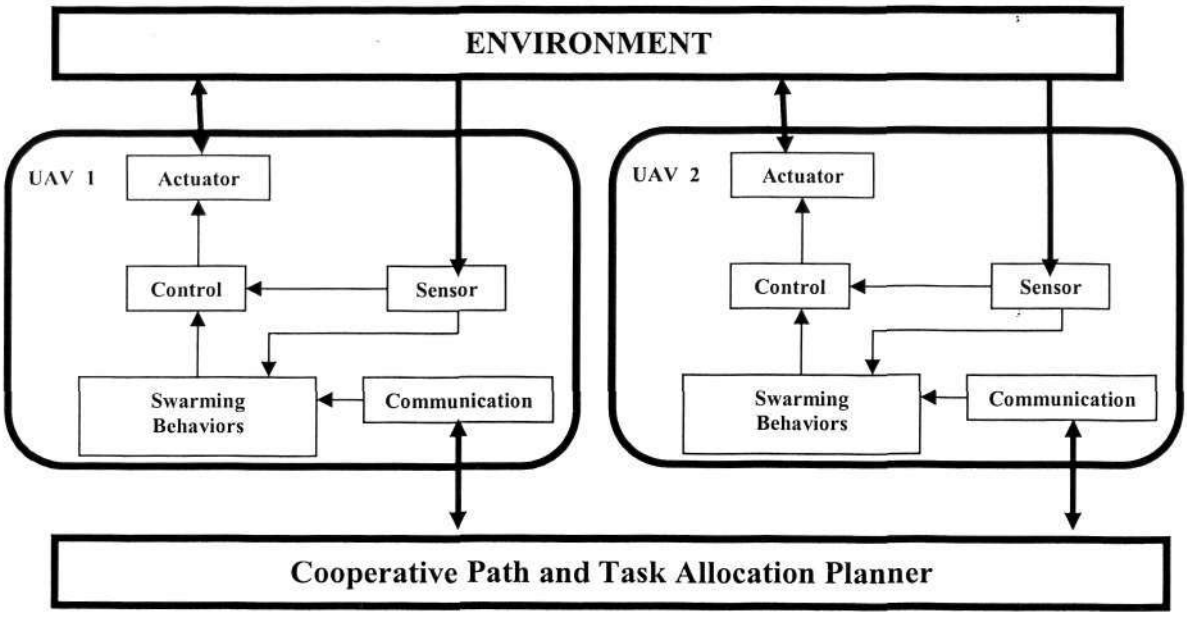

Figure 6.5: Two UAV modules are implemented as threads communicating with environment and Cooperative Planner

References

References: Chapter 1

1. Beard, R. and McLain, T., December 2003, *Multiple UAV Cooperative Search under Collision Avoidance and Limited Range Communication Constraints*, Proceedings of the IEEE Conference on Decision and Control, pp. 25-30, Maui, Hawaii.
2. Bruce T. Clough, 2002, "*Metrics, Schmetrics! How The Heck Do You Determine A UAV's Autonomy Anyway?*", Proceedings of the Performance Metrics for Intelligent Systems Workshop, Gaithersburg, Maryland.
3. Burstein, M., & McDermott, D., 1994, *Issues in the Development of Human-Computer Mixed-Initiative Planning System*, In M. Burstein (Ed.), Proceedings of the ARPA/Rome Laboratory Knowledge-Based Planning and Scheduling Initiative, (pp. 467-483), San Fransisco: Morgan Kaufmenn.
4. CAE, August 2005, *CAE STRIVE-SFX™*, URL: http://www.cae.com/www2004/Products_and_Services/Military_Simulation_and_Training/Modeling_and_Simulation/Software_and_Support/CAE_STRIVE-SFX.pdf
5. Coifman, B., McCord, M., Mishalani, R.G., Iswalt, M., Ji, Y., March 2006, *Roadway Traffic Monitoring from an Unmanned Aerial Vehicle*, Intelligent Transport Systems, IEEE Proceedings Volume 153, Issue 1, Page(s):11 – 20.
6. Cooke, N., Pringle, H. L., Pedersen, H. K., & Connor, O., 2006, (1st Eds), *Human Factors of Remotely Operated Vehicles*. Advances in Human Performance and Cognitive Engineering. Volume 7. Oxford, UK: Elsevier, pg. xix.
7. Ferguson, Michael G., Captain, U.S. Marine Corps, September 1999, *Stochastic Modeling of Naval Unmanned Aerial Vehicle Mishaps: Assessment of Potential Intervention Strategies,* MS in Operations Research, URL: http://stinet.dtic.mil/cgi-bin/GetTRDoc?AD=ADA371104&Location=U2&doc=GetTRDoc.pdf
8. Gugerty, L., 2004, *Challenges UAV Operators Face in Maintaining Spatial Orientation*, 1st Annual Workshop, Cognitive Engineering Research Institute, Chandler, Arizona.
9. Huang, H. M., E. Messina, and J. Albus, 2003, *Toward a generic model for Autonomy Levels For Unmanned Systems (ALFUS)*, Proc. PerMIS.
10. Johnson, E. and Fontaine, S., 2001, *Use of Flight Simulation to Complement Flight Testing of Low-Cost UAVs*, AIAA Modeling and Simulation Technologies Conference.
11. Kingston, D. B., April 2004, *Implementation Issues of Real-Time Trajectory Generation on Small UAVs*, Master's thesis, Brigham Young University, Provo, Utah 84602, URL: http://www.ee.byu.edu/magicc/publications/thesis/DerekKingston.pdf .

12. Leduc, P.A. Ph.D., Rash, C.E., M.S., Manning, S.D., M.S, 2005, *Human Factors in UAV Accidents*, Special Operations Technology Online Archives, Volume: 3, Issue: 8.

13. McGarity, M.S., May 9-12, 2005, *Simulation Across the UAV Life Cycle – Lessons Learnt*, SimTecT 2005, Sydney.

14. McManus, Iain and Greer, Duncan, and Walker, Rodney, 2003, *UAV Avionics "Hardware in the Loop" Simulator*, In Proceedings 10th Australian International Aerospace Congress, Brisbane, Queensland, Australia.

15. Peck, Michael, May 2003, *Pentagon Unhappy About Drone Aircraft Reliability*, National Defense Magazine.

16. Scerri, P., Pynadath, D., and Tambe, M., 2002, *Adjustable Autonomy for the Real World*, In Agent Autonomy, pg 163-190, Kluwer.

17. Vick, A.J., Stillion, J., Frelinger, D.R., Kvitky, J., Lambeth, B.S., Marquis, J.P., Waxman, M., 2002, *Aerospace Operations in Urban Environments*, RAND Project AIR FORCE Monograph Report, pg. 77, 83, and 87.

18. Weibel, R.E. and Hansman, R.J., 2005, *An Integrated Approach to Evaluating Risk Mitigation Measures for UAV Conceptual operations in the NAS*, AIAA-2005-6957.

19. World Resources: 1998–1999, *A Guide to the Global Environment*, 1998, Oxford: Oxford University Press, p. 146.

References: Chapter 2

20. Advanced Energy Tech, 2006, *Thunder Power Batteries TP8000-5S4PLV*, URL: http://www.thunderpower-batteries.com./Li-PolyBatteries.htm .

21. Altair Integrated System Flight Demonstration Project, 2005, *Channel Islands Survey/Day-Night*, UAS Flight Demonstration Project 2005, URL: http://uav.noaa.gov/altair/flights/flight1b.html .

22. Boryz, D. and R. Colgren, 15-18 August 2005, *Advances in Intelligent Autopilot Systems for Unmanned Aerial Vehicles*, Proceedings of the 2005 AIAA Modeling and Simulation Technologies Conference, San Francisco, CA.

23. Buretta A., Fowler, R., Germroth, M., Hayes, B., Miller, T., Neblett, E., Statzer, M., 1 May 2003, *Low-cost Expendable UAV*, Final Report, Virginia Tech, Department of Aerospace and Ocean Engineering.

24. Eyster, K., Hershberger, M., Rumbaugh, M., Cates, P., 2006, The Flying Slug, Design Report, 2005/2006 AIAA Foundation, Cessna/ONR Student, Design Build Fly Competition.

25. Griffiths, S., Saunders, J., Curtis, A., McLain, T., Beard, R., *Obstacle and Terrain Avoidance for Miniature Aerial Vehicles*, IEEE Robotics and Automation Magazine (submitted for review)

26. imSAR, 2006, *Insitu A-20 SAR*, URL: http://www.imsar.net/NanoSAR_DataSheet.pdf

27. King, Ellis T., 2004, *Distributed Coordination and Control Experiments on a Multi-UAV Testbed*, Massachusetts Institute of Technology. Dept. of Aeronautics and Astronautics.

28. Knacke, T.W., 1992, Parachute Recovery Systems-Design Manual, Para Publishing, Santa Barbara, California

29. Kotwani, K., Sane, S.K., Arya, H., Sudhakar, K., December 16-18, 2004, Experimental *Characterization of Propulsion System for Mini Aerial Vehicle*, 31st National Conference on FMFP, Jadavpur University, Kolkata.

30. Kyuho Lee, 2004, *Development of UAV for Wildlife Surveillance*, Master's thesis, University of Florida.

31. NASA Dryden, April 3, 2003, *Flight Demonstrations Evaluate Collision-Avoidance Technology*,

32. Obester, A., Keane, A. J., Scanlan, J., and Bresslo, N. W., 2005, *Conceptual Design of UAV Airframes Using a Generic Geometry Service*, AIAA Infotech@Aerospace, Arlington, VA.

33. Peter van Blyenburgh, March 2006, *UAV Systems: Global Review*, UVS International, Avionics'06 Conference, Amsterdam, Netherlands.

34. Plettenberg Elektromotoren, 2005, *Outrunner Brushless Motor XTRA 25-14*, URL: http://www.icare-rc.com/plettenberg_xtra25.htm .

35. Puscov, J., 2002, *Flight System Implementation in a UAV*, Diploma's Dissertation, AlbaNova University Center, Stockholm, Sweden.

36. Quigley, M., Barber, B., and Goodrich, M.A., August 2005, *Towards Real-World Searching with Fixed-Wing Mini-UAVs*, IEEE International Conference on Intelligent Robots and Systems (IROS), *Edmonton,* Alberta.

37. Riseborough, P., 20-23 July 2004, *Automatic Take-Off and Landing Control for Small UAVs*, Control Conference, 2004. 5th Asian, p.754- 762 Vol.2.

38. Saunders, J. B., Call, B., Curtis, A., Beard, R. W., and McLain, T. W., 2005, *Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles*, AIAA Infotech at Aerospace, Paper no. AIAA-2005-6950.

39. Thermoteknik Systems Ltd, 2006, *MIRICLE 110K*, URL: http://www.thermoteknix.com/Brochures/miricle/MIRICLE_110K.pdf .

40. Weibel, R.E. and Hansman, R.J., March 2005, *Safety Considerations for Operation of Unmanned Aerial Vehicles in the National Airspace System*, MIT International Center for Air Transportation Report, No. ICAT 2005-01, Cambridge, MA.

41. Yeung, W., Placido, P., Tang, E., Wasi, S., Genova, A., Wigley, D., Khorrami, F., 2006, *The Anatomy of Athena: The Engineering Behind Polytechnic University's UAV*, Journal Paper for the Fourth Annual Student Unmanned Aerial Vehicles (UAV) Competition.

42. Zaugg,E.C., Hudson,D.L., Long,D.G., 31 Jul. - 4 Aug. 2006, *The BYU microSAR: A Small, Student-Built SAR for UAV Operation, Proceedings of the International Geoscience and Remote Sensing Symposium*, Denver, Colorado.

References: Chapter 3

43. Chang, R., Lindsay, P., December 2005, *A Simulator for Exploring Autonomous Control of Multiple UAVs at Non-Radar Controlled Airstrips*, Intelligent Sensors, Sensor Networks & Information Processing Conference, 391-396.

44. Collinson, R. P. G., 1996, *Introduction to Avionics Systems*, 2nd ed., Kluwer Academic, pg. 86 and pg. 93

45. Goktogan, A., Nettleton, E., Ridley M., and Sukkarieh, S., 2003, *Real Time Multi-UAV Simulator*, in: IEEE. Internat. Conference on Robotics and Automation, Taipei.

46. Goldsmith, Andrea, 2005, *Wireless Communications*, New York: Cambridge University Press

47. Jones, E.D., Randy S. Roberts, and T. C. Steve Hsia, 2003, *STOMP: A Software Architecture for the Design and Simulation UAV-based Sensor Networks*, Proceedings of the IEEE International Conference on Robotics and Automation.

48. Kotwani, K., Sane, S.K., Arya, H., Sudhakar, K., December 16-18, 2004, Experimental *Characterization of Propulsion System for Mini Aerial Vehicle*, 31st National Conference on FMFP, Jadavpur University, Kolkata.

49. Lluch, D., 2002, *Building Multi-UAV Simulation Methods*, Paper No. AIAA-2002-4495, AIAA MST

50. McManus, Iain and Greer, Duncan, and Walker, Rodney, 2003, *UAV Avionics "Hardware in the Loop" Simulator*, In Proceedings 10th Australian International Aerospace Congress, Brisbane, Queensland, Australia.

51. McLain, T., Chandler, P., Rasmussen, S., and Pachter, M., June 2001, *Cooperative Control of UAV Rendezvous*. In Proceeding of the ACC.

52. McNally, S., 1999, *Binary triangle trees and terrain tessellation [J], Computer & Graphics*, 20 (4): 541～566

53. Myeong-Wuk, J., Reddy, S., Tosic, P., Chen, L., Agha, G., October 26-29, 2003, *An Actor-based Simulation for Studying UAV Coordination*, 15th European Simulation Symposium (ESS 2003), pp. 593-601, Delft, Netherlands.

54. Quigley, M., Barber, B., and Goodrich, M.A., August 2005, *Towards Real-World Searching with Fixed-Wing Mini-UAVs*, IEEE International Conference on Intelligent Robots and Systems (IROS), *Edmonton,* Alberta.

55. Riseborough, P., 20-23 July 2004, *Automatic Take-Off and Landing Control for Small UAVs*, Control Conference, 2004. 5th Asian, p.754- 762 Vol.2.

56. Robbins D., Anderson M., 1998, *Formation Flight as a Cooperative Game*, AIAA-98-4124, AIAA GNC

57. Savla, K., Frazzoli, E., and Bullo, F., June 2005, *On The Point-to-Point and Traveling Salesperson Problems for Dubins' Vehicle,* American Control Conference, Portland.

58. Wee Ching, P. July 2006, *Robotic Simulator: D5. Final Review*, Project No. *POD03039716*, Robotic Research Center, Nanyang Technological University.

59. Yeung, W., Placido, P., Tang, E., Wasi, S., Genova, A., Wigley, D., Khorrami, F., 2006, *The Anatomy of Athena: The Engineering Behind Polytechnic University's UAV*, Journal Paper for the Fourth Annual Student Unmanned Aerial Vehicles (UAV) Competition.

References: Chapter 4

60. Abt, C.C., 1964, *War Gaming*. International Science & Technology, 32, 29-37.

61. AIAA, 1998, *Guide for the Verification and Validation of Computational Fluid Dynamics Simulations*, American Institute of Aeronautics and Astronautics, AIAA-G-077-1998, Reston, VA.

62. Coyle, R.G., 1977, Management System Dynamics, Chichester, John Wiley and Sons.

63. Forrester, J.W. and Senge, P., 1980, *Tests for building confidence in System Dynamics Models*, TIMS Studies in the Management Sciences, Vol 14, pp209-228.

64. Garzia, R.F. and Garzia, M.R., 1990, *Network Modeling, Simulation, and Analysis*, Marcel Decker, New York.

65. Giannanasi, F., Lovett, P. and Godwin, A.N., 2001, *Enhancing Confidence in Discrete Event Simulations*, Computers in Industry, Vol 44, pp141-157.

66. Hale, F.J., 1984, *Introduction to Aircraft Performance, Selection, and Design*. John Wiley & Sons, Inc, pg.137 and pg.152

67. Khazanchi, D., August 1996, *A Framework for the Validation of IS Concepts*, Proceedings of the Second Annual Association for Information Systems Americas Conference, Phoenix, Arizona.

68. Kleijnen, J.P.C, 1999, *Validation of Models: Statistical Techniques and Data Availability*, Proceedings of the 1999 Winter Simulation Conference, P. A. Farrington, H. B. Nembhard, D. T. Sturrock and G. W Evans, (eds.), pp647- 654.

69. Martis, M. S., 2006, *Validation of Simulation Based Models: A Theoretical Outlook*, The Electronic Journal of Business Research Methods Volume 4 Issue 1, pp 39 -46.

70. Pace, D.K., 2004, *Modeling and Simulation Verification and Validation Challenges*, Johns Hopkins Apl Technical Digest, Volume 25, Number 2.

71. Schlesinger, S., 1979, *Terminology for Model Credibility Simulation*, 32, 3, 104-4.

72. Shannon, R.E., 1975, *System Simulation,* The Art and Science Englewood Cliffs: Prentice Hall.

73. Sterman, J.D., 1984, *Appropriate Summary Statistics for Evaluating the Historic Fit of System Dynamics Models*, Dynamica, Vol 10, pp51-66.

74. Sterman, J.D., 2000, *Business Dynamics: Systems Thinking and Modeling for a Complex World*, Irwin McGraw-Hill, p845.

75. Whitner, R. B. and Balci, O., 4-6 December 1989, *Guidelines for selecting and using simulation model verification techniques*, In Proceedings of the 1989 Winter Simulation Conference, Washington, D.C.

References: Chapter 5

76. Civil Aviation Safety Authority Australia, July 2002, *UAV Operations, Design Specification, Maintenance, and Training of Human Resources*, Advisory Circular AC 101-1(0)

77. Griffiths, S., Saunders, J., Curtis, A., McLain, T., Beard, R., *Obstacle and Terrain Avoidance for Miniature Aerial Vehicles*, IEEE Robotics and Automation Magazine (submitted for review)

78. Hrabar, S. E. and Sukhatme, G.S., Oct 2006, *Optimum Camera Angle for Optic Flow-Based Centering Response*, IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3922 – 3927

79. Hrabar, S. and Sukhatme, G., 2004, *A Comparison of Two Camera Configurations for Optic-Flow Based Navigation of a UAV Through Urban Canyons*, IEEE/RSJ International Conference on Intelligent.

80. Muratet, L., Doncieux, S., Briere, Y., and Meyer J.A. , 2005, *A Contribution to Vision-Based Autonomous Helicopter Flight in Urban Environments*. Robotics and Autonomous Systems.

81. Pisanich, G., Morris, S., 2001, *Fielding An Amphibious UAV: Development, Results, and Lessons Learned*; NASA Ames Research Center and MLB, Palo Alto, CA.

82. Saunders, J. B., Call, B., Curtis, A., Beard, R. W., and McLain, T. W., 2005, *Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles*, AIAA Infotech at Aerospace, Paper no. AIAA-2005-6950.

83. Sherrod, P.H., 2003, *Classification and Regression Trees and Support Vector Machines for Predictive Modeling and Forecasting*, URL: http://www.dtreg.com/DTREG.pdf

84. Shim, D., Chung, H., Kim,H. J., Sastry, S., August 2005, *Autonomous Exploration in Unknown Urban Environments for Unmanned Aerial Vehicles*, AIAA GN&C Conference, San Francisco.

85. United Nations, 24 February 2005, *World Population Prospects: The 2004 Revision*, Social Affairs, New York

86. Wagter, C.D. and Mulder, J.A., August 2005, *Towards Vision-Based UAV Situation Awareness*, AIAA Guidance, Navigation, and Control Conference and Exhibit, AIAA 2005-5872, San Francisco, California.

References: Chapter 6

87. Hart, D.M., Craig-Hart, P.A., 2004, *Reducing swarming theory to practice for UAV Control*, Aerospace Conference, Proceedings. IEEE: Page(s): 3050- 3063 Vol.5.

88. Jategaonkar, R. V., 2006, *Flight Vehicle System Identification : A Time Domain Methodology*, American Institute of Aeronautics and Astronautics

89. Jayabalan, N., 2005, *Aerodynamics of a Flying Wing UAV*, Department of Mechanical Engineering, National University of Singapore

90. Parunak, H. V. D., Purcell, M., and O'Connell, R., 2002, *Digital Pheromones for Autonomous Coordination of Swarming UAV's*, In Proceedings of First AIAA Unmanned Aerospace Vehicles, Systems, Technologies, and Operations Conference, AIAA

91. Platanitis, G., and Shkarayev, S., September 26-29, 2005, *Integration of an Autopilot for a Micro Air Vehicle*, Arlington, VA, AIAA-2005-7066

92. Saunders, J. B., Call, B., Curtis, A., Beard, R. W., and McLain, T. W., 2005, *Static and Dynamic Obstacle Avoidance in Miniature Air Vehicles*, AIAA Infotech at Aerospace, Paper no. AIAA-2005-6950.

APPENDICES

Appendix A. Simulator Development

Both Chapter 3 and Appendix A describes about the simulator but if Chapter 3 concerns more about *What have been built?*, then Appendix A is interested on *How was it built?* In this project, the development of UAV simulator is based on the iterative process of prototyping, testing, analyzing, and refining works as illustrated in Figure a.1. Challenges and difficulties were experienced during the development as the Localization and Correction of Failures are presented at the end of this section.

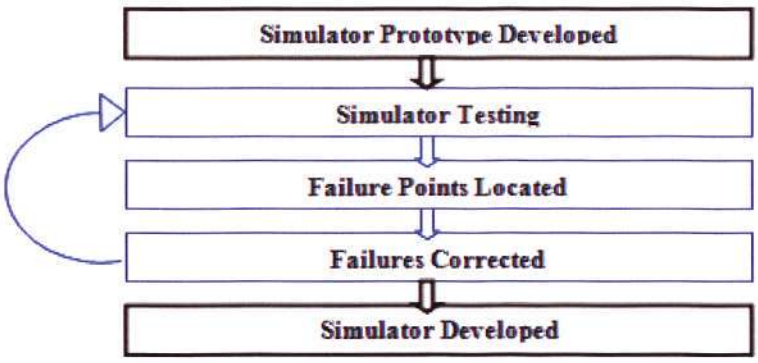

Figure a.1: Iterative Process of Simulator Development

A1 Prototype of Simulator

During its first stage, simulator was developed as a prototype. Software prototyping is the process of creating an incomplete model of the future full-featured software program as an aspect of a *proof of concept*, which can be used to let software engineer some insight into the accuracy of initial project estimates, to allow users to have a first idea of the completed program, and to let the clients early evaluate the program.

The prototyping technique in this project is based on Evolutionary Prototyping. The main goal is to build a robust prototype in a structured manner and constantly refine it. The reason for this is that the evolutionary prototype, when built, forms the heart of the new system, and the improvements and further requirements will be built on to it (Crinnion, 1991).

The prototype was written in a higher level language – Matlab due to its faster development as compared to conventional language, such as C or C++. This is also because Robbins and Anderson (1998) architecture was written in Matlab by Lluch (2002) to demonstrate swarming behaviors. The main architecture of his simulator can be seen in Figure a.2.

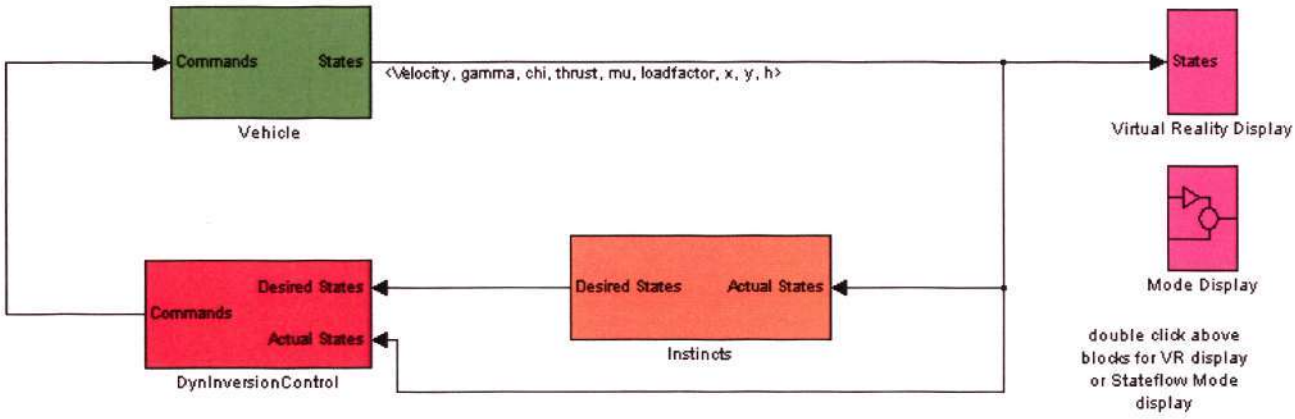

Figure a.2: Architecture of Lluch (2002) Simulator

Lluch (2002) software was then examined for any potential developments relevant to this project. The result concludes that the architecture of this software is open, flexible, and expandable for the future

developments of simulator into WPP or lower level, such as TT as illustrated earlier in Figure 1.9. Additionally, the architecture is based on the closed loop control system, which allows operator and/or WPP to input high level instructions and let the controller interpret these instructions into low level commands, which are then followed by aerodynamic motions.

The demonstrated swarming behaviors are based on 4 hierarchical instincts that are typical to potential field representing repulsion and attraction. These instinctive behaviors are represented as 4 velocity components (see Figure a.3) as follows.

- Aircraft Collision Avoidance $\vec{V}_a$ that prevents collisions among UAVs and is put at the highest priority
- Obstacle Avoidance $\vec{V}_o$, which prevents the collision to nearest obstacle
- Target Acquisition $\vec{V}_t$ that guides UAV to particular point of target
- Group binding $\vec{V}_g$ that ensures UAVs to stick together in a group. It is positioned at the lowest priority

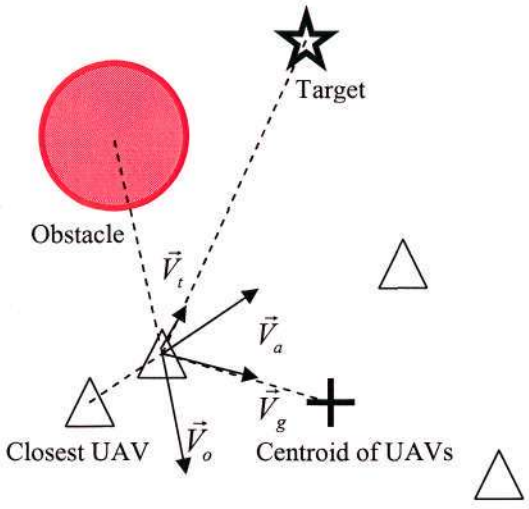

Figure a.3: Diagram of 4 hierarchical velocity components

These instincts all weigh in to dictate what UAV will be commanded to do. By examining the maximum weight of the instincts, the UAV can be said to be in one of four modes. It is important to note that all instincts come into play when each vehicle decides how to maneuver, yet only one instinct will dominate at any particular time.

The atomic motions of attraction and repulsion are not only useful to perform instinctive behaviors for maintaining the UAV robustness, but also beneficial to be expanded into deliberative behaviors for the optimization of mission performance. In this project, deliberative behaviors were firstly realized as automatic takeoff and landing. Both were developed from a set of pre-programmed go-to-waypoints, which were the results of target acquisition instinct mentioned earlier. More details of pre-programmed behaviors can be found at *Chapter 5. Study on Simulated Behaviors.*

Some other UAV simulators are expandable at different autonomy levels as well. For example, Gancet et al (2005) equipped their simulator with Multi-Level Executive that is able to alternate the autonomy level from centralized to distributed decisional capability. Similarly, the architecture of Myeong-Wuk (2003) and Chang (2005) consists of hierarchical modules that are grouped based on different autonomy levels, such as Cooperative Module, which manages the allocation of multi UAVs for a set of defined targets, Global Control module that manages waypoints received from Ground Control System (GCS), and Trajectory Plan Module, which performs navigation capability, e.g. waypoint acquisition, conflict detection and resolution.

In addition, the separation of intelligent system, controller, and physical agent opened the opportunity of providing the simulator with more detail models. Based on Figure a.2, Lluch (2002) software was equipped only with aerodynamic and basic controller model. By adding sensor, propulsion, and environmental models, the simulator becomes more realistic and it was then regarded as the prototype of UAV simulator as illustrated in Figure a.4.

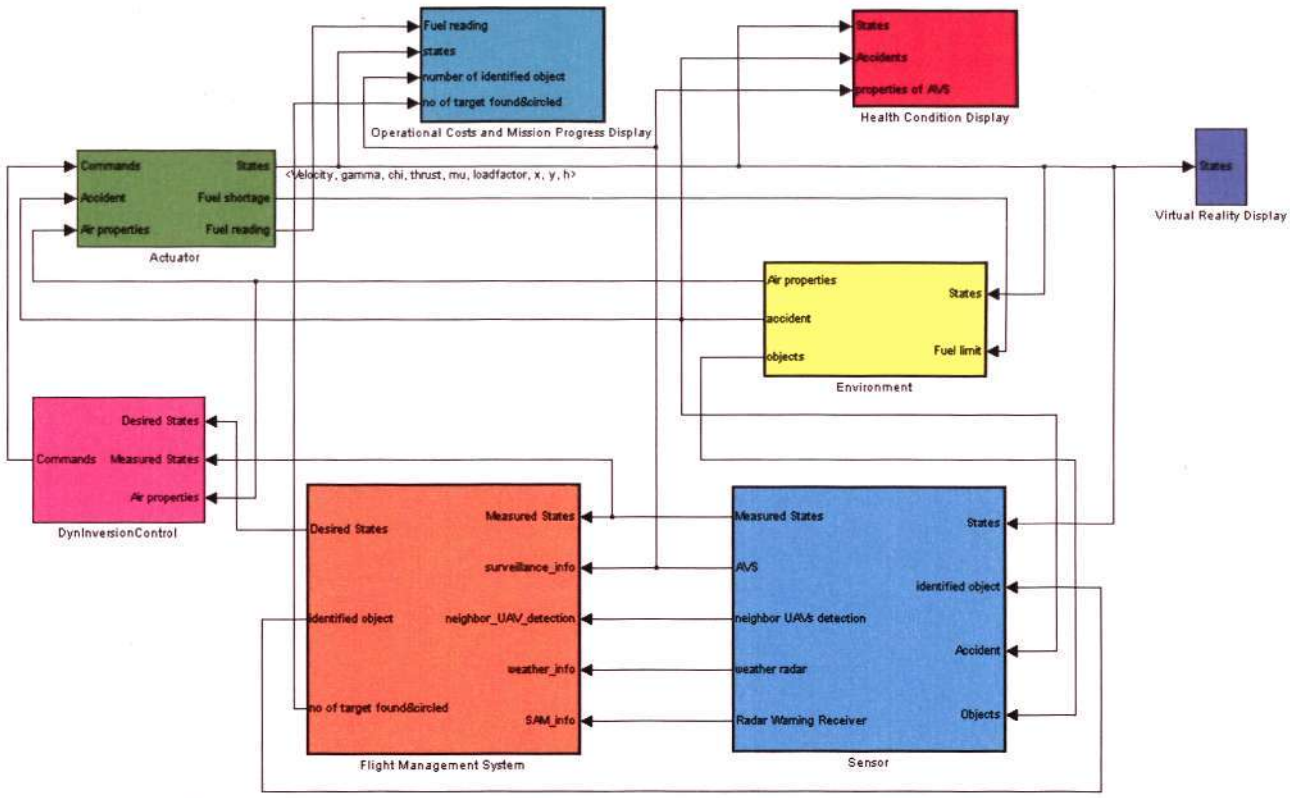

Figure a.4: The Architecture of Prototype developed from Lluch (2002) Simulator

So having the prototype completely built, there were some generated requirements and lessons learned. They are listed in the followings.

i) Modifying the aerodynamic parameters of UAV

The aerodynamic properties of original UAV model were typical to High Altitude Endurance UAV, e.g. *Global Hawk*. Based on this fact, there was a need to change aircraft parameters, e.g. wing area, weight, and etc. into the category of mini UAV. The modified parameters are listed in section B2. RRCUAV Parameters.

ii) Practicability

Lack of proofs on the technology supporting swarming behaviors, which were demonstrated in Lluch (2002) simulator, raised an issue that the simulation must be based on existing technology, which is commercially available or academically proven. In his simulation, he assumed individual UAVs were able to process information and make decision onboard and in real time. These are practically untrue since current technology is only able to deploy multiple UAVs in centralized manner, which all UAVs' perception and decision making fully rely at the ground control station.

So that is why, the development of UAV should start from the simplest behavior and thus, the lowest autonomy level to express the most valid relation. The addition of sensor models required the conceptual validation of existing sensor technology, which later led to catalog studies on potential sensors fulfilling mini UAV design requirements. These works has been mainly addressed in *Chapter 2. Conceptual Design of RRCUAV*.

iii) Supportive graphic and execution speed for real time simulation

The addition of environmental models required supportive graphics and visualization tools performing reasonable computation speed. In Matlab, 3D object rendering is very limited. At the altitude of more than 500 m, the terrain was disappeared due to lack of memory space and it requires more time to execute behaviors, which are based on onboard planning.

The drawback of Matlab later motivated the conversion of simulation programming language into C++ for better rendering and faster execution speed. Furthermore, other relevant project by Wee Ching (2006) simulating autonomous ground vehicles, provided a relevant platform and potential development

of mixed multi-agent collaboration for aerial and ground vehicles in the future. Although both are simulated in similar platform, the operation of ground vehicle and UAV is purely independent in this project.

A2 Localization and Correction of Failures in *ROBOSIM*

Software development plans drawn on this project are motivated from the development of autonomous behaviors, which are discussed in much detail in *Chapter 5. Study on Simulated Behavior*. During the process of development, there were number of failures or errors that need to be fixed. The followings are the chronological refinements of the simulator as the problems or errors are identified. Some tools that help on the analysis and performance study are included as well in this section.

A2.1 Porting UAV module on ROBOSIM

i) Conversion of UAV module from Matlab to C++

Generally, a continuous dynamic simulator performs numerical solution of differential equations. The simulation program periodically solves all the equations and uses the numbers to change the state and output of the simulation. The Simulink feedback-loop of the prototype was replaced by state updater in *ROBOSIM*. It principally works based on *While Loop*, as illustrated in Figure a.5.

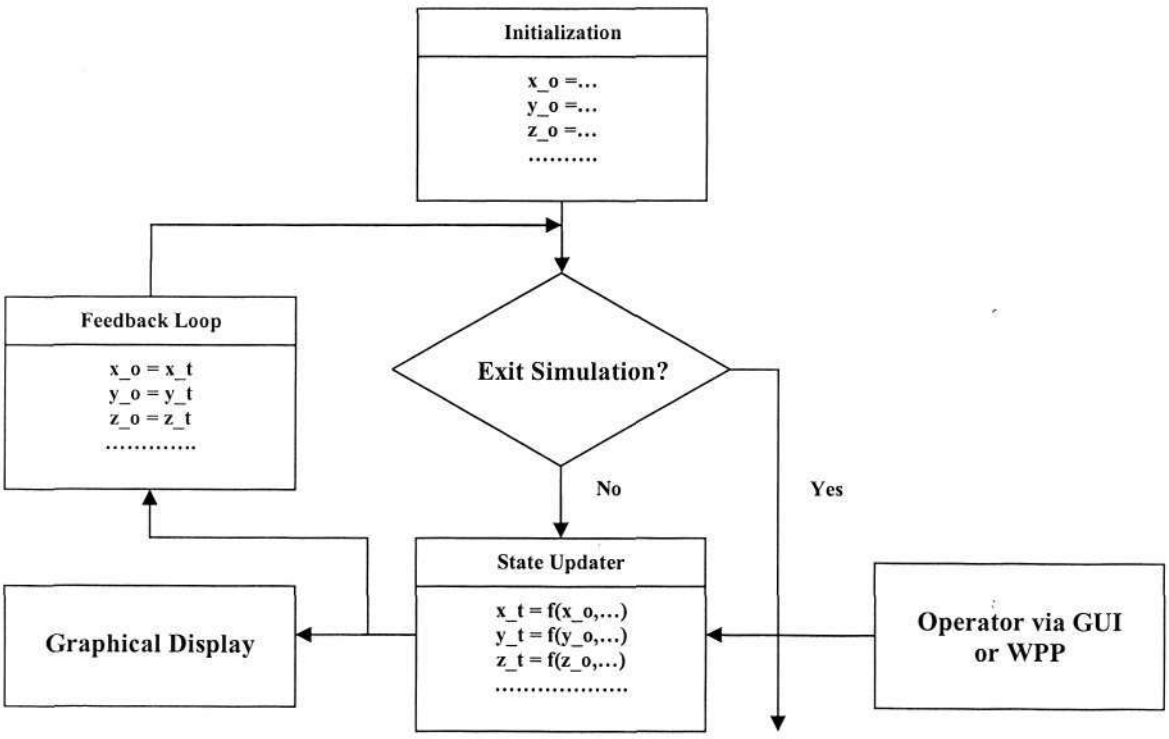

Figure a.5: While-Loop of State Update

In this loop, each parameter is initialized and indicated by subscript "o", e.g. *x_o* for initial x coordinate position. After some calculations, some initial parameters will be updated into variables, which are indicated by subscript "t", e.g. *x_t* for updated x coordinate position. These parameters are different from fixed parameters, such as weight and wing area of *RRCUAV*. Variable parameters are changed according to the calculation, which is formulated in *Appendix B. Mathematical Models of RSuav Module and Environment* and it is based on the instructions from operator or WPP. The updated parameters will be then assigned back to initial one by e.g. *x_o = x_t;* The process will be iteratively executed as long as simulation runs. It can be terminated by pressing either "Exit Simulation" GUI button or "Esc" keyboard button.

ii) Interfacing the outputs of *RSuav module* on *ROBOSIM*

The six variable outputs of x, y, and z coordinate and orientation angle of pitch γ, yaw χ, and roll ϕ could not be directly interfaced into graphic display of *ROBOSIM*. This is because the initial platform involving ground vehicle module only makes use of its angular and translational speed to display its position. Therefore, there was a need to modify the initial platform in order to accommodate the six variables generated by UAV model.

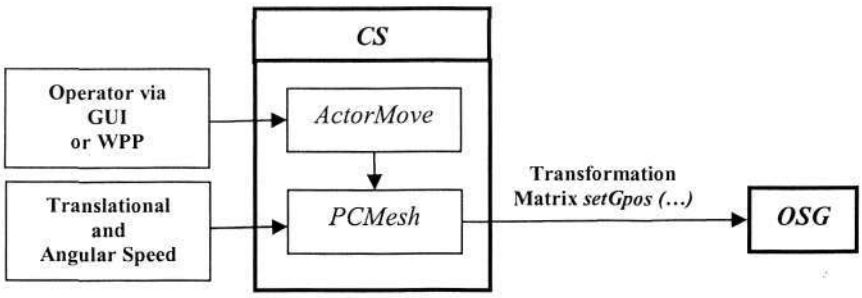

Figure a.6: The Original Platform of Graphic Display Interface

Based on the original platform illustrated in Figure a.6, operator or WPP sends motion instruction to *ActorMove*. The information is then the passed to *PCMesh*, which receives translational and angular speed of ground vehicle to update the position of ground vehicle. After that, the updated position is sent from *CS* to *OSG* for graphic visualization through transformation matrix *setGpos (...)*.

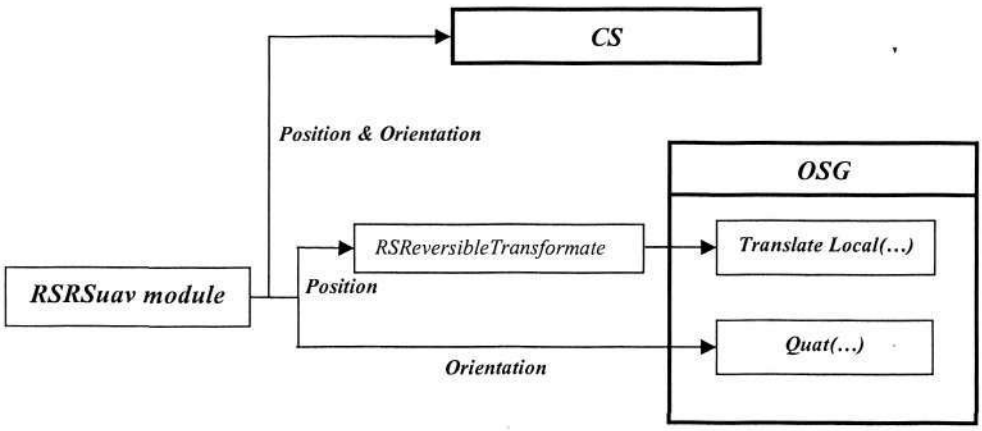

Figure a.7: The Modified Platform applied for *RSuav* module

The modification of original platform is illustrated in Figure a.7 and described as follows. *RRCUAV* position and orientation are distributed to both *CS* and *OSG*. Since *RSuav* module uses different coordinate systems to *OSG*, coordinate transformation function *RSReversibleTransformate* illustrated in Figure a.8 is therefore provided.

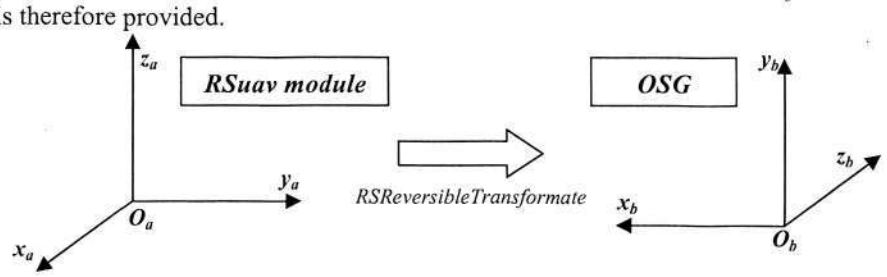

Figure a.8: Coordinate Transformation from *RSuav* module to OSG

The information received by *OSG* is then represented as transformation matrix comprising the translation and rotation of 3D model. Translation is generated from function *TranslateLocal(x_o,y_o,z_o)* and rotation by quaternion function *Quat(ϕ,γ,χ)*. Although the modification spends more computational

cost for simulation, this is the most effective way of interfacing *RRCUAV*'s position and orientation to *ROBOSIM* graphic display.

A2.2. Go-to-Waypoint

i) Structural Modification

The modification of graphic display interface results in the structural change of go-to-waypoint of *RSuav* module as well. In original platform, set of generated waypoints are scheduled by *First In First Out (FIFO)* algorithm as the array of selected waypoint is received from *RSclient*, which is responsible for getting the information regarding the ground vehicle's speed and orientation and sending motion commands to drive the *ATRV*. *RSplanner* then directs the vehicle moving towards the defined waypoint and checks if vehicle has achieved the waypoint. *RSplanner* described in Figure a.9 is the Artificial Intelligence (AI) plug-in that provides the basic navigation capability equipped with two hierarchical behaviors: Obstacle Avoidance and Go-to-Waypoint.

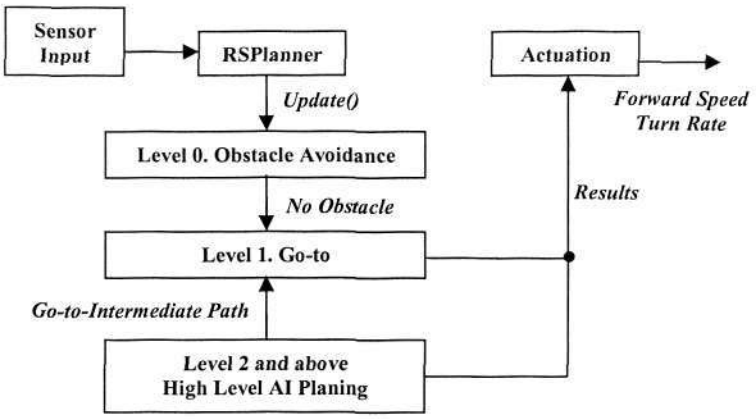

Figure a.9: *RSplanner* (Wee Ching, 2006)

In contrast, *RSuav* module employs its own go-to-waypoint scheduler and path planner. The go-to-waypoint scheduler is slightly different from *RSplanner*. In *RSplanner*, ground vehicle stops at the last waypoint. On the other hand, the go-to-waypoint scheduler of *RSuav* module makes *RRCUAV* looped back to the first waypoint simply because it cannot hover at the same place for longer time.

The exclusion of *RSclient* and *RSplanner* plug-in is also because the kinematics state between aircraft and ground vehicle is entirely different. Additionally, aircraft motion is not as stable as ground vehicle so it requires a more sophisticated controller to minimize the difference between high level instructions coming from operator or WPP, and the kinematics states of *RRCUAV*.

ii) Visualization Tools

To effectively study the accuracy and stability of go-to-waypoint operation, the visualization of trajectory track and target indicator needs to be provided. The trajectory plotter was then developed from Linux built-in plotter – *GNUplot*, which constantly reads *UAV.dat* that *RSuav* module periodically writes its position on. During the plotting, coordinate is automatically scaled and it can be visualized as 3D view as shown in Figure a.10.

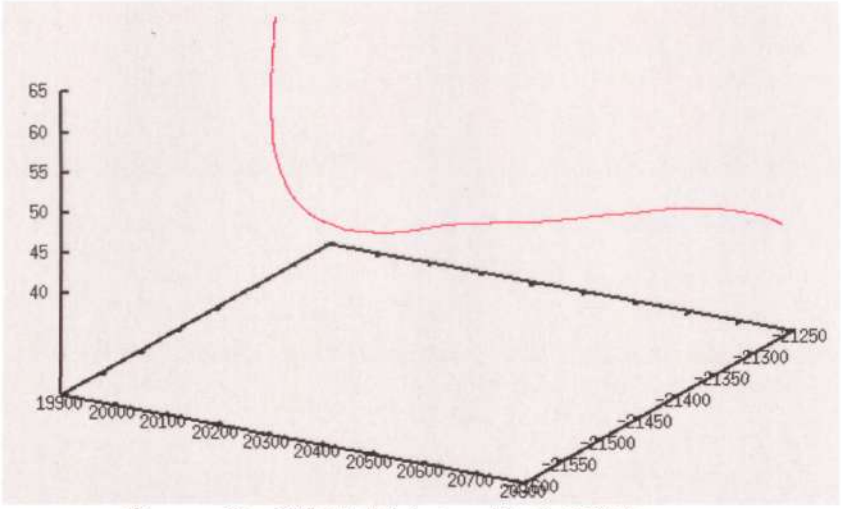

Figure a.10: *GNUplot* Trajectory Plot in 3D View

Furthermore, semi transparent sphere with defined radius is visualized in graphic display to indicate the waypoint. By having the sphere as shown in Figure a.11, the go-to-waypoint accuracy can be estimated if *RRCUAV* achieves the waypoint within defined radius.

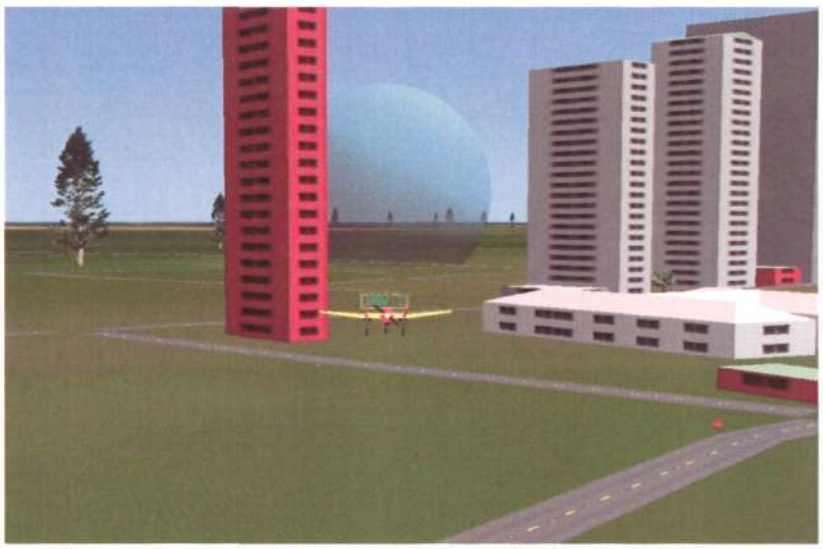

Figure a.11: Semi transparent sphere indicating the waypoint

A2.3. Automatic Takeoff and Landing

i) Height Adjustment

The prototype of simulator set the whole terrain flat or as 0 meter height in Matlab virtual environment. At the first time of its implementation in *ROBOSIM*, this caused *RRCUAV* walked underground during takeoff and landing. This is because in *ROBOSIM*, most of terrain is higher than 0 meter.

By adopting height adjustment function *pHF->FindAltitude* of ground vehicle module, *RRCUAV* lying on the ground finally moved according to terrain height. This function grabs the longitude and latitude coordinates of *RRCUAV* and then assigns the corresponding terrain height to *RRCUAV* altitude coordinate.

ii) Crash Detection Function

The visualization of ground impact is useful to assess the safety of landing. The intensity of ground impact is indicated by the vertical speed of *RRCUAV* touching the ground. In this model, vertical speed of

less than -5 m/s (negative indicates descent) is assumed to damage *RRCUAV* and thus, its commanded thrust is therefore set to zero that later causes *RRCUAV* loses its control. Furthermore, function *EnMgr->addExplosionToRobot* enhances the sense of reality by deploying visual effect of explosion as shown in Figure a.12.

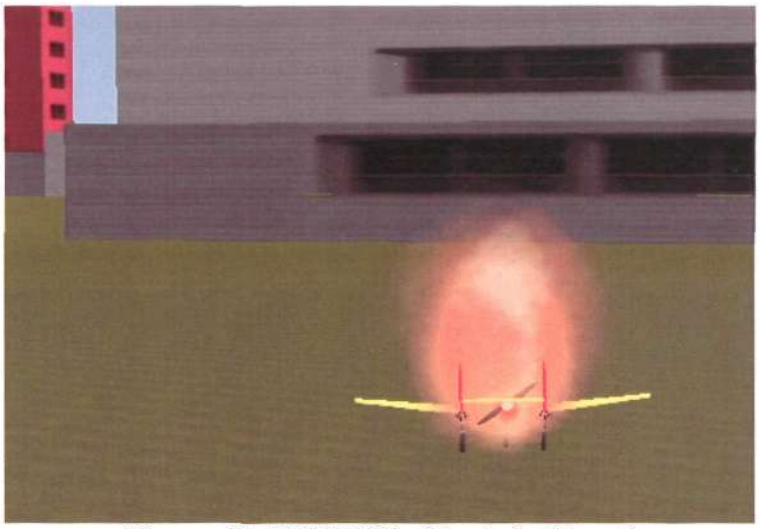

Figure a.12: *RRCUAV* Crashing to the Ground

A2.4. Obstacle Avoidance

i) Graphic Collision Detection of Laser Beam

The model of laser beam working based on CD had been present in ground vehicle module earlier. That is why the model of laser rangefinder, which was implemented in *RSuav* module, is the modification of SICK Laser Radar (LADAR) illustrated in Figure a.13.

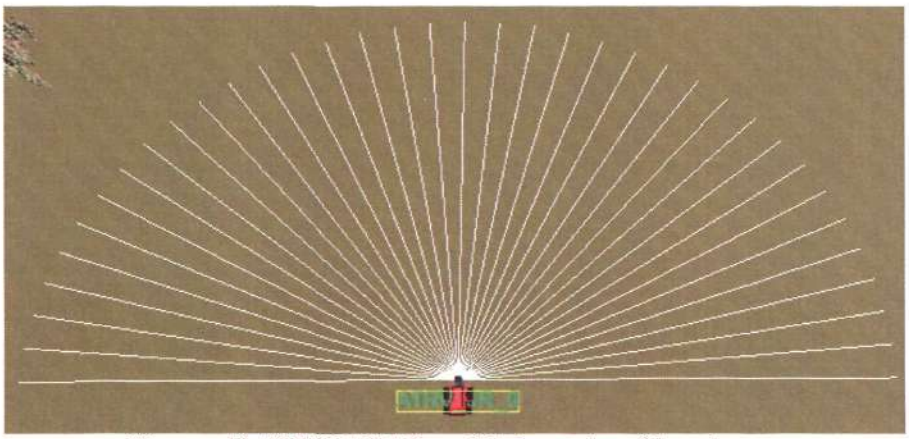

Figure a.13: SICK LADAR modeled as series of laser beams

In Figure a.13, range of SICK LADAR is represented as a group of beams with 5^{o} angular resolution. By having the corresponding left and right of these beams from the center, Laser Rangefinder is therefore modeled on the nose of *RRCUAV* with defined laser opening angle θ (see Figure 2.13).

ii) Automatic Generated Scenarios for Experimental Tests

In chapter 5, experimental test is performed to measure how factors, e.g. building width, laser range finder resolution, and etc affect the performance of *RRCUAV* obstacle avoidance. It is expected that the result of experimental test will be able to determine the most significant factor to mission performance, e.g. robustness. To make the test run more efficient, different scenarios are generated automatically. Each scenario requires *RRCUAV* moving towards particular waypoint from its initial position.

Figure a.14 illustrates the diagram flow of automatic generated scenarios that progresses if *RRCUAV* either crashes the building or achieves its minimum distance to defined waypoint. *RRCUAV* is then re-deployed to a new scenario with different environments and /or sensor specifications.

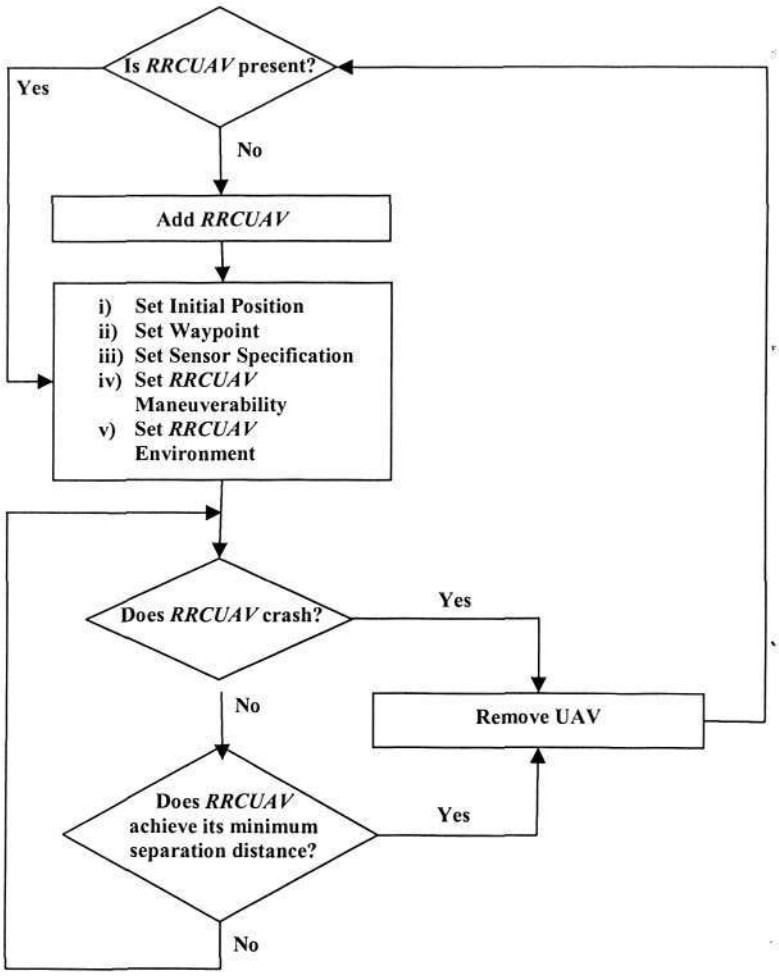

Figure a.14: Flow Diagram of *RRCUAV* Experimental Test on Obstacle Avoidance

iii) Visualization Tools

Having an extra background of 2D building in *GNUplot* will be helpful to study the experimental test of obstacle avoidance. The visualization will provide some knowledge about how *RRCUAV* operates in both low and high constrained environment and in what condition *RRCUAV* might fail its obstacle avoidance maneuver. The *GNUplot* was then modified into something shown Figure a.15.

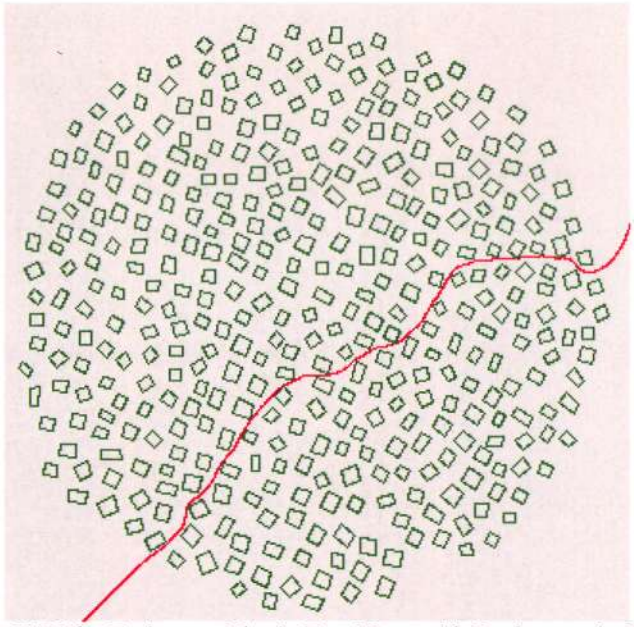

Figure a.15: *GNUplot* Trajectory Plot in Top View with Background of 2D Building

References: Appendix A

1. Chang, R., Lindsay, P., December 2005, *A Simulator for Exploring Autonomous Control of Multiple UAVs at Non-Radar Controlled Airstrips*, Intelligent Sensors, Sensor Networks & Information Processing Conference, 391-396.
2. Crinnion, J. 1991, Evolutionary Systems Development, *A Practical Guide to The Use of Prototyping within a Structured Systems Methodology*. Plenum Press, New York, Page 18.
3. Gancet, J., Hattenberger, G., Alami, R., and Lacroix, S., 2005, *Task Planning and Control for A Multi-UAV System: Architecture and Algorithms*, IEEE Intl. Conf. on Intelligent Robots and Systems (IROS), Edmonton, CAN.
4. Myeong-Wuk, J., Reddy, S., Tosic, P., Chen, L., Agha, G., October 26-29, 2003, *An Actor-based Simulation for Studying UAV Coordination*, 15th European Simulation Symposium (ESS 2003), pp. 593-601, Delft, Netherlands.
5. Lluch D., 2002, *Building Multi-UAV Simulation Methods*, Paper No. AIAA-2002-4495, AIAA MST.
6. Robbins D., Anderson M., 1998, *Formation Flight as a Cooperative Game*, AIAA-98-4124, AIAA GNC.
7. Wee Ching, P., July 2006, *Robotic Simulator: D5. Final Review*, Project No. *POD03039716*, Robotic Research Center, Nanyang Technological University.

Appendix B. Mathematical Models of *RSuav* Module and Environment

This section provides the mathematical descriptions of models in *RSuav* module and Environments.

B1. Models of *RRCUAV* Systems

B1.1 Aerodynamic Properties

Model of aerodynamic system consists of 9 actual states. They are as follows.

(i) Thrust (T)

It plays as driving force of aircraft. In this simulation, International unit system is used and thrust is calculated in N.

(ii) Load Factor (n)

It is a ratio between lifting force and weight of aircraft. Load factor must be greater or equal to one to lift up the aircraft.

(iii) Bank Angle (ϕ)

Aircraft needs to roll at certain angle so it can change its direction. This angle is called as Bank or rolling angle as described in Figure b.1. This is the only state that causes aircraft to turn.

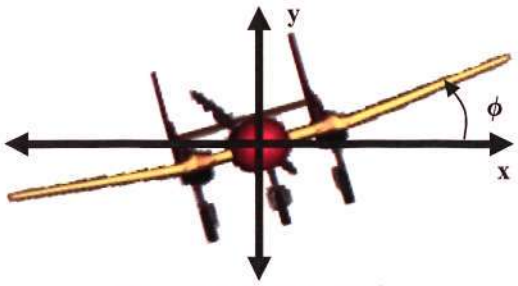

Figure b.1: Bank angle ϕ

(iv) Position in x, y, z coordinate

Each position of aircraft will be expressed in x, y, z coordinate.

(v) Speed (V)

Please be reminded that speed is a scalar. It represents the magnitude of velocity

(vi) Flight path angle (γ)

Figure b.2 shows flight path/pitching angle. During take off and landing, pitching angle changes rapidly.

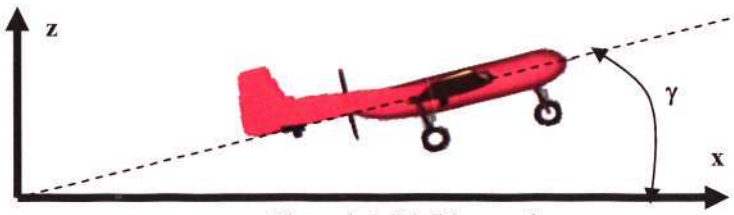

Figure b.2: Pitching angle γ

(vii) Flight path heading angle (χ)

Flight path heading/yawing angle lies in x-y plan as can be seen in Figure b.3.

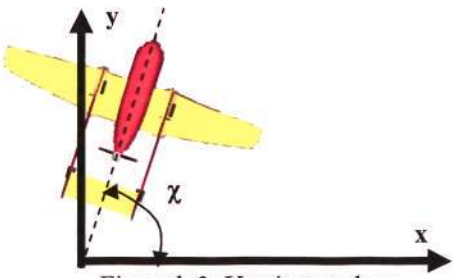

Figure b.3: Yawing angle χ

B1.2 Intelligent System

Based on Robbins and Anderson (1998) architecture, Intelligent System generates *Desired States*, which are originated from $\vec{V}_T$ ' – a unit vector indicating the intended direction in 3D space. $\vec{V}_T$ ' is then multiplied by aircraft speed V and the product is defined as instructed velocity $\vec{V}_T$ formulated in equation b.1.

$$V_T = V_T' \times V \tag{b.1}$$

This value is then split into three coordinate elements: V_{Tx}, V_{Ty}, and V_{Tz}, which are later used to determine *Desired States*. They are represented into desired speed V_d, pitching γ_d, and heading χ_d as calculated from Equation b.2 to b.4.

$$V_d = \sqrt{V_{Tx}^2 + V_{Ty}^2 + V_{Tz}^2} \tag{b.2}$$

$$\gamma_d = \tan^{-1}\left(\frac{V_{Tz}}{\sqrt{V_{Tx}^2 + V_{Ty}^2}}\right) \tag{b.3}$$

$$\chi_d = \tan^{-1}\left(V_{Ty} / V_{Tx}\right) \tag{b.4}$$

B1.3 Dynamic Inversion Control

Commanded thrust T_c is formulated based on the sum of air drag D, projected weight along longitudinal axis, and the gap between desired and actual velocity as formulated in:

$$T_c = D + \omega_v . W . \frac{\left(V_d - V\right)}{g} + W . \sin\gamma \tag{b.5}$$

Since aircraft cannot reverse, commanded thrust T_c must be a positive value and it is limited to maximum thrust T_{max}.

On the other hand, commanded load factor n_c is derived from the gap between desired and actual pitching, and the gap between desired and actual yawing. The upper and lower limits of commanded load factor n_c are defined based on lifting coefficient Cl_{min} and Cl_{max}. The equations are constructed as follows.

$$n_c . \cos\varphi_c = D + \omega_\gamma . V . \frac{\left(\gamma_d - \gamma\right)}{g} + \cos\gamma = a \tag{b.6}$$

$$n_c . \sin\varphi_c = \omega_\chi . V . \frac{\left(\chi_d - \chi\right)}{g} . \cos\gamma = b \tag{b.7}$$

$$n_c^2 = a^2 + b^2 \tag{b.8}$$

$$n_{stall\_min} = \frac{\rho . V^2 . S . Cl_{min}}{2 . W} \tag{b.9}$$

$$n_{stall\_max} = \frac{\rho . V^2 . S . Cl_{max}}{2 . W} \tag{b.10}$$

Lastly, commanded bank angle ϕ_c is simply calculated from the inverse tangent of lateral to longitudinal commanded lift factor n_c as described in Equation b.11.

$$\varphi_c = \tan^{-1}\left(b / a\right) \tag{b.11}$$

B1.4 Aerodynamic

The value of 9 actual states: V, T, n, γ, χ, ϕ, x, y, and z are established based on the integration of each first derivative described from equation b.12 to b.22. Every mathematical integration of the first derivative is according to its defined initial condition. For example, the current x position is calculated from the integration of its first derivative dx/dt with initial condition of $x = 0\ m$.

Acceleration dV/dt is determined from the excess thrust, which is the gap between commanded thrust, and air drag and weight of aircraft respect to inclination.

$$dV/dT = g \times \left\{ \frac{(T_c - D)}{W} - \sin\gamma \right\} \qquad (b.12)$$

, where total drag force D is expressed as the sum of surface and polar drags respectively.

$$D = \frac{\rho.V^2.S.C_{do}}{2} + \frac{2.k.n^2.W^2}{\rho.V^2.S} \qquad (b.13)$$

However, when flaps are deployed during landing, the equation of drag force D becomes:

$$D = \tfrac{1}{2}.\rho.V^2.S\left(C_{do} + k.C_{l_{flaps}}{}^2 \right) \qquad (b.14)$$

, where lifting coefficient of flaps is set to 2.2.

Pitching rate $d\gamma/dt$ and yawing rate $d\chi/dt$ are derived from the followings.

$$d\gamma/dt = \left(g/V \right) \times \left\{ n_c.\cos\varphi_c - \cos\gamma \right\} \qquad (b.15)$$

During takeoff and landing, pitching angle is only limited to ± 1° due to ground restriction.

$$d\chi/dt = \frac{g.n_c.\sin\varphi_c}{V.\cos\gamma} \qquad (b.16)$$

The rate of thrust dT/dt, bank $d\phi/dt$, and load factor dn/dt are proportional to the gap between actual and commanded values with defined time constant respectively.

$$dT/dt = \frac{(T_C - T)}{\tau_T} \qquad (b.17)$$

$$d\varphi/dt = \frac{(\varphi_C - \varphi)}{\tau_\varphi} \qquad (b.18)$$

$$dn/dt = \frac{(n_C - n)}{\tau_n} \qquad (b.19)$$

Lastly, the speed elements in x, y, and z direction are expressed as:

$$dx/dt = V.\cos\gamma.\cos\chi \qquad (b.20)$$

$$dy/dt = V.\cos\gamma.\sin\chi \qquad (b.21)$$

$$dz/dt = V.\sin\gamma \qquad (b.22)$$

B1.5 Propulsion

Maximum Engine Power P_{max_engine} is formulated as:

$$P_{max\_engine} = V_{max\_motor} \times I_{max\_motor} \times \eta_{engine} \times \eta_{propeller} \qquad (b.23)$$

, where based on *section 2.2.1 Propulsion*, engine efficiency η_{engine} is 84%, and propeller efficiency is assumed to be 60% according to Kotwani et al (2004).

This value is then used to derive máximum thrust T_{max} as follows.

$$T_{\max} = \frac{P_{\max\_engine}}{V} \qquad (b.24)$$

To measure battery drain rate, power consumption P is determined as:

$$P = \frac{T \times V}{\eta_{engine} \times \eta_{propeller}} \qquad (b.25)$$

Lastly, battery capacity in Watt-hour is formulated as follows.

$$Capacity = Battery\_no \times V_{\max\_battery} \times I_{\max\_battery} \times 3600 \qquad (b.26)$$

The last two variables are then implemented in *RSuav* module to determine the battery life time.

B2. *RRCUAV* Parameters

Table b.1 lists *RRCUAV* parameters involved in the calculation of *RSuav* module.

Table b.1: *RRCUAV* Parameters

Parameter	Symbol	Value	Unit
Weight	W	88.29	Newton
surface area	S	0.45	m^2
max.lift coeff	Cl_{max}	1.40	
min.lift coeff	Cl_{min}	-0.50	
induced drag coeff	k	0.07	
parasite drag coeff	C_{do}	0.04	
thrust time constant	τ_T	1.00	Sec
bank angle constant	τ_ϕ	0.50	Sec
load factor time constant	τ_n	0.50	Sec
speed bandwidth	ω_v	0.85	Sec
pitching bandwidth	ω_γ	1.00	Sec
yawing bandwidth	ω_χ	Depending on the max. allowable bank angle ensuring stability, e.g. ω_χ=7.0 representing max. ϕ=48.2°	Sec
max. input voltage of motor	$V_{max\ motor}$	33.30	Volt
max. current of motor	$I_{max\ motor}$	43.60	Ampere
engine efficiency, e.g. heat loss, friction	η_{engine}	0.84	
conservative assumption of propeller efficiency	$\eta_{propeller}$	0.60	
max. input voltage of battery	$V_{max\ battery}$	18.50	Volt
max. current of battery	$I_{max\ battery}$	8.00	Ampere
battery number	$Battery\_no$	3.00	

B3. COESA Atmosphere Models

The troposphere is located between earth surface and altitude of 11.02 km. Within this region, the temperature T_h decreases linearly and the pressure P_r decreases exponentially. For the temperature T_h and the pressure P_r, the international units curve fits for the troposphere are:

$$T_h = 15 - 0.00649 \times z \qquad (b.27)$$

$$P_r = 101314.08 \times \left\{ \frac{1.8 \times T_h + 491.7}{518.6} \right\}^{5.256} \qquad (b.28)$$

, where the temperature T_h is given in Celsius degrees, pressure P_r in Pa, and z is the altitude in meter. In each zone air density ρ (in kg/m^3) is derived from the equation of state:

$$\rho = 0.006 \times \frac{P_r}{1.8 \times T_h + 491.7} \qquad (b.29)$$

Based on Equation b.27 to b.29, the profile of air density over altitude is shown in Figure b.4.

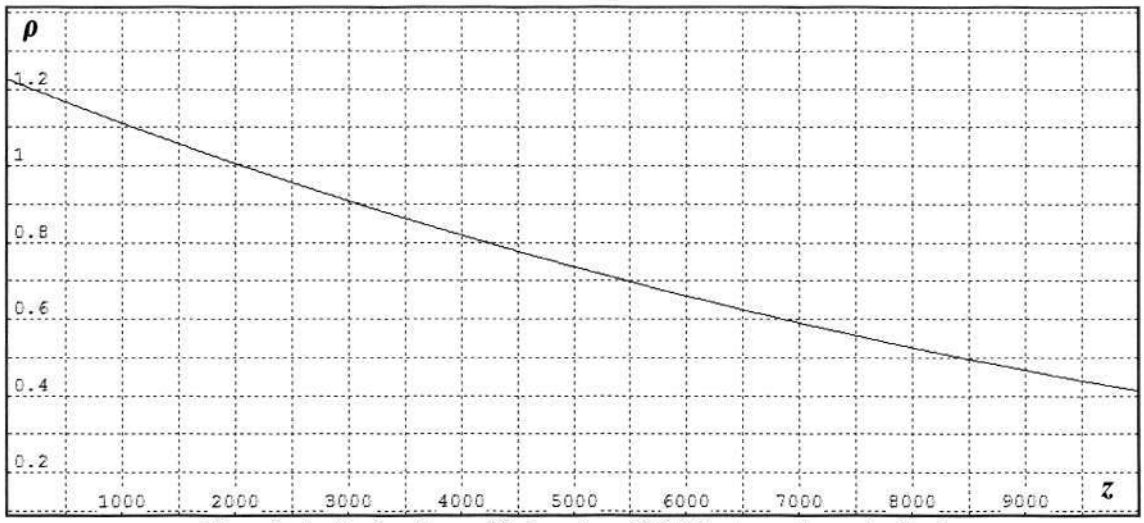

Figure b.4: Air density profile based on COESA atmosphere standard

References: Appendix B

1. Kotwani, K., Sane, S.K., Arya, H., Sudhakar, K., December 16-18, 2004, *Experimental Characterization of Propulsion System for Mini Aerial Vehicle*, 31st National Conference on FMFP, Jadavpur University, Kolkata.
2. Robbins D., Anderson M., 1998, *Formation Flight as a Cooperative Game*, AIAA-98-4124, AIAA GNC

Appendix C. Pseudo-codes of *RRCUAV* Autonomous Behaviors

The following pseudo-code describes decision making process in the Intelligent System of *RSuav* module.

C1. Automatic Takeoff

The pseudo-code of automatic takeoff described in the following explains that *RRCUAV* needs to accelerate into 16.6 m/s (10% margin to stall speed) to perform takeoff. As it reaches stall speed, it carries out go-to-waypoint of $x = 100\ m$, $y = 0\ m$, $z = 30\ m$ and thus, spirally ascends to defined coordinate. If the gap distance between *RRCUAV* and defined point is smaller than its turning radius and about to increase, it means *RRCUAV* has just reached its closest position to defined waypoint. Therefore, it then proceeds to cruising mode

Name: Automatic Takeoff
Goal: To make *RRCUAV* moving in spiral at 30 m altitude

```
1: set weight_uav := 88.29
2: set air_density :=1.225
3: set reference_area := 0.45
4: set Clmax := 1.4
5: set speed_stall := sqrt(weight_uav/(0.5×air_density×reference_area×Cl_max))
6: set instructed_speed_t_ta := 16.6
7: if speed_uav ≥ stall_speed then
8:         set wp_x := 100
9:         set wp_y := 0
10:        set wp_z := 30
11:        set turn_radius := speed_uav^2/g/tan(bank_uav)
12:        set gap_x := wp_x - x_uav
13:        set gap_y := wp_y - y_uav
14:        set gap_z := wp_z - z_uav
15:        set gap_distance_old := set gap_distance_new
16:        set gap_distance := sqrt(gap_x^2+ gap_y^2+ gap_z^2)
17:        set gap_distance_new := gap_distance
18:        if gap_distance_new ≤ turn_radius and gap_distance_old < gap_distance_new then
19:                Call Go_to-waypoint
20:        endif
21:        set instructed_velocity_x_ta := gap_x/gap_distance
22:        set instructed_velocity_y_ta := gap_y/gap_distance
23:        set instructed_velocity_z_ta := gap_z/gap_distance
24: endif
```

C2. Go-to-waypoints

As RRCUAV steps into cruising mode, number and coordinate of waypoints can be defined either from operator through GUI or automatically generated from WPP. Waypoint scheduler selects current waypoint *i RRCUAV* heading on. As it passes through the last waypoint, it goes back to waypoint *i=1*.

Intended direction $\vec{V}_T$ ' is described as the unit vector of *gap_distance* in x, y, and z coordinate and later defined as the velocity vector of target acquisition *instructed_velocity_ta*. To minimize power consumption, instructed speed is set to be 16.6 (10% margin to stall speed).

```
Name: Go-to-waypoints
Goal: To go-to a number of defined waypoints in closed loop trajectory based on defined
      sequence
```

```
1: set g := 9.81
2: set no_wp := no_wp_operator_or_intelligent_planner_defined
3: set wp_x := wp_x_operator_or_intelligent_planner_defined
4: set wp_y := wp_y_operator_or_intelligent_planner_defined
5: set wp_z := wp_z_operator_or_intelligent_planner_defined
6: set turn_radius := speed_uav^2/g/tan(bank_uav)
7: set gap_x := wp_x[i]- x_uav
8: set gap_y := wp_y[i] - y_uav
9: set gap_z := wp_z[i] - z_uav
10: set gap_distance_old := set gap_distance_new
11: set gap_distance := sqrt(gap_x^2+ gap_y^2+ gap_z^2)
12: set gap_distance_new := gap_distance
13: if gap_distance_new ≤ turn_radius and gap_distance_old < gap_distance_new then
14:       i++
15:       if i > no_wp then
16:       set i := 1
17:       endif
18: endif
19: set instructed_velocity_x_ta := gap_x/gap_distance
20: set instructed_velocity_y_ta := gap_y/gap_distance
21: set instructed_velocity_z_ta := gap_z/gap_distance
22: set instructed_speed_t_ta := 16.6
```

C3. Instinctive Behavior on Obstacle Avoidance

Obstacle avoidance is executed based on *if-then* condition as described in the following pseudocode. If measured distance is within defined *threshold*, then obstacle avoidance is activated by having *scale_factor_ca_obstacle* set into 1.0. The direction of turn is then selected by adding *±PI/2* to *current_yaw_angle* based on which *measured_distance_of_right_sensor* and *measured_distance_of_left_sensor* is shorter. The escape maneuver is executed at any speed ranged from stall to maximum speed. In this case, 22.5 m/s is selected.

```
Name: Obstacle Avoidance
Goal: To instinctively avoid any nearby static obstacle
```

```
1: set max_range :=365.76
2: set threshold := 50
3: if measured_distance_of_right_sensor < thershold or measured_distance_of_left_sensor < thershold then
4:       set scale_factor_ca_obstacle := 1
5: else
6:       set scale_factor_ca_obstacle := 0
7: end if
8:       set scale_factor_ca := scale_factor_ca_obstacle
9:if measured_distance_of_right_sensor < measured_distance_of_left_sensor then
10:      set instructed_velocity_x_t_ca := cos(current_yaw_angle+ PI/2)
11:      set instructed_velocity_y_t_ca := sin(current_yaw_angle + PI/2)
12:      set instructed_velocity_z_t_ca := 0
13: end if
14:if measured_distance_of_right_sensor > measured_distance_of_left_sensor then
15:      set instructed_velocity_x_t_ca := cos(current_yaw_angle - PI/2)
16:      set instructed_velocity_y_t_ca := sin(current_yaw_angle - PI/2)
17:      set instructed_velocity_z_t_ca := 0
18: end if
19:set instructed_speed_t_ca := 22.5
```

Automatic takeoff, go-to-waypoint, and automatic landing are developed from target acquisition instinct. On the other hand, obstacle avoidance is based on collision avoidance instinct. Since target

acquisition is less important than collision avoidance, its scale factor is defined as the remaining portion of total scale factor or *scale_factor_ta = 1-scale_factor_ca*. Intended direction $\vec{V}_T$ ' is then formulated as a linear combination of these two instinctive behaviors. Aircraft speed will be defined based on the behavior, which has bigger scale factor.

Name: Instructed Velocity Goal: To set Instructed Velocity as the combination of Collision Avoidance and Target Acquisition velocity vector
1: set scale_factor_ta := 1-scale_factor_ca 2: set instructed_velocity_x_t := scale_factor_ca×instructed_velocity_x_t_ca+ scale_factor_ta×instructed_velocity_x_t_ta 3: set instructed_velocity_y_t := scale_factor_ca×instructed_velocity_y_t_ca+ scale_factor_ta×instructed_velocity_y_t_ta 4: set instructed_velocity_z_t := scale_factor_ca×instructed_velocity_z_t_ca+ scale_factor_ta×instructed_velocity_z_t_ta 5: if scale_factor_ca > scale_factor_ta then 6: set instructed_speed_t := instructed_speed_t_ca 7: else 8: set instructed_speed_t := instructed_speed_t_ta 9: end if

C4. Automatic Landing

The following pseudocode describes that automatic landing is basically the combination of go-to-waypoint in three different locations: (i) approaching point at *x=-200 m, y=130 m, and z=50 m*, (ii) spiral ascent/descent at *x=-200 m, y=130 m, and z=30 m*, and (iii) landing point. The only differences are that firstly, conditional element *abs(x_uav-(-200)) < 5 and y_uav < 130 and abs(z_uav-30) ≤0.5 and z_dot > -0.5* needs to be fulfilled during spiral ascent/descent to maintain the state of *RRCUAV* within particular region and vertical speed. Secondly, when landing is executed, flaps are deployed by setting *Clmax=2.2*. This then reduces stall speed *speed_stall* as well as forward speed by increasing drag defined as $Drag = 0.5*air\_density*reference\_area*speed\_uav^2*(Cdo+k*Clmax^2)$.

As it touches the ground or if conditional statement *z_uav ≤z_ground* is fulfilled then instructed velocity is set to positive x, *instructed_velocity_x_t_ta = 0.707* and negative z direction, *instructed_velocity_z_t_ta = -0.707* to kill lifting force and to create friction, which later slow the *RRCUAV* down.

Name: Automatic Landing Goal: To land at reasonable vertical speed, accuracy, and time
1: set Cdo := 0.04 2: set k := 0.07 3: set instructed_speed_t_ta := 16.6 4: set wp_x := -200 5: set wp_y := 130 6: set wp_z := 50 7: set turn_radius := speed_uav^2/g/tan(bank_uav) 8: set gap_x := wp_x - x_uav 9: set gap_y := wp_y - y_uav 10: set gap_z := wp_z - z_uav 11: set gap_distance_old := set gap_distance_new 12: set gap_distance := sqrt(gap_x^2+ gap_y^2+ gap_z^2) 13: set gap_distance_new := gap_distance 14: if gap_distance_new ≤ turn_radius and gap_distance_old < gap_distance_new then 15: Call Spiral_Descent_Ascent 16: endif

```
17: Spiral_Descent_Ascent:
18: set wp_x := -200
19: set wp_y := 130
20: set wp_z := 30
21: set gap_x := wp_x - x_uav
22: set gap_y := wp_y - y_uav
23: set gap_z := wp_z - z_uav
24: set gap_distance_old := set gap_distance_new
25: set gap_distance := sqrt(gap_x^2+ gap_y^2+ gap_z^2)
26: set gap_distance_new := gap_distance
27: if abs(x_uav-(-200)) < 5 and y_uav < 130 and abs(z_uav-30) ≤ 0.5 and z_dot > -0.5 then
28:         Call Landing
29: endif
30: Landing:
31: set Clmax := 2.2
32: set instructed_speed_t_ta :=1.1* speed_stall;
33: set Drag := 0.5*air_density*reference_area* speed_uav^2*(Cdo+ k* Cl^2)
34: set gap_x := 0
35: set gap_y := 0
36: set gap_z := -1
37: if z_uav ≤ z_ground then
38:         Call Braking
39: endif
40: set instructed_velocity_x_ta := gap_x/gap_distance
41: set instructed_velocity_y_ta := gap_y/gap_distance
42: set instructed_velocity_z_ta := gap_z/gap_distance
43: Braking:
44: set instructed_speed_t_ta := 0
45: set instructed_velocity_x_t_ta := 0.707
46: set instructed_velocity_y_t_ta := 0
47: set instructed_velocity_z_t_ta := -0.707
```

Lastly, emergency landing is executed if the conditional element expressed as *(max_distance_to_runway/speed_stall×power_consumption_auto_land×N)* ≥ *battery_capacity* is fulfilled. Maximum possible traveled distance is defined as *max_distance_to_runway = fabs((x_uav –(-200)) +fabs(y_uav -130)+fabs(z_uav -50)+200+130+50.*

Name: Conditional Statement of Emergency Automatic Landing
Goal: To activate emergency landing

```
1: set Vmax_battery :=18.5
2: set Imax_battery := 8
3: set battery_number := 3
4: set N := 1.5
5: set battery_capacity =battery_number×Vmax_battery×Imax_battery×3600
6: set power_consumption_auto_land := 500
7: set max_distance_to_runway := fabs((x_uav -(-200)) + fabs(y_uav -130)+ fabs(z_uav -50)+ 200+ 130+ 50
8: if (max_distance_to_runway/speed_stall×power_consumption_auto_land×N) ≥ battery_capacity then
9:         Call Auto_Landing
10: end if
```

Appendix D. Experimental Test and Result

This section is presented to describe the details of experimental procedures and its results.

D1. Automatic Landing

To measure the performance of automatic landing, 24 tests are conducted at different initial altitudes: 100, 300, and 500 m, and incoming angles: 0, ±45°, ±90°, ±135°, and 180°. The records of landing point in x and y coordinate are presented in Table d.1.

Table d.1: Landing Coordinate (meter)

Landing - X Coordinate				Landing - Y Coordinate			
Height /Angle	100	300	500	Height /Angle	100	300	500
0	99.20	97.70	101.70	0	34.30	34.40	34.20
45	100.80	100.80	100.20	45	34.20	34.30	39.30
90	102.30	102.20	100.70	90	34.10	34.30	34.50
135	97.90	97.90	101.70	135	34.20	34.20	34.20
180	97.50	97.70	103.00	180	34.20	34.30	34.20
-135	101.00	98.60	97.60	-135	34.50	34.10	34.20
-90	95.30	94.00	98.10	-90	34.20	34.20	34.40
-45	96.20	108.50	103.00	-45	34.50	32.30	34.50

The landing positions in Table d.2 are then averaged in Table d.2.

Table d.2: Landing Coordinate Summary (meter)

Landing Average	
X Coordinate	Y Coordinate
99.73	34.41

The accuracy is then calculated as the gap between landing positions in Table d.2 and the average values shown in Table d.3.

Table d.3: Landing Accuracy (meter)

Accuracy - X Coordinate				Accuracy - Y Coordinate			
Height /Angle	100	300	500	Height /Angle	100	300	500
0	0.53	2.03	1.97	0	0.11	0.01	0.21
45	1.07	1.07	0.47	45	0.21	0.11	4.89
90	2.57	2.47	0.97	90	0.31	0.11	0.09
135	1.83	1.83	1.97	135	0.21	0.21	0.21
180	2.23	2.03	3.27	180	0.21	0.11	0.21
-135	1.27	1.13	2.13	-135	0.09	0.31	0.21
-90	4.43	5.73	1.63	-90	0.21	0.21	0.01
-45	3.53	8.77	3.27	-45	0.09	2.11	0.09

The accuracy is then summarized into Table d.4.

Table d.4: Landing Accuracy Summary (meter)

Accuracy - Average		Accuracy - Std. Dev	
X Coordinate	Y Coordinate	X Coordinate	Y Coordinate
2.43	0.44	1.82	1.03

In terms of vertical speed, all automatic landing tests result in -2.0 m/s on touching down (see Table d.5) and thus, standard deviation is zero as shown in Table d.6.

Table d.5. Vertical Speed (m/s)

Vertical Speed			
Height /Angle	100	300	500
0	-2.00	-2.00	-2.00
45	-2.00	-2.00	-2.00
90	-2.00	-2.00	-2.00
135	-2.00	-2.00	-2.00
180	-2.00	-2.00	-2.00
-135	-2.00	-2.00	-2.00
-90	-2.00	-2.00	-2.00
-45	-2.00	-2.00	-2.00

Table d.6. Vertical Speed Summary (m/s)

Vertical Speed	
Average	Std. Dev
-2.00	0.00

Landing times are shown in Table d.7 and they are then summarized into Table d.8.

Table d.7. Landing Time (minute)

Landing Time			
Height /Angle	100	300	500
0	2.65	2.92	2.49
45	2.58	2.54	2.05
90	2.51	2.47	2.56
135	3.01	2.97	2.49
180	2.94	2.90	2.42
-135	3.91	2.44	2.91
-90	3.34	1.65	2.96
-45	3.40	2.26	2.44

Table d.8. Landing Time Summary (minute)

Landing Time	
Average	Std. Dev
2.70	0.46

Since *RRCUAV* bounces once on landing, it spends landing distance, which is one a half longer than its takeoff distance. The data of landing distance and its summary are shown in Table d.9 and d.10.

Table d.9. Landing Distance (meter)

Landing Distance			
Height /Angle	100	300	500
0	52.00	52.00	52.00
45	52.10	52.00	52.10
90	51.80	52.10	52.10
135	52.00	52.00	52.00
180	52.00	52.10	52.00
-135	52.00	52.10	52.00
-90	52.00	52.00	52.10
-45	52.10	52.00	52.10

Table d.10. Landing Distance Summary (meter)

Landing Distance	
Average	Std. Dev
52.03	0.07

D2. Obstacle Avoidance

Factors affecting the performance of obstacle avoidance are listed in this experiment. The motivation of choosing the values of each parameter is explained. The procedure of experiment is then described and it is followed by the result and simple guide on how to use DTREG software to analyze experimental data.

D2.1. Experimental Tests I. Finding the Significance of Parameters

Tuning Parameters

- Environment

Based on the general characteristic of Singapore business district and residential area, the virtual environments are modeled based on the specification given in Table d.11.

Table d.11. The Specification of Synthetic Urban Area

Category	Business District	Residential Area
Avg. Build. Length (m)	40	80
Avg. Build. Width (m)	35	19
Avg. Build. Sep. Dist. (m)	35	19
Avg. Precinct Sep. Dist. (m)	35	37
Range of Build. Length (m)	30-50	28-140
Range of Build. Width (m)	9-40	9-47
Range of Build. Sep. Dist. (m)	10-50	9-75
Range of Precinct Sep. Dist. (m)	10-50	18-70
Avg. Build. Height (m)	50	150
Range of Build. Height (m)	40-70	100-200
Avg. Road Width (m)	10	10
Diameter of Area (km)	1.6	1.6

Figure d.1 shows the top view of synthetic business district and residential area for experimental test. The buildings in business district are distributed uniformly, while the ones in residential area are patterned perpendicular or parallel one another. Both areas are 1.6 km in diameter with the average road width of 10 m. By having an area of 1.6 km diameter, hopefully the chance of the *RRCUAV* flying among tall buildings is bigger. If it is too small, obstacle avoidance might direct the *RRCUAV* to move around the circular area instead of passing through it.

Figure d.1: Synthetic Business District (Left) and Residential Area (Right)

❑ Maneuverability

The *RRCUAV* maneuver is produced from the combination of aircraft speed and bank angle. Based on the study of go-to-waypoint stability in *section 5.1.2. Go-to-Waypoints*, bank angle must be limited. Aircraft speed is then divided into three:

- The first one is near to stall speed, 16.1 m/s with maximum bank angle of 16.5°and it results in maximum load factor n_{max} of 1.04 and steady state turning radius of 90 m.
- The second one is moderate speed of 22.7 m/s with maximum bank angle of 48.2° and it results in maximum load factor n_{max} of 1.5 and steady state turning radius of 45 m.
- Lastly, the most maneuverable *RRCUAV* is set to be 30.3 m/s with maximum bank angle of 65.9°and it results in maximum load factor n_{max} of 2.45 and steady state turning radius of 40 m.

❑ Sensor Configuration

Figure d.2 illustrates the variation of sensor configurations in this experimental test. The opening laser angles θ are divided into three: nearly parallel – 10°, symmetric – 45°, and nearly perpendicular – 80° to body axis.

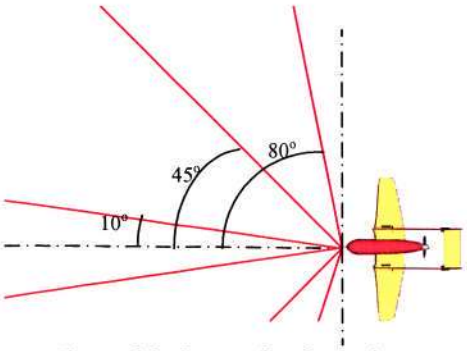

Figure d.2: Sensor Configurations

❑ Sensor Threshold

Figure d.3 illustrates defined threshold, which is represented as blue dots along the laser beam. As any objects detected within this dot, the *RRCUAV* will execute obstacle avoidance. There are five values of sensor threshold to be tested. They are 20, 40, 60, 80, and 100 m. Although maximum range of laser reaches up to 365.8 m, sensor threshold to be tested is limited to 100 m only. This is because for both business district and residential area, building width is ranged between 9 and 47 m only. It seems that the effectiveness of obstacle avoidance in both areas saturates at 100 m threshold. With minimum turning radius of 25.2 m, the minimum sensor threshold is decided to be 20 m.

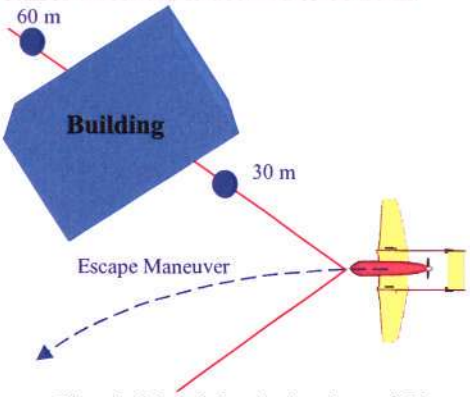

Figure d.3: Sensor Threshold and the Activation of Obstacle Avoidance

❑ Sensor Loop Rate

Sensor loop rate is defined as the frequency of information update on measured distance. This information comes from laser rangefinder to the autopilot and has been modeled with defined accuracy and resolution. The rate to be tested is divided into: 3 Hz, 6 Hz, and 10 Hz, which is the maximum sensor loop rate.

Procedural Tests

As illustrated in Figure d.4, the experiment is conducted by alternating five parameters mentioned above. In each combination of five parameters, *RRCUAV* flies over 8 different planned trajectories: *ae*, *bf*, *cg*, *dh*, *ea*, *fb*, *gc*, and *hd* in each area as shown in Figure d.5. The first letter describes the initial point and the last one is the intended destination.

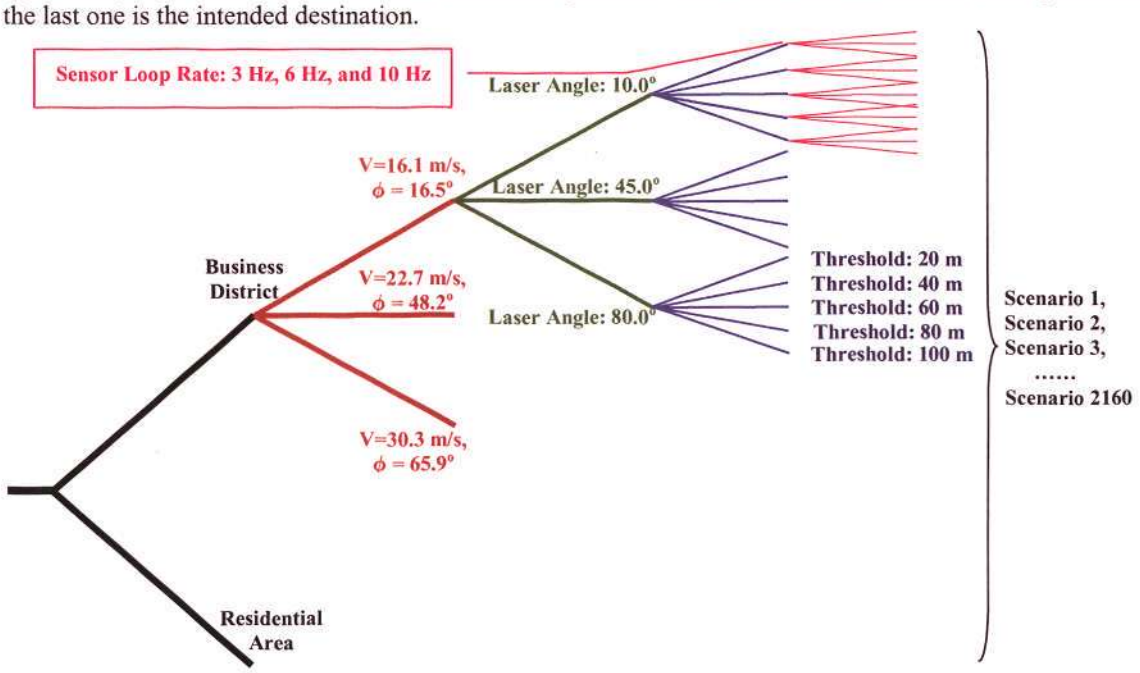

Figure d.4: Procedure of Experiment I

Therefore the total number of experimental test becomes 2160, which is the product of:

$$\begin{aligned} \text{No\_Test} &= \text{No\_Environ} \times \text{No\_Manuever} \times \text{No\_S\_Config} \times \text{No\_S\_Thers} \times \text{No\_S\_LoopRate} \times \text{No\_Traject} \\ &= 2 \times 3 \times 3 \times 5 \times 3 \times 8 \\ &= 2160 \text{ Scenarios} \end{aligned}$$

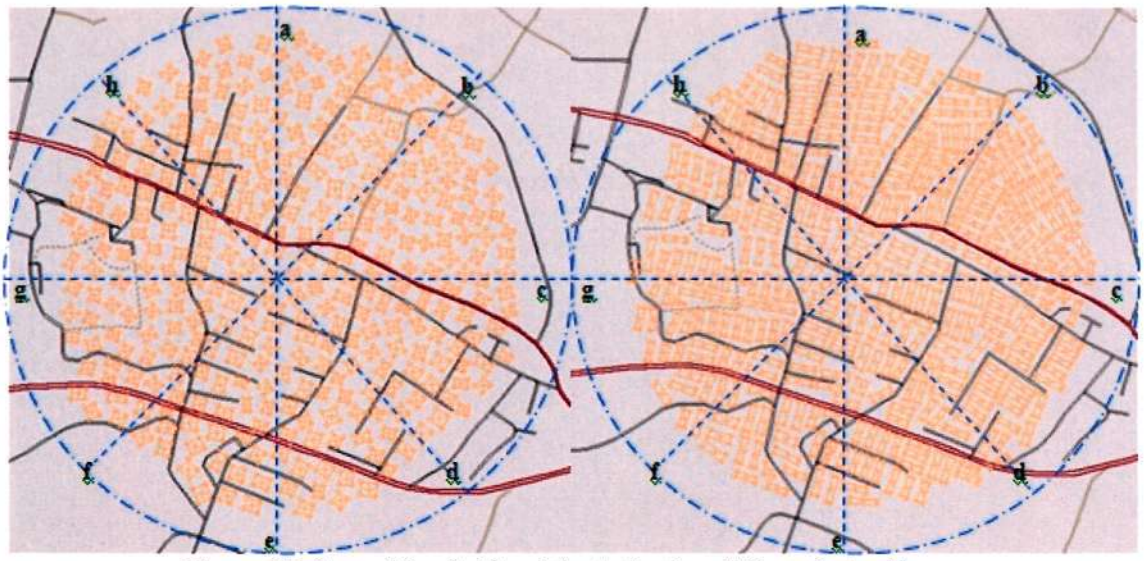

Figure d.5: Sensor Threshold and the Activation of Obstacle Avoidance

If the ratio between *RRCUAV*'s traveled distance and area diameter exceeds $\pi/2$, it is more likely that *RRCUAV* moves along the circumference of area. Any scenarios fall under this condition becomes invalid and thus, they will be rejected for analysis.

Results

Before discussing the result of experiment 1, Figure d.6 illustrates the chronology of *RRCUAV* obstacle avoidance in urban area, where multiple obstacles present in merely uniform distribution. Green and red buildings are the ones, which are more than 50 m high and considered as obstacle in *RRCUAV* operation.

There are two scenario examples shown here. The green trajectory performed by the *RRCUAV*, which is operated among green buildings with average separation distance of 75 m. Secondly, the red trajectory is operated among green *and* red buildings with average separation distance of 30 m.

Point 1, 2,… or i, ii,… shows the *RRCUAV* position, where Laser rangefinder detects the presence of building (1), (2),… or (i), (ii),… respectively at threshold of 365.76 m (maximum range). This is also the place, where *RRCUAV* executes obstacle avoidance. As shown in Figure d.6, green trajectory is more linear than the red one since it performs less number of turns as the consequence of lower urban density.

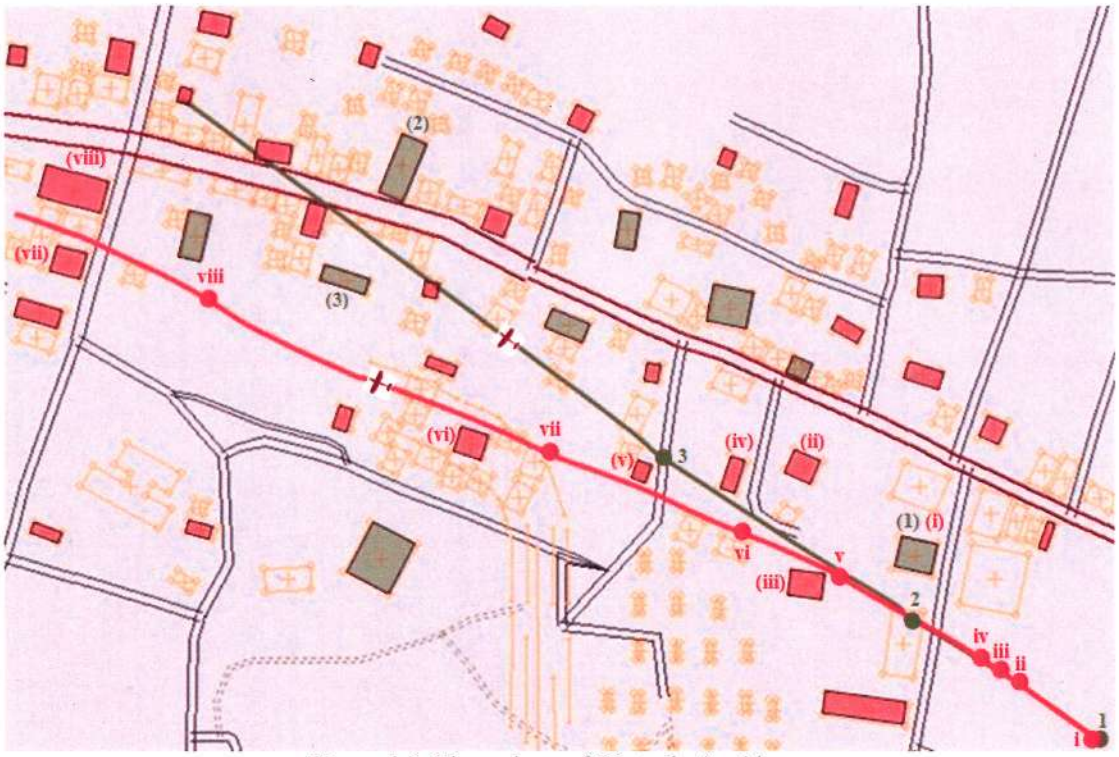

Figure d.6: Chronology of Obstacle Avoidance

The complete list of experimental result can be seen at the CD attached in this report. The sample flight trajectories of various scenarios in this experiment are shown from Figure d.7 to Figure d.9.

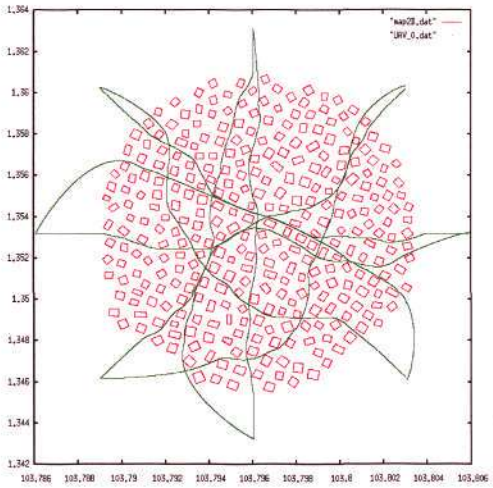

Figure d.7: Scenario No. 105 to 112, Business District, Speed=16.1m/s, Laser Angle=10°, Threshold=100 m, Loop Rate=6 Hz

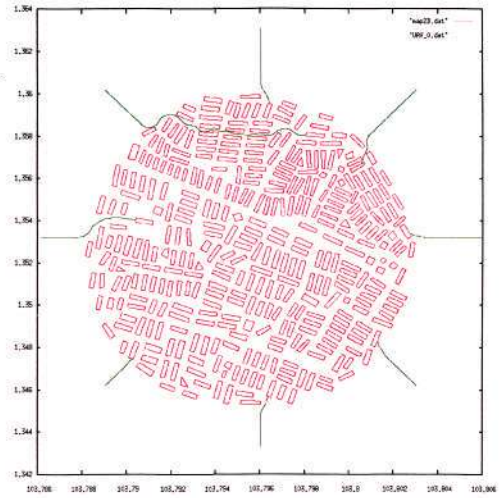

Figure d.8: Scenario No. 1625 to 1632, Residential Area, Speed=22.7m/s, Laser Angle=45°, Threshold=60 m, Loop Rate=10 Hz

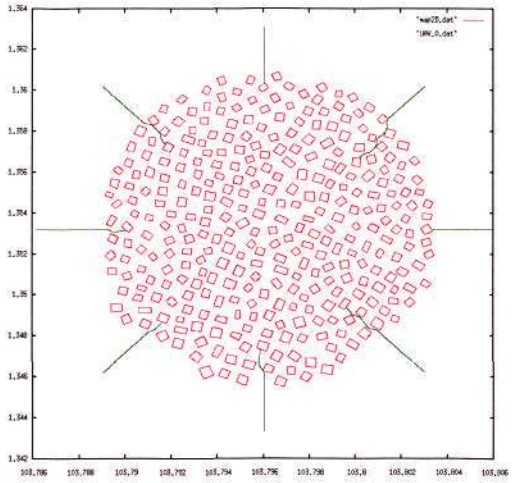

Figure d.9: Scenario No. 2065 to 2072, Business District, Speed=30.3m/s, Laser Angle=80°, Threshold=40 m, Loop Rate=3Hz

D2.2. Experimental Test II. Determining the Robustness

The tuning parameters for second experiment are listed and three scenarios from the first experiment are chosen and then tested in second experiment. After that, the procedure of experiment is described. Lastly, the result of experiment II is shown.

Tuning Parameters

- Environment

To determine the robustness of *RRCUAV*, more areas are required. Besides building width and separation distance determining the characters of area (business district or residential area), area diameter is also believed to affect the robustness of *RRCUAV*.

Therefore, total of 30 areas are established as shown from Figure d.10 to Figure d.15. The areas are built in random configuration and created based on the characteristic shown in Table d.11 earlier.

Figure d.10: Area 1 to 5, Business Districts, and Area Diameter: 1600 m

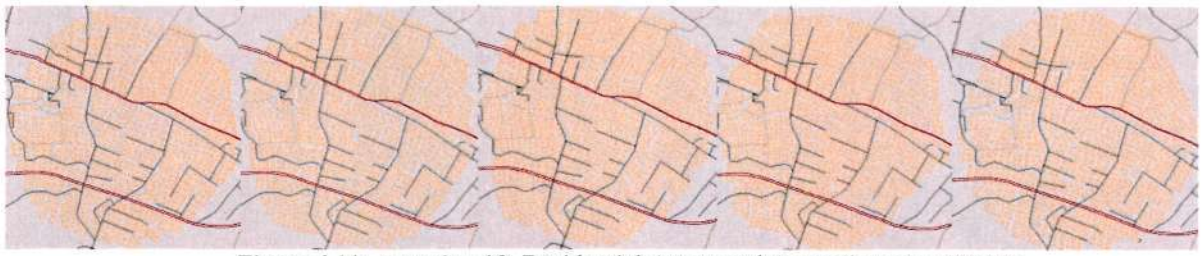

Figure d.11: Area 6 to 10, Residential Areas, and Area Diameter: 1600 m

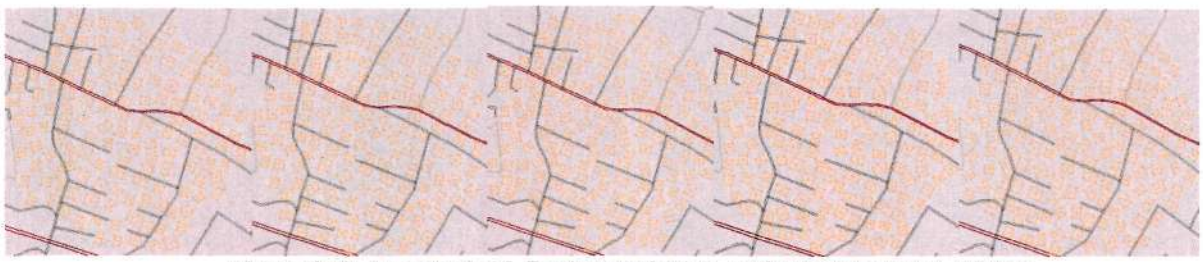

Figure d.12. Area 11 to 15, Business Districts, and Area Diameter: 1000 m

Figure d.13: Area 16 to 20, Residential Areas, and Area Diameter: 1000 m

Figure d.14: Area 21 to 25, Business Districts, and Area Diameter: 500 m

Figure d.15: Area 26 to 30, Residential Areas, and Area Diameter: 500 m

❑ *RRCUAV* Properties

To create the *RRCUAV* with optimum robustness, scenarios, which the *RRCUAV* survives during the tests (shown in Table d.12), need to be analyzed. All scenarios in Table d.12 have 10° laser angle. Based on robustness, the best three scenarios come from 22.7 m/s only.

Table d.12: Scenario with survived UAV

Laser Angle	Threshold	Maneuverability	Survived
10°	40 m	22.7 m/s	6
		30.3 m/s	11
	60 m	16.1 m/s	14
		22.7 m/s	30
		30.3 m/s	18
	80 m	16.1 m/s	28
		22.7 m/s	42
		30.3 m/s	9
	100 m	16.1 m/s	33
		22.7 m/s	38
		30.3 m/s	4
SUM			233

However, diversity of aircraft properties needs to be created in second experiment. This is useful to see how speed and threshold might affect the robustness in business district and residential area. Therefore, three scenarios are then selected as follows.

- Laser Angle=10°, Threshold=80m, V=22.7m/s, Loop Rate=6Hz
- Laser Angle=10°, Threshold=100m, V=16.1 m/s, Loop Rate=6Hz
- Laser Angle=10°, Threshold=60m, V=30.3 m/s, Loop Rate=6Hz

Procedural Tests

The experiment is conducted by alternating environmental parameters and aircraft properties mentioned earlier. In each combination, the *RRCUAV* flies over 20 different planned trajectories, which are set up in similar way to first experiment.

Therefore the total number of experimental test becomes 1800, which is the product of:

$$\text{No\_Test} = \text{No\_Environ} \times \text{No\_Aircraft\_Properties} \times \text{No\_Traject}$$
$$= 30 \times 3 \times 20$$
$$= 1800 \text{ Scenarios}$$

If the ratio between *RRCUAV*'s traveled distance and area diameter exceeds $\pi/2$, it is more likely that *RRCUAV* moves along the circumference of area. Any scenarios fall under this condition becomes invalid and thus, they will be rejected for analysis.

Results

The complete list of experimental result can be seen at the CD attached in this report. The summary of result can be seen in Table d.13.

Table d.13: Result of Second Experiment

Area Diameter	Aircraft Properties			Robustness		
	Laser Angle	Threshold	Maneuverability	Total	Business	Resident
1600 m	10°	80 m	22.7 m/s	63.9%	81.0%	45.1%
		100 m	16.1 m/s	64.0%	97.0%	29.9%
		60 m	30.3 m/s	47.5%	83.0%	12.0%
1000 m	10°	80 m	22.7 m/s	84.5%	97.0%	69.1%
		100 m	16.1 m/s	76.0%	98.0%	52.2%
		60 m	30.3 m/s	60.0%	86.0%	34.0%
500 m	10°	80 m	22.7 m/s	88.0%	98.0%	76.5%
		100 m	16.1 m/s	80.1%	99.0%	59.1%
		60 m	30.3 m/s	80.4%	96.0%	63.8%

The sample flight trajectories of various scenarios in this experiment are shown from Figure d.16 to Figure d.18.

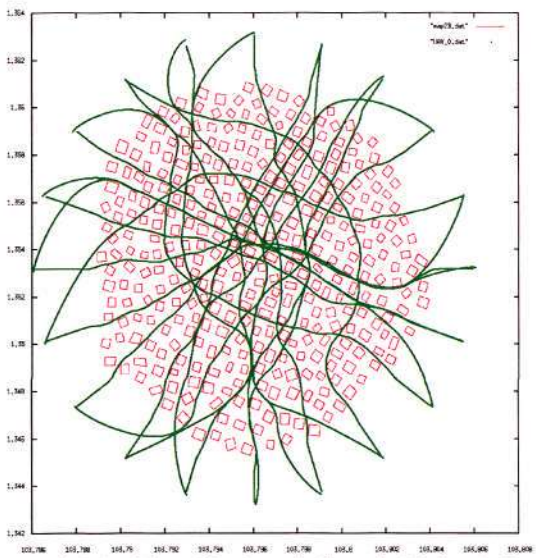

Figure d.16: Scenario No. 21 to 40, Area 1, Speed=16.1 m/s, Laser Angle=10°, Threshold=100 m, Loop Rate=6Hz

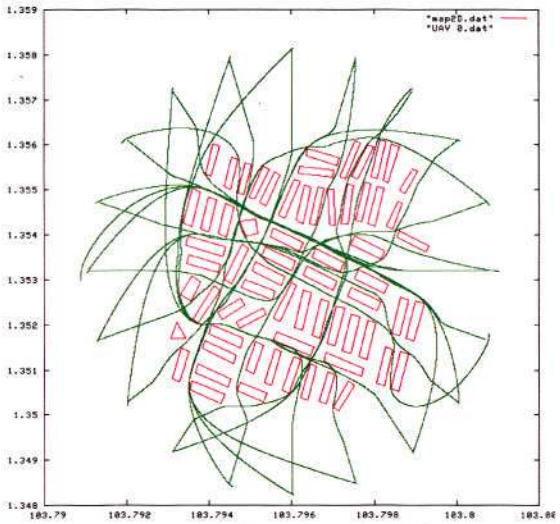

Figure d.17: Scenario No. 1741 to 1760, Area 30, Speed=22.7 m/s, Laser Angle=10°, Threshold=80 m, Loop Rate=6Hz

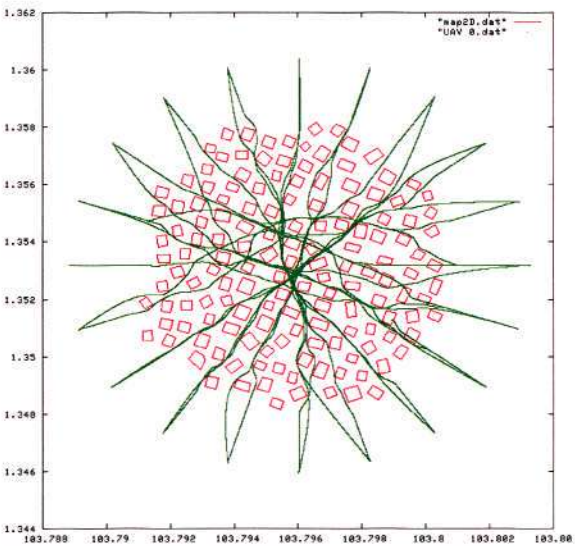

Figure d.18: Scenario No. 761 to780, Area 13, Speed=30.3 m/s, Laser Angle=10°, Threshold=60 m, Loop Rate=6Hz

D.2.3. Applying Experimental Data on DTREG Software

DTREG is used to analyze the result of experiment I to determine the significance of each parameter over obstacle avoidance performance. Demo version can be installed at URL: http://www.dtreg.com/DownloadDemo.htm . To create a new project, click button and a dialog box shown in Figure d.19 will appear.

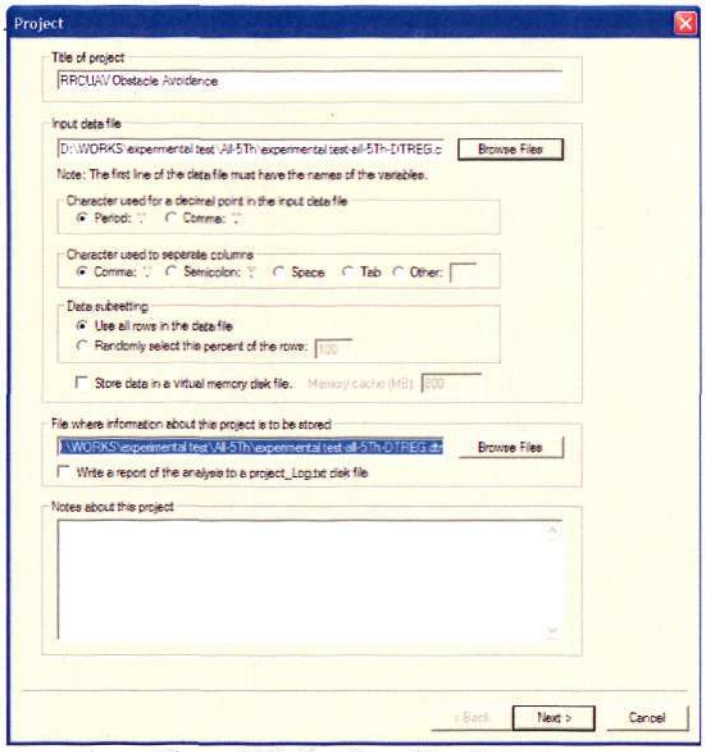

Figure d.19: Creating a New Project

After that, load experimental data, which is formatted in .csv and click Next > button. Dialog box shown in Figure d.20 will be then displayed.

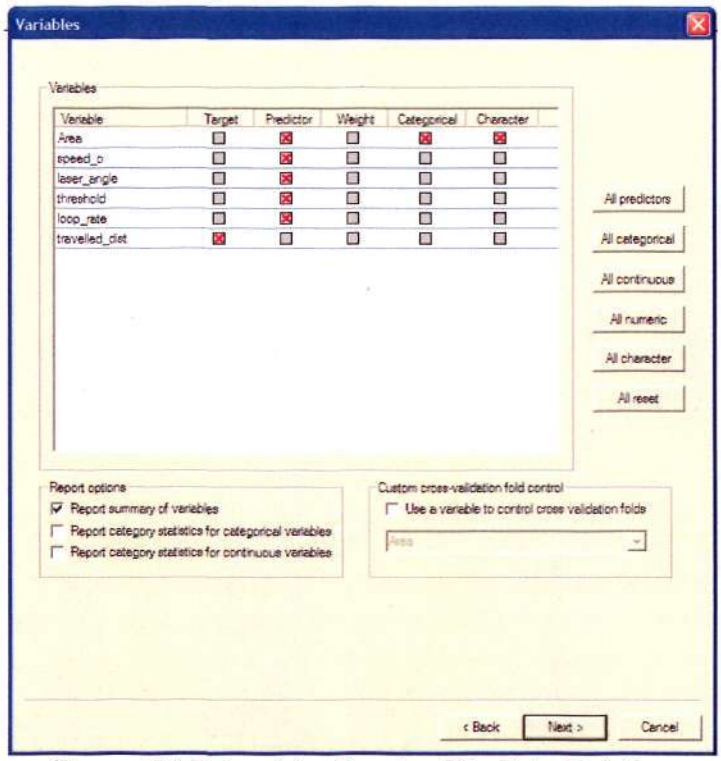

Figure d.20: Determining Target and Predictor Variables

Based on the data of experiment I, traveled distance is chosen as target variable and thus, 'Target' box is checked in dialog box shown in Figure d.20. The remaining parameters are checked in 'Predictor'

box category. Since area is the only categorical parameter (not continuous), 'Categorical' box is therefore checked.

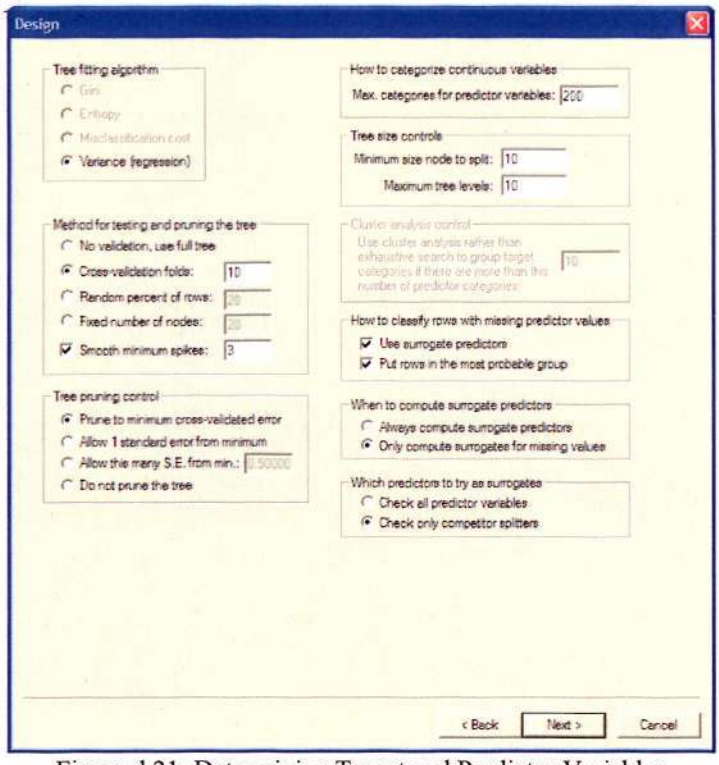

Figure d.21: Determining Target and Predictor Variables

The next dialogue box is shown in Figure d.21. These are the default settings to run the analysis of DT in DTREG.

After Next > button is clicked, click ⇨ button to run the analysis. The result of analysis is shown as follows.

```
Starting analysis at 18-Jul-2007 11:26:29
DTREG version 6.2 (Demonstration Version)
http://www.dtreg.com

============ Project Parameters ============

Project title: Decision Tree Analysis on RRCUAV Obstacle Avoidance
Project file: D: /WORKS/experimental test/Experiment I-DTREG.dtr
Target variable: travelled_dist
Number of predictor variables: 5
Type of model: Single tree
Maximum splitting levels: 10
Type of analysis: Regression
Splitting algorithm: Least squares
Variable weights: Equal
Minimum size node to split: 10
Max. categories for continuous predictors: 200
Use surrogate splitters for missing values: Yes
Always compute surrogate splitters: No
Tree pruning and validation method: Cross validation
Number of cross-validation folds: 10
Tree pruning criterion: Minimum cost complexity (0.00 S.E.)

============ Input Data ============

Input data file: D:/WORKS/experimental test/Experiment I -DTREG.csv
Number of variables (data columns): 6
Data subsetting: Use all data rows
Number of data rows: 1728
Total weight for all rows: 1728
Rows with missing target or weight values: 0
Rows with missing predictor values: 0

--- Statistics for target variable: travelled_dist ---
Mean value = 349.53822
Standard deviation = 572.26514
Minimum value = 0
Maximum value = 2773.63

============ Summary of Variables ============

  Variable       Class       Type          Missing rows     Categories
---------      -----       -----         ------------     ----------
Area           Predictor   Categorical          0               2
speed_o        Predictor   Continuous           0               3
laser_angle    Predictor   Continuous           0               3
threshold      Predictor   Continuous           0               4
loop_rate      Predictor   Continuous           0               3
travelled_dist Target      Continuous           0

============ Model Size Summary Report ============

Maximum depth of the tree = 10
Total number of group splits = 202
The full tree has 203 terminal (leaf) nodes.
The minimum validation relative error occurs with 14 nodes.
The relative error value is 0.2393 with a standard error of 0.0170
The tree will be pruned from 203 to 14 nodes.
```

------ Validation Statistics ------

Nodes	Val cost	Val std. err.	RS cost	Complexity	
69	0.2599	0.0192	0.2029	8.461496	
68	0.2596	0.0192	0.2029	8.839817	
66	0.2596	0.0192	0.2030	8.962410	
64	0.2595	0.0192	0.2030	9.027721	
63	0.2592	0.0192	0.2031	10.610552	
62	0.2591	0.0192	0.2031	10.739141	
61	0.2589	0.0192	0.2031	11.236565	
60	0.2585	0.0192	0.2032	13.466334	
59	0.2577	0.0191	0.2032	15.377075	
58	0.2577	0.0191	0.2033	16.527965	
57	0.2577	0.0191	0.2033	16.594683	
56	0.2575	0.0191	0.2034	16.947173	
55	0.2574	0.0191	0.2034	17.599413	
54	0.2573	0.0191	0.2035	18.085309	
53	0.2572	0.0191	0.2036	20.370776	
52	0.2571	0.0191	0.2036	22.039245	
51	0.2571	0.0192	0.2037	23.338894	
50	0.2571	0.0192	0.2038	26.691946	
49	0.2571	0.0192	0.2039	28.296467	
46	0.2571	0.0191	0.2041	28.791886	
45	0.2574	0.0192	0.2042	30.874728	
44	0.2560	0.0191	0.2043	40.929854	
43	0.2568	0.0192	0.2045	46.790549	
42	0.2568	0.0192	0.2046	50.633605	
41	0.2566	0.0191	0.2048	53.504266	
40	0.2564	0.0191	0.2050	55.969967	
39	0.2559	0.0191	0.2052	57.658817	
38	0.2554	0.0190	0.2054	65.204195	
36	0.2554	0.0190	0.2058	72.185721	
35	0.2558	0.0191	0.2060	72.796223	
34	0.2556	0.0190	0.2063	85.929519	
33	0.2558	0.0191	0.2065	86.148456	
32	0.2549	0.0191	0.2068	95.137446	
31	0.2546	0.0189	0.2072	116.728367	
30	0.2563	0.0191	0.2076	129.620913	
29	0.2564	0.0191	0.2080	138.626115	
28	0.2567	0.0191	0.2084	140.142989	
27	0.2566	0.0192	0.2089	145.327787	
26	0.2570	0.0192	0.2094	182.165327	
24	0.2570	0.0192	0.2106	182.913819	
23	0.2556	0.0189	0.2111	191.694601	
22	0.2535	0.0188	0.2118	232.785479	
21	0.2522	0.0187	0.2126	251.410203	
19	0.2521	0.0187	0.2142	259.768478	
18	0.2492	0.0184	0.2151	278.999257	
17	0.2476	0.0181	0.2160	310.764482	
16	0.2410	0.0173	0.2171	360.860911	
15	0.2419	0.0172	0.2190	624.292400	
14	0.2393	0.0170	0.2213	753.507355	<-- Min. validation error
13	0.2435	0.0174	0.2238	824.504691	
12	0.2476	0.0177	0.2271	1081.995436	
11	0.2474	0.0179	0.2306	1149.634966	
10	0.2441	0.0178	0.2348	1358.771718	
9	0.2474	0.0178	0.2413	2129.466883	
8	0.2633	0.0187	0.2572	5214.699316	
7	0.3206	0.0222	0.2799	7424.568037	
4	0.3713	0.0235	0.3624	9011.662656	
3	0.3958	0.0240	0.3934	10130.440456	

```
   2      0.6302      0.0180          0.6297  77379.380278
   1      1.0000      0.0000          1.0000 121282.109210


  ============ Analysis of Variance ============

Variance in initial data sample = 327487.39
Residual (unexplained) variance after tree fitting = 72476.832
Proportion of variance explained = 0.77869  (77.869%)

MSE (Mean Squared Error) = 72476.832
MAE (Mean Absolute Error) = 161.6986
MAPE (Mean Absolute Percentage Error) = 1045.4299


  ============ Lift and Gain ============

  --- Lift and Gain for training data ---

 Tree      Mean       Cum %     Cum %   Cum      % of      % of
 Node      Target     Population Target  Gain  Population Target  Lift
-----  --------- ---------- ------ ------ ------- ------ ----
   91  1800.7900    1.39       7.16   5.15   1.39    7.16   5.15
   77  1774.9210    6.94      35.37   5.09   5.56   28.21   5.08
   90  1358.4517    8.33      40.76   4.89   1.39    5.40   3.89
  135  1350.3652   10.19      47.92   4.70   1.85    7.15   3.86
   81  1181.4158   11.57      52.61   4.55   1.39    4.69   3.38
  124  1040.4158   12.96      56.75   4.38   1.39    4.13   2.98
  134   984.8932   13.89      59.36   4.27   0.93    2.61   2.82
   53   957.3992   16.67      66.96   4.02   2.78    7.61   2.74
   52   678.2869   19.44      72.35   3.72   2.78    5.39   1.94
  125   633.5406   20.83      74.87   3.59   1.39    2.52   1.81
   80   314.8613   22.22      76.12   3.43   1.39    1.25   0.90
   40   184.6554   25.00      77.59   3.10   2.78    1.47   0.53
    6   113.6978   33.33      80.30   2.41   8.33    2.71   0.33
    3   103.2842  100.00     100.00   1.00  66.67   19.70   0.30

  Average gain = 3.879
  Mean value of target variable = 349.53822


  ============ Overall Importance of Variables ============

 Variable     Importance
---------  ---------
laser_angle     100.000
threshold        78.522
speed_o          20.495
Area             10.565
loop_rate         0.680

Finished the analysis at 18-Jul-2007 11:26:31
Analysis run time:  00:02.52
```

After that, to view the decision tree diagram, click button and to see variable importance diagram, click Charts and then click 'Variable importance'. Figure 5.13 and 5.14 shown earlier are DT and variable importance diagram of the first experiment.